

Tarea 3.2: Diseño físico:

- Optimizar el diseño físico de la base de datos, incluyendo la elección de tipos de almacenamiento, índices y particionamiento de tablas.
- Configurar los parámetros de PostgreSQL para mejorar el rendimiento.

Objetivo: Optimizar el diseño físico de la base de datos, incluyendo la elección de tipos de almacenamiento, índices y particionamiento de tablas. Configurar los parámetros de PostgreSQL para mejorar el rendimiento.

Acciones realizadas:

1. Elección de tipos de almacenamiento:

- Se seleccionaron tipos de datos adecuados para cada columna, como VARCHAR para textos, NUMERIC para valores decimales y DATE para fechas.
- Se utilizó SERIAL para las claves primarias autoincrementables.

2. Creación de índices:

- Se crearon índices en las columnas frecuentemente utilizadas en consultas, como:
 - Matricula en la tabla Vehiculo.
 - DNI_NIE en la tabla Conductor.
 - ID_Ruta en la tabla Ruta.
- Se justificó la creación de índices compuestos para consultas que involucran múltiples columnas.

3. Particionamiento de tablas:

- Se propuso el particionamiento de la tabla Ruta por fecha (FechaInicio) para mejorar el rendimiento en consultas que filtran por rangos de tiempo.
- Se consideró el particionamiento de la tabla Mantenimiento por TipoMantenimiento para facilitar consultas específicas.

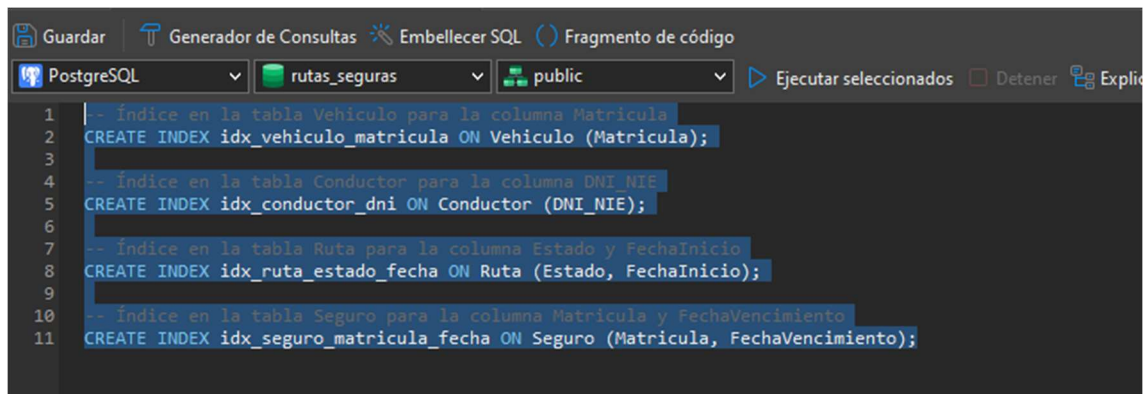
4. Configuración de PostgreSQL:

- Se ajustaron los siguientes parámetros en el archivo postgresql.conf para mejorar el rendimiento:
 - shared_buffers: Aumentado para mejorar la caché en memoria.
 - work_mem: Aumentado para optimizar operaciones de ordenación y agrupación.

- maintenance_work_mem: Aumentado para acelerar operaciones de mantenimiento como la creación de índices.

Scripts SQL para la creación de índices, particionamiento de tablas y configuración de parámetros de PostgreSQL.

1. Scripts para la creación de índices, los índices mejoran el rendimiento de las consultas al permitir búsquedas más rápidas en columnas frecuentemente utilizadas.



