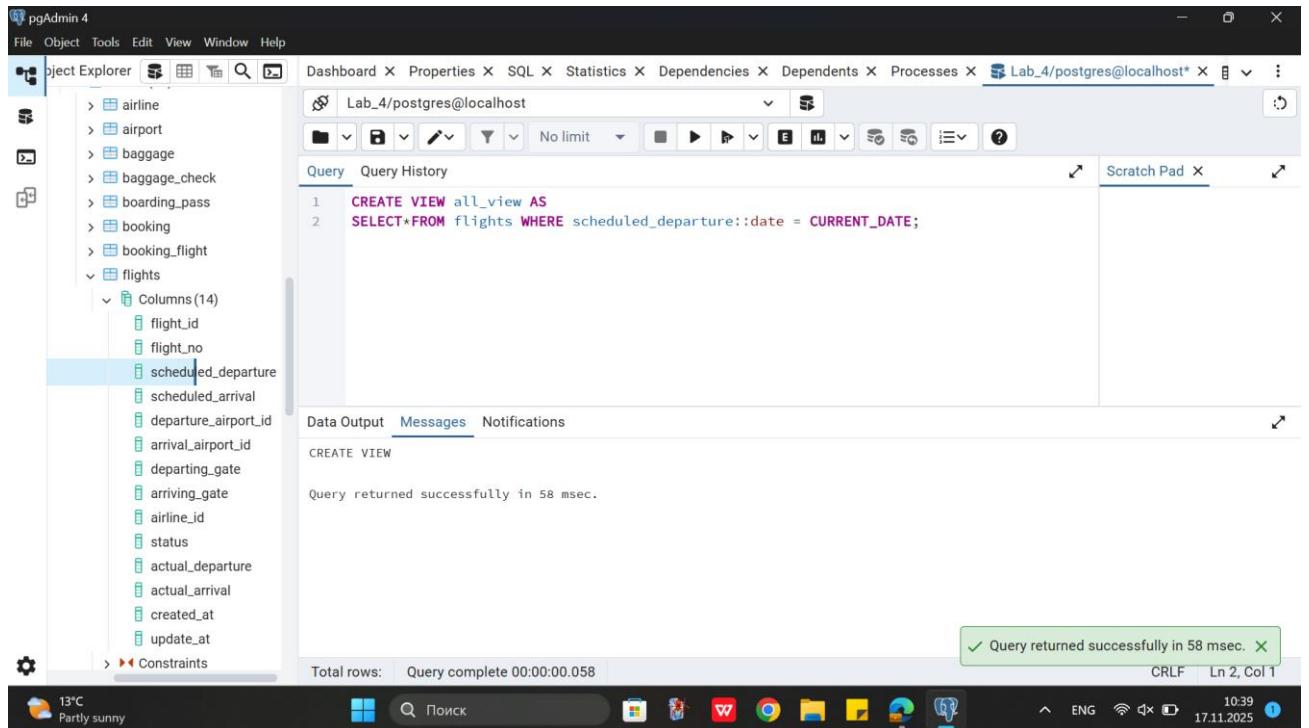


# Laboratory work 8

1. Create a view to show details of all flights that are departing on a specific date.



The screenshot shows the pgAdmin 4 interface. In the left sidebar, under the 'flights' table, the 'Columns (14)' section is expanded, showing various flight-related fields like 'flight\_id', 'flight\_no', 'scheduled\_departure', etc. In the main query editor, the following SQL code is written:

```
1 CREATE VIEW all_view AS
2 SELECT * FROM flights WHERE scheduled_departure::date = CURRENT_DATE;
```

