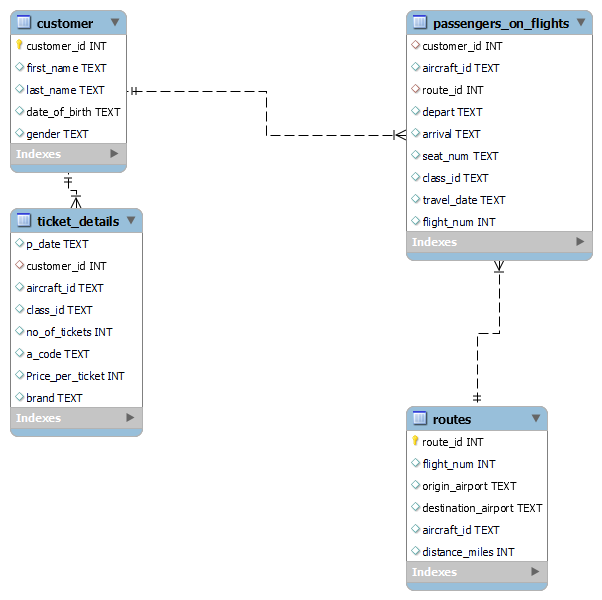
* **Create an ER diagram for the given airlines database.**



* **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.**

alter table routes

modify column flight\_num int check (flight\_num between 1000 and 2000),

modify column distance\_miles int check(distance\_miles >0);

alter table routes

modify column route\_id int unique;

* **Write a query to display all the passengers (customers) who have travelled in routes 01 to 25. Take data  from the passengers\_on\_flights table.**

select \* from passengers\_on\_flights where route\_id between 1 and 25;

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

select count(no\_of\_tickets) as no\_of\_passengers , sum(price\_per\_ticket) as total\_revenue from ticket\_details

where class\_id="bussiness";

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

select concat(first\_name," " ,last\_name) as full\_name from customer;

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

Select

customer.customer\_id, customer.first\_name,customer.last\_name,customer.date\_of\_birth

from customer

inner join ticket\_details on customer.customer\_id=ticket\_details.customer\_id;

* **Write a query to identify the customer’s first name and last name based on their customer ID and brand (Emirates) from the ticket\_details table.**

select customer.first\_name, customer.last\_name from customer

inner join ticket\_details on customer.customer\_id=ticket\_details.customer\_id;

select \* from ticket\_details;

* **Write a query to identify the customers who have travelled by *Economy Plus* class using**

**Group By and Having clause on the passengers\_on\_flights table.**

select customer\_id, class\_id, count(flight\_num) as no\_of\_flight

from passengers\_on\_flights where class\_id="economy plus" group by

customer\_id, class\_id;

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

select

if (sum(no\_of\_tickets\*price\_per\_ticket)>10000,"True","False")

from ticket\_details;

* **Write a query to find the maximum ticket price for each class using window functions on the ticket\_details table.**

select t.\*,

max(price\_per\_ticket) over(partition by class\_id) as max\_ticket\_price

from ticket\_details t;

* **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 route\_id1

on passengers\_on\_flights(route\_id);

select \* from where route\_id=4;

* **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;

* **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\*no\_of\_tickets) as

total\_price\_of\_all\_tickets from ticket\_details

group by aircraft\_id with rollup;

* **Write a query to create a view with only business class customers along**

**with the brand of airlines.**

create view business\_class\_view as

select brand, class\_id from ticket\_details

where class\_id="bussiness";

* **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.**

CREATE DEFINER=`root`@`localhost` PROCEDURE `distance`(dis int)

BEGIN

select \* from routes where distance\_miles>dis;

END;

call distance(2000);

* **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 DEFINER=`root`@`localhost` PROCEDURE `groups`()

BEGIN

select flight\_num, aircraft\_id, distance\_miles,

case

when distance\_miles>=0 and distance\_miles<=2000 then "short distance

travel(SDT)"

when distance\_miles>=2000 and distance\_miles<=6500 then "intermediate

distance travel(IDT)"

when distance\_miles>=6500 then "long distance travel(LDT)"

else "Not Correct"

end as distance\_group

from routes order by distance\_group desc;

call groups;

* **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.***

CREATE DEFINER=`root`@`localhost` PROCEDURE

`complimentary\_services`()

BEGIN

select p\_date, customer\_id, class\_id ,

case

when class\_id="bussiness" then "Yes"

When class\_id="economy plus" then "Yes "

else "No"

End As Complimentary\_Services

from ticket\_details;

END;

call complimentary\_services;