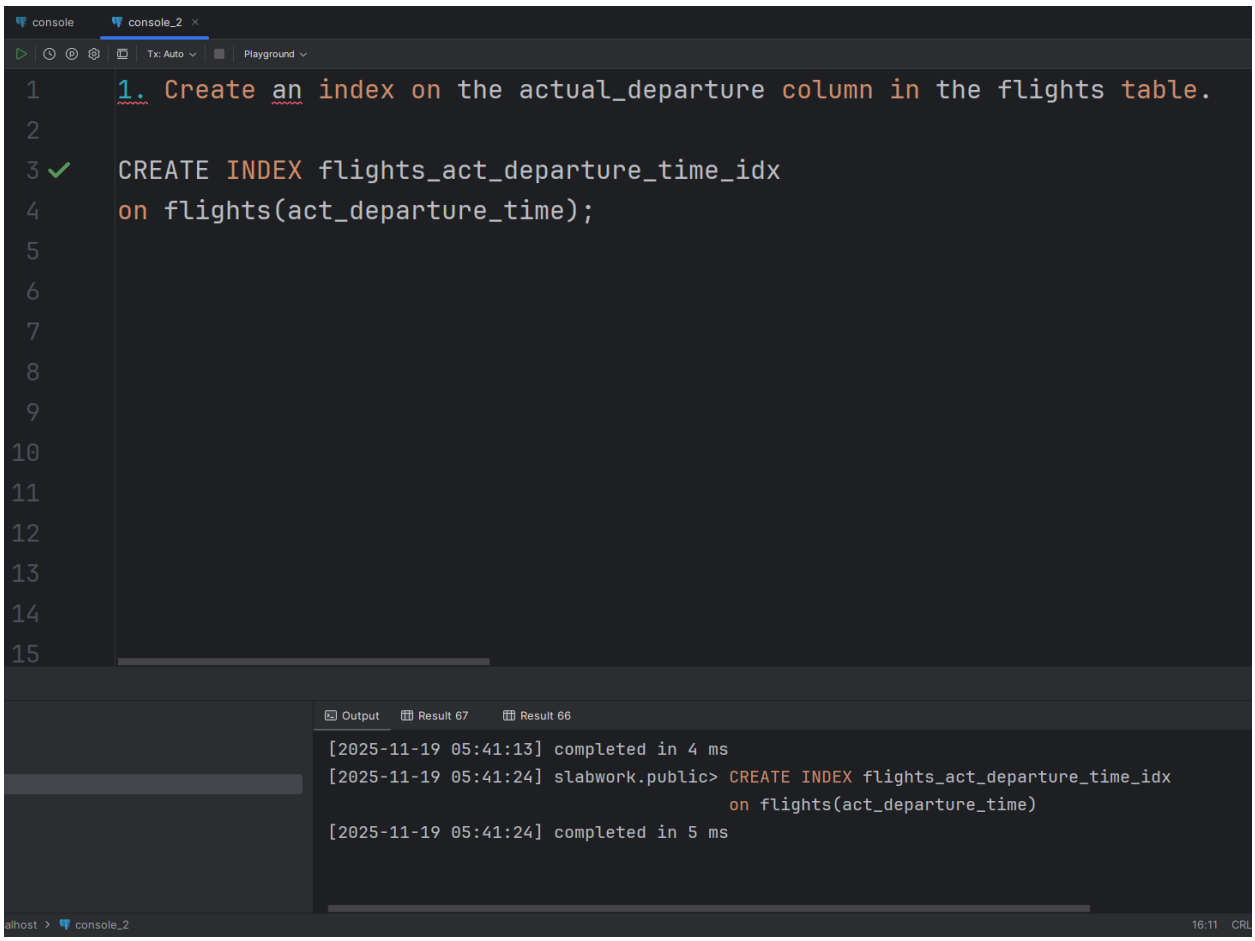
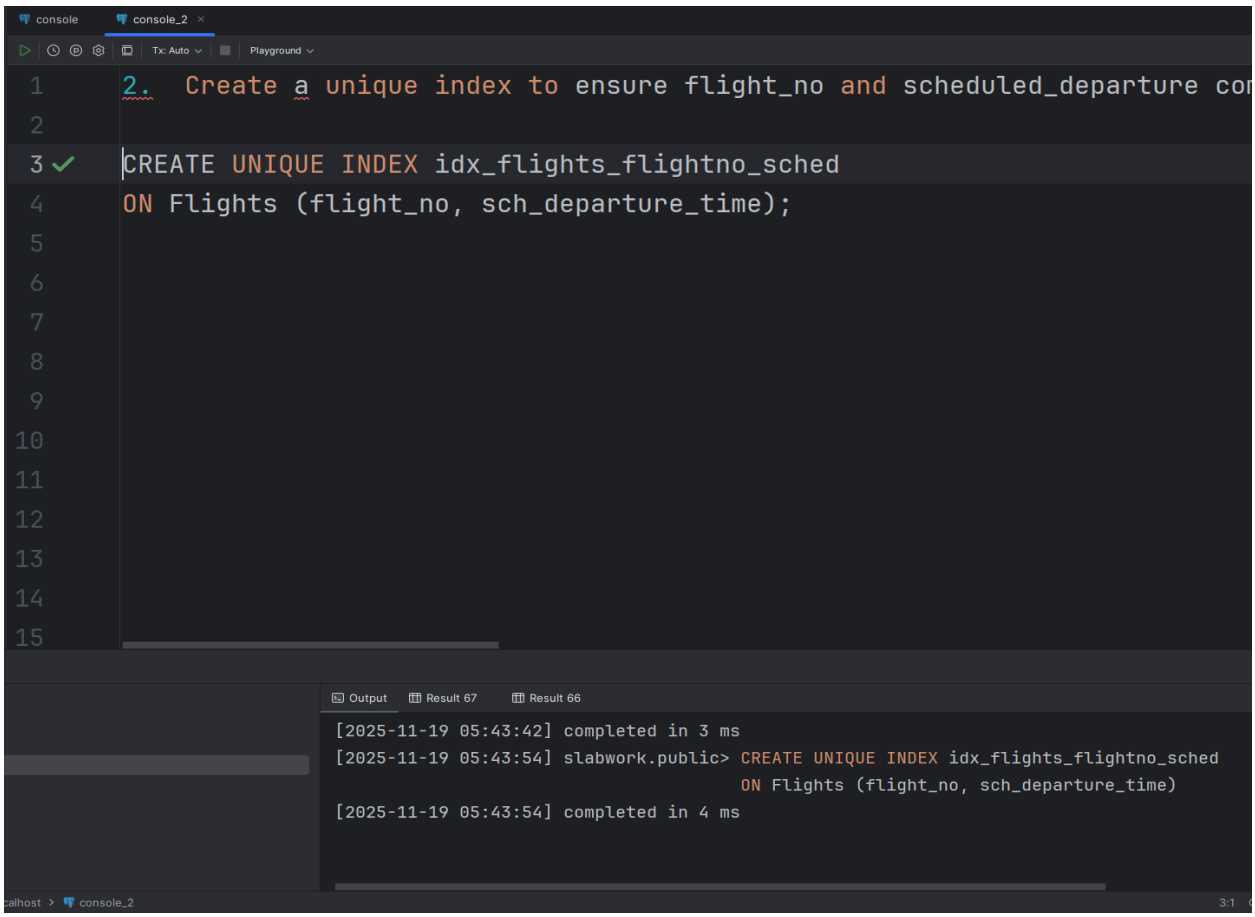
1)


console
console_2
Tx: Auto
Playground
1. Create an index on the actual_departure column in the flights table.
CREATE INDEX flights_act_departure_time_idx
on flights(act_departure_time);
Output
Result 67
Result 66
[2025-11-19 05:41:13] completed in 4 ms
[2025-11-19 05:41:24] slabwork.public> CREATE INDEX flights_act_departure_time_idx
on flights(act_departure_time)
[2025-11-19 05:41:24] completed in 5 ms
console_2

