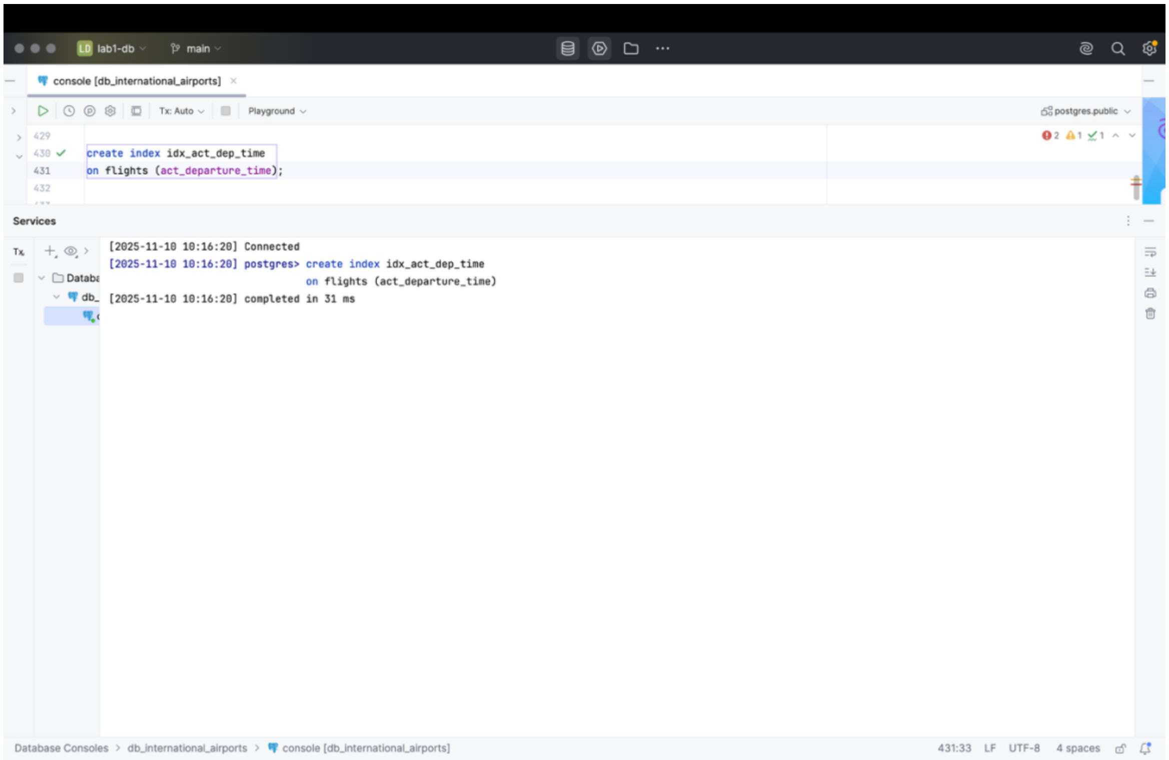
Lab 7

1. Create an index on the actual_departure column in flights table.

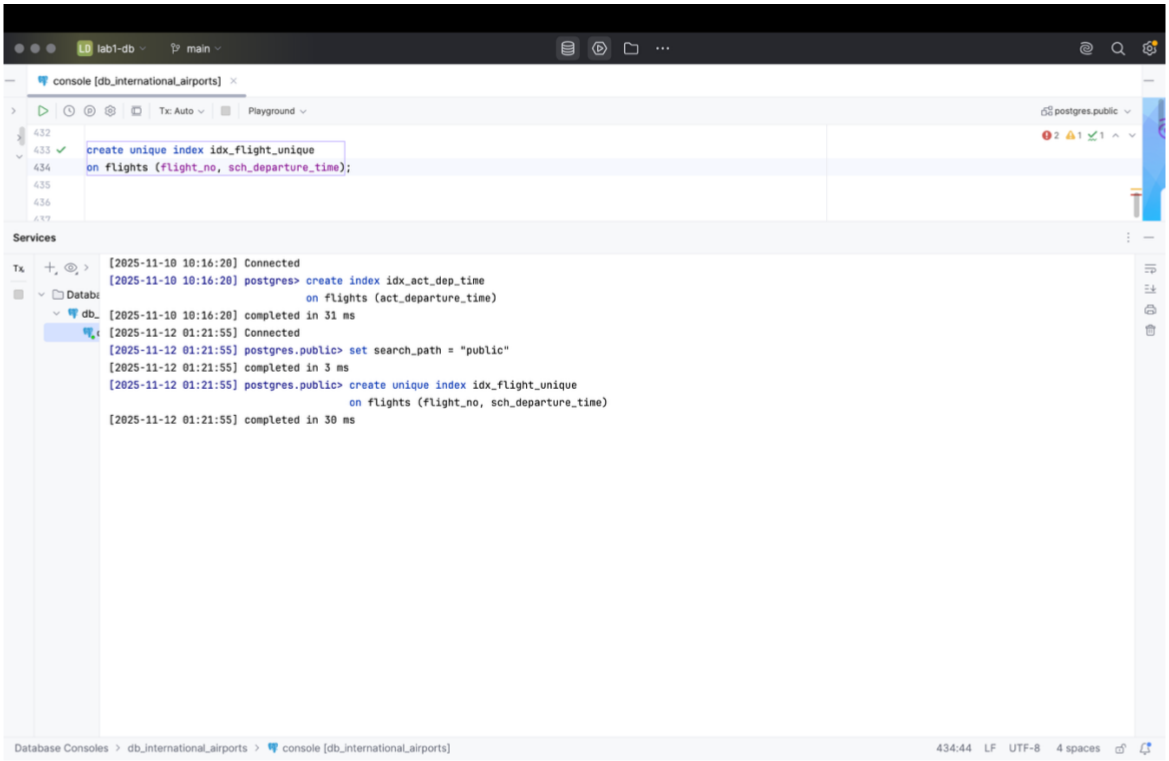


lab1-db
main
console [db_international_airports]
Tx: Auto
Playground
postgres.public
Services
Database Consoles > db_international_airports > console [db_international_airports]
431:33 LF UTF-8 4 spaces

```
create index idx_act_dep_time
on flights (act_departure_time);
```

```
[2025-11-10 10:16:20] Connected
[2025-11-10 10:16:20] postgres> create index idx_act_dep_time
                                on flights (act_departure_time)
[2025-11-10 10:16:20] completed in 31 ms
```

