

pgAdmin 4

File Object Tools Edit View Window Help

Welcome × postgres/postgres@PostgreSQL 17\* × postgres/postgres@PostgreSQL 17

No limit ▾ E II ✓

Query History

Query Scratch Pad

```
1 CREATE OR REPLACE FUNCTION insert_flight_and_return_id (
2     p_sch_departure_time TIMESTAMP, p_sch_arrival_time TIMESTAMP,
3     p_departing_airport_id INTEGER, p_arriving_airport_id INTEGER,
4     p_departing_gate VARCHAR(50), p_arriving_gate VARCHAR(50),
5     p_airline_id INTEGER,
6     p_act_departure_time TIMESTAMP, p_act_arrival_time TIMESTAMP
7 )
8 RETURNS INTEGER
9 LANGUAGE sql
10 AS $$
11 INSERT INTO flights (sch_departure_time, sch_arrival_time, departing_airport_id, arriving_airport_id, departing_gate, arriving_gate, airline_id, act_departure_time, act_arrival_time)
12 VALUES (
13     p_sch_departure_time, p_sch_arrival_time,
14     p_departing_airport_id, p_arriving_airport_id,
15     p_departing_gate, p_arriving_gate,
16     p_airline_id,
17     p_act_departure_time, p_act_arrival_time
18 )
19 RETURNING flight_id;
20 $$;
21 SELECT insert_flight_and_return_id(
22     '2025-12-06 15:00:00', -- p_sch_departure_time
23     '2025-12-06 17:45:00', -- p_sch_arrival_time
24     305, -- p_departing_airport_id
25     401, -- p_arriving_airport_id
26     'C01', -- p_departing_gate
27     'D10', -- p_arriving_gate
28     8, -- p_airline_id
29     '2025-12-06 15:10:00', -- p_act_departure_time
30     '2025-12-06 17:50:00' -- p_act_arrival_time
31 );
```

Data Output Messages Notifications

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

	Insert_flight_and_return_id
1	209

Total rows: 1 Query complete 00:00:00.147 ✓ Successfully run. Total query runtime: 147 msec. 1 rows affected. CRLF Ln 31, Col 3

