

1 task

The screenshot shows the PostgreSQL IDE interface. On the left, the Object Explorer displays a database schema with tables like 'baggage', 'boarding_pass', and 'booking'. The main editor window shows a SQL query being executed. The query is a PL/pgSQL procedure that inserts a new flight record into the 'flights' table. The procedure is named 'insert_new_flight' and takes several parameters: flight_id, flight_no, scheduled_departure, scheduled_arrival, departure_airport_id, arrival_airport_id, departing_gate, arriving_gate, airline_id, status, actual_departure, actual_arrival, created_at, and update_at. The procedure is called with the following values: (1007, 'ES-AN', '2021-05-06', '2023-12-16', 20, 16, '95', '26', 33, 'Boarding', '2021-11-28', '2025-11-28', '2023-10-08', '2024-05-07'). The query is executed successfully, returning 0 rows.

```
--lab10
--1
CREATE PROCEDURE insert_new_flight(flight_id INT, flight_no VARCHAR, sch_dep DATE, sch_arr DATE, dep_air_id INT,
arr_air_id INT, dep_gate VARCHAR, arr_gate VARCHAR, air_id INT, status VARCHAR, act_dep DATE, act_arr DATE,
cr_at DATE, upd_at DATE)
LANGUAGE plpgsql
AS $$
BEGIN
INSERT INTO flights(flight_id, flight_no, scheduled_departure, scheduled_arrival, departure_airport_id,
arrival_airport_id, departing_gate, arriving_gate, airline_id, status, actual_departure,
actual_arrival, created_at, update_at)
VALUES(flight_id, flight_no, sch_dep, sch_arr, dep_air_id, arr_air_id, dep_gate, arr_gate, air_id, status, act_dep,
act_arr, cr_at, upd_at);
END;
$$;

CALL insert_new_flight(1007, 'ES-AN', '2021-05-06', '2023-12-16', 20, 16, '95',
'26', 33, 'Boarding', '2021-11-28', '2025-11-28', '2023-10-08', '2024-05-07');
```

Data Output Messages Notifications

CALL

Query returned successfully in 87 msec.

Total rows: Query complete 00:00:00.087 CRLF Ln 6123, Col 1

2 task

The screenshot shows the PostgreSQL IDE interface. On the left, the Object Explorer displays a database schema with tables like 'baggage', 'boarding_pass', and 'booking'. The main editor window shows a SQL query being executed. The query is a PL/pgSQL procedure that updates the status of a flight record in the 'flights' table. The procedure is named 'update_status' and takes two parameters: p_flight_id and p_new_status. The procedure is called with the following values: (1007, 'delayed'). The query is executed successfully, returning 0 rows.

```
--2
CREATE PROCEDURE update_status(
p_flight_id INT,
p_new_status VARCHAR)
LANGUAGE plpgsql
AS $$
BEGIN
UPDATE flights SET status = 'delayed', update_at = NOW()
WHERE flight_id = p_flight_id;
END;
$$;

CALL update_status(1007, 'delayed');
```

Data Output Messages Notifications

CALL

Query returned successfully in 61 msec.

Total rows: Query complete 00:00:00.061 CRLF Ln 6144, Col 1

3 task

The screenshot shows the PostgreSQL IDE interface. On the left, the Object Explorer displays the database schema, including tables like 'baggage' and 'booking'. The main editor shows a PL/pgSQL procedure named 'get_flights_from_airport'. The procedure takes an integer 'p_airport_id' and returns a cursor 'flight_cursor'. It uses a 'FOR' loop to iterate over flights from the 'flights' table, filtered by 'departure_airport_id = p_airport_id', ordered by 'scheduled_departure'. The procedure calls 'CALL get_flights_from_airport(1, 'cur_flights');' and 'FETCH ALL FROM cur_flights;'. The Data Output pane shows the results of the query, displaying a table with columns: flight_id, flight_no, scheduled_departure, scheduled_arrival, and status. The table contains 9 rows of data. A status bar at the bottom indicates 'Successfully run. Total query runtime: 81 msec. 48 rows affected.'

```

--3
CREATE PROCEDURE get_flights_from_airport(
    p_airport_id INT,
    OUT flight_cursor REFCURSOR
)
LANGUAGE plpgsql
AS $$
BEGIN
    flight_cursor := 'cur_flights';
    OPEN flight_cursor FOR
    SELECT flight_id, flight_no, scheduled_departure, scheduled_arrival, status
    FROM flights
    WHERE departure_airport_id = p_airport_id
    ORDER BY scheduled_departure;
END;
$$;

CALL get_flights_from_airport(1, 'cur_flights');
FETCH ALL FROM cur_flights;

