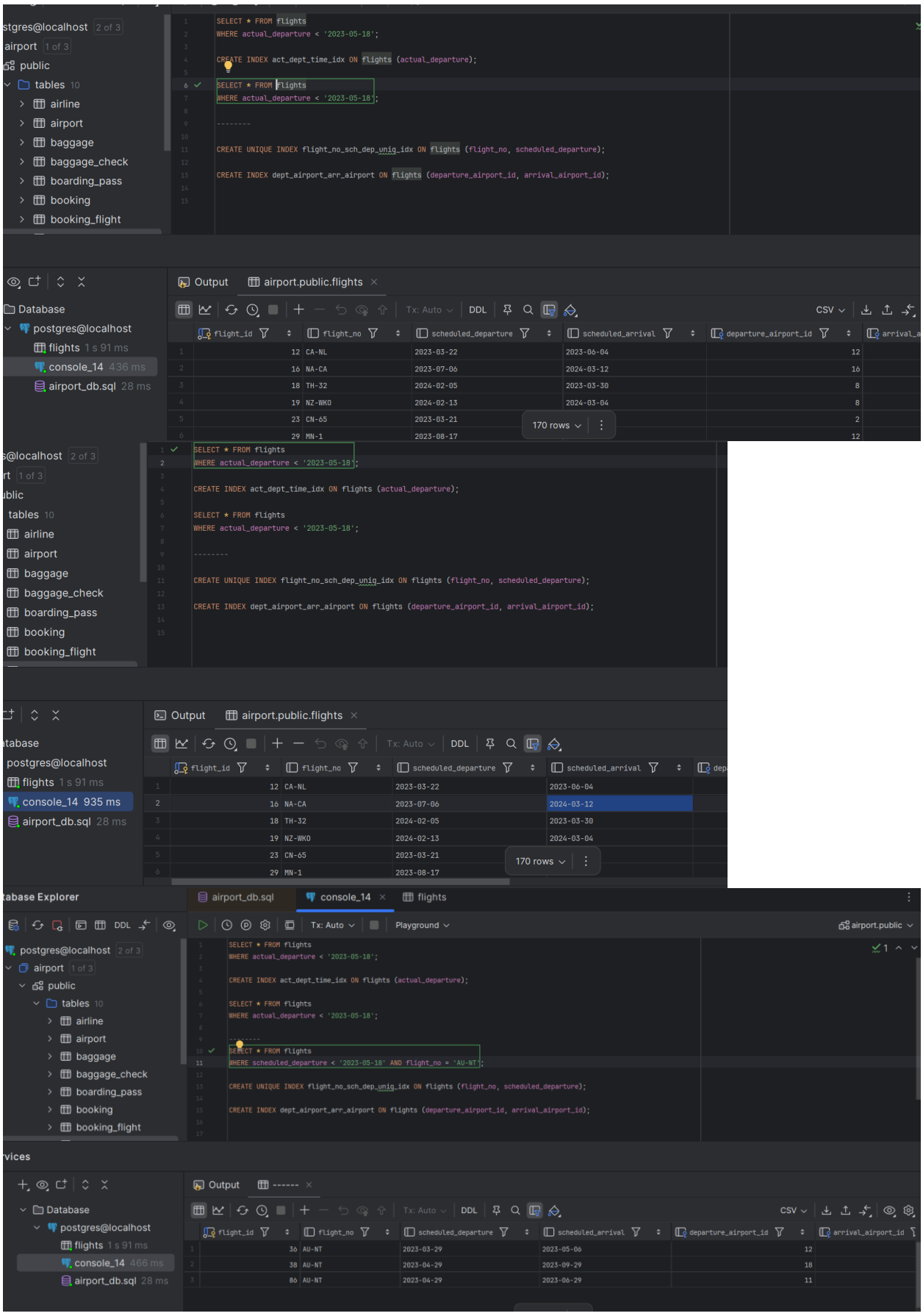