1. Create a stored procedure to insert a new flight into the flights table.

pgAdmin 4

File Object Tools Edit View Window Help

Welcome × postgres/postgres@PostgreSQL 17\* ×

postgres/postgres@PostgreSQL 17

No limit

Query History

Execute script F5

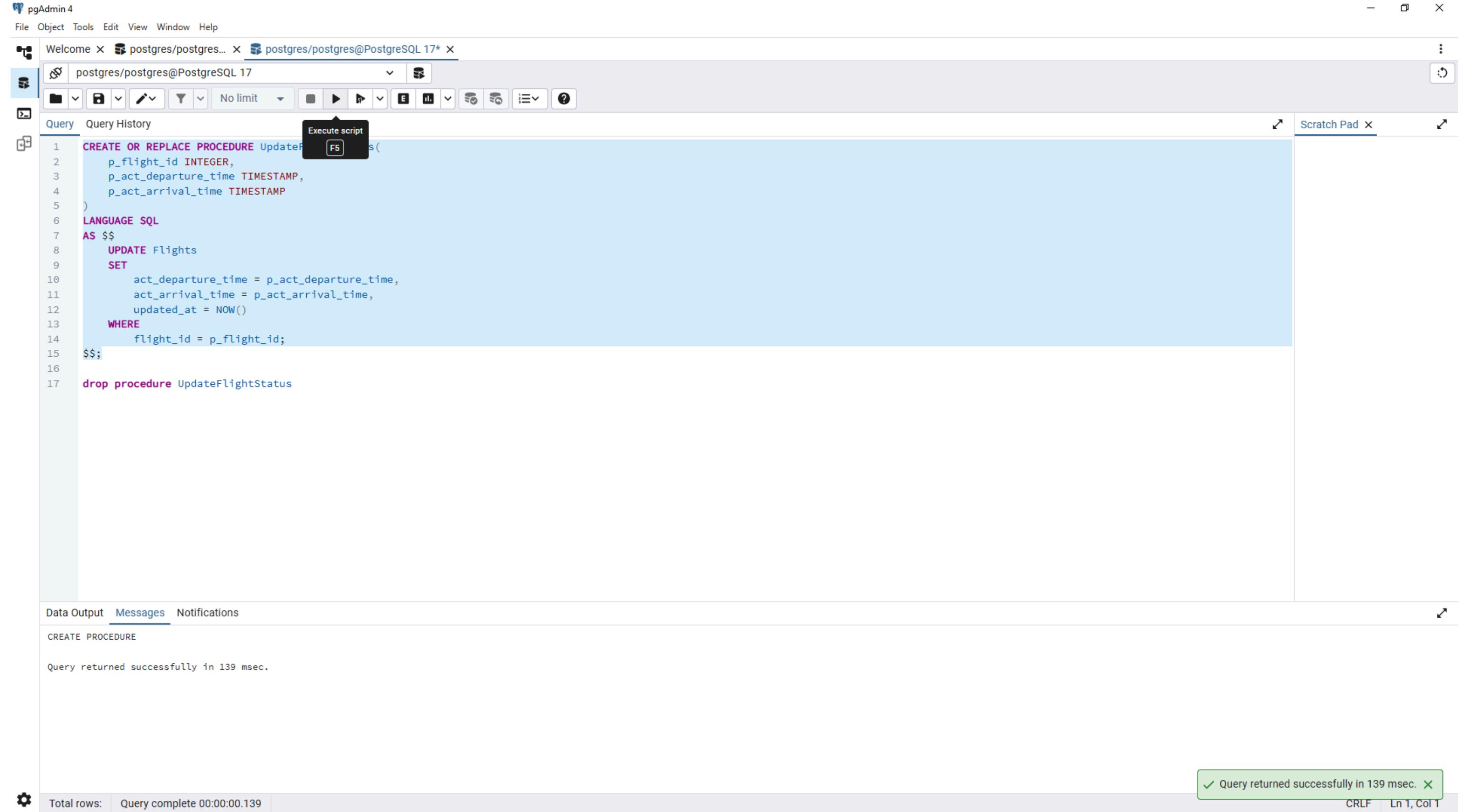
```
1 CREATE OR REPLACE PROCEDURE UpdateFlightStatus(
2     p_flight_id INTEGER,
3     p_act_departure_time TIMESTAMP,
4     p_act_arrival_time TIMESTAMP
5 )
6 LANGUAGE SQL
7 AS $$
8     UPDATE Flights
9     SET
10        act_departure_time = p_act_departure_time,
11        act_arrival_time = p_act_arrival_time,
12        updated_at = NOW()
13     WHERE
14        flight_id = p_flight_id;
15 $$;
16
17 drop procedure UpdateFlightStatus
```

Data Output Messages Notifications

CREATE PROCEDURE

Query returned successfully in 139 msec.

Total rows: Query complete 00:00:00.139 ✓ CRLF Ln 1, Col 1



2. Create a stored procedure to update the status of a flight.

pgAdmin 4

File Object Tools Edit View Window Help

Welcome × postgres/postgres@PostgreSQL 17\* ×

postgres/postgres@PostgreSQL 17 No limit

Query History

Query Scratch Pad

```
1 CREATE OR REPLACE FUNCTION GetFlightsByDepartureAirport(
2     p_departing_airport_id INTEGER
3 )
4 RETURNS TABLE (
5     flight_id INTEGER,
6     departure_time TIMESTAMP,
7     arrival_time TIMESTAMP,
8     arriving_airport_id INTEGER,
9     departing_gate TEXT
10 )
11 LANGUAGE plpgsql
12 AS $$
```

BEGIN

```
14     RETURN QUERY
15     SELECT
16         f.flight_id,
17         f.sch_departure_time,
18         f.sch_arrival_time,
19         f.arriving_airport_id,
20         f.departing_gate
21     FROM
22         Flights f
23     WHERE
24         f.departing_airport_id = p_departing_airport_id;
25 END;
$$;
```

```
28 SELECT * FROM GetFlightsByDepartureAirport(228);
29
30 drop FUNCTION GetFlightsByDepartureAirport
```

Data Output Messages Notifications

Showing rows: 1 to 200 Page No: 1 of 1

	flight_id	departure_time	arrival_time	arriving_airport_id	departing_gate
1	1	2025-09-28 12:26:18.609531	2025-09-28 10:01:06.71777	203	G34
2	2	2025-09-30 15:42:27.206758	2025-10-05 00:58:30.008069	203	G60
3	3	2025-10-06 11:04:47.171199	2025-10-10 17:38:00.626649	203	G36

Total rows: 200 Query complete 00:00:00.165

Successfully run. Total query runtime: 165 msec. 200 rows affected.

