

Laboratory work 8

VIEW.

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

The screenshot shows a database IDE with the following components:

- Database Explorer:** Shows a PostgreSQL database named 'airport_db' with a 'public' schema containing tables like 'airline', 'airport', 'baggage', 'boarding_pass', 'booking', 'booking_flight', 'flights', 'passengers', 'security_check', and 'views'.
- Console:** Contains the SQL code to create a view:

```
1 CREATE VIEW Flights_By_Date AS
2 SELECT *
3 FROM Flights
4 WHERE actual_departure = '2023-05-23';
5
```
- Output:** Displays the results of the view, showing 4 rows of flight data with columns: flight_id, flight_no, scheduled_departure, scheduled_arrival, departure_airport_id, and arrival_airport_id.

flight_id	flight_no	scheduled_departure	scheduled_arrival	departure_airport_id	arrival_airport_id
10	AU-NT	2023-11-25	2023-07-26	19	1
227	UG-212	2023-12-24	2023-04-18	16	1
494	ML-2	2023-04-23	2023-03-31	14	1
639	EC-S	2024-03-08	2023-06-20	20	1

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

The screenshot shows a database IDE with the following components:

- Database Explorer:** Shows the same PostgreSQL database 'airport_db' with the 'public' schema. The 'booking' table is highlighted.
- Console:** Contains the SQL code to create a view:

```
1 CREATE VIEW Bookings_Next_Week AS
2 SELECT b.*
3 FROM Booking b
4 JOIN Flights f ON b.booking_id = f.flight_id
5 WHERE f.actual_departure BETWEEN CURRENT_DATE AND CURRENT_DATE + INTERVAL '7 days';
6
```
- Output:** Displays the results of the view, showing 0 rows of booking data with columns: booking_id, passenger_id, booking_platform, created_at, update_at, status, price, and ticket_dis.

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

The screenshot shows a database IDE with the following components:

- Database Explorer:** Shows the database structure for 'postgres@localhost'. The 'public' schema contains tables: 'airline', 'airport', 'baggage', 'baggage_check', 'boarding_pass', 'booking', 'booking_flight', 'flights', 'passengers', 'security_check', and 'views'.
- Console:** Contains the SQL code to create the view 'Top_5_Popular_Routes'.
- Output:** Displays the execution results of the SQL code.
- Services:** Shows the execution timeline for the 'console' service.

SQL Code:

```
1 CREATE VIEW Top_5_Popular_Routes AS
2 SELECT
3     f.country_of_residence,
4     f.country_of_citizenship,
5     COUNT(b.booking_id) AS booking_count
6 FROM passengers f
7 JOIN booking b ON f.passenger_id = b.booking_id
8 GROUP BY f.country_of_residence, f.country_of_citizenship
9 ORDER BY booking_count DESC
10 LIMIT 5;
```

Execution Results:

country_of_residence	country_of_citizenship	booking_count
China	China	16
Indonesia	China	6
Philippines	China	4
Philippines	Canada	3
Russia	China	3

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

The screenshot shows a database IDE with the following components:

- Database Explorer:** Shows the database structure for 'postgres@localhost'. The 'public' schema contains tables: 'airline', 'airport', 'baggage', 'baggage_check', 'boarding_pass', 'booking', 'booking_flight', 'flights', 'passengers', 'security_check', and 'views'.
- Console:** Contains the SQL code to create the view 'Flights.ByAirline'.
- Output:** Displays the execution results of the SQL code.
- Services:** Shows the execution timeline for the 'console' service.

