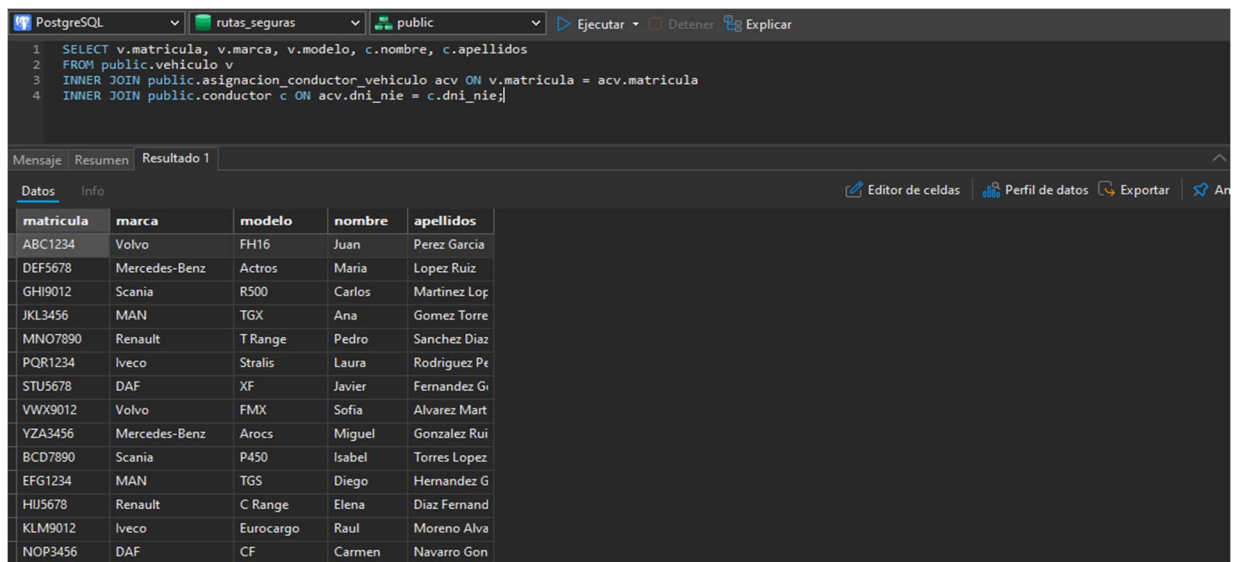


### Tarea 2.3: Consultas avanzadas:

- Realizar uniones (INNER JOIN, LEFT JOIN, RIGHT JOIN) para combinar datos de múltiples tablas.
- Utilizar subconsultas y expresiones correlacionadas para resolver consultas complejas.
- Crear vistas materializadas para mejorar el rendimiento de consultas repetitivas.

I. Uniones (JOIN), estas consultas combinan datos de múltiples tablas para obtener resultados más completos.

a) INNER JOIN : Obtener los vehículos asignados a conductores junto con sus nombres.



