

Taller Bases de Datos

Juan Sebastian Dueñas Robayo

833539

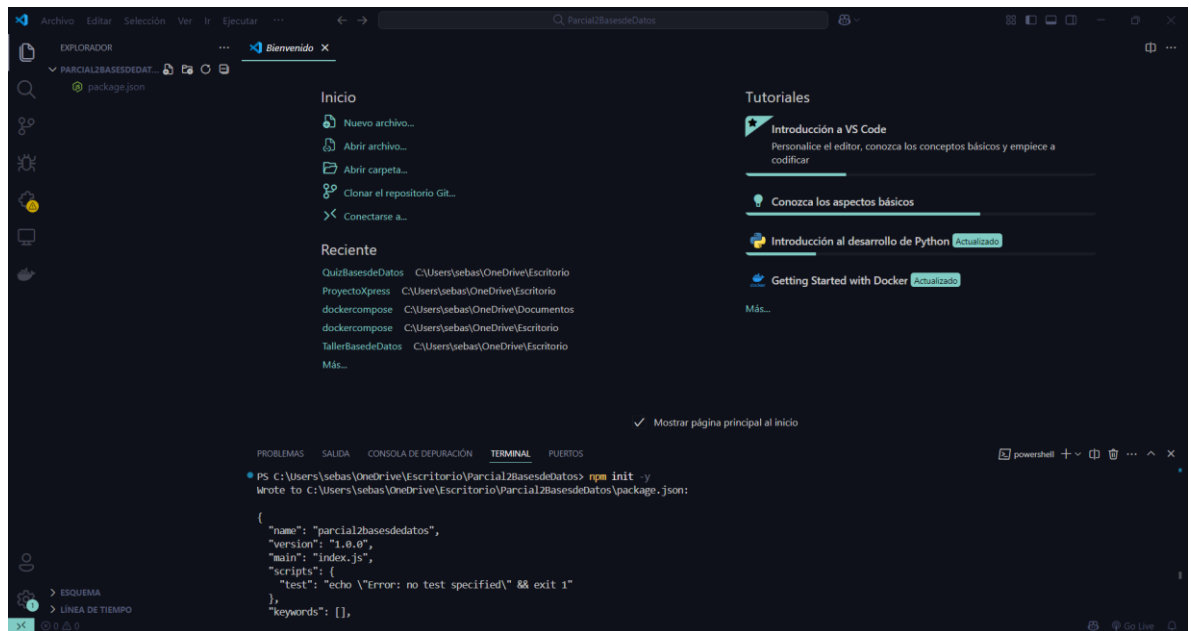
Corporación Universitaria Minuto de Dios