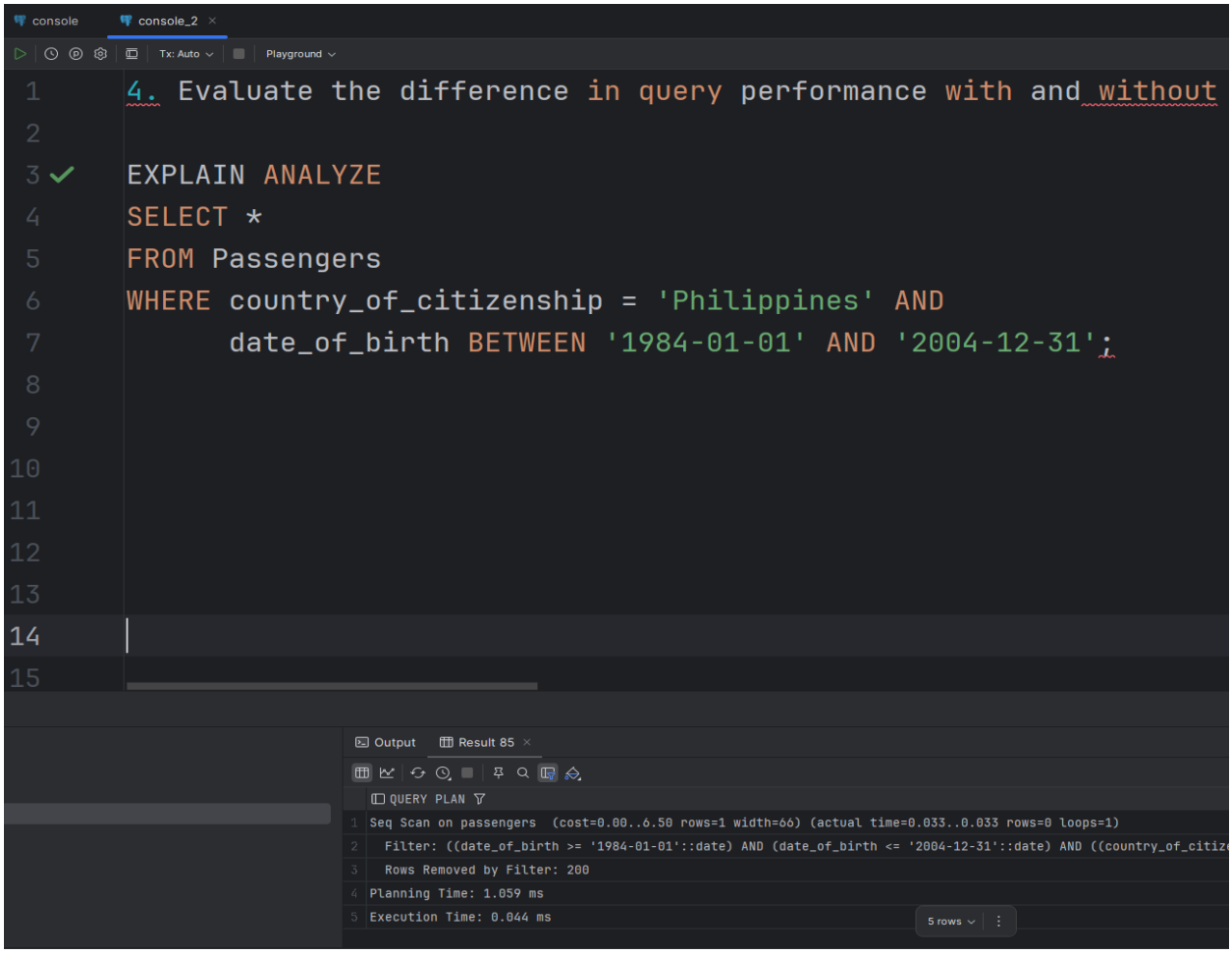
2)

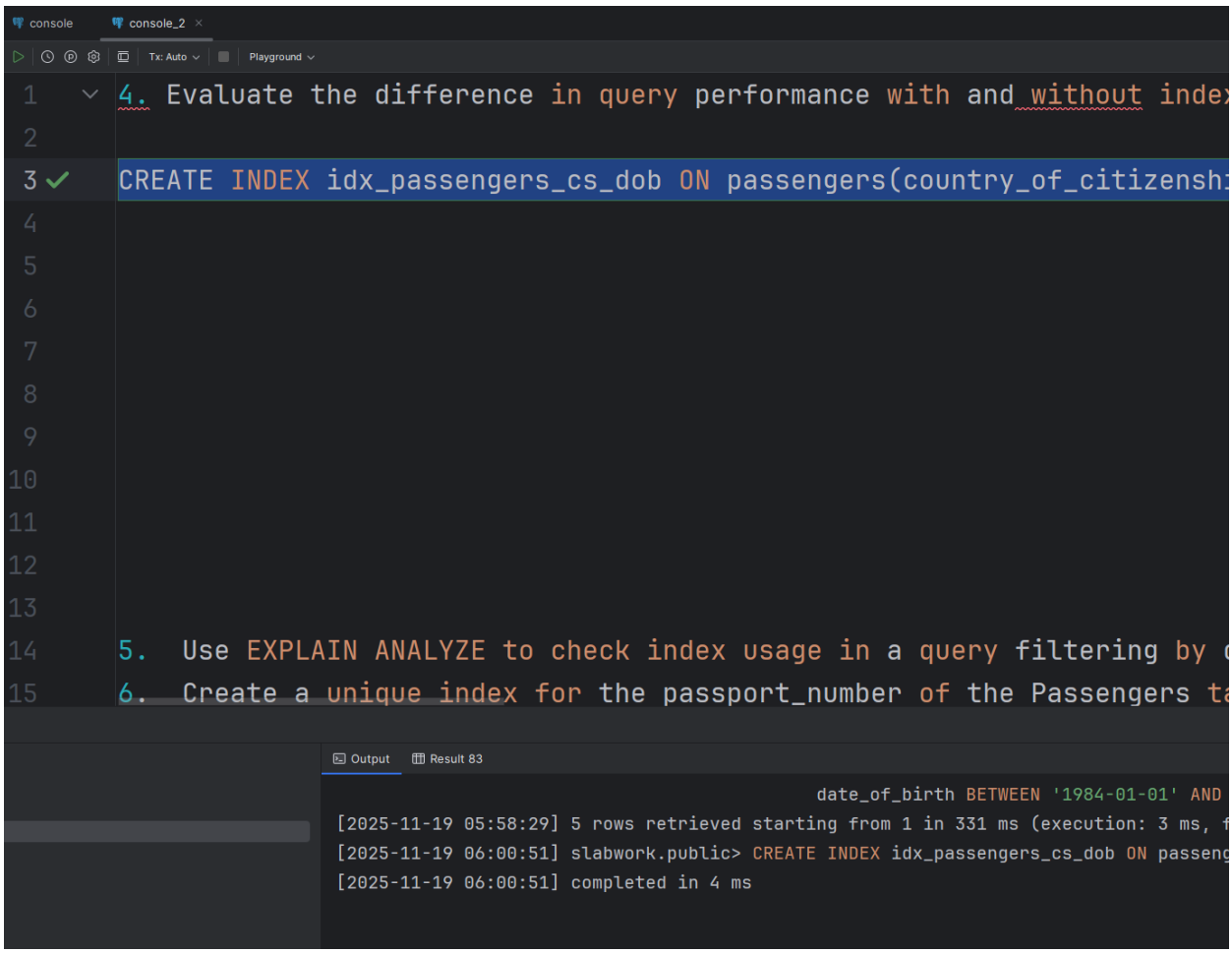

console
console_2
Tx: Auto
Playground
2. Create a unique index to ensure flight_no and scheduled_departure co
CREATE UNIQUE INDEX idx_flights_flightno_sched
ON Flights (flight_no, sch_departure_time);
Output
Result 67
Result 66
[2025-11-19 05:43:42] completed in 3 ms
[2025-11-19 05:43:54] slabwork.public> CREATE UNIQUE INDEX idx_flights_flightno_sched
ON Flights (flight_no, sch_departure_time)
[2025-11-19 05:43:54] completed in 4 ms
console_2