2. Create a unique index to ensure flight_no and scheduled_departure combinations are unique.

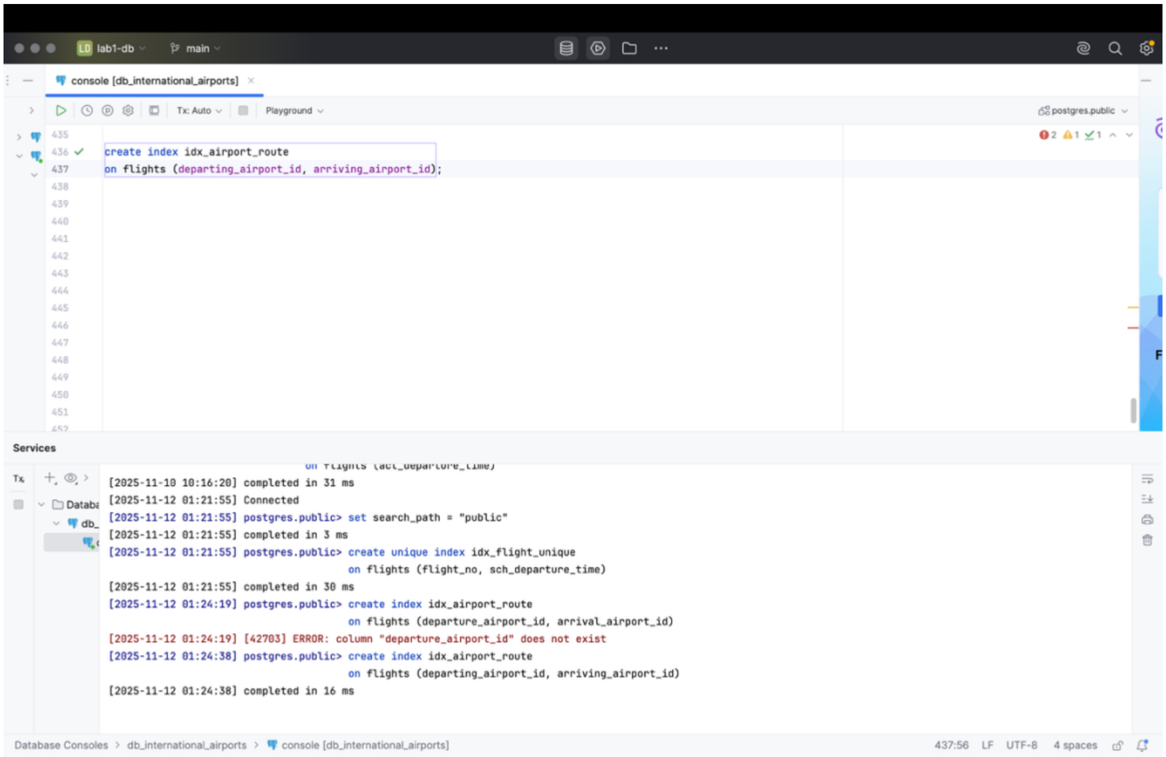


lab1-db
main
console [db_international_airports]
Tx: Auto
Playground
postgres.public
Services
Database Consoles > db_international_airports > console [db_international_airports]
434:44 LF UTF-8 4 spaces

```
create unique index idx_flight_unique
on flights (flight_no, sch_departure_time);
```

```
[2025-11-10 10:16:20] Connected
[2025-11-10 10:16:20] postgres> create index idx_act_dep_time
                                on flights (act_departure_time)
[2025-11-10 10:16:20] completed in 31 ms
[2025-11-12 01:21:55] Connected
[2025-11-12 01:21:55] postgres.public> set search_path = "public"
[2025-11-12 01:21:55] completed in 3 ms
[2025-11-12 01:21:55] postgres.public> create unique index idx_flight_unique
                                       on flights (flight_no, sch_departure_time)
[2025-11-12 01:21:55] completed in 30 ms
```

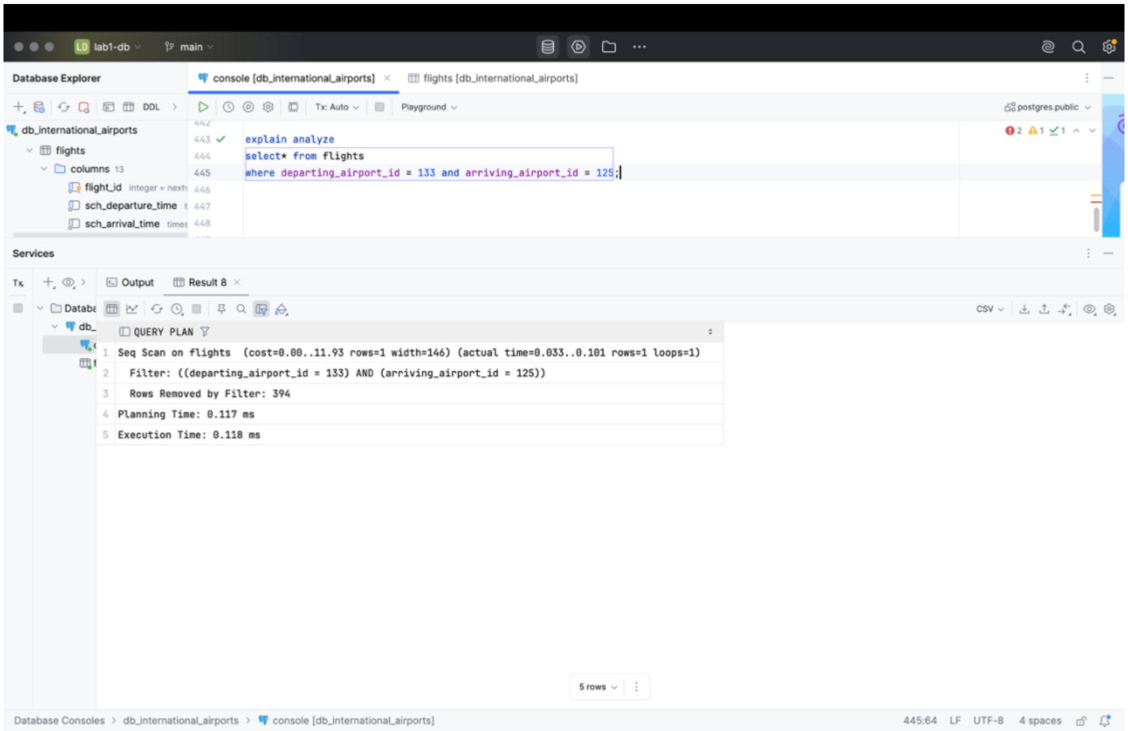
3. Create a composite index on the departure_airport_id and arrival_airport_id columns.