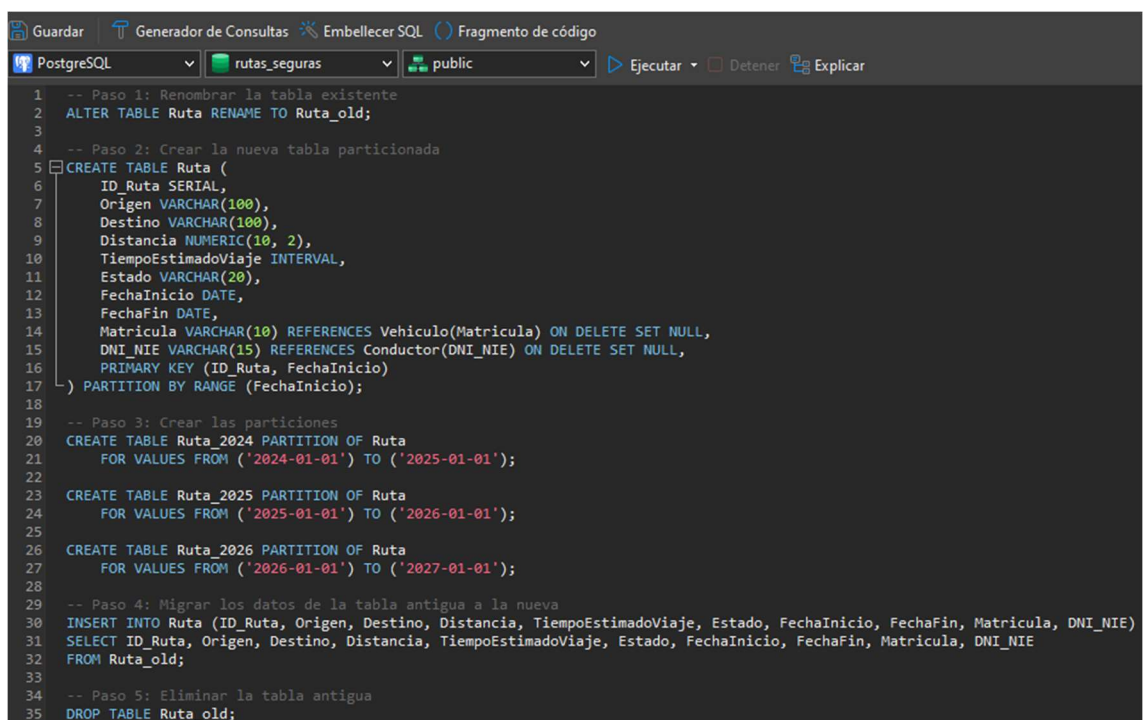
```

1  -- Índice en la tabla Vehiculo para la columna Matricula
2  CREATE INDEX idx_vehiculo_matricula ON Vehiculo (Matricula);
3
4  -- Índice en la tabla Conductor para la columna DNI_NIE
5  CREATE INDEX idx_conductor_dni ON Conductor (DNI_NIE);
6
7  -- Índice en la tabla Ruta para la columna Estado y FechaInicio
8  CREATE INDEX idx_ruta_estado_fecha ON Ruta (Estado, FechaInicio);
9
10 -- Índice en la tabla Seguro para la columna Matricula y FechaVencimiento
11 CREATE INDEX idx_seguro_matricula_fecha ON Seguro (Matricula, FechaVencimiento);

```

2. Scripts para el particionamiento de tablas, el particionamiento de tablas mejora el rendimiento al dividir una tabla grande en partes más pequeñas, lo que facilita la gestión y la consulta de datos.

