

1)

The screenshot shows the pgAdmin 4 interface. The query editor contains the following SQL code:

```

1 CREATE OR REPLACE PROCEDURE add_flight(
2   p_flight_id INT,
3   p_flight_no VARCHAR,
4   p_scheduled_departure DATE,
5   p_scheduled_arrival DATE,
6   p_departure_airport_id INT,
7   p_arrival_airport_id INT,
8   p_departing_gate VARCHAR,
9   p_arriving_gate VARCHAR,
10  p_airline_id INT,
11  p_status VARCHAR
12 )
13 LANGUAGE plpgsql
14 AS $$
15 BEGIN
16   INSERT INTO flights (
17     flight_id,
18     flight_no,
19     scheduled_departure,
20     scheduled_arrival,
21     departure_airport_id,
22     arrival_airport_id,
    
```

The Data Output tab shows the results of the query, displaying a table with 12 columns and 5 rows of data. The status of the flights is 'Boarding' or 'Delayed'.

id	scheduled_departure	scheduled_arrival	departure_airport_id	arrival_airport_id	departing_gate	arriving_gate	airline_id	status	actual_departure	actual_arrival	created_at	update_at
970	3-06-07	2023-06-12	13	12	0239	70	9	Boarding	2023-12-20	2023-03-18	2024-02-07	2023-09-17
971	3-04-19	2023-06-05	9	7	[null]	328	49	Boarding	2023-09-30	2023-12-11	2023-05-14	2024-01-28
972	3-11-25	2023-12-30	20	6	3292	249	32	Delayed	2024-03-10	2023-03-27	2023-07-04	2023-05-03
973	3-04-23	2024-01-18	19	2	241	22	13	Boarding	2023-05-25	2024-01-31	2024-03-16	2023-06-06
974	3-09-01	2023-03-23	7	15	5640	4804	6	Boarding	2023-04-12	2023-10-11	2023-06-18	2023-09-22
975	3-08-25	2023-05-19	17	13	55	676	32	Delayed	2023-07-29	2024-01-15	2024-02-29	2023-10-15

Total rows: 1001 Query complete 00:00:00.191

2)

The screenshot shows the pgAdmin 4 interface. The query editor contains the following SQL code:

```

1 create or replace procedure status_update(
2   p_flight_id INT,
3   p_status VARCHAR
4 )
5 language plpgsql
6 as $$
7 begin
8   update flights
9   set status = p_status
10  where flight_id = p_flight_id;
11 end;
12 $$;
13
14 call status_update(2, 'Delayed');
15
16
17 select * from flights
    
```

The Data Output tab shows the results of the query, displaying a table with 12 columns and 5 rows of data. The status of the flights is 'Boarding' or 'Delayed'.

id	scheduled_departure	scheduled_arrival	departure_airport_id	arrival_airport_id	departing_gate	arriving_gate	airline_id	status	actual_departure	actual_arrival	created_at	update_at
970	3-06-07	2023-06-12	13	12	0239	70	9	Boarding	2023-12-20	2023-03-18	2024-02-07	2023-09-17
971	3-04-19	2023-06-05	9	7	[null]	328	49	Boarding	2023-09-30	2023-12-11	2023-05-14	2024-01-28
972	3-11-25	2023-12-30	20	6	3292	249	32	Delayed	2024-03-10	2023-03-27	2023-07-04	2023-05-03
973	3-04-23	2024-01-18	19	2	241	22	13	Boarding	2023-05-25	2024-01-31	2024-03-16	2023-06-06
974	3-09-01	2023-03-23	7	15	5640	4804	6	Boarding	2023-04-12	2023-10-11	2023-06-18	2023-09-22
975	3-08-25	2023-05-19	17	13	55	676	32	Delayed	2023-07-29	2024-01-15	2024-02-29	2023-10-15

Total rows: Query complete 00:00:00.084

Query returned successfully in 84 msec.