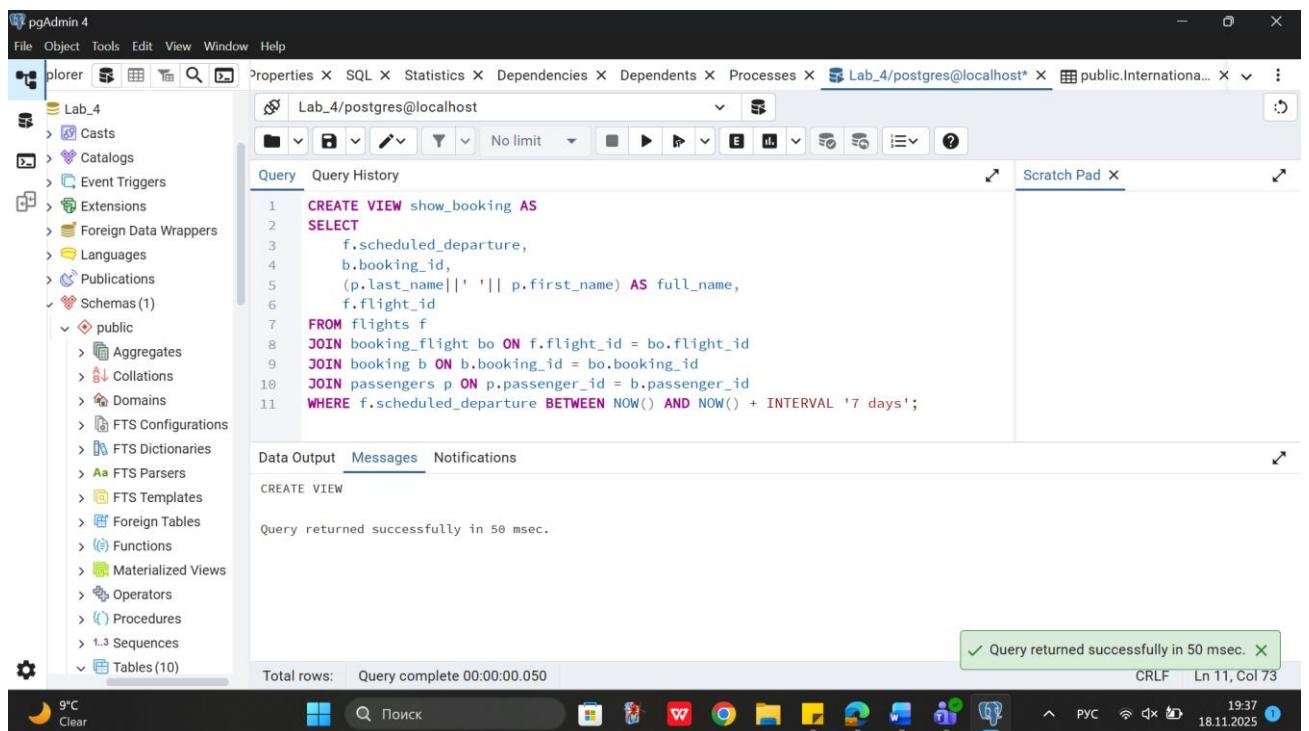
The 'Messages' tab shows the result of the query execution:

CREATE VIEW

Query returned successfully in 58 msec.

A green status bar at the bottom right indicates: ✓ Query returned successfully in 58 msec. X

2. Create a view that shows bookings for flights scheduled to depart within the next week.



The screenshot shows the pgAdmin 4 interface. In the left sidebar, under the 'Schemas(1) / public' section, the 'Tables (10)' section is expanded, showing various tables like 'Aggregates', 'Collations', 'Domains', etc. In the main query editor, the following SQL code is written:

```
1 CREATE VIEW show_booking AS
2 SELECT
3     f.scheduled_departure,
4     b.booking_id,
5     (p.last_name || ' ' || p.first_name) AS full_name,
6     f.flight_id
7 FROM flights f
8 JOIN booking_flight bo ON f.flight_id = bo.flight_id
9 JOIN booking b ON b.booking_id = bo.booking_id
10 JOIN passengers p ON p.passenger_id = b.passenger_id
11 WHERE f.scheduled_departure BETWEEN NOW() AND NOW() + INTERVAL '7 days';
```

The 'Messages' tab shows the result of the query execution:

CREATE VIEW

Query returned successfully in 50 msec.

