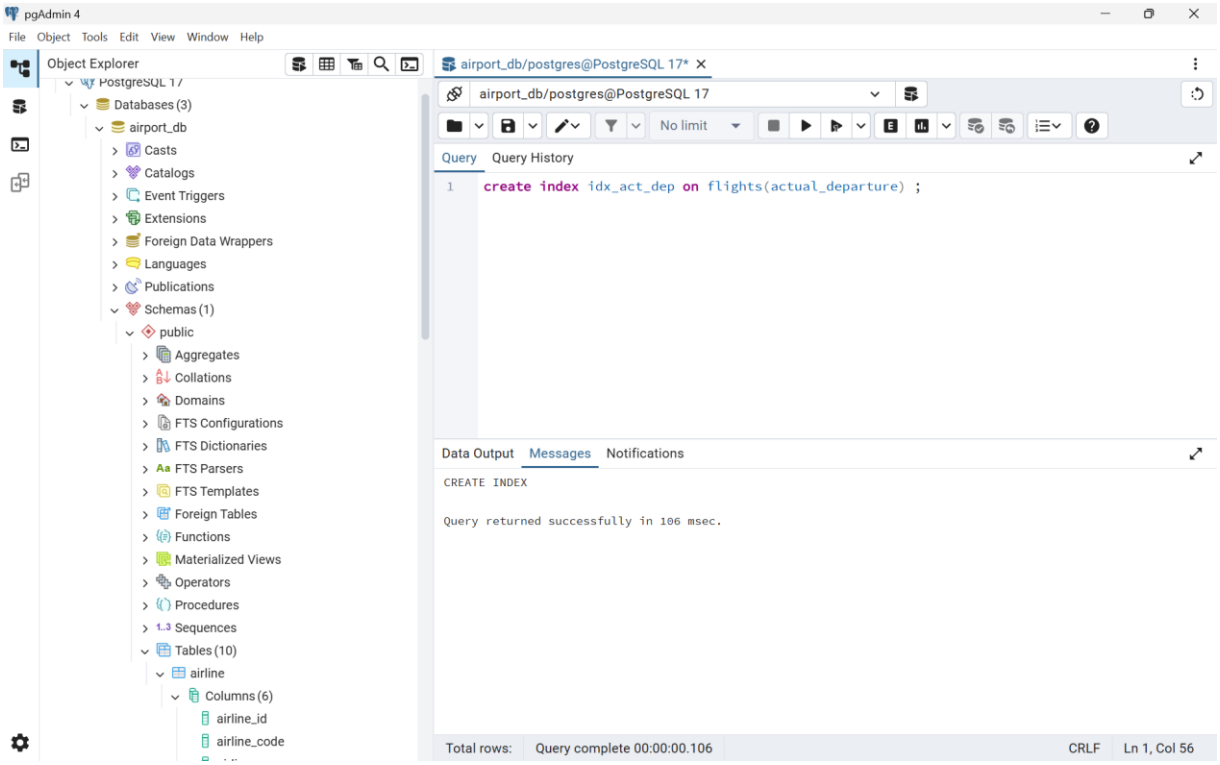
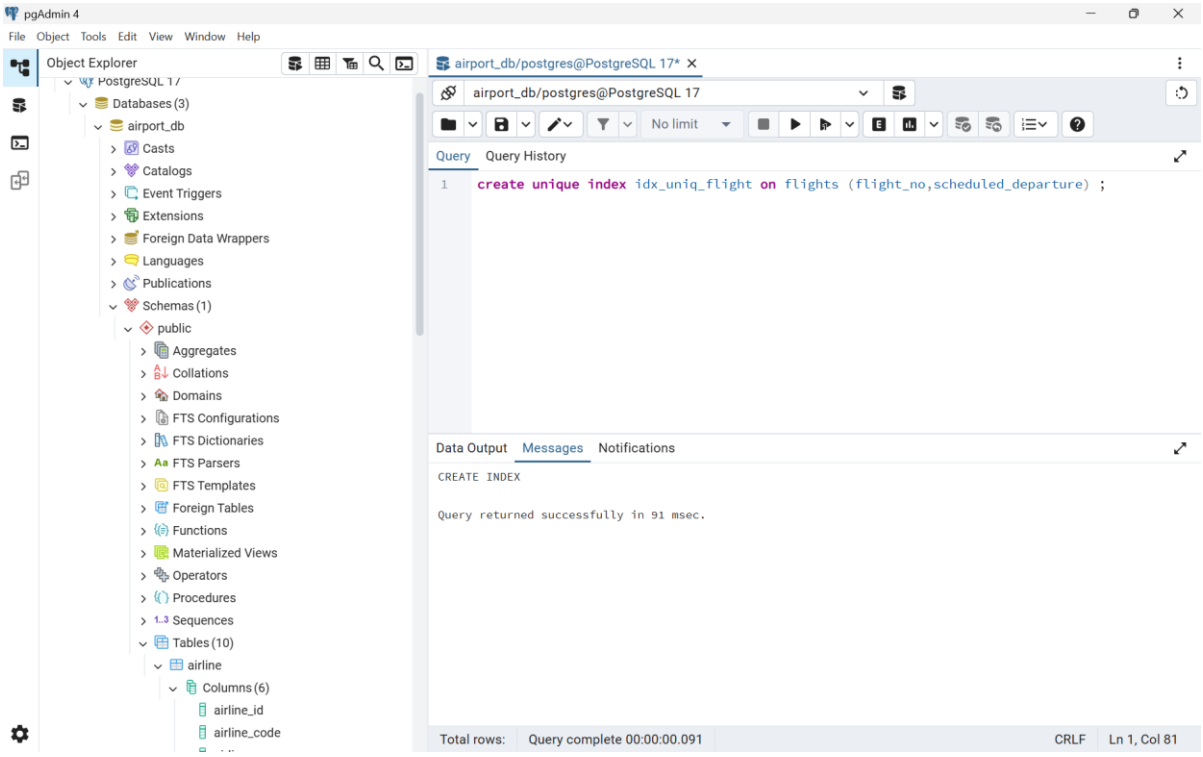
Laboratory work 7

