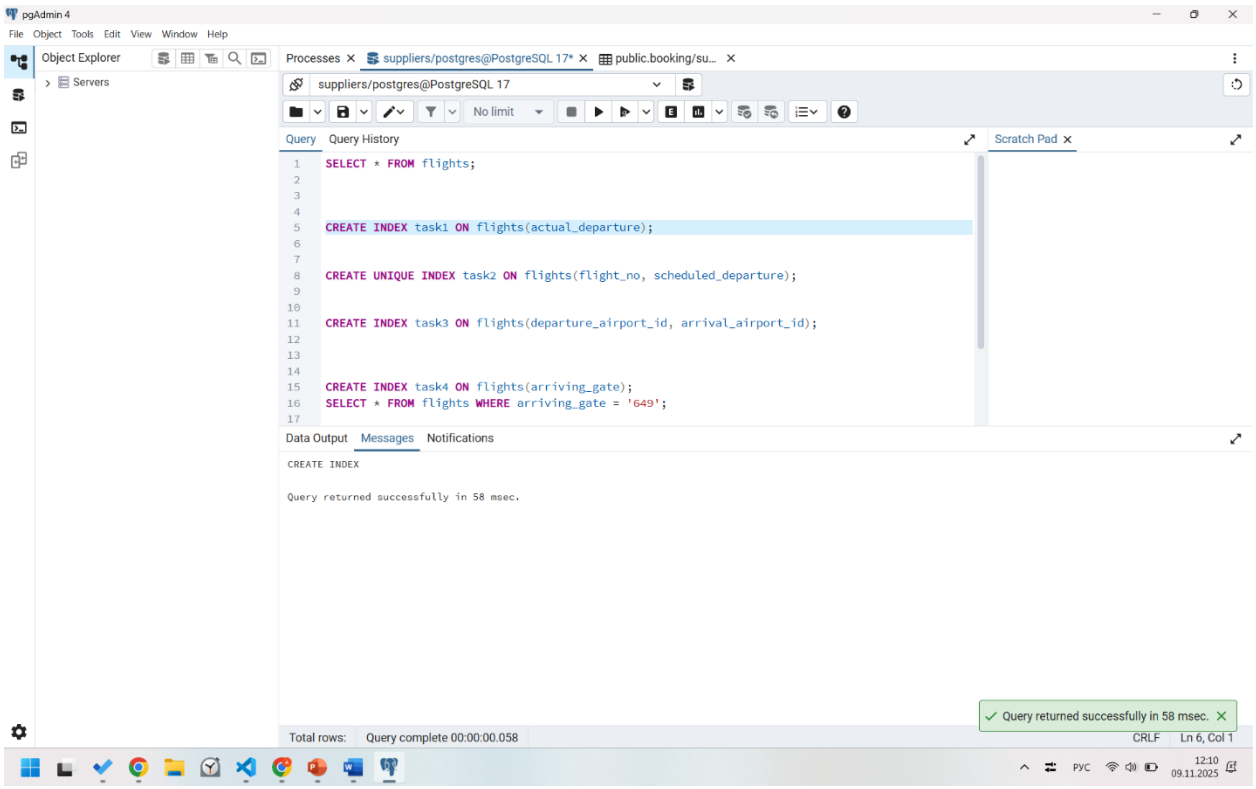
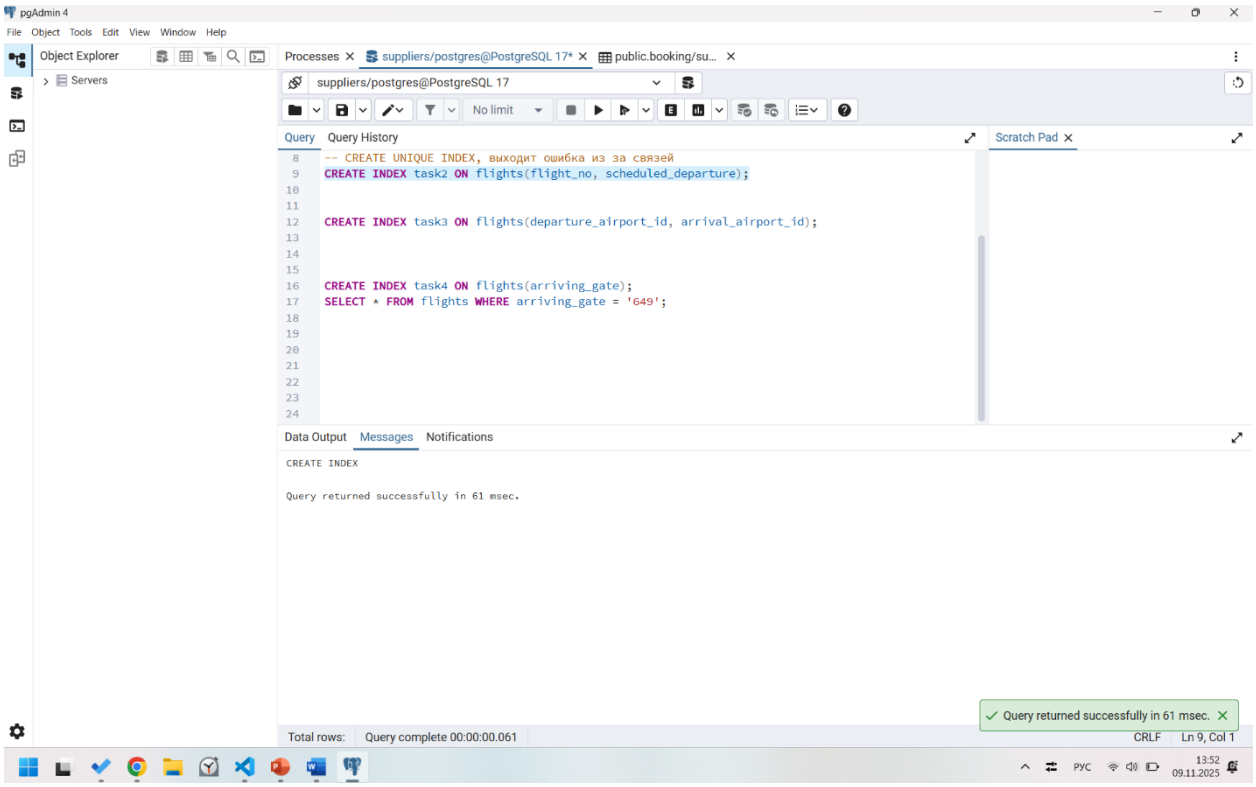
