

# **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.

```
11 // Endpoints para Personas
12 app.post('/api/personas', async (req, res) => {
13   const { nombre, apellido1, apellido2, dni } = req.body;
14   try {
15     const result = await client.query(
16       'INSERT INTO Persona (nombre, apellido1, apellido2, dni) VALUES ($1, $2, $3, $4) RETURNING *'
17       [nombre, apellido1, apellido2, dni]
18     );
19     res.status(201).json(result.rows[0]);
20   } catch (err) {
21     res.status(500).json({ error: err.message });
22   }
23 });
```

The screenshot shows a REST client interface with the following details:

- Method:** POST
- URL:** http://localhost:3000/api/personas
- Body Type:** JSON
- Request Body:**

```
{
  "nombre": "Tralalero",
  "apellido1": "Tralala",
  "apellido2": "Trulichina",
  "dni": "316543283"
}
```
- Status:** 201 Created
- Response Time:** 90 ms
- Response Size:** 368 B
- Response Body:**

```
{
  "id": 105,
  "nombre": "Tralalero",
  "apellido1": "Tralala",
  "apellido2": "Trulichina",
  "dni": "316543283"
}
```

Shepor's Org Free BaseDeDatosPaTodo Connect Enable branching Feedback

Table Editor

Filter Sort Insert

RLS disabled Role postgres Realtime off API Docs

	Id Int4	nombre varchar	apellido1 varchar	apellido2 varchar	dni varchar
	105	Tralalero	Tralala	Trulichina	316543283

schema public

+ New table

Search tables...

coche

persona

## Mostrar personas

```

25 app.get('/api/personas', async (req, res) => {
26   try {
27     const result = await client.query('SELECT * FROM Persona');
28     res.status(200).json(result.rows);
29   } catch (err) {
30     res.status(500).json({ error: err.message });
31   }
32 });
33

```

GET http://localhost:3000/api/personas Send

Params Auth Headers (7) Body Scripts Settings Cookies

Body 200 OK • 90 ms • 9.08 KB

JSON Preview Visualize

```

1  [
2    {
3      "id": 2,
4      "nombre": "María",
5      "apellido1": "Martínez",
6      "apellido2": "Sánchez",
7      "dni": "23456789B"
8    },
9    {
10     "id": 3,
11     "nombre": "Carlos",
12     "apellido1": "Rodríguez",
13     "apellido2": "Fernández",
14     "dni": "34567890C"
15   },
16   {
17     "id": 4,
18     "nombre": "Ana",
19     "apellido1": "López",
20     "apellido2": "Gómez",
21     "dni": "45678901D"
22   }
23 ]

```

## Mostrar persona por ID

```
34 app.get('/api/personas/:id', async (req, res) => {
35   const { id } = req.params;
36   try {
37     const result = await client.query('SELECT * FROM Persona WHERE id = $1', [id]);
38     if (result.rows.length === 0) {
39       return res.status(404).json({ error: 'Persona no encontrada' });
40     }
41     res.status(200).json(result.rows[0]);
42   } catch (err) {
43     res.status(500).json({ error: err.message });
44   }
45 });
```

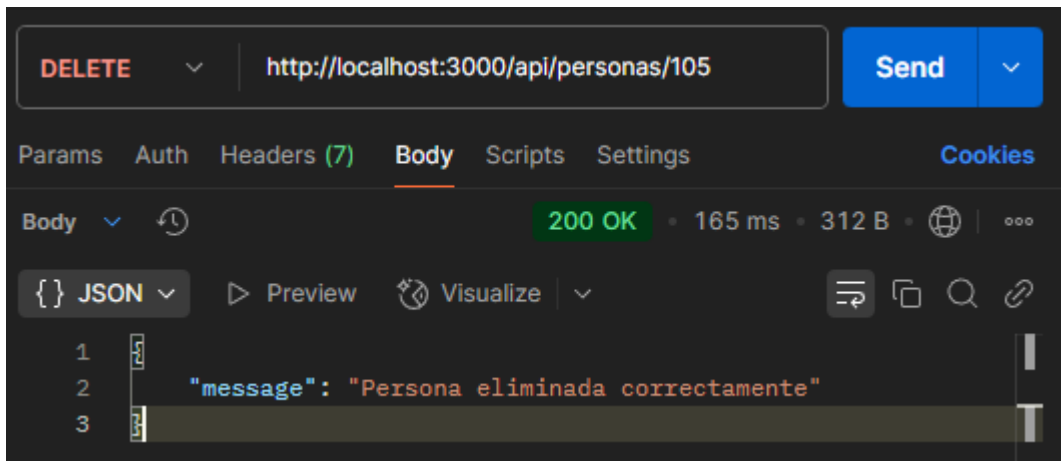
The screenshot shows a web browser interface for testing HTTP requests. The method is set to GET and the URL is `http://localhost:3000/api/personas/84`. The response status is 200 OK, with a response time of 135 ms and a body size of 354 B. The response body is displayed in JSON format, showing the details of a person with ID 84.

```
{
  "id": 84,
  "nombre": "Emilia",
  "apellido1": "Barreiro",
  "apellido2": "Vela",
  "dni": "45456789F"
}
```

