**Parcial Bases de Datos Api**

**Edwin Camilo Rodriguez Arredondo**

**ID:841424**

**Bases de Datos Masiva**

**6° Semestre Ingeniería de Sistemas**

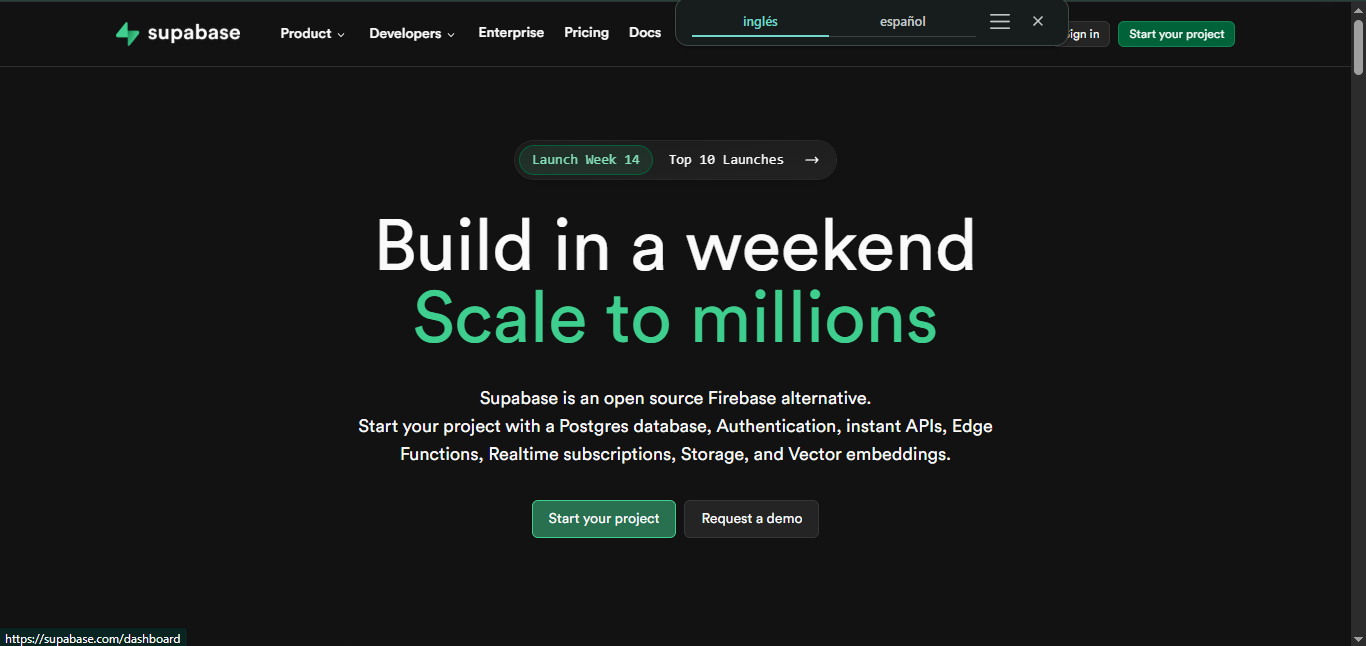
