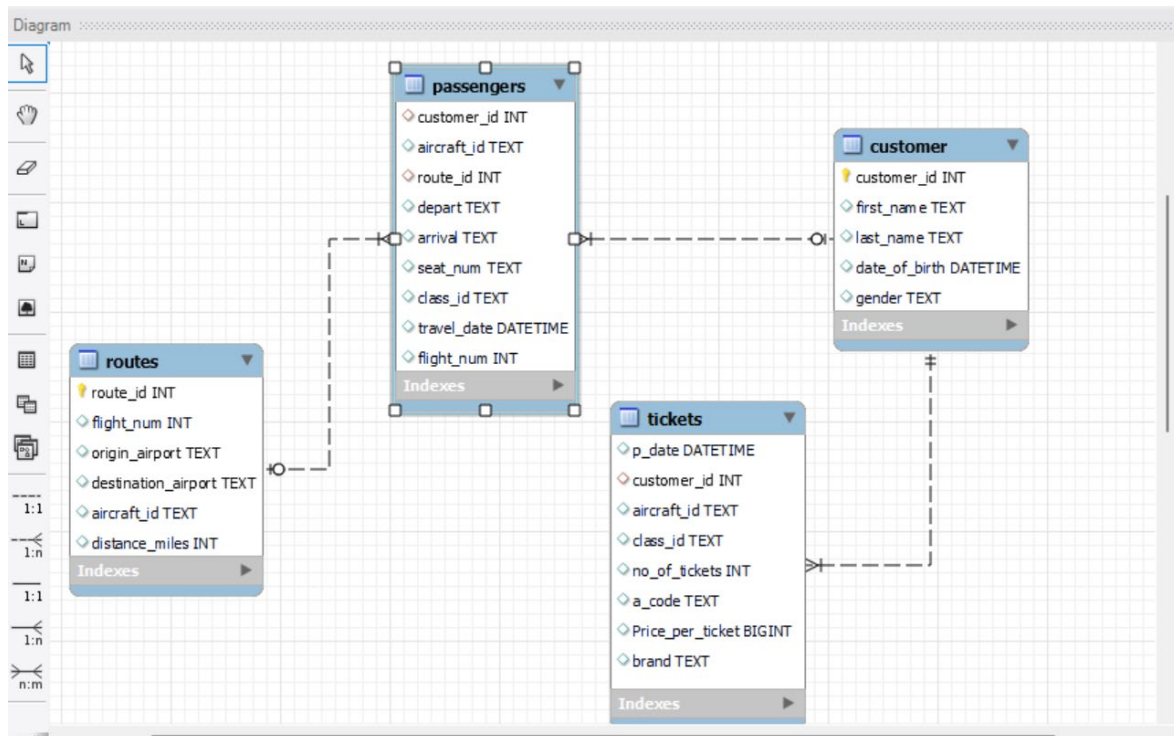


Q1. ER Diagram



Q2. We were advised to not to create a new table for this. Hence altered the existing table to add constraints.

Did this using both “table data import” wizard and using MYSQL queries. Using the wizard , for the Customer data set- data type changed for date_of_birth as below.

The screenshot shows the "Table Data Import" wizard in a database management tool. The "Configure Import Settings" tab is active. The detected file format is CSV, and the encoding is UTF-8. The columns to be imported are listed below:

Source Column	Field Type
customer_id	int
first_name	text
last_name	text
date_of_birth	datetime
gender	text

Below the column list, a preview of the data is shown:

customer_id	first_name	last_name	date_of_bi...	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

The date format is set to %d-%m-%Y. The wizard includes navigation buttons: "< Back", "Next >", and "Cancel".

Passenger's details table-travel date data type changed as below.