## Actualizar persona por ID

```
47 app.put('/api/personas/:id', async (req, res) => {
48   const { id } = req.params;
49   const { nombre, apellido1, apellido2, dni } = req.body;
50
51   try {
52     const result = await client.query(
53       'UPDATE Persona SET nombre = $1, apellido1 = $2, apellido2 = $3, dni = $4 WHERE id = $5 RETURNING *',
54       [nombre, apellido1, apellido2, dni, id]
55     );
56     if (result.rows.length === 0) {
57       return res.status(404).json({ error: 'Persona no encontrada' });
58     }
59     res.status(200).json(result.rows[0]);
60   } catch (err) {
61     res.status(500).json({ error: err.message });
62   }
63 });
```

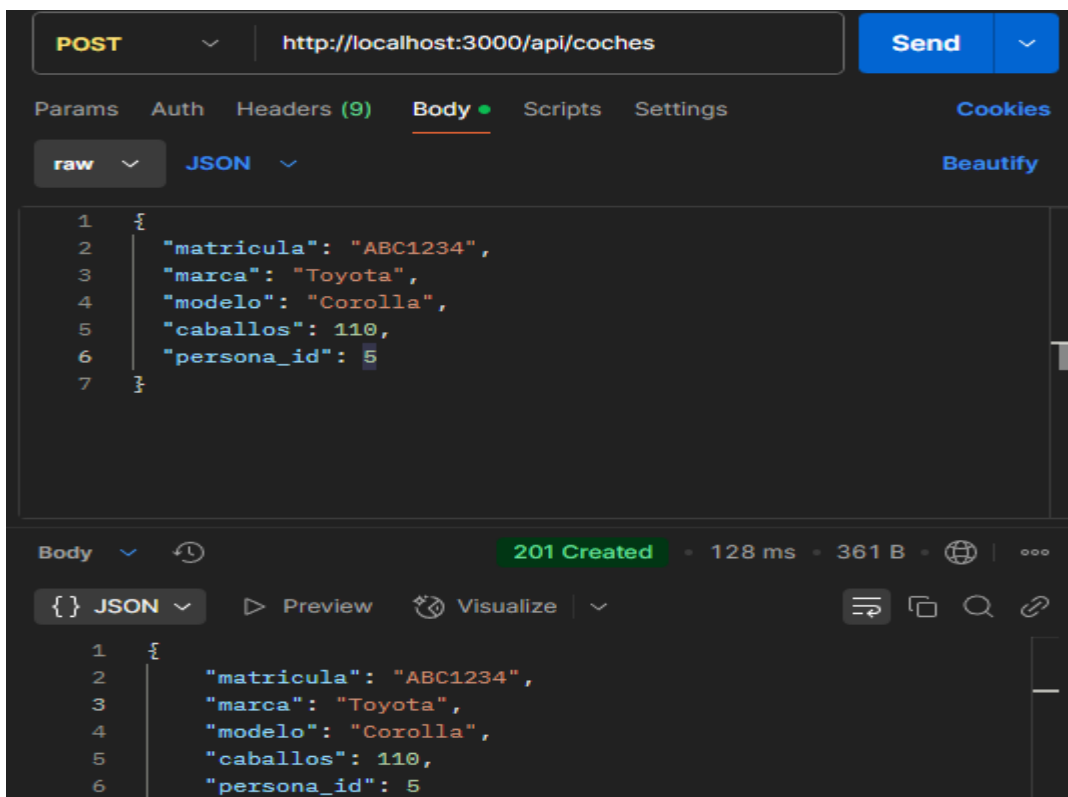




- Endpoints para coches.

