

The screenshot shows two instances of the DataGrip IDE interface, each displaying a database console for the PostgreSQL database.

**Top Console (console\_10):**

- Code: A SQL script named `ins_flight.sql` containing a procedure definition and a call to it.
- Output: Result of the query `SELECT p_new_flight_id` showing the value `3334`.
- Services: Shows the database connection `postgres@localhost` and various other consoles like `console_4`, `console_8`, etc.
- File Explorer: Shows a folder named `studying` containing various files including `Lab 8` and `Unity`.

**Bottom Console (console\_10):**

- Code: A SQL script named `upd_flight_status.sql` containing a procedure definition and a call to it.
- Output: Result of the query `SELECT p_new_flight_id` showing the value `3334`.
- Services: Shows the database connection `postgres@localhost` and various other consoles like `console_4`, `console_8`, etc.
- File Explorer: Shows a folder named `studying` containing various files including `Lab 8` and `Unity`.

The screenshot shows the DataGrip IDE interface. The top navigation bar includes tabs for 'studying' and 'main'. Below the tabs, there are several database connection icons labeled 'sole\_2', 'console', 'console\_4', 'console\_5', 'console\_6', 'console\_7', 'console\_8', 'console\_9', 'console\_10', and 'airport\_db.sql'. A 'Playground' dropdown is also present. The main area contains a code editor with the following SQL script:

```
51  create or replace function get_flights_from_airport (
52      p_airport_id int
53  )
54  returns setof flights
55  language plpgsql
56  as $$
57  begin
58      return query
59      select f.flight_id, f.flight_no, f.scheduled_departure, f.scheduled_arrival, f.actual_departure, f.status
60      from flights f
61      where f.departure_airport_id = p_airport_id;
62
63      raise notice 'Returned flights from airport %', p_airport_id;
64  end;
65  $$;
66
67
68  select * from get_flights_from_airport( 1 );
69
70
71  --4
72  create or replace function avg_delay_airport (
73      p_airport_id int
74  )
75  returns numeric
```

To the right of the code editor is a 'Files' browser window showing a directory structure with files like 'ads', 'calculus', 'database', etc.

The bottom section of the interface is titled 'Services' and shows a tree view of database connections under 'postgres@localhost':

- postgres@localhost
  - console\_4
  - airport\_db.sql 27 ms
  - console\_8
  - console\_5
  - console
  - console\_2
  - console\_10 509 ms
  - console\_9
  - console\_6
  - console\_7

On the far right, there is an 'Output' tab showing a result set with 49 rows, and a 'Result 49' tab displaying the following data:

flight_id	flight_no	scheduled_departure	scheduled_arrival	departure_airport_id
49	994 US-IN	2023-05-06	2024-03-11	
50	2001 AB-123	2024-05-01	2024-05-02	
51	2002 AB-123	2024-05-01	2024-05-02	

At the bottom of the interface, status information includes '51 rows', 'CSV v', and file statistics: '88:28 CRLF 4 spaces'.

The screenshot shows two instances of the DataGrip IDE interface, each displaying a PostgreSQL database console. The top instance is focused on creating a function named `avg_delay_airport`. The SQL code defines a function that takes an airport ID as input and returns the average delay. It includes a query that joins the `passenger` and `booking` tables to calculate the average delay per passenger. The output window shows the result of the function execution, which is 6.614035 seconds.

```
create or replace function avg_delay_airport (
    $$
    select avg_delay_airport( p.airport_id ) ;
$$
--5
create or replace procedure list_passengers_for_flight (
    in p_flight_no varchar
)
language plpgsql
as $$

begin
    return query
        select p.passenger_id, p.first_name, p.last_name
        from passengers p
        join booking b on p.passenger_id = b.passenger_id
        join booking_flight bf on b.booking_id = bf.booking_id
        join booking_flight bf_1<->1..n: on bf.booking_id = bf_.booking_id
;
```