CRLF Ln 28, Col 49

3. Create a stored procedure that returns a list of flights departing from a specific airport.

pgAdmin 4

File Object Tools Edit View Window Help

Welcome × postgres/postgres@PostgreSQL 17\* ×

postgres/postgres@PostgreSQL 17

No limit

Query History

Scratch Pad

```
1 CREATE OR REPLACE FUNCTION CalculateAverageArrivalDelay(p_arriving_airport_id INTEGER)
2 RETURNS INTERVAL
3 LANGUAGE plpgsql
4 AS $$
5 DECLARE avg_delay INTERVAL;
6 BEGIN
7     SELECT AVG(f.act_arrival_time - f.sch_arrival_time) INTO avg_delay
8     FROM Flights f
9     WHERE f.arriving_airport_id = p_arriving_airport_id
10        AND f.act_arrival_time IS NOT NULL
11        AND f.act_arrival_time > f.sch_arrival_time;
12     RETURN avg_delay;
13 END;
14 $$;
15 SELECT * FROM CalculateAverageArrivalDelay(2);
16
17 drop FUNCTION CalculateAverageArrivalDelay
```

Data Output Messages Notifications

calculateaveragearrivaldelay interval

1	[null]

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

Total rows: 1 Query complete 00:00:00.109

Successfully run. Total query runtime: 109 msec. 1 rows affected.

CRLF Ln 15, Col 1

The screenshot shows the pgAdmin 4 interface with a query editor and a results pane. The query editor contains SQL code to create a function named 'CalculateAverageArrivalDelay' that takes an integer parameter for the arriving airport ID and returns an interval representing the average delay. The function uses a WHERE clause to filter flights where the actual arrival time is greater than the scheduled arrival time and is not null. The results pane shows the function was successfully created and executed, returning a single row with a null value for the average delay.

4. Create a function to calculate the average delay time of flights arriving at a specific airport.

pgAdmin 4

File Object Tools Edit View Window Help

Welcome × postgres/postgres@PostgreSQL 17\* ×

postgres/postgres@PostgreSQL 17

No limit

Query History

Query Scratch Pad

```
1 CREATE OR REPLACE FUNCTION ListPassengersForFlight(
2     p_flight_id INTEGER
3 )
4 RETURNS TABLE (
5     passenger_id INTEGER,
6     first_name VARCHAR(50), last_name VARCHAR(50),
7     passport_number VARCHAR(20),
8     booking_id INTEGER
9 )
10 LANGUAGE plpgsql
11 AS $$
12 BEGIN
13     RETURN QUERY
14     SELECT
15         p.passenger_id,
16         p.first_name,
17         p.last_name,
18         p.passport_number,
19         b.booking_id
20     FROM Booking b
21     JOIN Passengers p ON b.passenger_id = p.passenger_id
22     WHERE b.flight_id = p_flight_id
23     ORDER BY p.last_name, p.first_name;
24 END;
25 $$;
26 SELECT * FROM ListPassengersForFlight(136);
27 drop FUNCTION ListPassengersForFlight
```

Data Output Messages Notifications

Showing rows: 1 to 129 Page No: 1 of 1

	passenger_id	first_name	last_name	passport_number	booking_id
1	70	FN_f0ab27	LN_f16938	79FE10BA	100

Total rows: 129 Query complete 00:00:00.092 ✓ Successfully run. Total query runtime: 92 msec. 129 rows affected. CRLF Ln 25, Col 4

