

```
console [lab4@localhost] ~ - airline [lab4@localhost] airport [lab4@localhost] booking [lab4@localhost]
```

```
> Tx: Auto Playground lab4.public
```

```
6088 CREATE OR REPLACE PROCEDURE add_flight(
6089     arrival_airport_id,
6090     departing_gate,
6091     arriving_gate,
6092     airline_id,
6093     status,
6094     created_at,
6095     update_at
6096 )
6097 VALUES (
6098     flight_no p_flight_no,
6099     scheduled_departure p_scheduled_departure,
6100     scheduled_arrival p_scheduled_arrival,
6101     departure_airport_id p_departure_airport_id,
6102     arrival_airport_id p_arrival_airport_id,
6103     departing_gate p_departing_gate,
6104     arriving_gate p_arriving_gate,
6105     airline_id p_airline_id,
6106     status p_status,
6107     created_at CURRENT_DATE,
6108     update_at CURRENT_DATE
6109 );
6109
```

```
--
```

```
COMMENT ON PROCEDURE add_flight IS
--
```

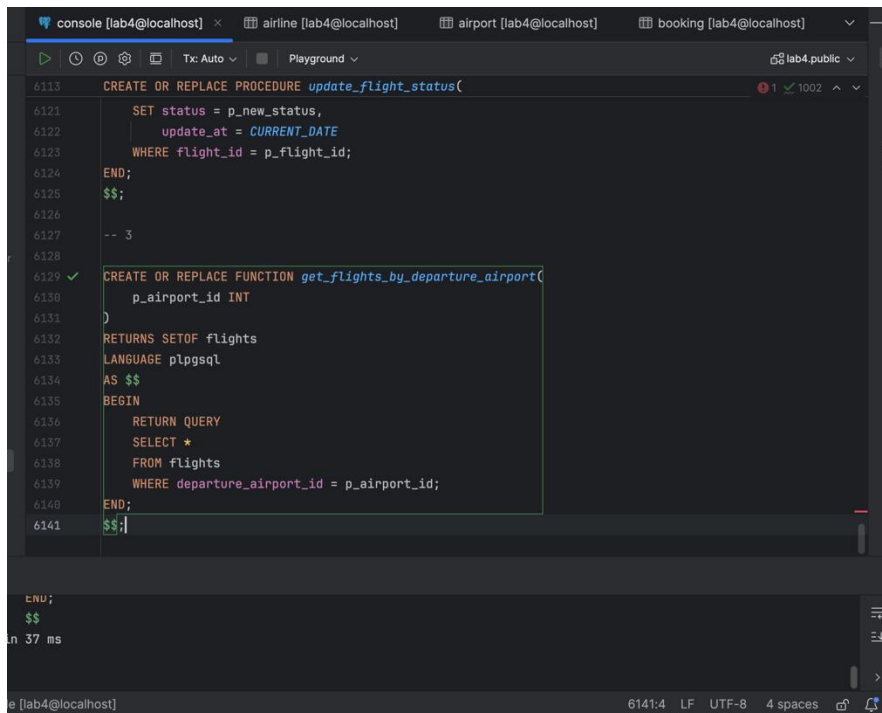
```
);
```

```
in 93 ms
```

```
le [lab4@localhost]
```

```
console [lab4@localhost] x airline [lab4@localhost] airport [lab4@localhost] booking [lab4@localhost]
--
CREATE OR REPLACE PROCEDURE add_flight(
    $$;
-- 2
CREATE OR REPLACE PROCEDURE update_flight_status(
    p_flight_id INT,
    p_new_status VARCHAR
)
LANGUAGE plpgsql
AS $$
BEGIN
    UPDATE flights
    SET status = p_new_status,
        update_at = CURRENT_DATE
    WHERE flight_id = p_flight_id;
END;
$$;
```