3)

The screenshot shows the pgAdmin 4 interface with a query editor containing the following SQL code:

```

1 CREATE OR REPLACE PROCEDURE get_flights_from_airport(
2     IN p_airport_id INT
3 )
4 LANGUAGE plpgsql
5 AS $$
6 BEGIN
7
8     RAISE NOTICE 'Flights departing from airport_id = %', p_airport_id;
9
10
11     SELECT f.flight_id,
12            f.sch_departure_time,
13            f.sch_arrival_time,
14            a.airport_name AS departing_airport,
15            f.departing_gate,
16            f.arriving_gate,
17            f.airline_id
18     FROM Flights f
19     JOIN Airport a ON f.departing_airport_id = a.airport_id
20     WHERE f.departing_airport_id = p_airport_id;
21 END;
22 $$;

```

The query was executed successfully, returning a message: "Query returned successfully in 62 msec." The status bar at the bottom indicates "Total rows: Query complete 00:00:00.062".

4)

The screenshot shows the pgAdmin 4 interface with a query editor containing the following SQL code:

```

1 CREATE OR REPLACE FUNCTION avg_delay(p_airport_id INT)
2 RETURNS INTERVAL
3 LANGUAGE plpgsql
4 AS $$
5 DECLARE
6     result INTERVAL;
7 BEGIN
8     SELECT AVG(f.actual_arrival - f.scheduled_arrival)
9     INTO result
10    FROM flights f
11    WHERE f.arrival_airport_id = p_airport_id
12          AND f.actual_arrival IS NOT NULL
13          AND f.scheduled_arrival IS NOT NULL;
14
15     RETURN result;
16 END;
17 $$;
18
19 select avg_delay(13)

```

The query was executed successfully, returning a message: "Query returned successfully in 62 msec." The status bar at the bottom indicates "Total rows: 1 Query complete 00:00:00.060".

The Data Output tab shows the following result:

avg_delay
-00:00:14.916667

The status bar at the bottom indicates "Showing rows: 1 to 1 Page No: 1 of 1".

5)

The screenshot shows the pgAdmin 4 interface with a query window open. The query is a PL/pgSQL procedure named `passengers_for_flight` that takes an integer `p_flight_id` as input. The procedure body includes a `SELECT` statement that joins the `passengers` table (`p`) with the `booking` table (`b`) on `p.passenger_id = b.passenger_id` and filters for `b.flight_id = p_flight_id`. The procedure is executed successfully, returning the message "Query returned successfully in 56 msec."

```

1 CREATE OR REPLACE PROCEDURE passengers_for_flight(
2   p_flight_id INT
3 )
4 LANGUAGE plpgsql
5 AS $$
6 BEGIN
7   SELECT p.passenger_id, p.first_name, p.last_name
8   FROM passengers p
9   JOIN booking b ON p.passenger_id = b.passenger_id
10  WHERE b.flight_id = p_flight_id;
11 END;
12 $$;
13 |
14 call passengers_for_flight(5);

```

Data Output Messages Notifications

CREATE PROCEDURE

Query returned successfully in 56 msec.

Total rows: Query complete 00:00:00.056

AWON +0.21%

Полиск

KA3

22:35 02.12.2025

6)

The screenshot shows the pgAdmin 4 interface with a query window open. The query is a PL/pgSQL procedure named `top_passenger` that declares a variable `top_passenger_id` of type `INT`. The procedure body includes a `SELECT` statement that joins the `passengers` table (`p`) with the `booking` table (`b`) on `p.passenger_id = b.passenger_id` and the `booking_flight` table (`bf`) on `b.booking_id = bf.booking_id`. The results are grouped by `p.passenger_id` and ordered by `bf.flight_id` in descending order, limited to 1 row. The procedure is executed successfully, returning the message "Query returned successfully in 65 msec."

```

1 CREATE OR REPLACE PROCEDURE top_passenger()
2 LANGUAGE plpgsql
3 AS $$
4 DECLARE
5   top_passenger_id INT;
6 BEGIN
7   SELECT p.passenger_id
8   INTO top_passenger_id
9   FROM passengers p
10  JOIN booking b ON p.passenger_id = b.passenger_id
11  JOIN booking_flight bf ON b.booking_id = bf.booking_id
12  GROUP BY p.passenger_id
13  ORDER BY bf.flight_id DESC
14  LIMIT 1;
15
16   RAISE NOTICE 'Passenger with most flights: %', top_passenger_id;
17 END;
18 $$;
19 call top_passenger();

```

Data Output Messages Notifications

NOTICE: Passenger with most flights: 68

CALL

Query returned successfully in 65 msec.