lab1-db
main
console [db_international_airports]
Tx: Auto
Playground
postgres.public
Services
Database Consoles > db_international_airports > console [db_international_airports]
437:56 LF UTF-8 4 spaces

```
create index idx_airport_route
on flights (departing_airport_id, arriving_airport_id);
```

```
                                  on flights (act_departure_time)
[2025-11-10 10:16:20] completed in 31 ms
[2025-11-12 01:21:55] Connected
[2025-11-12 01:21:55] postgres.public> set search_path = "public"
[2025-11-12 01:21:55] completed in 3 ms
[2025-11-12 01:21:55] postgres.public> create unique index idx_flight_unique
                                       on flights (flight_no, sch_departure_time)
[2025-11-12 01:21:55] completed in 30 ms
[2025-11-12 01:24:19] postgres.public> create index idx_airport_route
                                       on flights (departure_airport_id, arrival_airport_id)
[2025-11-12 01:24:19] [42703] ERROR: column "departure_airport_id" does not exist
[2025-11-12 01:24:38] postgres.public> create index idx_airport_route
                                       on flights (departing_airport_id, arriving_airport_id)
[2025-11-12 01:24:38] completed in 16 ms
```

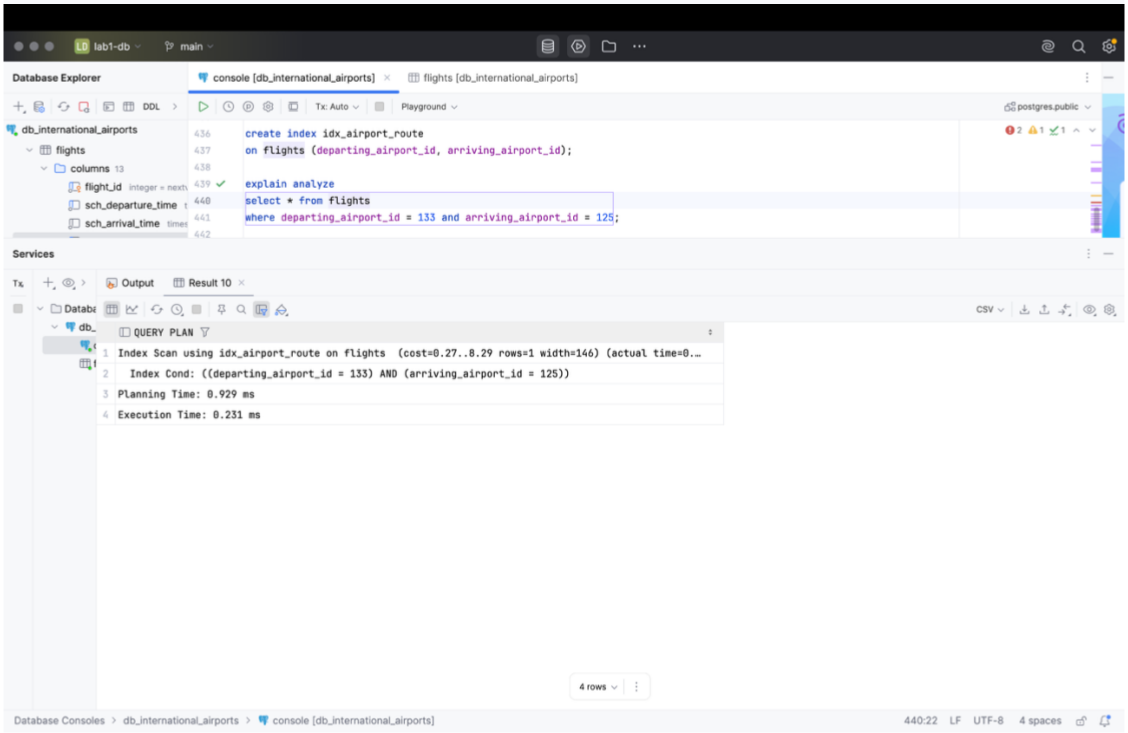
4. Evaluate the difference in query performance with and without indexes. Measure performance differences.



```
explain analyze
select* from flights
where departing_airport_id = 133 and arriving_airport_id = 125;
```

```
QUERY PLAN
Seq Scan on flights  (cost=0.00..11.93 rows=1 width=146) (actual time=0.033..0.101 rows=1 loops=1)
  Filter: ((departing_airport_id = 133) AND (arriving_airport_id = 125))
  Rows Removed by Filter: 394
Planning Time: 0.117 ms
Execution Time: 0.118 ms
```

lab1-db
main
Database Explorer
console [db_international_airports]
flights [db_international_airports]
DDL
Tx: Auto
Playground
postgres.public
db_international_airports
flights
columns 13
flight_id integer = nextv
sch_departure_time
sch_arrival_time
Services
Output
Result 8
CSV
5 rows
Database Consoles > db_international_airports > console [db_international_airports]
445:64 LF UTF-8 4 spaces