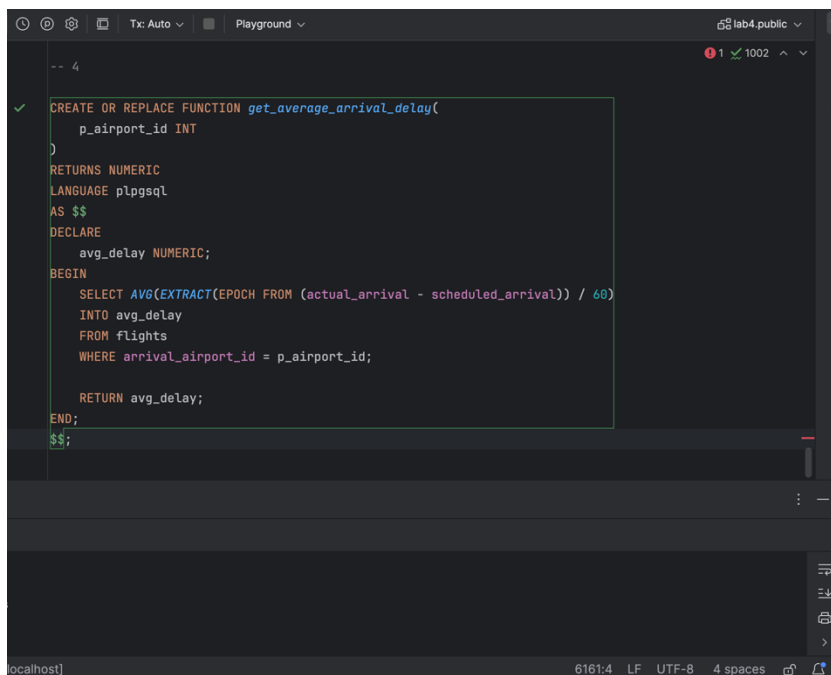
3.



The screenshot shows a SQL Playground interface with a dark theme. The top bar includes tabs for 'console [lab4@localhost]', 'airline [lab4@localhost]', 'airport [lab4@localhost]', and 'booking [lab4@localhost]'. The main editor area contains SQL code for creating a procedure and a function. The procedure 'update\_flight\_status' is defined with parameters 'p\_new\_status' and 'p\_flight\_id'. The function 'get\_flights\_by\_departure\_airport' is defined with parameter 'p\_airport\_id' and returns a set of flights. The code is syntax-highlighted, and a green box highlights the function definition. The bottom status bar shows '6141:4 LF UTF-8 4 spaces'.

```
6113 CREATE OR REPLACE PROCEDURE update_flight_status(  
6121     SET status = p_new_status,  
6122     update_at = CURRENT_DATE  
6123     WHERE flight_id = p_flight_id;  
6124 END;  
6125 $$;  
6126  
6127 -- 3  
6128  
6129 ✓ CREATE OR REPLACE FUNCTION get_flights_by_departure_airport(  
6130     p_airport_id INT  
6131 )  
6132 RETURNS SETOF flights  
6133 LANGUAGE plpgsql  
6134 AS $$  
6135 BEGIN  
6136     RETURN QUERY  
6137     SELECT *  
6138     FROM flights  
6139     WHERE departure_airport_id = p_airport_id;  
6140 END;  
6141 $$;
```

4.



The screenshot shows a SQL Playground interface with a dark theme. The top bar includes tabs for 'console [lab4@localhost]', 'airline [lab4@localhost]', 'airport [lab4@localhost]', and 'booking [lab4@localhost]'. The main editor area contains SQL code for creating a function 'get\_average\_arrival\_delay'. The function takes 'p\_airport\_id' as input and returns a numeric value representing the average delay. The code is syntax-highlighted, and a green box highlights the function definition. The bottom status bar shows '6161:4 LF UTF-8 4 spaces'.

```
-- 4  
✓ CREATE OR REPLACE FUNCTION get_average_arrival_delay(  
    p_airport_id INT  
)  
RETURNS NUMERIC  
LANGUAGE plpgsql  
AS $$  
DECLARE  
    avg_delay NUMERIC;  
BEGIN  
    SELECT AVG(EXTRACT(EPOCH FROM (actual_arrival - scheduled_arrival)) / 60)  
    INTO avg_delay  
    FROM flights  
    WHERE arrival_airport_id = p_airport_id;  
  
    RETURN avg_delay;  
END;  
$$;
```