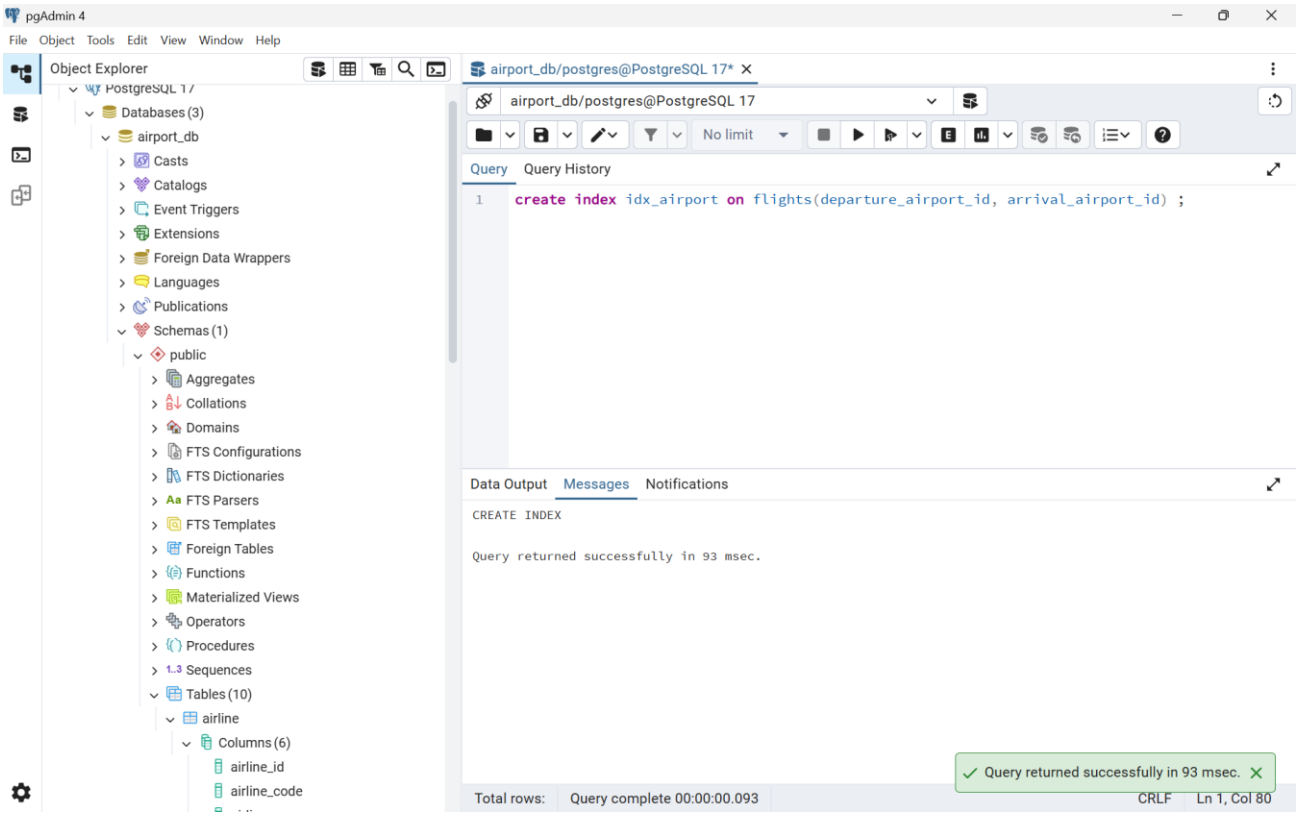
1. Create an index on the actual_departure column in the flights table.



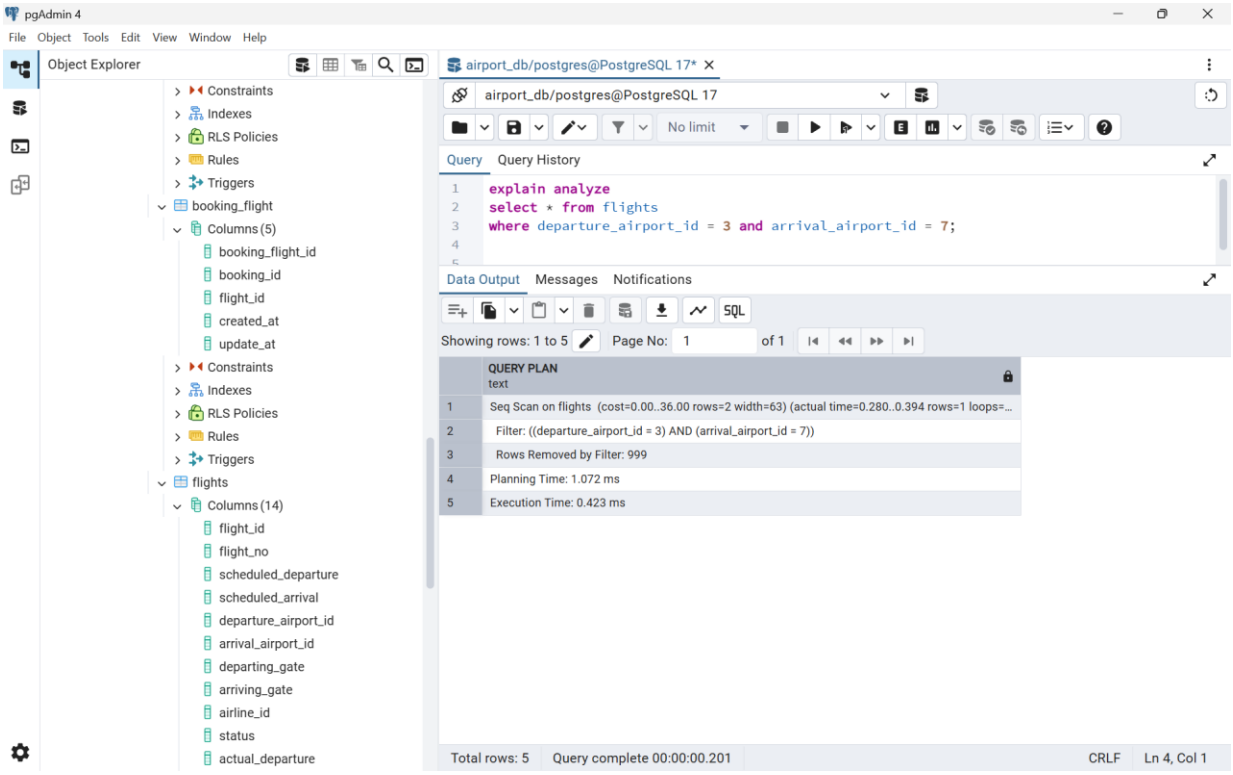
2. Create a unique index to ensure flight_no and scheduled_departure combinations are unique.