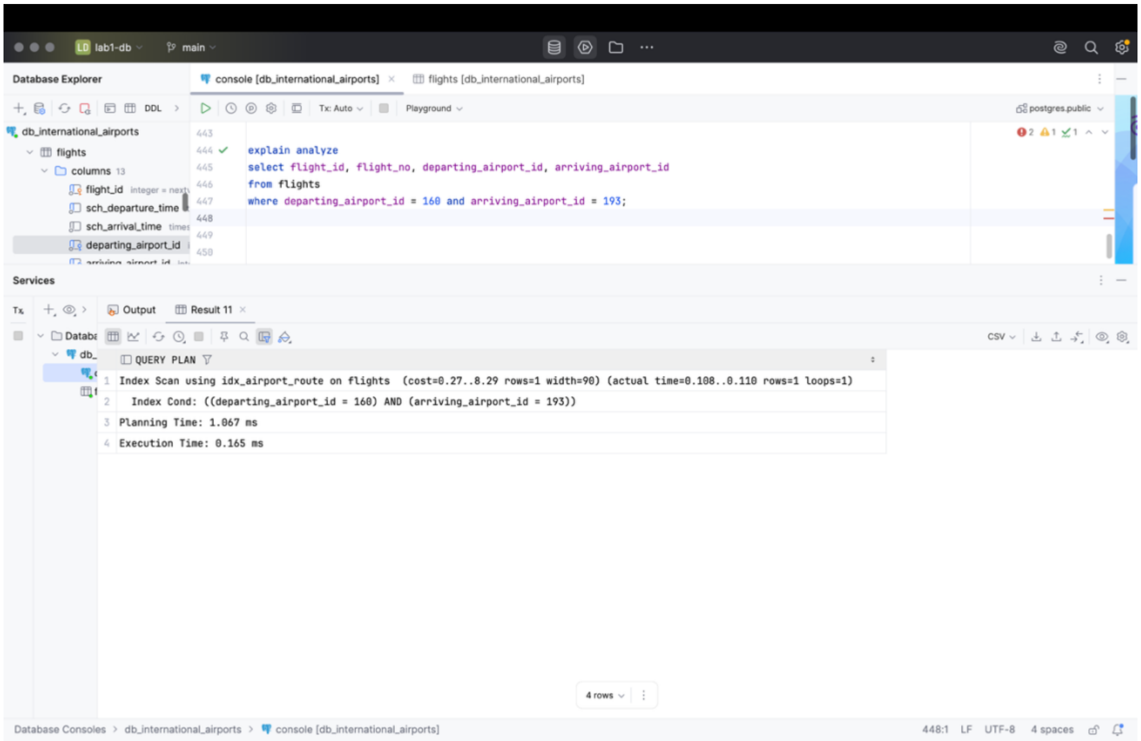


```
create index idx_airport_route
on flights (departing_airport_id, arriving_airport_id);

explain analyze
select * from flights
where departing_airport_id = 133 and arriving_airport_id = 125;
```

```
QUERY PLAN
Index Scan using idx_airport_route on flights  (cost=0.27..8.29 rows=1 width=146) (actual time=0.…
  Index Cond: ((departing_airport_id = 133) AND (arriving_airport_id = 125))
Planning Time: 0.929 ms
Execution Time: 0.231 ms
```

lab1-db
main
Database Explorer
console [db_international_airports]
flights [db_international_airports]
DDL
Tx: Auto
Playground
postgres.public
db_international_airports
flights
columns 13
flight_id integer = nextv
sch_departure_time
sch_arrival_time
Services
Output
Result 10
CSV
4 rows
Database Consoles > db_international_airports > console [db_international_airports]
440:22 LF UTF-8 4 spaces

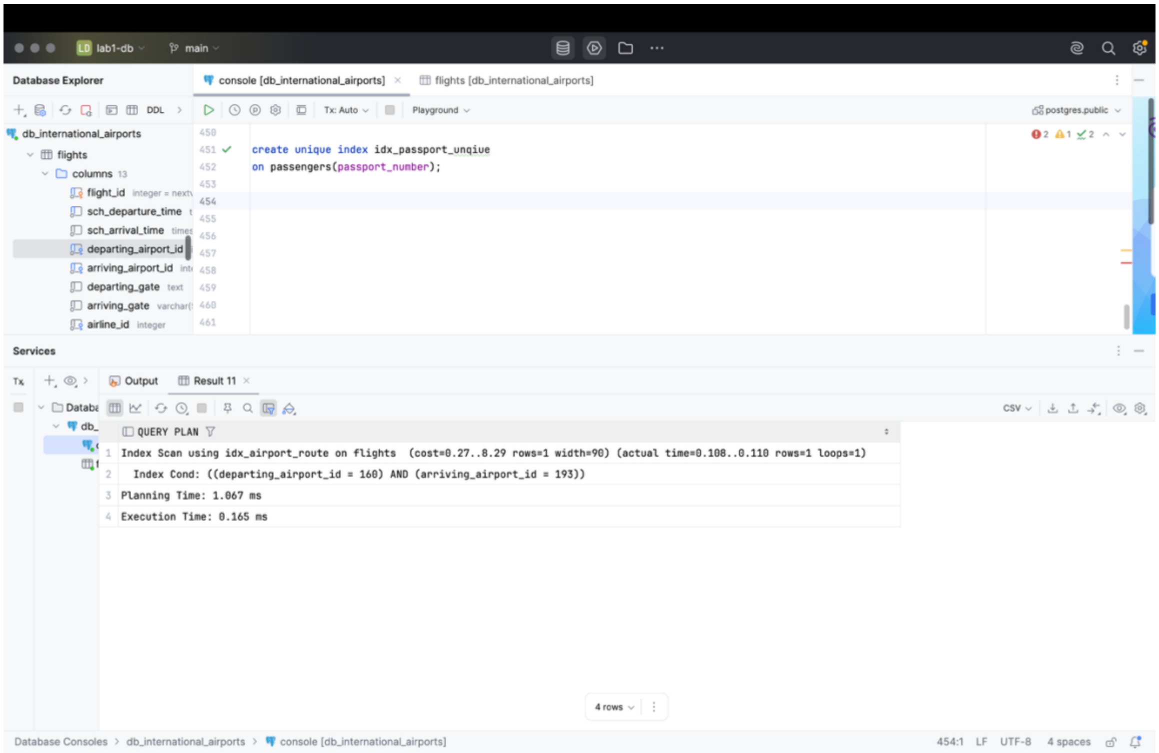
5. Use EXPLAIN ANALYZE to check index usage in a query filtering by departure_airport and arrival_airport.



```
explain analyze
select flight_id, flight_no, departing_airport_id, arriving_airport_id
from flights
where departing_airport_id = 160 and arriving_airport_id = 193;
```