Ing. William Alexander Matallana Porras

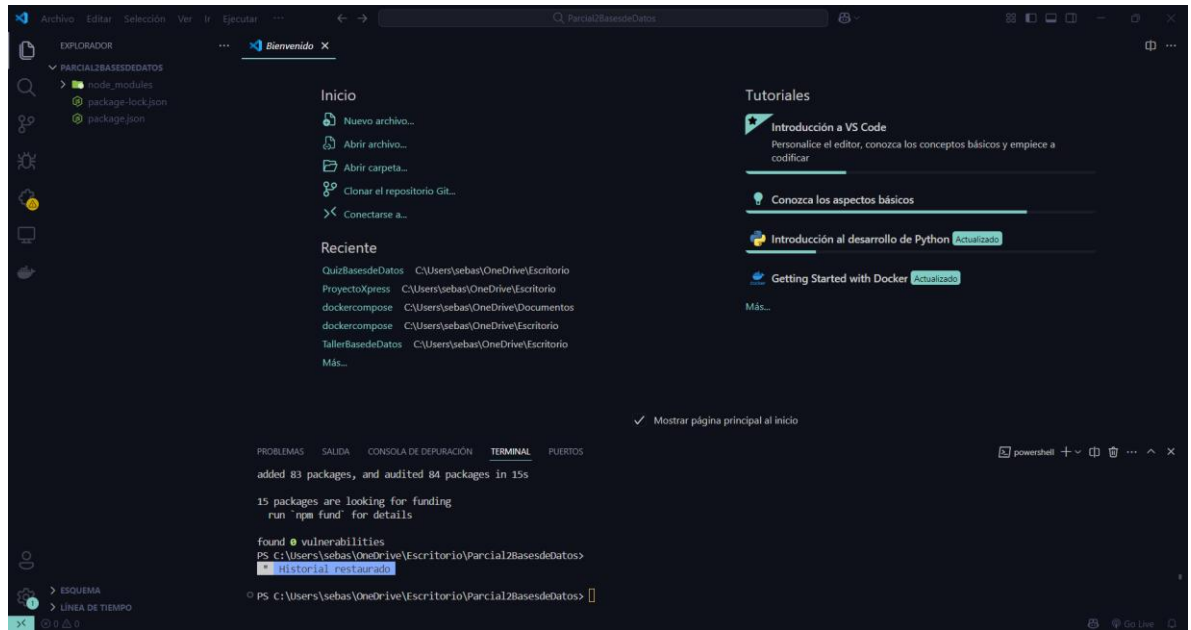
10-60747: Bases de Datos masivas

2025

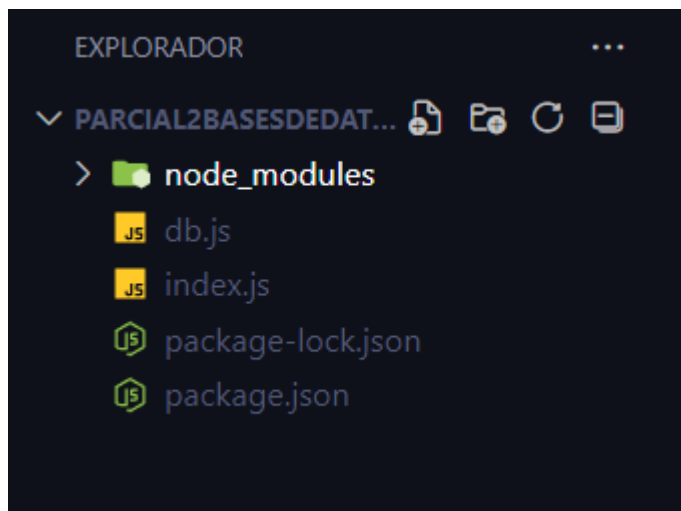
1. Comenzamos creando una carpeta ya sea en nuestro escritorio que es donde vamos a manejar nuestro proyecto en visual studio.
2. Una vez creada lo abrimos con visual studio e inicializamos con el comando **npm init -y**, y este nos creará un archivo. Json



