

The screenshot shows the DBeaver IDE interface with a PostgreSQL connection active. The left sidebar displays the database structure under 'Servers (2)'. The main area has tabs for 'Query' and 'Query History', with 'Query' selected. A large code block represents a PostgreSQL function definition and a call to it. The code includes several parameters and constant values. The 'Data Output' tab at the bottom shows the result of executing the code, which is a single line: 'CALL'. Below that, a message indicates the query was successful: 'Query returned successfully in 2 secs 265 msec.' The status bar at the bottom shows 'Total rows: 0' and 'Query complete 00:00:02.265'.

```
31      )
32      VALUES (
33          p_flight_no,
34          p_scheduled_departure,
35          p_scheduled_arrival,
36          p_departure_airport_id,
37          p_arrival_airport_id,
38          p_departing_gate,
39          p_arriving_gate,
40          p_airline_id,
41          p_status,
42          p_actual_departure,
43          p_actual_arrival,
44          CURRENT_DATE,
45          CURRENT_DATE
46      );
47 END;
48 $$;
49
50 CALL insert_new_flight(
51     'KC123',
52     '2025-12-01',
53     '2025-12-01',
54     1,
55     2,
56     'A5',
57     'B3',
58     1,
59     'Scheduled',
60     NULL,
61     NULL
62 );
63
```

Data Output Messages Notifications

CALL

Query returned successfully in 2 secs 265 msec.

Total rows: 0 | Query complete 00:00:02.265 | LF | Ln 63, Col 1

airport_database/postgres@PostgreSQL 17*

Servers (2)
PostgreSQL 17
Databases (4)
Login/Group Roles
Tablespaces
PostgreSQL 18

airport_database/postgres@PostgreSQL 17

No limit

Query History

```
1 CREATE OR REPLACE PROCEDURE update_flight_status(
2     p_flight_id    INT,
3     p_new_status   VARCHAR
4 )
5 LANGUAGE plpgsql
6 AS $$
7 BEGIN
8     UPDATE flights
9     SET status      = p_new_status,
10        update_at  = CURRENT_DATE
11     WHERE flight_id = p_flight_id;
12
13 IF NOT FOUND THEN
14     RAISE NOTICE 'Flight with id % not found.', p_flight_id;
15 END IF;
16
17 $$;
18
19 CALL update_flight_status(1, 'Delayed');
20
```

Data Output Messages Notifications

CALL

Query returned successfully in 1 secs 676 msec.

Total rows: Query complete 00:00:01.676

✓ Query returned successfully in 1 secs 676 msec. ✘

LF Ln 20, Col 1

airport_database/postgres@PostgreSQL 17* X

Servers (2) airport_database/postgres@PostgreSQL 17

Databases (4) No limit

Login/Group Roles Tablespaces

PostgreSQL 18

Query History

Execute script F5

```
1 CREATE OR REPLACE FUNCTION get_flights_from_airport(
2     p_departure_airport_id INT
3 )
4 RETURNS TABLE (
5     flight_id             INT,
6     flight_no              VARCHAR,
7     scheduled_departure   DATE,
8     scheduled_arrival     DATE,
9     departure_airport_id  INT,
10    arrival_airport_id    INT,
11    airline_id             INT,
12    status                 VARCHAR
13 )
14 LANGUAGE sql
15 AS $$
```

SELECT

```
17     flight_id,
18     flight_no,
19     scheduled_departure,
20     scheduled_arrival,
21     departure_airport_id,
22     arrival_airport_id,
23     airline_id,
24     status
25 FROM flights
26 WHERE departure_airport_id = p_departure_airport_id
27 ORDER BY scheduled_departure;
28 $$;
```

30 SELECT * FROM get_flights_from_airport(1);

31

Data Output Messages Notifications

Showing rows: 1 to 50 Page No: 1 of 1

	flight_id	flight_no	scheduled_departure	scheduled_arrival	departure_airport_id	arrival_airport_id	airline_id	status
1	168	IL-D	2023-03-21	2023-05-24	1	18	12	Delayed

Total rows: 50 Query complete 00:00:00.577

LF Ln 31, Col 1

airport_database/postgres@PostgreSQL 17* X

Servers (2) airport_database/postgres@PostgreSQL 17

Databases (4) No limit

Login/Group Roles Tablespaces

PostgreSQL 18

Query History

```
1 CREATE OR REPLACE FUNCTION avg_arrival_delay(
2     p_arrival_airport_id INT
3 )
4 RETURNS NUMERIC(6,2)
5 LANGUAGE sql
6 AS $$
7     SELECT AVG(actual_arrival - scheduled_arrival)
8     FROM flights
9     WHERE arrival_airport_id = p_arrival_airport_id
10    AND actual_arrival IS NOT NULL;
11 $$;
12
13 SELECT avg_arrival_delay(2) AS average_delay_days;
14
```

Data Output Messages Notifications

average_delay_days numeric

	average_delay_days	numeric
1	22.750000000000000	

Showing rows: 1 to 1 | Page No: 1 of 1 | < < > >>

Total rows: 1 Query complete 00:00:01.426

✓ Successfully run. Total query runtime: 1 secs 426 msec. 1 rows affected.

