Evidencias Peticiones Postman

Luis Alejandro Alarcón Chaves

ID 792825

ingeniería en sistemas, corporación universitaria el minuto de Dios

Bases de Datos Masivas

NRC-10-60747

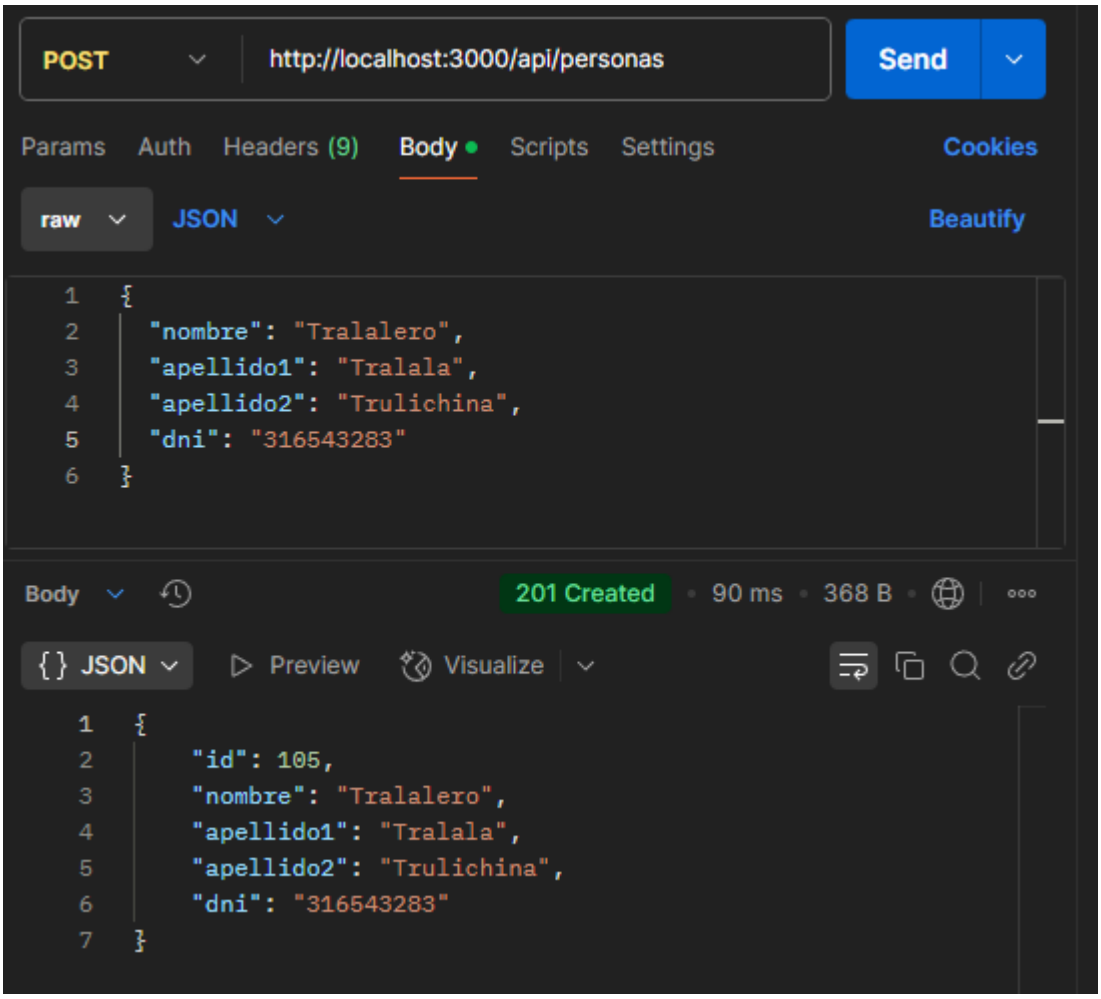
Prof. William Alexander Matallana Porras

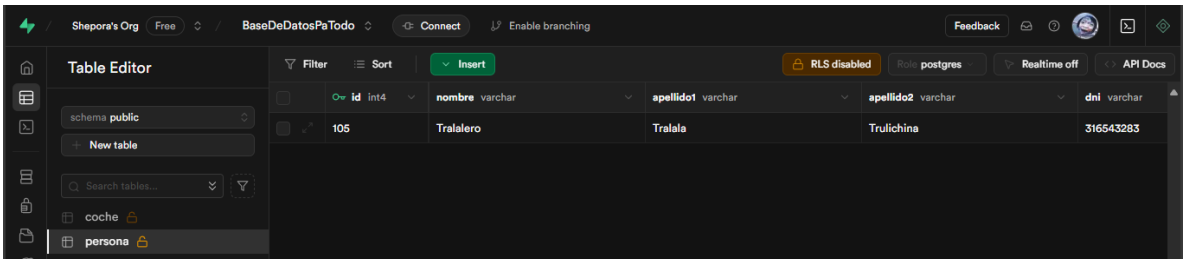
Abril 12, 2025.

- **Endpoints para Personas**

Agregar persona.