3. A su vez con el comando **npm install express pg cors dotenv**, terminamos de instalar los modulos de express



4. Luego de eso crearemos nuestro index.js y nuestro db.js fuera de la carpeta de node_modules

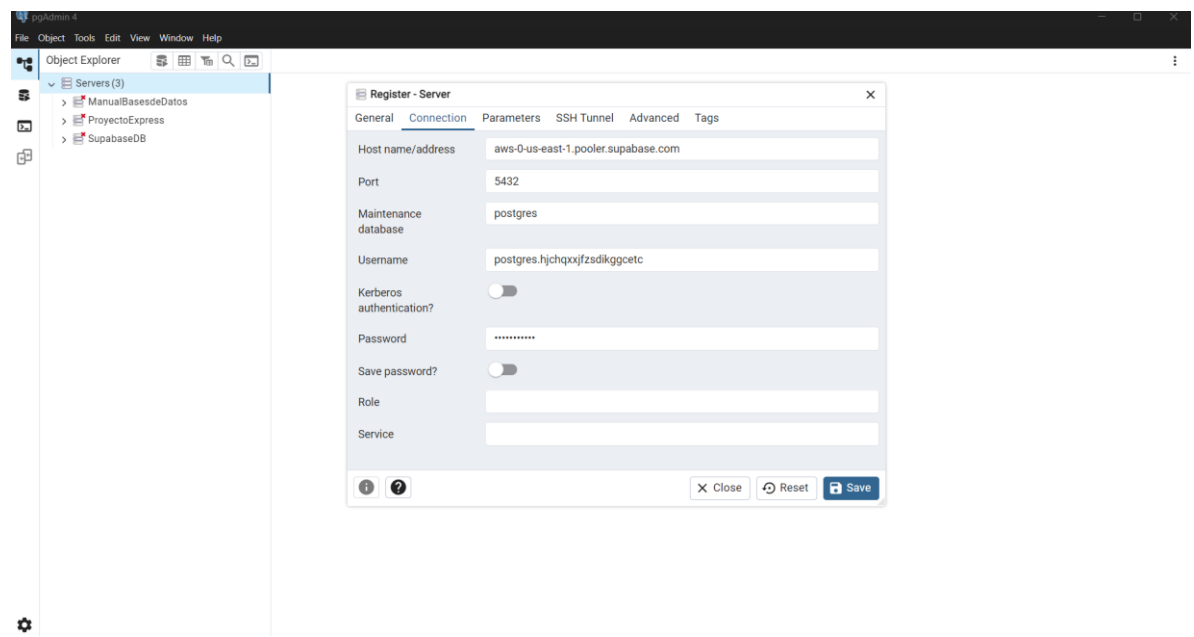


5. Abriremos nuestro pgadmin y registramos un nuevo servidor, le colocamos el nombre de nuestra preferencia y para la conexión hacemos lo siguiente:
Nos dirigimos al supabase en nuestro proyecto y seleccionamos la parte de “Connect”, y copiaremos el que se llama **SESSION POOLER**

postgresql://postgres.vgzevuspoddouyqqqt:[YOUR-PASSWORD]@aws-0-us-east-1.pooler.supabase.com:5432/postgres

El host siempre será la parte señalada en color rojo y nuestro usuario la parte señalada en color azul, el password es el mismo que uno asigna al momento de crear el proyecto en supabase

6. En el Pgadmin se observaría de la siguiente manera



7. Editamos el db.js para hacer la conexión con supabase, se vería de la siguiente manera (Maneja los mismos datos antes insetados en pgadmin)

```

1  const { Pool } = require('pg');
2
3  const pool = new Pool({
4    host: 'aws-0-us-east-1.pooler.supabase.com',
5    port: 5432,
6    user: 'postgres.hjchqxxjzsdikgctc',
7    password: 'Sebastrox05',
8    database: 'postgres',
9    ssl: {
10     rejectUnauthorized: false
11   }
12 });
13
14 pool.connect((err) => {
15   if (err) {
16     console.error('❌ Error en la conexión con Supabase:', err);
17   } else {
18     console.log('✅ Conectado a la base de datos de Supabase');
19   }
20 });
21
22 module.exports = pool;

```

added 83 packages, and audited 84 packages in 15s

15 packages are looking for funding
run `npm fund` for details

found 0 vulnerabilities

PS C:\Users\sebas\OneDrive\Escritorio\Parcial2BasesdeDatos>

8. Antes de manipular nuestro `index.js`, en supabase específicamente en el apartado de **SQL EDITOR**, crearemos las tablas correspondientes

```

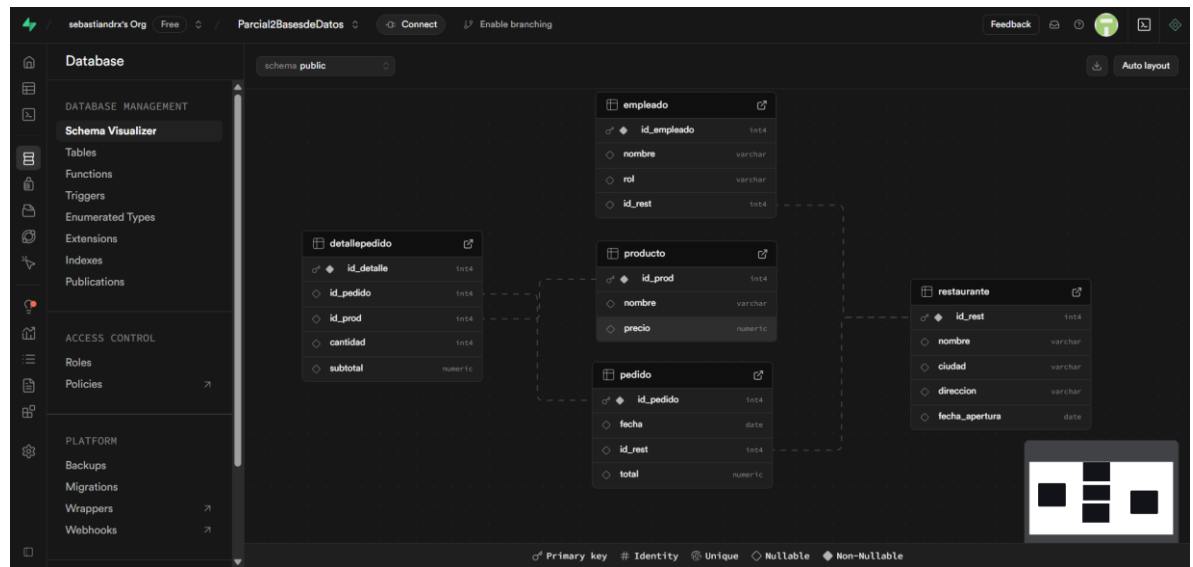
1  CREATE TABLE restaurante (
2    id_rest INT PRIMARY KEY,
3    nombre VARCHAR(100),
4    ciudad VARCHAR(100),
5    direccion VARCHAR(150),
6    fecha_apertura DATE
7  );
8
9  CREATE TABLE empleado (
10   id_empleado INT PRIMARY KEY,
11   nombre VARCHAR(100),
12   rol VARCHAR(50),
13   id_rest INT REFERENCES restaurante(id_rest)
14 );
15
16 CREATE TABLE producto (
17   id_prod INT PRIMARY KEY,
18   nombre VARCHAR(100),
19   precio NUMERIC(10,2)
20 );
21
22 CREATE TABLE pedido (
23   id_pedido INT PRIMARY KEY,
24   fecha DATE,
25   id_rest INT REFERENCES restaurante(id_rest),
26   total NUMERIC(10,2)
27 );

