

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 top-left corner, it says "pgAdmin 4". The title bar indicates the connection is "lab\_8/postgres@PostgreSQL 17\*". The main area is a "Query" window containing the following SQL code:

```

1 create view specific_date as
2 select * from flights
3 where scheduled_departure = '2024-01-01'
4
5
6 select * from specific_date
    
```

Below the query window is a "Data Output" tab showing the results of the query. The results are as follows:

flight_id	flight_no	scheduled_departure	scheduled_arrival	departure_airport_id	arrival_airport_id	departing_gate	arriving_gate	airline_id	status	actual_departure	actual_date	
1	410	US-OK	2024-01-01	2023-10-03	10	6	2013	62	8	Delayed	2023-09-18	2023
2	730	US-AK	2024-01-01	2023-04-08	17	1	59	374	10	Boarding	2023-06-08	2024

The status bar at the bottom left shows "Total rows: 2 Query complete 00:00:00.129". The system tray at the bottom right shows the date and time as "16.11.2025".

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 top-left corner, it says "pgAdmin 4". The title bar indicates the connection is "lab\_8/postgres@PostgreSQL 17\*". The main area is a "Query" window containing the following SQL code:

```

1 CREATE VIEW bookings_next_week AS
2 SELECT
3     b.booking_id,
4     b.passenger_id,
5     b.status,
6     b.price,
7     f.flight_id,
8     f.scheduled_departure,
9     f.scheduled_arrival
10    FROM booking AS b
11    JOIN booking_flight AS bf
12      ON b.booking_id = bf.booking_id
13    JOIN flights AS f
14      ON bf.flight_id = f.flight_id
15    WHERE f.scheduled_departure >= NOW()
16      AND f.scheduled_departure < NOW() + INTERVAL '7 days';
17
18    SELECT flight_id, scheduled_departure
19    FROM flights
20    ORDER BY scheduled_departure desc;
21
    
```

The "Messages" tab below the query window shows "CREATE VIEW". The status bar at the bottom left shows "Query returned successfully in 71 msec." The system tray at the bottom right shows the date and time as "17.11.2025".

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

```

CREATE OR REPLACE VIEW top_5_routes AS
SELECT
    f.departure_airport_id,
    dep.airport_name AS departure_airport,
    f.arrival_airport_id,
    arr.airport_name AS arrival_airport,
    COUNT(DISTINCT b.booking_id) AS booking_count
FROM booking AS b
JOIN booking_flight AS bf
    ON b.booking_id = bf.booking_id
JOIN flights AS f
    ON bf.flight_id = f.flight_id
JOIN airport AS dep
    ON f.departure_airport_id = dep.airport_id
JOIN airport AS arr
    ON f.arrival_airport_id = arr.airport_id
GROUP BY
    f.departure_airport_id,
    dep.airport_name,
    f.arrival_airport_id,
    arr.airport_name
ORDER BY booking_count DESC

```

The screenshot shows the pgAdmin 4 interface with the SQL tab selected. A new view named 'top\_5\_routes' is being created. The code defines the view to select flight routes based on the number of bookings, joining four tables: booking, booking\_flight, flights, and airport. The results are grouped by departure and arrival airports and ordered by the count of bookings in descending order. The Data Output tab shows the top 5 results:

	departure_airport_id	departure_airport	arrival_airport_id	arrival_airport	booking_count
1	10	Henri Coandă International Airport	7	Armidale Airport	18
2	13	Figari Sud-Corse Airport	4	Garbaharey Airport	14
3	4	Garbaharey Airport	10	Henri Coandă International Airport	14
4	6	Hana Airport	16	Zephyrhills Municipal Airport	14
5	14	Industrial Airpark	7	Armidale Airport	11

Total rows: 5 Query complete 00:00:00.091

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

```

create or replace view specific_airline as
select
    f.flight_id,
    f.scheduled_departure,
    f.scheduled_arrival,
    f.departure_airport_id,
    f.arrival_airport_id,
    f.airline_id,
    a.airline_name,
    a.airline_code,
    a.airline_country
from flights as f
join airline as a
on f.airline_id = a.airline_id
where airline_name = 'IPC'
select * from specific_airline