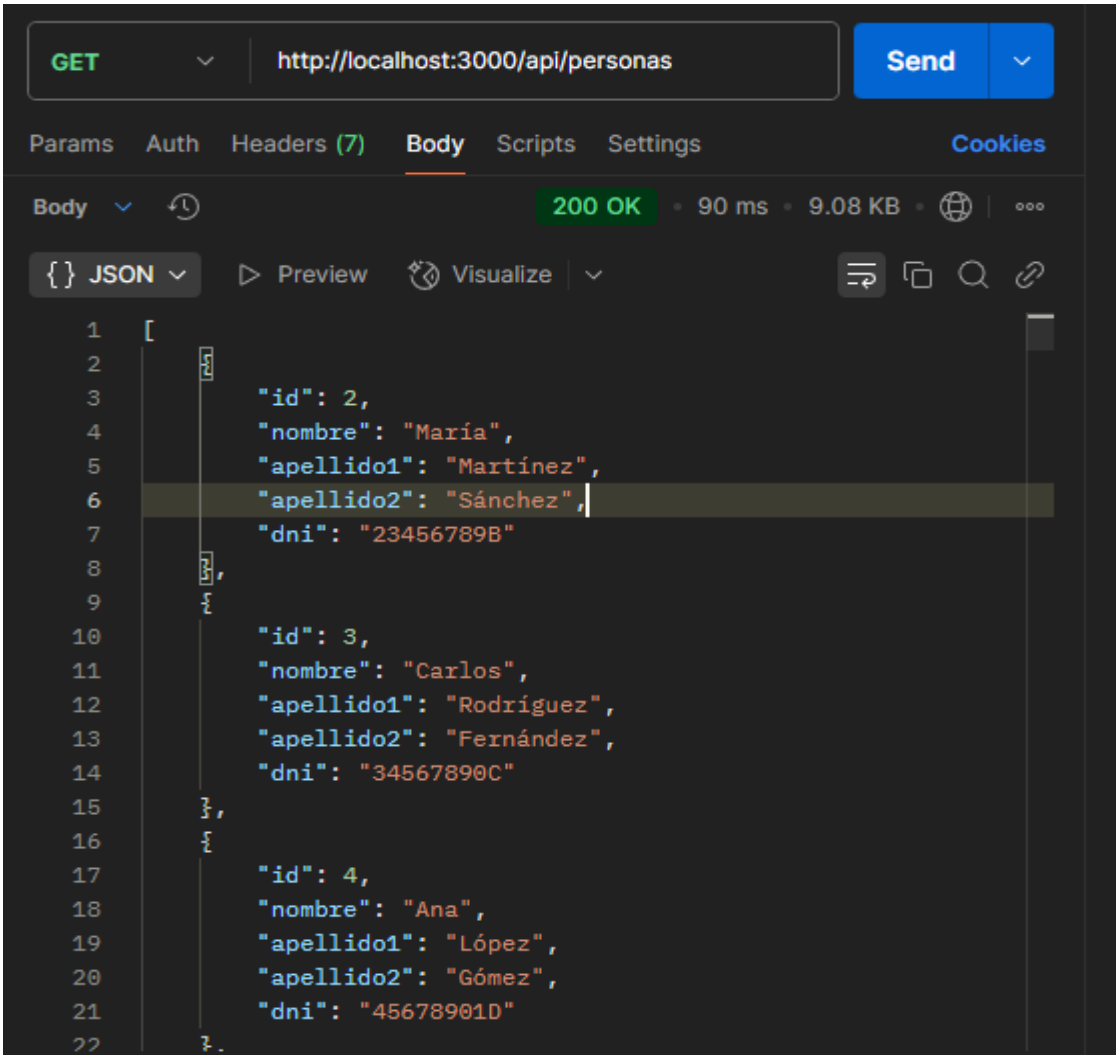
```
// Endpoints para Personas
app.post('/api/personas', async (req, res) => {
  const { nombre, apellido1, apellido2, dni } = req.body;
  try {
    const result = await client.query(
      'INSERT INTO Persona (nombre, apellido1, apellido2, dni) VALUES ($1, $2, $3, $4) RETURNING *
      [nombre, apellido1, apellido2, dni]
    );
    res.status(201).json(result.rows[0]);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});
```





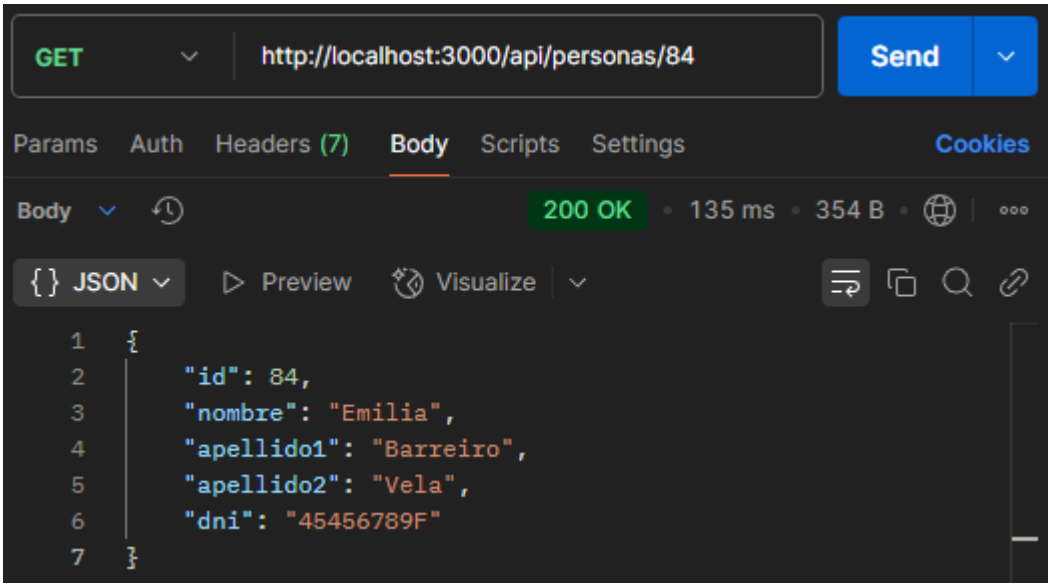
Mostrar personas

```
app.get('/api/personas', async (req, res) => {
  try {
    const result = await client.query('SELECT * FROM Persona');
    res.status(200).json(result.rows);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});
```