The bottom instance of the IDE is focused on creating a function named `list_passengers_for_flight`. This function takes a flight ID as input and returns all passengers for that flight. It uses similar joins between `passenger`, `booking`, and `booking_flight` tables. The output window shows the results for a specific flight ID, listing two passengers with IDs 32, first names Elfrida, and last names Schukert.

```
create or replace function list_passengers_for_flight (
    p.last_name
)
from passengers p
join booking b
    on p.passenger_id = b.passenger_id
join booking_flight bf
    |<->1..n: on b.booking_id = bf.booking_id
    where bf.flight_id = p.flight_id;

raise notice 'Returned passengers for flight %', p.flight_id;
end;
$$;

```

Both instances of the IDE also show a sidebar titled "Files" containing a list of files and folders related to the project, such as `studyng`, `vscode`, `ads`, `calculus`, `database`, `ict`, `Lab 8`, `unity`, `коды`, `сертификаты`, and `эцп`.

The screenshot shows the DataGrip IDE interface with the following details:

- Top Bar:** Shows tabs for 'studying' (selected), 'main', and several database consoles (console\_2, console, console\_4, console\_5, console\_6, console\_7, console\_8, console\_9, console\_10). The 'airport\_db.sql' tab is also visible.
- Code Area:** Contains PostgreSQL code for creating a function and a procedure. The function 'passenger\_with\_most\_flights()' selects the passenger with the highest number of flights. The procedure 'flights\_delayed\_24h()' is also defined.
- Services Panel:** Shows a list of database connections and their status.
- Output Tab:** Displays the result of the query 'select \* from passenger\_with\_most\_flights();'. The result table has columns: passenger\_id, first\_name, last\_name, and flights\_count. One row is shown: passenger\_id 21, first\_name Merillin, last\_name Jewess, and flights\_count 34.
- File Explorer:** Shows the file structure under 'studying' including various projects and files like 'vscode', 'ads', 'calculus', etc.

The screenshot shows the DBeaver IDE interface with the following details:

- Top Bar:** Shows multiple tabs like 'studying', 'main', 'sole\_2', 'console', 'console\_4', 'console\_5', 'console\_6', 'console\_7', 'console\_8', 'console\_9', 'console\_10' (selected), and 'airport\_db.sql'. The 'Files' tab is also visible.
- Left Panel:** Shows a tree view of the project structure under 'studying' with various databases and files listed.
- Code Editor:** Displays a PostgreSQL query for creating a function named 'flights\_delayed\_24h()'. The query selects flight details and calculates the delay between scheduled and actual departure times for flights delayed by more than 24 hours. A green checkmark indicates the query was successful.
- Services:** Shows a list of database connections, with 'postgres@localhost' selected.
- Output Tab:** Shows the results of the executed query, which returns 494 rows of flight information.
- Bottom Status Bar:** Shows the connection path 'Database Consoles > postgres@localhost > console\_10', and the session details '205.37 CRLF UTF-8 4 spaces'.

flight_id	flight_no	scheduled_departure	actual_departure	delay
1	2 US-NM	2023-07-21 00:00:00.000000	2024-02-09 00:00:00.000000	0 years 0 mons 203 days 0 h
2	3 FI-OL	2023-03-29 00:00:00.000000	2024-02-21 00:00:00.000000	0 years 0 mons 329 days 0 h
3	5 RO-DJ	2023-07-03 00:00:00.000000	2023-11-18 00:00:00.000000	0 years 0 mons 138 days 0 h
4	6 CA-SK	2023-07-07 00:00:00.000000	2024-02-19 00:00:00.000000	0 years 0 mons 227 days 0 h
5	7 AU-TAS	2023-10-12 00:00:00.000000	2023-12-04 00:00:00.000000	0 years 0 mons 53 days 0 h
6	9 IN-OR	2023-05-18 00:00:00.000000	2023-04-17 00:00:00.000000	0 years 0 mons 30 days 0 h
7	11 TH-S7	2023-03-28 00:00:00.000000	2024-02-08 00:00:00.000000	0 years 0 mons 315 days 0 h

