

1)

The screenshot shows a database console interface with a dark theme. On the left, the 'Database Explorer' pane shows a tree structure with 'database' at the top, followed by 'airport', 'postgres', 'information_schema', 'pg_catalog', and 'public'. Under 'public', there is a 'tables' folder containing 'airline', 'airport', 'baggage', 'baggage_check', 'boarding_pass', 'booking', 'booking_flight', 'flights', 'passengers', and 'security_check'. The 'flights' table is selected. The main console area shows a PL/pgSQL procedure named 'insert_flight' being executed. The procedure code is as follows:

```
1 CREATE OR REPLACE PROCEDURE insert_flight(  
2     flight_id INT, flight_no VARCHAR(50), scheduled_departure DATE,  
3     scheduled_arrival DATE, departure_airport_id INT, arrival_airport_id INT, departing_gate VARCHAR(50),  
4     arriving_gate VARCHAR(50), airline_id INT, status VARCHAR(50),  
5     actual_departure DATE, actual_arrival DATE, created_at DATE, update_at DATE  
6 )  
7 AS $$  
8 BEGIN  
9     INSERT INTO flights (flight_id, flight_no, scheduled_departure,  
10    scheduled_arrival, departure_airport_id, arrival_airport_id, arrival_airport_id,  
11    departing_gate, arriving_gate, airline_id, status, actual_departure, actual_arrival, created_at, update_at)  
12    VALUES (flight_id, flight_no, scheduled_departure,  
13    scheduled_arrival, departure_airport_id, arrival_airport_id, arrival_airport_id,  
14    departing_gate, arriving_gate, airline_id, status, actual_departure, actual_arrival, created_at, update_at);  
15    RAISE NOTICE 'Flight % successfully added', flight_no;  
16 END;  
17 $$ LANGUAGE plpgsql;
```

Below the console, the 'Services' pane shows the execution details of the procedure. It displays the same code as the console, followed by the execution time: '[2025-12-02 17:56:04] completed in 36 ms'.

2)

The screenshot shows the same database console interface as the first one. The 'Database Explorer' pane is identical. The main console area shows a PL/pgSQL procedure named 'update_flight_status' being executed. The procedure code is as follows:

```
1 CREATE OR REPLACE PROCEDURE update_flight_status(  
2     p_flight_id INT,  
3     p_new_status VARCHAR(20)  
4 )  
5 AS $$  
6 BEGIN  
7     UPDATE flights  
8     SET status = p_new_status  
9     WHERE flight_id = p_flight_id;  
10  
11    RAISE NOTICE 'Flight % status updated to: %', p_flight_id, p_new_status;  
12 END;  
13 $$ LANGUAGE plpgsql;
```

Below the console, the 'Services' pane shows the execution details of the procedure. It displays the same code as the console, followed by the execution time: '[2025-12-02 18:00:59] completed in 6 ms'.

3)

The screenshot shows a database console interface with a dark theme. On the left, the 'Database Explorer' pane shows a tree view of the database structure, including 'database', 'airport', 'postgres', 'information_schema', 'pg_catalog', and 'public'. The 'public' schema contains a 'tables' folder with various tables like 'airline', 'airport', 'baggage', 'baggage_check', 'boarding_pass', 'booking', 'booking_flight', 'flights', 'passengers', and 'security_check'. The main editor area displays a SQL script for creating a function named 'get_flights_by_departure_airport'. The script is in PL/pgSQL and includes a RETURNING clause. The 'Services' pane at the bottom shows the execution of the function, with a message indicating it completed in 6 ms. The status bar at the bottom indicates the file encoding is UTF-8 and the line ending is CRLF.

```
1 CREATE OR REPLACE FUNCTION get_flights_by_departure_airport(  
2   p_departure_airport VARCHAR(50)  
3 )  
4 RETURNS TABLE (  
5   flight_id INT, flight_no VARCHAR(50), scheduled_departure DATE,  
6   scheduled_arrival DATE, departure_airport_id INT, arrival_airport_id INT, departing_gate VARCHAR(50),  
7   arriving_gate VARCHAR(50), airline_id INT, status VARCHAR(50),  
8   actual_departure DATE, actual_arrival DATE, created_at DATE, update_at DATE  
9 )  
10 AS $$  
11 BEGIN  
12   RETURN QUERY  
13     SELECT  
14       f.flight_id,  
15       f.flight_no,  
16       air.airport_id,  
17       air.airport_name  
18     FROM flights f  
19     JOIN airport air ON f.departure_airport_id = air.airport_id  
20     WHERE air.airport_name = 'Fort Worth Alliance Airport';  
21 END;  
22 $$ LANGUAGE plpgsql;
```