LF | Ln 14, Col 1

airport_database/postgres@PostgreSQL 17* X

Servers (2) airport_database/postgres@PostgreSQL 17

Databases (4) No limit

Login/Group Roles E Tablespaces

PostgreSQL 18

Query History

```
1 CREATE OR REPLACE FUNCTION get_passengers_by_flight(
2     p_flight_no VARCHAR
3 )
4 RETURNS TABLE (
5     passenger_id      INT,
6     first_name        VARCHAR,
7     last_name         VARCHAR,
8     date_of_birth    DATE,
9     gender            VARCHAR,
10    passport_number  VARCHAR
11 )
12 LANGUAGE sql
13 AS $$
```

14 SELECT

```
15     p.passenger_id,
16     p.first_name,
17     p.last_name,
18     p.date_of_birth,
19     p.gender,
20     p.passport_number
21     FROM passengers p
22     JOIN booking b      ON p.passenger_id = b.passenger_id
23     JOIN booking_flight bf ON b.booking_id  = bf.booking_id
24     JOIN flights f      ON bf.flight_id   = f.flight_id
25     WHERE f.flight_no = p_flight_no
26     ORDER BY p.last_name, p.first_name;
27 $$;
```

28

```
29     SELECT * FROM get_passengers_by_flight('KC123');
```

30

Data Output Messages Notifications

SQL

	passenger_id	first_name	last_name	date_of_birth	gender	passport_number
	integer	character varying	character varying	date	character varying	character varying

Total rows: 0 Query complete 00:00:00.802

Successfully run. Total query runtime: 802 msec. 0 rows affected.

LF | Ln 30, Col 1

airport_database/postgres@PostgreSQL 17* X

Servers (2) PostgreSQL 17 Databases (4) Login/Group Roles Tablespaces PostgreSQL 18

airport_database/postgres@PostgreSQL 17

No limit

Execute script F5

Query History

```
1 CREATE OR REPLACE PROCEDURE get_top_passenger_no_cursor()
2 LANGUAGE plpgsql
3 AS $$
4 DECLARE
5     top_passenger RECORD;
6 BEGIN
7     SELECT
8         p.passenger_id,
9         p.first_name,
10        p.last_name,
11        COUNT(bf.flight_id) AS total_flights
12    INTO top_passenger
13    FROM passengers p
14    JOIN booking b      ON p.passenger_id = b.passenger_id
15    JOIN booking_flight bf ON b.booking_id = bf.booking_id
16    GROUP BY p.passenger_id, p.first_name, p.last_name
17    ORDER BY total_flights DESC
18    LIMIT 1;
19
20    RAISE NOTICE
21        'Passenger ID: %, Name: % %, Total Flights: %',
22        top_passenger.passenger_id,
23        top_passenger.first_name,
24        top_passenger.last_name,
25        top_passenger.total_flights;
26 END;
27 $$;
28
29 CALL get_top_passenger_no_cursor();
30
```

Data Output Messages Notifications

NOTICE: Passenger ID: 68, Name: Sheela Roux, Total Flights: 6
CALL
Query returned successfully in 704 msec.

Total rows: Query complete 00:00:00.704 ✓ Query returned successfully in 704 msec. LF Ln 30, Col 1

airport_database/postgres@PostgreSQL 17* X

Servers (2) airport_database/postgres@PostgreSQL 17

Databases (4) No limit

Query History

```

2 LANGUAGE plpgsql
3 AS $$ 
4 BEGIN
5     CREATE TEMP TABLE IF NOT EXISTS temp_delayed_flights AS
6         SELECT
7             flight_id,
8             flight_no,
9             scheduled_departure,
10            actual_departure,
11            scheduled_arrival,
12            actual_arrival,
13            departure_airport_id,
14            arrival_airport_id,
15            airline_id,
16            status,
17            (actual_departure - scheduled_departure) AS delay_days
18        FROM flights
19        WHERE actual_departure IS NOT NULL
20            AND scheduled_departure IS NOT NULL
21            -- задержка больше 1 дня (24 часов)
22            AND (actual_departure - scheduled_departure) > 1
23        ORDER BY delay_days DESC;
24
25        RAISE NOTICE 'Delayed flights over 24h inserted into temp_delayed_flights';
26    END;
27 $$;
28
29 CALL get_flights_delayed_over_24h_no_cursor();
30
31 SELECT * FROM temp_delayed_flights;

```

Data Output Messages Notifications

Showing rows: 1 to 492 Page No: 1 of 1

	flight_id integer	flight_no character varying (50)	scheduled_departure date	actual_departure date	scheduled_arrival date	actual_arrival date	departure_airport_id integer	arrival_airport_id integer	airline_id integer	status character varying (50)
1	952	US-AZ	2023-03-26	2024-03-11	2023-12-19	2023-08-13	14	14	43	Boarding
2	711	CA-NU	2023-03-23	2024-02-29	2023-12-12	2023-06-18	12	3	33	Boarding