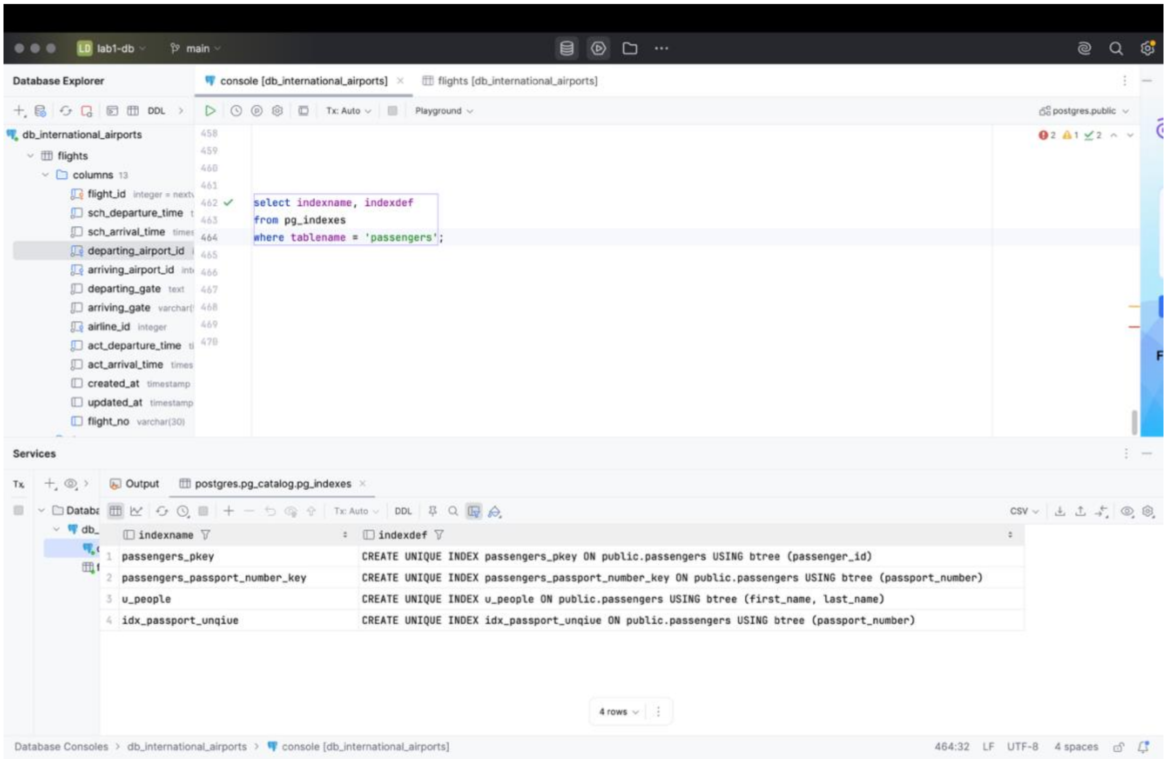
```
QUERY PLAN
1 Index Scan using idx_airport_route on flights  (cost=0.27..8.29 rows=1 width=90) (actual time=0.108..0.110 rows=1 loops=1)
2   Index Cond: ((departing_airport_id = 160) AND (arriving_airport_id = 193))
3 Planning Time: 1.067 ms
4 Execution Time: 0.165 ms
```

6. Create a unique index for the passport_number of the Passengers table. Check if the index was created or not. Insert into the table two new passengers. Explain in your own words what is going on in the output.



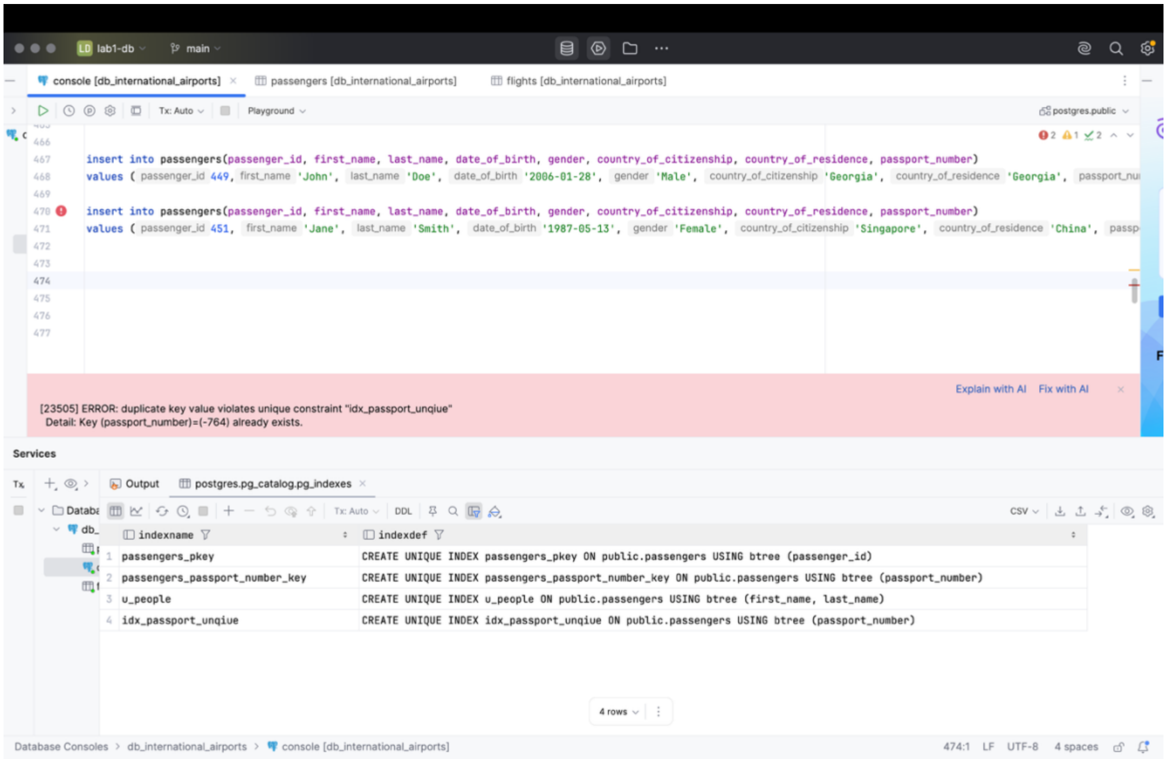
```
create unique index idx_passport_unqiue
on passengers(passport_number);
```

```
QUERY PLAN
1 Index Scan using idx_airport_route on flights  (cost=0.27..8.29 rows=1 width=90) (actual time=0.108..0.110 rows=1 loops=1)
2   Index Cond: ((departing_airport_id = 160) AND (arriving_airport_id = 193))
3 Planning Time: 1.067 ms
4 Execution Time: 0.165 ms
```



```sql
select indexname, indexdef
from pg_indexes
where tablename = 'passengers';
```

