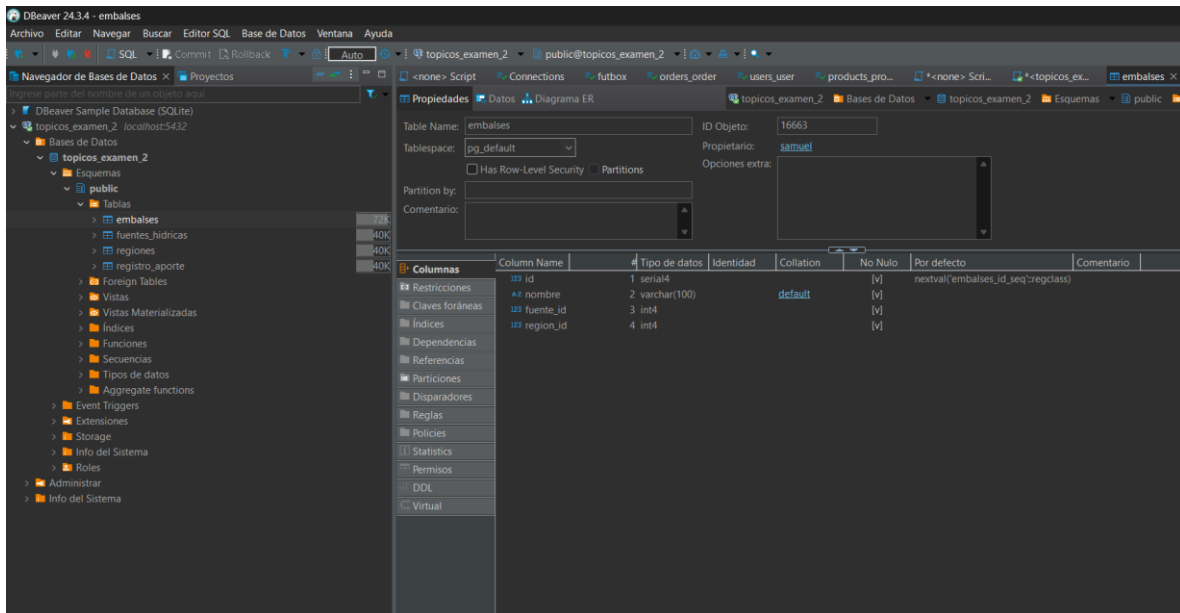


Primero creamos la base de datos y aca vemos el esquema.



Creamos el usuario y le damos los privilegios minimos para hacer CRUD.

```
CREATE USER 'samuel' WITH PASSWORD 'hola';
```

```
GRANT CONNECT ON DATABASE postgres TO samuel;

CREATE DATABASE temicos_examen_2;

\1

GRANT CONNECT ON DATABASE temicos_examen_2 TO samuel;

GRANT USAGE ON SCHEMA public TO samuel;

GRANT SELECT, INSERT, DELETE ON ALL TABLES IN SCHEMA public TO samuel;

GRANT CREATE, USAGE ON SCHEMA public TO samuel;

ALTER DEFAULT PRIVILEGES IN SCHEMA public
GRANT SELECT, INSERT, UPDATE, CREATE ON TABLES TO samuel;
```

Este es el primer plan de ejecución para la primera consulta del esquema inicial.

Resultados 1 ×	
EXPLAIN ANALYZE WITH dias_por_ano AS (SELECT ...)	
Enter a SQL expression to filter results (use Ctrl+Space)	
Grilla	○ AZ QUERY PLAN
1	Merge Join (cost=2766.54..3112.79 rows=29 width=72) (actual time=34.211..37.536 rows=12 loops=1)
2	Merge Cond: ((EXTRACT(year FROM registro_aporte.fecha)) = (((2023)::numeric))
3	-> GroupAggregate (cost=2766.50..3097.25 rows=2940 width=48) (actual time=34.184..37.500 rows=13 loops=1)
4	Group Key: (EXTRACT(year FROM registro_aporte.fecha)), registro_aporte.region_hidrologica
5	-> Sort (cost=2766.50..2840.00 rows=29400 width=44) (actual time=33.704..34.590 rows=20602 loops=1)
6	Sort Key: (EXTRACT(year FROM registro_aporte.fecha)), registro_aporte.region_hidrologica, registro_aporte.fecha
7	Sort Method: quicksort Memory: 1805kB
8	-> Seq Scan on registro_aporte (cost=0.00..584.50 rows=29400 width=44) (actual time=0.023..4.945 rows=29400 loops=1)
9	-> Sort (cost=0.04..0.05 rows=2 width=8) (actual time=0.018..0.020 rows=7 loops=1)
10	Sort Key: (((2023)::numeric))
11	Sort Method: quicksort Memory: 25kB
12	-> Result (cost=0.00..0.03 rows=2 width=8) (actual time=0.004..0.006 rows=2 loops=1)
13	-> Append (cost=0.00..0.03 rows=2 width=8) (actual time=0.002..0.003 rows=2 loops=1)
14	-> Result (cost=0.00..0.01 rows=1 width=8) (actual time=0.001..0.001 rows=1 loops=1)
15	-> Result (cost=0.00..0.01 rows=1 width=8) (actual time=0.000..0.000 rows=1 loops=1)
16	Planning Time: 0.190 ms
17	Execution Time: 37.746 ms

Este es el segundo plan de ejecución para la segunda consulta del esquema inicial.