- Particionamiento de la tabla Ruta (existente) por fecha (FechaInicio):



```

1  -- Paso 1: Renombrar la tabla existente
2  ALTER TABLE Ruta RENAME TO Ruta_old;
3
4  -- Paso 2: Crear la nueva tabla particionada
5  CREATE TABLE Ruta (
6      ID_Ruta SERIAL,
7      Origen VARCHAR(100),
8      Destino VARCHAR(100),
9      Distancia NUMERIC(10, 2),
10     TiempoEstimadoViaje INTERVAL,
11     Estado VARCHAR(20),
12     FechaInicio DATE,
13     FechaFin DATE,
14     Matricula VARCHAR(10) REFERENCES Vehiculo(Matricula) ON DELETE SET NULL,
15     DNI_NIE VARCHAR(15) REFERENCES Conductor(DNI_NIE) ON DELETE SET NULL,
16     PRIMARY KEY (ID_Ruta, FechaInicio)
17 ) PARTITION BY RANGE (FechaInicio);
18
19 -- Paso 3: Crear las particiones
20 CREATE TABLE Ruta_2024 PARTITION OF Ruta
21     FOR VALUES FROM ('2024-01-01') TO ('2025-01-01');
22
23 CREATE TABLE Ruta_2025 PARTITION OF Ruta
24     FOR VALUES FROM ('2025-01-01') TO ('2026-01-01');
25
26 CREATE TABLE Ruta_2026 PARTITION OF Ruta
27     FOR VALUES FROM ('2026-01-01') TO ('2027-01-01');
28
29 -- Paso 4: Migrar los datos de la tabla antigua a la nueva
30 INSERT INTO Ruta (ID_Ruta, Origen, Destino, Distancia, TiempoEstimadoViaje, Estado, FechaInicio, FechaFin, Matricula, DNI_NIE)
31 SELECT ID_Ruta, Origen, Destino, Distancia, TiempoEstimadoViaje, Estado, FechaInicio, FechaFin, Matricula, DNI_NIE
32 FROM Ruta_old;
33
34 -- Paso 5: Eliminar la tabla antigua
35 DROP TABLE Ruta_old;

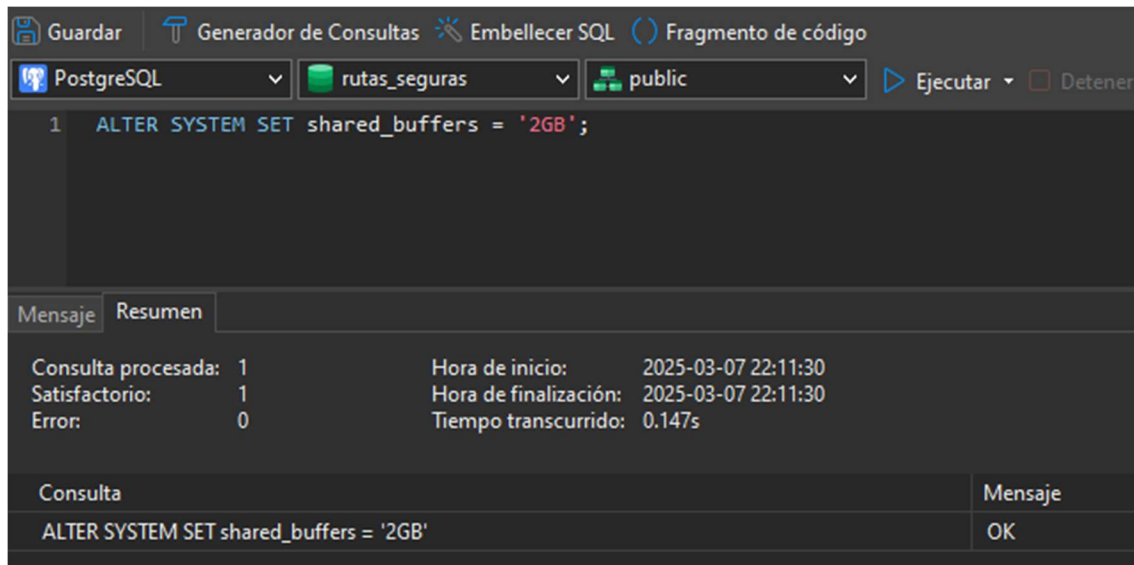
```

3. Scripts para la configuración de parámetros de PostgreSQL

La configuración de parámetros en PostgreSQL permite optimizar el rendimiento del servidor de base de datos.

Configuración de parámetros en postgresql.conf

- Aumentar el tamaño de `shared_buffers`, este parámetro controla la cantidad de memoria que PostgreSQL utiliza para almacenar datos en caché.