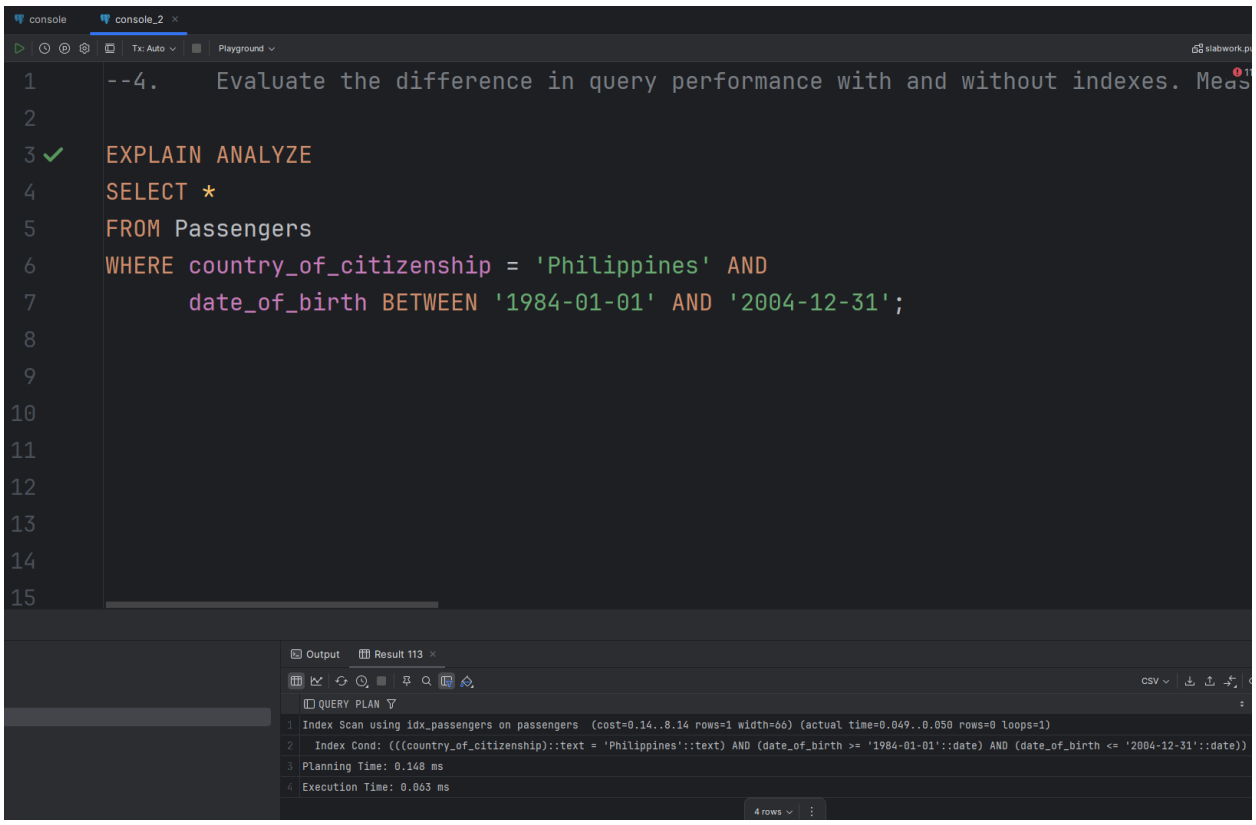
3)


console
console_2
Tx: Auto
Playground
3. Create a composite index on the departure_airport_id and arrival_airport_i
CREATE INDEX idx_flights_departure_arrival
ON Flights (departing_airport_id, arriving_airport_id);
ON Flights (flight_no, sch_departure_time)
[2025-11-19 05:43:54] completed in 4 ms
[2025-11-19 05:48:17] slabwork.public> CREATE INDEX idx_flights_departure_arrival
ON Flights (departing_airport_id, arriving_airport_id)
[2025-11-19 05:48:17] completed in 5 ms
6:1 CRLF UTF-8

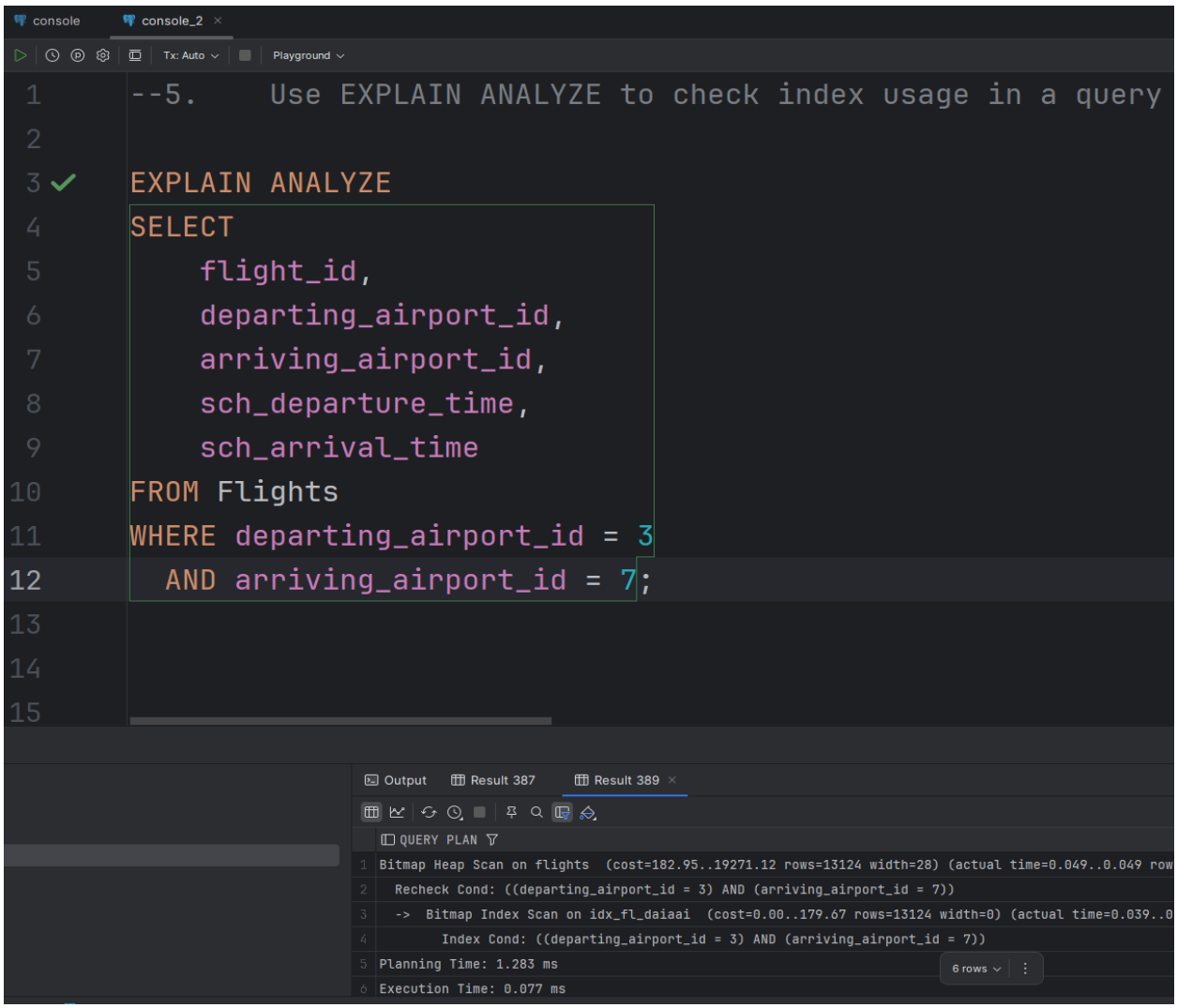
4)