A green status bar at the bottom right indicates: ✓ Query returned successfully in 50 msec. X

3. Create a view to show the top 5 most popular flight routes based on the number of bookings.

The screenshot shows the pgAdmin 4 interface. On the left, the Object Explorer displays a database structure with tables like 'flights' and 'booking\_flight'. The central area contains a SQL query window with the following code:

```
CREATE VIEW popular_routes AS
SELECT
    f.departure_airport_id,
    f.arrival_airport_id,
    count(b.booking_id) AS total
FROM flights f
JOIN booking_flight bo ON f.flight_id = bo.flight_id
JOIN booking b ON b.booking_id = bo.booking_id
GROUP BY f.departure_airport_id, f.arrival_airport_id
ORDER BY total DESC LIMIT 5;
```

Below the query window is a Data Output tab showing the results of the query:

	departure_airport_id	arrival_airport_id	total
1	10	7	18
2	4	10	15
3	13	4	14
4	6	16	14
5	14	7	11

Total rows: 5 | Query complete 00:00:00.051 | CRLF | Ln 1, Col 1

4. Create a view that lists all flights for a specific airline.

The screenshot shows the pgAdmin 4 interface. On the left, the Object Explorer displays a database structure with tables like 'airline' and 'flights'. The central area contains a SQL query window with the following code:

```
CREATE VIEW specific_airline AS
SELECT a.airline_name, a.airline_id, f.flight_id
FROM flights f
JOIN airline a ON a.airline_id = f.airline_id
WHERE a.airline_name = 'IPC';

SELECT * FROM specific_airline;
```

Below the query window is a Data Output tab showing the results of the query:

airline_name	airline_id	flight_id
IPC	1	13
IPC	1	33
IPC	1	36
IPC	1	73
IPC	1	78
IPC	1	143
IPC	1	170

Total rows: 32 | Query complete 00:00:00.056 | CRLF | Ln 5, Col 30

5. Modify the view created in task 4 to show only flights departing within the next 7 days for a specific airline

The screenshot shows the pgAdmin 4 interface. In the left sidebar, under the 'flights' table, there is a 'Columns(14)' section. The 'Query' tab in the center contains the following SQL code:

```
1 CREATE VIEW show_airline AS
2 SELECT
3     f.scheduled_departure,
4     f.flight_id,
5     a.airline_name,
6     a.airline_id
7 FROM flights f
8 JOIN airline a ON a.airline_id = f.airline_id
9 WHERE a.airline_name = 'IPC'
10 AND f.scheduled_departure BETWEEN NOW() AND NOW() + INTERVAL '7 days';
11
12
13
14
```

The 'Data Output' tab at the bottom shows the structure of the view:

scheduled_departure	flight_id	airline_name	airline_id
date	integer	character varying (50)	integer

Total rows: 0 Query complete 00:00:00.077 CRLF Ln 10, Col 71

6. Create a view to show flights that are delayed by more than 24 hours.

The screenshot shows the pgAdmin 4 interface. In the left sidebar, under the 'booking' table, there is a 'Columns(8)' section. The 'Query' tab in the center contains the following SQL code:

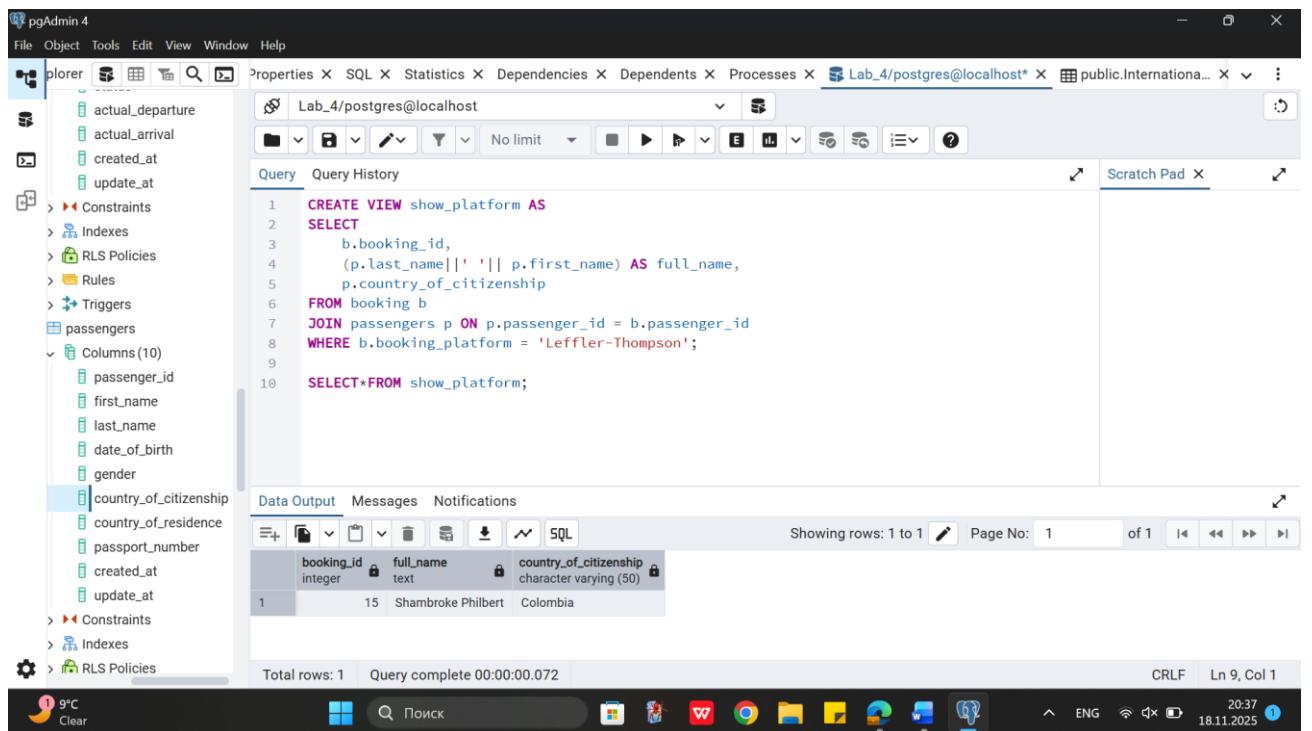
```
1 CREATE VIEW show_24 AS
2 SELECT * FROM flights
3 WHERE actual_departure - scheduled_departure > 1
```

The 'Data Output' tab at the bottom shows the message:

CREATE VIEW  
Query returned successfully in 53 msec.

✓ Query returned successfully in 53 msec. X  
Total rows: 0 Query complete 00:00:00.053 CRLF Ln 3, Col 49

7. Create a view in which you can display the full name and country of origin of passengers who made bookings on Leffler-Thompson platform. Then show the list of that passengers.



The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database schema with tables like 'actual\_departure', 'actual\_arrival', 'created\_at', 'update\_at', 'passenger', and 'bookings'. The 'bookings' table is currently selected. The main query editor window contains the following SQL code:

```

CREATE VIEW show_platform AS
SELECT
    b.booking_id,
    (p.last_name||' '||p.first_name) AS full_name,
    p.country_of_citizenship
FROM booking b
JOIN passengers p ON p.passenger_id = b.passenger_id
WHERE b.booking_platform = 'Leffler-Thompson';

SELECT*FROM show_platform;

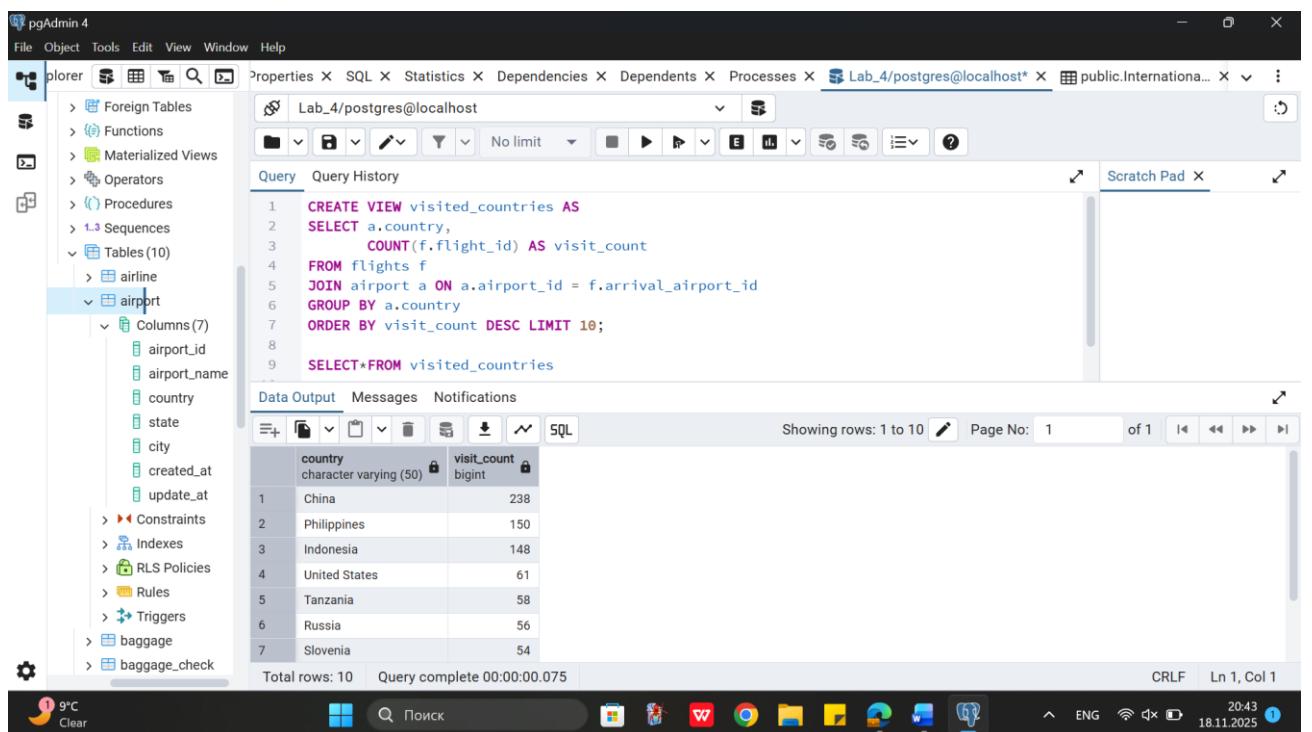
```

