

Parcial segundo corte

Estudiante

Carlos Andrés Castellanos Arias

Id:667626

Sesto Semestre

Profesor:

William Alexander Matallana Porras

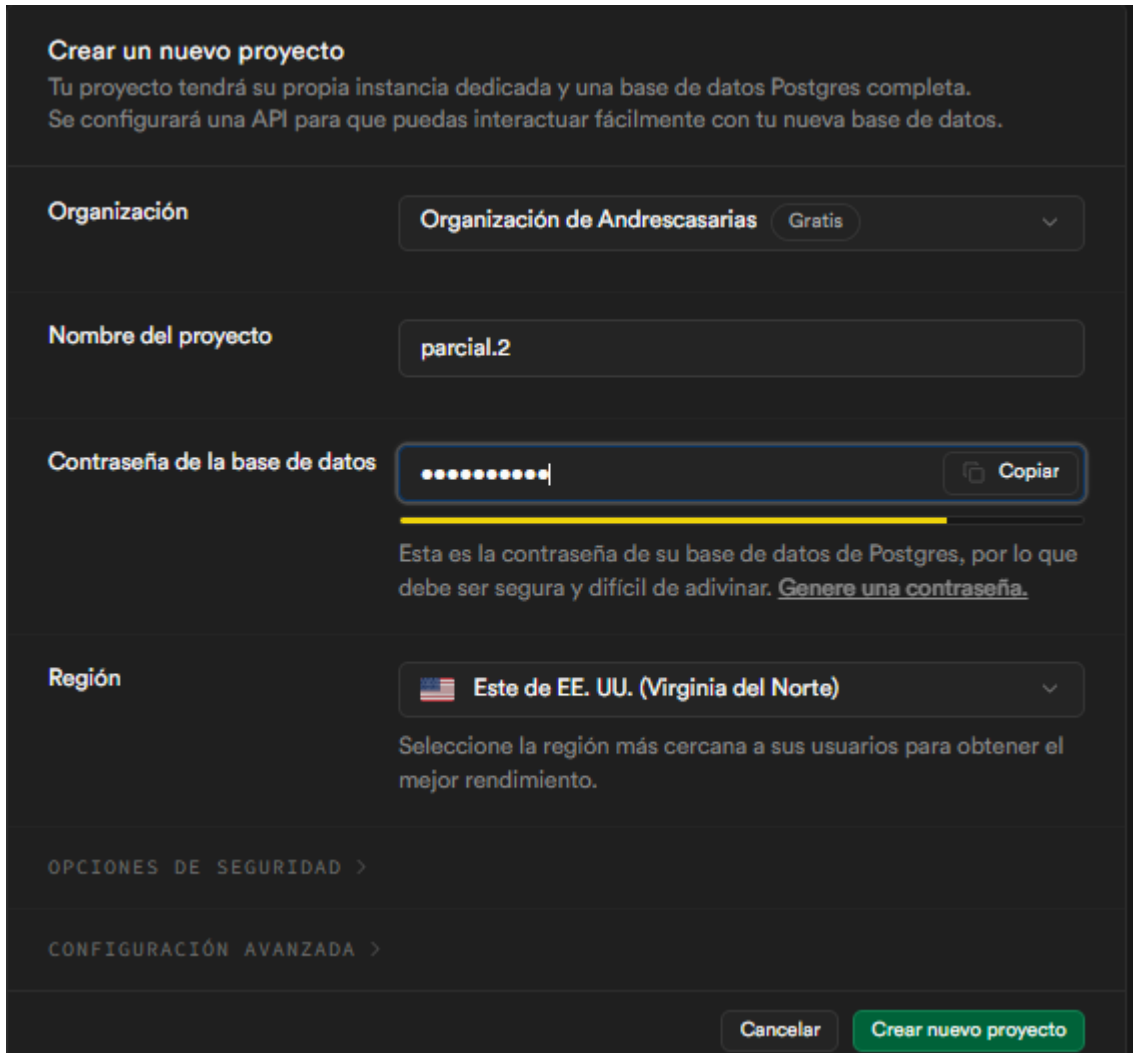
UNIMINUTO - CORPORACIÓN UNIVERSITARIA MINUTO DE DIOS

ZIPAQUIRÁ

Bases de datos masivas

2025

- Primero se crea el proyecto en supabase



Crear un nuevo proyecto
Tu proyecto tendrá su propia instancia dedicada y una base de datos Postgres completa.
Se configurará una API para que puedas interactuar fácilmente con tu nueva base de datos.

Organización
Organización de Andrescasarias Gratis

Nombre del proyecto
parcial.2

Contraseña de la base de datos
[Oculto] Copiar

Esta es la contraseña de su base de datos de Postgres, por lo que debe ser segura y difícil de adivinar. [Genere una contraseña.](#)

Región
🇺🇸 Este de EE. UU. (Virginia del Norte)

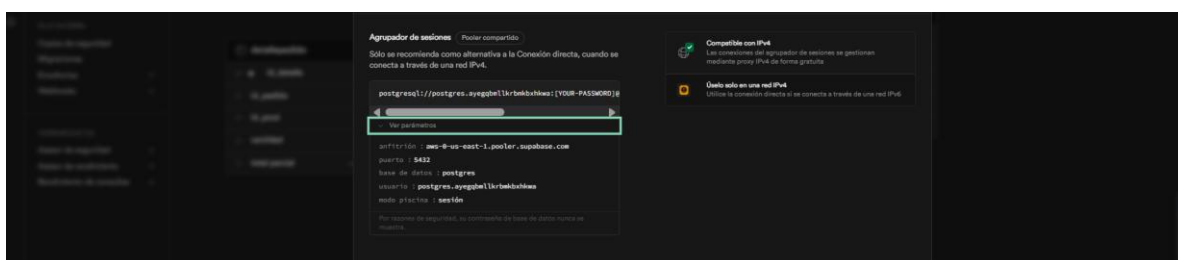
Seleccione la región más cercana a sus usuarios para obtener el mejor rendimiento.

[OPCIONES DE SEGURIDAD >](#)

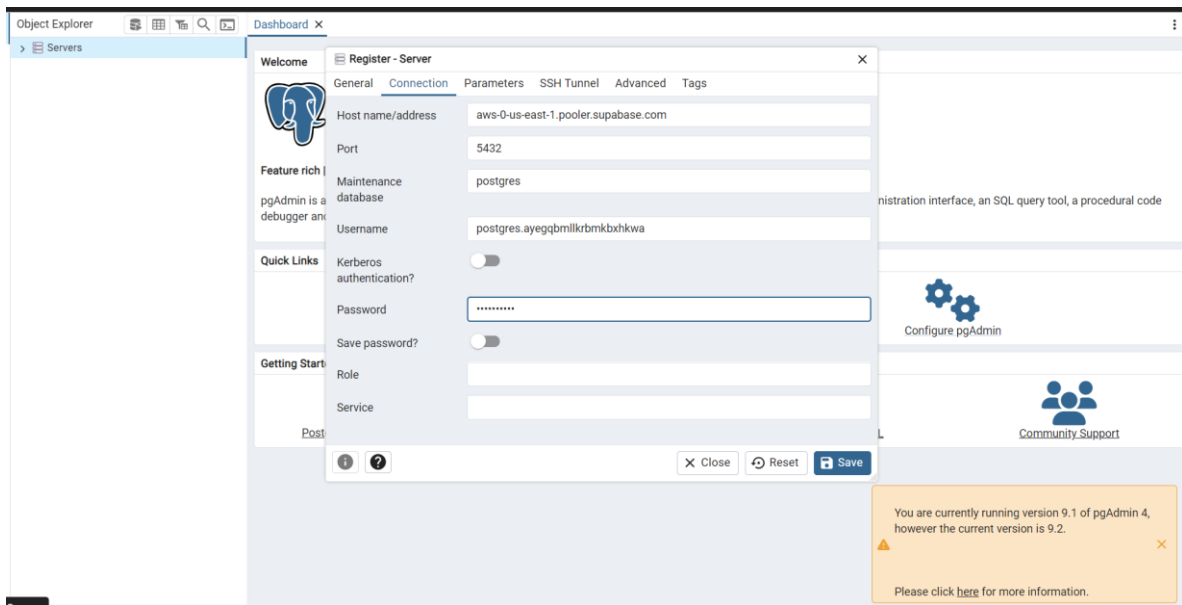
[CONFIGURACIÓN AVANZADA >](#)