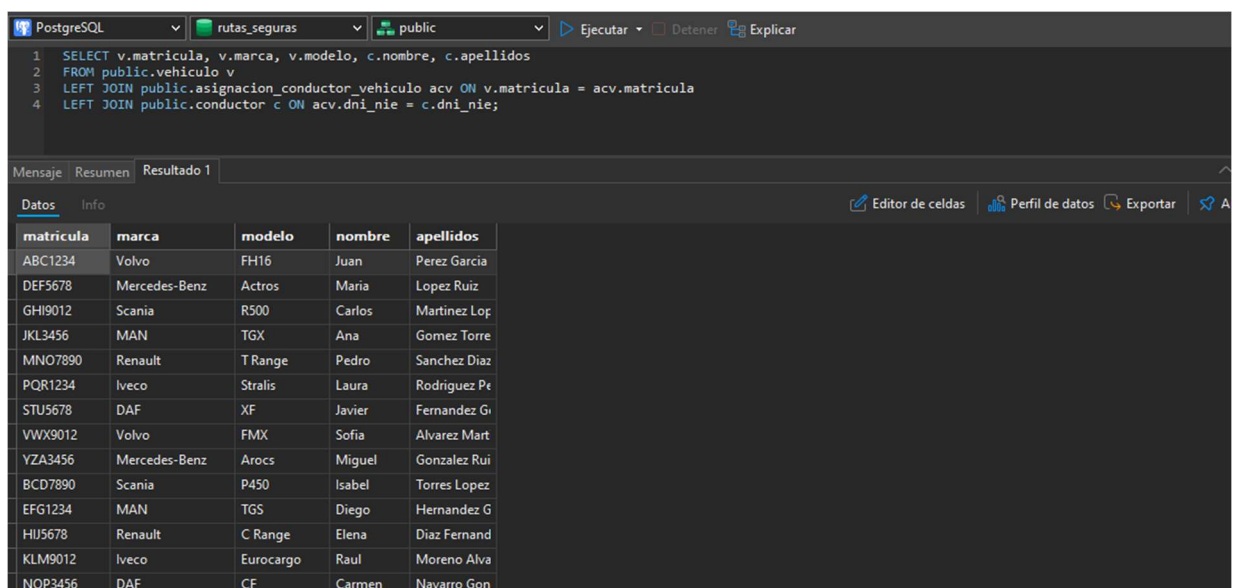
The screenshot shows a PostgreSQL query window with the following SQL code:

```
1 SELECT v.matricula, v.marca, v.modelo, c.nombre, c.apellidos
2 FROM public.vehiculo v
3 INNER JOIN public.asignacion_conductor_vehiculo acv ON v.matricula = acv.matricula
4 INNER JOIN public.conductor c ON acv.dni_nie = c.dni_nie;
```

The query results are displayed in a table with the following columns: **matricula**, **marca**, **modelo**, **nombre**, and **apellidos**. The results show 15 rows of data, representing vehicles assigned to drivers.

matricula	marca	modelo	nombre	apellidos
ABC1234	Volvo	FH16	Juan	Perez Garcia
DEF5678	Mercedes-Benz	Actros	Maria	Lopez Ruiz
GHI9012	Scania	R500	Carlos	Martinez Lopez
JKL3456	MAN	TGX	Ana	Gomez Torre
MNO7890	Renault	T Range	Pedro	Sanchez Diaz
PQR1234	Iveco	Stralis	Laura	Rodriguez Perez
STU5678	DAF	XF	Javier	Fernandez Garcia
VWX9012	Volvo	FMX	Sofia	Alvarez Martin
YZA3456	Mercedes-Benz	Arocs	Miguel	Gonzalez Ruiz
BCD7890	Scania	P450	Isabel	Torres Lopez
EFG1234	MAN	TGS	Diego	Hernandez Garcia
HIJ5678	Renault	C Range	Elena	Diaz Fernandez
KLM9012	Iveco	Eurocargo	Raul	Moreno Alva
NOP3456	DAF	CF	Carmen	Navarro Gonzalez

b) LEFT JOIN : Mostrar todos los vehículos y los conductores asignados (si existen).