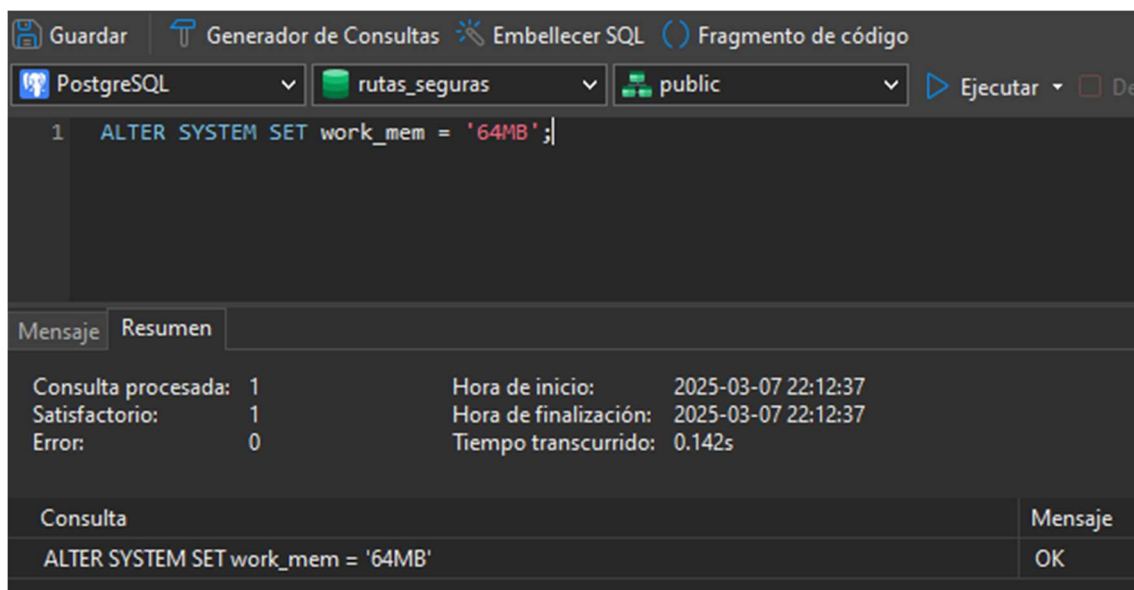


```
1 ALTER SYSTEM SET shared_buffers = '2GB';
```

Mensaje		Resumen
Consulta procesada:	1	Hora de inicio: 2025-03-07 22:11:30
Satisfactorio:	1	Hora de finalización: 2025-03-07 22:11:30
Error:	0	Tiempo transcurrido: 0.147s

Consulta	Mensaje
ALTER SYSTEM SET shared_buffers = '2GB'	OK

- Aumentar el tamaño de `work_mem`, este parámetro controla la cantidad de memoria que se asigna para operaciones de ordenación y agrupación.

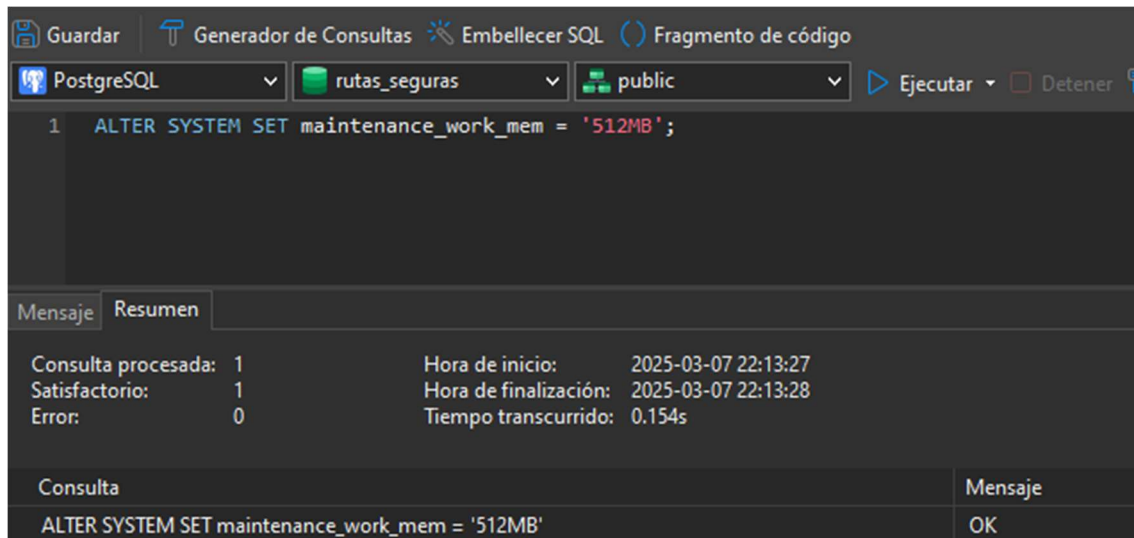


```
1 ALTER SYSTEM SET work_mem = '64MB';
```

Mensaje		Resumen
Consulta procesada:	1	Hora de inicio: 2025-03-07 22:12:37
Satisfactorio:	1	Hora de finalización: 2025-03-07 22:12:37
Error:	0	Tiempo transcurrido: 0.142s