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 });
  }
});
```



<input type="checkbox"/>	9900STU	Mazda	CX-5	165	19
<input type="checkbox"/>	ABC1234	Toyota	Corolla	110	5

Page 1 of 1 100 rows 99 records Refresh Data Definition

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 });
  }
});
```

GET http://localhost:3000/api/coches Send

Params Auth Headers (9) Body • Scripts Settings Cookies

Body 200 OK • 97 ms • 8.79 KB

JSON Preview Visualize

```
[
  {
    "matricula": "2345BCD",
    "marca": "Honda",
    "modelo": "Civic",
    "caballos": 160,
    "persona_id": 2
  },
  {
    "matricula": "3456CDE",
    "marca": "Ford",
    "modelo": "Focus",
    "caballos": 125,
    "persona_id": 3
  },
  {
    "matricula": "4567DEF",
    "marca": "Volkswagen",
    "modelo": "Golf",
    "caballos": 150,
    "persona_id": 4
  }
]
```

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 });
  }
});
```

GET http://localhost:3000/api/coches/2345BCD Send

Params Auth Headers (9) **Body** Scripts Settings Cookies

Body 200 OK 95 ms 353 B 🌐 ⋮

{} JSON ▶ Preview 🔗 Visualize ⌵ ⌵ 🔍 🔗

```
1 {
2   "matricula": "2345BCD",
3   "marca": "Honda",
4   "modelo": "Civic",
5   "caballos": 160,
6   "persona_id": 2
7 }
```

<input type="checkbox"/>	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 });
  }
});
```



PUT ▼ http://localhost:3000/api/coches/2345BCD Send ▼

Params Auth Headers (9) **Body** ● Scripts Settings Cookies

raw ▼ JSON ▼ Beautify

```
1 {
2   "matricula": "2345BCD",
3   "marca": "Toyota",
4   "modelo": "Corolla",
5   "caballos": 140,
6   "persona_id": 2
7 }
```

Body ▼ 🕒 **200 OK** • 97 ms • 356 B • 🌐 ⋮

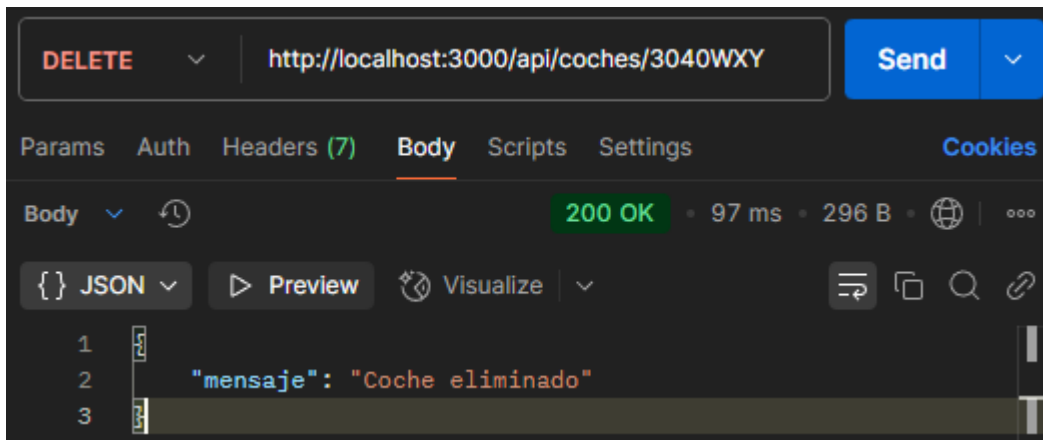
{ } JSON ▼ ▶ Preview 🔗 Visualize ▼ ≡ 📄 🔍 🔗

```
1 {
2   "matricula": "2345BCD",
3   "marca": "Toyota",
4   "modelo": "Corolla",
5   "caballos": 140,
6   "persona_id": 2
7 }
```

<input type="checkbox"/>	2345BCD	Toyota	Corolla	140	2
--------------------------	---------	--------	---------	-----	---