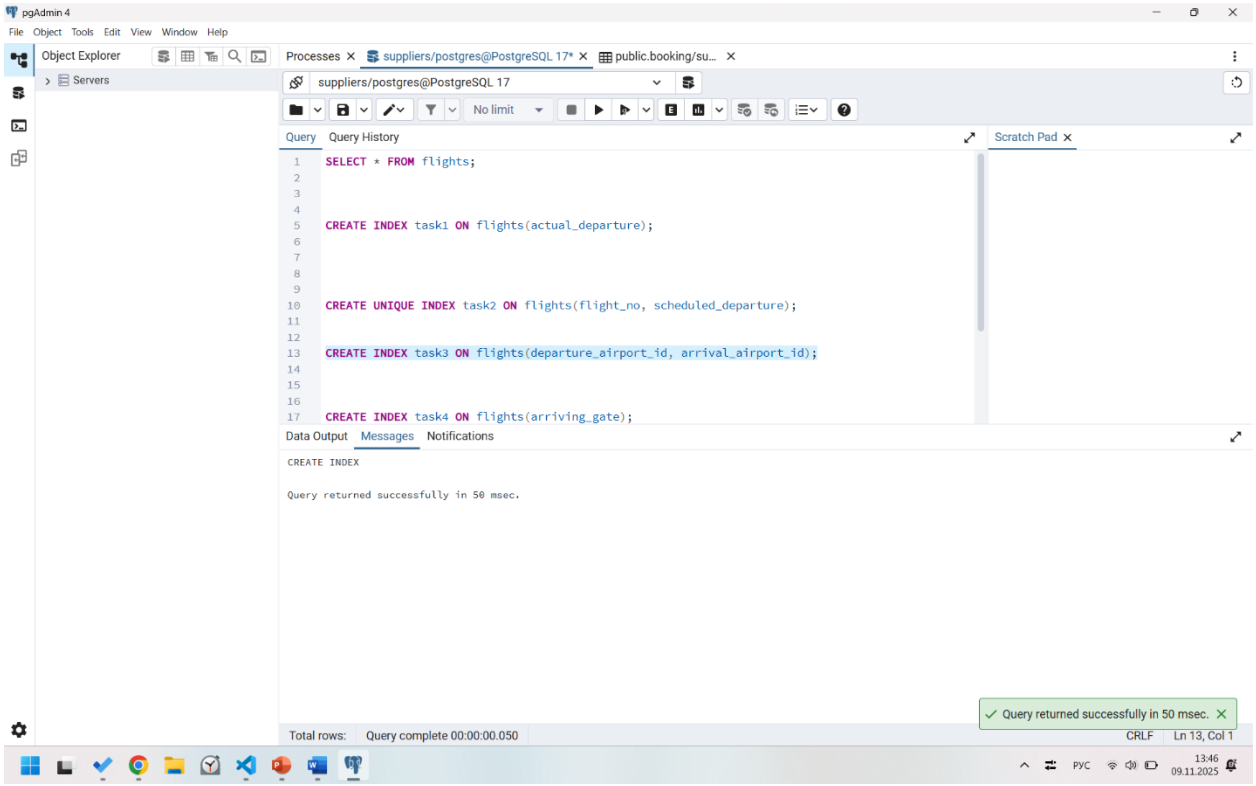