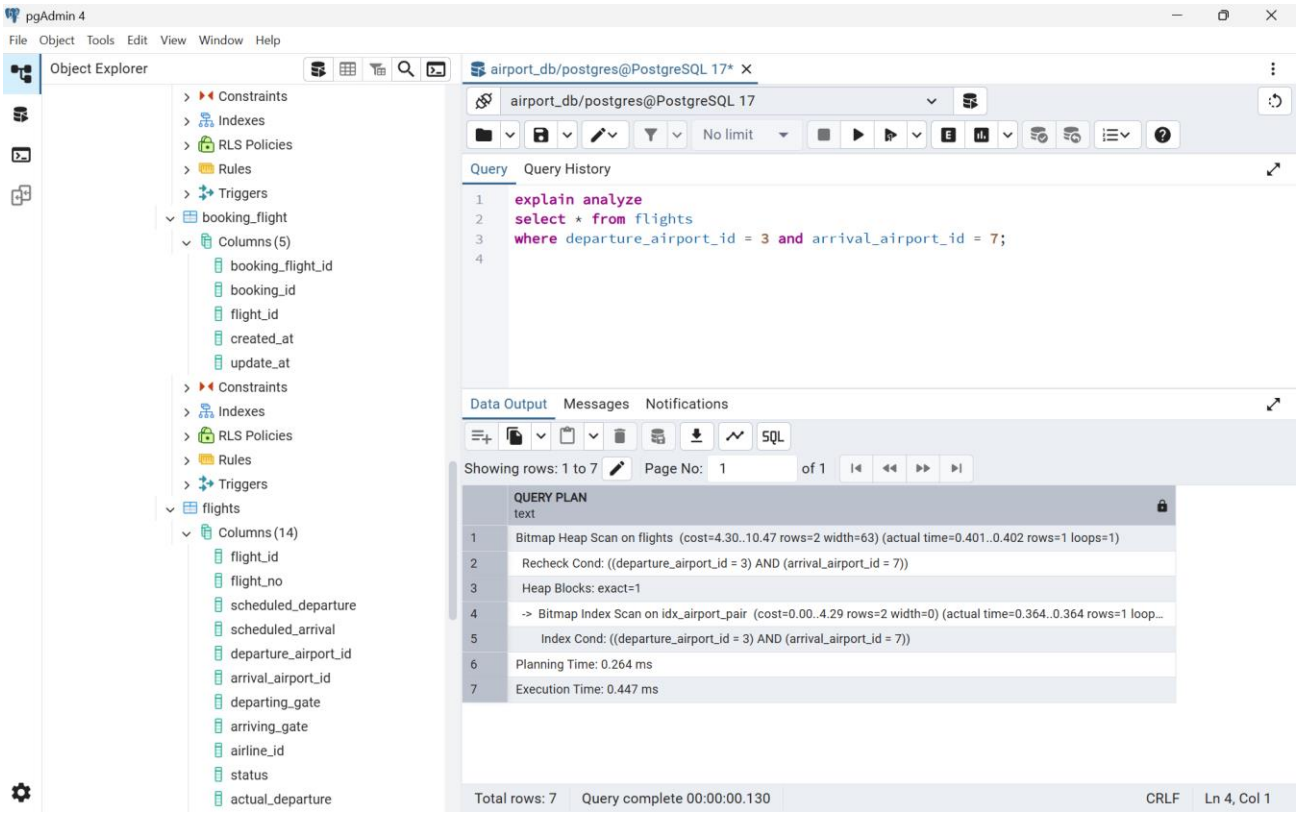


3. Create a composite index on the departure_airport_id and arrival_airport_id columns.

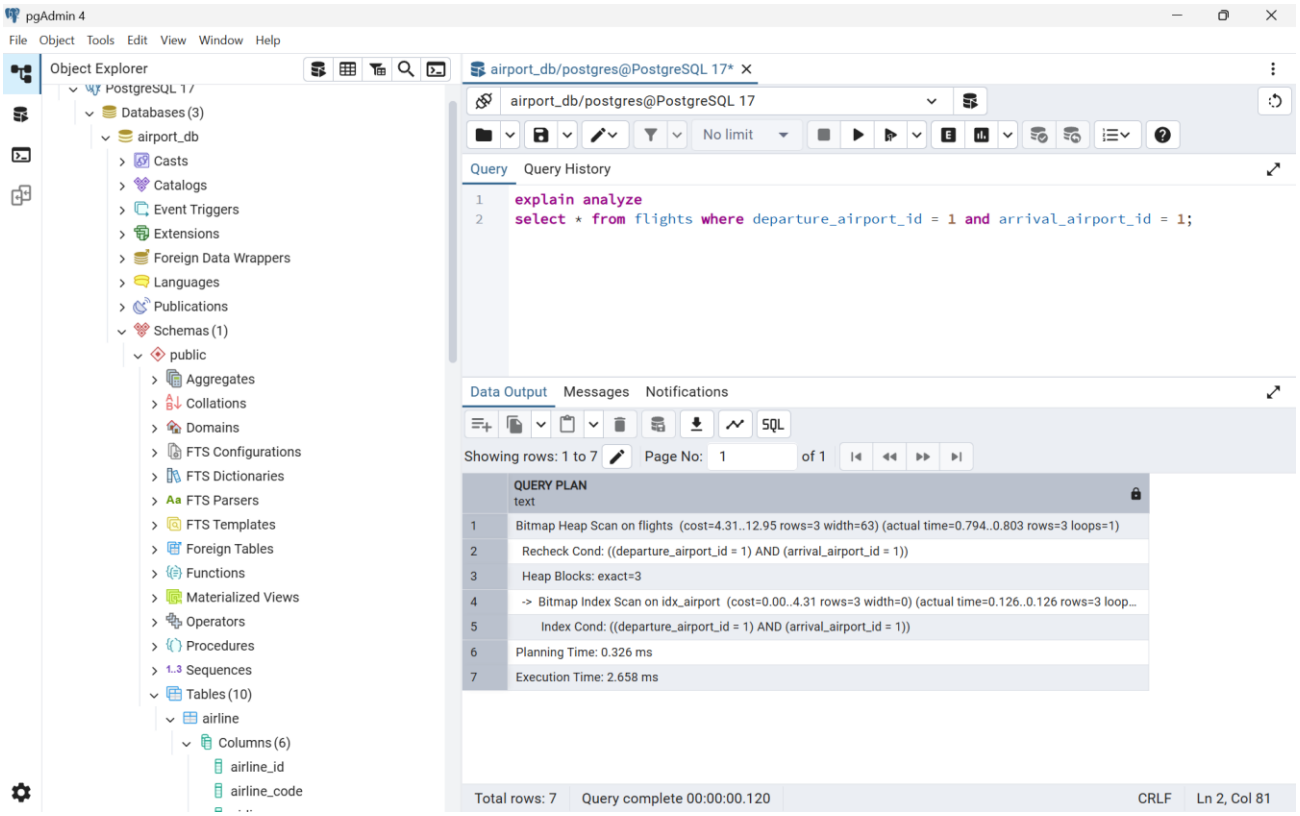


4. Evaluate the difference in query performance