The screenshot shows the DataGrip IDE interface for managing a PostgreSQL database. The top navigation bar includes tabs for 'studying' and 'main', and several database consoles: 'sole\_2', 'console', 'console\_4', 'console\_5', 'console\_6', 'console\_7', 'c...', 'Files Alt+2', 'console\_9', 'console\_10', and 'airport\_db.sql'. The 'console\_10' tab is active.

The left sidebar displays the 'Services' tree, which includes a connection to 'postgres@localhost' and various database consoles. The 'Output' and 'Result 68 ×' panes are visible at the bottom, showing the results of a query:

```
CREATE OR REPLACE FUNCTION flights_per_airline()
RETURNS TABLE("airline_id" int, "flights_count" int)
AS $$
BEGIN
    RETURN QUERY
    SELECT f.airline_id, COUNT(*)
    FROM flights f
    GROUP BY f.airline_id;
END;
$$;
```

SELECT \* FROM flights\_per\_airline();

```
--9
CREATE OR REPLACE PROCEDURE avg_ticket_price (
    IN p_flight_no VARCHAR
)
```

The 'Result' pane displays the following data:

airline_id	flights_count
1	42
2	29
3	4
4	34
5	41
6	40
7	46

Database Consoles > postgres@localhost > console\_10

The screenshot shows the DataGrip IDE interface with the following details:

- Top Bar:** Shows tabs for consoles (console\_2, console, console\_4, ..., console\_10) and an airport\_db.sql file.
- Left Sidebar:** Shows a tree view of the project structure under "Files".
- Code Editor:** Displays a PostgreSQL function definition named `avg_price` and a call to it.

```
229 create or replace function avg_price(
230     --> numeric;
231     begin
232         select avg(price)
233             into v_avg
234             from booking b
235             join booking_flight bf 1->1..n on b.booking_id=bf.booking_id
236             where bf.flight_id=p_flight_id;
237
238         return v_avg;
239     end;
240     $$;
241
242
243
244
245
246
247
248 ✓ select avg_price( p_flight_id 3334);
249 --10
250 create or replace procedure most_expensive_f...
```

- Output Window:** Shows the result of the `avg_price(3334)` query.

avg_price	:
1	<null>

- Services Tab:** Shows a list of database connections.
- Bottom Status Bar:** Shows the current session as `postgres@localhost`, the transaction count (Tx), and file encoding.

The screenshot shows the DataGrip IDE interface with the following details:

- Top Bar:** Shows tabs for 'main' and 'airport\_db.sql'. A warning icon indicates 1 error.
- Code Editor:** Displays a PostgreSQL function definition named 'most\_expensive\_flight'. The code uses common table expressions (CTEs) to join tables 'flight' and 'booking' and then selects the top record ordered by price.
- Output Window:** Shows the result of executing the function, which returns a single row: flight\_id 915, flight\_no NP-SA, departure\_airport\_id 1, arrival\_airport\_id 10, and price 9977.57.
- Services Panel:** Lists database connections under 'postgres@localhost' and a transaction 'Tx'.
- File Explorer:** Shows a directory structure under 'studying' containing various files and folders related to study projects.

```
create or replace function most_expensive_flight()
  returns table(flight_id int, flight_no text, departure_airport_id int, arrival_airport_id int, price numeric)
as $$
begin
    with f as (
        select * from flight
        inner join booking bf on f.flight_id = bf.flight_id
        inner join booking b on bf.booking_id = b.booking_id
    )
    select * from f
    order by b.price desc
    limit 1;
end;
$$;
```

flight_id	flight_no	departure_airport_id	arrival_airport_id	price
1	915	NP-SA	10	9977.57