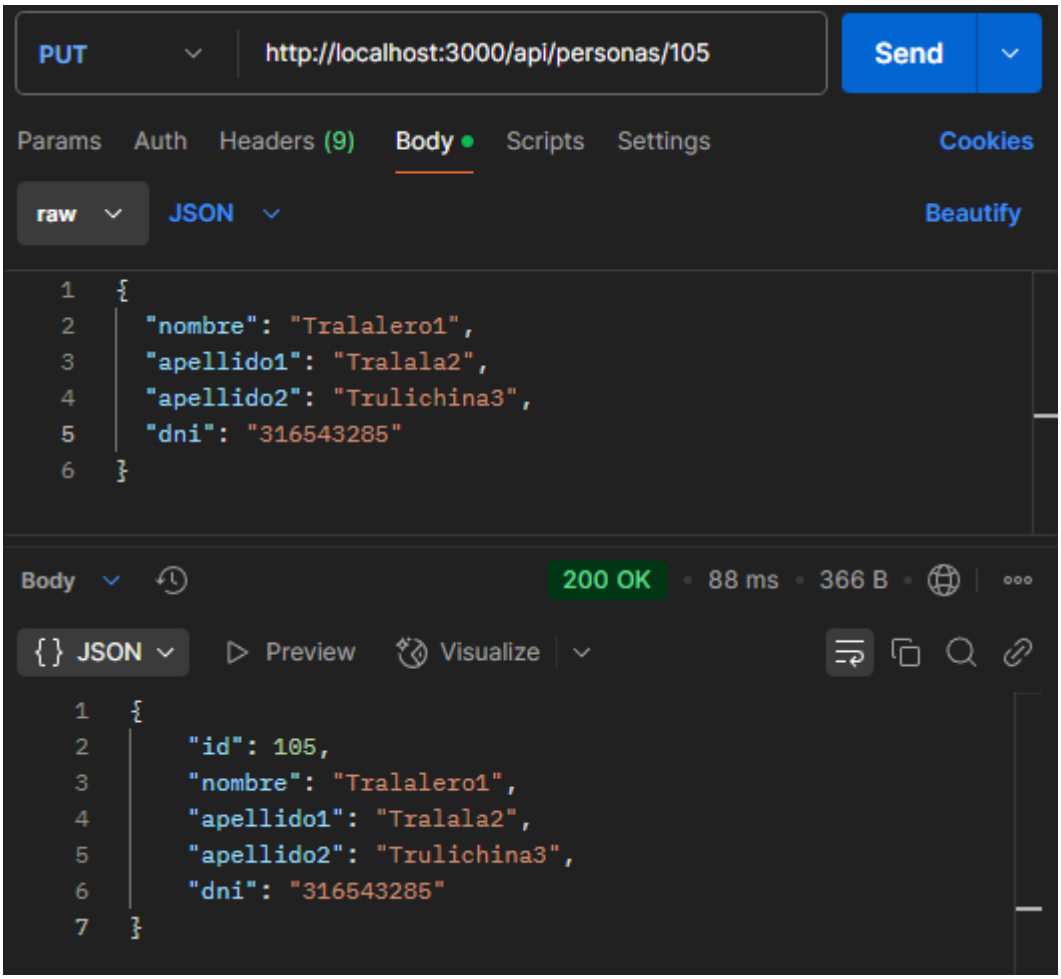
Mostrar persona por ID

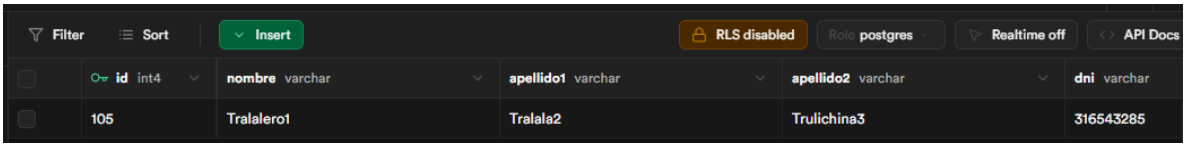
```
app.get('/api/personas/:id', async (req, res) => {
  const { id } = req.params;
  try {
    const result = await client.query('SELECT * FROM Persona WHERE id = $1', [id]);
    if (result.rows.length === 0) {
      return res.status(404).json({ error: 'Persona no encontrada' });
    }
    res.status(200).json(result.rows[0]);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});
```



Actualizar persona por ID

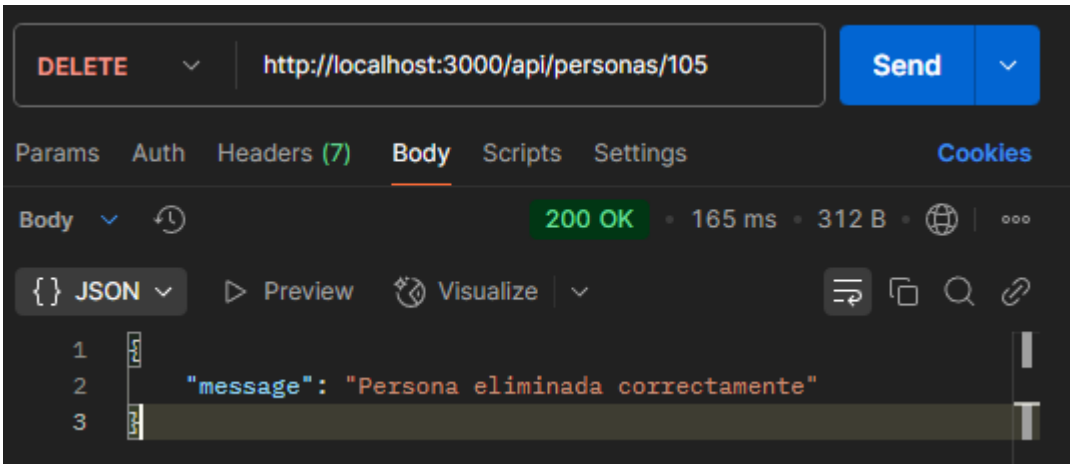
```
app.put('/api/personas/:id', async (req, res) => {
  const { id } = req.params;
  const { nombre, apellido1, apellido2, dni } = req.body;

  try {
    const result = await client.query(
      'UPDATE Persona SET nombre = $1, apellido1 = $2, apellido2 = $3, dni = $4 WHERE id = $5 RETURNING *',
      [nombre, apellido1, apellido2, dni, id]
    );
    if (result.rows.length === 0) {
      return res.status(404).json({ error: 'Persona no encontrada' });
    }
    res.status(200).json(result.rows[0]);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});
```





