

TASK 1

A screenshot of the pgAdmin 4 interface. The title bar shows "Airport/postgres@PostgreSQL 17* public.flights/Airport...". The main window has two tabs: "Query" and "Query History", with "Query" selected. The query editor contains the following PostgreSQL code:

```
1 CREATE OR REPLACE PROCEDURE add_new_flight(
2     p_flight_id INT,
3     p_sch_departure_time TIMESTAMP,
4     p_sch_arrival_time TIMESTAMP,
5     p_departing_airport_id INT,
6     p_arriving_airport_id INT,
7     p_departing_gate VARCHAR(50),
8     p_arriving_gate VARCHAR(50),
9     p_airline_id INT,
10    p_act_departure_time TIMESTAMP,
11    p_act_arrival_time TIMESTAMP,
12    p_flight_no CHAR(20)
13 )
14 LANGUAGE plpgsql
15 AS $$
16 BEGIN
17     INSERT INTO flights (
18         flight_id,
19         sch_departure_time,
20         sch_arrival_time,
21         departing_airport_id,
22         arriving_airport_id,
23         departing_gate,
24         arriving_gate,
25         airline_id,
26         act_departure_time,
27         act_arrival_time,
28         flight_no,
29         created_at,
30         updated_at
31     )
32     VALUES(
33         p_flight_id,
34         p_sch_departure_time,
35         p_sch_arrival_time,
36         p_departing_airport_id,
37         p_arriving_airport_id,
38         p_departing_gate,
39         p_arriving_gate,
40         p_airline_id,
41         p_act_departure_time,
42         p_act_arrival_time,
43         p_flight_no,
44         CURRENT_TIMESTAMP,
45         CURRENT_TIMESTAMP
46     );
47 END;
48 $$;
```

The status bar at the bottom shows "Total rows: 0", "CRLF", "Ln 17, Col 26", "23:25", and "30.11.2025".

```
,  
VALUES(  
    p_flight_id,  
    p_sch_departure_time,  
    p_sch_arrival_time,  
    p_departing_airport_id,  
    p_arriving_airport_id,  
    p_departing_gate,  
    p_arriving_gate,  
    p_airline_id,  
    p_act_departure_time,  
    p_act_arrival_time,  
    p_flight_no,  
    CURRENT_TIMESTAMP,  
    CURRENT_TIMESTAMP  
);  
END;  
$$;
```

TASK 2

The screenshot shows the pgAdmin 4 interface with the following details:

- Object Explorer:** Displays the database schema. Under the "flights" table, the "Columns (13)" section is expanded, listing columns: flight_id, sch_departure_time, sch_arrival_time, departing_airport_id, arriving_airport_id, departing_gate, arriving_gate, airline_id, act_departure_time, act_arrival_time, created_at, updated_at, and flight_no.
- Query Editor:** Shows a PostgreSQL CREATE OR REPLACE PROCEDURE statement named "update_flight_arrival". The code updates the "flights" table by setting the "act_arrival_time" and "updated_at" fields to the provided values and the current timestamp respectively, where the condition is that the "flight_id" matches the parameter "p_flight_id".

```
1 CREATE OR REPLACE PROCEDURE update_flight_arrival(
2     p_flight_id INT,
3     p_act_arrival_time TIMESTAMP
4 )
5 LANGUAGE plpgsql
6 AS $$
7 BEGIN
8     UPDATE flights
9     SET
10        act_arrival_time = p_act_arrival_time,
11        updated_at = CURRENT_TIMESTAMP
12     WHERE flight_id = p_flight_id;
13 END;
14 $$;
```
- Data Output:** Shows the message "CREATE PROCEDURE" followed by "Query returned successfully in 51 msec."
- Status Bar:** At the bottom, it displays "Total rows: 0" and "Query complete 00:00:00.051".
- System Tray:** Shows standard Windows icons for search, task view, file explorer, and other system functions.
- Bottom Right:** Includes the date and time (30.11.2025, 23:35), battery level (47%), and network status.