Cancelar Crear nuevo proyecto

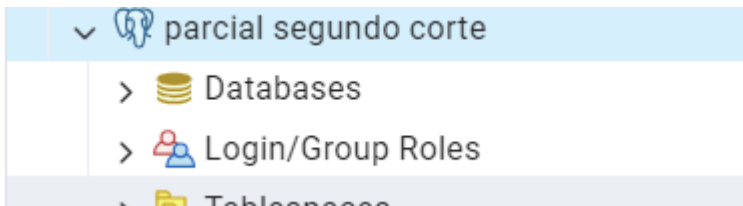
- Después entramos a conexión y buscamos los datos que nos sirven



- Nos vamos a pgadmin hacer la conexión del servidor de supabase

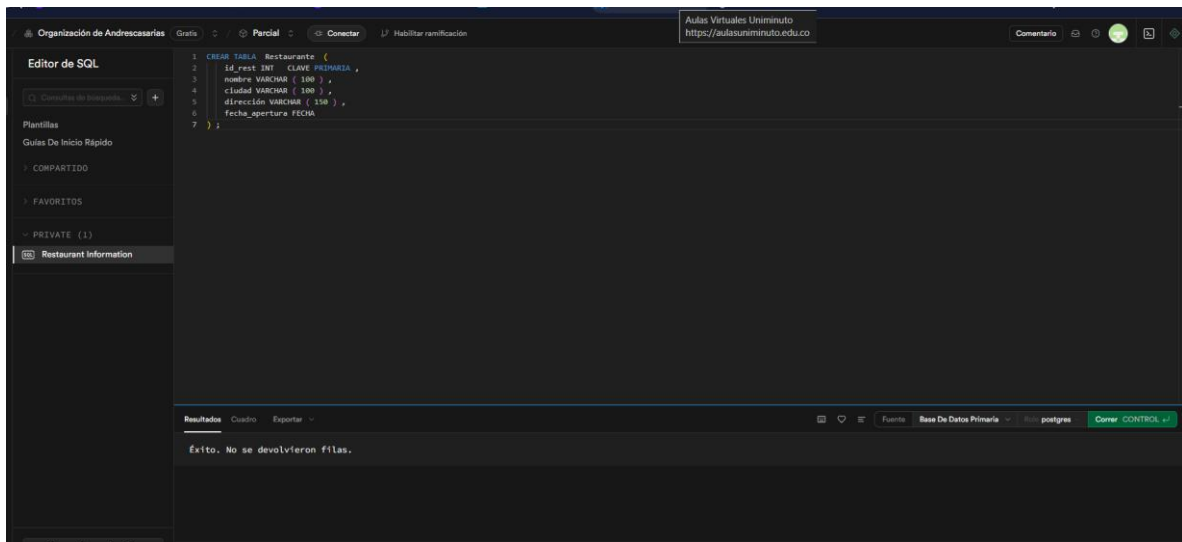


- Ya tenemos la conexión hecha

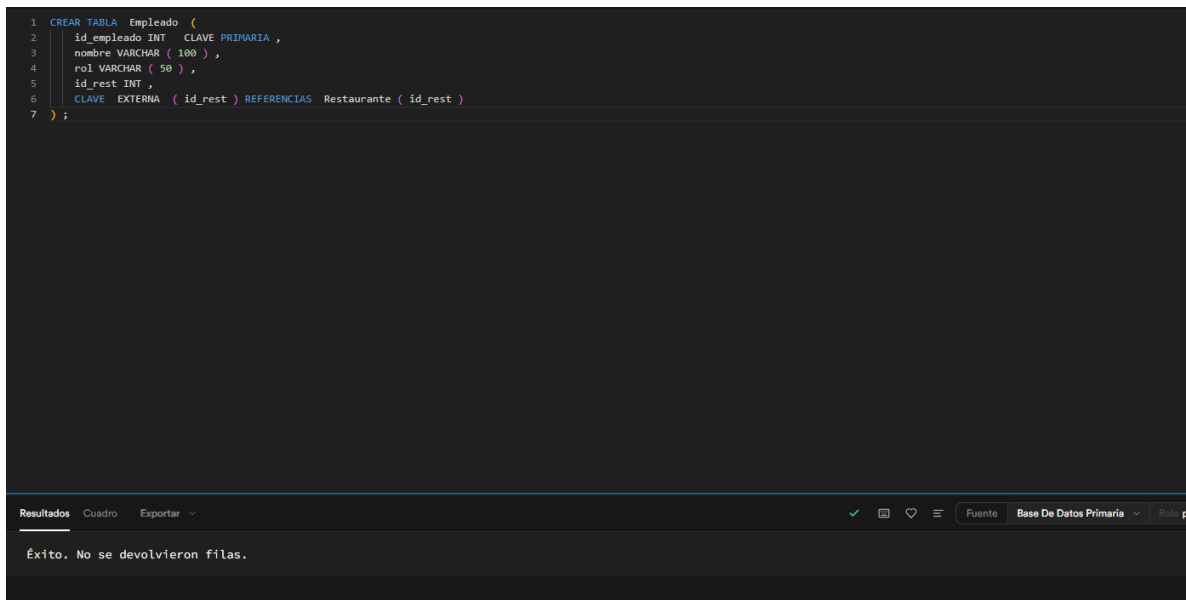


Creamos las tablas

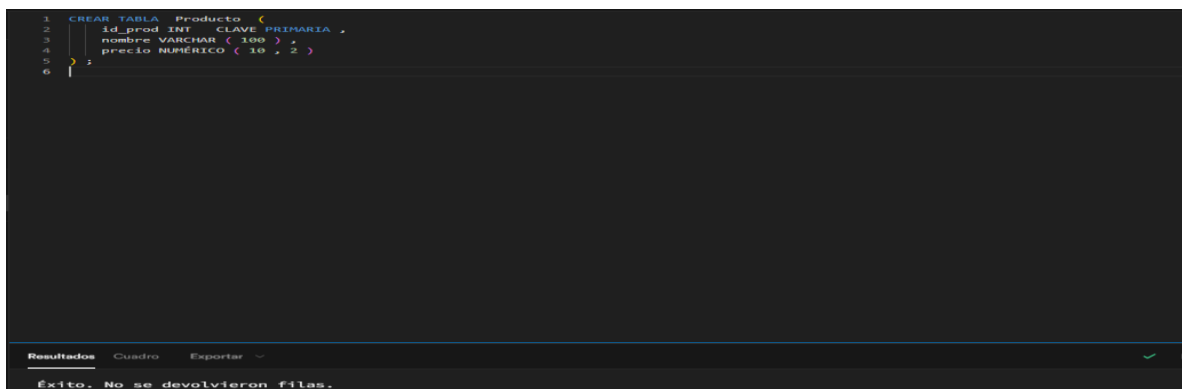
- Tabla restaurante



- Tabla empleado



- Tabla producto



- Tabla pedido

The screenshot shows a SQL Editor interface with a sidebar on the left containing 'Plantillas', 'Guías De Inicio Rápido', 'COMPARTIDO', 'FAVORITOS', 'PRIVATE (1)', and 'Restaurant Information'. The main editor area contains the following SQL code:

```
1 CREATE TABLE Pedido (
2   id_pedido INT CLAVE PRIMARIA ,
3   fecha FECHA ,
4   id_rest INT ,
5   total NUMÉRICO ( 10 , 2 ) ,
6   CLAVE EXTERNA ( id_rest ) REFERENCIAS Restaurante ( id_rest )
7 ) ;
```

At the bottom, the 'Resultados' tab is active, displaying the message: 'Éxito. No se devolvieron filas.'

- Tabla Detallepedido

The screenshot shows a SQL Editor interface with a sidebar on the left containing 'Plantillas', 'Guías De Inicio Rápido', 'COMPARTIDO', 'FAVORITOS', 'PRIVADO (1)', and 'Información del restaurante'. The main editor area contains the following SQL code:

```
1 CREATE TABLE Detallepedido (
2   id_detalle INT CLAVE PRIMARIA ,
3   id_pedido INT ,
4   id_prod INT ,
5   cantidad INT ,
6   subtotal NUMÉRICO ( 10 , 2 ) ,
7   CLAVE EXTRANJERA ( id_pedido ) REFERENCIAS Pedido ( id_pedido ) ,
8   CLAVE EXTERNA ( id_prod ) REFERENCIAS Producto ( id_prod )
9 ) ;
10
```

At the bottom, the 'Resultados' tab is active, displaying the message: 'Éxito. No se devolvieron filas.'

- Insertar datos los 50 datos por cada tabla

[supabase.com/dashboard/project/ayeggbmllkrmbxkhwa/qsl/1457132e-04ec-4856-b003-65593263745d](#)

Inicia sesión en tu c... Bookmarks Aplicaciones Diseño sin título YouTube Correos Castellanos... Aulas Virtuales Uni... Google Académico main.py - taller pyth...

Organización de Andrecasarias Gratis Parcial Conectar Habilitar ramificación

Editor de SQL

Plantillas Consultas de ejemplo +

Gües De Inicio Rápido

- COMPARTIDO
- FAVORITOS
- PRIVATE (1)
- Información del restaurante**

```

1 INSERT EN restaurante (id_rest, nombre, ciudad, dirección, fecha_apertura) VALUES
2 ('La casa de la Abuela', 'Bogotá', 'Calle 123 # 45-67', '2020-01-15'),
3 ('El Sincro Pasa', 'Medellín', 'Carrera 10A # 22-18', '2018-09-28'),