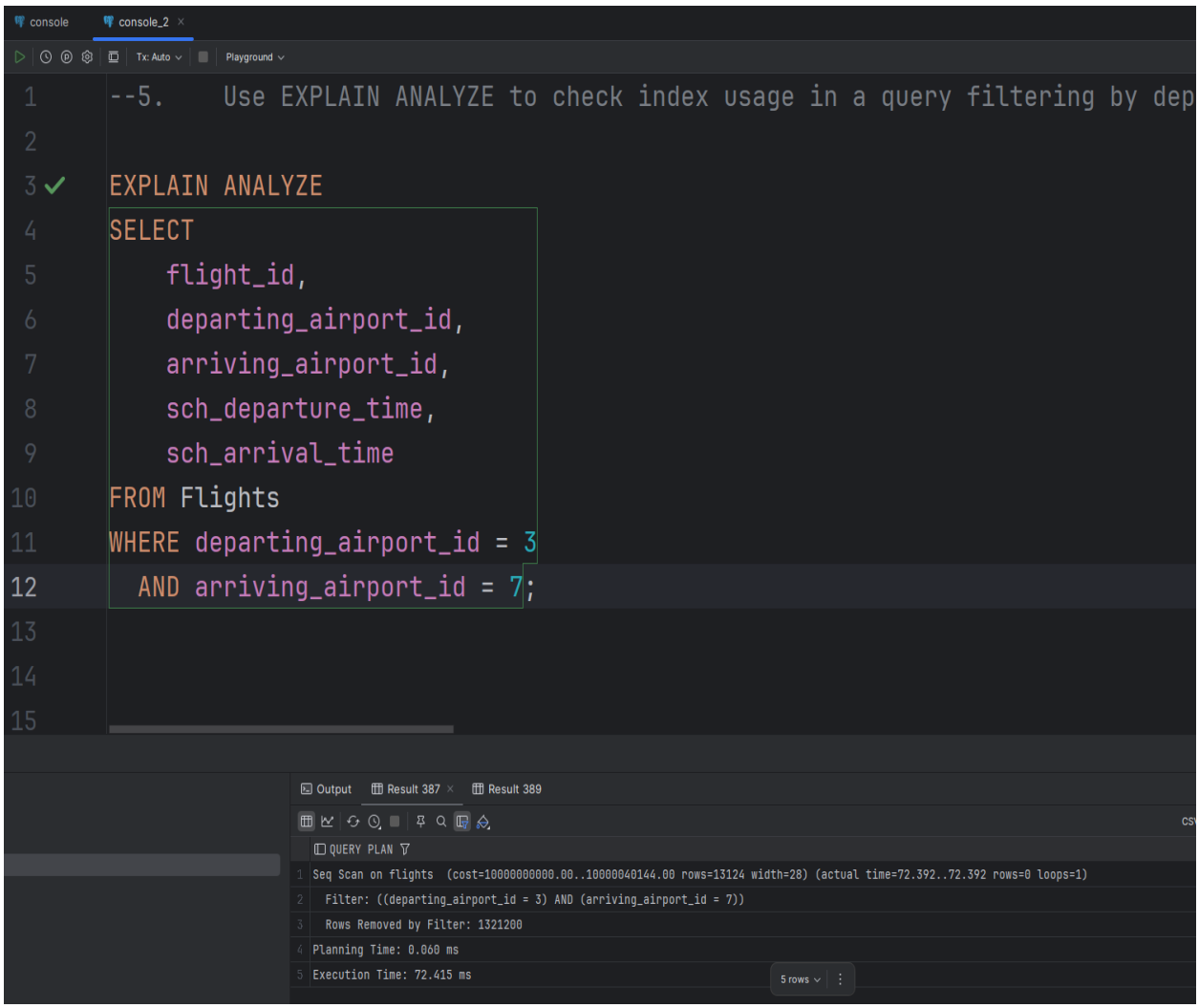

console
console_2
Tx: Auto
Playground
4. Evaluate the difference in query performance with and without
EXPLAIN ANALYZE
SELECT *
FROM Passengers
WHERE country_of_citizenship = 'Philippines' AND
date_of_birth BETWEEN '1984-01-01' AND '2004-12-31';
Output
Result 85
QUERY PLAN
Seq Scan on passengers (cost=0.00..6.50 rows=1 width=66) (actual time=0.033..0.033 rows=0 loops=1)
Filter: ((date_of_birth >= '1984-01-01'::date) AND (date_of_birth <= '2004-12-31'::date) AND ((country_of_citiz
Rows Removed by Filter: 200
Planning Time: 1.059 ms
Execution Time: 0.044 ms
5 rows

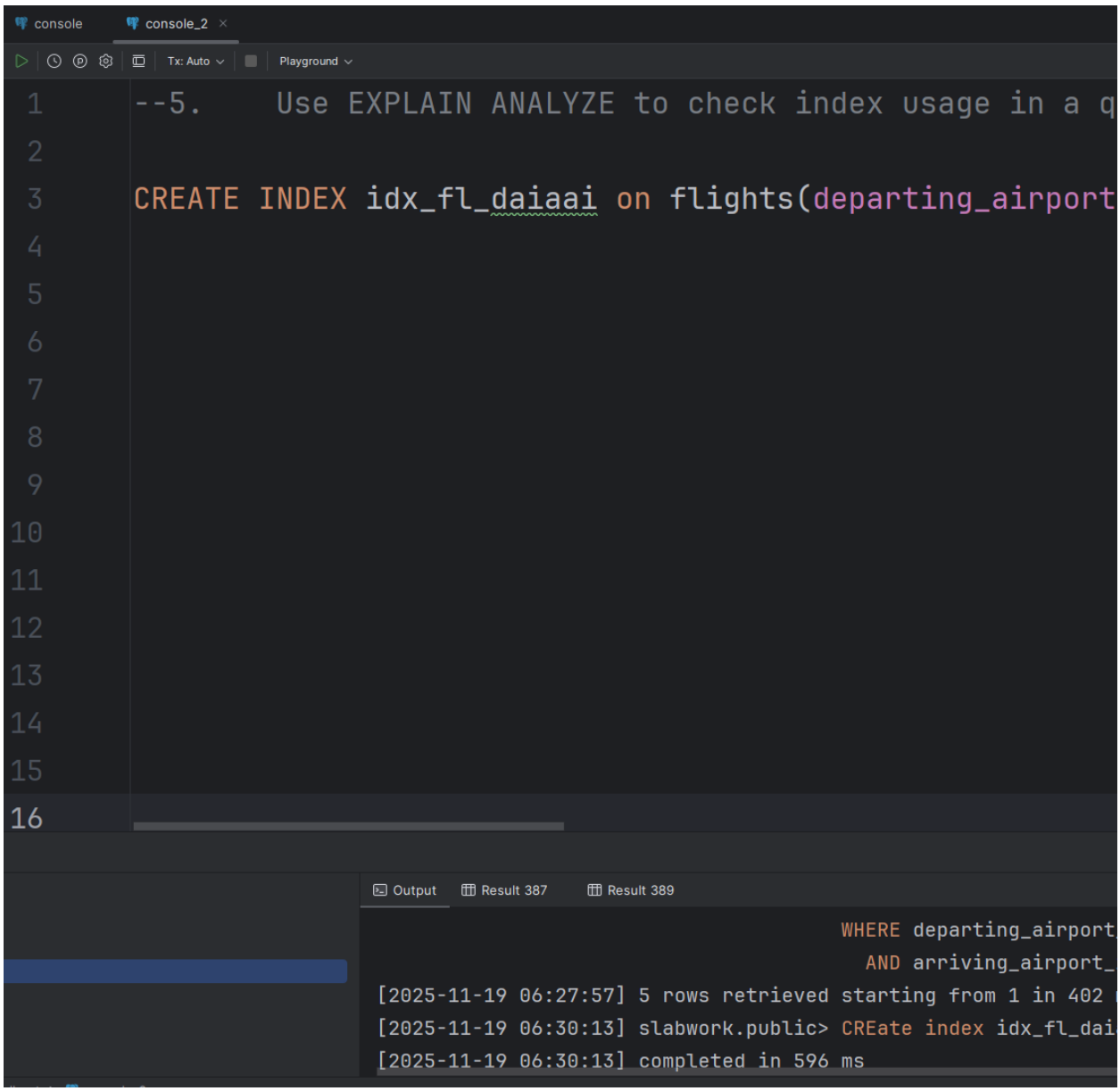

console
console_2
Tx: Auto
Playground
4. Evaluate the difference in query performance with and without index
CREATE INDEX idx_passengers_cs_dob ON passengers(country_of_citizenshi
5. Use EXPLAIN ANALYZE to check index usage in a query filtering by
6. Create a unique index for the passport_number of the Passengers ta
Output
Result 83
date_of_birth BETWEEN '1984-01-01' AND
[2025-11-19 05:58:29] 5 rows retrieved starting from 1 in 331 ms (execution: 3 ms, f
[2025-11-19 06:00:51] slabwork.public> CREATE INDEX idx_passengers_cs_dob ON passeng
[2025-11-19 06:00:51] completed in 4 ms


console
console_2
Tx: Auto
Playground
--4. Evaluate the difference in query performance with and without indexes. Meas
EXPLAIN ANALYZE
SELECT *
FROM Passengers
WHERE country_of_citizenship = 'Philippines' AND
date_of_birth BETWEEN '1984-01-01' AND '2004-12-31';
Output
Result 113
CSV
QUERY PLAN
Index Scan using idx_passengers on passengers (cost=0.14..8.14 rows=1 width=66) (actual time=0.049..0.050 rows=0 loops=1)
Index Cond: (((country_of_citizenship)::text = 'Philippines'::text) AND (date_of_birth >= '1984-01-01'::date) AND (date_of_birth <= '2004-12-31'::date))
Planning Time: 0.148 ms
Execution Time: 0.063 ms
4 rows