5.

```
6171
6172 ✓ CREATE OR REPLACE FUNCTION get_passengers_by_flight_no(
6173     p_flight_no VARCHAR
6174 )
6175 RETURNS TABLE (
6176     passenger_id INT,
6177     first_name VARCHAR,
6178     last_name VARCHAR
6179 )
6180 LANGUAGE plpgsql
6181 AS $$
6182 BEGIN
6183     RETURN QUERY
6184     SELECT p.passenger_id,
6185            p.first_name,
6186            p.last_name
6187     FROM passengers p
6188     JOIN booking b
6189         ON p.passenger_id = b.passenger_id
        AND b.flight_id = p_flight_no;
END;
$
0 ms
b4@localhost] 6196:4 LF UTF-8 4 spaces
```

6.

```
6202 CREATE OR REPLACE FUNCTION get_top_passenger()
6210 AS $$
6211 BEGIN
6212     RETURN QUERY
6213     SELECT
6214         p.passenger_id,
6215         p.first_name,
6216         p.last_name,
6217         COUNT(bf.flight_id) AS flights_count
6218     FROM passengers p
6219     JOIN booking b
6220         ON p.passenger_id = b.passenger_id
6221     JOIN booking_flight bf
6222         ON b.booking_id = bf.booking_id
6223     GROUP BY
6224         p.passenger_id,
6225         p.first_name,
6226         p.last_name
6227     ORDER BY flights_count DESC
6228     LIMIT 1;
6229 END;
6230 $$;
$
4 ms
b4@localhost] 6230:4 LF UTF-8 4 spaces
```

7.

```
6231 -- 7
6232
6233
6234 ✓ CREATE OR REPLACE PROCEDURE get_flights_delayed_more_than_24h(
6235     INOUT ref refcursor
6236 )
6237 LANGUAGE plpgsql
6238 AS $$
6239 BEGIN
6240     OPEN ref FOR
6241     SELECT f.*
6242     FROM flights f
6243     WHERE f.actual_arrival IS NOT NULL
6244           AND f.scheduled_arrival IS NOT NULL
6245           AND (f.actual_arrival::timestamp - f.scheduled_arrival::timestamp) > INTERVAL '24 hours';
6246 END;
6247 $$;
```

14 ms

lab4@localhost 6247:4 LF UTF-8 4 spaces

8.

```
247
248
249 -- 8
250
251 ✓ CREATE OR REPLACE FUNCTION count_flights_by_airline()
252 RETURNS TABLE (
253     airline_id INT,
254     airline_name VARCHAR,
255     flights_count BIGINT
256 )
257 LANGUAGE plpgsql
258 AS $$
259 BEGIN
260     RETURN QUERY
261     SELECT
262         a.airline_id,
263         a.airline_name,
264         COUNT(f.flight_id) AS flights_count
265     FROM airline a
266     LEFT JOIN flights f
267         ON a.airline_id = f.airline_id
268     GROUP BY a.airline_id, a.airline_name
269     ORDER BY flights_count DESC;
```

ms

lab4@localhost 6271:4 LF UTF-8 4 spaces

9.

```
6251 CREATE OR REPLACE FUNCTION count_flights_by_airline()
6270 END;
6271 $$;
6272
6273 -- 9
6274
6275 ✓ CREATE OR REPLACE PROCEDURE get_average_ticket_price(
6276     p_flight_id INT,
6277     OUT avg_price NUMERIC
6278 )
6279 LANGUAGE plpgsql
6280 AS $$
6281 BEGIN
6282     SELECT AVG(b.price)
6283     INTO avg_price
6284     FROM booking b
6285     JOIN booking_flight bf
6286         ON b.booking_id = bf.booking_id
6287     WHERE bf.flight_id = p_flight_id;
6288 END;
6289 $$;
```

10.

```
92
93 -- 10
94
95 ✓ CREATE OR REPLACE PROCEDURE get_flight_with_highest_ticket_price(
96     INOUT ref refcursor
97 )
98 LANGUAGE plpgsql
99 AS $$
100 BEGIN
101     OPEN ref FOR
102     SELECT
103         f.flight_no,
104         f.departure_airport_id,
105         f.arrival_airport_id,
106         b.price AS highest_price
107     FROM flights f
108     JOIN booking_flight bf
109         ON f.flight_id = bf.flight_id
110     JOIN booking b
111         ON bf.booking_id = b.booking_id
112     ORDER BY b.price DESC
113     LIMIT 1;
```