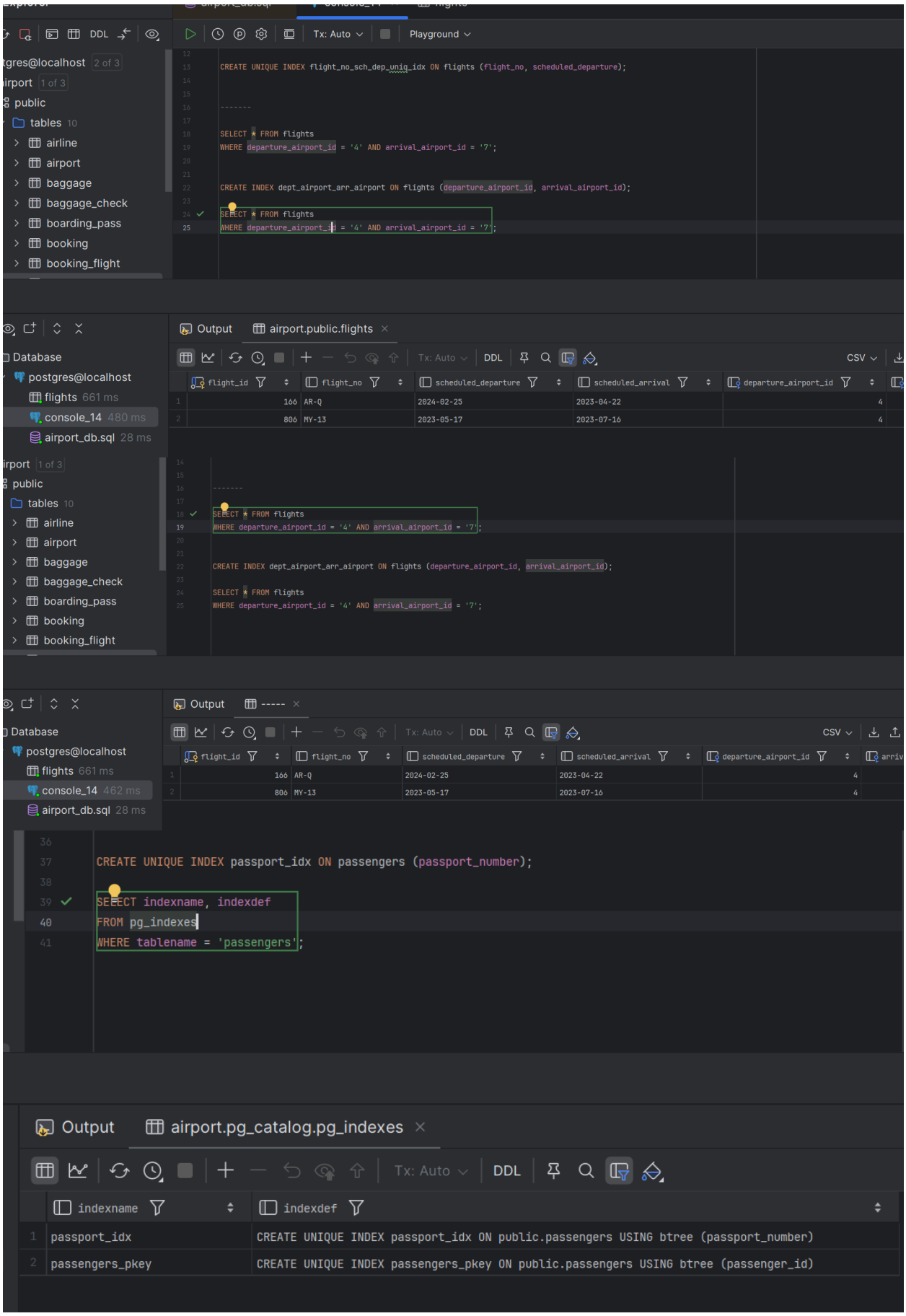

