

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

The screenshot shows the DBeaver interface with the following details:

- Database Navigator:** Shows the `flights` table under the `public` schema.
- Script Editor:** Displays the SQL command: `CREATE INDEX idx_actual_departure ON flights(act_departure_time)`.
- Statistics Panel:** Shows the execution statistics for the query.

Name	Value
Updated Rows	0
Execute time	0.022s
Start time	Tue Nov 11 21:49:13 ALMT 2025
Finish time	Tue Nov 11 21:49:13 ALMT 2025
Query	CREATE INDEX idx_actual_departure ON flights(act_departure_time)

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

The screenshot shows the DBeaver interface with the following details:

- Database Navigator:** Shows the `flights` table under the `public` schema.
- Script Editor:** Displays the SQL command: `CREATE UNIQUE INDEX idx_flight_sched_unique ON flights(flight_id, sch_departure_time);`.
- Statistics Panel:** Shows the execution statistics for the query.

Name	Value
Updated Rows	0
Execute time	0.011s
Start time	Tue Nov 11 22:28:24 ALMT 2025
Finish time	Tue Nov 11 22:28:24 ALMT 2025
Query	CREATE UNIQUE INDEX idx_flight_sched_unique ON flights(flight_id, sch_departure_time)

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

The screenshot shows the DBeaver 25.2.0 interface. The top bar displays the title "DBeaver 25.2.0 - <postgres> Script". The left sidebar shows the "Database Navigator" with a tree view of databases, schemas, and tables. The "Tables" section for the "flights" table is expanded, showing columns like flight_id, sch_departure_time, and departing_airport_id. The main panel contains a script editor with the following SQL command:

```
CREATE INDEX idx_airport_pair ON flights(departing_airport_id, arriving_airport_id);
```

Below the script editor, the "Statistics" tab is open, showing performance metrics for the last query execution. The "Updated Rows" section indicates 0 rows updated. The "Execute time" is 0.0002s, and the "Start time" and "Finish time" are both Tue Nov 11 22:29:59 ALMT 2025. The "Query" field shows the same CREATE INDEX command as the editor.

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

The screenshot displays the Beaver 25.2.0 application interface. On the left, a SQL editor window shows a query to create an index on the `flights` table, followed by an `EXPLAIN ANALYZE` command. The results pane below it shows the execution plan and statistics for the query. On the right, another SQL editor window shows a modified query where the `arriving_airport_id` is set to 43. The results pane below it shows the execution plan and statistics for this modified query.

Left Window (Query Editor):

```
CREATE INDEX idx_airport_pair ON flights(departing_airport_id, arriving_airport_id);
EXPLAIN ANALYZE
SELECT * FROM flights
WHERE departing_airport_id = '2' AND arriving_airport_id = '43';
```

Left Window (Results):

Results 1 X

```
EXPLAIN ANALYZE SELECT * FROM flights Enter a SQL expression to filter results (use Ctrl+Space)
```

AZ QUERY PLAN

1	Seq Scan on flights (cost=0.00..6.00 rows=1 width=72) (actual time=0.040..0.041 rows=0 loops=1)
2	Filter: ((departing_airport_id = 2) AND (arriving_airport_id = 43))
3	Rows Removed by Filter: 200
4	Planning Time: 0.124 ms
5	Execution Time: 0.056 ms

Right Window (Query Editor):

```
DROP INDEX IF EXISTS idx_airport_pair;
EXPLAIN ANALYZE
SELECT * FROM flights
WHERE departing_airport_id = '2' AND arriving_airport_id = '43';
```

Right Window (Results):

AZ QUERY PLAN

1	Seq Scan on flights (cost=0.00..6.00 rows=1 width=72) (actual time=0.151..0.151 rows=0 loops=1)
2	Filter: ((departing_airport_id = 2) AND (arriving_airport_id = 43))
3	Rows Removed by Filter: 200
4	Planning Time: 8.350 ms
5	Execution Time: 1.176 ms

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

DBeaver 25.2.0 - <postgres> Script

Auto postgres public *<postgres>... <none> adas... <postgres> ... flig

Database Naviga... X Projects

Connections Album public *<postgres>... <none> adas... <postgres> ... flig

filter connections by name

Databases

postgres

Schemas

public

Tables

- > airline 112K
- > airport 96K
- > baggage 56K
- > baggage_check 56K
- > boarding_pass 56K
- > booking 64K
- > booking_flight 56K
- > flights 128K

Columns

- 123 flight_id (int4)
- sch_departure_time (timestamp)
- sch_arrival_time (timestamp)
- 123 departing_airport_id (int4)
- 123 arriving_airport_id (int4)
- AZ departing_gate (varchar(50))
- AZ arriving_gate (varchar(50))
- 123 airline_id (int4)
- act_departure_time (timestamp)
- act_arrival_time (timestamp)
- created_at (timestamp)
- updated_at (timestamp)

Constraints

Foreign Keys

Indexes

Dependencies

Preferences

EXPLAIN ANALYZE

```
SELECT * FROM flights
WHERE departing_airport_id = '2' AND arriving_airport_id = '43';
```