4 ('Mariscos del Pacífico', 'Cali', 'Avenida 5M # 10-05', '2002-03-01'),
5 ('Sabores de México', 'Bogotá', 'Calle 5E # 06-08', '2005-11-18'),
6 ('Pasta e vino', 'Medellín', 'Carrera 43 # 15-15', '2021-07-04'),
7 ('Sushi Master', 'Cali', 'Avenida 26 # 34-12', '2017-09-28'),
8 ('Cerveza & la Ciénaga', 'Bogotá', 'Calle 10W # 7-20', '2002-01-12'),
9 ('Los vegetarianos feliz', 'Medellín', 'Carrera 80 # 33-65', '2014-06-14'),
10 ('La Arquería', 'Cali', 'Avenida 1 # 44-55', '2020-12-09'),
11 ('Homerogarden Gourmet', 'Bogotá', 'Calle 4D # 24-05', '2018-08-01'),
12 ('Comida Árabe', 'Medellín', 'Carrera 51 # 10-25', '2002-06-18'),
13 ('Pollo frito express', 'Cali', 'Avenida 39 # 67-09', '2018-02-11'),
14 ('Pizzas del mundo', 'Bogotá', 'Calle 26 # 10-03', '2012-09-25'),
15 ('El Sabor de Asia', 'Medellín', 'Carrera 70 # 55-66', '2017-04-19'),
16 ('Empanadas Frescas', 'Cali', 'Avenida 14 # 21-21', '2003-04-05'),
17 ('Tacos y Aguacate', 'Bogotá', 'Calle 05 # 12-34', '2010-11-07'),
18 ('Crepes & Waffles', 'Medellín', 'Carrera 40 # 38-38', '2020-07-01'),
19 ('Jugos Naturales', 'Cali', 'Avenida 6 # 56-70', '2013-01-21'),
20 ('Comida de mar', 'Bogotá', 'Calle 33 # 10-10', '2002-08-12'),
21 ('El Café de la Mañana', 'Medellín', 'Carrera 05 # 46-48', '2018-09-08'),
22 ('Postres Deliciosos', 'Cali', 'Avenida 3A # 79-82', '2001-03-13'),
23 ('Dondeines y Amigos', 'Bogotá', 'Calle 07 # 20-54', '2017-12-01'),
24 ('Comida Peruana', 'Medellín', 'Carrera 5 # 36-52', '2003-08-27'),
25 ('Asados Argentinos', 'Cali', 'Avenida 20 # 04-06', '2016-04-11'),
26 ('Comida Española', 'Bogotá', 'Calle 22 # 11-01', '2008-11-04'),
27 ('El brunch Perfecto', 'Medellín', 'Carrera 5 # 41-75', '2013-03-08'),
28 ('Comida rústica', 'Cali', 'Avenida 42 # 01-09', '2002-07-22'),
29 ('Comida Salvadoreña', 'Bogotá', 'Calle 60 # 29-33', '2018-06-15'),
30 ('El Rey del Shawarma', 'Medellín', 'Carrera 5 # 37-61', '2001-02-26'),
31 ('Comida vietnamita', 'Cali', 'Avenida 33 # 02-95', '2007-08-30'),
32 ('Comida Italiana', 'Bogotá', 'Calle 75 # 13-02', '2003-09-10'),
33 ('El Jardín Secreto', 'Medellín', 'Carrera 63 # 43-73', '2016-05-03'),
34 ('Comida Mexicana', 'Cali', 'Avenida 46 # 03-12', '2008-12-14'),
35 ('Comida China', 'Bogotá', 'Calle 70 # 31-97', '2019-04-17'),
36 ('El Sabor del Caribe', 'Medellín', 'Carrera 12 # 39-65', '2002-08-24'),
37 ('Comida vegana', 'Cali', 'Avenida 43 # 05-10', '2013-01-22'),
38 ('Comida Francesa', 'Bogotá', 'Calle 81 # 33-41', '2001-11-09')