Consulta	Mensaje
ALTER SYSTEM SET work_mem = '64MB'	OK

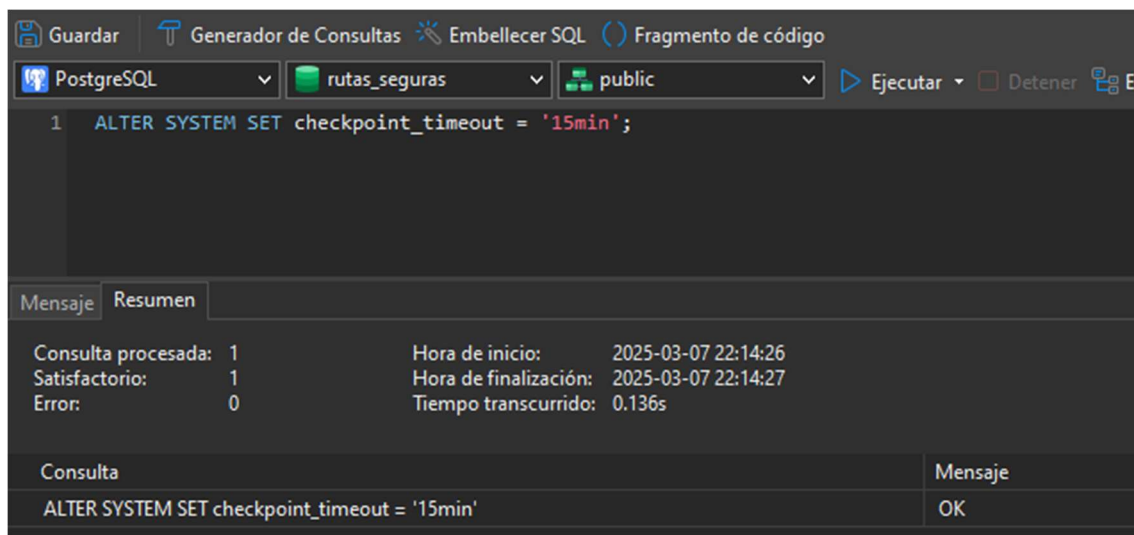
- Aumentar el tamaño de `maintenance_work_mem`, este parámetro controla la cantidad de memoria que se asigna para operaciones de mantenimiento, como la creación de índices.



```
1 ALTER SYSTEM SET maintenance_work_mem = '512MB';
```

Consulta	Mensaje
ALTER SYSTEM SET maintenance_work_mem = '512MB'	OK

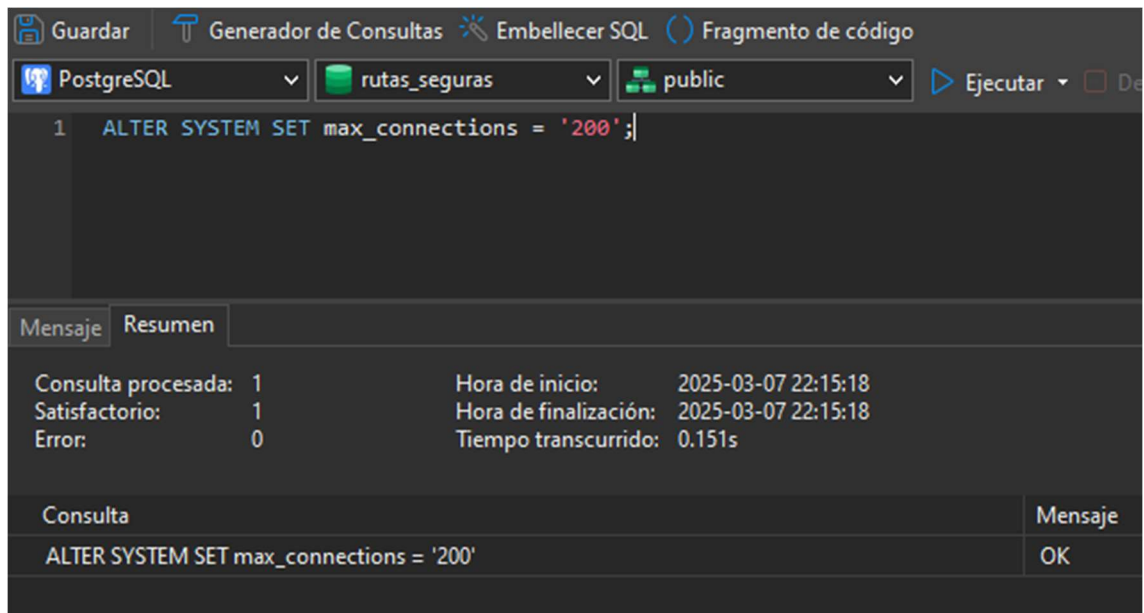
- Ajustar el parámetro `checkpoint_timeout`, este parámetro controla la frecuencia con la que PostgreSQL realiza puntos de control (checkpoints). Un valor más alto reduce la sobrecarga de E/S.



```
1 ALTER SYSTEM SET checkpoint_timeout = '15min';
```