Total rows: 492 Query complete 00:00:00.463

LF Ln 32, Col 1

airport_database/postgres@PostgreSQL 17* X

Servers (2) PostgreSQL 17 Databases (4) Login/Group Roles Tablespaces PostgreSQL 18

airport_database/postgres@PostgreSQL 17 No limit

Query History

```
1 CREATE OR REPLACE FUNCTION count_flights_per_airline()
2 RETURNS TABLE (
3     airline_id      INT,
4     airline_name    VARCHAR,
5     total_flights  INT
6 )
7 LANGUAGE sql
8 AS $$
```

9 SELECT

```
10        a.airline_id,
11        a.airline_name,
12        COUNT(f.flight_id) AS total_flights
13     FROM airline a
14     LEFT JOIN flights f ON a.airline_id = f.airline_id
15     GROUP BY a.airline_id, a.airline_name
16     ORDER BY total_flights DESC;
17 $$;
```

```
18
19     SELECT * FROM count_flights_per_airline();
20
```

Data Output Messages Notifications

Showing rows: 1 to 50 Page No: 1 of 1

	airline_id integer	airline_name character varying	total_flights integer
1	1	IPC	33
2	43	KMA	30
3	29	NHT	29
4	19	CSC	28
5	47	DNU	27
6	16	DUC	25
7	23	SPI	25
8	26	YHB	25
9	8	QIG	25

Total rows: 50 Query complete 00:00:03.215 LF Ln 20, Col 1

The screenshot shows the DBeaver IDE interface for managing a PostgreSQL database. The left sidebar lists servers and databases, with 'PostgreSQL 17' selected. The main area displays a SQL query window containing a stored procedure definition:

```
1 CREATE OR REPLACE PROCEDURE avg_ticket_price_for_flight(
2     p_flight_id INT
3 )
4 LANGUAGE plpgsql
5 AS $$
6 DECLARE
7     avg_price NUMERIC(10,2);
8 BEGIN
9     SELECT AVG(price) INTO avg_price
10    FROM booking b
11   JOIN booking_flight bf ON b.booking_id = bf.booking_id
12  WHERE bf.flight_id = p_flight_id;
13
14    IF avg_price IS NULL THEN
15        RAISE NOTICE 'No bookings found for flight ID %', p_flight_id;
16    ELSE
17        RAISE NOTICE 'Average ticket price for flight ID % is %',
18                  p_flight_id, avg_price;
19    END IF;
20 END;
21 $$;
22
23 CALL avg_ticket_price_for_flight(1);
24
```

An 'Execute script' button with the F5 key is highlighted. Below the code, the 'Messages' tab shows the execution results:

NOTICE: No bookings found for flight ID 1
CALL

Query returned successfully in 1 secs 188 msec.

A green status bar at the bottom right indicates: ✓ Query returned successfully in 1 secs 188 msec. X

Bottom status bar: Total rows: | Query complete 00:00:01.188 | LF | Ln 24, Col 1

The screenshot shows the pgAdmin 4 interface for a PostgreSQL database. The left sidebar lists servers and databases, with 'PostgreSQL 17' selected. The main window displays a query editor containing a PL/pgSQL procedure named 'get_most_expensive_flight'. The code uses multiple joins to find the most expensive flight and handles the result with RAISE NOTICE statements. The bottom pane shows the output of the query execution.

```
CREATE OR REPLACE PROCEDURE get_most_expensive_flight()
LANGUAGE plpgsql
AS $$

DECLARE
    top_flight RECORD;
BEGIN
    SELECT
        f.flight_no,
        dep.airport_name AS departure_airport,
        arr.airport_name AS arrival_airport,
        b.price          AS ticket_price
    INTO top_flight
    FROM flights f
    JOIN booking_flight bf ON f.flight_id = bf.flight_id
    JOIN booking b      ON bf.booking_id = b.booking_id
    JOIN airport dep   ON f.departure_airport_id = dep.airport_id
    JOIN airport arr   ON f.arrival_airport_id = arr.airport_id
    ORDER BY b.price DESC
    LIMIT 1;

    IF top_flight IS NULL THEN
        RAISE NOTICE 'No flights with bookings found.';
    ELSE
        RAISE NOTICE
            'Flight: %, Departure: %, Arrival: %, Ticket Price: %',
            top_flight.flight_no,
            top_flight.departure_airport,
            top_flight.arrival_airport,
            top_flight.ticket_price;
    END IF;
END;
$$;

CALL get_most_expensive_flight();
```

Data Output Messages Notifications

NOTICE: Flight: NP-SA, Departure: Henri Coandă International Airport, Arrival: Armidale Airport, Ticket Price: 9977.57
CALL

Total rows: Query complete 00:00:00.609 LF Ln 35, Col 1