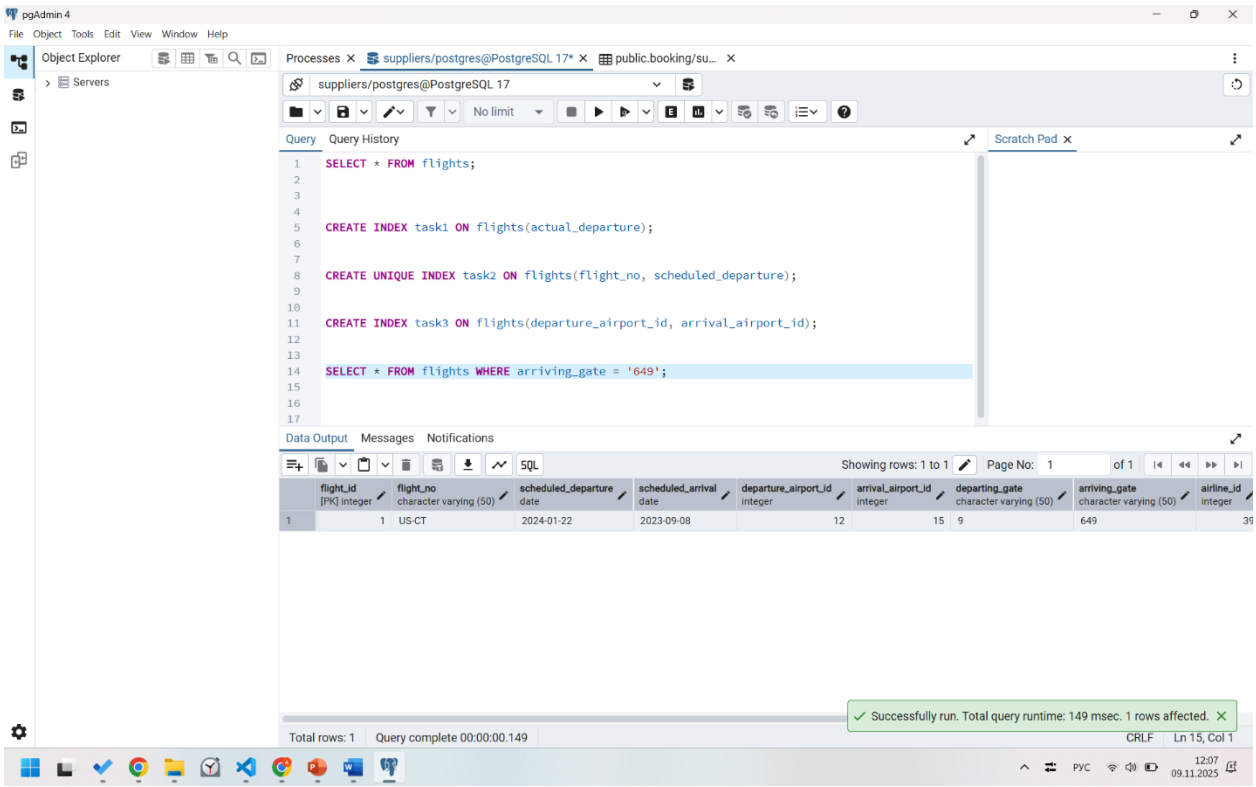
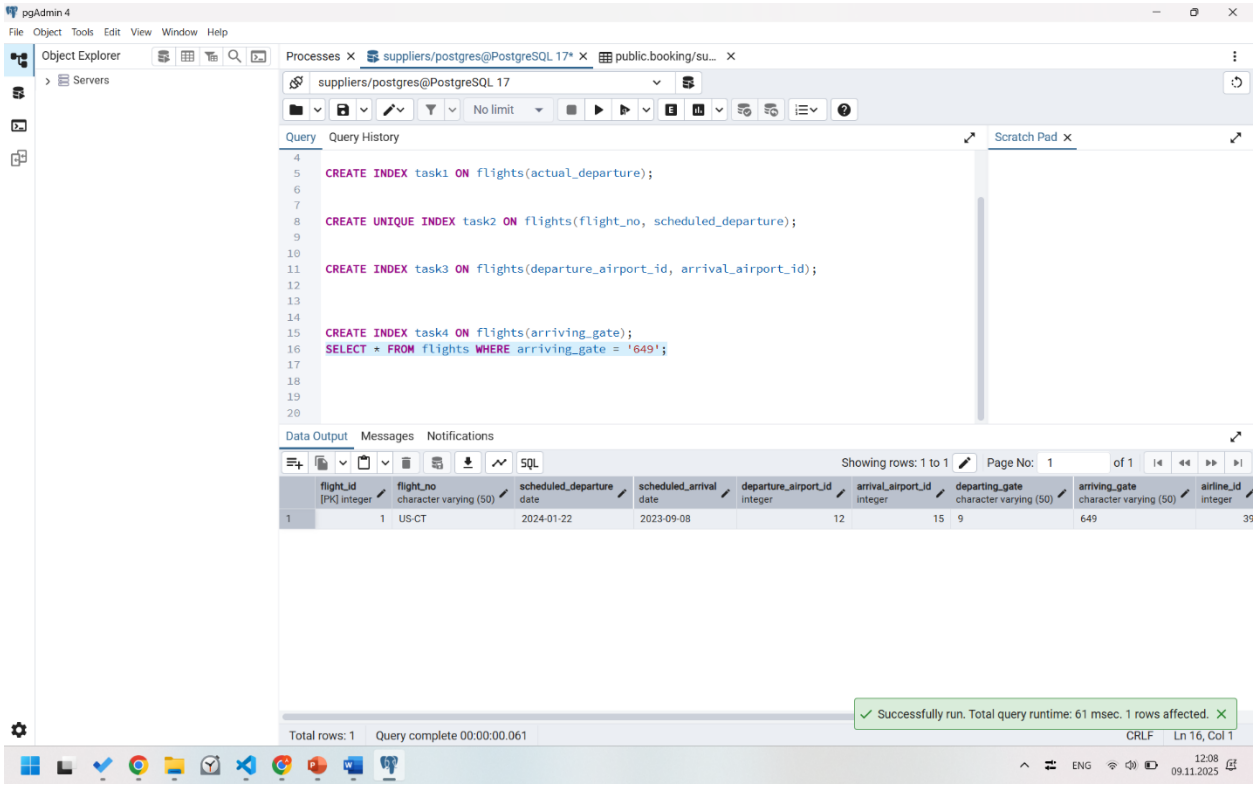
Lab 07