TASK 3

pgAdmin 4

File Object Tools Edit View Window Help

Object Explorer Airport/postgres@PostgreSQL 17* public.flights/Airport...

Query History Scratch Pad

Query

```
1 CREATE OR REPLACE PROCEDURE get_departing_flights(
2     p_departure_airport_id INT
3 )
4 LANGUAGE plpgsql
5 AS $$ 
6 BEGIN
7     SELECT
8         flight_id,
9             sch_departure_time,
10            sch_arrival_time,
11            departing_airport_id,
12            arriving_airport_id,
13            departing_gate,
14            arriving_gate,
15            airline_id,
16            act_departure_time,
17            act_arrival_time,
18            created_at,
19            updated_at,
20            flight_no
21        FROM flights
22        WHERE departing_airport_id = p_departure_airport_id;
23 END;
$$;
```

Data Output Messages Notifications

CREATE PROCEDURE

Query returned successfully in 51 msec.

Total rows: Query complete 00:00:00.051

✓ Query returned successfully in 51 ms

CRLF Ln 23, Col 4

23:39 ENG 47 30.11.2025

TASK 4

pgAdmin 4

File Object Tools Edit View Window Help

Object Explorer Airport/postgres@PostgreSQL 17* public.flights/Airport...

Airport/postgres@PostgreSQL 17 No limit E I

Query Scratch Pad

```
CREATE OR REPLACE FUNCTION calculate_average_delay(
    p_arrival_airport_id INT
)
RETURNS INTERVAL
LANGUAGE plpgsql
AS $$

DECLARE
    avg_delay INTERVAL;

BEGIN
    SELECT AVG(act_arrival_time - sch_arrival_time)
    INTO avg_delay
    FROM flights
    WHERE arriving_airport_id = p_arrival_airport_id
        AND act_arrival_time IS NOT NULL
        AND sch_arrival_time IS NOT NULL;

    RETURN avg_delay;
END;
$$;
```

Data Output Messages Notifications

CREATE FUNCTION

Query returned successfully in 45 msec.

✓ Query returned successfully in 45 msec. ✘

Total rows: Query complete 00:00:00.045

CRLF Ln 19, Col 4

23:43 46 KA3 30.11.2025

The screenshot shows the pgAdmin 4 interface. On the left, the Object Explorer tree displays a database structure with nodes like 'baggage', 'baggage_check', 'boarding_pass', 'booking', 'booking_flight', and 'flights'. The 'flights' node is selected, revealing its 13 columns: 'flight_id', 'sch_departure_time', 'sch_arrival_time', 'departing_airport_id', 'arriving_airport_id', 'departing_gate', 'arriving_gate', 'airline_id', 'act_departure_time', 'act_arrival_time', 'created_at', 'updated_at', and 'flight_no'. Below the tree are sections for 'Constraints', 'Indexes', 'RLS Policies', 'Rules', and 'Triggers'. The main pane contains a query editor with the SQL code for creating the 'calculate_average_delay' function. The code uses the plpgsql language to select the average difference between actual arrival time and scheduled arrival time for flights where the arriving airport ID matches the parameter and both times are not null. The function returns an interval type. The status bar at the bottom indicates the query was completed successfully in 45 msec. A green notification bar at the bottom right also states 'Query returned successfully in 45 msec.'