The results pane shows one row of data:

booking_id	full_name	country_of_citizenship
15	Shambroke Philbert	Colombia

Total rows: 1 Query complete 00:00:00.072

8. Create a view that shows top 10 most visited countries.



The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database schema with various objects like Foreign Tables, Functions, Materialized Views, Operators, Procedures, Sequences, and tables such as 'airline' and 'airport'. The 'airport' table is currently selected. The main query editor window contains the following SQL code:

```

CREATE VIEW visited_countries AS
SELECT a.country,
       COUNT(f.flight_id) AS visit_count
FROM flights f
JOIN airport a ON a.airport_id = f.arrival_airport_id
GROUP BY a.country
ORDER BY visit_count DESC LIMIT 10;

SELECT*FROM visited_countries

```

The results pane shows the top 10 most visited countries:

country	visit_count
China	238
Philippines	150
Indonesia	148
United States	61
Tanzania	58
Russia	56
Slovenia	54

Total rows: 10 Query complete 00:00:00.075

9. Update any of the created views by adding new information in the view table.  
Show results.

The screenshot shows the pgAdmin 4 interface. In the left sidebar, under the 'bookings\_flight' table, a new view named 'visited\_countries' is being created. The SQL query in the main pane is:

```
1 CREATE OR REPLACE VIEW visited_countries AS
2 SELECT a.country,
3        COUNT(f.flight_id) AS visit_count,
4        COUNT(b.booking_id) AS visit_count_bo
5 FROM flights f
6 JOIN airport a ON a.airport_id = f.arrival_airport_id
7 JOIN bookings_flight b ON b.flight_id = f.flight_id
8 GROUP BY a.country
9 ORDER BY visit_count DESC LIMIT 10;
10
11 SELECT*FROM visited_countries
```

The 'Data Output' tab shows the results of the query:

country	visit_count	visit_count_bo
China	232	232
Indonesia	167	167
Philippines	137	137
Tanzania	78	78
United States	62	62

Total rows: 10 Query complete 00:00:00.095

10. Drop all existing views.

The screenshot shows the pgAdmin 4 interface. In the left sidebar, under the 'bookings\_flight' table, a SQL command is run to drop all existing views:

```
1 DROP VIEW IF EXISTS
2 visited_countries,
3 show_platform,
4 show_24,
5 show_airline,
6 specific_airline,
7 popular_routes,
8 show_booking,
9 all_view
10 CASCADE;
```

The 'Data Output' tab shows the result:

DROP VIEW

Query returned successfully in 80 msec.

Total rows: 0 Query complete 00:00:00.080