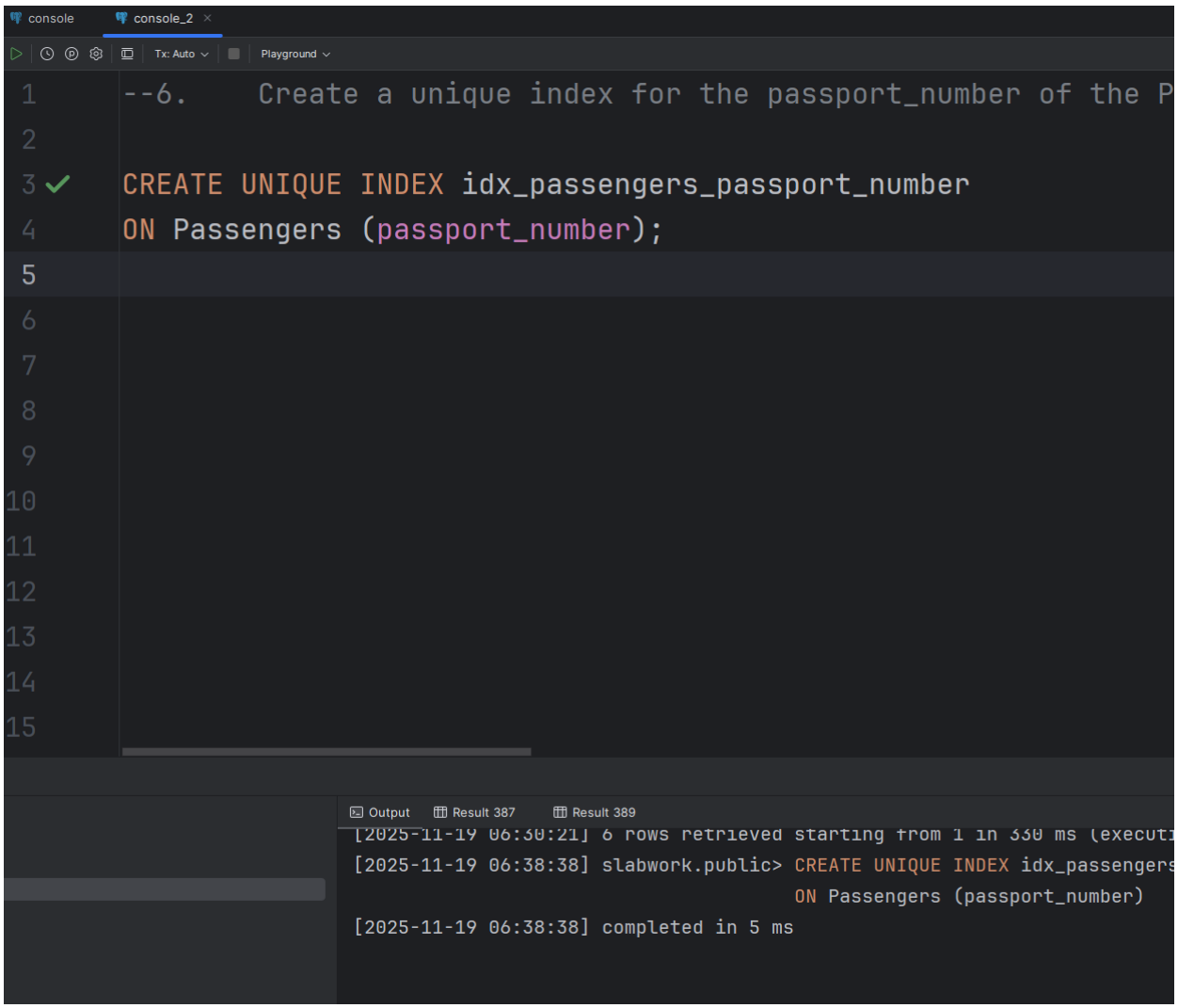
5)

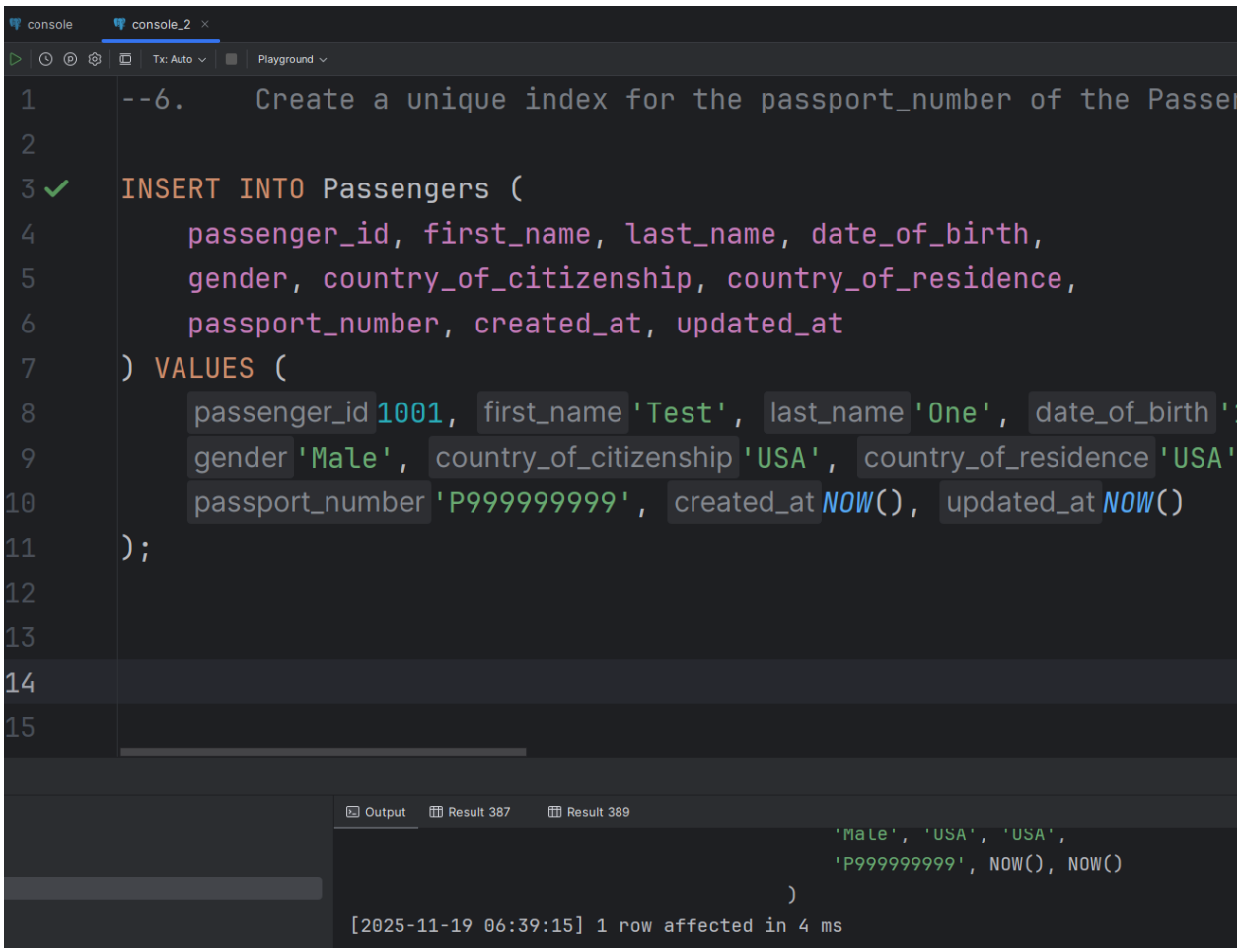

console
console_2
Tx: Auto
Playground
--5. Use EXPLAIN ANALYZE to check index usage in a query
EXPLAIN ANALYZE
SELECT
flight_id,
departing_airport_id,
arriving_airport_id,
sch_departure_time,
sch_arrival_time
FROM Flights
WHERE departing_airport_id = 3
AND arriving_airport_id = 7;
Output
Result 387
Result 389
QUERY PLAN
Bitmap Heap Scan on flights (cost=182.95..19271.12 rows=13124 width=28) (actual time=0.049..0.049 row
Recheck Cond: ((departing_airport_id = 3) AND (arriving_airport_id = 7))
-> Bitmap Index Scan on idx_fl_daiaai (cost=0.00..179.67 rows=13124 width=0) (actual time=0.039..0
Index Cond: ((departing_airport_id = 3) AND (arriving_airport_id = 7))
Planning Time: 1.283 ms
Execution Time: 0.077 ms
6 rows