TASK 5

The screenshot shows the pgAdmin 4 interface with the following details:

- Object Explorer:** On the left, it lists database objects under the "flights" table, including columns like flight_id, sch_departure_time, and act_arrival_time.
- Query Tab:** The main area contains a SQL script for creating a stored procedure.

```
1 CREATE OR REPLACE PROCEDURE get_passengers_by_flight(
2     p_flight_no CHAR(20)
3 )
4 LANGUAGE plpgsql
5 AS $$
6 BEGIN
7     SELECT
8         p.passenger_id,
9         p.first_name,
10        p.last_name,
11        p.passport_number,
12        b.booking_id,
13        b.seat,
14        b.status
15    FROM passengers p
16    JOIN booking b ON p.passenger_id = b.passenger_id
17    JOIN flights f ON b.flight_id = f.flight_id
18    WHERE f.flight_no = p_flight_no;
19
20 $$;
```
- Data Output Tab:** Shows the message "CREATE PROCEDURE".
- Messages Tab:** Shows the message "Query returned successfully in 55 msec."
- Scratch Pad Tab:** An empty tab for testing queries.
- Status Bar:** At the bottom right, it says "Query returned successfully in 55 msec." and "Ln 20, Col 4".

TASK 6

The screenshot shows the pgAdmin 4 interface with the following details:

- Title Bar:** pgAdmin 4
- Menu Bar:** File, Object, Tools, Edit, View, Window, Help
- Object Explorer:** Shows a tree view of database objects:
 - flights (selected)
 - Columns (13): flight_id, sch_departure_time, sch_arrival_time, departing_airport_id, arriving_airport_id, departing_gate, arriving_gate, airline_id, act_departure_time, act_arrival_time, created_at, updated_at, flight_no
 - Constraints
 - Indexes
 - RLS Policies
 - Rules
 - Triggers
 - passengers
 - security_check
 - Trigger Functions
 - Types
 - Views
- Query Tab:** Contains the following PostgreSQL code:

```
1 CREATE OR REPLACE PROCEDURE find_most_frequent_flyer()
2 LANGUAGE plpgsql
3 AS $$
4 BEGIN
5     SELECT
6         p.passenger_id,
7         p.first_name,
8         p.last_name,
9         p.passport_number,
10        COUNT(b.booking_id) AS total_flights
11    FROM passengers p
12    JOIN booking b ON p.passenger_id = b.passenger_id
13    GROUP BY p.passenger_id, p.first_name, p.last_name, p.passport_number
14    ORDER BY total_flights DESC
15    LIMIT 1;
16 END;
17 $$;
```
- Data Output Tab:** Shows the message: "CREATE PROCEDURE". Below it, "Query returned successfully in 54 msec."
- Scratch Pad Tab:** Shows a green success message: "Query returned successfully in 54 msec."
- System Tray:** Shows icons for search, file, browser, and system status.
- Bottom Status Bar:** Total rows: 0, Query complete 00:00:00.054, CRLF, Ln 17, Col 4, 00:00, 48, KAZ, 01.12.2025

TASK 7

pgAdmin 4

File Object Tools Edit View Window Help

Object Explorer Airport/postgres@PostgreSQL 17* public.flights/Airport...

Query Scratch Pad

```
CREATE OR REPLACE PROCEDURE find_flights_delayed_over_24h()
LANGUAGE plpgsql
AS $$

DECLARE
    flight_record RECORD;
BEGIN
    FOR flight_record IN
        SELECT
            f.flight_id,
            f.flight_no,
            f.sch_departure_time,
            f.act_departure_time,
            (f.act_departure_time - f.sch_departure_time) AS delay_interval
        FROM flights f
        WHERE (f.act_departure_time - f.sch_departure_time) > INTERVAL '24 hours'
    LOOP
        RAISE NOTICE 'Flight ID: %, Number: %, Scheduled: %, Actual: %, Delay: %',
                    flight_record.flight_id,
                    flight_record.flight_no,
                    flight_record.sch_departure_time,
                    flight_record.act_departure_time,
                    flight_record.delay_interval;
    END LOOP;
END;
$$;
```

Data Output Messages Notifications

CREATE PROCEDURE

Query returned successfully in 51 msec.