```

The screenshot shows the pgAdmin 4 interface with the SQL tab selected. A new view named 'specific\_airline' is being created, specifically for the airline 'IPC'. The code joins the flights and airline tables to filter flights where the airline name is 'IPC'. The Data Output tab shows the results for 32 flights:

	flight_id	scheduled_departure	scheduled_arrival	departure_airport_id	arrival_airport_id	airline_id	airline_name	airline_code	airline_country
1	13	2024-01-16	2023-06-02	13	20	1	IPC	SCIP	Russia
2	33	2023-09-21	2023-11-29	4	19	1	IPC	SCIP	Russia
3	36	2023-05-29	2023-05-06	12	14	1	IPC	SCIP	Russia
4	73	2023-12-26	2023-04-30	18	9	1	IPC	SCIP	Russia
5	78	2023-08-28	2023-08-02	20	11	1	IPC	SCIP	Russia
6	143	2023-08-20	2023-10-24	12	9	1	IPC	SCIP	Russia

Total rows: 32 Query complete 00:00:00.116

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

```

pgAdmin 4
File Object Tools View Window Help
Welcome lab_8/postgres@PostgreSQL 17* x
lab_8/postgres@PostgreSQL 17
Query History
Scratch Pad x
Query
1 v create or replace view specific_airline as
2 select
3     f.flight_id,
4     f.scheduled_departure,
5     f.scheduled_arrival,
6     f.departure_airport_id,
7     f.arrival_airport_id,
8     f.airline_id,
9     a.airline_name,
10    a.airline_code,
11    a.airline_country
12   from flights as f
13  join airline as a
14    on f.airline_id = a.airline_id
15   where airline_name = 'IPC'
16     and scheduled_departure >= now()
17     and scheduled_departure < now() + interval '7 days';
18
19 select * from specific_airline
20
21
22
Data Output Messages Notifications
Flight ID | Scheduled Departure | Scheduled Arrival | Departure Airport ID | Arrival Airport ID | Airline ID | Airline Name | Airline Code | Airline Country
-----|-----|-----|-----|-----|-----|-----|-----|-----

```

Total rows: 0 Query complete 00:00:00.124 CRLF Ln 13, Col 31

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

```

pgAdmin 4
File Object Tools Edit View Window Help
Welcome lab_8/postgres@PostgreSQL 17* x
lab_8/postgres@PostgreSQL 17
Query History
Scratch Pad x
Query
1 v create or replace view flight_24 as
2 select * from flights
3 where actual_departure::timestamp - scheduled_departure::timestamp > interval '24 hours';
4
5
6 select * from flight_24
Data Output Messages Notifications
Showing rows: 1 to 494 Page No: 1 of 1
Flight ID | Flight No | Scheduled Departure | Scheduled Arrival | Departure Airport ID | Arrival Airport ID | Departing Gate | Arriving Gate | Airline ID | Status | Actual Departure | Act Date
-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----
1 | 2 | US-NM | 2023-07-21 | 2023-09-17 | 13 | 16 | 4216 | 90 | 34 | Boarding | 2024-02-09 | 20
2 | 3 | FI-OL | 2023-03-29 | 2023-08-01 | 18 | 12 | 47 | 9 | 34 | Boarding | 2024-02-21 | 20
3 | 5 | RO-DJ | 2023-07-03 | 2023-11-28 | 6 | 2 | 626 | 171 | 14 | Check-in open | 2023-11-18 | 20
4 | 6 | CA-SK | 2023-07-07 | 2023-09-11 | 18 | 1 | 1576 | 72 | 34 | Check-in open | 2024-02-19 | 20
5 | 7 | AU-TAS | 2023-10-12 | 2024-02-24 | 15 | 18 | 867 | 15 | 10 | Delayed | 2023-12-04 | 20
6 | 9 | IN-OR | 2023-05-18 | 2023-09-19 | 6 | 9 | 659 | 621 | 13 | Delayed | 2023-06-17 | 20
Total rows: 494 Query complete 00:00:00.127 CRLF Ln 6, Col 24

```

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.

```

pgAdmin 4
File Object Tools Edit View Window Help
Welcome lab_8/postgres@PostgreSQL 17* 
lab_8/postgres@PostgreSQL 17
Query History
Scratch Pad
Query
1 v create or replace view Leffler_Thompson as
2 select p.first_name, p.last_name, p.country_of_citizenship, b.booking_platform
3 from passengers as p
4 join booking as b
5 on p.passenger_id = b.passenger_id
6 where booking_platform = 'Leffler-Thompson'
7
8
9
10
11
12 select * from Leffler_Thompson

```

The screenshot shows the pgAdmin 4 interface with a query window containing SQL code to create a view named 'Leffler\_Thompson'. The code selects the first name, last name, and country of citizenship of passengers who have made bookings on the 'Leffler-Thompson' platform. After running the query, a data output window shows a single row with the following data:

	first_name	last_name	country_of_citizenship	booking_platform
1	Philbert	Shambroke	Colombia	Leffler-Thompson

The status bar at the bottom indicates 'Total rows: 1 Query complete 00:00:00.104' and a green message 'Successfully run. Total query runtime: 104 msec. 1 rows affected.'

