

ticket_details	
◇ p_date TEXT	
◇ customer_id INT	
◇ aircraft_id TEXT	
◇ class_id TEXT	
◇ no_of_tickets INT	
◇ a_code TEXT	
◇ Price_per_ticket INT	
◇ brand TEXT	
Indexes ▶	

customer	
🔑 customer_id INT	
◇ first_name TEXT	
◇ last_name TEXT	
◇ date_of_birth TEXT	
◇ gender TEXT	
Indexes ▶	

passengers_on_flights	
◇ customer_id INT	
◇ aircraft_id TEXT	
◇ route_id INT	
◇ depart TEXT	
◇ arrival TEXT	
◇ seat_num TEXT	
◇ class_id TEXT	
◇ travel_date TEXT	
◇ flight_num INT	
◇ passengers_on_flightscol VARCHAR(45)	
Indexes ▶	

route_details	
◇ route_id INT	
◇ flight_num INT	
◇ origin_airport VARCHAR(5)	
◇ destination_airport VARCHAR(5)	
◇ aircraft_id VARCHAR(15)	
◇ distance_miles INT	
Indexes ▶	

routes	
🔑 route_id INT	
◇ flight_num INT	
◇ origin_airport TEXT	
◇ destination_airport TEXT	
◇ aircraft_id TEXT	
◇ distance_miles INT	
Indexes ▶	



/\* 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. \*/

```
create table route_details (  
    route_id int, foreign key (route_id) references routes(route_id),  
    flight_num int check (flight_num > 1100),  
    origin_airport varchar(5) ,  
    destination_airport varchar(5) ,  
    aircraft_id varchar(10) ,  
    distance_miles int check (distance_miles > 0),  
    constraint unique_route_id unique (route_id)  
);
```

Result Grid



Filter Rows:

Export:



Wrap Cell Content:



route\_id

flight\_num

origin\_airport

destination\_airport





aircraft\_id

distance\_miles

```

41  /* 3. Write a query to display all the passengers (customers)
42  who have traveled in routes 01 to 25. Take data from the passengers_on_flights table. */
43
44  select p.customer_id, p.route_id, first_name, last_name
45  from passengers_on_flights p
46  inner join customer c
47  on p.customer_id = c.customer_id
48  group by p.customer_id, p.route_id, first_name, last_name
49  having p.route_id between 1 and 25
50
51  -- 4. number of passengers and total revenue in business class

```

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

	customer_id	route_id	first_name	last_name
▶	2	4	Steve	Ryan
	1	9	Julie	Sam
	5	12	Aaron	Kim
	5	18	Aaron	Kim
	4	5	Cathenna	Emily
	7	20	Anderson	Stewart
	5	22	Aaron	Kim
	4	4	Cathenna	Emily
	11	5	Roger	Walson
	17	13	Catherine	Shad
	9	15	Leo	Travis
	11	4	Roger	Walson
	10	10	Melvin	Tracy
	15	14	Linda	William
	13	13	Solomon	Walter
	22	22	Pheny	Eri
	24	14	Calvin	Willis

Result 1 x

```
51  /* 4. Write a query to identify the number of passengers and total revenue
52  in business class from the ticket_details table. */
53
54  select class_id as Class,
55         count(*) as "Number of passengers",
56         concat("$",sum(Price_per_ticket * no_of_tickets)) as "Total revenue"
57  from ticket_details t
58  where class_id = 'Business'
59  group by class_id
60
61  -- 5. display the full name of the customer
```

Result Grid



Filter Rows:

Export:



Wrap Cell Content:



	Class	Number of passengers	Total revenue
►	Business	13	\$6034

```

60
61  /* 5. Write a query to display the full name of the customer
62  by extracting the first name and last name from the customer table. */
63
64  select customer_id,
65         concat(first_name, ' ', last_name) as "Full Name",
66         date_of_birth,
67         gender
68  from customer

```

Result Grid



Filter Rows:

Export:



Wrap Cell Content:




	customer_id	Full Name	date_of_birth	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
	6	Alexander Scot	12-02-1985	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
	12	Shirley Wally	25-06-1997	F
	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

```

70  /* 6. Write a query to extract the customers who have registered and booked a ticket.
71  Use data from the customer and ticket_details tables. */
72
73  select t.customer_id,
74         concat(first_name, ' ', last_name) as "Full Name",
75         p_date,
76         class_id,
77         brand
78  from customer c
79  inner join ticket_details t
80  on c.customer_id = t.customer_id
81  where no_of_tickets = 1
82  group by t.customer_id, p_date, class_id, brand