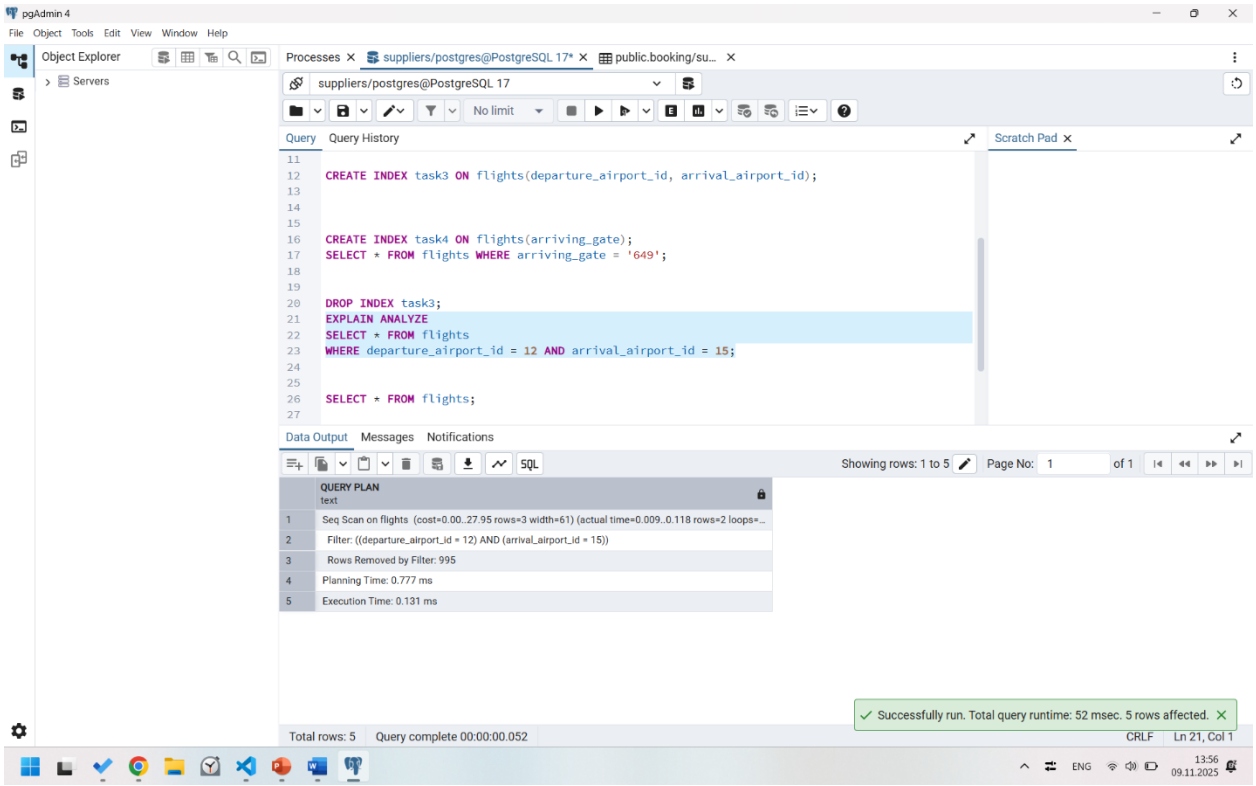

pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Servers
Processes
suppliers/postgres@PostgreSQL 17*
public.booking/su...
suppliers/postgres@PostgreSQL 17
No limit
Query
Query History
Scratch Pad
SELECT * FROM flights;
CREATE INDEX task1 ON flights(actual_departure);
CREATE UNIQUE INDEX task2 ON flights(flight_no, scheduled_departure);
CREATE INDEX task3 ON flights(departure_airport_id, arrival_airport_id);
CREATE INDEX task4 ON flights(arriving_gate);
SELECT * FROM flights WHERE arriving_gate = '649';
Data Output
Messages
Notifications
CREATE INDEX
Query returned successfully in 58 msec.
Total rows:
Query complete 00:00:00.058
Query returned successfully in 58 msec.
CRLF
Ln 6, Col 1
РУС
12:10
09.11.2025


pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Servers
Processes
suppliers/postgres@PostgreSQL 17*
public.booking/su...
suppliers/postgres@PostgreSQL 17
No limit
Query
Query History
Scratch Pad
-- CREATE UNIQUE INDEX, выходит ошибка из за связей
CREATE INDEX task2 ON flights(flight_no, scheduled_departure);
CREATE INDEX task3 ON flights(departure_airport_id, arrival_airport_id);
CREATE INDEX task4 ON flights(arriving_gate);
SELECT * FROM flights WHERE arriving_gate = '649';
Data Output
Messages
Notifications
CREATE INDEX
Query returned successfully in 61 msec.
Total rows:
Query complete 00:00:00.061
Query returned successfully in 61 msec.
CRLF
Ln 9, Col 1
РУС
13:52
09.11.2025


pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Servers
Processes
suppliers/postgres@PostgreSQL 17*
public.booking/su...
suppliers/postgres@PostgreSQL 17
No limit
Query
Query History
Scratch Pad
1 SELECT * FROM flights;
5 CREATE INDEX task1 ON flights(actual_departure);
10 CREATE UNIQUE INDEX task2 ON flights(flight_no, scheduled_departure);
13 CREATE INDEX task3 ON flights(departure_airport_id, arrival_airport_id);
17 CREATE INDEX task4 ON flights(arriving_gate);
Data Output
Messages
Notifications
CREATE INDEX
Query returned successfully in 50 msec.
Query returned successfully in 50 msec.
Total rows:
Query complete 00:00:00.050
CRLF
Ln 13, Col 1
13:46
09.11.2025