Table Data Import

Configure Import Settings

Detected file format: csv

Encoding: utf-8

Columns:

Column	Field Type
route_id	int
depart	text
arrival	text
seat_num	text
class_id	text
travel_date	datetime
flight_num	int

customer_id	aircraft_id	route_id	depart	arrival	seat_num	class_id	travel_date	flight_num
2	A321	34	CRW	COD	01B	Business	26/1/2019	1117
2	767-301ER	4	JFK	LAX	01E	Economy	2/9/2018	1114
1	ERJ142	9	DEN	LAX	01EP	Economy Pl...	26/12/2019	1119
1	CRJ900	30	BUR	STT	01FC	FirstClass	4/11/2018	1140
5	767-301ER	12	ABI	ADK	02B	Business	2/7/2018	1122

Date format: %d-%m-%Y

< Back Next > Cancel

For tickets_details data set- changed the P-date data type as date_time below

Table Data Import

Configure Import Settings

Detected file format: csv

Encoding: utf-8

Columns:

Source Column	Field Type
p_date	datetime
customer_id	int
aircraft_id	text
class_id	text
no_of_tickets	int
a_code	text

p_date	customer_id	aircraft_id	class_id	no_of_tickets	a_code	Price_per_...	brand
26-12-2018	27	767-301ER	Economy	1	DAL	130	Emirates
02-02-2020	22	ERJ142	Economy Pl...	1	AGB	220	Jet Airways
03-03-2020	21	CRJ900	Bussiness	1	BOH	490	Bristish Air...
04-04-2020	4	767-301ER	FirstClass	1	AGB	390	Emirates
05-05-2020	5	ERJ142	Economy	1	CTM	120	Jet Airways

Date format: %d-%m-%Y

< Back Next > Cancel

Then used SQL query as below to do add further constraints.

The screenshot shows the SQL Developer interface with the following SQL queries in the main editor:

```

1 • USE aircargo;
2 • ALTER table routes
3     ADD CONSTRAINT PRIMARY KEY (route_id),      -- primary key constraint
4     ADD UNIQUE ( route_id ),                  -- unique constraint
5     ADD CHECK (route_id is not null),          -- check constraint
6     ADD CHECK (distance_miles > 0);            -- check constraint
7
8 • DESCRIBE routes;
9
10 -- 3
  
```

Below the queries, the 'Result Grid' displays the table structure for 'routes':

Field	Type	Null	Key	Default	Extra
route_id	int	NO	PRI	NULL	
flight_num	int	NO		NULL	
origin_airport	text	YES		NULL	
destination_airport	text	YES		NULL	
aircraft_id	text	NO		NULL	
distance_miles	int	YES		NULL	

On the left, the 'SCHEMAS' pane shows the 'aircargo' database structure with tables: customer, passengers, routes, and tickets. The 'routes' table is selected, showing its columns: route_id (int, PK), flight_num (int), origin_airport (text), destination_airport (text), aircraft_id (text), and distance_miles (int).

Q3.

The screenshot shows the SQL Developer interface with the following SQL query in the main editor:

```

9
10 -- 3
11 -- Write a query to display all the passengers (customers) who have traveled in routes 01 to 25.
12 -- Take data from the passengers_on_flights table.
13
14 • SELECT *
15     FROM passengers
16     WHERE route_id BETWEEN '01' AND '25';
17
18
19
  
```

Below the queries, the 'Result Grid' displays the data for the query:

	customer_id	aircraft_id	route_id	depart	arrival	seat_num	class_id	travel_date	flight_num
▶	2	767-301ER	4	JFK	LAX	01E	Economy	2018-09-02 00:00:00	1114
	1	ERJ142	9	DEN	LAX	01EP	Economy Plus	2019-12-26 00:00:00	1119
	5	767-301ER	12	ABI	ADK	02B	Bussiness	2018-07-02 00:00:00	1122
	5	ERJ142	18	ANI	BGR	02E	Economy	2020-05-06 00:00:00	1128
	4	767-301ER	5	LAX	JFX	02FC	First Class	2020-04-06 00:00:00	1115
	7	767-301ER	20	AVL	BOI	03B	Bussiness	2020-07-08 00:00:00	1130
	5	ERJ142	22	BGR	BJI	03E	Economy	2020-05-31 00:00:00	1132

On the left, the 'SCHEMAS' pane shows the 'aircargo' database structure. The 'routes' table is selected, showing its columns: route_id (int, PK), flight_num (int), origin_airport (text), destination_airport (text), aircraft_id (text), and distance_miles (int).