Total rows: Query complete 00:00:00.065

USD/INR +0.33%

Полиск

ENG

23:22 02.12.2025

7)

pgAdmin 4

Welcome lab_10/postgres@PostgreSQL 17* X

lab_10/postgres@PostgreSQL 17

Query Query History

```

1 CREATE OR REPLACE PROCEDURE flights_delayed_24h()
2 LANGUAGE plpgsql
3 AS $$
4 BEGIN
5     RAISE NOTICE 'Flights delayed more than 24 hours:';
6     PERFORM flight_id, flight_no, actual_arrival - scheduled_arrival AS delay
7     FROM flights
8     WHERE actual_arrival - scheduled_arrival > INTERVAL '24 hours';
9 END;
10 $$;
11 CALL flights_delayed_24h();

```

Data Output Messages Notifications

CREATE PROCEDURE

Query returned successfully in 54 msec.

Total rows: Query complete 00:00:00.054

USD/RUR +0.13%

Полиск

ENG 23:23 02.12.2025

✓ Query returned successfully in 54 msec. X

CRLF Ln 1, Col 1

8)

pgAdmin 4

Welcome lab_10/postgres@PostgreSQL 17* X

lab_10/postgres@PostgreSQL 17

Query Query History

```

1 CREATE OR REPLACE FUNCTION count_flights_per_airline()
2 RETURNS TABLE(airline_id INT, flight_count INT)
3 LANGUAGE plpgsql
4 AS $$
5 BEGIN
6     RETURN QUERY
7     SELECT airline_id, COUNT(*)
8     FROM flights
9     GROUP BY airline_id;
10 END;
11 $$;
12 select * from count_flights_per_airline();
13

```

Data Output Messages Notifications

CREATE FUNCTION

Query returned successfully in 87 msec.

Total rows: Query complete 00:00:00.087

AW01 +0.11%

Полиск

ENG 23:24 02.12.2025

✓ Query returned successfully in 87 msec. X

CRLF Ln 1, Col 1

[illegible]

The screenshot displays the pgAdmin 4 web interface. The top navigation bar includes 'File', 'Object', 'Tools', 'Edit', 'View', 'Window', and 'Help'. The main window is titled 'lab_10/postgres@PostgreSQL 17*'. Below the title bar, there's a toolbar with various icons for file operations, query execution, and settings. The 'Query' tab is active, showing a SQL script in a text editor. The script defines a procedure named 'most_expensive_flight' that selects flight details from a table 'bookings' and joins it with a table 'flights'. The procedure is written in PL/pgSQL. Below the query editor, the 'Data Output' tab is visible, showing the message 'Query returned successfully in 48 msec.'.

```

1 CREATE OR REPLACE PROCEDURE most_expensive_flight(
2     OUT o_flight_id INT,
3     OUT o_departing_airport INT,
4     OUT o_arriving_airport INT,
5     OUT o_ticket_price DECIMAL
6 )
7 LANGUAGE plpgsql
8 AS $$
9 BEGIN
10     SELECT
11         b.flight_id,
12         f.departing_airport_id,
13         f.arriving_airport_id,
14         b.price
15     INTO
16         o_flight_id,
17         o_departing_airport,
18         o_arriving_airport,
19         o_ticket_price
20     FROM bookings b
21     JOIN flights f ON b.flight_id = f.flight_id
22     ORDER BY b.price DESC
23 
```

Data Output Messages Notifications

CREATE PROCEDURE

Query returned successfully in 48 msec.

Total rows: Query complete 00:00:00.048

4°C Mostly cloudy

Понед

ENG

23:30 02.12.2023