Eliminar persona por ID

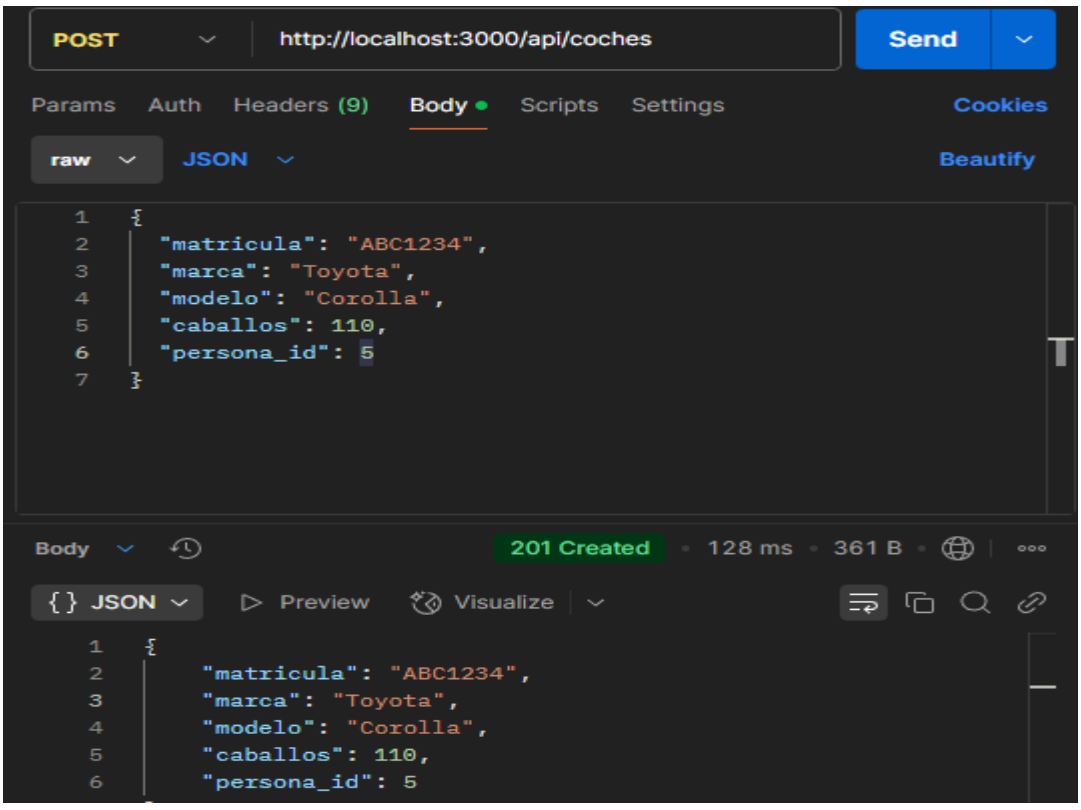
```
app.delete('/api/personas/:id', async (req, res) => {
  const { id } = req.params;

  try {
    // Primero eliminamos los coches relacionados
    await client.query('DELETE FROM Coche WHERE persona_id = $1', [id]);

    // Luego eliminamos la persona
    const result = await client.query('DELETE FROM Persona WHERE id = $1 RETURNING *', [id]);

    if (result.rows.length === 0) {
      return res.status(404).json({ error: 'Persona no encontrada' });
    }
    res.status(200).json({ message: 'Persona eliminada correctamente' });
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});
```

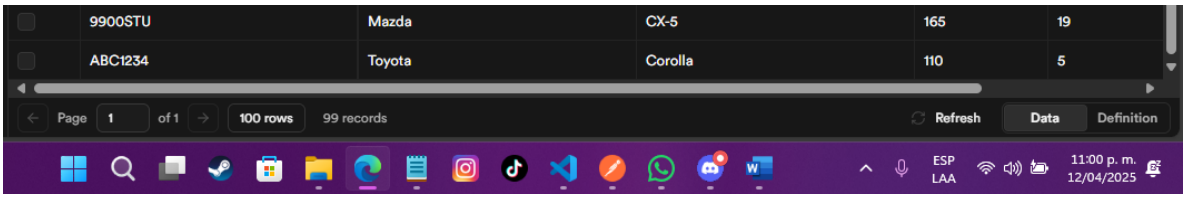


- **Endpoints para coches.**

Agregar coche.