5. Create a stored procedure that lists all passengers for a given flight number.

pgAdmin 4

File Object Tools Edit View Window Help

Welcome × postgres/postgres@PostgreSQL 17\* ×

postgres/postgres@PostgreSQL 17

No limit

Query History

Query Scratch Pad

```
1 CREATE OR REPLACE PROCEDURE FindMostFrequentPassenger(
2     OUT p_passenger_id INTEGER,
3     OUT p_first_name VARCHAR,
4     OUT p_last_name VARCHAR,
5     OUT p_total_flights_booked BIGINT
6 )
7 LANGUAGE plpgsql
8 AS $$
9 BEGIN
10    SELECT
11        p.passenger_id,
12        p.first_name, p.last_name,
13        COUNT(b.booking_id) INTO
14        p_passenger_id,
15        p_first_name, p_last_name,
16        p_total_flights_booked
17    FROM Passengers p
18    JOIN Booking b ON p.passenger_id = b.passenger_id
19    GROUP BY p.passenger_id, p.first_name, p.last_name
20    ORDER BY COUNT(b.booking_id) DESC
21    LIMIT 1;
22
23 END;
24 $$;
25 CALL FindMostFrequentPassenger(NULL, NULL, NULL, NULL);
drop PROCEDURE FindMostFrequentPassenger
```

Data Output Messages Notifications

SQL

	p_passenger_id	p_first_name	p_last_name	p_total_flights_booked
1	70	FN_f0ab27	LN_f16938	129

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

Total rows: 1 Query complete 00:00:00.107 ✓ Successfully run. Total query runtime: 107 msec. 1 rows affected.

CRLF Ln 23, Col 4

The screenshot shows the pgAdmin 4 interface with a query editor window. The query editor contains a PostgreSQL stored procedure named 'FindMostFrequentPassenger'. The procedure uses the 'plpgsql' language and takes four OUT parameters: 'p\_passenger\_id' (integer), 'p\_first\_name' (character varying), 'p\_last\_name' (character varying), and 'p\_total\_flights\_booked' (bigint). It performs a SELECT statement that joins the 'Passengers' table with the 'Booking' table, groups by passenger ID and names, and orders by the count of bookings in descending order, limiting the result to one row. The procedure is then called with four NULL arguments, and finally, it is dropped. Below the query editor, the 'Data Output' tab is active, showing a single row of results from the executed query. The row contains the passenger ID (70), first name (FN\_f0ab27), last name (LN\_f16938), and total flights booked (129). A success message at the bottom right indicates the query was run successfully with a runtime of 107 msec and 1 row affected.

6. Create a stored procedure to find the passenger who has taken the greatest number of flights.

pgAdmin 4

File Object Tools Edit View Window Help

Welcome × postgres/postgres@PostgreSQL 17\* ×

postgres/postgres@PostgreSQL 17

No limit ▾ E ↻

Query History

Query Scratch Pad

```
1 CREATE OR REPLACE FUNCTION FindFlightsDelayedMoreThan24h()
2 RETURNS TABLE (
3     flight_id INTEGER,
4     scheduled_departure TIMESTAMP,
5     actual_departure TIMESTAMP,
6     delay_duration INTERVAL
7 )
8 LANGUAGE plpgsql
9 AS $$
```

BEGIN

```
11     RETURN QUERY
12     SELECT
13         f.flight_id,
14         f.sch_departure_time,
15         f.act_departure_time,
16         (f.act_departure_time - f.sch_departure_time) AS delay_duration
17     FROM Flights f
18     WHERE f.act_departure_time > f.sch_departure_time + INTERVAL '24 hours' AND f.act_departure_time IS NOT NULL;
19 END;
20 $$;
21 SELECT * FROM FindFlightsDelayedMoreThan24h();
22 drop FUNCTION FindFlightsDelayedMoreThan24h
```

Data Output Messages Notifications

Showing rows: 1 to 96 Page No: 1 of 1

	flight_id	scheduled_departure	actual_departure	delay_duration
	integer	timestamp without time zone	timestamp without time zone	interval
1	1	2025-09-28 12:26:18.609531	2025-10-11 14:27:00.713001	13 days 02:00:42.10347
2	2	2025-09-30 15:42:27.206758	2025-10-10 10:19:04.043316	9 days 18:36:36.836558
3	3	2025-10-06 11:04:47.171199	2025-10-16 02:05:45.025777	9 days 15:00:57.854578
4	4	2025-10-16 20:14:59.27743	2025-10-22 17:26:14.209645	5 days 21:11:14.932215
5	6	2025-10-10 07:45:10.725845	2025-10-20 00:54:33.408785	9 days 17:09:22.68294
6	8	2025-09-23 21:54:22.681495	2025-09-28 08:16:06.210953	4 days 10:21:43.529458
7	10	2025-09-28 20:18:12.483113	2025-10-05 02:58:08.945964	6 days 06:39:56.462851
8	12	2025-10-07 22:14:37.556397	2025-10-11 16:50:40.932978	3 days 18:36:03.376581

✓ Successfully run. Total query runtime: 54 msec. 96 rows affected.

