

Laboratory work 8

VIEW.

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

The screenshot shows the pgAdmin 4 interface. On the left is a tree view of database objects under 'flights'. In the center is a query editor window titled 'Query History' containing the following SQL code:

```
1 CREATE VIEW flights_by_date AS
2   SELECT flight_id,
3         flight_no,
4         scheduled_departure,
5         scheduled_arrival,
6         departure_airport_id,
7         arrival_airport_id,
8         airline_id,
9         status,
10        actual_departure,
11        actual_arrival,
12        created_at,
13        update_at
14   FROM flights;
```

Below the query editor, the 'Data Output' tab is selected, showing the message: 'CREATE VIEW'. At the bottom of the screen, a green status bar indicates: '✓ Query returned successfully in 54 msec. LF Ln 14, Col 14'.

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

The screenshot shows the pgAdmin interface with the following details:

- Object Navigator:** On the left, it lists various database objects including tables (fligf, dep, arriv), schemas (sch), and other database-related items.
- Query Editor:** The main area contains the SQL code for creating a view:


```

CREATE VIEW bookings_next_week AS
SELECT bf.booking_id,
bf.flight_id,
f.scheduled_departure,
f.scheduled_arrival,
f.departure_airport_id,
f.arrival_airport_id
FROM booking_flight bf JOIN flights f ON bf.flight_id = f.flight_id
WHERE f.scheduled_departure BETWEEN CURRENT_DATE AND CURRENT_DATE + INTERVAL '1 week'
      
```
- Data Output:** Below the query editor, it shows the message "Query returned successfully in 39 msec."
- Status Bar:** At the bottom right, it says "Query returned successfully in 39 msec. LF Ln 6, Col 20".

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

The screenshot shows the pgAdmin interface with the following details:

- Object Navigator:** On the left, it lists various database objects including tables (fligf, dep, arriv), schemas (sch), and other database-related items.
- Query Editor:** The main area contains the SQL code for creating a view:


```

CREATE VIEW top5_routes AS
SELECT
departure_airport_id,
arrival_airport_id,
COUNT(*) AS total_bookings
FROM booking_flight bf
JOIN flights f ON bf.flight_id = f.flight_id
GROUP BY departure_airport_id, arrival_airport_id
ORDER BY total_bookings DESC
LIMIT 5;
      
```
- Data Output:** Below the query editor, it shows the message "Query returned successfully in 53 msec."
- Status Bar:** At the bottom right, it says "Query returned successfully in 53 msec. LF Ln 3, Col 14".

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

The screenshot shows the pgAdmin 4 interface. On the left is the Object Browser tree, which includes a 'PostgreSQL FastAPI' node, a 'Local PostgreSQL' node, and a 'Databases (10)' node. Under 'Databases (10)', there is a 'database_subj' node, which contains 'Tables (10)' and an 'airline' table. The 'airline' table has six columns: 'airline_id', 'airline_code', 'airline_name', 'airline_count', and 'created_at'. The main window shows a query editor with the following SQL code:

```

1 CREATE VIEW flights_by_airlines AS
2 SELECT f.*, a.airline_name
3 FROM flights f
4 JOIN airline a ON f.airline_id = a.airline_id
5 WHERE a.airline_name = 'IPC';

```

The 'Messages' tab below the query editor shows the message: 'Query returned successfully in 74 msec.' A status bar at the bottom indicates 'Total rows: 0' and 'Query complete 00:00:00.074'.

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 with the same database structure as the previous screenshot. The query editor now contains the following modified SQL code:

```

1 CREATE VIEW flights_by_airlines AS
2 SELECT f.*, a.airline_name
3 FROM flights f
4 JOIN airline a ON f.airline_id = a.airline_id
5 WHERE a.airline_name = 'IPC'
6 AND f.scheduled_departure BETWEEN CURRENT_DATE
7 AND CURRENT_DATE + INTERVAL '7 days';

```

The 'Messages' tab shows the message: 'Query returned successfully in 37 msec.' A status bar at the bottom indicates 'Total rows: 0' and 'Query complete 00:00:00.037'.

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

The screenshot shows the pgAdmin 4 interface. On the left is a tree view of the database structure under 'PostgreSQL FastAPI(2)'. In the center is a query editor window titled 'database_subj/postgres@Local PostgreSQL'. The query is:

```

1 CREATE VIEW delayed_24h AS
2 SELECT *
3 FROM flights
4 WHERE actual_departure IS NOT NULL
5     AND actual_departure > scheduled_departure + INTERVAL '24 hours';

```

Below the query editor, the status bar shows 'Query returned successfully in 43 msec.' and 'Query complete 00:00:00.043'.

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. On the left is a tree view of the database structure under 'PostgreSQL FastAPI(2)'. In the center is a query editor window titled 'database_subj/postgres@Local PostgreSQL'. The query is:

```

1 CREATE VIEW leffler_passengers AS
2 SELECT
3     p.passenger_id,
4     p.first_name || ' ' || p.last_name AS full_name,
5     p.country_of_citizenship AS country,
6     b.booking_platform
7 FROM passengers p
8 JOIN booking b ON p.passenger_id = b.passenger_id
9 WHERE b.booking_platform = 'Leffler-Thompson';

```

Below the query editor, the status bar shows 'Query returned successfully in 41 msec.' and 'Query complete 00:00:00.041'.

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

The screenshot shows the pgAdmin interface with the database structure on the left and a query editor on the right. The query editor contains the following SQL code:

```

1 CREATE VIEW top_10_visited_countries AS
2 SELECT
3     a.country AS arrival_country,
4     COUNT(*) AS flights_count,
5     CURRENT_DATE AS created_at,
6     CURRENT_DATE AS update_at
7 FROM flights f
8 JOIN airport a
9      ON f.arrival_airport_id = a.airport_id
10 GROUP BY a.country
11 ORDER BY flights_count DESC
12 LIMIT 10;

```

The message bar at the bottom right indicates "Query returned successfully in 56 msec." and "Ln 12, Col 10".

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

The screenshot shows the pgAdmin interface with the database structure on the left and a query editor on the right. The query editor contains the following SQL code:

```

1 CREATE OR REPLACE VIEW flights_ipc_simple AS
2 SELECT
3     flight_id,
4     flight_no,
5     airline_id,
6     scheduled_departure,
7     scheduled_arrival,
8     CURRENT_DATE AS created_at,
9     CURRENT_DATE AS update_at
10    FROM flights
11   WHERE airline_id = 1;

```

The message bar at the bottom right indicates "Query returned successfully in 46 msec." and "Ln 11, Col 22".

10. Drop all existing views.

The screenshot shows the pgAdmin 4 interface with a query editor window open. The left sidebar displays a tree view of database objects, including schemas like 'public', 'airline', and 'subj', and tables like 'air', 'airp', 'cou', 'stat', 'city', 'crea', 'upd', 'flights', 'fligt', 'sch', 'dep', 'arriv', and 'depl'. The main window contains the following SQL code:

```
1 DROP VIEW IF EXISTS
2   flights_by_date,
3   bookings_next_week,
4   top5_routes,
5   flights_by_airline,
6   delayed_24h,
7   leffler_passengers,
8   top10_countries;
```

The status bar at the bottom indicates "Query complete 00:00:00.041".