Q4.

The screenshot shows a SQL IDE window with a query editor and a result grid. The query is designed to find the number of passengers and total revenue for the business class from the tickets table.

```

21  -- 4
22  -- Write a query to identify the number of passengers and total revenue in business class
23  -- from the ticket_details table.
24
25  • SELECT class_id,COUNT(Customer_id) AS NoOfPassengers ,
26      (SELECT SUM(no_of_tickets*Price_per_ticket) FROM tickets WHERE class_id= 'Bussiness')AS Revenue
27  FROM tickets
28  WHERE class_id LIKE 'Bussiness'
29  GROUP BY class_id;
30
31

```

The result grid displays the following data:

class_id	NoOfPassengers	Revenue
Bussiness	13	6034

The output pane shows a successful execution of the query, returning 1 row(s).

Q5.

The screenshot shows a SQL IDE window with a query editor and a result grid. The query is designed to display the full name of customers by concatenating their first and last names from the customer table.

```

33  -- 5
34  -- Write a query to display the full name of the customer by extracting the first name and last name
35  -- from the customer table
36
37  • SELECT first_name, last_name, CONCAT(first_name,' ', last_name) AS full_name
38  FROM customer;
39
40

```

The result grid displays the following data:

first_name	last_name	full_name
Julie	Sam	Julie Sam
Steve	Ryan	Steve Ryan
Morris	Lois	Morris Lois
Cathenna	Emily	Cathenna Emily
Aaron	Kim	Aaron Kim
Alexander	Scot	Alexander Scot
Anderson	Stewart	Anderson Stewart
Floyd	Ted	Floyd Ted
Leo	Travis	Leo Travis

The output pane shows a successful execution of the query, returning 50 row(s).

Q6.

SQL 3 Chapter3- HandsOn1 12072023 chap3-handson2 15072023 17072023 sqlproject* x routes - Table

Limit to 50000 rows

```

37
38 -- 6
39 -- Write a query to extract the customers who have registered and booked a ticket. Use data from the
40 -- customer and ticket_details tables
41
42 • SELECT *
43 FROM customer C
44 INNER JOIN tickets T ON C.customer_id = T.customer_id;
45
46

```

Result Grid

	customer_id	first_name	last_name	date_of_birth	gender	p_date	customer_id	aircraft_id	class_id
▶	27	Cherly	Vernon	1992-03-19 00:00:00	F	2018-12-26 00:00:00	27	767-301ER	Economy
	22	Pheny	Eri	1999-01-29 00:00:00	M	2020-02-02 00:00:00	22	ERJ142	Economy
	21	Chirsty	Josh	2004-01-10 00:00:00	M	2020-03-03 00:00:00	21	CRJ900	Business
	4	Cathenna	Emily	1977-09-14 00:00:00	F	2020-04-04 00:00:00	4	767-301ER	First Class
	5	Aaron	Kim	1991-02-18 00:00:00	M	2020-05-05 00:00:00	5	ERJ142	Economy
	7	Anderson	Stewart	1992-01-11 00:00:00	M	2020-07-07 00:00:00	7	767-301ER	Business

Result 23 x Read Only

Output

Action Output

#	Time	Action	Message
✓ 41	11:49:13	SELECT * FROM customer C INNER JOIN tickets T ON C.customer_id = T.customer_id;	50 row(s) returned

Q.7

SQL 3 Chapter3- HandsOn1 12072023 chap3-handson2 15072023 17072023 sqlproject* x routes - Table

Limit to 50000 rows

```

47 -- Write a query to identify the customer's first name and last name based on their customer ID and brand (Emirates)
48 -- from the ticket_details table
49 • SELECT DISTINCT T.customer_id, C.first_name, C.last_name, T.brand
50 FROM tickets T
51 LEFT JOIN customer C
52 ON T.customer_id= C.customer_id
53 WHERE brand = 'Emirates';
54