```

Resultados Cuando Exportar

Exitó. No se devolvieron filas.

```
Organización de Andrescasarias  Gratis  Parcial  Conectar  Habilitar ramificación

Editor de SQL

Plantillas
Guías De Inicio Rápido
COMPARTIDO
FAVORITOS
PRIVATE (1)
Información del restaurante

1 INSERTAR EN producto ( id_prod , nombre , precio ) VALORES
2 ( 1 , 'Hamburguesa clásica' , 10000 ) ,
3 ( 2 , 'Pizza Margarita' , 10000 ) ,
4 ( 3 , 'Sushi variado' , 25000 ) ,
5 ( 4 , 'Asadoado César' , 8000 ) ,
6 ( 5 , 'Ficus al pastor' , 10000 ) ,
7 ( 6 , 'Pasta Carbonara' , 10000 ) ,
8 ( 7 , 'Pollo a la Braster' , 14000 ) ,
9 ( 8 , 'Arroz con Queso' , 5000 ) ,
10 ( 9 , 'Jugo de Naranja' , 4000 ) ,
11 ( 10 , 'Café Americano' , 3000 ) ,
12 ( 11 , 'Limónada' , 5000 ) ,
13 ( 12 , 'Cerveza' , 6000 ) ,
14 ( 13 , 'Vino Tinto' , 20000 ) ,
15 ( 14 , 'Agua Mineral' , 2500 ) ,
16 ( 15 , 'Salsada' , 3000 ) ,
17 ( 16 , 'Papas Fritas' , 7000 ) ,
18 ( 17 , 'Arroz de cebolla' , 6500 ) ,
19 ( 18 , 'Mugret de Pollo' , 9000 ) ,
20 ( 19 , 'Alitas de Pollo' , 11000 ) ,
21 ( 20 , 'Sopa de Tomate' , 7500 ) ,
22 ( 21 , 'Crema de Champiñones' , 8500 ) ,
23 ( 22 , 'Caldado de Pollo' , 7000 ) ,
24 ( 23 , 'Arroz con Pollo' , 16000 ) ,
25 ( 24 , 'Frijoles con Chicharrón' , 13000 ) ,
26 ( 25 , 'Sandwich vesita' , 22000 ) ,
27 ( 26 , 'Alitas' , 15000 ) ,
28 ( 27 , 'Sancocho de gallina' , 20000 ) ,
29 ( 28 , 'Mondongo' , 17000 ) ,
30 ( 29 , 'Lechona' , 25000 ) ,
31 ( 30 , 'Tamales' , 11000 ) ,
32 ( 31 , 'Empanadas' , 4000 ) ,
33 ( 32 , 'Bafeludos' , 5500 ) ,
34 ( 33 , 'Churros' , 4500 ) ,
35 ( 34 , 'Flan de Caramelo' , 6000 ) ,
36 ( 35 , 'Torta de Chocolate' , 8000 ) ,
37 ( 36 , 'Molano de vainilla' , 5000 ) ,
38 ( 37 , 'Mousse de Maracuyá' , 7000 ) ,
```

```
supabase.com/dashboard/project/ayegqbmllkrbmkbxhkw/sql/1d57132e-04ec-4856-b003-655932637454

Inicio sesión en tu c...  Bookmarks  Aplicaciones  Diseño sin título  YouTube  Correo: Castellanos...  Aulas Virtuales Uni...  Google Académico  main.py - taller pyth...

Organización de Andrescasarias  Gratis  Parcial  Conectar  Habilitar ramificación

Editor de SQL

Plantillas
Guías De Inicio Rápido
COMPARTIDO
FAVORITOS
PRIVATE (1)
Información del restaurante

1 INSERTAR EN pedido ( id_pedido , fecha , id_rest , total ) VALORES
2 ( 1 , '2023-01-01' , 1 , 25000 ) ,
3 ( 2 , '2023-01-05' , 2 , 32000 ) ,
4 ( 3 , '2023-01-25' , 3 , 78000 ) ,
5 ( 4 , '2023-01-05' , 4 , 42000 ) ,
6 ( 5 , '2023-01-27' , 5 , 62000 ) ,
7 ( 6 , '2023-01-27' , 6 , 29000 ) ,
8 ( 7 , '2023-01-08' , 7 , 31000 ) ,
9 ( 8 , '2023-01-25' , 8 , 38000 ) ,
10 ( 9 , '2023-01-29' , 9 , 50000 ) ,
11 ( 10 , '2023-01-29' , 10 , 47000 ) ,
12 ( 11 , '2023-01-10' , 11 , 71000 ) ,
13 ( 12 , '2023-01-30' , 12 , 35000 ) ,
14 ( 13 , '2023-01-11' , 13 , 65000 ) ,
15 ( 14 , '2023-01-11' , 14 , 52000 ) ,
16 ( 15 , '2023-02-01' , 15 , 84000 ) ,
17 ( 16 , '2023-02-01' , 16 , 49000 ) ,
18 ( 17 , '2023-02-02' , 17 , 75000 ) ,
19 ( 18 , '2023-02-02' , 18 , 39000 ) ,
20 ( 19 , '2023-02-03' , 19 , 68000 ) ,
21 ( 20 , '2023-02-03' , 20 , 50000 ) ,
22 ( 21 , '2023-02-04' , 21 , 88000 ) ,
23 ( 22 , '2023-02-04' , 22 , 43000 ) ,
24 ( 23 , '2023-02-05' , 23 , 79000 ) ,
25 ( 24 , '2023-02-05' , 24 , 60000 ) ,
26 ( 25 , '2023-02-06' , 25 , 95000 ) ,
27 ( 26 , '2023-02-06' , 26 , 51000 ) ,
28 ( 27 , '2023-02-07' , 27 , 82000 ) ,
29 ( 28 , '2023-02-07' , 28 , 45000 ) ,
30 ( 29 , '2023-02-08' , 29 , 73000 ) ,
31 ( 30 , '2023-02-08' , 30 , 64000 ) ,
32 ( 31 , '2023-02-09' , 31 , 91000 ) ,
33 ( 32 , '2023-02-09' , 32 , 58000 ) ,
34 ( 33 , '2023-02-10' , 33 , 85000 ) ,
35 ( 34 , '2023-02-10' , 34 , 70000 ) ,
36 ( 35 , '2023-02-11' , 35 , 98000 ) ,
37 ( 36 , '2023-02-11' , 36 , 54000 ) ,
38 ( 37 , '2023-02-12' , 37 , 93000 ) ,

Routefiles  Estado  Copiar
```

```
supabase.com/dashboard/project/ayegqbmllkrbmkbxhkw/sql/1d57132e-04ec-4856-b003-655932637454

Inicio sesión en tu c...  Bookmarks  Aplicaciones  Diseño sin título  YouTube  Correo: Castellanos...  Aulas Virtuales Uni...  Google Académico  main.py - taller pyth...

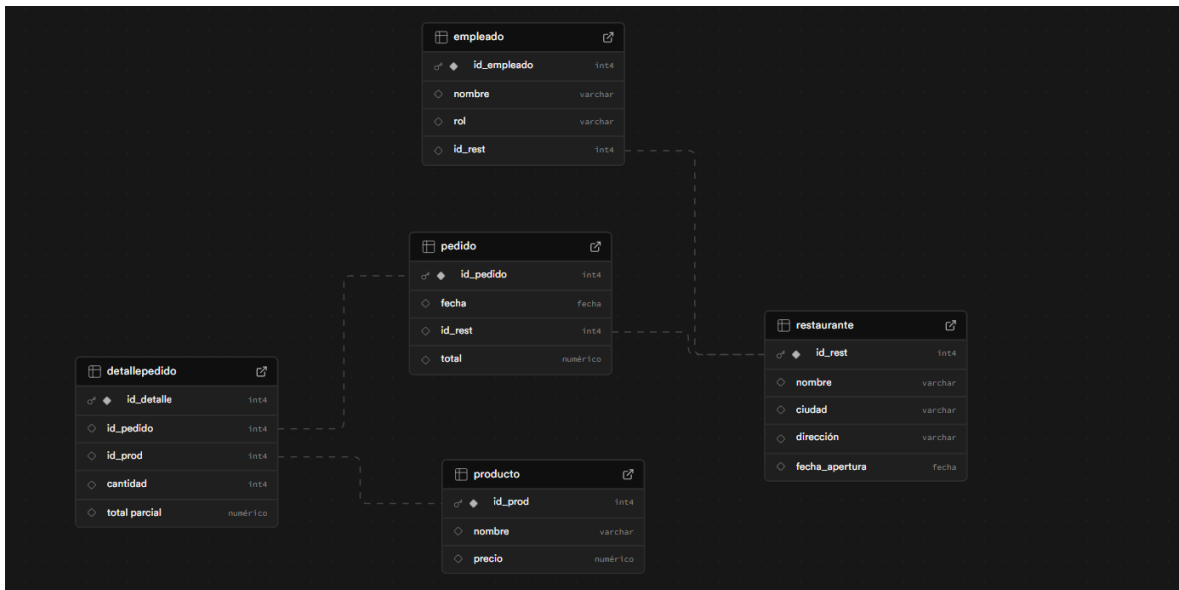
Organización de Andrescasarias  Gratis  Parcial  Conectar  Habilitar ramificación

Editor de SQL

Plantillas
Guías De Inicio Rápido
COMPARTIDO
FAVORITOS
PRIVATE (1)
Información del restaurante

1 INSERTAR EN detalle_pedido ( id_detalle , id_pedido , id_prod , cantidad , subtotal ) VALORES
2 ( 1 , 1 , 1 , 2 , 20000 ) ,
3 ( 2 , 1 , 6 , 1 , 10000 ) ,
4 ( 3 , 1 , 9 , 2 , 8000 ) ,
5 ( 4 , 2 , 2 , 1 , 10000 ) ,
6 ( 5 , 2 , 4 , 1 , 8000 ) ,
7 ( 6 , 2 , 15 , 3 , 9000 ) ,
8 ( 7 , 3 , 3 , 2 , 50000 ) ,
9 ( 8 , 3 , 7 , 1 , 14000 ) ,
10 ( 9 , 3 , 12 , 2 , 12000 ) ,
11 ( 10 , 3 , 25 , 1 , 22000 ) ,
12 ( 11 , 4 , 5 , 2 , 20000 ) ,
13 ( 12 , 4 , 10 , 2 , 6000 ) ,
14 ( 13 , 4 , 16 , 1 , 7000 ) ,
15 ( 14 , 5 , 1 , 1 , 22000 ) ,
16 ( 15 , 5 , 8 , 2 , 10000 ) ,
17 ( 16 , 5 , 20 , 1 , 7500 ) ,
18 ( 17 , 5 , 35 , 1 , 8000 ) ,
19 ( 18 , 5 , 42 , 1 , 12000 ) ,
20 ( 19 , 6 , 2 , 2 , 30000 ) ,
21 ( 20 , 6 , 11 , 3 , 10500 ) ,
22 ( 21 , 6 , 10 , 1 , 9000 ) ,
23 ( 22 , 7 , 3 , 1 , 25000 ) ,
24 ( 23 , 7 , 6 , 2 , 50000 ) ,
25 ( 24 , 7 , 16 , 1 , 2500 ) ,
26 ( 25 , 7 , 28 , 1 , 17000 ) ,
27 ( 26 , 7 , 39 , 1 , 10000 ) ,
28 ( 27 , 8 , 5 , 2 , 10000 ) ,
29 ( 28 , 8 , 9 , 3 , 12000 ) ,
30 ( 29 , 8 , 22 , 1 , 7000 ) ,
31 ( 30 , 8 , 32 , 1 , 3500 ) ,
32 ( 31 , 9 , 5 , 1 , 10000 ) ,
33 ( 32 , 9 , 12 , 2 , 12000 ) ,
34 ( 33 , 9 , 24 , 1 , 12000 ) ,
35 ( 34 , 9 , 37 , 1 , 7000 ) ,
36 ( 35 , 9 , 45 , 1 , 15000 ) ,
37 ( 36 , 10 , 1 , 3 , 30000 ) ,
38 ( 37 , 10 , 10 , 1 , 9000 ) ,
```