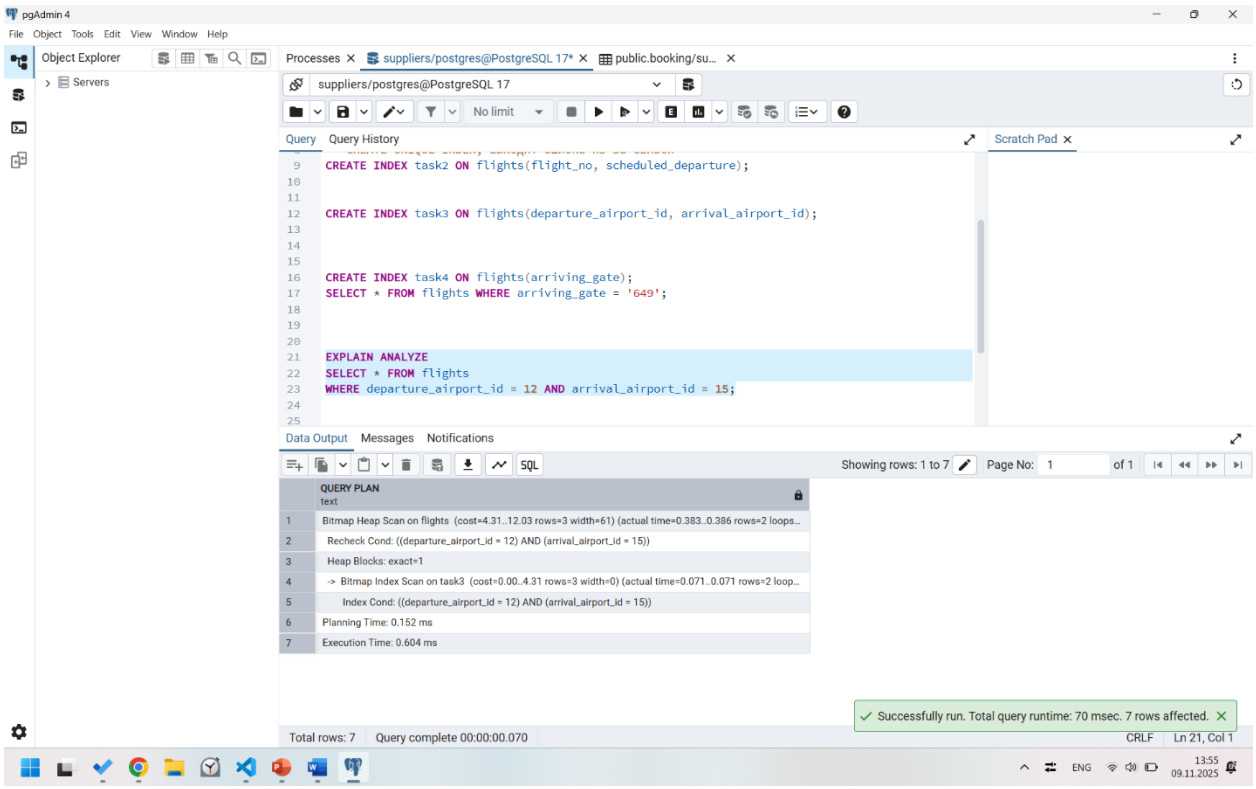

pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Servers
Processes
suppliers/postgres@PostgreSQL 17*
public.booking/su...
suppliers/postgres@PostgreSQL 17
No limit
Query
Query History
Scratch Pad
1 SELECT * FROM flights;
5 CREATE INDEX task1 ON flights(actual_departure);
8 CREATE UNIQUE INDEX task2 ON flights(flight_no, scheduled_departure);
11 CREATE INDEX task3 ON flights(departure_airport_id, arrival_airport_id);
14 SELECT * FROM flights WHERE arriving_gate = '649';
Data Output
Messages
Notifications
Showing rows: 1 to 1
Page No: 1
of 1
flight_id [PK] integer
flight_no character varying (50)
scheduled_departure date
scheduled_arrival date
departure_airport_id integer
arrival_airport_id integer
departing_gate character varying (50)
arriving_gate character varying (50)
airline_id integer
1 1 US-CT 2024-01-22 2023-09-08 12 15 9 649
Successfully run. Total query runtime: 149 msec. 1 rows affected.
Total rows: 1
Query complete 00:00:00.149
CRLF
Ln 15, Col 1
12:07
09.11.2025