```

Result Grid

	customer_id	first_name	last_name	brand
▶	27	Cherly	Vernon	Emirates
	4	Cathenna	Emily	Emirates
	7	Anderson	Stewart	Emirates
	9	Leo	Travis	Emirates
	11	Roger	Walson	Emirates
	25	Moss	Morris	Emirates
	18	Gloria	Richie	Emirates
	14	Carol	Vernon	Emirates
	19	Joyce	Paul	Emirates

Result 30 x Read Only

Output

Action Output

#	Time	Action	Message
✓ 52	13:21:50	SELECT DISTINCT T.customer_id, C.first_name, C.last_name, T.brand FROM tickets T LEFT JOIN customer C ON T.customer_id= C.customer_id WHERE brand = 'Emirates';	14 row(s) returned

Q8.

SQL 3 Chapter3- HandsOn1 12072023 chap3-handson2 15072023 17072023 sqlproject routes - Table

Limit to 50000 rows

```

55 -- 08
56 -- Write a query to identify the customers who have traveled by Economy Plus class using
57 -- Group By and Having clause on the passengers_on_flights table.
58 • SELECT customer_id, class_id
59 FROM passengers
60 GROUP BY customer_id, class_id
61 HAVING class_id = 'Economy Plus';
62

```

Result Grid

	customer_id	class_id
▶	1	Economy Plus
	8	Economy Plus
	11	Economy Plus
	17	Economy Plus
	19	Economy Plus
	22	Economy Plus
	32	Economy Plus
	47	Economy Plus
	50	Economy Plus

passengers 35 x

Output

Action Output

#	Time	Action	Message
74	13:41:35	SELECT customer_id, class_id FROM passengers GROUP BY customer_id, class_i...	9 row(s) returned

Q9.

SQL 3 Chapter3- HandsOn1 12072023 chap3-handson2 15072023 17072023 sqlproject routes - Table

Limit to 50000 rows

```

62
63 -- 09 Write a query to identify whether the revenue has crossed 10000 using the IF clause on the
64 -- ticket_details table.
65
66 • SELECT
67 IF (SUM(no_of_tickets * Price_per_ticket) > 10000, "Yes", "No") as RevenueCrossed
68 FROM tickets;
69
70
71

```

Result Grid

	RevenueCrossed
▶	Yes

Result 39 x

Output

Action Output

#	Time	Action	Message
82	14:30:52	select if (sum(no_of_tickets * Price_per_ticket) > 10000, "Yes", "No") from tickets L...	1 row(s) returned

Q10.

Navigator: SCHEMAS

Filter objects

aircargo

- Tables
 - customer
 - passengers
 - routes
 - tickets
- Views
 - myview
- Stored Procedures
- Functions
- practice
- sys
- test

Administration Schemas

Information

View: myview

Columns:

Column	Type
customer_id	int
class_id	text
brand	text

Object Info Session

SQL 3 Chapter3- HandsOn1 12072023 chap3-handson2 15072023 17072023 sqlproject routes - Table

Limit to 50000 rows

```

71 -- 10 Write a query to create a view with only business class customers along with the brand of airlines.
72
73 CREATE VIEW myview AS
74 SELECT customer_id, class_id, brand
75 FROM tickets
76 WHERE class_id = 'Bussiness' ;
77
78 SELECT * FROM myview;
  
```

Result Grid

customer_id	class_id	brand
21	Bussiness	British Airways
7	Bussiness	Emirates
11	Bussiness	Emirates
25	Bussiness	Emirates
24	Bussiness	Qatar Airways
29	Bussiness	Qatar Airways
2	Bussiness	Qatar Airways
29	Bussiness	Jet Airways

myview 43 x

Output

Action Output

#	Time	Action	Message
92	14:44:20	CREATE VIEW myview AS SELECT customer_id, class_id, brand FROM tickets W...	0 row(s) affected
93	14:44:24	SELECT * FROM myview LIMIT 0, 50000	13 row(s) returned

Q11.