```

Result Grid   Filter Rows:  Export:  Wrap Cell Content: 





	customer_id	Full Name	p_date	class_id	brand
▶	27	Cherly Vernon	26-12-2018	Economy	Emirates
	22	Pheny Eri	02-02-2020	Economy Plus	Jet Airways
	21	Chirsty Josh	03-03-2020	Business	British Airways
	4	Cathenna Emily	04-04-2020	First Class	Emirates
	5	Aaron Kim	05-05-2020	Economy	Jet Airways
	7	Anderson Stewart	07-07-2020	Business	Emirates
	8	Floyd Ted	08-08-2020	Economy Plus	Qatar Airways
	9	Leo Travis	09-09-2020	First Class	Emirates
	10	Melvin Tracy	10-10-2020	Economy	Qatar Airways
	11	Roger Walson	11-11-2020	Business	Emirates
	19	Joyce Paul	12-12-2020	Economy Plus	British Airways
	13	Solomon Walter	01-01-2019	First Class	Qatar Airways
	14	Carol Vernon	02-02-2019	Economy	Jet Airways
	25	Moss Morris	03-03-2019	Business	Emirates
	16	Chirstine Willis	04-04-2019	First Class	British Airways
	17	Catherine Shad	03-05-2019	Economy Plus	Qatar Airways
	18	Gloria Richie	06-06-2019	Economy	Emirates
	24	Calvin Willis	07-07-2019	Business	Qatar Airways
	20	Sara Oliver	09-08-2019	First Class	British Airways



```

84  /* 7. Write a query to identify the customer's first name and last name based
85  on their customer ID and brand (Emirates) from the ticket_details table. */
86
87  select  t.customer_id,
88          first_name,
89          last_name,
90          brand
91  from customer c
92  inner join ticket_details t
93  on c.customer_id = t.customer_id
94  where no_of_tickets = 1 and brand = 'Emirates'
95  group by t.customer_id, first_name, last_name, brand
96

```

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

	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
	5	Aaron	Kim	Emirates
	2	Steve	Ryan	Emirates
	31	James	Robert	Emirates
	49	Russell	Peter	Emirates
	44	Bily	Brian	Emirates



```

97  /* 8. Write a query to identify the customers who have traveled by Economy Plus class
98  using Group By and Having clause on the passengers_on_flights table. */
99
100  select p.customer_id,
101         concat(first_name, ' ', last_name) as "Full Name",
102         travel_date,
103         class_id
104  from customer c
105  inner join passengers_on_flights p
106  on c.customer_id = p.customer_id
107  group by p.customer_id, travel_date, class_id
108  having class_id = 'Economy Plus'
109
110

```

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

	customer_id	Full Name	travel_date	class_id
▶	1	Julie Sam	26-12-2019	Economy Plus
	8	Floyd Ted	09-08-2020	Economy Plus
	11	Roger Walson	02-08-2018	Economy Plus
	17	Catherine Shad	03-06-2019	Economy Plus
	19	Joyce Paul	13-01-2021	Economy Plus
	19	Joyce Paul	17-12-2020	Economy Plus
	22	Pheny Eri	09-02-2020	Economy Plus
	32	Chirstoper Sean	04-03-2021	Economy Plus
	47	Sophia Carl	15-12-2020	Economy Plus
	50	Rose Arthur	15-08-2020	Economy Plus

```
110  -- 9. Write a query to identify whether the revenue has crossed 10000 using the IF clause on the ticket_details table.
111
112  select sum(price_per_ticket * no_of_tickets) as Total_Revenue,
113         if(sum(price_per_ticket * no_of_tickets) > 10000, 'Yes', 'No')
114         as Revenue_crossed_10000
115  from ticket_details
116
```

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

	Total_Revenue	Revenue_crossed_10000
►	15369	Yes

Views

- ticket\_details
- vw\_business\_class

Stored Procedures

Functions

bcg\_wave

new\_bcg

sys

test

Administration Schemas

Information

View:

vw\_business\_class

Columns:

customer_id	int
Full Name	mediumtext
class_id	text
brand	text

```

108 having class_id = 'Economy Plus'
109
110 -- 9. Write a query to identify whether the revenue has crossed 10000 using the IF clause on the ticket_details table.
111
112 select sum(price_per_ticket * no_of_tickets) as Total_Revenue,
113        if(sum(price_per_ticket * no_of_tickets) > 10000, 'Yes', 'No')
114        as Revenue_crossed_10000
115 from ticket_details
116
117 -- 10. Write a query to create a view with only business class customers along with the brand of airlines.
118
119 create view vw_business_class as
120
121 select t.customer_id,
122        concat(first_name, ' ', last_name) as "Full Name",
123        t.class_id,
124        brand
125 from customer c
126 inner join passengers_on_flights p
127 on c.customer_id = p.customer_id
128 inner join ticket_details t
129 on p.customer_id = t.customer_id
130 group by p.customer_id, class_id, brand
131 having class_id = 'Business'
132
133