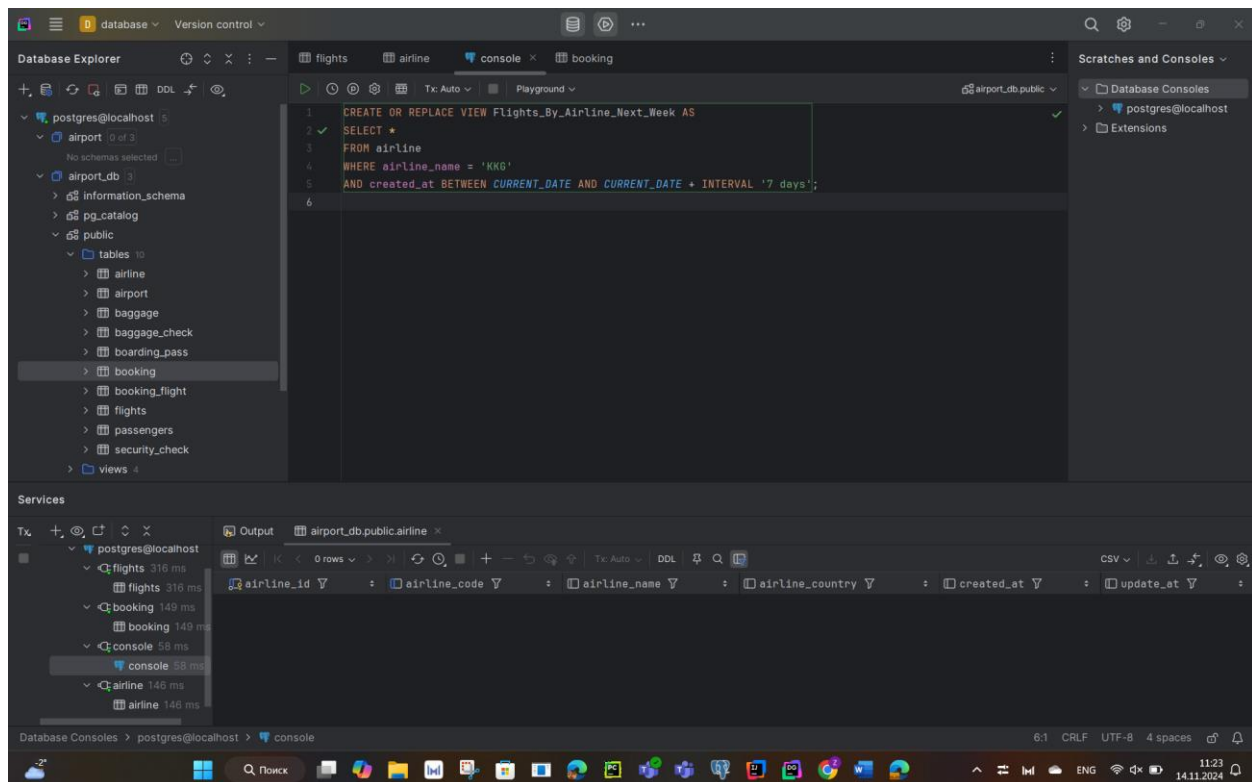
SQL Code:

```
1 CREATE VIEW Flights.ByAirline AS
2 SELECT *
3 FROM airline
4 WHERE airline_name = 'KKG';
```

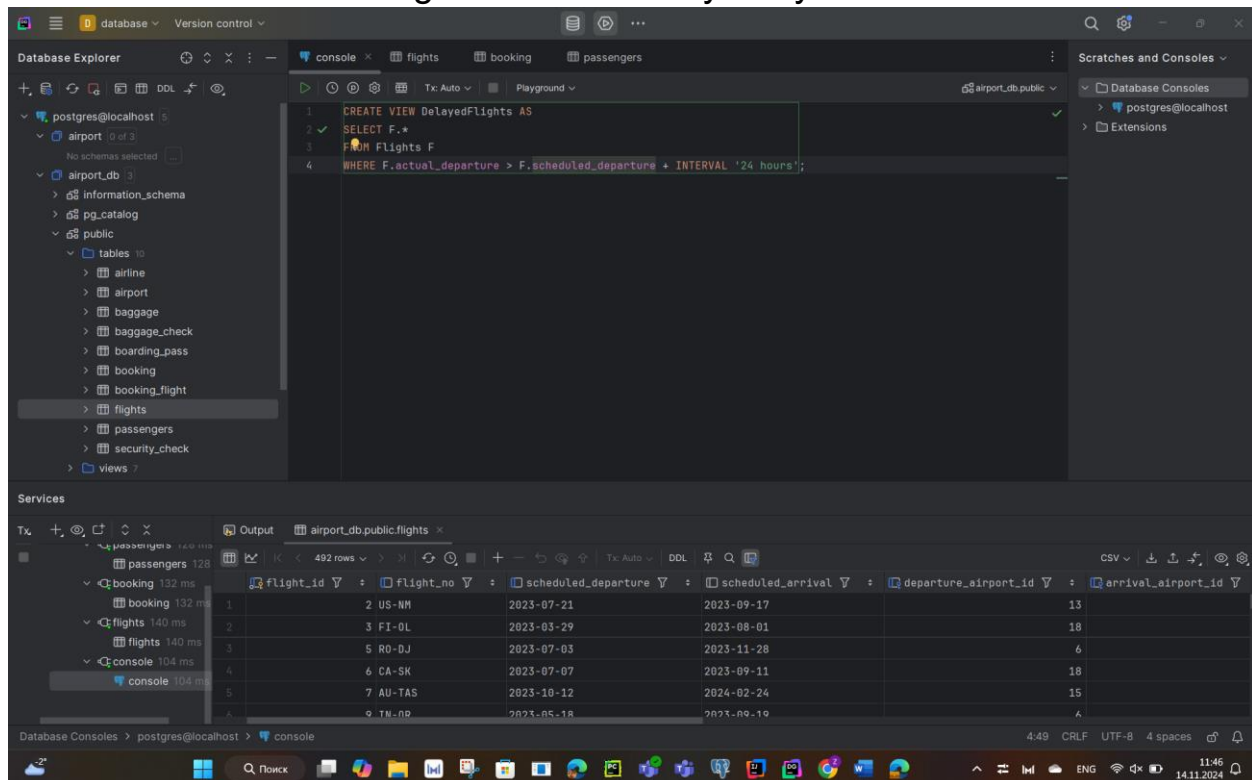
Execution Results:

airline_id	airline_code	airline_name	airline_country	created_at	update_at
15	SYKZ	KKG	Poland	2023-12-24	2023-10-30

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



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



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.

Database Explorer

- postgres@localhost
 - airport (0 of 3)
 - airport_db (3)
 - information_schema
 - pg_catalog
 - public
 - tables (10)
 - airline
 - airport
 - baggage
 - baggage_check
 - boarding_pass
 - booking
 - booking_flight
 - flights
 - passengers
 - security_check
 - views (5)

console

```

1 CREATE VIEW Passengers_Leffler_Thompson AS
2 SELECT
3   CONCAT(p.first_name, ' ', p.last_name) AS full_name,
4   p.country_of_citizenship
5 FROM Passengers p
6 JOIN Booking b ON p.passenger_id = b.passenger_id
7 JOIN Flights f ON b.booking_id = f.flight_id
8 WHERE b.booking_platform = 'Leffler-Thompson';
9 SELECT * FROM Passengers_Leffler_Thompson;
10

```

Services

Output: airport_db.public.pa...gers_leffler_thompson

full_name	country_of_citizenship
Philbert Shambroke	Colombia

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

Database Explorer

- postgres@localhost
 - airport (0 of 3)
 - airport_db (3)
 - information_schema
 - pg_catalog
 - public
 - tables (10)
 - airline
 - airport
 - baggage
 - baggage_check
 - boarding_pass
 - booking
 - booking_flight
 - flights
 - passengers
 - security_check
 - views (6)

console

```

1 CREATE VIEW Top_10_Visited_Countries AS
2 SELECT b.country_of_residence, COUNT(b.passenger_id) AS visit_count
3 FROM Flights f
4 JOIN passengers b ON f.flight_id = b.passenger_id
5 GROUP BY b.country_of_residence
6 ORDER BY visit_count DESC
7 LIMIT 10;
8

```

Services

Output: Result 22

country_of_residence	visit_count
China	40
Philippines	16
Indonesia	15
Russia	14
Brazil	9
Portugal	8

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

The screenshot shows a database IDE with a 'Database Explorer' on the left, a 'console' window in the center, and a 'Results' window at the bottom. The 'console' window contains the following SQL code:

```
1 CREATE OR REPLACE VIEW Flights_By_Date AS
2 SELECT f.*, f.flight_id
3 FROM Flights f
4 WHERE f.actual_departure = '2023-10-10';
5
```

The 'Results' window displays a table with 4 rows and 7 columns. The columns are: f.flight_id, flight_no, scheduled_departure, scheduled_arrival, departure_airport_id, and arrival_airport_id. The data is as follows:

f.flight_id	flight_no	scheduled_departure	scheduled_arrival	departure_airport_id	arrival_airport_id
4	RU-KR	2024-01-02	2023-03-18	3	
391	MH-NMU	2024-03-01	2023-06-05	7	
540	US-TX	2023-07-31	2023-05-28	9	
836	ET-SO	2023-10-14	2023-03-30	16	

10. Drop all existing views.

The screenshot shows the same database IDE as before, but the 'console' window now contains the following SQL code:

```
1 DROP VIEW IF EXISTS Flights_By_Date,
2 Bookings_Next_Week,
3 Top_5_Popular_Routes,
4 Flights_By_Airline,
5 Flights_By_Airline_Next_Week,
6 DelayedFlights,
7 Passengers_Leffler_Thompson,
8 Top_10_Visited_Countries;
9
```

The 'Results' window shows the output of the command, indicating that the views were successfully dropped. The output is as follows:

```
[2024-11-14 11:51:11] completed in 16 ms
airport_db.public> DROP VIEW IF EXISTS Flights_By_Date,
Bookings_Next_Week,
Top_5_Popular_Routes,
Flights_By_Airline,
Flights_By_Airline_Next_Week,
DelayedFlights,
Passengers_Leffler_Thompson,
Top_10_Visited_Countries
представление "Flights_by_date" не существует, пропускается
```