### Borrar coches por matricula.

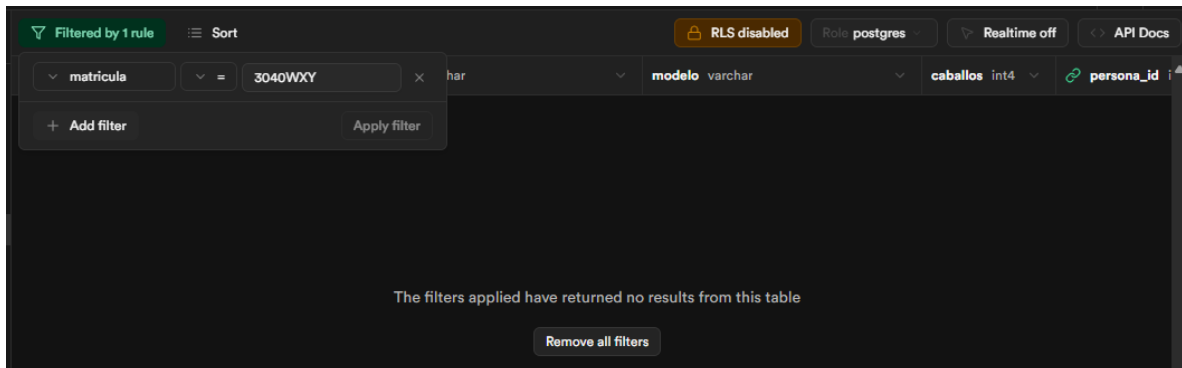
```
app.delete('/api/coches/:matricula', async (req, res) => {
  const { matricula } = req.params;
  try {
    const result = await client.query('DELETE FROM coche WHERE matricula = $1 RETURNING *', [matricula]);
    if (result.rows.length === 0) {
      return res.status(404).json({ error: 'Coche no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



<input type="checkbox"/>	3040WXY	Ford	Kuga	150	23
--------------------------	---------	------	------	-----	----

<input type="checkbox"/>	2345BCD	Toyota	Corolla	140	2
<input type="checkbox"/>	3344MNO	Kia	Sportage	140	13
<input type="checkbox"/>	3434ABC	Kia	Rio	100	53
<input type="checkbox"/>	3434EFG	Ford	S-Max	190	83
<input type="checkbox"/>	3434GHI	Kia	Ceed	140	33
<input type="checkbox"/>	3434KLM	Ford	Mondeo	150	63
<input type="checkbox"/>	3434OPQ	Kia	Picanto	70	93
<input type="checkbox"/>	3434QRS	Ford	Puma	125	43

Page 1 of 1 100 rows 98 records Refresh Data Definition



- 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 });
});
```

GET ▼ http://localhost:3000/api/personas/5/coches Send ▼

Params Auth Headers (7) **Body** Scripts Settings Cookies

Body ▼ 🕒 200 OK • 185 ms • 446 B • 🌐 | ⋮

{} JSON ▼ ▶ Preview 🔍 Visualize ▼ ☰ 📄 🔍 🔗

```

1  [
2    {
3      "matricula": "5678EFG",
4      "marca": "Renault",
5      "modelo": "Clio",
6      "caballos": 90,
7      "persona_id": 5
8    },
9    {
10     "matricula": "ABC1234",
11     "marca": "Toyota",
12     "modelo": "Corolla",
13     "caballos": 110,
14     "persona_id": 5
15   }
16 ]
```

Filtered by 1 rule Sort RLS disabled Role postgres Realtime off API Docs

<input type="checkbox"/>	<input checked="" type="checkbox"/> matricula	varchar	<input type="checkbox"/> marca	varchar	<input type="checkbox"/> modelo	varchar	<input type="checkbox"/> caballos	int4	<input checked="" type="checkbox"/> persona_id
<input type="checkbox"/>	5678EFG		Renault		Clio		90		5
<input type="checkbox"/>	ABC1234		Toyota		Corolla		110		5

- 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 });
  }
});
```

GET http://localhost:3000/api/coches/7788QRS/pe... Send

Params Auth Headers (7) **Body** Scripts Settings Cookies

Body 200 OK • 191 ms • 357 B • 🌐 ⋮

{} JSON ▶ Preview 🔗 Visualize ⌵ 🔄 📄 🔍 🔗

```
1 {
2   "id": 17,
3   "nombre": "Francisco",
4   "apellido1": "Blanco",
5   "apellido2": "Castro",
6   "dni": "77889900Q"
7 }
```

<input type="checkbox"/>	7080ABC	Seat	Leon	150	27
<input checked="" type="checkbox"/>	7788QRS	Skoda	Octavia	150	17
<input type="checkbox"/>	7878EFG	Skoda	Kodiaq	190	57
<input type="checkbox"/>	7878IJK	Seat	Tarraco	190	87
<input type="checkbox"/>	7878KLM	Skoda	Fabia	110	37
<input type="checkbox"/>	7878OPQ	Seat	Ateca	150	67
<input type="checkbox"/>	7878STU	Skoda	Karoq	150	97
<input type="checkbox"/>	7878UVW	Seat	Arona	115	47

Page 1 of 1 100 rows 98 records Refresh Data Definition

11:16 p. m. 12/04/2025