**Ingeniería de Sistemas**

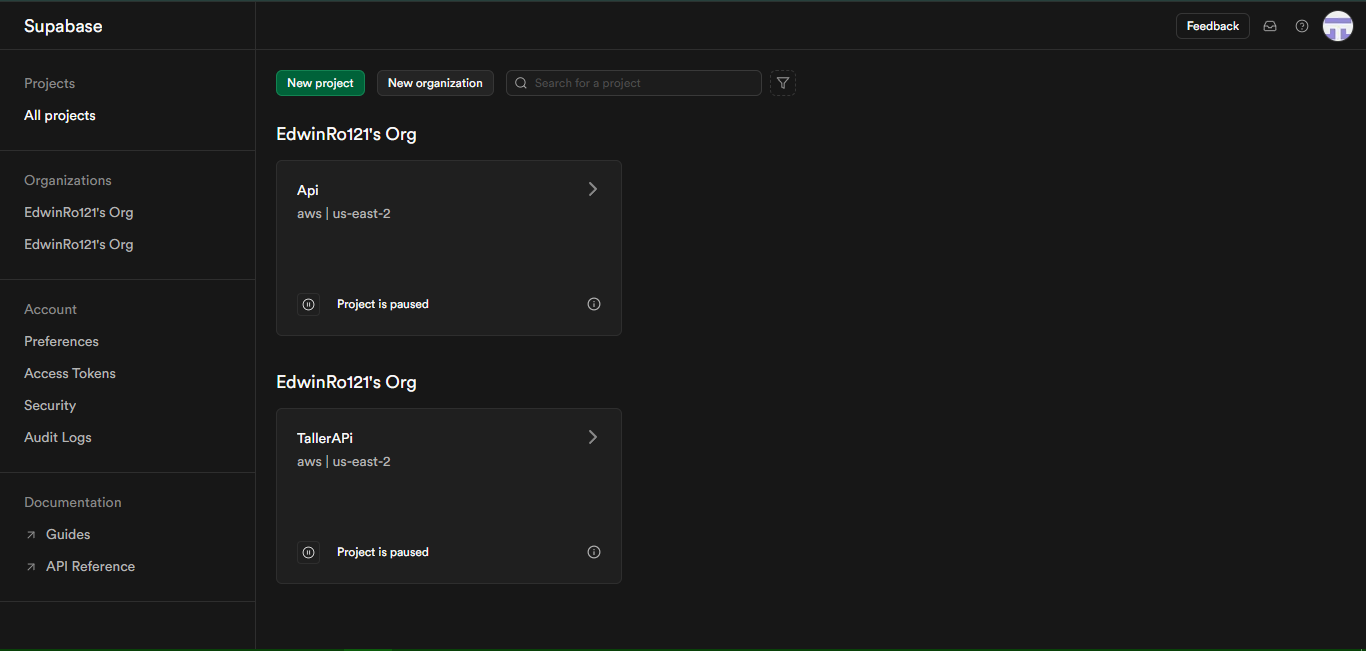
**Universidad UNIMINUTO**

**23 de abr. de 25**

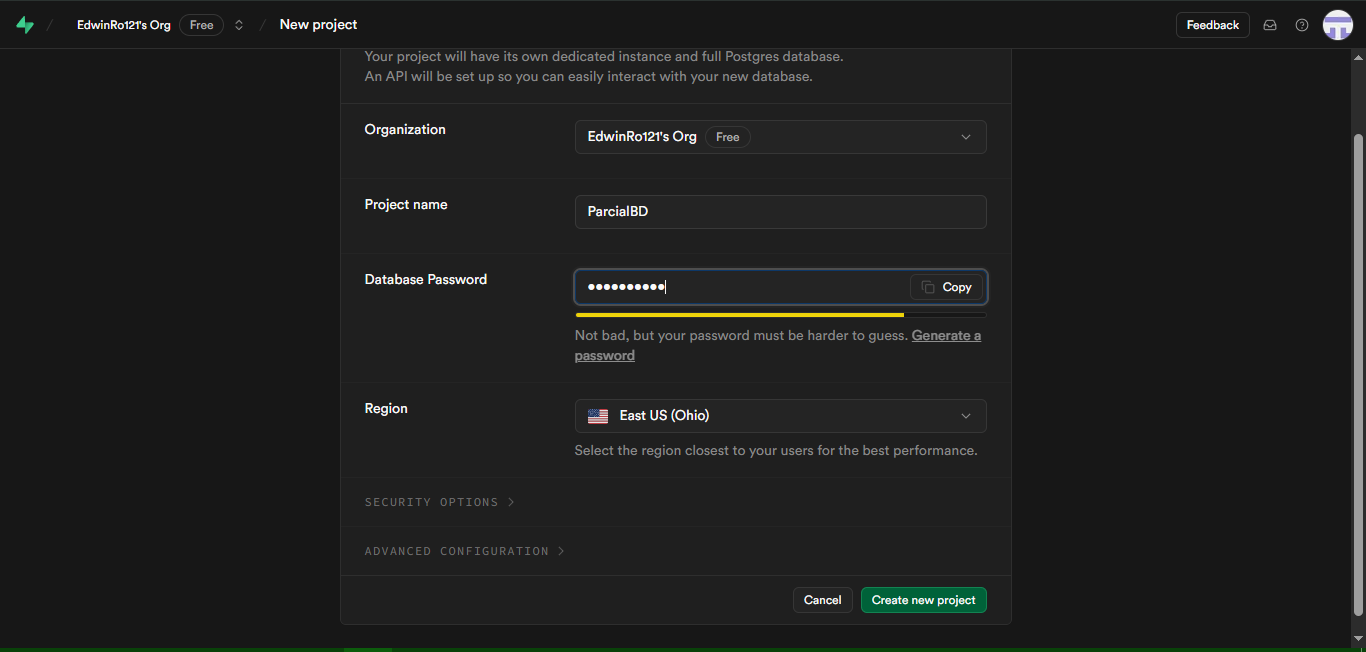
**Zipaquirá Cundinamarca**

* Se ingresa a