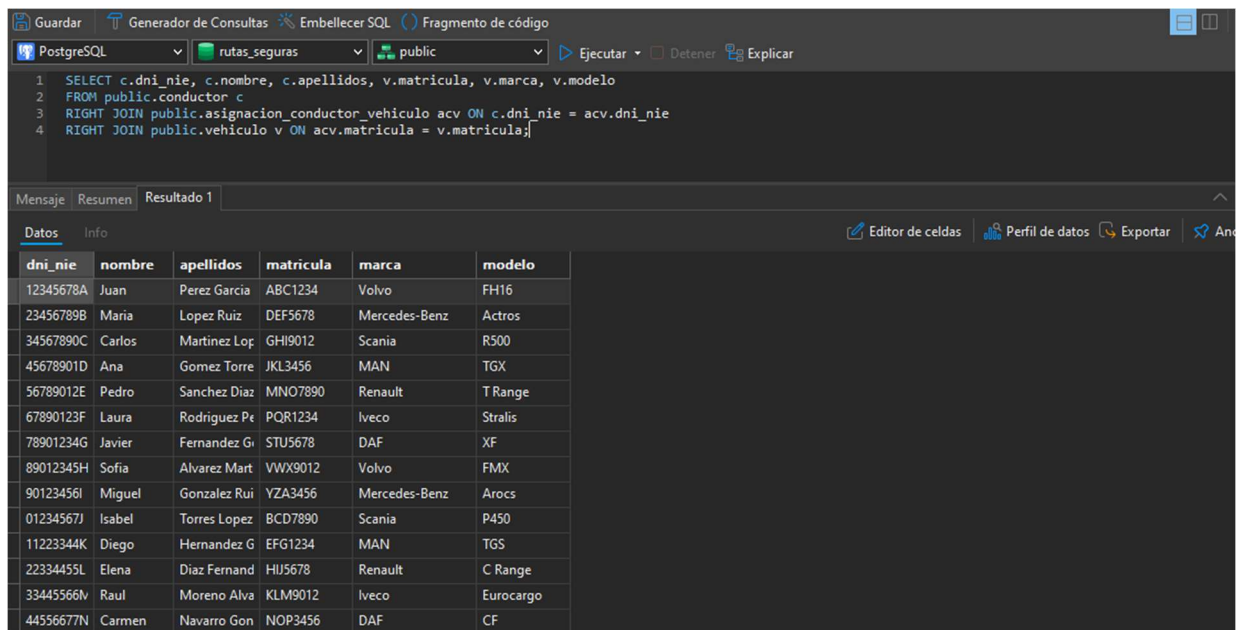
The screenshot shows a PostgreSQL query window with the following SQL code:

```
1 SELECT v.matricula, v.marca, v.modelo, c.nombre, c.apellidos
2 FROM public.vehiculo v
3 LEFT JOIN public.asignacion_conductor_vehiculo acv ON v.matricula = acv.matricula
4 LEFT JOIN public.conductor c ON acv.dni_nie = c.dni_nie;
```

The query results are displayed in a table with the following columns: **matricula**, **marca**, **modelo**, **nombre**, and **apellidos**. The results show 15 rows of data, representing vehicles and their assigned drivers. This query includes all vehicles from the previous query, as well as any additional vehicles that might exist in the `public.vehiculo` table but are not assigned to a driver.

matricula	marca	modelo	nombre	apellidos
ABC1234	Volvo	FH16	Juan	Perez Garcia
DEF5678	Mercedes-Benz	Actros	Maria	Lopez Ruiz
GHI9012	Scania	R500	Carlos	Martinez Lopez
JKL3456	MAN	TGX	Ana	Gomez Torre
MNO7890	Renault	T Range	Pedro	Sanchez Diaz
PQR1234	Iveco	Stralis	Laura	Rodriguez Perez
STU5678	DAF	XF	Javier	Fernandez Garcia
VWX9012	Volvo	FMX	Sofia	Alvarez Martin
YZA3456	Mercedes-Benz	Arocs	Miguel	Gonzalez Ruiz
BCD7890	Scania	P450	Isabel	Torres Lopez
EFG1234	MAN	TGS	Diego	Hernandez Garcia
HIJ5678	Renault	C Range	Elena	Diaz Fernandez
KLM9012	Iveco	Eurocargo	Raul	Moreno Alva
NOP3456	DAF	CF	Carmen	Navarro Gonzalez

c) RIGHT JOIN : Mostrar todos los conductores y los vehículos asignados (si existen).



The screenshot shows a PostgreSQL query editor with the following SQL query:

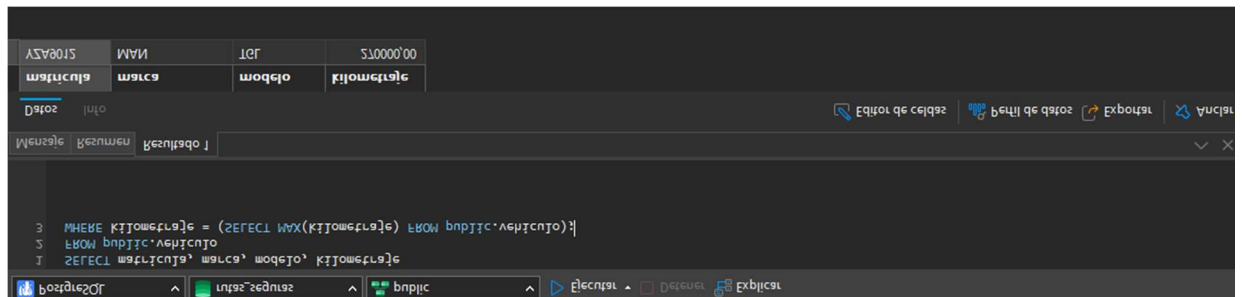
```
1 SELECT c.dni_nie, c.nombre, c.apellidos, v.matricula, v.marca, v.modelo
2 FROM public.conductor c
3 RIGHT JOIN public.asignacion_conductor_vehiculo acv ON c.dni_nie = acv.dni_nie
4 RIGHT JOIN public.vehiculo v ON acv.matricula = v.matricula;
```