```

Output

Action Output

#	Time	Action	Message
29	20:29:05	create view vw_business_class as select t.customer_id, concat(first_name, ' ', last_name) as "Full Name", t.class_id, brand from customer c in...	Error Code: 1050. Table 'vw_business_class' already exists
30	20:29:19	create view vw_business_class as select t.customer_id, concat(first_name, ' ', last_name) as "Full Name", t.class_id, brand from customer c in...	Error Code: 1050. Table 'vw_business_class' already exists
31	20:29:32	DROP VIEW 'bcg_sql_project'.vw_business_class	0 row(s) affected
32	20:29:53	create view vw_business_class as select t.customer_id, concat(first_name, ' ', last_name) as "Full Name", t.class_id, brand from customer c in...	0 row(s) affected

Object Info Session

Stored Procedures

- get\_passengers\_between\_routes

Functions

- bcg\_wave
- new\_bcg
- sys
- test

Administration Schemas

Information

No object selected

```

133
134  /* 11. Write a query to create a stored procedure to get the details of all passengers
135     flying between a range of routes defined in run time. Also, return an error message
136     if the table doesn't exist. */
137
138     delimiter $$
139  • create procedure get_passengers_between_routes
140     (in start_route int, in end_route int)
141  begin
142     if not exists
143         (select * from passengers_on_flights where route_id between start_route and end_route)
144     then
145         select "The table doesnt exist" AS error_message;
146     else
147         select c.first_name, c.last_name, p.*
148         from passengers_on_flights p
149         inner join
150         customer c
151         on p.customer_id = c.customer_id
152         where p.route_id between start_route and end_route;
153     end if;
154 end $$
155 delimiter ;
156

```

Output



Action Output

	#	Time	Action	Message	Duration / Fetch
✓	38	20:49:57	DROP PROCEDURE 'bcg_sql_project'. 'Dist_travel'	0 row(s) affected	0.000 sec
✓	39	20:50:00	DROP PROCEDURE 'bcg_sql_project'. 'get_passeng...	0 row(s) affected	0.000 sec
✓	40	20:50:02	DROP PROCEDURE 'bcg_sql_project'. 'get_routes_o...	0 row(s) affected	0.000 sec
✓	41	20:50:38	create procedure get_passengers_between_routes (in...	0 row(s) affected	0.000 sec

Object Info Session

Query Completed



views

- Stored Procedures
  - get\_passengers\_between\_routes
  - get\_routes\_over\_2000\_miles
- Functions
  - bcg\_wave
  - new\_bcg
  - sys
  - test

```

157  /* 12. Write a query to create a stored procedure that extracts all the details
158      from the routes table where the traveled distance is more than 2000 miles. */
159
160      delimiter $$
161  •   create procedure get_routes_over_2000_miles()
162      begin
163          select * FROM routes
164          where distance_miles > 2000;
165      end $$
166      delimiter ;