with and without indexes. Measure performance differences.





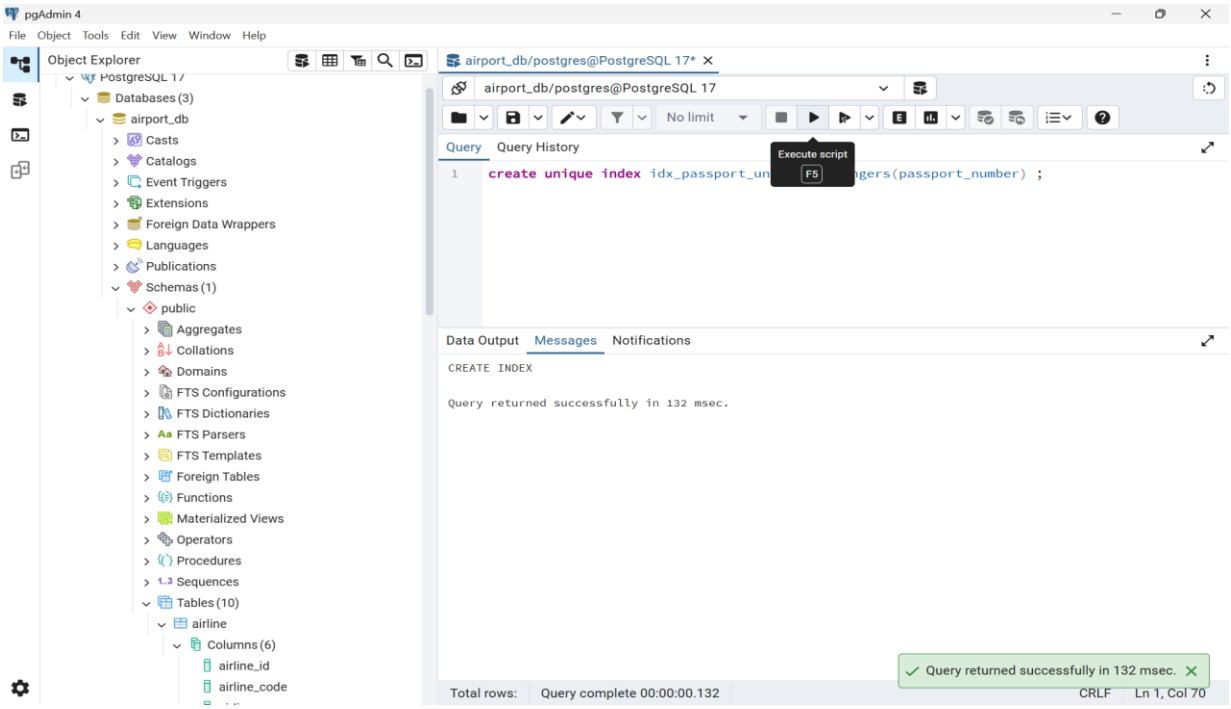
5. Use EXPLAIN ANALYZE to check index usage in a query filtering by departure_airport and arrival_airport.