```

Results Chart

Click Run to execute your query.

View running queries



9. Luego de insertar los 50 registros por tabla se vería de la siguiente manera

Name	Description	Rows (Estimated)	Size (Estimated)	Realtime Enabled	
detallepedido	No description	50	24 kB	×	5 columns [] ⋮
empleado	No description	50	24 kB	×	4 columns [] ⋮
pedido	No description	50	24 kB	×	4 columns [] ⋮
producto	No description	50	24 kB	×	3 columns [] ⋮
restaurante	No description	50	24 kB	×	5 columns [] ⋮

10. Luego de tener todos nuestros controller y routes con el index llamándolos a cada, corremos el servicio para empezar con las consultas en postman

```

const PORT = 3000;

const restauranteRoutes = require('./Routes/restaurantes.js');
app.use('/restaurantes', restauranteRoutes);

const empleadoRoutes = require('./Routes/empleados.js');
app.use('/empleados', empleadoRoutes);

const productoRoutes = require('./Routes/productos.js');
app.use('/productos', productoRoutes);

const pedidoRoutes = require('./Routes/pedidos.js');
app.use('/pedidos', pedidoRoutes);

const detalleRoutes = require('./Routes/detallesPedido.js');
app.use('/detalles', detalleRoutes);

//ConsultasNativas
const consultasRoutes = require('./Routes/consultas.js');
app.use('/consultas', consultasRoutes);

app.listen(PORT, () => {
  console.log(`Servidor corriendo en puerto ${PORT}`);
});

```

Restarting 'index.js'

- ✓ Servidor corriendo en puerto 3000
- ✓ Conectado a la base de datos de supabase

11. Consultas

- Obtener todos los productos de un pedido específico GET:

<http://localhost:3000/consultas/productos-pedido/1>

GET Get data

Parcial2BasesdeDatos / Pedido / Get data

GET Send

Params Authorization Headers (6) Body Scripts Settings Cookies

Query Params

Key	Value	Description
Key	Value	Description

Body Cookies Headers (7) Test Results (1/1) 200 OK 654 ms 370 B Save Response

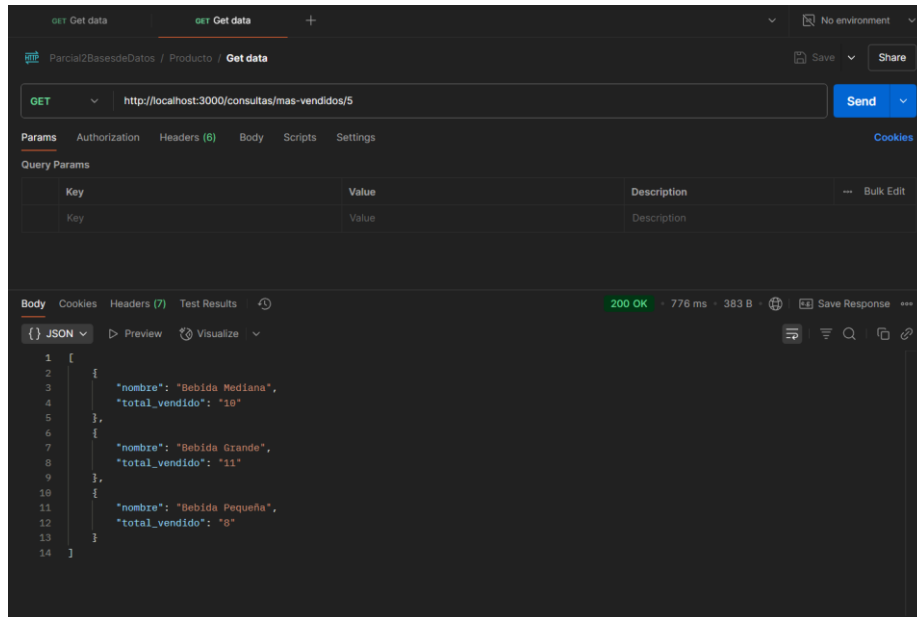
```

1 [
2   {
3     "nombre": "Hamburguesa Clásica",
4     "cantidad": 2,
5     "subtotal": "24000.00"
6   },
7   {
8     "nombre": "Bebida Pequeña",
9     "cantidad": 3,
10    "subtotal": "12000.00"
11  }
12 ]