Consulta	Mensaje
ALTER SYSTEM SET checkpoint_timeout = '15min'	OK

- Ajustar el parámetro `max_connections`, este parámetro controla el número máximo de conexiones simultáneas permitidas.



The screenshot shows a PostgreSQL query editor interface. The top bar includes buttons for 'Guardar', 'Generador de Consultas', 'Embellecer SQL', and 'Fragmento de código'. Below the bar, the database is set to 'PostgreSQL', the schema to 'rutas_seguras', and the user to 'public'. The query editor contains the following SQL command:

```
1 ALTER SYSTEM SET max_connections = '200';
```

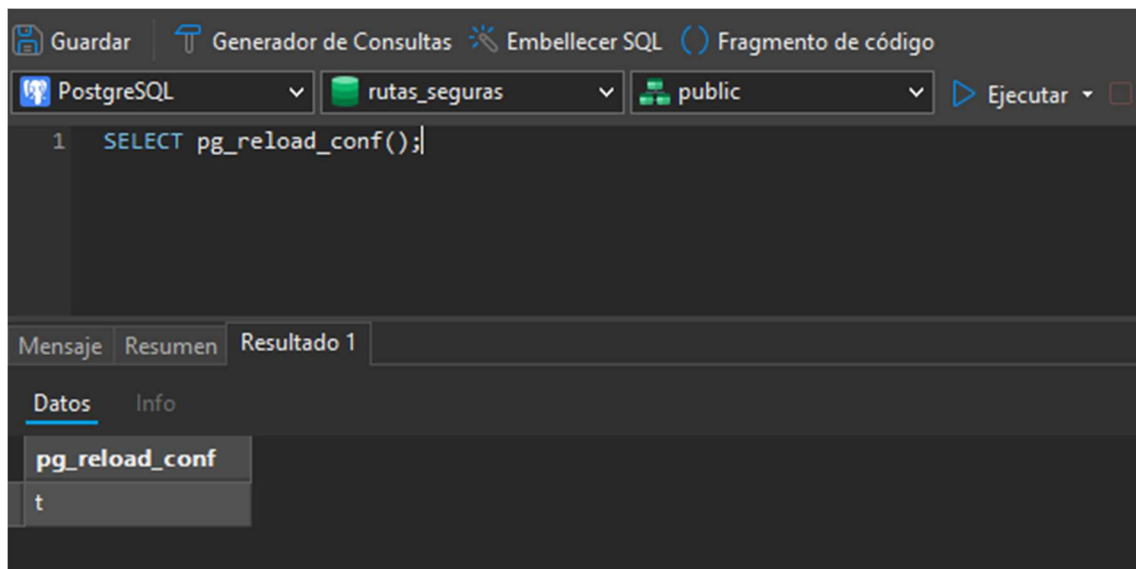
Below the query editor, the 'Mensaje' tab is active, displaying the execution summary:

Mensaje	
Consulta procesada:	1
Satisfactorio:	1
Error:	0
Hora de inicio:	2025-03-07 22:15:18
Hora de finalización:	2025-03-07 22:15:18
Tiempo transcurrido:	0.151s

Below the summary, the 'Consulta' tab is active, showing the executed command and its message:

Consulta	Mensaje
ALTER SYSTEM SET max_connections = '200'	OK

Recargar la configuración, después de realizar cambios en los parámetros, es necesario recargar la configuración para que surtan efecto.



The screenshot shows the same PostgreSQL query editor interface. The query editor contains the following SQL command:

```
1 SELECT pg_reload_conf();
```

Below the query editor, the 'Resultado 1' tab is active, displaying the results of the command. The 'Datos' sub-tab is selected, showing a single row of data:

pg_reload_conf
t