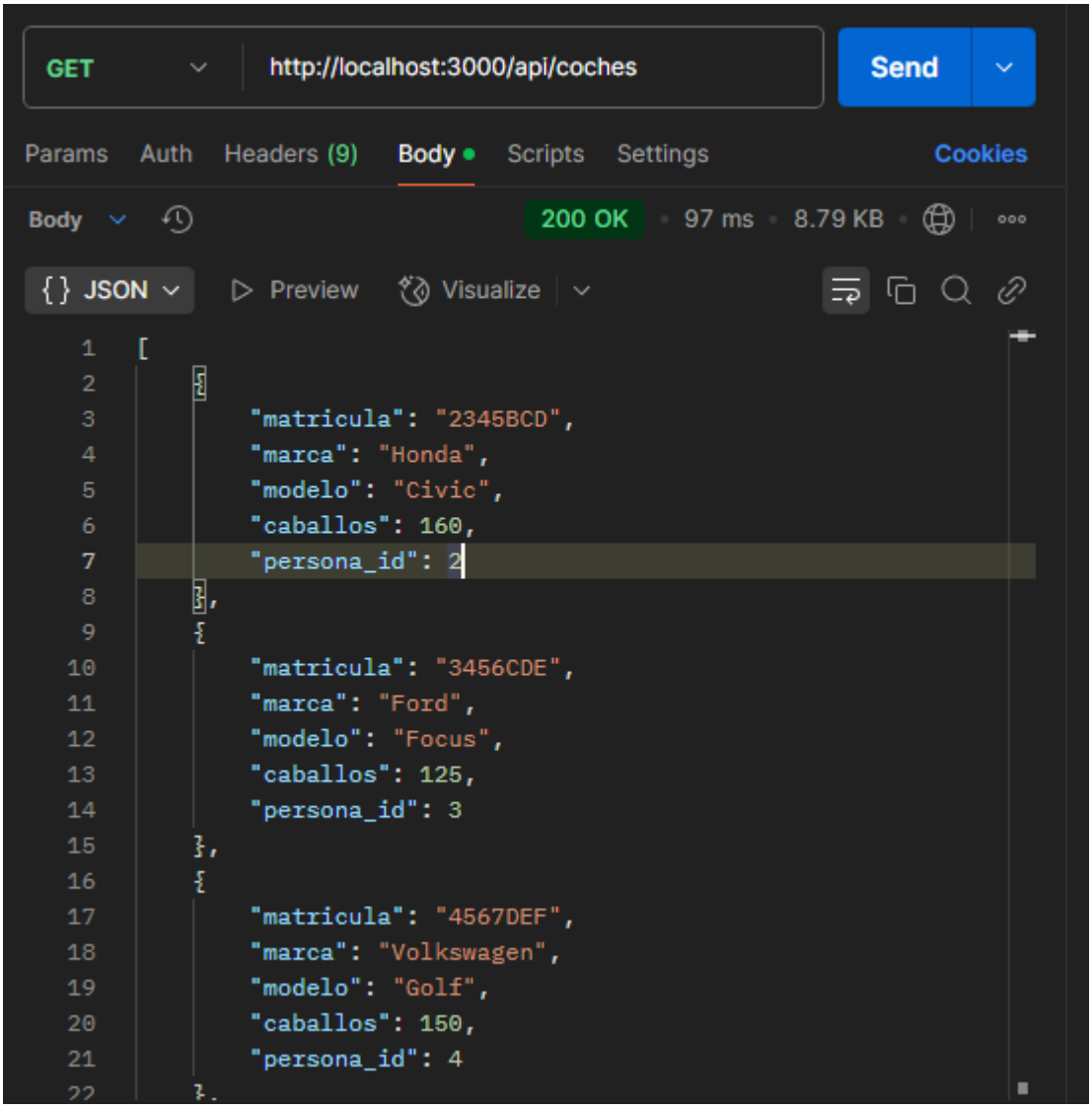
```
app.post('/api/coches', async (req, res) => {
  const { matricula, marca, modelo, caballos, persona_id } = req.body;
  try {
    const result = await client.query(
      'INSERT INTO Coche (matricula, marca, modelo, caballos, persona_id) VALUES ($1, $2, $3, $4, $5) RETURNING *',
      [matricula, marca, modelo, caballos, persona_id]
    );
    res.status(201).json(result.rows[0]);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});
```





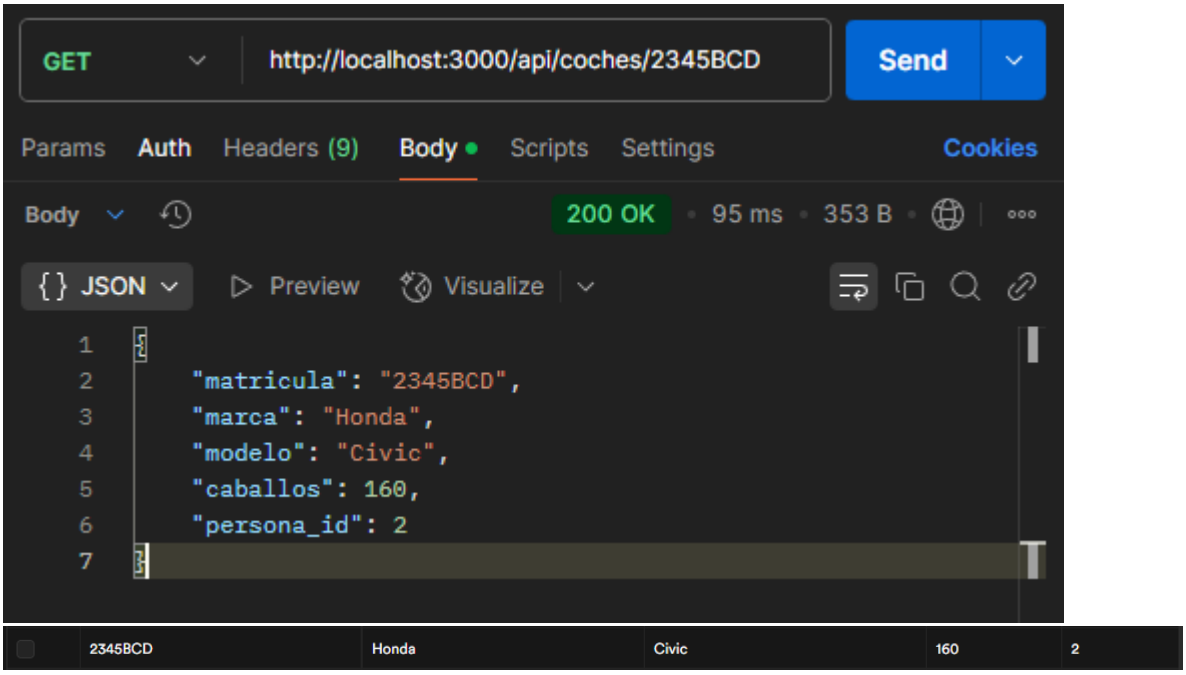
Ver todos los coches.

```
app.get('/api/coches', async (req, res) => {
  try {
    const result = await client.query('SELECT * FROM Coche');
    res.status(200).json(result.rows);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});
```



Ver coches por matricula.