Navigator: SCHEMAS

Filter objects

aircargo

- Tables
 - customer
 - passengers
 - routes
 - tickets
- Views
- Stored Procedures
 - DistanceByCategory
 - get_passengers_det
 - TraveledDistance
- Functions
- practice
- sys
- test

Administration Schemas

Information

Procedure: get_passengers_details

Parameters:

Parameter	Type
start_num	[IN] INT
end_num	[IN] INT

SQL 3 Chapter3- HandsOn1 12072023 chap3-handson2 15072023 17072023* sqlproject

Limit to 50000 rows

```

81 -- Write a query to create a stored procedure to get the details of all passengers flying between
82 -- a range of routes defined in run time. Also, return an error message if the table doesn't exist.
83
84 DELIMITER &&
85 CREATE PROCEDURE get_passengers_details(IN start_num INT, IN end_num INT)
86 BEGIN
87 SELECT * FROM passengers
88 WHERE route_id >= start_num AND route_id <= end_num ;
89 END &&
90 DELIMITER ;
91
92 CALL get_passengers_details(4,5);
93
  
```

Result Grid

customer_id	aircraft_id	route_id	depart	arrival	seat_num	class_id	travel_date	flight_num
2	767-301ER	4	JFK	LAX	01E	Economy	2018-09-02 00:00:00	1114
4	767-301ER	5	LAX	JFK	02FC	First Class	2020-04-06 00:00:00	1115
4	767-301ER	4	JFK	LAX	03FC	First Class	2020-04-30 00:00:00	1114
11	767-301ER	5	LAX	JFK	04B	Bussiness	2020-11-12 00:00:00	1115
11	767-301ER	4	JFK	LAX	05B	Bussiness	2020-11-09 00:00:00	1114

Result 54 x

Q12.

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'aircargo' database schema with a table named 'TraveledDistance'. The right pane shows the SQL script for creating and calling the stored procedure.

```

-- 12 Write a query to create a stored procedure that extracts all the details from the routes table where
-- the traveled distance is more than 2000 miles.
DELIMITER &&
CREATE PROCEDURE TraveledDistance()
BEGIN
SELECT * FROM routes WHERE distance_miles >2000;
END &&
DELIMITER ;

CALL TraveledDistance();

```

The 'Result Grid' shows the output of the stored procedure call, displaying a list of routes with columns: route_id, flight_num, origin_airport, destination_airport, aircraft_id, and distance_miles.

route_id	flight_num	origin_airport	destination_airport	aircraft_id	distance_miles
1	1111	EWB	HNL	767-301ER	4962
2	1112	HNL	EWB	767-301ER	4962
3	1113	EWB	LHR	A321	3466
4	1114	JFK	LAX	767-301ER	2475
5	1115	LAX	JFK	767-301ER	2475
6	1116	HNL	LAX	767-301ER	2556

The 'Output' pane shows the execution log, indicating that the stored procedure was created successfully and returned 24 rows.

Q13.

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'aircargo' database schema with a table named 'DistanceByCategory'. The right pane shows the SQL script for creating and calling the stored procedure.

```

DELIMITER &&
CREATE PROCEDURE DistanceByCategory()
BEGIN
SELECT distance_miles ,
CASE WHEN distance_miles > 6500 THEN 'long-distance travel (LDT)'
      WHEN (distance_miles > 2000 AND distance_miles <= 6500) THEN 'intermediate distance travel (IDT)'
      ELSE 'short distance travel (SDT)'
END AS Result
FROM routes ;
END &&
DELIMITER ;

CALL DistanceByCategory();

```

The 'Result Grid' shows the output of the stored procedure call, displaying a list of routes with columns: distance_miles and Result.

distance_miles	Result
3365	intermediate distance travel (IDT)
4300	intermediate distance travel (IDT)
2232	intermediate distance travel (IDT)
2445	intermediate distance travel (IDT)
2000	short distance travel (SDT)
1700	short distance travel (SDT)
1900	short distance travel (SDT)