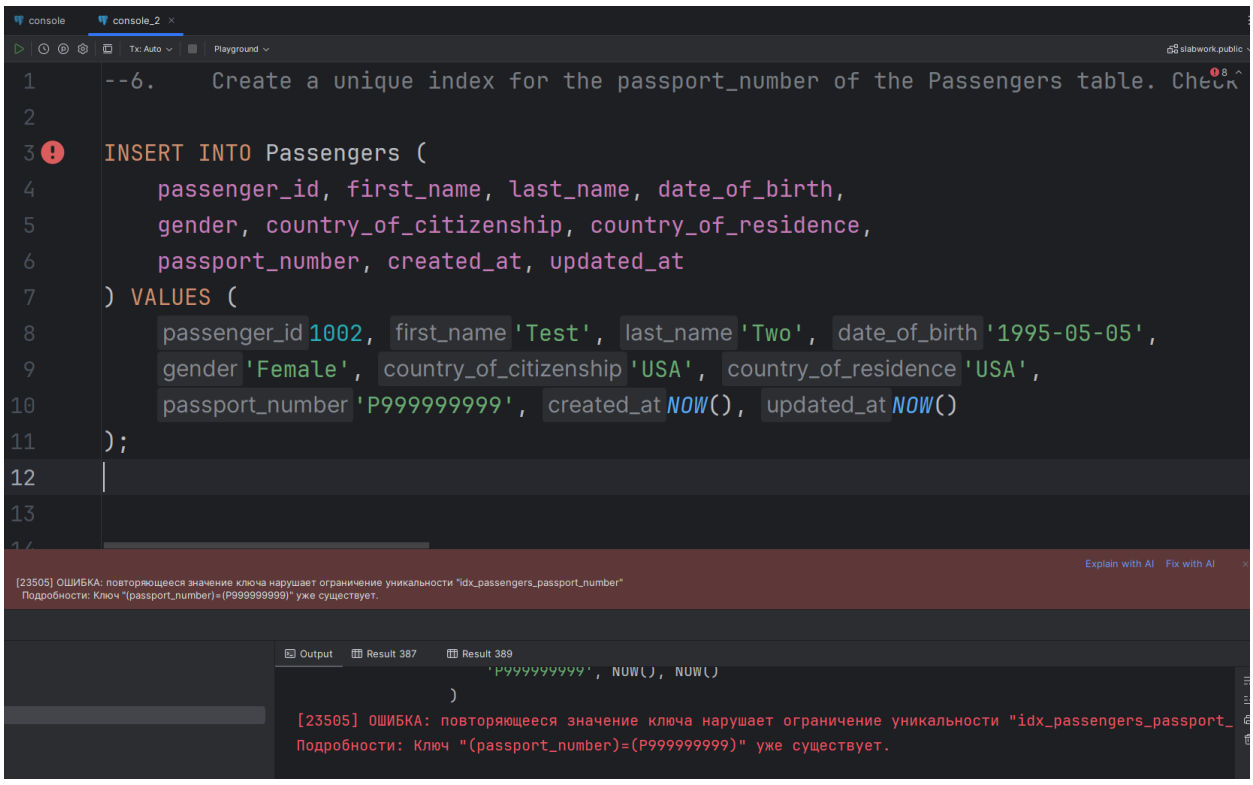

console
console_2
Tx: Auto
Playground
--5. Use EXPLAIN ANALYZE to check index usage in a query filtering by dep
EXPLAIN ANALYZE
SELECT
flight_id,
departing_airport_id,
arriving_airport_id,
sch_departure_time,
sch_arrival_time
FROM Flights
WHERE departing_airport_id = 3
AND arriving_airport_id = 7;
Output
Result 387
Result 389
QUERY PLAN
Seq Scan on flights (cost=10000000000.00..10000040144.00 rows=13124 width=28) (actual time=72.392..72.392 rows=0 loops=1)
Filter: ((departing_airport_id = 3) AND (arriving_airport_id = 7))
Rows Removed by Filter: 1321200
Planning Time: 0.060 ms
Execution Time: 72.415 ms
5 rows


console
console_2
Tx: Auto
Playground
--5. Use EXPLAIN ANALYZE to check index usage in a q
CREATE INDEX idx_fl_daiaai on flights(departing_airport
Output
Result 387
Result 389
WHERE departing_airport
AND arriving_airport_
[2025-11-19 06:27:57] 5 rows retrieved starting from 1 in 402
[2025-11-19 06:30:13] slabwork.public> CREate index idx_fl_dai
[2025-11-19 06:30:13] completed in 596 ms

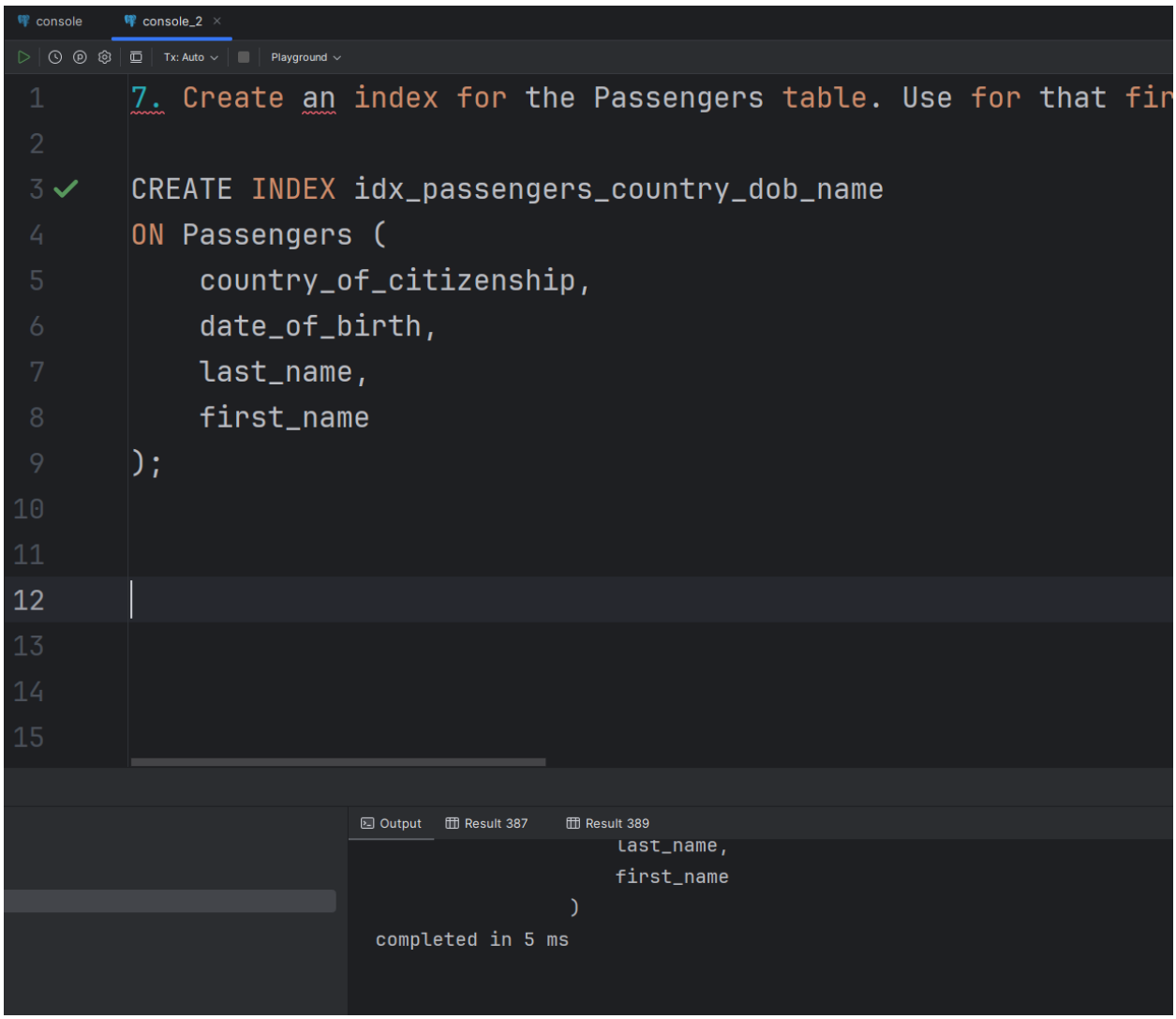
6)