	indexname	indexdef
1	passengers_pkey	CREATE UNIQUE INDEX passengers_pkey ON public.passengers USING btree (passenger_id)
2	passengers_passport_number_key	CREATE UNIQUE INDEX passengers_passport_number_key ON public.passengers USING btree (passport_number)
3	u_people	CREATE UNIQUE INDEX u_people ON public.passengers USING btree (first_name, last_name)
4	idx_passport_unqiue	CREATE UNIQUE INDEX idx_passport_unqiue ON public.passengers USING btree (passport_number)



lab1-db
main
console [db_international_airports]
passengers [db_international_airports]
flights [db_international_airports]
Tx: Auto
Playground
postgres.public
Explain with AI
Fix with AI
Services
Output
postgres.pg_catalog.pg_indexes
CSV
4 rows
Database Consoles > db_international_airports > console [db_international_airports]
474:1 LF UTF-8 4 spaces

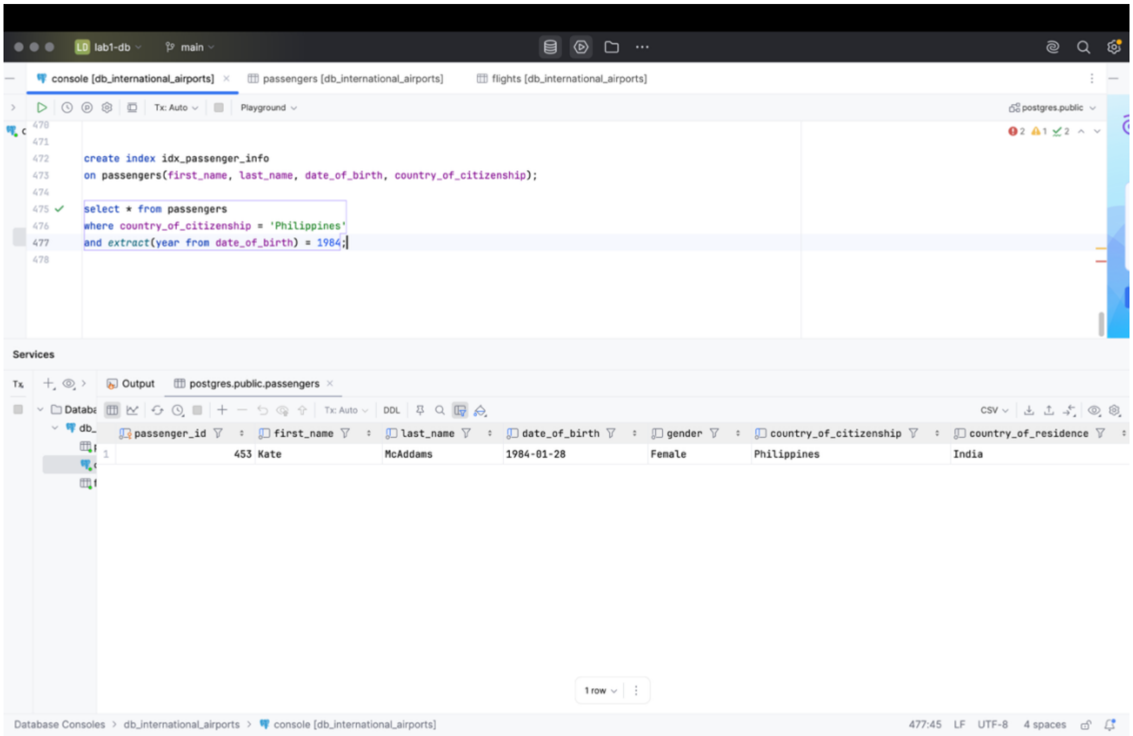
```sql
insert into passengers(passenger_id, first_name, last_name, date_of_birth, gender, country_of_citizenship, country_of_residence, passport_number)
values ( passenger_id 449, first_name 'John', last_name 'Doe', date_of_birth '2006-01-28', gender 'Male', country_of_citizenship 'Georgia', country_of_residence 'Georgia', passport_nu

insert into passengers(passenger_id, first_name, last_name, date_of_birth, gender, country_of_citizenship, country_of_residence, passport_number)
values ( passenger_id 451, first_name 'Jane', last_name 'Smith', date_of_birth '1987-05-13', gender 'Female', country_of_citizenship 'Singapore', country_of_residence 'China', passp
```

```
[23505] ERROR: duplicate key value violates unique constraint "idx_passport_unqiue"
 Detail: Key (passport_number)=(-764) already exists.
```

	indexname	indexdef
1	passengers_pkey	CREATE UNIQUE INDEX passengers_pkey ON public.passengers USING btree (passenger_id)
2	passengers_passport_number_key	CREATE UNIQUE INDEX passengers_passport_number_key ON public.passengers USING btree (passport_number)
3	u_people	CREATE UNIQUE INDEX u_people ON public.passengers USING btree (first_name, last_name)
4	idx_passport_unqiue	CREATE UNIQUE INDEX idx_passport_unqiue ON public.passengers USING btree (passport_number)

We created a unique index for passport column ensuring that no duplicates can be added into the table passengers. This error displays that index we created is working. The first time we successfully inserted a new passenger but then when we tried to add a passenger with the same passport number as the previous one, the passport index is not letting us to do it.