The results are displayed in a table with the following columns: dni\_nie, nombre, apellidos, matricula, marca, modelo.

dni_nie	nombre	apellidos	matricula	marca	modelo
12345678A	Juan	Perez Garcia	ABC1234	Volvo	FH16
23456789B	Maria	Lopez Ruiz	DEF5678	Mercedes-Benz	Actros
34567890C	Carlos	Martinez Lopez	GHI9012	Scania	R500
45678901D	Ana	Gomez Torre	JKL3456	MAN	TGX
56789012E	Pedro	Sanchez Diaz	MNO7890	Renault	T Range
67890123F	Laura	Rodriguez Perez	PQR1234	Iveco	Stralis
78901234G	Javier	Fernandez Garcia	STU5678	DAF	XF
89012345H	Sofia	Alvarez Martinez	VWX9012	Volvo	FMX
90123456I	Miguel	Gonzalez Ruiz	YZA3456	Mercedes-Benz	Arocs
01234567J	Isabel	Torres Lopez	BCD7890	Scania	P450
11223344K	Diego	Hernandez Garcia	EFG1234	MAN	TGS
22334455L	Elena	Diaz Fernandez	HIJ5678	Renault	C Range
33445566M	Raul	Moreno Alva	KLM9012	Iveco	Eurocargo
44556677N	Carmen	Navarro Gonzalez	NOP3456	DAF	CF

II. Subconsultas, estas consultas utilizan subconsultas para resolver problemas complejos.

a) Obtener los vehículos con el kilometraje más alto:



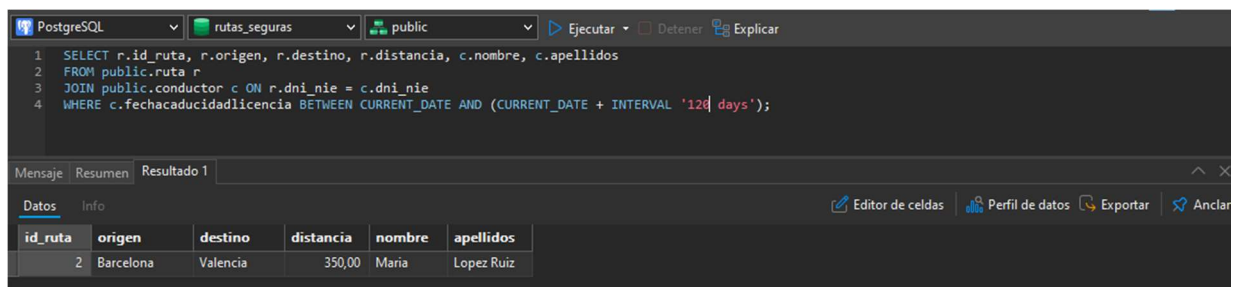
The screenshot shows a PostgreSQL query editor with the following SQL query:

```
1 WHERE kilometraje = (SELECT MAX(kilometraje) FROM public.vehiculo);
2 FROM public.vehiculo v
3 WHERE kilometraje = (SELECT MAX(kilometraje) FROM public.vehiculo);
```

The results are displayed in a table with the following columns: kilometraje, marca, modelo, matricula.

kilometraje	marca	modelo	matricula
500000	MAN	TGX	JKL3456

b) Listar las rutas realizadas por conductores cuya licencia caduque en los próximos 120 días:



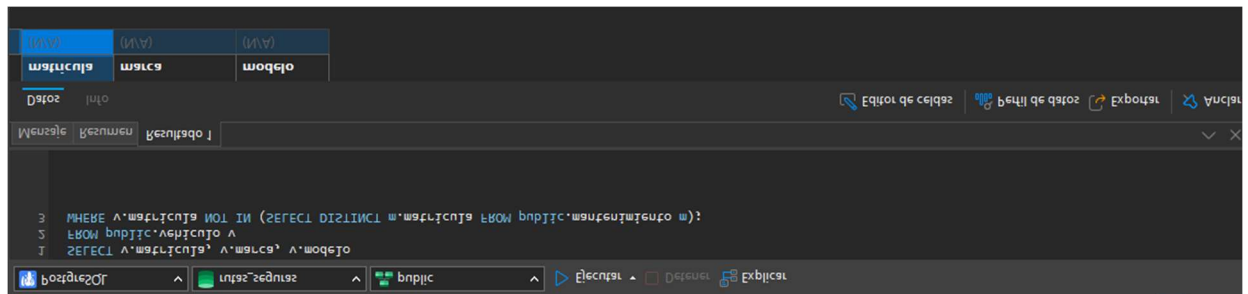
The screenshot shows a PostgreSQL query editor with the following SQL query:

```
1 SELECT r.id_ruta, r.origen, r.destino, r.distancia, c.nombre, c.apellidos
2 FROM public.ruta r
3 JOIN public.conductor c ON r.dni_nie = c.dni_nie
4 WHERE c.fecha caducidad licencia BETWEEN CURRENT_DATE AND (CURRENT_DATE + INTERVAL '120 days');
```

The results are displayed in a table with the following columns: id\_ruta, origen, destino, distancia, nombre, apellidos.

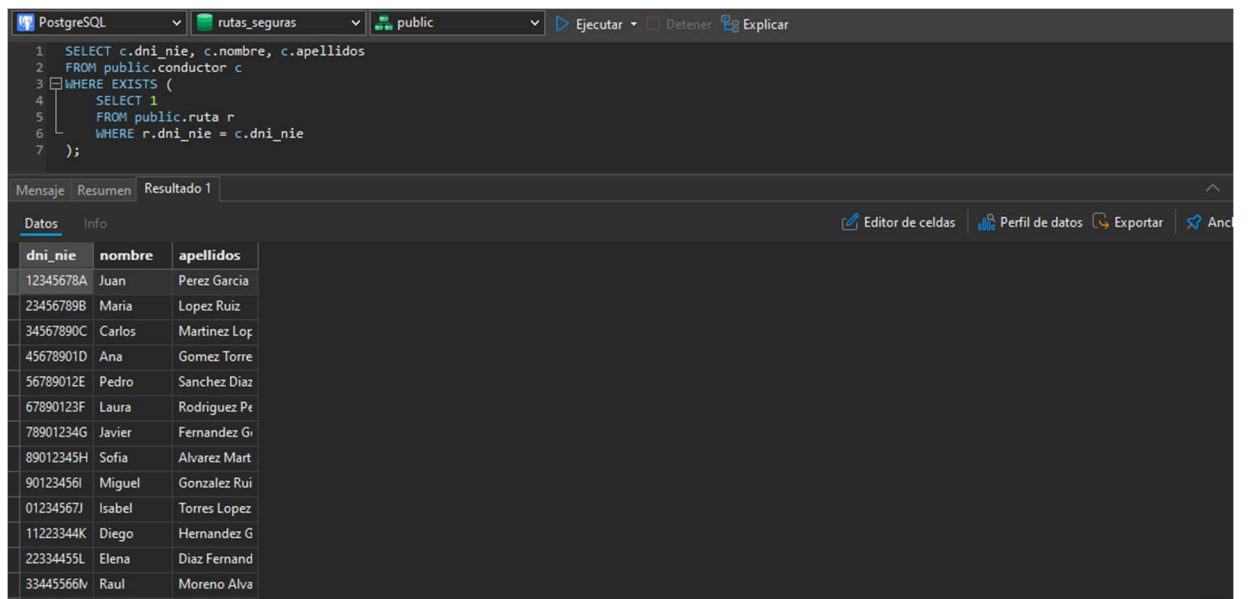
id_ruta	origen	destino	distancia	nombre	apellidos
2	Barcelona	Valencia	350,00	Maria	Lopez Ruiz

c) Obtener los vehículos que no han tenido mantenimiento:



III. Expresiones Correlacionadas, estas consultas usan expresiones correlacionadas para evaluar condiciones fila por fila.

a) Obtener los conductores que tienen al menos una ruta asignada:



b) Mostrar los vehículos cuyo consumo de combustible es mayor al promedio:

```
1 SELECT v.matricula, v.marca, v.modelo, v.consumocombustible
2 FROM public.vehiculo v
3 WHERE v.consumocombustible > (
4     SELECT AVG(consumocombustible)
5     FROM public.vehiculo
6 );
```

Mensaje Resumen Resultado 1

matricula	marca	modelo	consumocombustible
ABC1234	Volvo	FH16	30,50
GHI9012	Scania	R500	32,00
JKL3456	MAN	TGX	29,50
MNO7890	Renault	T Range	31,25
STU5678	DAF	XF	30,00
YZA3456	Mercedes-Benz	Arocs	31,50
EFG1234	MAN	TGS	30,75
KLM9012	Iveco	Eurocargo	29,75
NOP3456	DAF	CF	30,25
QRS7890	Volvo	FE	31,00
VWX5678	Scania	G410	30,50
EFG7890	Iveco	Daily	30,00

4. Vistas Materializadas, las vistas materializadas son útiles para mejorar el rendimiento de consultas repetitivas al almacenar los resultados físicamente.

a) Crear una vista materializada para mostrar el rendimiento de los vehículos operativos:

```
1 CREATE MATERIALIZED VIEW public.vista_materializada_rendimiento_vehiculos AS
2 SELECT v.matricula, v.marca, v.modelo, v.kilometraje, v.consumocombustible, v.estado
3 FROM public.vehiculo v
4 WHERE v.estado = 'Operativo';
```

Mensaje Resumen

Consulta procesada:	1	Hora de inicio:	2025-03-04 23:09:57
Satisfactorio:	1	Hora de finalización:	2025-03-04 23:09:58
Error:	0	Tiempo transcurrido:	0.226s

☐ Mostrar sólo los errores

Consulta	Mensaje	Consulta de h...	Tiempo de rec...
CREATE MATERIALIZED VIEW public.vista_materializada_rendimiento_vehiculos AS SELECT v.mat...	OK	0,200s	0,000s

b) Consultar la vista materializada:

PostgreSQL rutas\_seguras public Ejecutar Detener Explicar

1 SELECT \* FROM public.vista\_materializada\_rendimiento\_vehiculos;

Mensaje Resumen Resultado 1

Datos Info Editor de celdas Perfil de datos Exportar

matricula	marca	modelo	kilometraje	consumocombustible	estado
ABC1234	Volvo	FH16	150000,00	30,50	Operativo
DEF5678	Mercedes-Benz	Actros	120000,00	28,75	Operativo
GHI9012	Scania	R500	80000,00	32,00	Operativo
MNO7890	Renault	T Range	50000,00	31,25	Operativo
STU5678	DAF	XF	100000,00	30,00	Operativo
VWX9012	Volvo	FMX	130000,00	29,00	Operativo
YZA3456	Mercedes-Benz	Arocs	70000,00	31,50	Operativo
EFG1234	MAN	TGS	60000,00	30,75	Operativo
KLM9012	Iveco	Eurocargo	110000,00	29,75	Operativo
NOP3456	DAF	CF	140000,00	30,25	Operativo
QRS7890	Volvo	FE	90000,00	31,00	Operativo
VWX5678	Scania	G410	75000,00	30,50	Operativo
BCD3456	Renault	K Range	120000,00	29,25	Operativo
EFG7890	Iveco	Daily	150000,00	30,00	Operativo

c) Refrescar la vista materializada cuando los datos cambien:

PostgreSQL rutas\_seguras public Ejecutar Detener Explicar

1 REFRESH MATERIALIZED VIEW public.vista\_materializada\_rendimiento\_vehiculos;

Mensaje Resumen

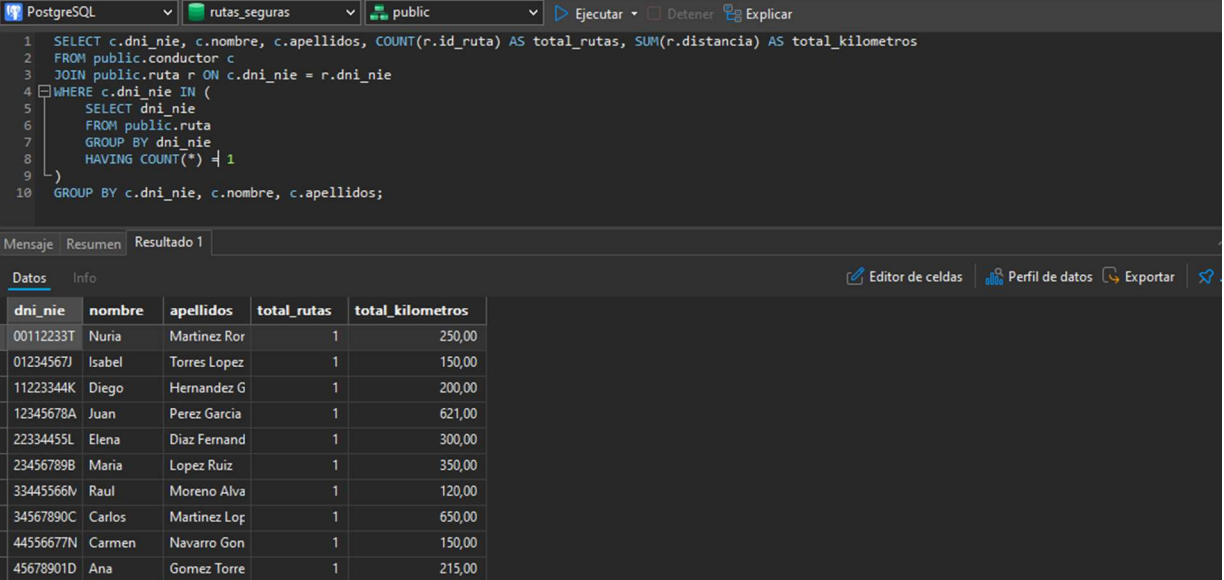
Consulta procesada: 1 Hora de inicio: 2025-03-04 23:11:51  
Satisfactorio: 1 Hora de finalizaci3n: 2025-03-04 23:11:51  
Error: 0 Tiempo transcurrido: 0.039s

Mostrar s3lo los errores

Consulta	Mensaje	Consulta de h...	Tiempo de rec...
REFRESH MATERIALIZED VIEW public.vista_materializada_rendimiento_vehiculos	OK	0,011s	0,000s

5. Ejemplo Combinado: Unión Compleja con Subconsulta, esta consulta combina varias técnicas para obtener un resultado detallado.

a) Obtener los conductores que han realizado una 1 rutas y mostrar el total de kilómetros recorridos:



The screenshot shows a PostgreSQL query editor with the following SQL query:

```
1 SELECT c.dni_nie, c.nombre, c.apellidos, COUNT(r.id_ruta) AS total_rutas, SUM(r.distancia) AS total_kilometros
2 FROM public.conductor c
3 JOIN public.ruta r ON c.dni_nie = r.dni_nie
4 WHERE c.dni_nie IN (
5     SELECT dni_nie
6     FROM public.ruta
7     GROUP BY dni_nie
8     HAVING COUNT(*) = 1
9 )
10 GROUP BY c.dni_nie, c.nombre, c.apellidos;
```

The results are displayed in a table with the following columns: dni\_nie, nombre, apellidos, total\_rutas, and total\_kilometros. The table contains 10 rows of data.

dni_nie	nombre	apellidos	total_rutas	total_kilometros
00112233T	Nuria	Martinez Ror	1	250,00
01234567J	Isabel	Torres Lopez	1	150,00
11223344K	Diego	Hernandez G	1	200,00
12345678A	Juan	Perez Garcia	1	621,00
22334455L	Elena	Diaz Fernand	1	300,00
23456789B	Maria	Lopez Ruiz	1	350,00
33445566N	Raul	Moreno Alva	1	120,00
34567890C	Carlos	Martinez Lop	1	650,00
44556677N	Carmen	Navarro Gon	1	150,00
45678901D	Ana	Gomez Torre	1	215,00