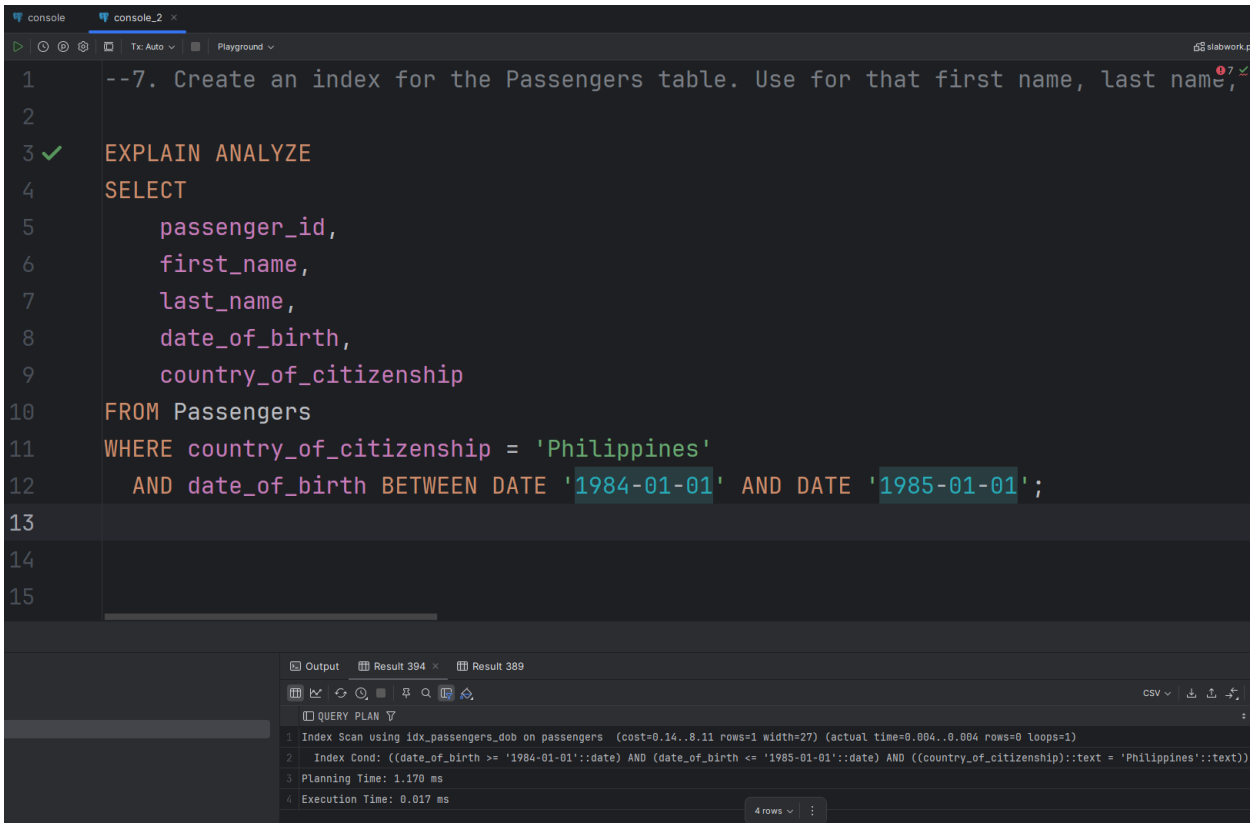

console
console_2
Tx: Auto
Playground
--6. Create a unique index for the passport_number of the P
CREATE UNIQUE INDEX idx_passengers_passport_number
ON Passengers (passport_number);
Output
Result 387
Result 389
[2025-11-19 06:38:38] slabwork.public> CREATE UNIQUE INDEX idx_passengers
ON Passengers (passport_number)
[2025-11-19 06:38:38] completed in 5 ms


console
console_2
Tx: Auto
Playground
--6. Create a unique index for the passport_number of the Passe
INSERT INTO Passengers (
passenger_id, first_name, last_name, date_of_birth,
gender, country_of_citizenship, country_of_residence,
passport_number, created_at, updated_at
) VALUES (
passenger_id 1001, first_name 'Test', last_name 'One', date_of_birth '
gender 'Male', country_of_citizenship 'USA', country_of_residence 'USA'
passport_number 'P999999999', created_at NOW(), updated_at NOW()
);
Output
Result 387
Result 389
'Male', 'USA', 'USA',
'P999999999', NOW(), NOW()
)
[2025-11-19 06:39:15] 1 row affected in 4 ms