```

| flight_id | flight_no | scheduled_departure | scheduled_arrival | status |
|-----------|-----------|---------------------|-------------------|----------|
| 1 | 168 | 2023-03-21 | 2023-05-24 | Delayed |
| 2 | 231 | 2023-04-04 | 2024-02-24 | Boarding |
| 3 | 335 | 2023-04-14 | 2023-10-18 | Delayed |
| 4 | 564 | 2023-04-27 | 2023-12-19 | Boarding |
| 5 | 165 | 2023-04-30 | 2023-08-20 | Boarding |
| 6 | 193 | 2023-05-02 | 2023-10-16 | Delayed |
| 7 | 994 | 2023-05-06 | 2024-03-11 | Delayed |
| 8 | 456 | 2023-05-16 | 2023-09-24 | Delayed |
| 9 | 61 | 2023-05-16 | 2024-03-14 | Delayed |

4 task

The screenshot shows the PostgreSQL IDE interface. The main editor displays a PL/pgSQL function named 'avg_delay'. The function takes an integer 'p_airport_id' and returns an interval. It uses a 'SELECT AVG' statement to calculate the average delay from the 'flights' table, filtered by 'arrival_airport_id = p_airport_id'. The function calls 'SELECT avg_delay(2);'. The Data Output pane shows the result of the query, displaying a single row with the value '00:00:22.75'. A status bar at the bottom indicates 'Successfully run. Total query runtime: 126 msec. 1 rows affected.'

```

--4
CREATE FUNCTION avg_delay(p_airport_id INT)
RETURNS INTERVAL
LANGUAGE plpgsql
AS $$
DECLARE
    avg_delay INTERVAL;
BEGIN
    SELECT AVG(actual_arrival - scheduled_arrival)
    INTO avg_delay
    FROM flights
    WHERE arrival_airport_id = p_airport_id;
    RETURN avg_delay;
END;
$$;

SELECT avg_delay(2);

```

| avg_delay |
|-------------|
| 00:00:22.75 |

5 task

Object Explorer

- Columns
- Constraints
- Indexes
- RLS Policies
- Rules
- Triggers
- booking
 - Columns (8)
 - booking_id
 - passenger_id
 - booking_platform
 - created_at
 - update_at
 - status
 - price
 - ticket_discount
 - Constraints
 - Indexes
 - RLS Policies
 - Rules
 - Triggers
 - booking_flight
 - Columns
 - flight_id
 - flight_no
 - scheduled_departure
 - scheduled_arrival
 - departure_airport_id
 - arrival_airport_id
 - departing_gate
 - arriving_date
 - Constraints
 - Indexes
 - RLS Policies
 - Rules
 - Triggers
- flights
 - Columns (14)
 - flight_id
 - flight_no
 - scheduled_departure
 - scheduled_arrival
 - departure_airport_id
 - arrival_airport_id
 - departing_gate
 - arriving_date
 - Constraints
 - Indexes
 - RLS Policies
 - Rules
 - Triggers

Query

```

6175
6176
6177 --5
6178 CREATE PROCEDURE list_passengers(p_flight_no VARCHAR, OUT ref refcursor)
6179 LANGUAGE plpgsql
6180 AS $$
6181 BEGIN
6182   ref := 'passengers_cursor';
6183   OPEN ref FOR
6184     SELECT p.passenger_id, p.first_name, p.last_name
6185     FROM passengers p
6186     JOIN booking b ON b.passenger_id = p.passenger_id
6187     JOIN booking_flight bf ON b.booking_id = bf.booking_id
6188     JOIN flights f ON f.flight_id = bf.flight_id
6189     WHERE f.flight_no = p_flight_no;
6190   END;
6191 $$;
6192 CALL list_passengers('US-CT', 'passenger_cursor');
6193 FETCH ALL FROM passengers_cursor;
6194