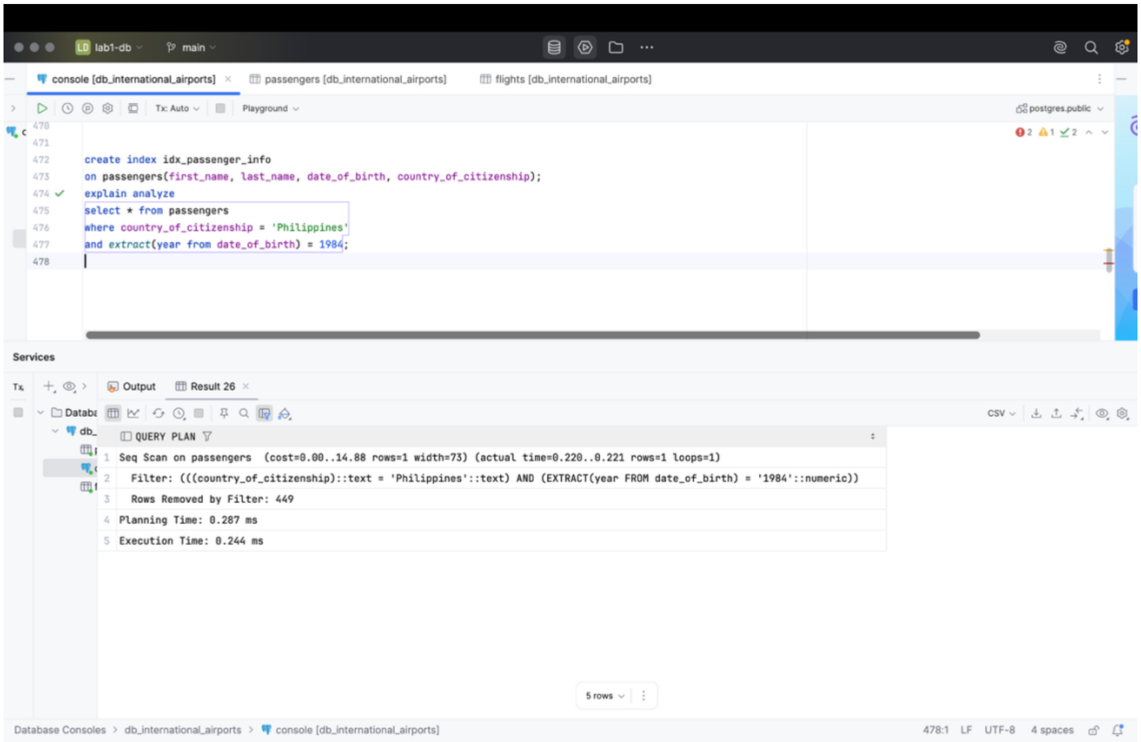
7. Create an index for the Passengers table. Use for that first name, last name, date of birth, country of citizenship. Then write a SQL query to find a passenger who was born in Philippines and was born in 1984 and check if the query uses indexes or not. Give explanation of the results.



```
create index idx_passenger_info
on passengers(first_name, last_name, date_of_birth, country_of_citizenship);

select * from passengers
where country_of_citizenship = 'Philippines'
and extract(year from date_of_birth) = 1984;
```

	passenger_id	first_name	last_name	date_of_birth	gender	country_of_citizenship	country_of_residence
1	453	Kate	McAddams	1984-01-28	Female	Philippines	India

lab1-db
main
console [db_international_airports]
passengers [db_international_airports]
flights [db_international_airports]
Tx: Auto
Playground
postgres.public
Services
Output
postgres.public.passengers
DDL
CSV
1 row
Database Consoles > db_international_airports > console [db_international_airports]
477:45 LF UTF-8 4 spaces



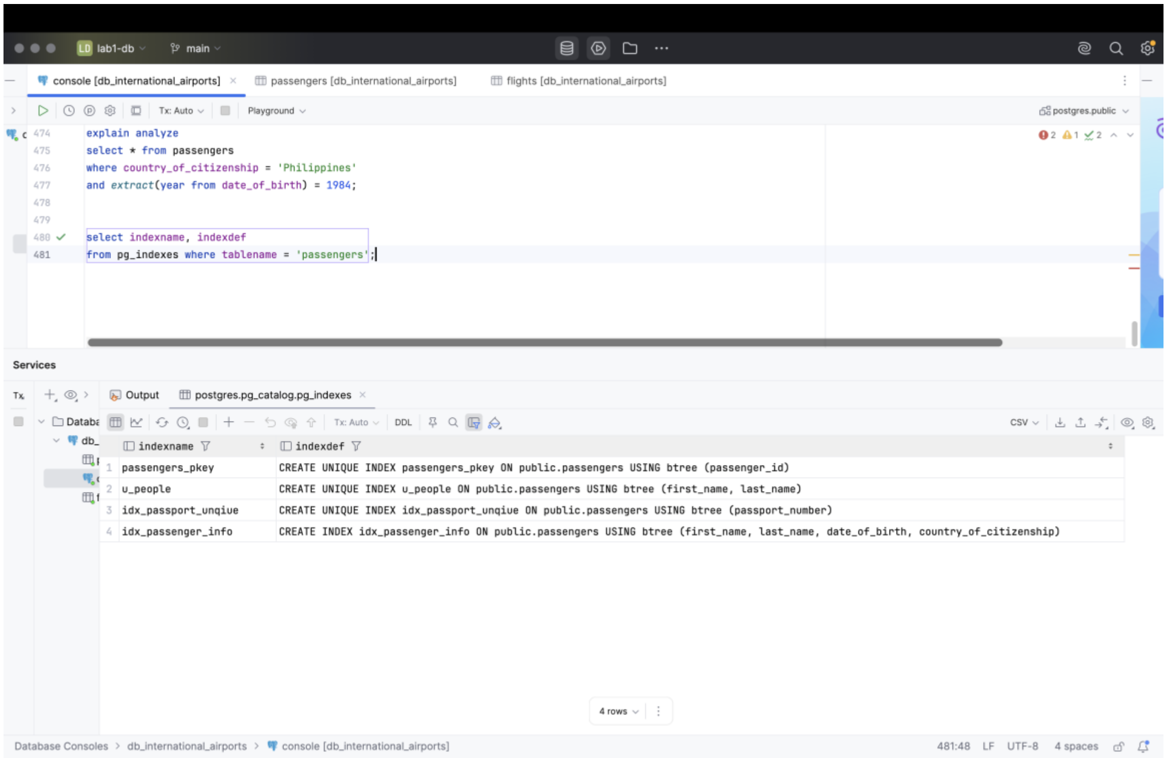
lab1-db
main
console [db_international_airports]
passengers [db_international_airports]
flights [db_international_airports]
Tx: Auto
Playground
postgres.public
Services
Output
Result 26
CSV
5 rows
Database Consoles > db_international_airports > console [db_international_airports]
478:1 LF UTF-8 4 spaces

```sql
create index idx_passenger_info
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;
```

```
QUERY PLAN
1 Seq Scan on passengers  (cost=0.00..14.88 rows=1 width=73) (actual time=0.220..0.221 rows=1 loops=1)
2   Filter: (((country_of_citizenship)::text = 'Philippines'::text) AND (EXTRACT(year FROM date_of_birth) = '1984'::numeric))
3   Rows Removed by Filter: 449
4 Planning Time: 0.287 ms
5 Execution Time: 0.244 ms
```