Services

```
JOIN airport air ON f.departure_airport_id = air.airport_id  
WHERE air.airport_name = 'Fort Worth Alliance Airport';  
END;  
$$ LANGUAGE plpgsql
```

[2025-12-02 18:12:58] completed in 6 ms

4)

The screenshot shows a database console interface with a dark theme. On the left, the 'Database Explorer' pane shows a tree view of the database structure, including 'database', 'airport', 'postgres', 'information_schema', 'pg_catalog', and 'public'. The 'public' schema contains a 'tables' folder with various tables like 'airline', 'airport', 'baggage', 'baggage_check', 'boarding_pass', 'booking', 'booking_flight', 'flights', 'passengers', and 'security_check'. The main editor area displays a SQL script for creating a function named 'calculates_average_delay'. The script is in PL/pgSQL and includes a RETURNING clause. The 'Services' pane at the bottom shows the execution of the function, with a message indicating it completed in 6 ms. The status bar at the bottom indicates the file encoding is UTF-8 and the line ending is CRLF.

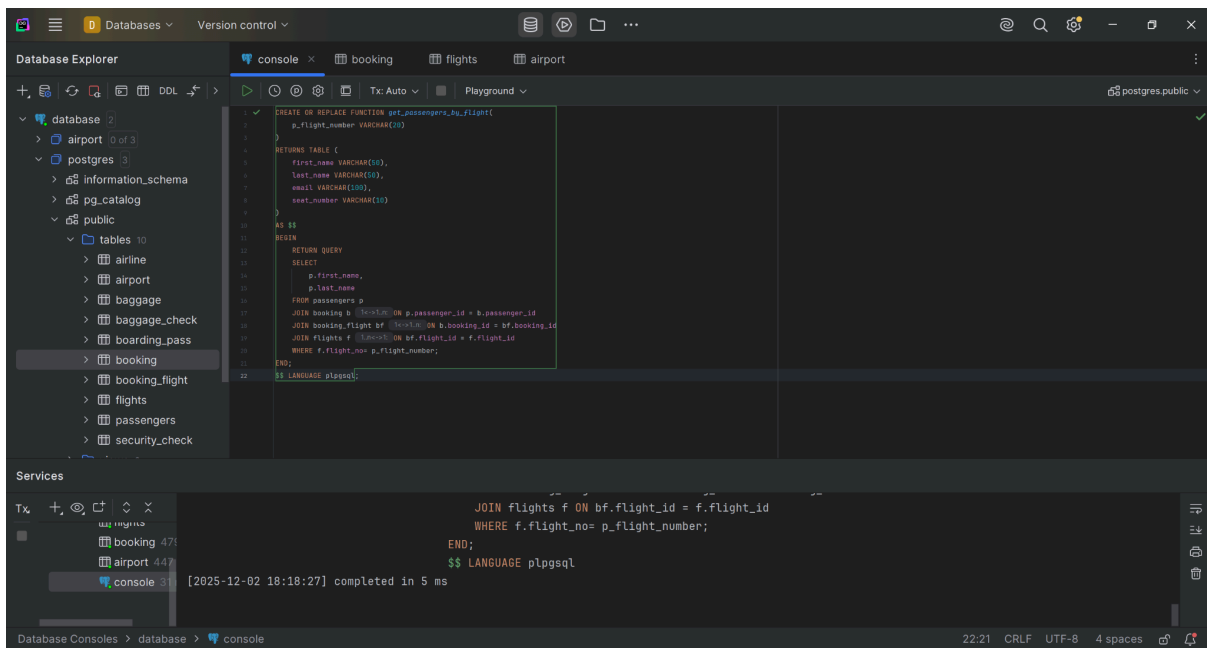
```
1 CREATE OR REPLACE FUNCTION calculates_average_delay(  
2   p_arrival_airport_id INT  
3 )  
4 RETURNS DECIMAL(10,2)  
5 AS $$  
6 DECLARE  
7   avg_delay DECIMAL(10,2);  
8 BEGIN  
9   SELECT AVG(  
10     EXTRACT(EPOCH FROM (actual_arrival - scheduled_arrival)) / 60  
11   ) INTO avg_delay  
12   FROM flights  
13   WHERE arrival_airport_id = p_arrival_airport_id  
14   AND actual_arrival > scheduled_arrival;  
15  
16   RETURN COALESCE(avg_delay, 0);  
17 END;  
18 $$ LANGUAGE plpgsql;
```