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

```

pgAdmin 4
File Object Tools Edit View Window Help
Welcome lab_8/postgres@PostgreSQL 17* 
lab_8/postgres@PostgreSQL 17
Query History
Scratch Pad
Query
1 v create or replace view top_10_countries as
2 select ap.country, count(*) as booking_count
3 from booking as b
4 join booking_flight as bf on b.booking_id = bf.booking_id
5 join flights as f on bf.flight_id = f.flight_id
6 join airport as ap on f.arrival_airport_id = ap.airport_id
7
8 group by ap.country
9 order by booking_count desc
10
11
12
13

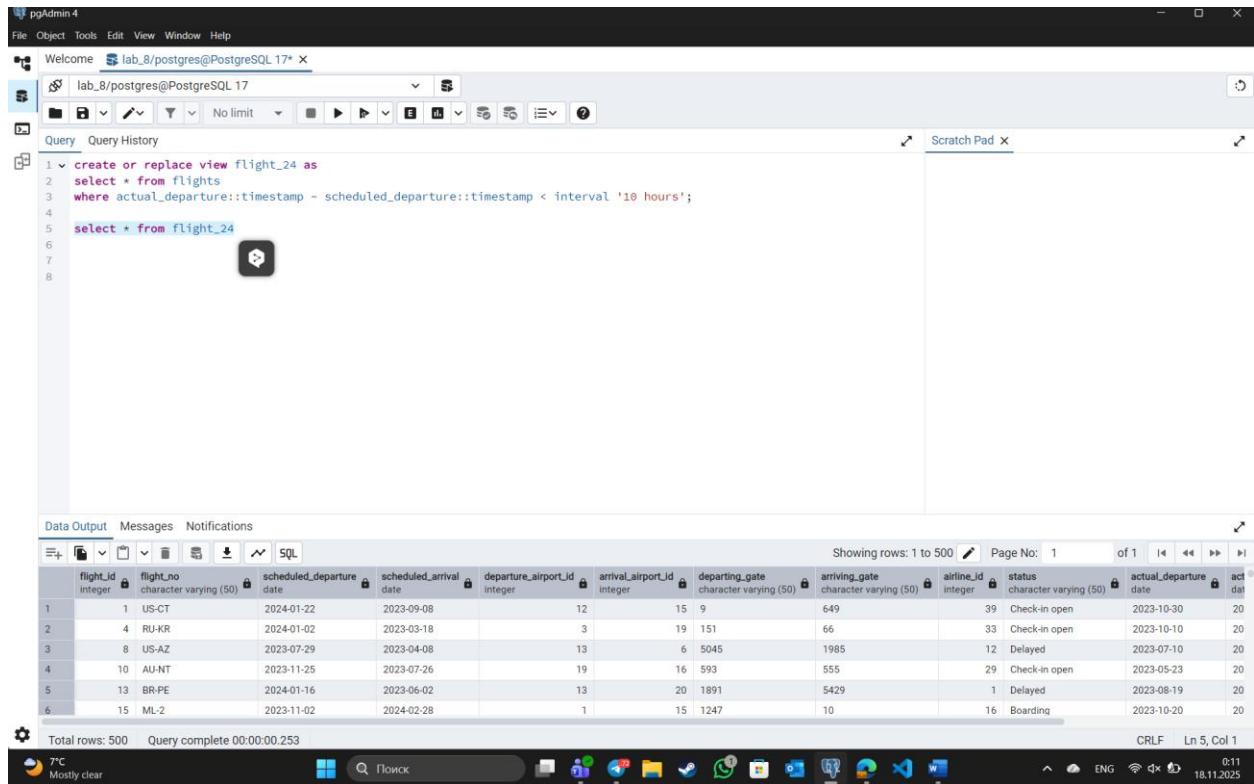
```

The screenshot shows the pgAdmin 4 interface with a query window containing SQL code to create a view named 'top\_10\_countries'. The code selects the country and the count of bookings for each country, grouping by country and ordering by the count in descending order. After running the query, a data output window shows the top 10 countries with their respective booking counts:

	country	booking_count
1	China	232
2	Indonesia	167
3	Philippines	137
4	Tanzania	78
5	United States	62
6	Russia	59
...	...	...

The status bar at the bottom indicates 'Total rows: 12 Query complete 00:00:00.092' and a green message 'Successfully run. Total query runtime: 92 msec. 12 rows affected.'