SELECT * FROM flights
WHERE actual_departure < '2023-05-18';
CREATE INDEX act_dept_time_idx ON flights (actual_departure);
SELECT * FROM flights
WHERE actual_departure < '2023-05-18';

CREATE UNIQUE INDEX flight_no_sch_dep_unig_idx ON flights (flight_no, scheduled_departure);
CREATE INDEX dept_airport_arr_airport ON flights (departure_airport_id, arrival_airport_id);
postgres@localhost 2 of 3
airport 1 of 3
public
tables 10
airline
airport
baggage
baggage_check
boarding_pass
booking
booking_flight
Output
airport.public.flights
Database
postgres@localhost
flights 1 s 91 ms
console_14 436 ms
airport_db.sql 28 ms
flight_id
flight_no
scheduled_departure
scheduled_arrival
departure_airport_id
12 CA-NL 2023-03-22 2023-06-04 12
16 NA-CA 2023-07-06 2024-03-12 16
18 TH-32 2024-02-05 2023-03-30 8
19 NZ-WKO 2024-02-13 2024-03-04 8
23 CN-65 2023-03-21 2
29 MN-1 2023-08-17 12
170 rows
console_14 935 ms
Database Explorer
airport_db.sql
console_14
flights
Playground
SELECT * FROM flights
WHERE scheduled_departure < '2023-05-18' AND flight_no = 'AU-NT';
Services
Output
36 AU-NT 2023-03-29 2023-05-06 12
38 AU-NT 2023-04-29 2023-09-29 18
86 AU-NT 2023-04-29 2023-06-29 11



Unique Binary tree index table doesn’t allow inserting duplicate values in it

```
CREATE UNIQUE INDEX passport_idx ON passengers (passport_number);

SELECT indexname, indexdef
FROM pg_indexes
WHERE tablename = 'passengers';

insert into passengers (passenger_id, first_name, last_name, date_of_birth, gender, country_of_citizenship, country_of_residence, passport_number, created_at, update_at) values
insert into passengers (passenger_id, first_name, last_name, date_of_birth, gender, country_of_citizenship, country_of_residence, passport_number, created_at, update_at) values
```

[23505] ОШИБКА: повторяющееся значение ключа нарушает ограничение уникальности "passengers_pkey"
Подробности: Ключ "(passenger_id)=(202)" уже существует.

the select query filter doesn't include firstname and lastname columns, though they affect the order in passenger_idx table. So doing binary search in this index table would bring wrong results i.e can skip some actual rows . So query planner chooses to do ordinary sequence scan instead of index scan