- Modelo desde supabase de las tablas



- Conexión a la base de datos que hacemos

```

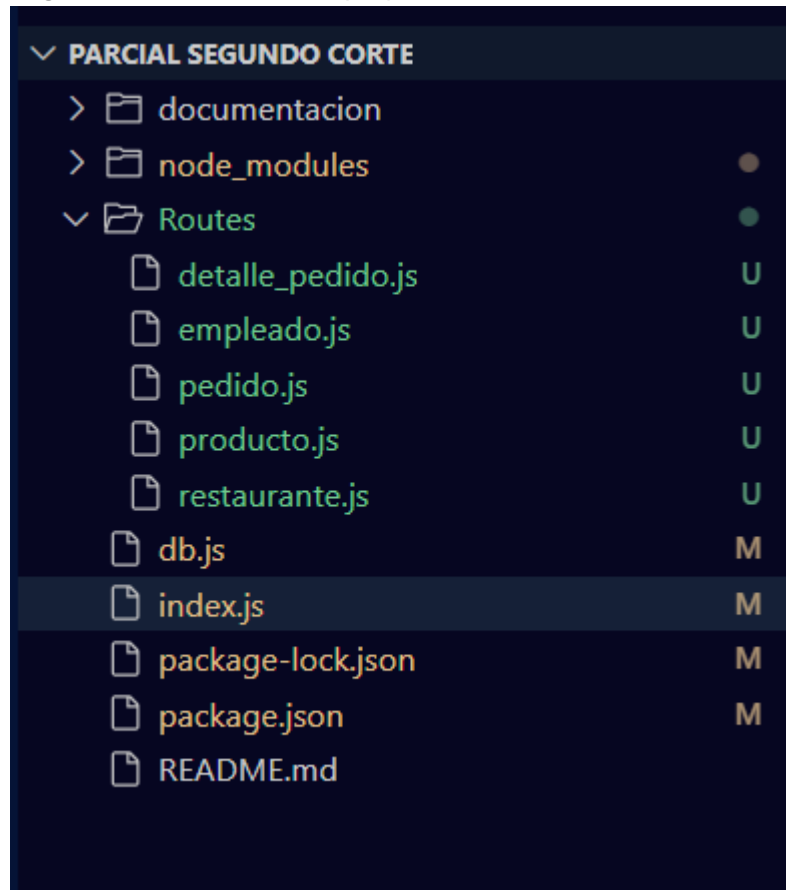
1 // db.js
2 const { Client } = require('pg');
3
4 const client = new Client({
5   host: 'aws-0-us-east-1.pooler.supabase.com',
6   port: 5432,
7   user: 'postgres.ayegqbml1krbmkbxhkwa',
8   password: '1002527276',
9   database: 'postgres',
10  ssl: {
11    rejectUnauthorized: false
12  },
13 });
14
15 client.connect()
16   .then(() => console.log('Conectado a PostgreSQL con pg'))
17   .catch(err => {
18     console.error('Error al conectar a la base de datos:', err.message);
19     process.exit(1);
20   });
21
22 module.exports = client;
23
24