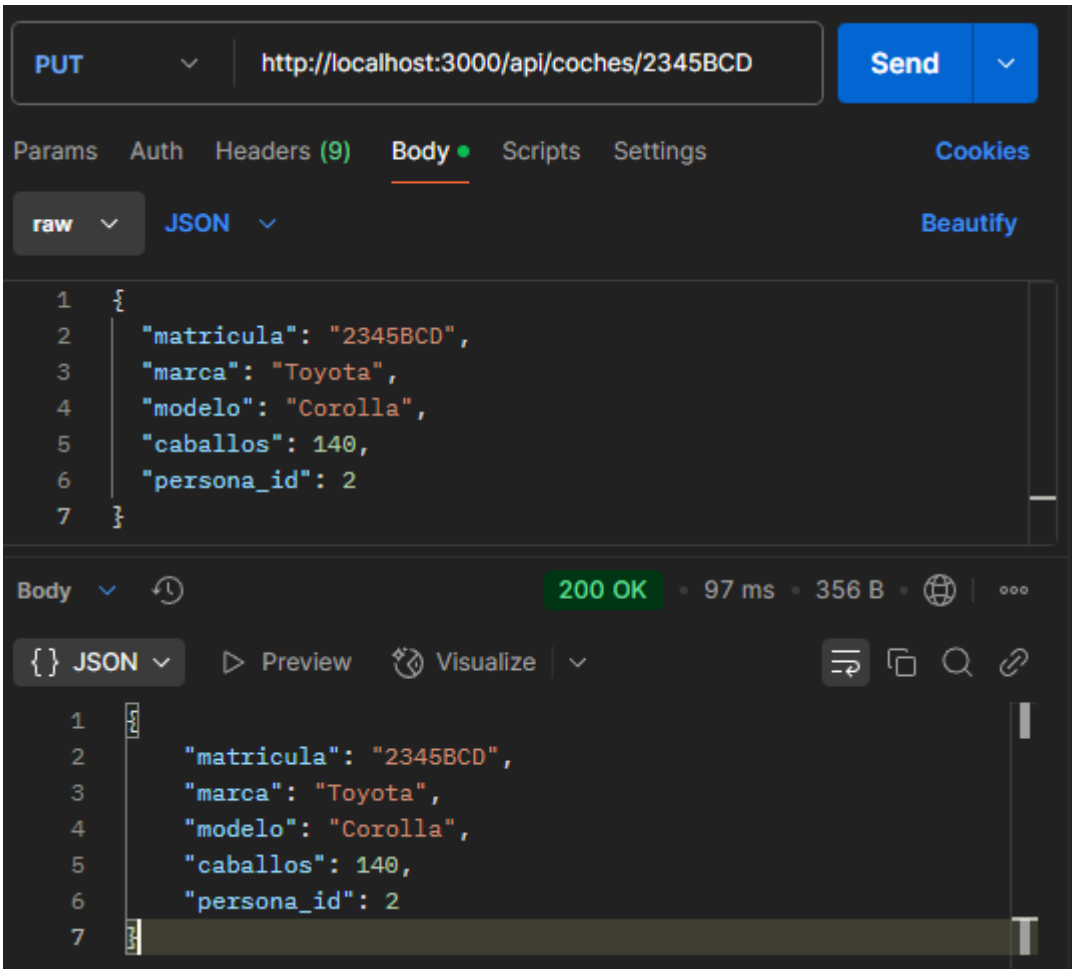
```
app.get('/api/coches/:matricula', async (req, res) => {
  const { matricula } = req.params;
  try {
    const result = await client.query('SELECT * FROM Coche WHERE matricula = $1', [matricula]);
    if (result.rows.length === 0) {
      return res.status(404).json({ error: 'Coche no encontrado' });
    }
    res.status(200).json(result.rows[0]);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});
```



	2345BCD	Honda	Civic	160	2

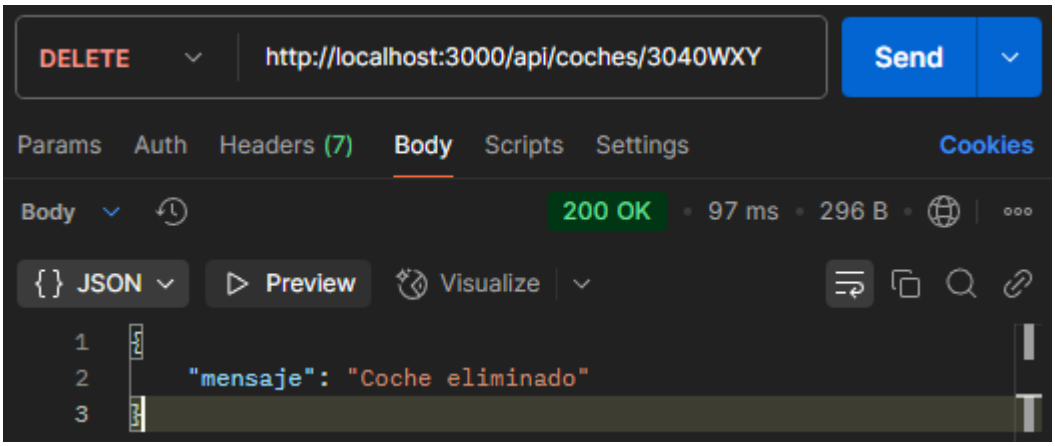
Actualizar coches por matricula.