Services

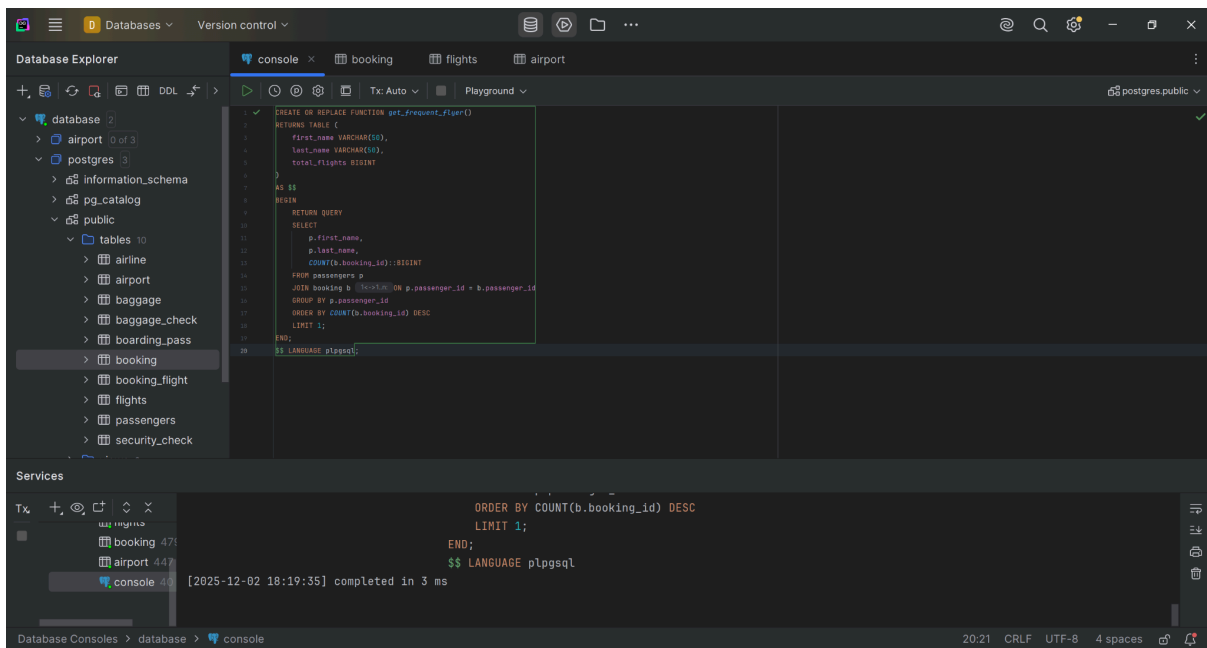
```
RETURN COALESCE(avg_delay, 0);  
END;  
$$ LANGUAGE plpgsql
```

[2025-12-02 18:17:12] completed in 6 ms

5)



6)



7)

The screenshot shows a database IDE with a 'Database Explorer' on the left and a 'console' tab active. The 'Database Explorer' shows a 'public' schema with a 'tables' folder containing 'airline', 'airport', 'baggage', 'baggage_check', 'boarding_pass', 'booking', 'booking_flight', 'flights', 'passengers', and 'security_check'. The 'console' tab displays a SQL query to create a function 'get_severely_delayed_flights()' that returns a table of flight details for flights delayed by more than 24 hours. The query is as follows:

```
1 CREATE OR REPLACE FUNCTION get_severely_delayed_flights()
2 RETURNS TABLE (
3     flight_number VARCHAR(20),
4     departure_code VARCHAR(10),
5     arrival_code VARCHAR(10),
6     delay_hours DECIMAL(10,2)
7 )
8 AS $$
9 BEGIN
10     RETURN QUERY
11     SELECT
12         f.flight_no,
13         dep.airport_name,
14         arr.airport_name,
15         EXTRACT(EPOCH FROM (f.actual_departure - f.scheduled_departure)) / 3600
16     FROM flights f
17     JOIN airport dep ON f.departure_airport_id = dep.airport_id
18     JOIN airport arr ON f.arrival_airport_id = arr.airport_id
19     WHERE (f.actual_departure - f.scheduled_departure) > INTERVAL '24 hours'
20     ORDER BY delay_hours DESC;
21 END;
22 $$ LANGUAGE plpgsql;
```

The 'Services' panel at the bottom shows the execution of the query, which completed in 4 ms. The status bar at the bottom indicates the file encoding is UTF-8 and the line ending is CRLF.