Total rows: 96 Query complete 00:00:00.054 CRLF Ln 20, Col 4

7. Create a stored procedure to find all flights that are delayed by more than 24 hours.

pgAdmin 4

File Object Tools Edit View Window Help

Welcome × postgres/postgres@PostgreSQL 17\* ×

postgres/postgres@PostgreSQL 17

No limit

Query History

Query Scratch Pad

```
1 CREATE OR REPLACE FUNCTION CountFlightsByAirline()
2   RETURNS TABLE (
3     airline_id INTEGER,
4     flight_count BIGINT
5   )
6 LANGUAGE plpgsql
7 AS $$
```

BEGIN

```
9   RETURN QUERY
10  SELECT F.airline_id, COUNT(F.flight_id) AS flight_count
11  FROM Flights F
12  GROUP BY F.airline_id
13  ORDER BY flight_count DESC;
```

END;

```
15 $$;
16 SELECT * FROM CountFlightsByAirline();
17 drop FUNCTION CountFlightsByAirline
```

Data Output Messages Notifications

Showing rows: 1 to 2 Page No: 1 of 1

	airline_id	flight_count
1	150	200
2	8	5

Total rows: 2 Query complete 00:00:00.075

Successfully run. Total query runtime: 75 msec. 2 rows affected.

CRLF Ln 15, Col 4

8. Create a function that counts the number of flights for each airline.

pgAdmin 4

File Object Tools Edit View Window Help

Welcome × postgres/postgres@PostgreSQL 17\* ×

postgres/postgres@PostgreSQL 17

No limit

Query History

Query Scratch Pad

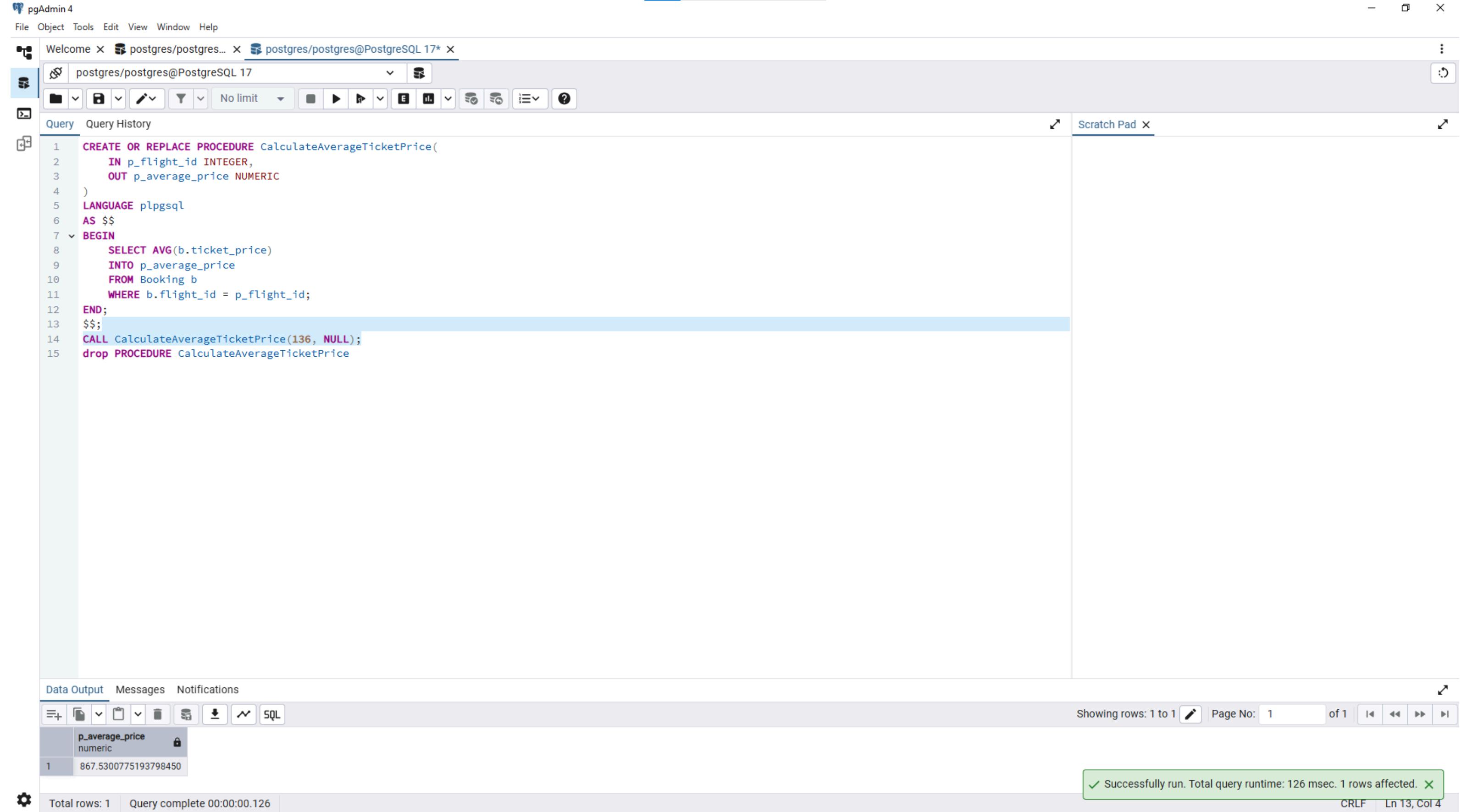
```
1 CREATE OR REPLACE PROCEDURE calculateAverageTicketPrice(
2     IN p_flight_id INTEGER,
3     OUT p_average_price NUMERIC
4 )
5 LANGUAGE plpgsql
6 AS $$
7 BEGIN
8     SELECT AVG(b.ticket_price)
9     INTO p_average_price
10    FROM Booking b
11   WHERE b.flight_id = p_flight_id;
12
13 $$;
14 CALL calculateAverageTicketPrice(136, NULL);
15 drop PROCEDURE calculateAverageTicketPrice
```

Data Output Messages Notifications