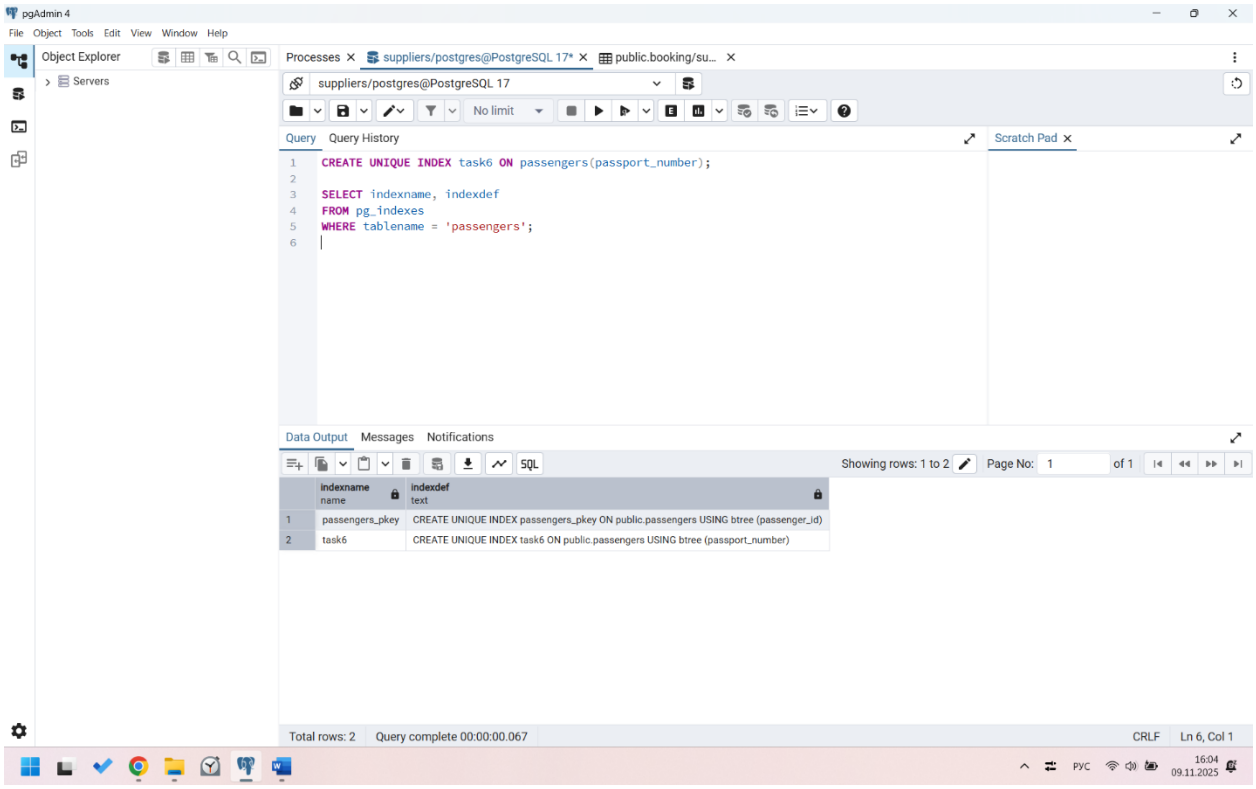


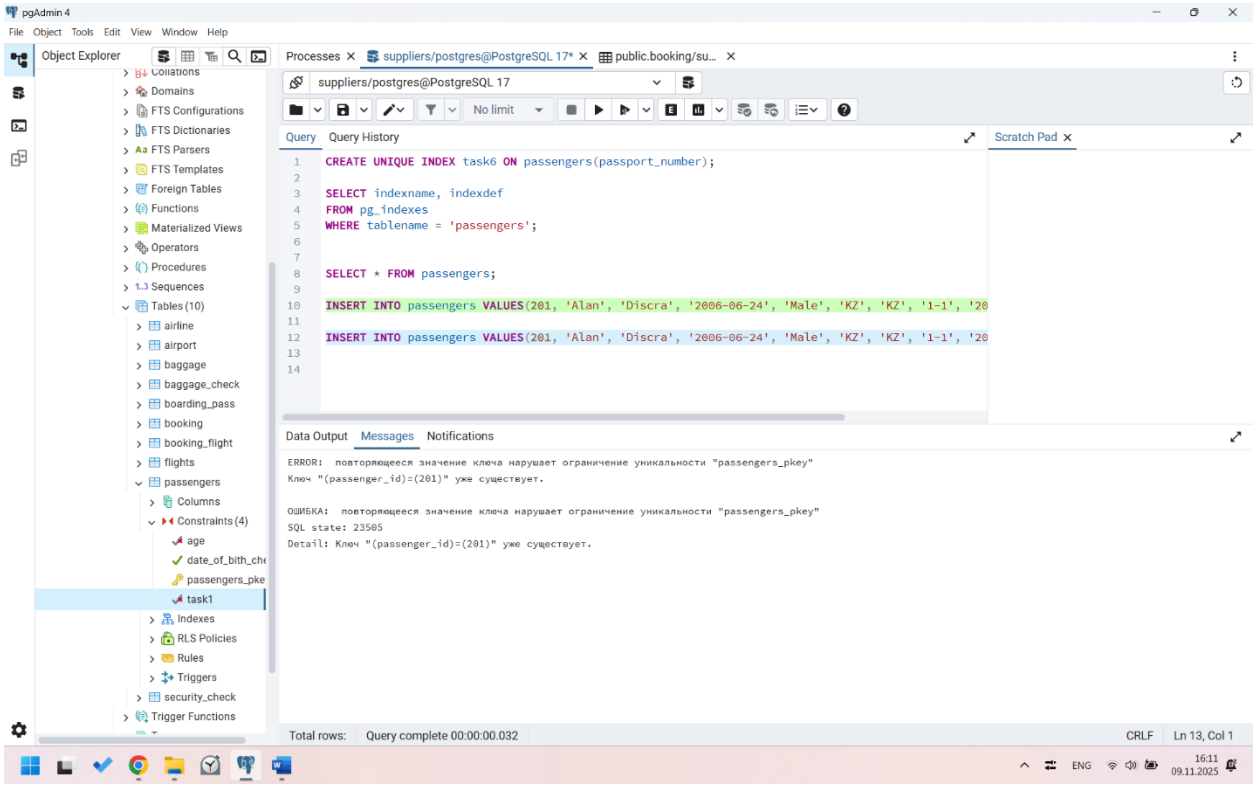
Without index:
Query time = 61 ms

With index:
Query time = 52 ms



Index Scan using task3 — means PostgreSQL uses index Index Cond: — shows the conditions under which the index operates Execution Time: — total execution time