Results 1 X

EXPLAIN ANALYZE SELECT * FROM flights | Enter a SQL expression to filter results (use Ctrl+Space)

Grid

1	AZ QUERY PLAN
2	Seq Scan on flights (cost=0.00..6.00 rows=1 width=72) (actual time=0.052..0.053 rows=0 loops=1)
3	Filter: ((departing_airport_id = 2) AND (arriving_airport_id = 43))
4	Rows Removed by Filter: 200
5	Planning Time: 1.092 ms
	Execution Time: 0.104 ms

Text

Record

Refresh Save Cancel Export data 200 5

5 row(s) fetched - 0.007s (0.001s fetch), on 2025-11-12 at 00:12:04

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?

DBeaver 25.2.0 - <postgres> Script

Database Naviga... X Projects

Connections Album public *<postgres>... X <none> adas... <postgres> .

Filter connections by name

Databases

postgres

Schemas

public

Tables

- airline 112K
- airport 96K
- baggage 56K
- baggage_check 56K
- boarding_pass 56K
- booking 64K
- booking_flight 56K
- flights 128K

Columns

- flight_id (int4)
- sch_departure_time (timestamp)
- sch_arrival_time (timestamp)
- departing_airport_id (int4)
- arriving_airport_id (int4)
- departing_gate (varchar(50))
- arriving_gate (varchar(50))
- airline_id (int4)
- act_departure_time (timestamp)
- act_arrival_time (timestamp)
- created_at (timestamp)
- updated_at (timestamp)

Constraints

Foreign Keys

Indexes

Dependencies

CREATE UNIQUE INDEX idx_passport_unique ON passengers(passport_number);

Statistics 1 X

Name	Value
Updated Rows	0
Execute time	0.013s
Start time	Wed Nov 12 00:12:25 ALMT 2025
Finish time	Wed Nov 12 00:12:25 ALMT 2025
Query	CREATE UNIQUE INDEX idx_passport_unique ON passengers(passport_number)

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.

Screenshot of DBeaver showing the creation of an index and its execution plan.

```

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

```

The results window shows the query plan:

```

explain ANALYZE SELECT * FROM passengers
WHERE country_of_citizenship = 'Philippines' AND date_of_birth = '1984-01-01';

1 Seq Scan on passengers (cost=0.00..6.00 rows=1 width=72) (actual time=0.059..0.060 rows=0 loops=1)
  Filter: (((country_of_citizenship)::text = 'Philippines'::text) AND (date_of_birth = '1984-01-01'::date))
  Rows Removed by Filter: 200
  Planning Time: 0.121 ms
  Execution Time: 0.075 ms

```

Statistics window for the created index:

Name	Value
Updated Rows	0
Execute time	0.008s
Start time	Wed Nov 12 00:13:41 ALMT 2025
Finish time	Wed Nov 12 00:13:41 ALMT 2025
Query	CREATE INDEX idx_passenger_data ON passengers(first_name, last_name, date_of_birth, country_of_citizenship)

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

Connections Album public *postgres... <none> adas... <postgres> ... Rights

```
④ SELECT indexname, indexdef
  FROM pg_indexes
 WHERE tablename = 'passengers';
```

72K

pg_indexes 1 ×

SELECT indexname, indexdef FROM pg_indexes Enter a SQL expression to filter results (use Ctrl+Space)

indexname	indexdef
pk_passengers	CREATE UNIQUE INDEX pk_passengers ON public.passengers USING btree (passenger_id)
idx_passport_unique	CREATE UNIQUE INDEX idx_passport_unique ON public.passengers USING btree (passport_number)
idx_passenger_data	CREATE INDEX idx_passenger_data ON public.passengers USING btree (first_name, last_name, date_of_birth, count)

Refresh Save Cancel Export data 200 3
3 row(s) fetched - 0.013s, on 2025-11-12 at 00:15:33

ALMT en Writable Smart Insert 4 : 1 : 75

мне нужно

DBeaver 25.2.0 - <postgres> Script

SQL Commit Rollback T Auto postgres public@postgres

base Naviga... Projects Connections

Connections by name

- act_arrival_time (timestamp)
- created_at (timestamp)
- updated_at (timestamp)

Constraints Foreign Keys Indexes Dependencies References Partitions Triggers Rules Policies

passengers 72K

Columns

- passenger_id (int4)
- first_name (varchar(50))
- last_name (varchar(50))
- date_of_birth (date)
- gender (varchar(50))
- country_of_citizenship (varchar(50))
- country_of_residence (varchar(50))
- passport_number (varchar(20))
- created_at (timestamp)
- updated_at (timestamp)

Constraints Foreign Keys Indexes Dependencies References Partitions Triggers

Statistics 1 ×

Name	Value
Updated Rows	0
Execute time	0.008s
Start time	Wed Nov 12 00:16:43 ALMT 2025
Finish time	Wed Nov 12 00:16:43 ALMT 2025
Query	DROP INDEX IF EXISTS idx_passport_unique