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

	p_average_price
1	867.5300775193798450

Total rows: 1 Query complete 00:00:00.126 ✓ Successfully run. Total query runtime: 126 msec. 1 rows affected. CRLF Ln 13, Col 4



9. Create a stored procedure to calculate the average ticket price for a specific flight.

pgAdmin 4

File Object Tools Edit View Window Help

Welcome × postgres/postgres@PostgreSQL 17\* ×

postgres/postgres@PostgreSQL 17

No limit

Query History

Query Scratch Pad

```
1 CREATE OR REPLACE FUNCTION FindMostExpensiveFlight()
2 RETURNS TABLE (
3     flight_id INTEGER,
4     departure_airport_id INTEGER,
5     arrival_airport_id INTEGER,
6     highest_ticket_price NUMERIC
7 )
8 LANGUAGE plpgsql
9 AS $$
```

BEGIN

```
11     RETURN QUERY
12     SELECT
13         f.flight_id,
14         f.departing_airport_id,
15         f.arriving_airport_id,
16         b.ticket_price AS highest_ticket_price
17     FROM Booking b
18     JOIN Flights f ON b.flight_id = f.flight_id
19     ORDER BY b.ticket_price DESC
20     LIMIT 1;
21 END;
22 $$;
```

```
23 SELECT * FROM FindMostExpensiveFlight();
24 drop FUNCTION FindMostExpensiveFlight
```

Data Output Messages Notifications

flight\_id departure\_airport\_id arrival\_airport\_id highest\_ticket\_price

	flight_id	departure_airport_id	arrival_airport_id	highest_ticket_price
1	136	228	203	1185.78

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

Successfully run. Total query runtime: 96 msec. 1 rows affected.

Total rows: 1 Query complete 00:00:00.096 CRLF Ln 23, Col 1

10. Create a stored procedure to find the flight with the highest ticket price. The procedure should return the flight number, the departure and arrival airports, and the ticket price for the most expensive flight.