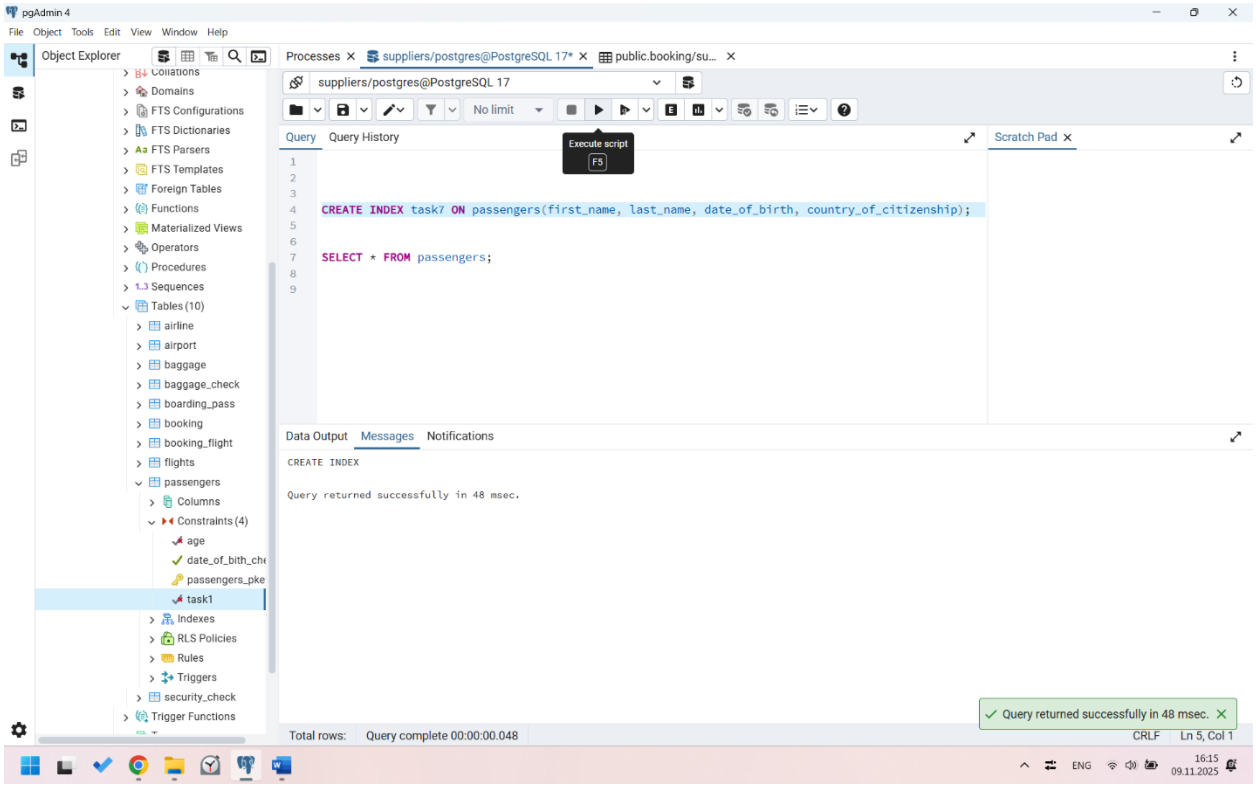


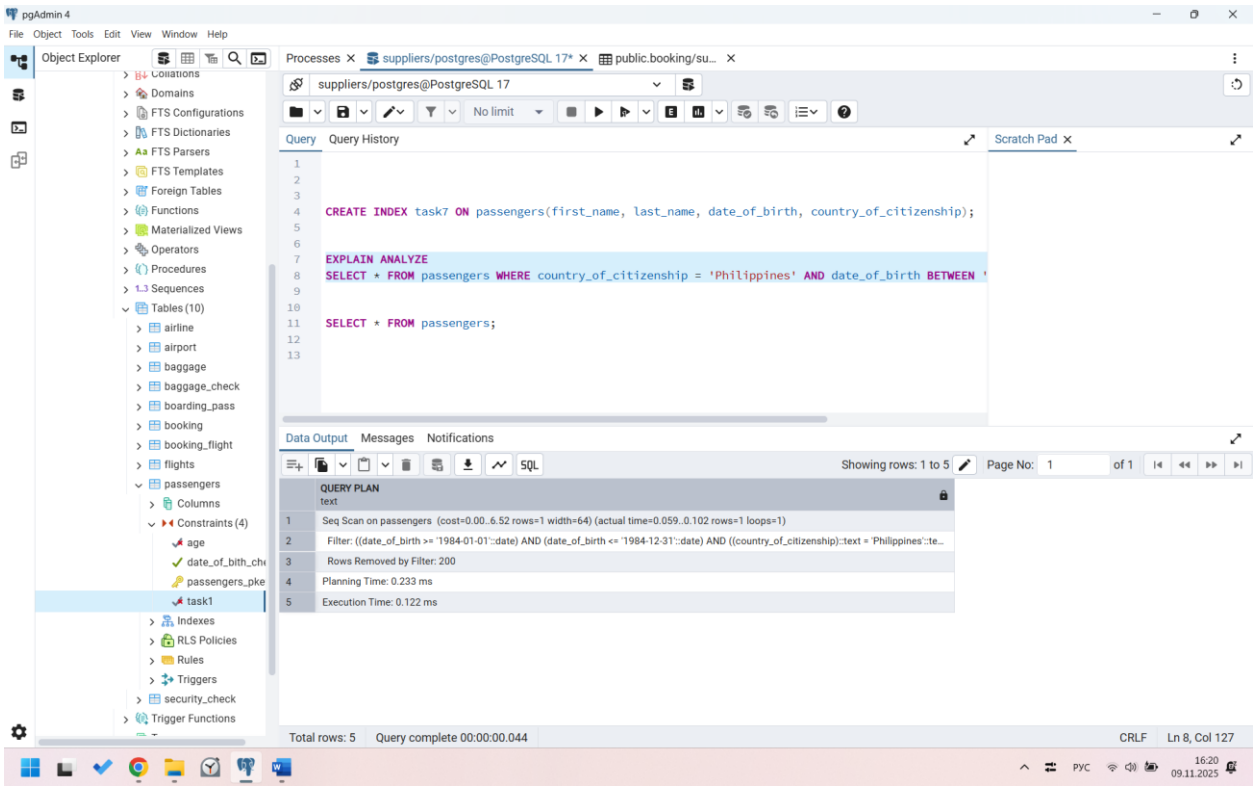
When I created the **unique index** on passport_number, PostgreSQL started enforcing the rule that every value in that column must be unique.

The first passenger inserted successfully because the passport number was new.

The second insert failed with an error, because the same passport_number already existed in the table.

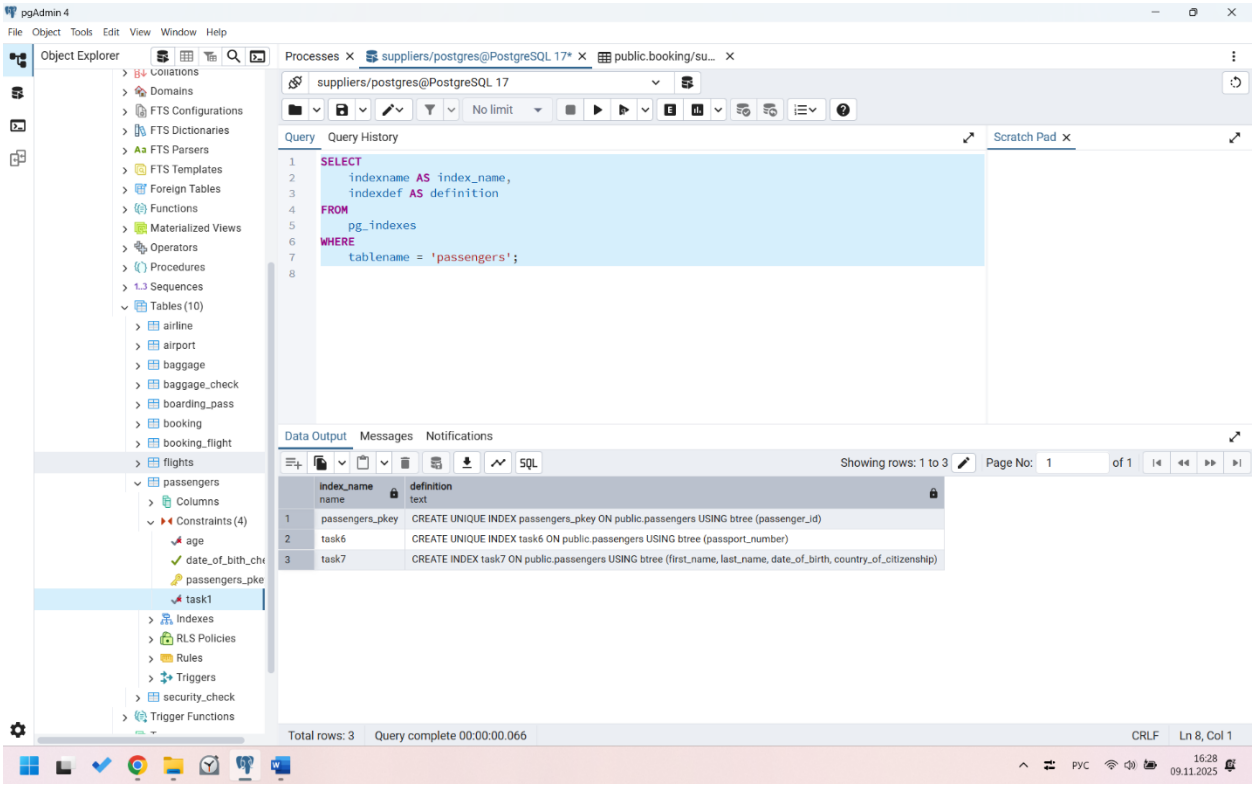
This confirms that the **unique index works correctly** — it prevents duplicate passport numbers from being added.