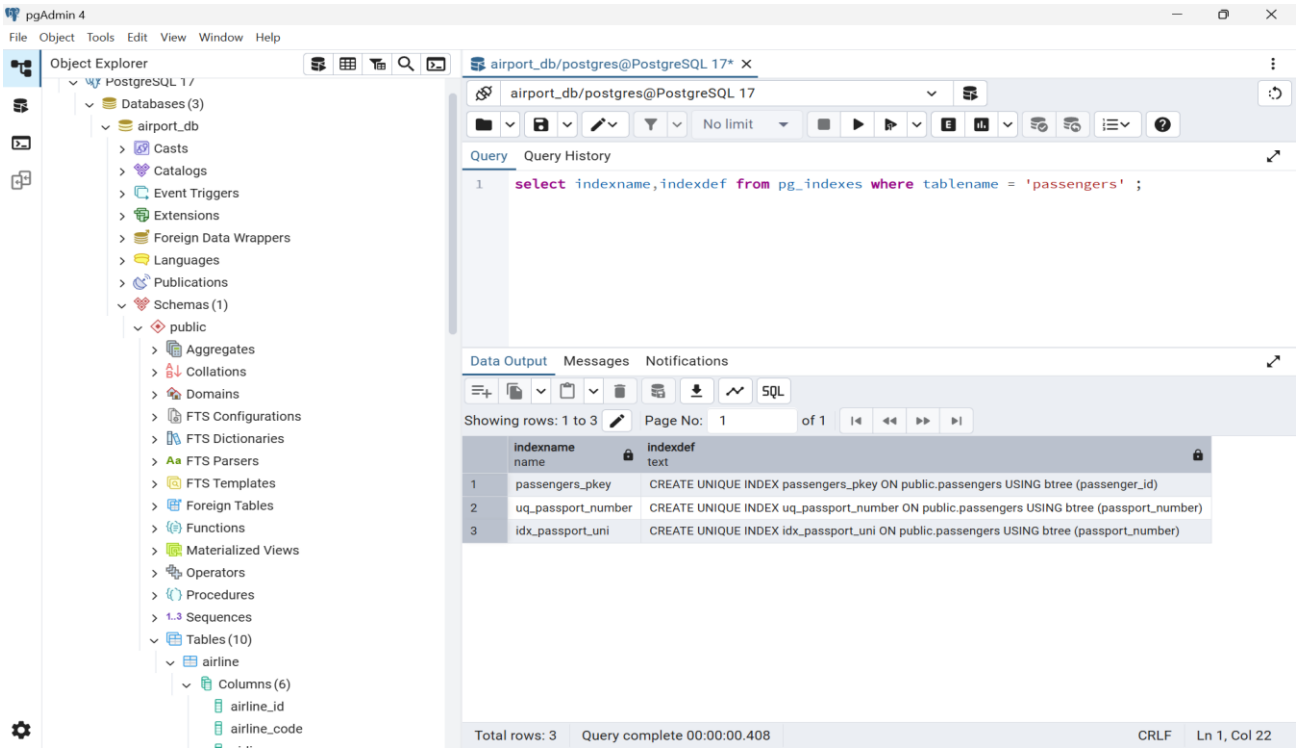
6. Create a unique index for the passport_number of the Passengers table. Check if the index was created or not. Insert into the table two new passengers.

Explain in your own words what is going on in the output?