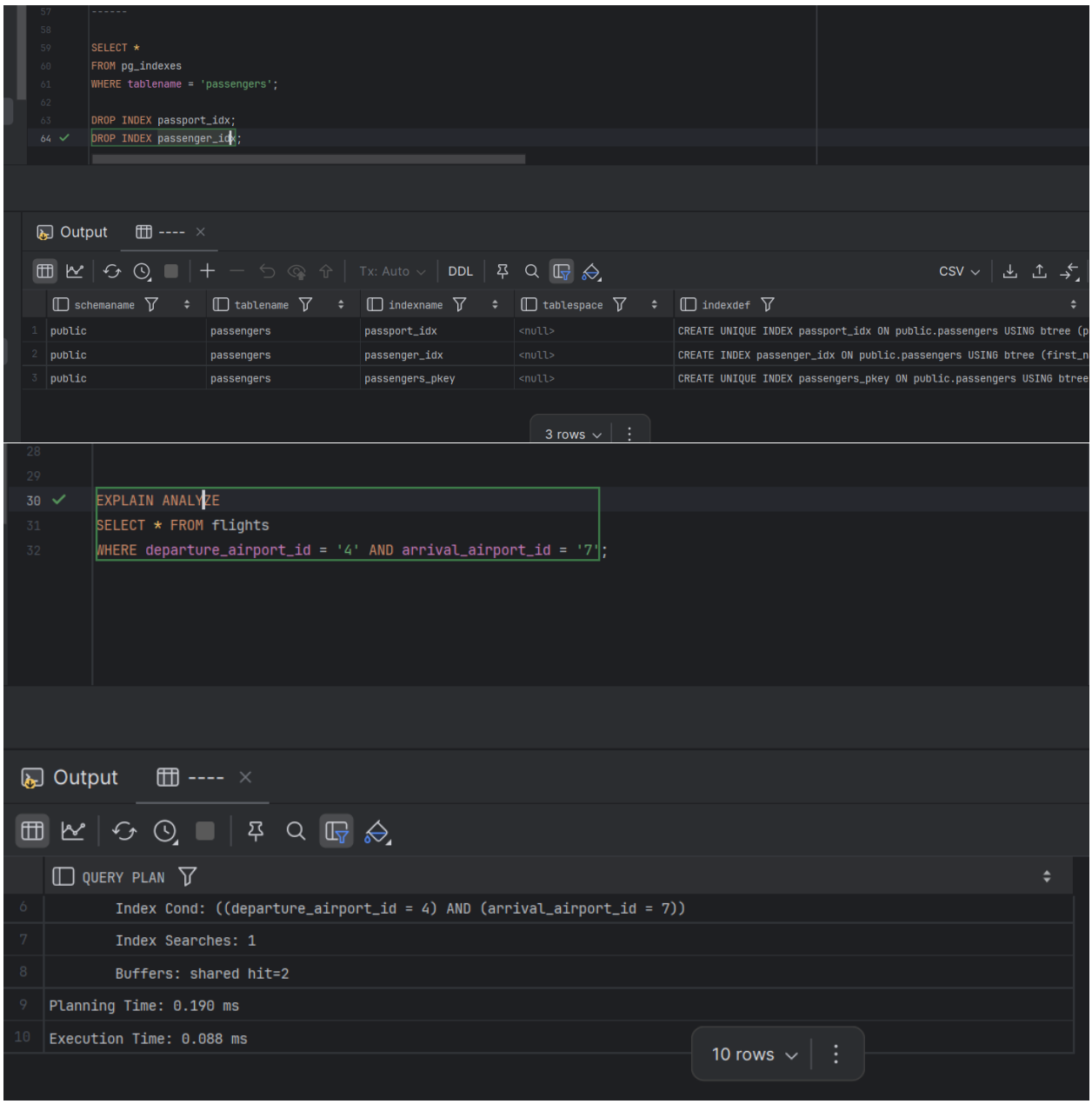
```
CREATE INDEX passenger_idx ON passengers(first_name, last_name, date_of_birth, country_of_citizenship);

EXPLAIN ANALYZE
SELECT * FROM passengers
WHERE country_of_citizenship = 'Philippines' AND EXTRACT(year from date_of_birth) = 1984;
```

Output | Result 26

QUERY PLAN
Seq Scan on passengers (cost=0.00..6.54 rows=1 width=64) (actual time=0.027..0.085 rows=1.00 loops=1)
Filter: (((country_of_citizenship)::text = 'Philippines'::text) AND (EXTRACT(year FROM date_of_birth) = '1984'::nun
Rows Removed by Filter: 201
Buffers: shared hit=3
Planning Time: 0.135 ms
Execution Time: 0.102 ms

6 rows


SELECT *
FROM pg_indexes
WHERE tablename = 'passengers';
DROP INDEX passport_idx;
DROP INDEX passenger_idx;
Output
Tx: Auto
DDL
CSV
schemaname
tablename
indexname
tablespace
indexdef
1 public passengers passport_idx <null> CREATE UNIQUE INDEX passport_idx ON public.passengers USING btree (p
2 public passengers passenger_idx <null> CREATE INDEX passenger_idx ON public.passengers USING btree (first_n
3 public passengers passengers_pkey <null> CREATE UNIQUE INDEX passengers_pkey ON public.passengers USING btree
3 rows
EXPLAIN ANALYZE
SELECT * FROM flights
WHERE departure_airport_id = '4' AND arrival_airport_id = '7';
Output
QUERY PLAN
Index Cond: ((departure_airport_id = 4) AND (arrival_airport_id = 7))
Index Searches: 1
Buffers: shared hit=2
Planning Time: 0.190 ms
Execution Time: 0.088 ms
10 rows