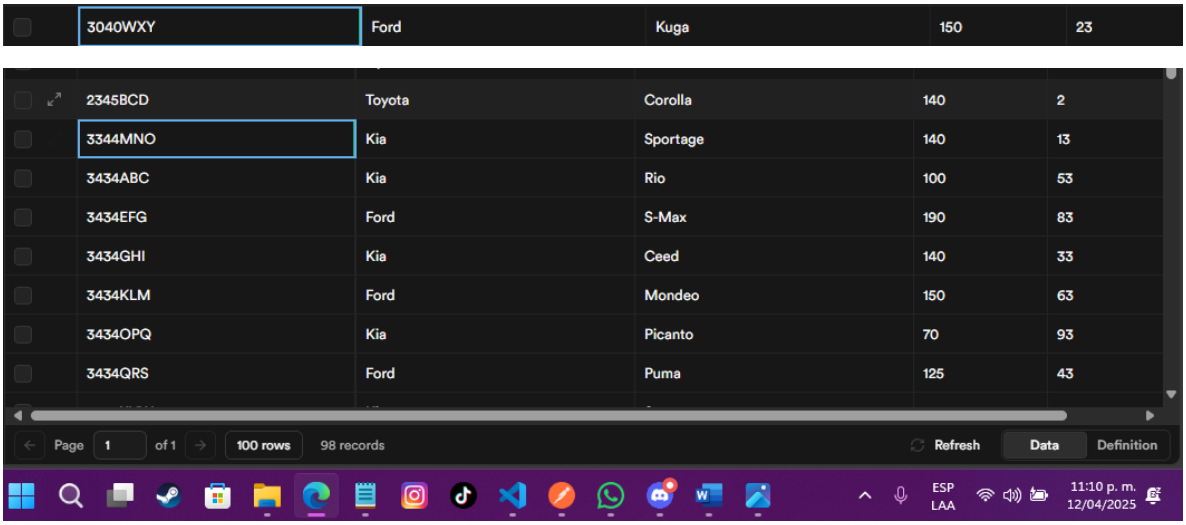
```
app.put('/api/coches/:matricula', async (req, res) => {
  const { matricula } = req.params;
  const { marca, modelo, caballos, persona_id } = req.body;
  try {
    const result = await client.query(
      'UPDATE coche SET marca = $1, modelo = $2, caballos = $3, persona_id = $4 WHERE matricula = $5 RETURNING *',
      [marca, modelo, caballos, persona_id, matricula]
    );
    if (result.rows.length === 0) {
      return res.status(404).json({ error: 'Coche no encontrado' });
    }
    res.status(200).json(result.rows[0]);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});
```

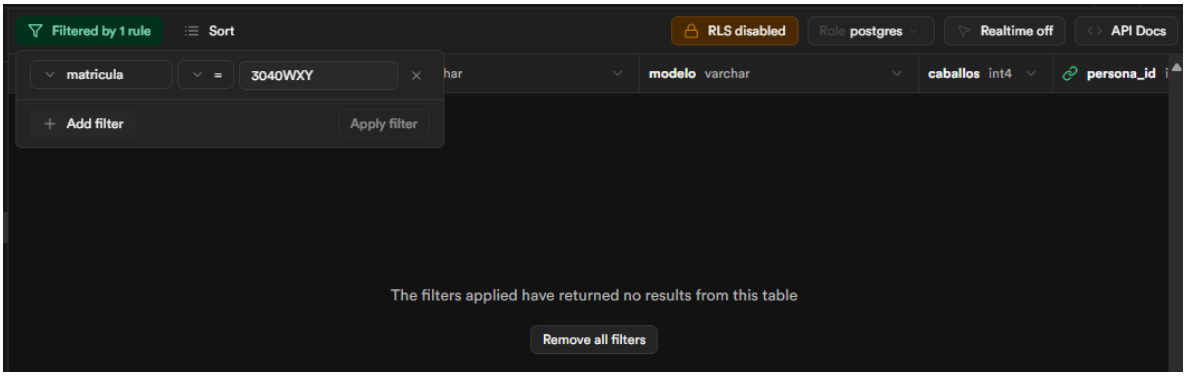


Borrar coches por matricula.

```
app.delete('/api/coches/:matricula', async (req, res) => {
  const { matricula } = req.params;
  try {
    const result = await client.query('DELETE FROM coche WHERE matricula = $1 RETURNING *', [matricula]);
    if (result.rows.length === 0) {
      return res.status(404).json({ error: 'Coche no encontrado' });
    }
    res.status(200).json({ mensaje: 'Coche eliminado' });
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});
```