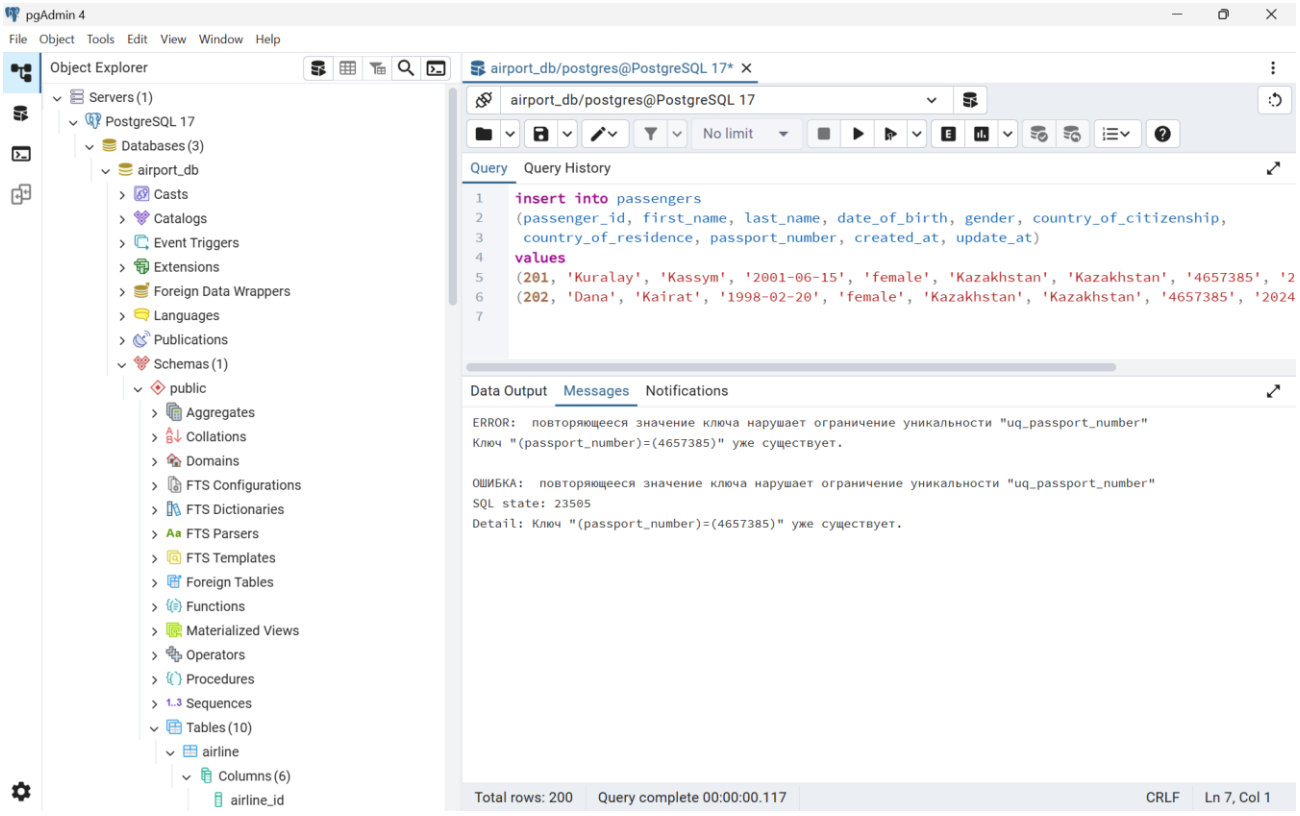
create



Check

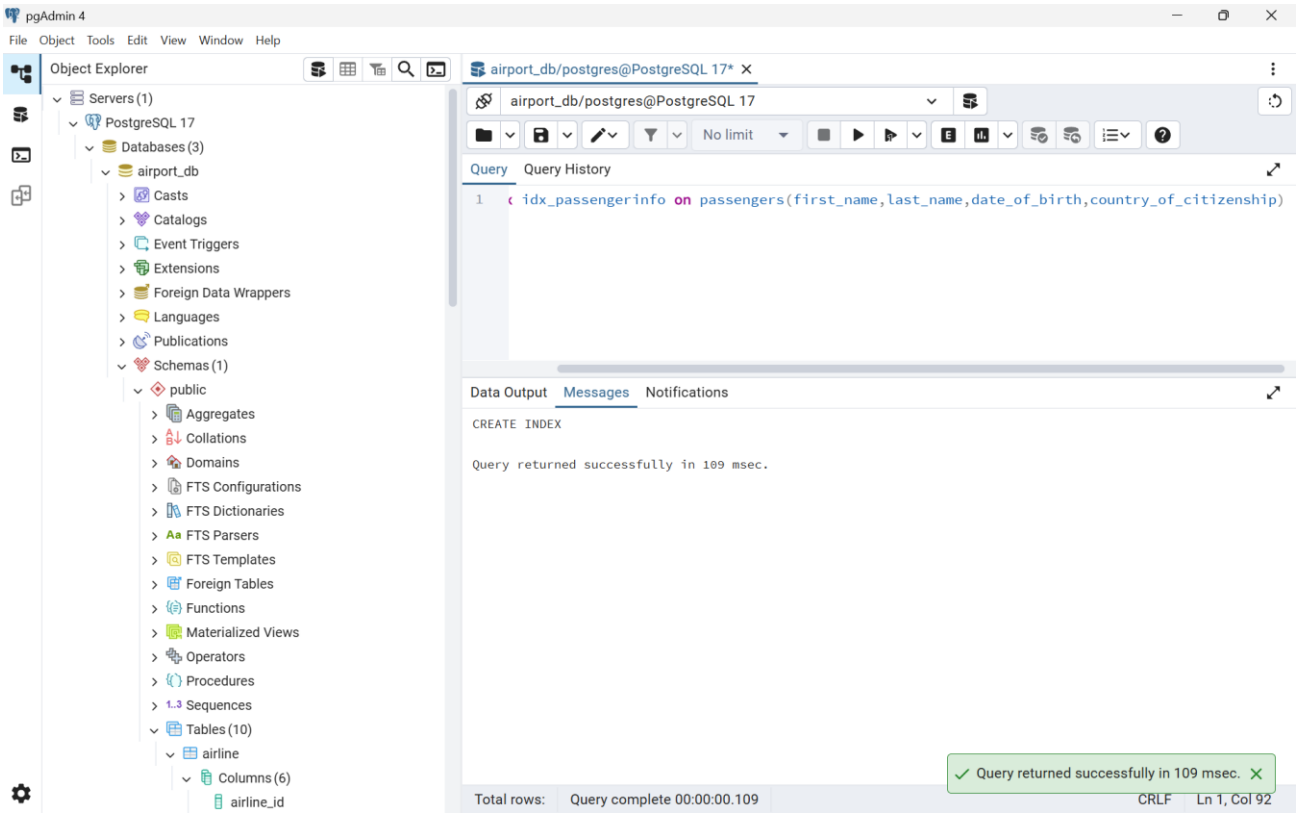


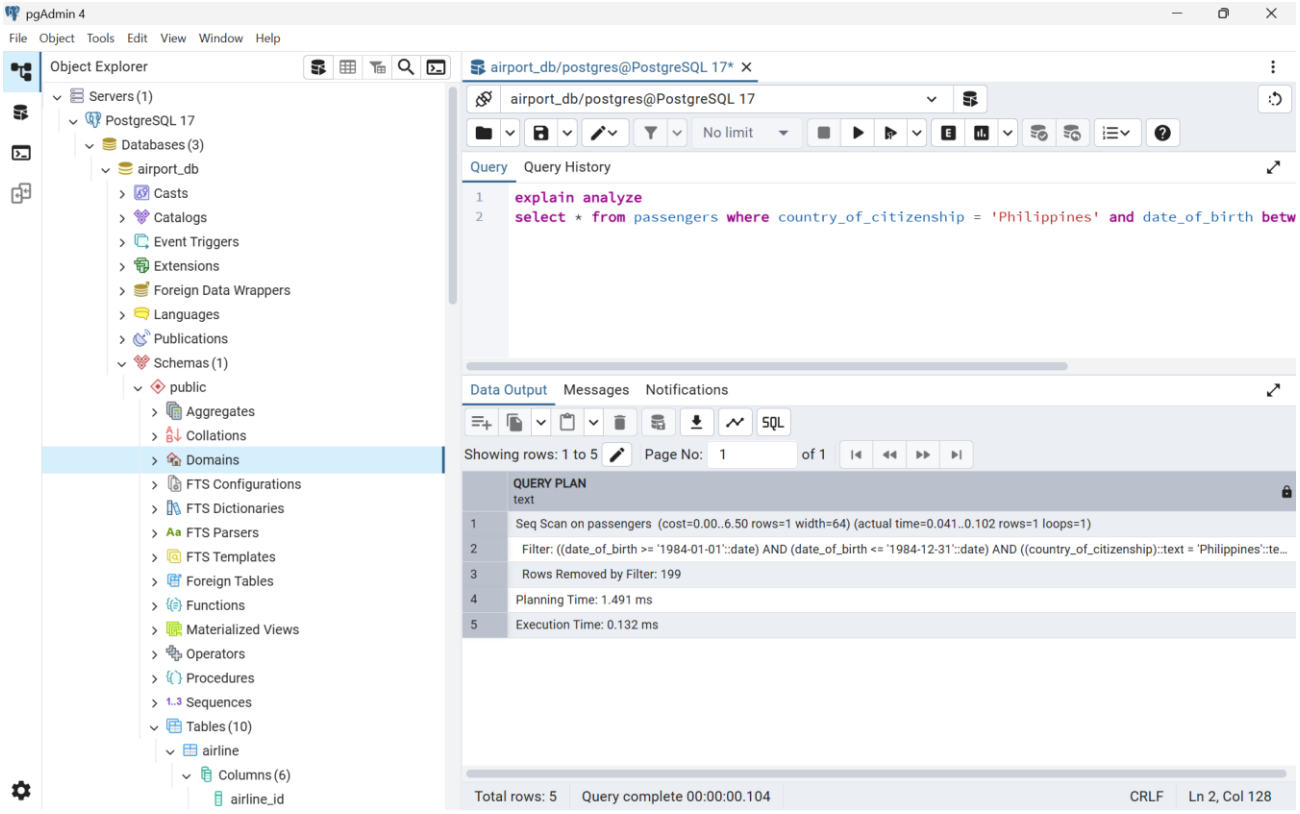
Insert 2 passengers



The unique index allows only one unique passport number per passenger,when I inserted 2 new passengers,it gave an error,because this 2 passengers have the same passport number. This happens because unique index prevents duplicate passport numbers in the table.