```


Verificamos que la conexión este bien

```
PS C:\Users\Andres castellanos\Desktop\parcial segundo corte> node index.js
Servidor escuchando en http://localhost:5000
Conectado a PostgreSQL con pg
```

Organizamos las rutas del proyecto



- Creación Index.js

```
index.js > PORT
const express = require('express');
const cors = require('cors');
const app = express();
const PORT = 5000;

app.use(cors());
app.use(express.json());
app.use(express.urlencoded({ extended: true }));

// Conexión a rutas
app.use('/api', require('./rutas/restaurante'));
app.use('/api', require('./rutas/empleado'));
app.use('/api', require('./rutas/producto'));
app.use('/api', require('./rutas/pedido'));
app.use('/api', require('./rutas/detalle'));

// Ruta de prueba
app.get('/api/prueba', (req, res) => {
  res.send('API funcionando correctamente 🚀');
});

app.listen(PORT, () => {
  console.log(`Servidor corriendo en http://localhost:${PORT}/api`);
});
```

- Creación de api restaurante

```
1 const express = require('express');
2 const router = express.Router();
3 const client = require('../bd');
4
5 // Crear restaurante
6 router.post('/restaurante/insertar', (req, res) => {
7   const { id_rest, nombre, ciudad, direccion, fecha_apertura } = req.body;
8   const query = 'INSERT INTO restaurante (id_rest, nombre, ciudad, direccion, fecha_apertura) VALUES ($1, $2, $3, $4, $5)';
9   client.query(query, [id_rest, nombre, ciudad, direccion, fecha_apertura])
10    .then(() => res.status(201).json({ mensaje: "Restaurante creado con éxito" }))
11    .catch(error => res.status(500).json({ error: error.message }));
12 });
13
14 // Obtener todos Los restaurantes
15 router.get('/restaurante/obtener', (req, res) => {
16   client.query('SELECT * FROM restaurante')
17   .then(result => res.status(200).json({ data: result.rows }))
18   .catch(error => res.status(500).json({ error: error.message }));
19 });
20
21 // Actualizar restaurante
22 router.put('/restaurante/actualizar/:id', (req, res) => {
23   const { id } = req.params;
24   const { nombre, ciudad, direccion, fecha_apertura } = req.body;
25   const query = 'UPDATE restaurante SET nombre=$1, ciudad=$2, direccion=$3, fecha_apertura=$4 WHERE id_rest=$5';
26   client.query(query, [nombre, ciudad, direccion, fecha_apertura, id])
27   .then(result => {
28     if (result.rowCount === 0) {
29       return res.status(404).json({ mensaje: "Restaurante no encontrado" });
30     }
31   });
32 });
```

- Creación de api pedido

```
db.js M index.js M restaurante.js U pedido.js U x producto.js U detalle_pedido.js U
Routes > pedido.js > ...
1  const express = require('express');
2  const router = express.Router();
3  const client = require('../bd');
4
5  router.post('/pedido/insertar', (req, res) => {
6    const { id_pedido, fecha, id_rest, total } = req.body;
7    const query = 'INSERT INTO pedido (id_pedido, fecha, id_rest, total) VALUES ($1, $2, $3, $4)';
8    client.query(query, [id_pedido, fecha, id_rest, total])
9      .then(() => res.status(201).json({ mensaje: "Pedido creado con éxito" }))
10     .catch(error => res.status(500).json({ error: error.message }));
11  });
12
13  router.get('/pedido/obtener', (req, res) => {
14    client.query('SELECT * FROM pedido')
15      .then(result => res.status(200).json({ data: result.rows }))
16      .catch(error => res.status(500).json({ error: error.message }));
17  });
18
19  router.put('/pedido/actualizar/:id', (req, res) => {
20    const { id } = req.params;
21    const { fecha, id_rest, total } = req.body;
22    const query = 'UPDATE pedido SET fecha=$1, id_rest=$2, total=$3 WHERE id_pedido=$4';
23    client.query(query, [fecha, id_rest, total, id])
24      .then(result => {
25        if (result.rowCount === 0) {
26          return res.status(404).json({ mensaje: "Pedido no encontrado" });
27        }
28        res.status(200).json({ mensaje: "Pedido actualizado con éxito" });
29      })
30  });
```

- Creación de api producto

```
Routes > producto.js > ...
1  const express = require('express');
2  const router = express.Router();
3  const client = require('../bd');
4
5  router.post('/producto/insertar', (req, res) => {
6    const { id_prod, nombre, precio } = req.body;
7    const query = 'INSERT INTO producto (id_prod, nombre, precio) VALUES ($1, $2, $3)';
8    client.query(query, [id_prod, nombre, precio])
9      .then(() => res.status(201).json({ mensaje: "Producto creado con éxito" }))
10     .catch(error => res.status(500).json({ error: error.message }));
11  });
12
13  router.get('/producto/obtener', (req, res) => {
14    client.query('SELECT * FROM producto')
15      .then(result => res.status(200).json({ data: result.rows }))
16      .catch(error => res.status(500).json({ error: error.message }));
17  });
18
19  router.put('/producto/actualizar/:id', (req, res) => {
20    const { id } = req.params;
21    const { nombre, precio } = req.body;
22    const query = 'UPDATE producto SET nombre=$1, precio=$2 WHERE id_prod=$3';
23    client.query(query, [nombre, precio, id])
24      .then(result => {
25        if (result.rowCount === 0) {
26          return res.status(404).json({ mensaje: "Producto no encontrado" });
27        }
28        res.status(200).json({ mensaje: "Producto actualizado con éxito" });
29      })
30  });
```