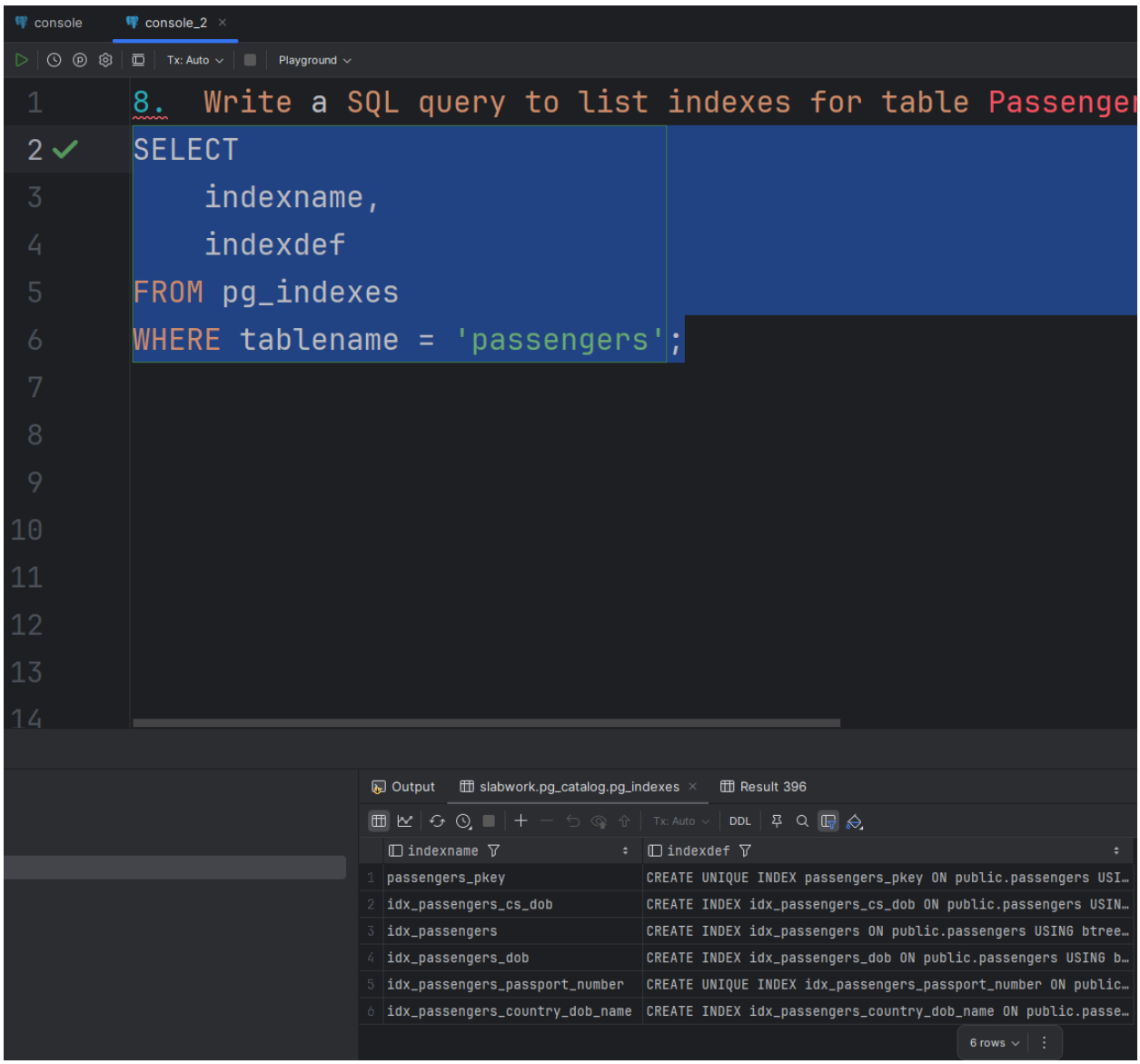

console
console_2
Tx: Auto
Playground
slabwork.public
--6. Create a unique index for the passport_number of the Passengers table. Check
INSERT INTO Passengers (
passenger_id, first_name, last_name, date_of_birth,
gender, country_of_citizenship, country_of_residence,
passport_number, created_at, updated_at
) VALUES (
passenger_id 1002, first_name 'Test', last_name 'Two', date_of_birth '1995-05-05',
gender 'Female', country_of_citizenship 'USA', country_of_residence 'USA',
passport_number 'P999999999', created_at NOW(), updated_at NOW()
);
Explain with AI
Fix with AI
[23505] ОШИБКА: повторяющееся значение ключа нарушает ограничение уникальности "idx_passengers_passport_number"
Подробности: Ключ "(passport_number)=(P999999999)" уже существует.
Output
Result 387
Result 389
'P999999999', NOW(), NOW()
)
[23505] ОШИБКА: повторяющееся значение ключа нарушает ограничение уникальности "idx_passengers_passport_
Подробности: Ключ "(passport_number)=(P999999999)" уже существует.

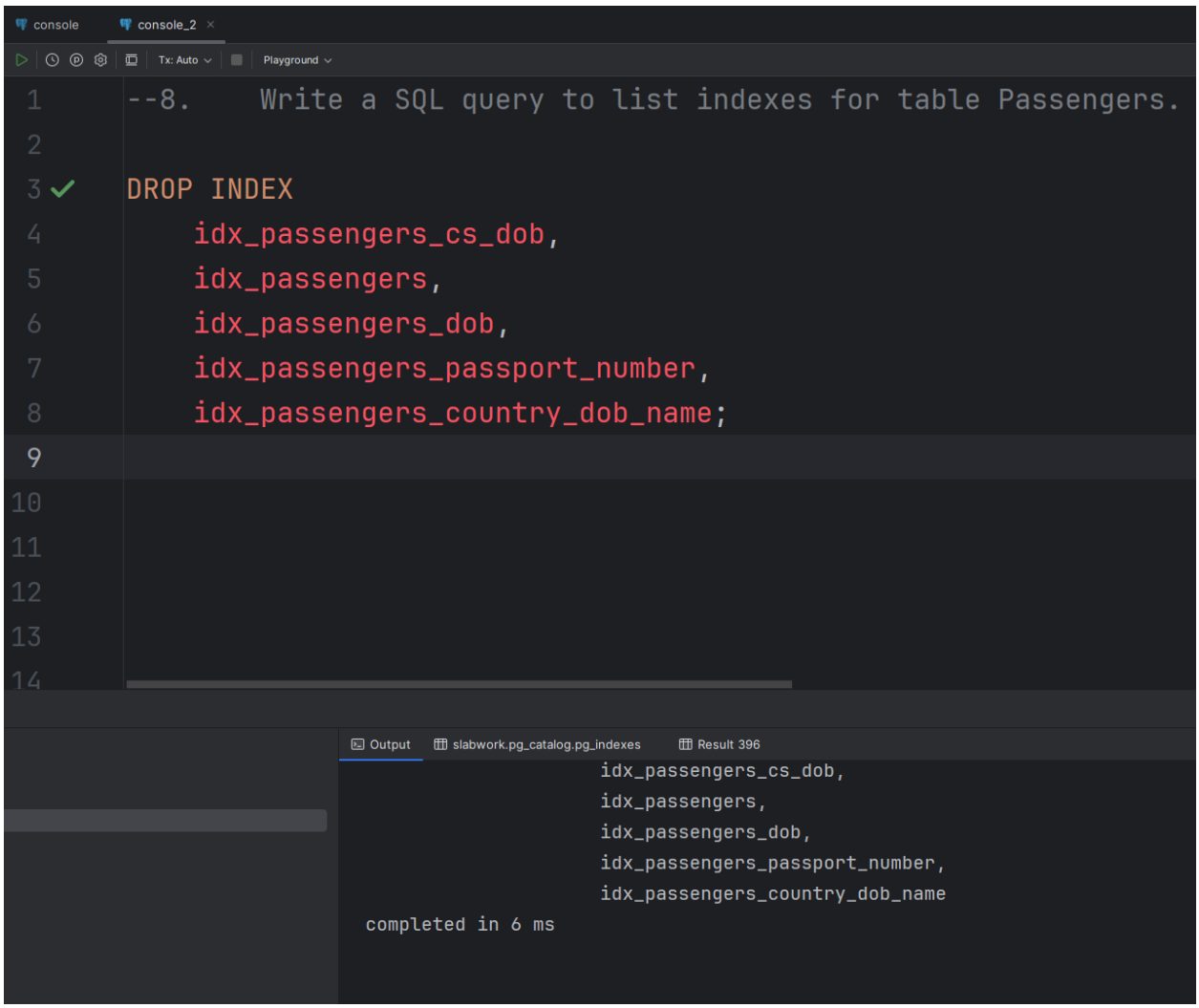
7)


console
console_2
Tx: Auto
Playground
7. Create an index for the Passengers table. Use for that fir
CREATE INDEX idx_passengers_country_dob_name
ON Passengers (
country_of_citizenship,
date_of_birth,
last_name,
first_name
);
Output
Result 387
Result 389
last_name,
first_name
)
completed in 5 ms


console
console_2
Tx: Auto
Playground
--7. Create an index for the Passengers table. Use for that first name, last name,
EXPLAIN ANALYZE
SELECT
passenger_id,
first_name,
last_name,
date_of_birth,
country_of_citizenship
FROM Passengers
WHERE country_of_citizenship = 'Philippines'
AND date_of_birth BETWEEN DATE '1984-01-01' AND DATE '1985-01-01';
Output
Result 394
Result 389
CSV
QUERY PLAN
Index Scan using idx_passengers_dob on passengers (cost=0.14..8.11 rows=1 width=27) (actual time=0.004..0.004 rows=0 loops=1)
Index Cond: ((date_of_birth >= '1984-01-01'::date) AND (date_of_birth <= '1985-01-01'::date) AND ((country_of_citizenship)::text = 'Philippines'::text))
Planning Time: 1.170 ms
Execution Time: 0.017 ms
4 rows

8)


console
console_2
Tx: Auto
Playground
8. Write a SQL query to list indexes for table Passenger
SELECT
indexname,
indexdef
FROM pg_indexes
WHERE tablename = 'passengers';
Output
slabwork.pg_catalog.pg_indexes
Result 396
Tx: Auto
DDL
indexname
indexdef
1 passengers_pkey CREATE UNIQUE INDEX passengers_pkey ON public.passengers USI…
2 idx_passengers_cs_dob CREATE INDEX idx_passengers_cs_dob ON public.passengers USIN…
3 idx_passengers CREATE INDEX idx_passengers ON public.passengers USING btree…
4 idx_passengers_dob CREATE INDEX idx_passengers_dob ON public.passengers USING b…
5 idx_passengers_passport_number CREATE UNIQUE INDEX idx_passengers_passport_number ON public…
6 idx_passengers_country_dob_name CREATE INDEX idx_passengers_country_dob_name ON public.passe…
6 rows


console
console_2
Tx: Auto
Playground
--8. Write a SQL query to list indexes for table Passengers.
DROP INDEX
idx_passengers_cs_dob,
idx_passengers,
idx_passengers_dob,
idx_passengers_passport_number,
idx_passengers_country_dob_name;
Output
slabwork.pg_catalog.pg_indexes
Result 396
idx_passengers_cs_dob,
idx_passengers,
idx_passengers_dob,
idx_passengers_passport_number,
idx_passengers_country_dob_name
completed in 6 ms