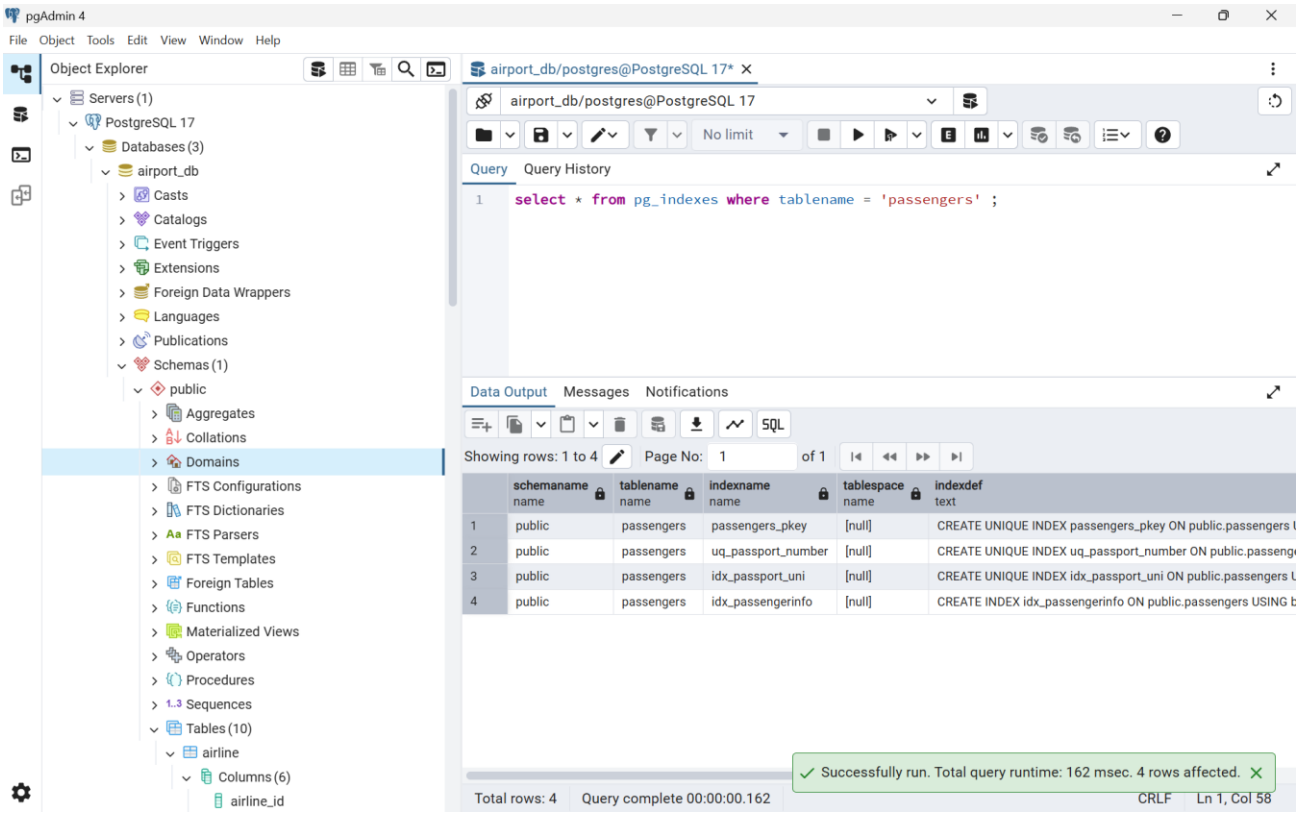
7. Create an index for the Passengers table. Use for that first name, last name, date of birth and country of citizenship. Then, write a SQL query to find a passenger who was born in Philippines and was born in 1984 and check if the query uses indexes or not. Give the explanation of the results.