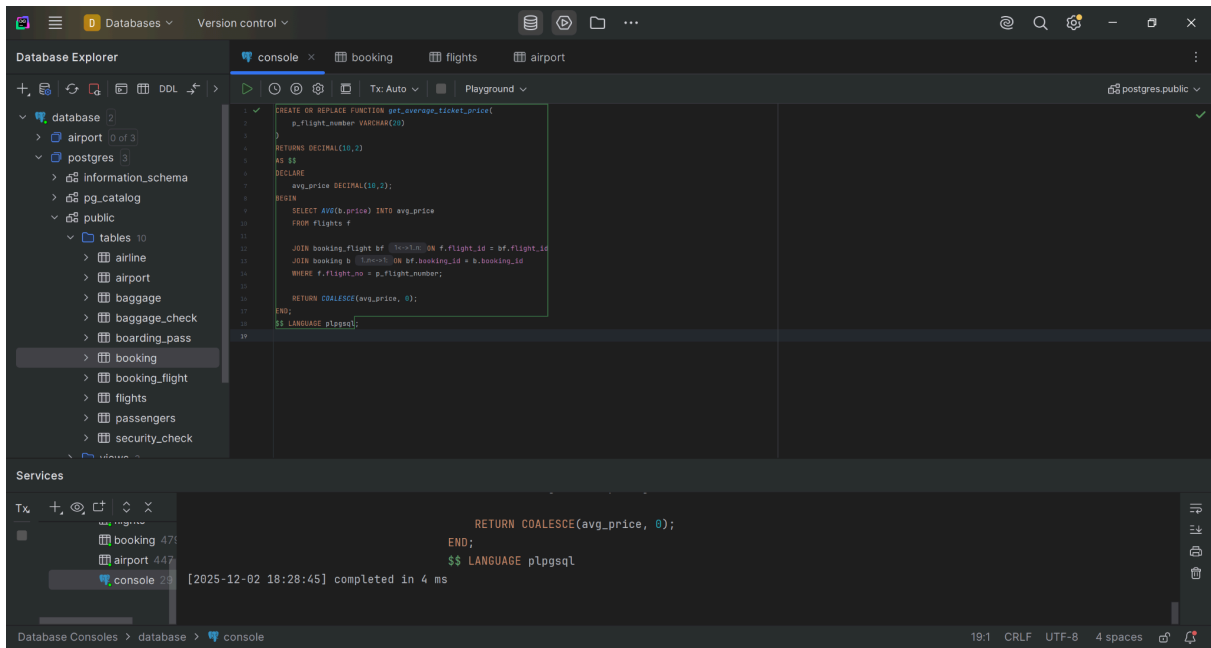
8)

The screenshot shows the same database IDE with a 'console' tab active. The 'Database Explorer' shows the same 'public' schema with a 'tables' folder containing 'airline', 'airport', 'baggage', 'baggage_check', 'boarding_pass', 'booking', 'booking_flight', 'flights', 'passengers', and 'security_check'. The 'console' tab displays a SQL query to create a function 'count_flights_per_airline()' that returns a table of flight counts per airline. The query is as follows:

```
1 CREATE OR REPLACE FUNCTION count_flights_per_airline()
2 RETURNS TABLE (
3     p_airline_id INT,
4     flight_count INT
5 )
6 AS $$
7 BEGIN
8     SELECT COUNT(*) INTO flight_count
9     FROM flights
10    WHERE airline_id = p_airline_id;
11    RETURN flight_count;
12 END;
13 $$ LANGUAGE plpgsql;
```

The 'Services' panel at the bottom shows the execution of the query, which completed in 4 ms. The status bar at the bottom indicates the file encoding is UTF-8 and the line ending is CRLF.

9)



10)