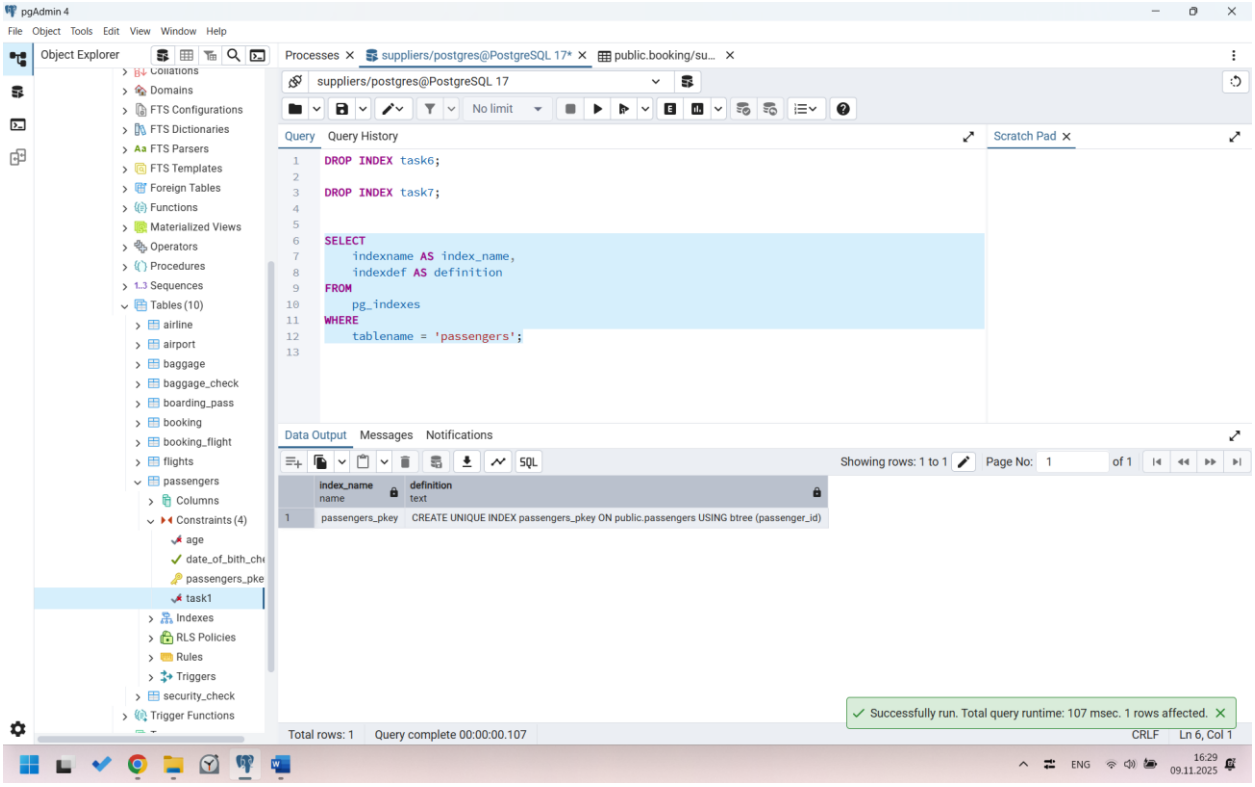




Although I created a composite index on (first_name, last_name, date_of_birth, country_of_citizenship), PostgreSQL did not use it because my query filters only on the last two columns.
For composite indexes, PostgreSQL can only use them effectively if the filtering starts with the leftmost

columns.


pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Processes
suppliers/postgres@PostgreSQL 17*
public.booking/su...
suppliers/postgres@PostgreSQL 17
No limit
Query
Query History
Scratch Pad
SELECT
indexname AS index_name,
indexdef AS definition
FROM
pg_indexes
WHERE
tablename = 'passengers';
Data Output
Messages
Notifications
Showing rows: 1 to 3
Page No: 1
of 1
index_name
name
definition
text
1 passengers_pkey CREATE UNIQUE INDEX passengers_pkey ON public.passengers USING btree (passenger_id)
2 task6 CREATE UNIQUE INDEX task6 ON public.passengers USING btree (passport_number)
3 task7 CREATE INDEX task7 ON public.passengers USING btree (first_name, last_name, date_of_birth, country_of_citizenship)
Total rows: 3
Query complete 00:00:00.066
CRLF
Ln 8, Col 1
16:28
09.11.2025


pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Processes
suppliers/postgres@PostgreSQL 17*
public.booking/su...
suppliers/postgres@PostgreSQL 17
No limit
Query
Query History
Scratch Pad
DROP INDEX task6;
DROP INDEX task7;
SELECT
indexname AS index_name,
indexdef AS definition
FROM
pg_indexes
WHERE
tablename = 'passengers';
Data Output
Messages
Notifications
Showing rows: 1 to 1
Page No: 1
of 1
index_name
name
definition
text
1 passengers_pkey CREATE UNIQUE INDEX passengers_pkey ON public.passengers USING btree (passenger_id)
Successfully run. Total query runtime: 107 msec. 1 rows affected.
Total rows: 1
Query complete 00:00:00.107
CRLF
Ln 6, Col 1
16:29
09.11.2025