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

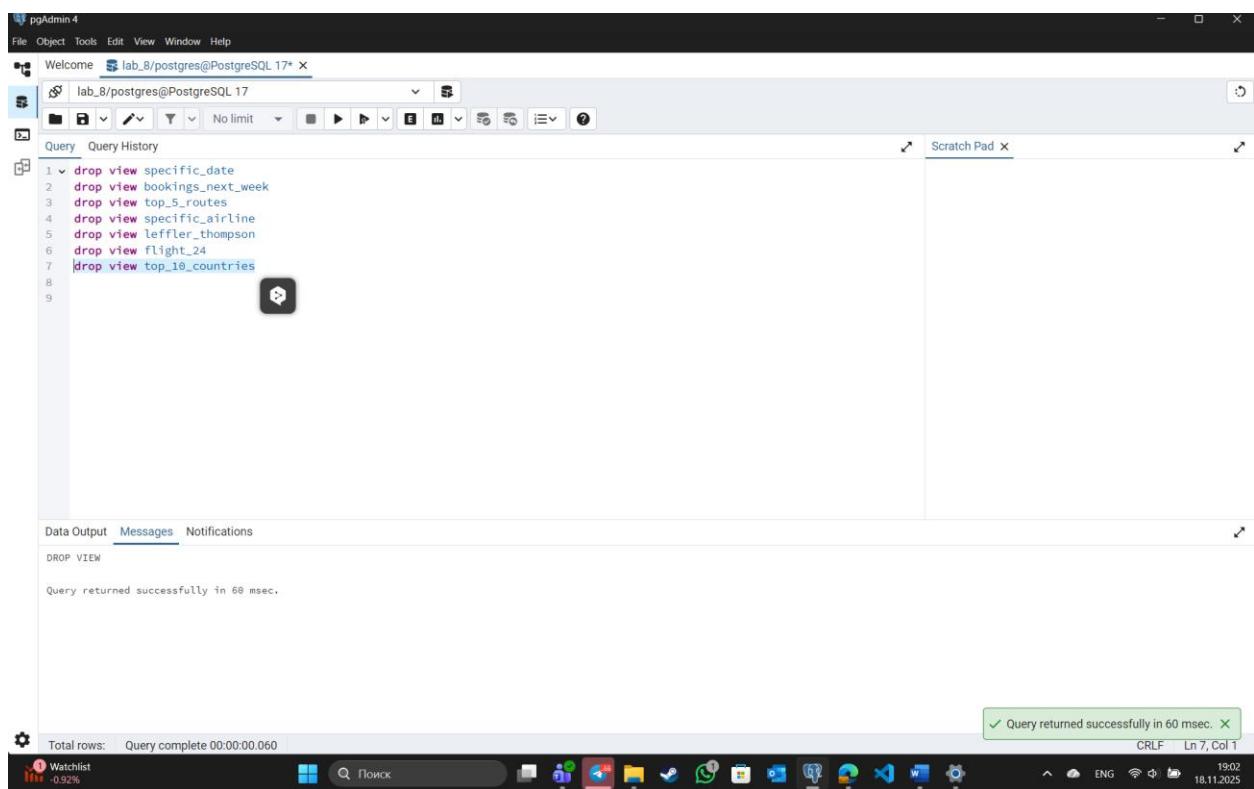


The screenshot shows the pgAdmin 4 interface. In the top-left pane, there is a query editor window with the following SQL code:

```
1 ✓ create or replace view flight_24 as
2 select * from flights
3 where actual_departure::timestamp - scheduled_departure::timestamp < interval '10 hours';
4
5 select * from flight_24
6
7
8
```

In the bottom-right pane, there is a data output grid showing flight information. The columns are: flight\_id, flight\_no, scheduled\_departure, scheduled\_arrival, departure\_airport\_id, arrival\_airport\_id, departing\_gate, arriving\_gate, airline\_id, status, actual\_departure, and act\_date. The data includes rows for various flights like US-GT, RU-KR, US-AZ, AU-NT, BR-PE, and ML-2.

## 6. Drop all existing views.



The screenshot shows the pgAdmin 4 interface. In the top-left pane, there is a query editor window with the following SQL code:

```
1 ✓ drop view specific_date
2 drop view bookings_next_week
3 drop view top_5_routes
4 drop view specific_airline
5 drop view leffler_thompson
6 drop view flight_24
7 drop view top_10_countries
8
9
```

In the bottom-right pane, there is a data output grid showing the results of the drop command. It displays the message "DROP VIEW" and "Query returned successfully in 60 msec." A green success message box is visible at the bottom right.