```

Administration Schemas

Information

No object selected

Output

Action Output

#	Time	Action	Message	Duration / Fetch
✓ 26	20:27:52	DROP PROCEDURE 'bcg_sql_project'. 'Dist_travel'	0 row(s) affected	0.016 sec
✓ 27	20:27:55	DROP PROCEDURE 'bcg_sql_project'. 'get_passeng...	0 row(s) affected	0.000 sec
✓ 28	20:27:57	DROP PROCEDURE 'bcg_sql_project'. 'get_routes_o...	0 row(s) affected	0.000 sec
✗ 29	20:29:05	create view vw_business_class as select t.customer...	Error Code: 1050. Table 'vw_business_class' already e...	0.000 sec
✗ 30	20:29:19	create view vw_business_class as select t.customer...	Error Code: 1050. Table 'vw_business_class' already e...	0.000 sec
✓ 31	20:29:32	DROP VIEW 'bcg_sql_project'. 'vw_business_class'	0 row(s) affected	0.000 sec
✓ 32	20:29:53	create view vw_business_class as select t.customer...	0 row(s) affected	0.000 sec
✓ 33	20:34:49	create procedure get_passengers_between_routes (in...	0 row(s) affected	0.000 sec
✓ 34	20:38:12	create procedure get_routes_over_2000_miles() BEGI...	0 row(s) affected	0.000 sec
✓ 35	20:38:20	call bcg_sql_project.get_routes_over_2000_miles()	24 row(s) returned	0.016 sec / 0.000 sec
✓ 36	20:44:44	create procedure Dist_travel () BEGIN select case wh...	0 row(s) affected	0.015 sec
✗ 37	20:49:10	create procedure get_passengers_between_routes (in...	Error Code: 1304. PROCEDURE get_passengers_bet...	0.000 sec
✓ 38	20:49:57	DROP PROCEDURE 'bcg_sql_project'. 'Dist_travel'	0 row(s) affected	0.000 sec
✓ 39	20:50:00	DROP PROCEDURE 'bcg_sql_project'. 'get_passeng...	0 row(s) affected	0.000 sec
✓ 40	20:50:02	DROP PROCEDURE 'bcg_sql_project'. 'get_routes_o...	0 row(s) affected	0.000 sec
✓ 41	20:50:38	create procedure get_passengers_between_routes (in...	0 row(s) affected	0.000 sec
✓ 42	20:52:49	create procedure get_routes_over_2000_miles() begin...	0 row(s) affected	0.000 sec

Object Info Session

**SCHEMAS**

Filter objects

- bcg\_sql\_project**
  - Tables
    - customer
    - passengers\_on\_flights
    - route\_details
    - routes
    - ticket\_details
  - Views
  - Stored Procedures
    - Dist\_travel
    - get\_passengers\_between\_routes
    - get\_routes\_over\_2000\_miles
  - Functions
- bcg\_wave
- new\_bcg
- sys
- test

Administration Schemas

Information

No object selected

```

170  /*13. Write a query to create a stored procedure that groups the distance traveled by
171      each flight into three categories. The categories are, short distance travel (SDT)
172      for >=0 AND <= 2000 miles, intermediate distance travel (IDT) for >2000 AND <=6500,
173      and long-distance travel (LDT) for >6500. */
174
175
176      delimiter $$
177  •   create procedure Dist_travel ()
178      begin
179          select case
180              when distance_miles between 0 and 2000 then 'Short Distance Travel'
181              when distance_miles >2000 and distance_miles <=6500 then 'Intermediate Distance Travel'
182              else 'Long Distance Travel'
183          end
184          as dist_class,
185          route_id from routes
186          order by distance_miles, dist_class, route_id;
187      end $$
188      delimiter ;
189
190

```

Output

Action Output

	#	Time	Action	Message	Duration / Fetch
✓	31	20:29:32	DROP VIEW 'bcg_sql_project'.vw_business_class	0 row(s) affected	0.000 sec
✓	32	20:29:53	create view vw_business_class as select t.customer...	0 row(s) affected	0.000 sec
✓	33	20:34:49	create procedure get_passengers_between_routes (in...	0 row(s) affected	0.000 sec
✓	34	20:38:12	create procedure get_routes_over_2000_miles() BEGI...	0 row(s) affected	0.000 sec
✓	35	20:38:20	call bcg_sql_project.get_routes_over_2000_miles()	24 row(s) returned	0.016 sec / 0.000 sec
✓	36	20:44:44	create procedure Dist_travel () BEGIN select case wh...	0 row(s) affected	0.015 sec
✗	37	20:49:10	create procedure get_passengers_between_routes (in...	Error Code: 1304. PROCEDURE get_passengers_bet...	0.000 sec
✓	38	20:49:57	DROP PROCEDURE 'bcg_sql_project'.Dist_travel	0 row(s) affected	0.000 sec
✓	39	20:50:00	DROP PROCEDURE 'bcg_sql_project'.get_passeng...	0 row(s) affected	0.000 sec
✓	40	20:50:02	DROP PROCEDURE 'bcg_sql_project'.get_routes_o...	0 row(s) affected	0.000 sec
✓	41	20:50:38	create procedure get_passengers_between_routes (in...	0 row(s) affected	0.000 sec
✓	42	20:52:49	create procedure get_routes_over_2000_miles() begin...	0 row(s) affected	0.000 sec
✓	43	20:58:26	create procedure Dist_travel () begin select case whe...	0 row(s) affected	0.000 sec