Total rows: Query complete 00:00:00.051

CRLF Ln 25, Col 4

Пойск

00:09
49 РУС 01.12.2025

TASK 8

The screenshot shows the pgAdmin 4 interface with the following details:

- Object Explorer:** On the left, under the "flights" table, the "Columns (13)" section is expanded, listing columns such as flight_id, sch_departure_time, sch_arrival_time, departing_airport_id, arriving_airport_id, departing_gate, arriving_gate, airline_id, act_departure_time, act_arrival_time, created_at, updated_at, and flight_no.
- Query Editor:** The main area contains the SQL code for creating a function.

```
1 CREATE OR REPLACE FUNCTION count_flights_per_airline_simple()
2 RETURNS TABLE(
3     airline_id INT,
4     flight_count BIGINT
5 )
6 LANGUAGE plpgsql
7 AS $$
8 BEGIN
9     RETURN QUERY
10    SELECT
11        f.airline_id,
12        COUNT(f.flight_id) AS flight_count
13    FROM flights f
14    GROUP BY f.airline_id
15    ORDER BY flight_count DESC;
16 END;
17 $$;
```
- Data Output:** Below the query editor, it shows "CREATE FUNCTION" and "Query returned successfully in 52 msec."
- Status Bar:** At the bottom, it displays "Total rows: 0" and "Query complete 00:00:00.052". It also includes a green message box: "✓ Query returned successfully in 52 msec. X". Other status indicators include "CRLF", "Ln 17, Col 4", and a date/time stamp "01.12.2025".

TASK 9

The screenshot shows the pgAdmin 4 interface with the following details:

- Object Explorer:** On the left, under the "flights" table, the "Columns (13)" section is expanded, listing columns such as flight_id, sch_departure_time, sch_arrival_time, departing_airport_id, arriving_airport_id, departing_gate, arriving_gate, airline_id, act_departure_time, act_arrival_time, created_at, updated_at, and flight_no.
- Query Tab:** The main area contains a database query for creating a stored procedure. The query is as follows:

```
CREATE OR REPLACE PROCEDURE calculate_avg_ticket_price(
    p_flight_id INT,
    OUT p_avg_price DECIMAL,
    OUT p_flight_no CHAR(20)
)
LANGUAGE plpgsql
AS $$$
BEGIN
    SELECT
        AVG(b.ticket_price),
        f.flight_no
    INTO
        p_avg_price,
        p_flight_no
    FROM booking b
    JOIN flights f ON b.flight_id = f.flight_id
    WHERE b.flight_id = p_flight_id
    GROUP BY f.flight_no;
END;
$$;
```

Below the query, the message "CREATE PROCEDURE" is displayed, followed by "Query returned successfully in 48 msec."

At the bottom, the status bar shows "Total rows: 0" and "Query complete 00:00:00.048".

TASK 10

pgAdmin 4

File Object Tools Edit View Window Help

Object Explorer

Airport/postgres@PostgreSQL 17* public.flights/Airport

Query History

```
CREATE OR REPLACE PROCEDURE find_most_expensive_flight(
    OUT p_flight_no CHAR(20),
    OUT p_departure_airport VARCHAR(50),
    OUT p_arrival_airport VARCHAR(50),
    OUT p_ticket_price DECIMAL
)
LANGUAGE plpgsql
AS $$

BEGIN
    SELECT
        f.flight_no,
        dep.airport_name,
        arr.airport_name,
        b.ticket_price
    INTO
        p_flight_no,
        p_departure_airport,
        p_arrival_airport,
        p_ticket_price
    FROM booking b
    JOIN flights f ON b.flight_id = f.flight_id
    JOIN airport dep ON f.departing_airport_id = dep.airport_id
    JOIN airport arr ON f.arriving_airport_id = arr.airport_id
    ORDER BY b.ticket_price DESC
    LIMIT 1;
END;
$$;
```

Data Output Messages Notifications

CREATE PROCEDURE

Total rows: Query complete 00:00:00.048 CRLF Ln 28, Col 1

Поиск

00:24 48 KA3 01.12.2025