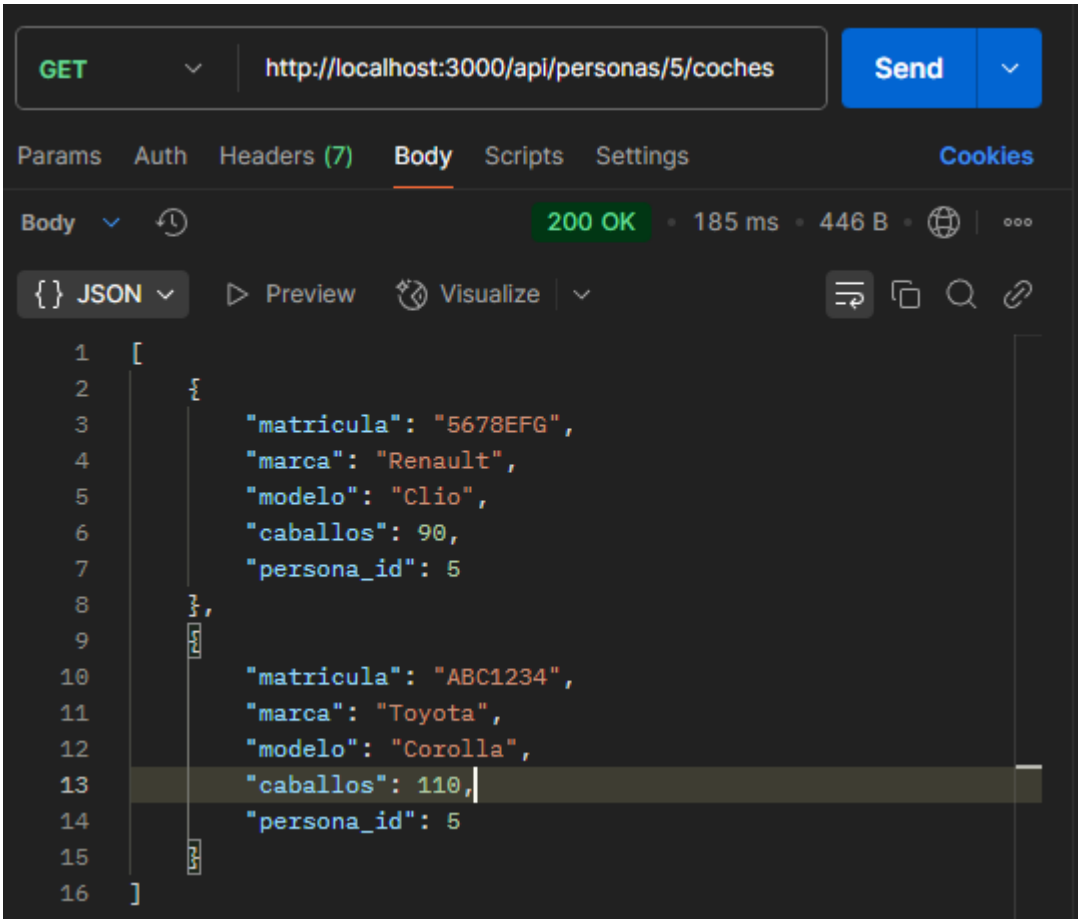


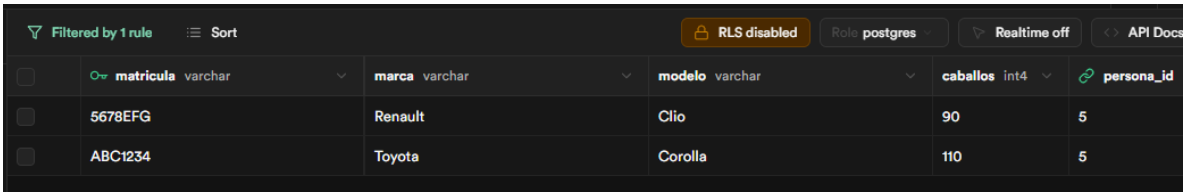




- **Endpoint para obtener los coches de una persona.**

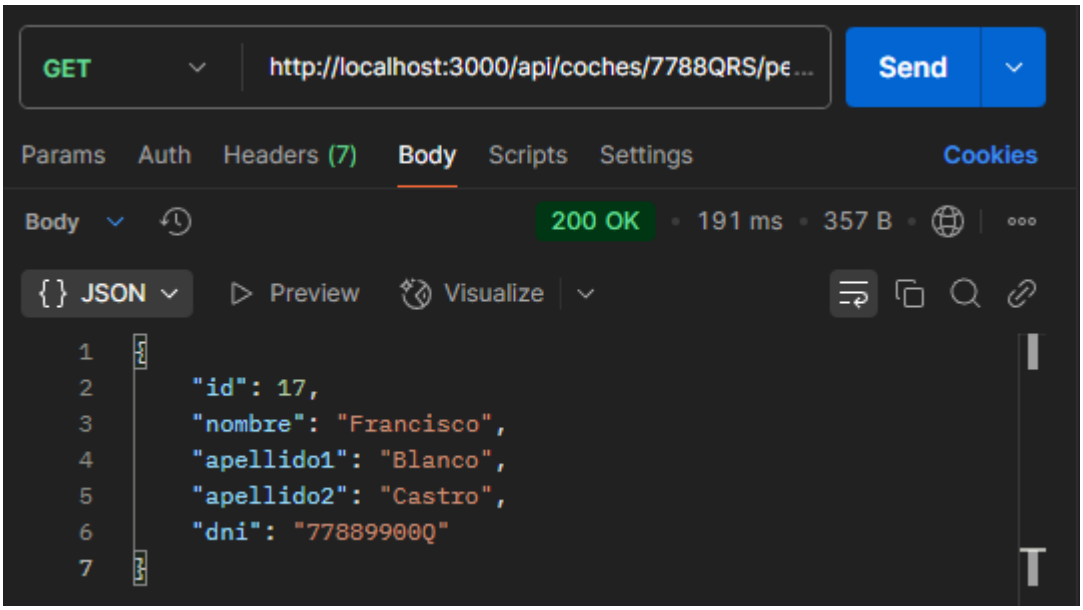
```
app.delete('/api/coches/:matricula', async (req, res) => {
  const { matricula } = req.params;
  try {
    const result = await client.query('DELETE FROM coche WHERE matricula = $1 RETURNING *', [matricula]);
    if (result.rows.length === 0) {
      return res.status(404).json({ error: 'Coche no encontrado' });
    }
    res.status(200).json({ mensaje: 'Coche eliminado' });
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});
```





- **Endpoint para obtener la persona dueña de un coche.**

```
// Endpoint para obtener la persona dueña de un coche
app.get('/api/coches/:matricula/persona', async (req, res) => {
  const { matricula } = req.params;
  try {
    // Verificar si el coche existe
    const coche = await client.query('SELECT * FROM Coche WHERE matricula = $1', [matricula]);
    if (coche.rows.length === 0) {
      return res.status(404).json({ error: 'Coche no encontrado' });
    }

    const result = await client.query(
      'SELECT p.* FROM Persona p JOIN Coche c ON p.id = c.persona_id WHERE c.matricula = $1',
      [matricula]
    );
    res.status(200).json(result.rows[0]);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
});
```



	7080ABC	Seat	Leon	150	27
	7788QRS	Skoda	Octavia	150	17
	7878EFG	Skoda	Kodiaq	190	57
	7878IJK	Seat	Tarraco	190	87
	7878KLM	Skoda	Fabia	110	37
	7878OPQ	Seat	Ateca	150	67
	7878STU	Skoda	Karoq	150	97
	7878UVW	Seat	Arona	115	47

Page 1 of 1 100 rows 98 records Refresh Data Definition