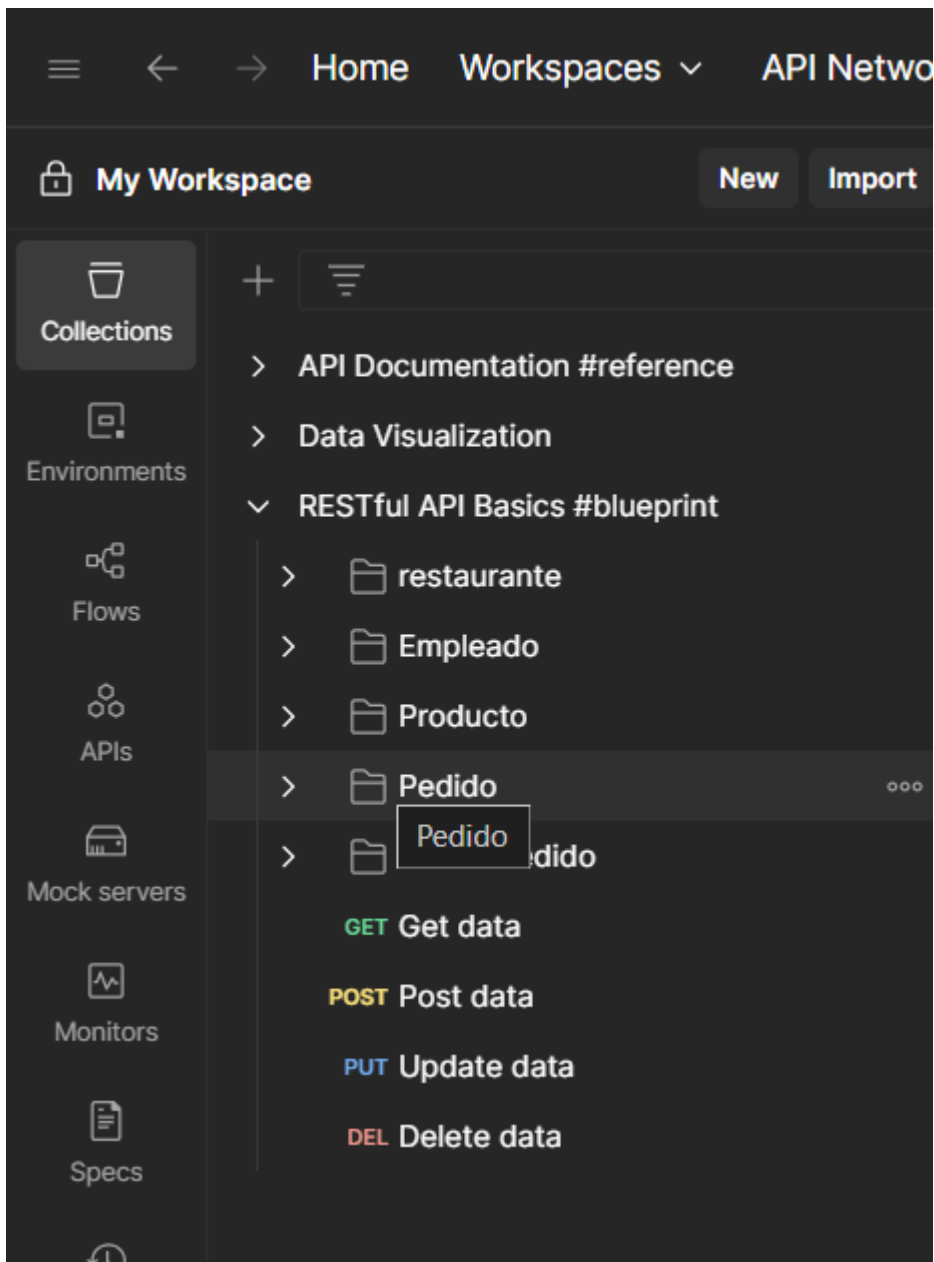
- Creación de api detalle_pedido

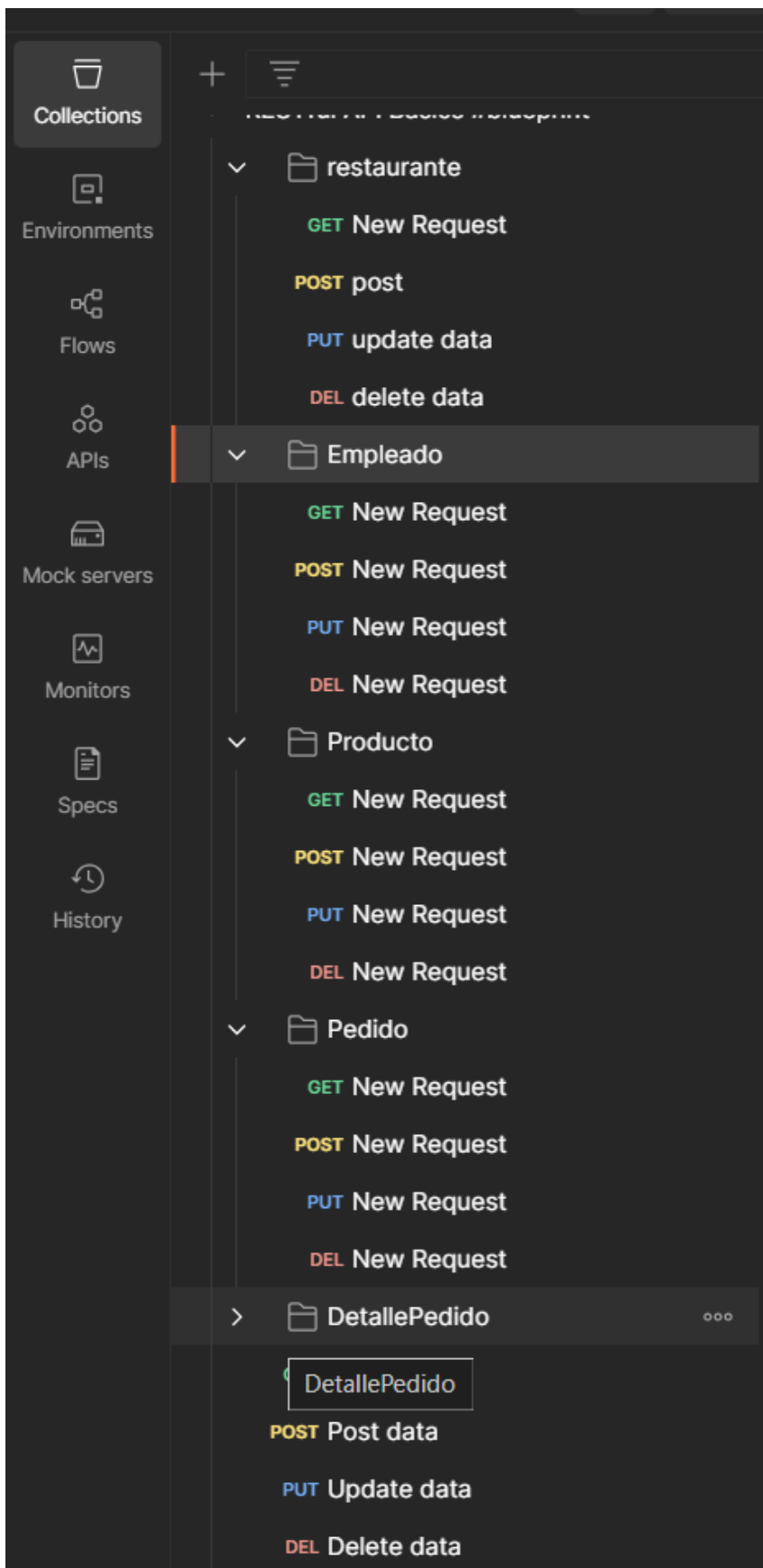
```
Routes > detalle_pedido.js > ...
1  const express = require('express');
2  const router = express.Router();
3  const client = require('../bd');
4
5  router.post('/detalle/insertar', (req, res) => {
6    const { id_detalle, id_pedido, id_prod, cantidad, subtotal } = req.body;
7    const query = 'INSERT INTO detalle_pedido (id_detalle, id_pedido, id_prod, cantidad, subtotal) VALUES ($1, $2, $3, $4, $5)';
8    client.query(query, [id_detalle, id_pedido, id_prod, cantidad, subtotal])
9      .then(() => res.status(201).json({ mensaje: "Detalle de pedido creado con éxito" }))
10     .catch(error => res.status(500).json({ error: error.message }));
11  });
12
13  router.get('/detalle/obtener', (req, res) => {
14    client.query('SELECT * FROM detalle_pedido')
15      .then(result => res.status(200).json({ data: result.rows }))
16      .catch(error => res.status(500).json({ error: error.message }));
17  });
18
19  router.put('/detalle/actualizar/:id', (req, res) => {
20    const { id } = req.params;
21    const { id_pedido, id_prod, cantidad, subtotal } = req.body;
22    const query = 'UPDATE detalle_pedido SET id_pedido=$1, id_prod=$2, cantidad=$3, subtotal=$4 WHERE id_detalle=$5';
23    client.query(query, [id_pedido, id_prod, cantidad, subtotal, id])
24      .then(result => {
25        if (result.rowCount === 0) {
26          return res.status(404).json({ mensaje: "Detalle no encontrado" });
27        }
28        res.status(200).json({ mensaje: "Detalle actualizado con éxito" });
29      })
30  });
```