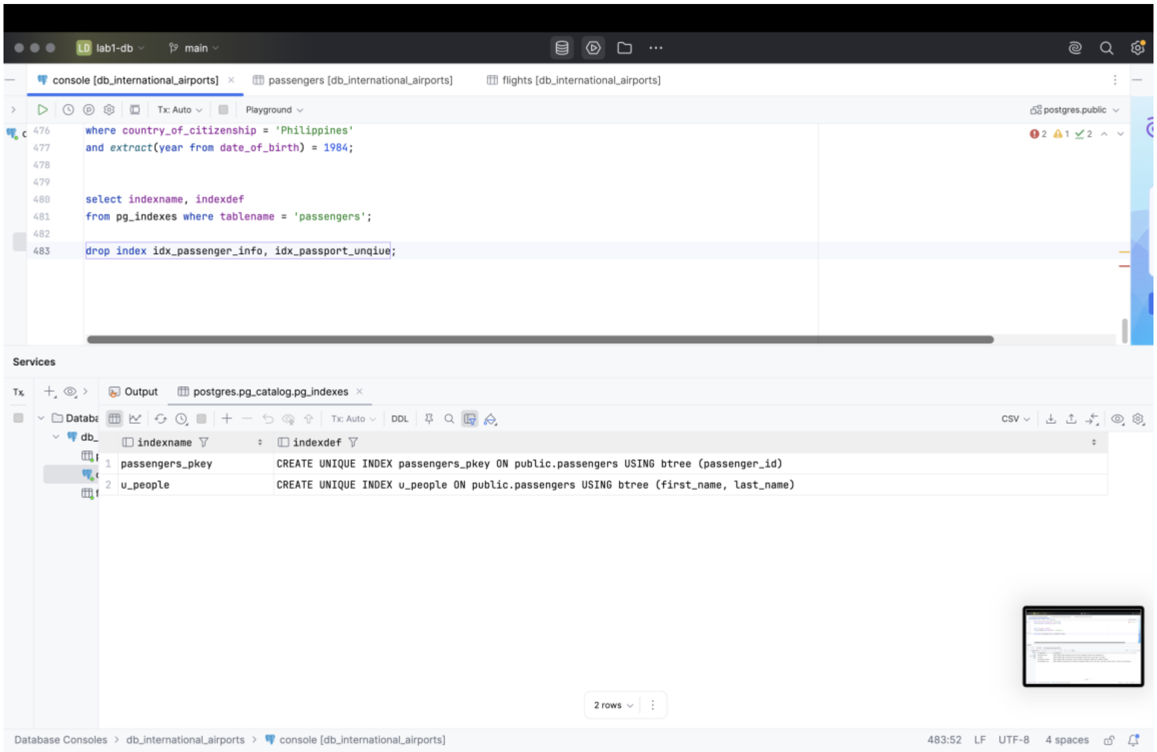
As we can see here a query did not use a composite index, we created to search for a passenger who was born in Philippines in 1984. Probably because the filter does not use the first indexed columns but instead uses ‘first_name’, ‘last_name’. Also, we used the function extract to find the year of birth so the query cannot use the index on that column.

8. Write a SQL query to list indexes for table Passengers. After delete the created indexes.


lab1-db
main
console [db_international_airports]
passengers [db_international_airports]
flights [db_international_airports]
Tx: Auto
Playground
postgres.public
Services
Output
postgres.pg_catalog.pg_indexes
CSV
4 rows
Database Consoles › db_international_airports › console [db_international_airports]
481:48 LF UTF-8 4 spaces

```sql
explain analyze
select * from passengers
where country_of_citizenship = 'Philippines'
and extract(year from date_of_birth) = 1984;


select indexname, indexdef
from pg_indexes where tablename = 'passengers';
```

	indexname	indexdef
1	passengers_pkey	CREATE UNIQUE INDEX passengers_pkey ON public.passengers USING btree (passenger_id)
2	u_people	CREATE UNIQUE INDEX u_people ON public.passengers USING btree (first_name, last_name)
3	idx_passport_unqiue	CREATE UNIQUE INDEX idx_passport_unqiue ON public.passengers USING btree (passport_number)
4	idx_passenger_info	CREATE INDEX idx_passenger_info ON public.passengers USING btree (first_name, last_name, date_of_birth, country_of_citizenship)


lab1-db
main
console [db_international_airports]
passengers [db_international_airports]
flights [db_international_airports]
Tx: Auto
Playground
postgres.public
Services
Output
postgres.pg_catalog.pg_indexes
CSV
2 rows
Database Consoles › db_international_airports › console [db_international_airports]
483:52 LF UTF-8 4 spaces

```sql
where country_of_citizenship = 'Philippines'
and extract(year from date_of_birth) = 1984;


select indexname, indexdef
from pg_indexes where tablename = 'passengers';

drop index idx_passenger_info, idx_passport_unqiue;
```

	indexname	indexdef
1	passengers_pkey	CREATE UNIQUE INDEX passengers_pkey ON public.passengers USING btree (passenger_id)
2	u_people	CREATE UNIQUE INDEX u_people ON public.passengers USING btree (first_name, last_name)