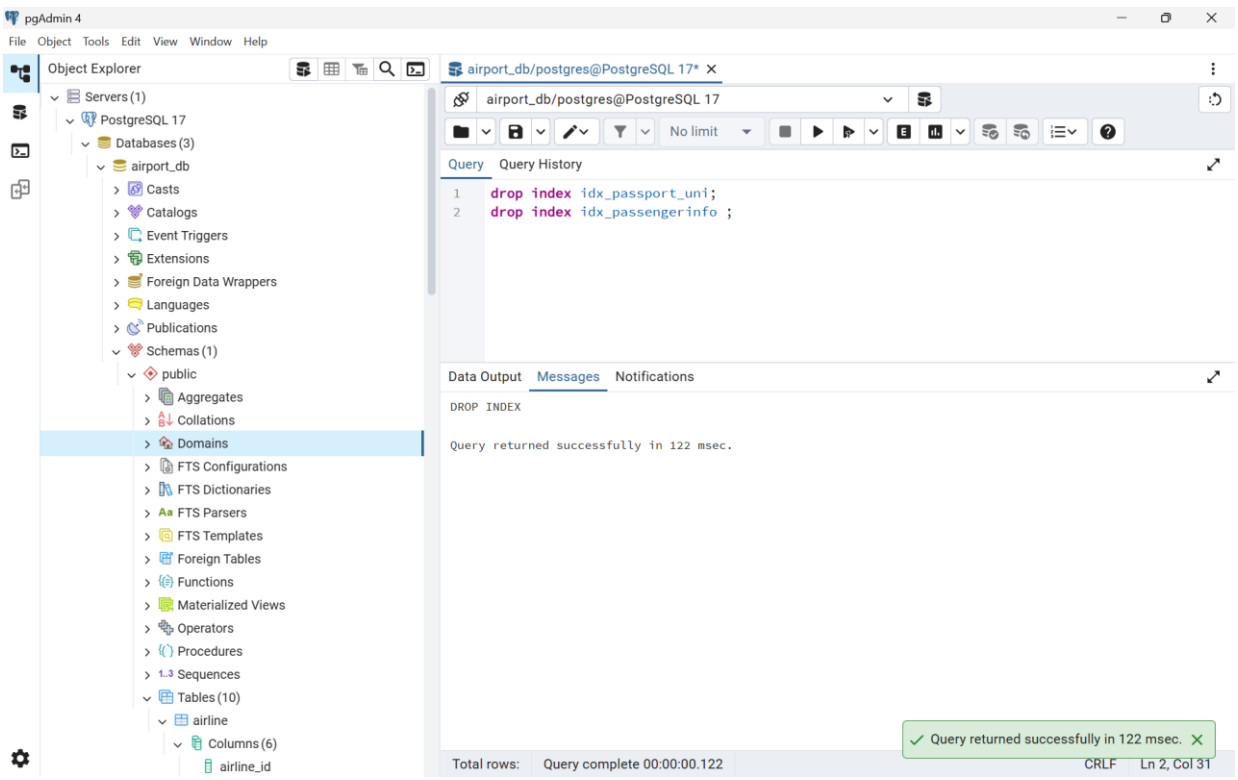




The query used seq scan , so it checked every roq in the table to find this passengers. It didn't use index that was created,because index is used from left to right,starting with the first columns that were defined in the index,also the table is small,so it would be faster than using the index , to scan all rows.

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


pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Servers (1)
PostgreSQL 17
Databases (3)
airport_db
Casts
Catalogs
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Publications
Schemas (1)
public
Aggregates
Collations
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (10)
airline
Columns (6)
airline_id
airport_db/postgres@PostgreSQL 17*
airport_db/postgres@PostgreSQL 17
No limit
Query
Query History
1 select * from pg_indexes where tablename = 'passengers' ;
Data Output
Messages
Notifications
SQL
Showing rows: 1 to 4
Page No: 1
of 1
schemaname name
tablename name
indexname name
tablespace name
indexdef text
1 public passengers passengers_pkey [null] CREATE UNIQUE INDEX passengers_pkey ON public.passengers
2 public passengers uq_passport_number [null] CREATE UNIQUE INDEX uq_passport_number ON public.passenge
3 public passengers idx_passport_uni [null] CREATE UNIQUE INDEX idx_passport_uni ON public.passengers
4 public passengers idx_passengerinfo [null] CREATE INDEX idx_passengerinfo ON public.passengers USING b
Successfully run. Total query runtime: 162 msec. 4 rows affected.
Total rows: 4
Query complete 00:00:00.162
CRLF
Ln 1, Col 58


pgAdmin 4
File Object Tools Edit View Window Help
Object Explorer
Servers (1)
PostgreSQL 17
Databases (3)
airport_db
Casts
Catalogs
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Publications
Schemas (1)
public
Aggregates
Collations
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (10)
airline
Columns (6)
airline_id
airport_db/postgres@PostgreSQL 17*
airport_db/postgres@PostgreSQL 17
No limit
Query
Query History
1 drop index idx_passport_uni;
2 drop index idx_passengerinfo ;
Data Output
Messages
Notifications
DROP INDEX
Query returned successfully in 122 msec.
Query returned successfully in 122 msec.
Total rows:
Query complete 00:00:00.122
CRLF
Ln 2, Col 31