```

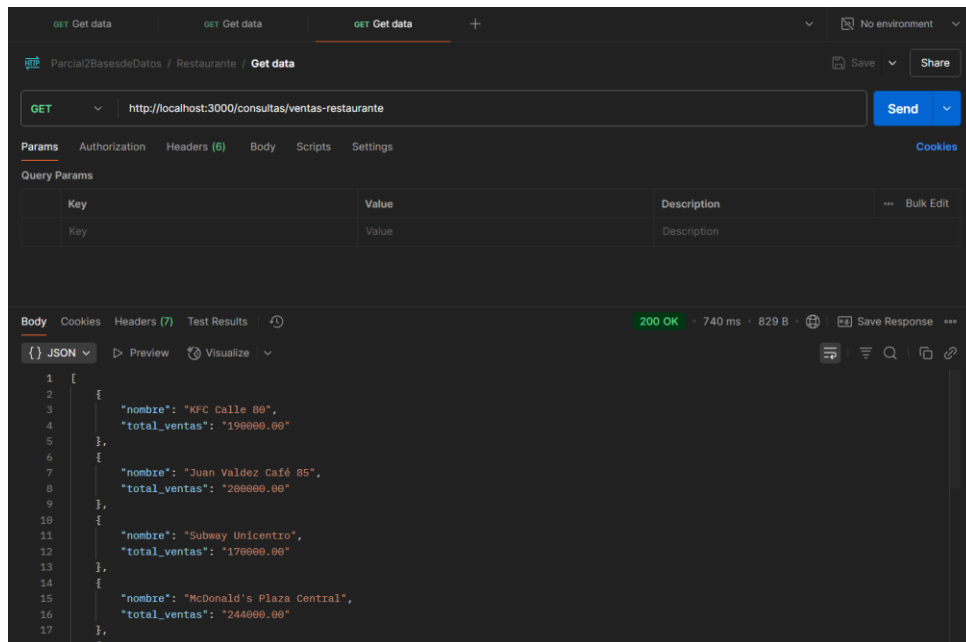
- Obtener los productos más vendidos (más de X unidades) GET:

<http://localhost:3000/consultas/mas-vendidos/5>



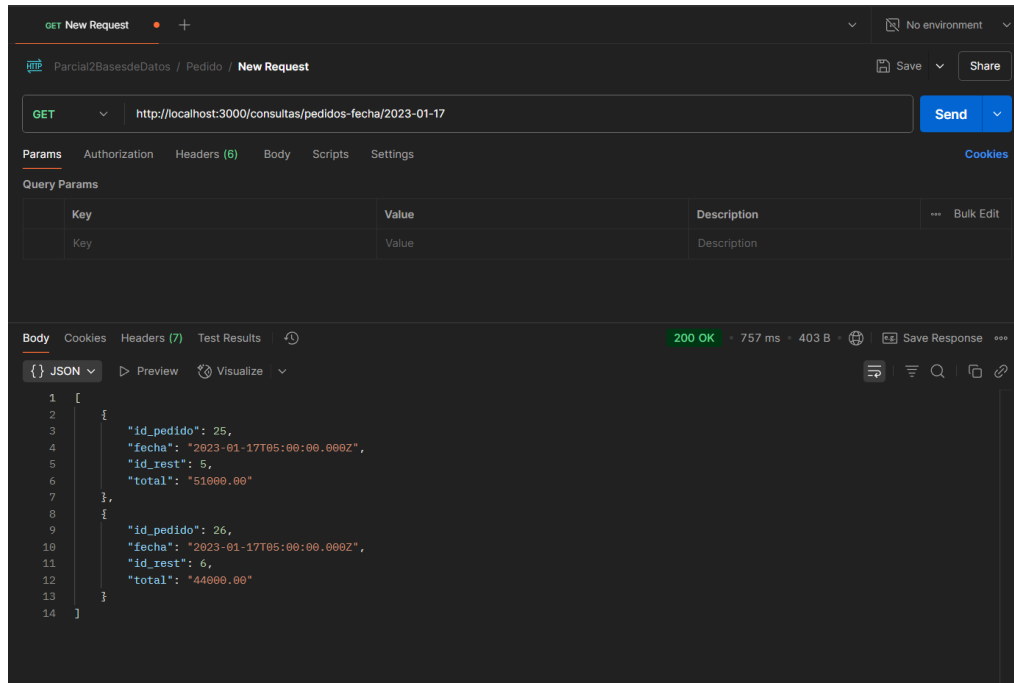
- Obtener el total de ventas por restaurante GET:

<http://localhost:3000/consultas/ventas-restaurante>



- Obtener los pedidos realizados en una fecha específica GET:

<http://localhost:3000/consultas/pedidos-fecha/2023-01-17>



GET New Request

Parcial2BasesdeDatos / Pedido / New Request

GET <http://localhost:3000/consultas/pedidos-fecha/2023-01-17> Send

Params Authorization Headers (6) Body Scripts Settings Cookies

Query Params

Key	Value	Description
Key	Value	Description

Body Cookies Headers (7) Test Results

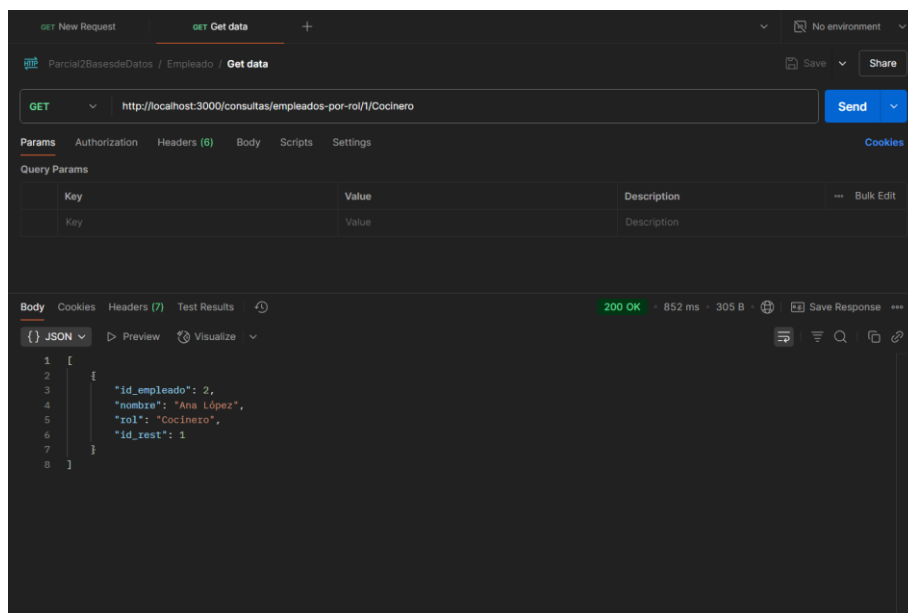
200 OK · 757 ms · 403 B Save Response

JSON Preview Visualize

```
1 [
2   {
3     "id_pedido": 25,
4     "fecha": "2023-01-17T05:00:00.000Z",
5     "id_rest": 5,
6     "total": "51600.00"
7   },
8   {
9     "id_pedido": 26,
10    "fecha": "2023-01-17T05:00:00.000Z",
11    "id_rest": 6,
12    "total": "44800.00"
13  }
14 ]
```

- Obtener los empleados por rol en un restaurante GET:

<http://localhost:3000/consultas/empleados-por-rol/1/Cocinero>



GET New Request

Parcial2BasesdeDatos / Empleado / Get data

GET <http://localhost:3000/consultas/empleados-por-rol/1/Cocinero> Send

Params Authorization Headers (6) Body Scripts Settings Cookies

Query Params

Key	Value	Description
Key	Value	Description

Body Cookies Headers (7) Test Results

200 OK · 852 ms · 305 B Save Response

JSON Preview Visualize

```
1 {
2   {
3     "id_empleado": 2,
4     "nombre": "Ana López",
5     "rol": "Cocinero",
6     "id_rest": 1
7   }
8 }
```