- Creación de api empleados

```
Routes > empleado.js > <unknown>
1  const express = require('express');
2  const router = express.Router();
3  const client = require('../bd');
4
5  router.post('/empleado/insertar', (req, res) => {
6    const { id_empleado, nombre, rol, id_rest } = req.body;
7    const query = 'INSERT INTO empleado (id_empleado, nombre, rol, id_rest) VALUES ($1, $2, $3, $4)';
8    client.query(query, [id_empleado, nombre, rol, id_rest])
9      .then(() => res.status(201).json({ mensaje: "Empleado creado con éxito" }))
10     .catch(error => res.status(500).json({ error: error.message }));
11  });
12
13  router.get('/empleado/obtener', (req, res) => {
14    client.query('SELECT * FROM empleado')
15      .then(result => res.status(200).json({ data: result.rows }))
16      .catch(error => res.status(500).json({ error: error.message }));
17  });
18
19  router.put('/empleado/actualizar/:id', (req, res) => {
20    const { id } = req.params;
21    const { nombre, rol, id_rest } = req.body;
22    const query = 'UPDATE empleado SET nombre=$1, rol=$2, id_rest=$3 WHERE id_empleado=$4';
23    client.query(query, [nombre, rol, id_rest, id])
24      .then(result => {
25        if (result.rowCount === 0) {
```

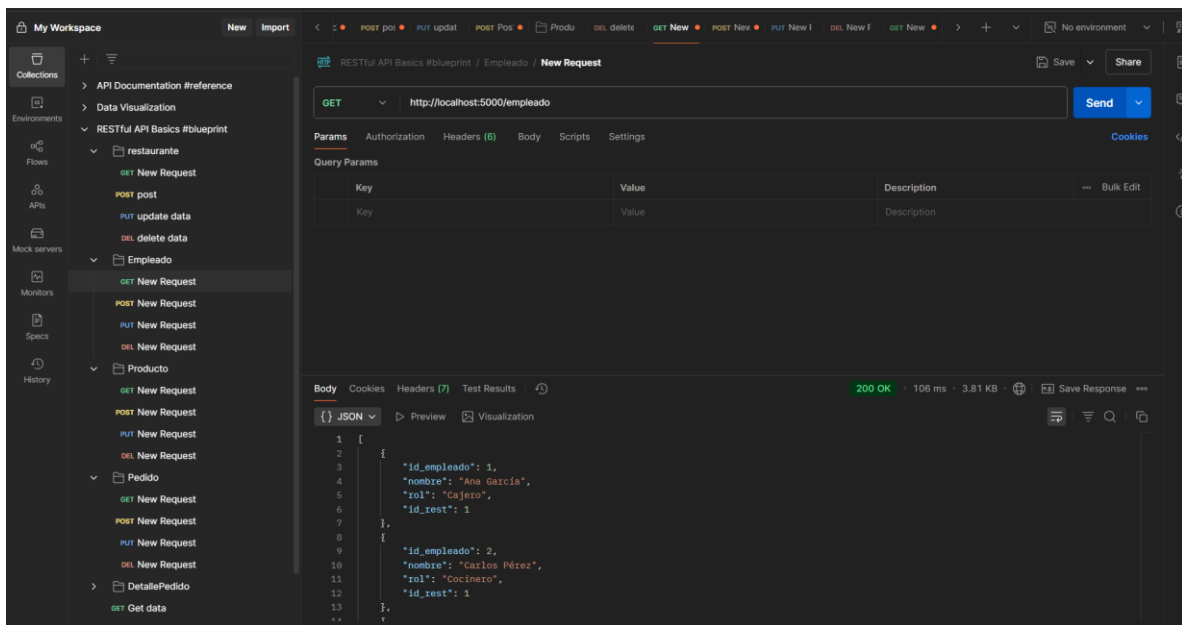
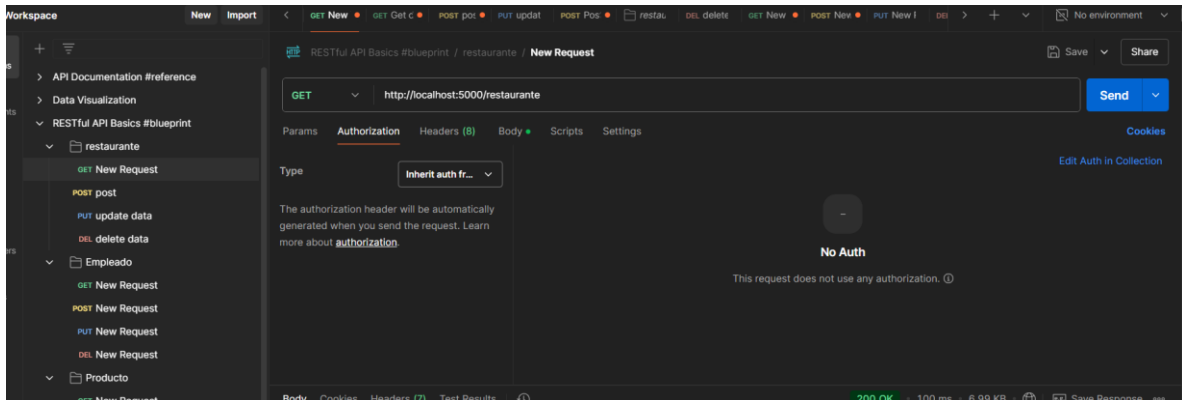
- Ordenamos en postman para cada api hacemos unas peticiones





Peticiones

1.get de cada api



work

Search Postman Ctrl K

Invite Upgrade

RESTful API Basics #blueprint / Pedido / New Request

GET http://localhost:5000/pedido Send

Params Authorization Headers (6) Body Scripts Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

1

Body Cookies Headers (7) Test Results

200 OK 92 ms 4.32 KB

JSON Preview Visualization

```
1 [
2   {
3     "id_pedido": 1,
4     "fecha": "2023-01-25T05:00:00.000Z",
5     "id_rest": 1,
6     "total": "55000.00"
7   },
8   {
9     "id_pedido": 2,
10    "fecha": "2023-01-25T05:00:00.000Z",
11    "id_rest": 2,
12    "total": "32000.00"
13  },
14 ]
```

RESTful API Basics #blueprint / Producto / New Request

GET http://localhost:5000/producto Send

Params Authorization Headers (6) Body Scripts Settings

Query Params

Key	Value	Description
Key	Value	Description

Body Cookies Headers (7) Test Results

200 OK 94 ms 3.12 KB

JSON Preview Visualization

```
1 [
2   {
3     "id_prod": 1,
4     "nombre": "Hamburguesa Clásica",
5     "precio": "12000.00"
6   },
7   {
8     "id_prod": 2,
9     "nombre": "Pizza Margarita",
10    "precio": "15000.00"
11  },
12  {
13    "id_prod": 3,
14    "nombre": "Sushi Variado"
15  }
16 ]
```