```

Data Output

| passenger_id [PK] integer | first_name character varying (50) | last_name character varying (50) |
|------------------------------|--------------------------------------|-------------------------------------|
| 9 | Alfonso | Phillipou |
| 32 | Elfrida | Schukert |

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

Total rows: 2 Query complete 00:00:00.072 CRLF Ln 6188, Col 45

6 task

Object Explorer

- Columns
- Constraints
- Indexes
- RLS Policies
- Rules
- Triggers
- booking
 - Columns (8)
 - booking_id
 - passenger_id
 - booking_platform
 - created_at
 - update_at
 - status
 - price
 - ticket_discount
 - Constraints
 - Indexes
 - RLS Policies
 - Rules
 - Triggers
 - booking_flight
 - Columns
 - flight_id
 - flight_no
 - scheduled_departure
 - scheduled_arrival
 - departure_airport_id
 - arrival_airport_id
 - departing_gate
 - arriving_date
 - Constraints
 - Indexes
 - RLS Policies
 - Rules
 - Triggers
- flights
 - Columns (14)
 - flight_id
 - flight_no
 - scheduled_departure
 - scheduled_arrival
 - departure_airport_id
 - arrival_airport_id
 - departing_gate
 - arriving_date
 - Constraints
 - Indexes
 - RLS Policies
 - Rules
 - Triggers

Query

```

6194
6195
6196
6197
6198
6199 --6
6200 CREATE PROCEDURE get_top_passenger(OUT ref REFCURSOR)
6201 LANGUAGE plpgsql
6202 AS $$
6203 BEGIN
6204   ref := 'passenger_cursor';
6205   OPEN ref FOR
6206     SELECT p.passenger_id, p.first_name || ' ' || p.last_name,
6207     COUNT(bf.flight_id) AS total_flights
6208     FROM passengers p
6209     JOIN booking b ON p.passenger_id = b.passenger_id
6210     JOIN booking_flight bf ON b.booking_id = bf.booking_id
6211     GROUP BY p.passenger_id, p.first_name || ' ' || p.last_name
6212     ORDER BY total_flights DESC
6213     LIMIT 1;
6214   END;
6215 $$;
6216
6217 BEGIN;
6218 CALL get_top_passenger('passenger_cursor');
6219 FETCH ALL FROM passenger_cursor;
6220 COMMIT;
6221

```

Data Output

COMMIT

Query returned successfully in 41 msec.

Total rows: Query complete 00:00:00.041 CRLF Ln 6194, Col 1

7 task

The screenshot shows the PostgreSQL IDE interface. On the left, the Object Explorer displays a database schema with tables like booking, booking_flight, flights, and passengers. The 'flights' table is selected, showing its columns: flight_id, flight_no, scheduled_departure, scheduled_arrival, departure_airport_id, arrival_airport_id, departing_gate, arriving_gate, airline_id, status, actual_departure, actual_arrival, created_at, and update_at.

The main query editor displays the following SQL code:

```

--7
CREATE PROCEDURE delayed_flights(ref OUT refcursor)
LANGUAGE plpgsql
AS
$$
BEGIN
    ref := 'delayed_flights_cursor';

    OPEN ref FOR
    SELECT * FROM flights
    WHERE (actual_arrival - actual_departure) > 1;

END;
$$;

BEGIN;
CALL delayed_flights('refcursor');
FETCH ALL FROM "delayed_flights_cursor";
COMMIT;

```

The Data Output pane at the bottom shows the message: "Query returned successfully in 63 msec."

8 task

The screenshot shows the PostgreSQL IDE interface. The Object Explorer on the left shows the 'flights' table selected. The main query editor displays the following SQL code:

```

CREATE FUNCTION count_flights()
RETURNS TABLE(
    airline_id INT,
    airline_name VARCHAR,
    count_flights BIGINT)
LANGUAGE plpgsql
AS
$$
BEGIN
    RETURN QUERY
    SELECT a.airline_id, a.airline_name, COUNT(f.flight_id) AS count_flights
    FROM airline a
    LEFT JOIN flights f ON f.airline_id = a.airline_id
    GROUP BY a.airline_id, a.airline_name;
END;
$$;

SELECT * FROM count_flights();

