Source Code

Answer 2:

- ```
  SELECT * FROM customer
  WHERE customer_id IN (SELECT DISTINCT customer_id FROM passengers_on_flights WHERE route_id between 1 and 25)
  ORDER BY customer_id;
  ```

Answer 3:

- ```
  SELECT COUNT(DISTINCT customer_id) AS num_passenger, SUM(no_of_tickets * price_per_ticket) AS total_revenue
  FROM ticket_details
  WHERE class_id = 'Bussiness';
  ```

Answer 4:

- ```
  SELECT CONCAT(first_name, " ", last_name) AS Full_name
  FROM customer;
  ```

Answer 5: There are two ways to solve:-

1. ```
   SELECT DISTINCT c.customer_id, c.first_name, c.last_name
   FROM customer c
   JOIN ticket_details td
   ON c.customer_id = td.customer_id
   WHERE no_of_tickets > 0
   ORDER BY customer_id ASC;
   ```

2. ```
   SELECT first_name, last_name FROM customer
   WHERE customer_id
   IN (SELECT DISTINCT tb.customer_id FROM customer c,ticket_details tb);
   ```

Answer 6: There are two ways to solve:-

- ```
  SELECT first_name, last_name FROM customer
  WHERE customer_id IN (SELECT DISTINCT customer_id FROM ticket_details WHERE brand = "Emirates");
  ```

- ```
  SELECT DISTINCT c.customer_id, c.first_name, c.last_name FROM customer c
  JOIN ticket_details td
  ON c.customer_id = td.customer_id
  WHERE brand = "Emirates"
  ORDER BY 1;
  ```
  ```

Source Code

Answer 7: There are two ways to solve:-

- ```
  SELECT * FROM customer c
  INNER JOIN (SELECT DISTINCT customer_id FROM passengers_on_flights pof WHERE class_id = "economy plus")
  pof
  ON c.customer_id = pof.customer_id;
  ```

- ```
  SELECT c.customer_id, c.first_name, c.last_name FROM customer c
  INNER JOIN passengers_on_flights pof
  ON c.customer_id = pof.customer_id
  WHERE pof.class_id = 'Economy Plus'
  GROUP BY c.customer_id, c.first_name, c.last_name
  HAVING COUNT(*) >= 1;
  ```

Answer 8:

- ```
  SELECT IF(SUM(no_of_tickets * price_per_ticket) >= 10000, 'Crossed 10K', 'Not Crossed 10K') AS revenue_check
  FROM ticket_details;
  ```

Answer 9: -- create a new user

- CREATE USER if not exists 'Anubhav'@'127.0.0.1' IDENTIFIED BY 'password';

  -- grant selected privileges to the new user

- GRANT SELECT, INSERT, UPDATE, DELETE ON database_name.* TO new_user;

  -- grant all privileges to the new user

- GRANT all privileges ON airlines TO 'Anubhav'@'127.0.0.1';

Answer 10:

- ```
  SELECT DISTINCT class_id, MAX(price_per_ticket) OVER (PARTITION BY class_id) AS max_ticket_price
  FROM ticket_details
  ORDER BY max_ticket_price;
  ```

Answer 11:

- ```
  EXPLAIN SELECT * FROM passengers_on_flights WHERE route_id = 4;
  ```
  ```

Source Code

Answer 12:

- CREATE INDEX idx_route_id ON passengers_on_flights (route_id);

- EXPLAIN SELECT * FROM passengers_on_flights WHERE route_id = 4;

Answer 13: WITH ROLLUP: The WITH ROLLUP option adds extra rows to the result set that provide subtotal and grand total values for each customer_id and for all customers, respectively.

- 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;

  WITHOUT ROLLUP:

- SELECT customer_id, aircraft_id, SUM(price_per_ticket * no_of_tickets) AS total_price
  FROM ticket_details
  GROUP BY customer_id, aircraft_id
  ORDER BY customer_id, aircraft_id;

Answer 14: CREATE VIEW

- CREATE VIEW Bussiness_Class_Customers As
  SELECT c.*, td.brand FROM customer c
  INNER JOIN (SELECT DISTINCT customer_id, brand FROM ticket_details WHERE class_id = 'Bussiness' ORDER BY customer_id) td
  ON c.customer_id = td.customer_id;

  DISPLAY VIEW

- SELECT * FROM Bussiness_Class_Customers;

Answer 15: CREATE PROCEDURE

- DELIMITER //
  CREATE PROCEDURE Check_distance()
  Begin
  SELECT * FROM routes WHERE distance_miles > 2000;
  End //
  DELIMITER ;

  CALL PROCEDURE

- CALL check_distance;

Source Code

Answer 16:

- ```
  DELIMITER //
  CREATE FUNCTION group_dist(dist INT)
  returns VARCHAR(10)
  deterministic
  Begin
          DECLARE dist_cat CHAR(3);
          IF dist BETWEEN 0 AND 2000 THEN
                  SET dist_cat = 'SDT';

          ELSEIF dist BETWEEN 2001 AND 6500 THEN
                  SET dist_cat = 'IDT';
          ELSEIF dist > 6500 THEN
                  SET dist_cat = 'LDT';
          End IF;
          return(dist_cat);
  END //
  ```

- ```
  CREATE PROCEDURE group_dist_proc()
  Begin
          SELECT flight_num, distance_miles, group_dist(distance_miles) AS distance_category FROM routes;
  End //
  DELIMITER ;
  ```

- ```
  CALL group_dist_proc;
  ```
  ```

Source Code

Answer 17: Without Stored procedure:

- SELECT p_date, customer_id, class_id,
  CASE
  WHEN class_id in ('Bussiness', 'Economy Plus') THEN 'Yes'
  Else 'No'
  End as complimentary_service FROM ticket_details;

With store Procedure:

- DELIMITER //
  CREATE FUNCTION check_comp_serv (cls varchar (15))
  returns char (3)
  deterministic
  Begin
  declare comp_ser char (3);
  If cls in ('Bussiness', 'Economy Plus') then
  set comp_ser = 'Yes';
  Else
  set comp_ser = 'NO';
  End if;
  return(comp_ser);
  End //

- CREATE PROCEDURE check_comp_serv_proc()
  Begin
  SELECT p_date, customer_id, class_id, check_comp_serv(class_id) AS complimentary_service FROM ticket_details;
  End //
  DELIMITER ;

- CALL check_comp_serv_proc;

Answer 18:

- SELECT * FROM customer
  WHERE last_name = 'Scott'
  LIMIT 1;