Resultados 1 ×	
EXPLAIN ANALYZE WITH aportes_2024 AS (SELECT ...)	
Enter a SQL expression to filter results (use Ctrl+Space)	
Grilla	○ AZ QUERY PLAN
1	Subquery Scan on aportes_2024 (cost=663.29..665.60 rows=42 width=105) (actual time=9.006..11.184 rows=44 loops=1)
2	-> GroupAggregate (cost=663.29..665.18 rows=42 width=73) (actual time=9.002..11.155 rows=44 loops=1)
3	Group Key: registro_aporte.serie_hidrologica
4	-> Sort (cost=663.29..663.66 rows=147 width=15) (actual time=8.952..9.591 rows=11591 loops=1)
5	Sort Key: registro_aporte.serie_hidrologica
6	Sort Method: quicksort Memory: 747kB
7	-> Seq Scan on registro_aporte (cost=0.00..658.00 rows=147 width=15) (actual time=0.087..5.167 rows=11591 loops=1)
8	Filter: (EXTRACT(year FROM fecha) = '2024'::numeric)
9	Rows Removed by Filter: 17809
10	Planning Time: 0.091 ms
11	Execution Time: 11.216 ms

Plan de ejecución primera consulta del esquema final

	Grilla	Texto
Resultados 1 x		Enter a SQL expression to filter results (use Ctrl+Space)
T	AZ QUERY PLAN	
1	Sort (cost=73138.62..73293.62 rows=62000 width=286)	
2	Sort Key: e.nombre, (EXTRACT(year FROM (generate_series('2023-01-01'::date)::timestamp with time zone, ('2024-12-31'::date)::timestamp with time zone) generate_series(1, 12, 1)))::text)	
3	-> Hash Join (cost=53377.70..59936.42 rows=62000 width=286)	
4	Hash Cond: (e_1.id = e.id)	
5	-> HashAggregate (cost=53360.72..58979.47 rows=62000 width=52)	
6	Group Key: e_1.id, EXTRACT(year FROM (generate_series('2023-01-01'::date)::timestamp with time zone, ('2024-12-31'::date)::timestamp with time zone) generate_series(1, 12, 1)))::text)	
7	-> Merge Join (cost=18848.47..24685.72 rows=310000 width=40)	
8	Merge Cond: (e_1.region_id = r.id)	
9	-> Index Scan using idx_embalses_region_id on embalses e_1 (cost=0.15..52.80 rows=310 width=8)	
10	-> Materialize (cost=18848.32..20952.15 rows=140000 width=16)	
11	-> Merge Left Join (cost=18848.32..20602.15 rows=140000 width=16)	
12	Merge Cond: ((r.id = ra.id_region) AND ((generate_series('2023-01-01'::date)::timestamp with time zone, ('2024-12-31'::date)::timestamp with time zone) generate_series(1, 12, 1)))::text)	
13	-> Sort (cost=16127.32..16477.32 rows=140000 width=12)	
14	Sort Key: r.id, (generate_series('2023-01-01'::date)::timestamp with time zone, ('2024-12-31'::date)::timestamp with time zone) generate_series(1, 12, 1)))::text)	
15	-> Nested Loop (cost=0.00..1766.77 rows=140000 width=12)	
16	-> ProjectSet (cost=0.00..5.02 rows=1000 width=8)	
17	-> Result (cost=0.00..0.01 rows=1 width=0)	
18	-> Materialize (cost=0.00..12.10 rows=140 width=4)	
19	-> Seq Scan on regiones r (cost=0.00..11.40 rows=140 width=4)	
20	-> Sort (cost=2721.00..2794.50 rows=29400 width=12)	
21	Sort Key: ra.id_region, ra.fecha	
22	-> Seq Scan on registro_aporte ra (cost=0.00..539.00 rows=29400 width=12)	
23	-> Hash (cost=13.10..13.10 rows=310 width=222)	
24	-> Seq Scan on embalses e (cost=0.00..13.10 rows=310 width=222)	

Plan de ejecución segunda consulta esquema final

	A2 QUERY PLAN
1	Sort (cost=710.82..711.39 rows=228 width=318)
2	Sort Key: resumen.embalse_nombre
3	-> Subquery Scan on resumen (cost=696.76..701.89 rows=228 width=318)
4	-> HashAggregate (cost=696.76..699.04 rows=228 width=286)
5	Group Key: e.id
6	-> Nested Loop (cost=0.16..695.05 rows=228 width=228)
7	-> Seq Scan on registro_aporte ra (cost=0.00..686.00 rows=147 width=10)
8	Filter: (EXTRACT(year FROM fecha) = '2024':numeric)
9	-> Memoize (cost=0.16..0.52 rows=2 width=226)
10	Cache Key: ra.id_region
11	Cache Mode: logical
12	-> Index Scan using idx_embalses_region_id on embalses e (cost=0.15..0.51 rows=2 width=226)
13	Index Cond: (region_id = ra.id_region)