```

The Data Output pane at the bottom shows the results of the query, displaying 7 rows of data:

| airline_id | airline_name | count_flights |
|------------|--------------|---------------|
| 42 | SJS | 14 |
| 29 | NHT | 29 |
| 4 | KHS | 13 |
| 34 | SMM | 19 |
| 41 | RBR | 16 |
| 40 | CII | 24 |
| 46 | YBQ | 15 |

9 task

Object Explorer

- MY DB
 - Servers (1)
 - PostgreSQL 17
 - Databases (6)
 - airport_2
 - Casts
 - Catalogs
 - Event Triggers
 - Extensions
 - Foreign Data Wrappers
 - Languages
 - Publications
 - Schemas (1)
 - public
 - Aggregates
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions
 - Materialized Views
 - Operators
 - Procedures
 - Sequences
 - Tables (10)
 - airline
 - Columns (6)
 - airline_id
 - airline_code
 - airline_name
 - airline_country
 - created_at
 - update_at

Query

```

6267 --9
6268
6269 CREATE PROCEDURE average(p_flight_no VARCHAR,
6270 OUT avg_price NUMERIC)
6271 LANGUAGE plpgsql
6272 AS
6273 $$
6274 BEGIN
6275 SELECT AVG(price)
6276 INTO avg_price FROM booking b
6277 JOIN booking_flight bf ON b.booking_id = bf.booking_id
6278 JOIN flights f ON f.flight_id = bf.flight_id
6279 WHERE f.flight_no = p_flight_no;
6280 END;
6281 $$;
6282 |
6283 CALL average('US-NM', NULL);
6284

```

Data Output

| avg_price |
|-----------------------|
| 3761.5575000000000000 |

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

Total rows: 1 Query complete 00:00:00.089

You are currently running version 9.6 of pgAdmin 4, however the current version is 9.9.

Please click [here](#) for more information.

CRLF Ln 6282, Col 1

10 task

Object Explorer

- RLS Policies
- Rules
- Triggers
- baggage_check
 - Columns
 - Constraints
 - Indexes
 - RLS Policies
 - Rules
 - Triggers
- boarding_pass
 - Columns
 - Constraints
 - Indexes
 - RLS Policies
 - Rules
 - Triggers
- booking
 - Columns (8)
 - booking_id
 - passenger_id
 - booking_platform
 - created_at
 - update_at
 - status
 - price
 - ticket_discount
 - Constraints
 - Indexes
 - RLS Policies
 - Rules
 - Triggers
- booking_flight
 - Columns
 - Constraints
 - Indexes
 - RLS Policies

Query

```

6284 --10
6285
6286 CREATE PROCEDURE highest_flight(OUT p_flight_no VARCHAR, OUT p_dep VARCHAR, OUT p_arr VARCHAR, OUT p_ma
6287 LANGUAGE plpgsql
6288 AS
6289 $$
6290 BEGIN
6291 SELECT f.flight_no, a_dep.airport_name, a_arr.airport_name,
6292 MAX(b.price)
6293 INTO p_flight_no, p_dep, p_arr, p_max_price
6294 FROM booking b
6295 JOIN booking_flight bf ON b.booking_id = bf.booking_id
6296 JOIN flights f ON f.flight_id = bf.flight_id
6297 JOIN airport a_dep ON a_dep.airport_id = f.departure_airport_id
6298 JOIN airport a_arr ON a_arr.airport_id = f.arrival_airport_id
6299 GROUP BY f.flight_no, a_dep.airport_name, a_arr.airport_name
6300 ORDER BY MAX(b.price) DESC
6301 LIMIT 1;
6302 END;
6303 $$;
6304 CALL highest_flight(NULL, NULL, NULL, NULL);
6305

```

Data Output

| p_flight_no | p_dep | p_arr | p_max_price |
|-------------|--------------------|------------------------------------|-------------|
| CN-35 | Garbaharey Airport | Henri Coandă International Airport | 9851.86 |

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

Total rows: 1 Query complete 00:00:00.072

You are currently running version 9.6 of pgAdmin 4, however the current version is 9.9.

Please click [here](#) for more information.

CRLF Ln 6304, Col 1