supabase y se crar un nuevo proyecto

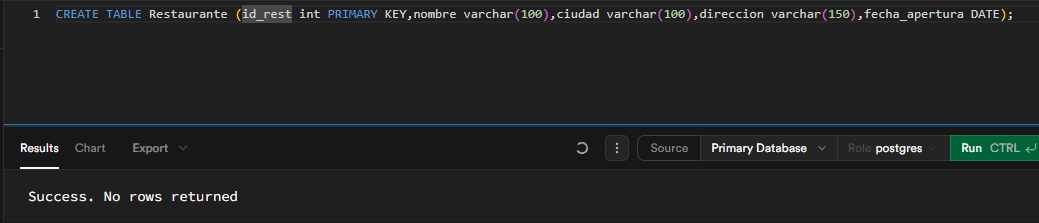


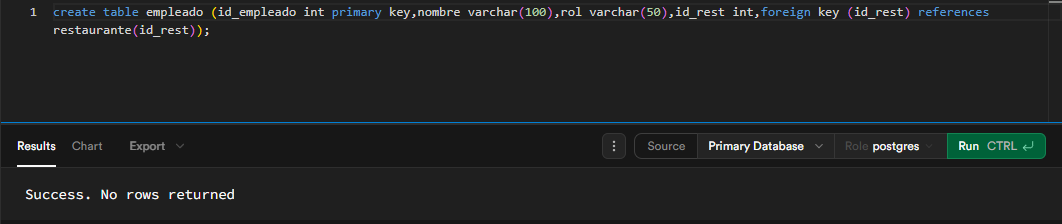


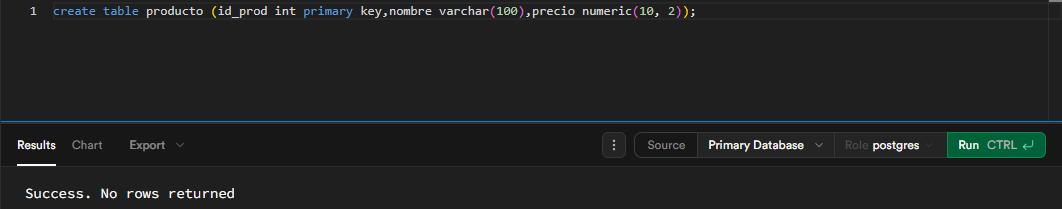
* Se asigna el nombre y contraseña

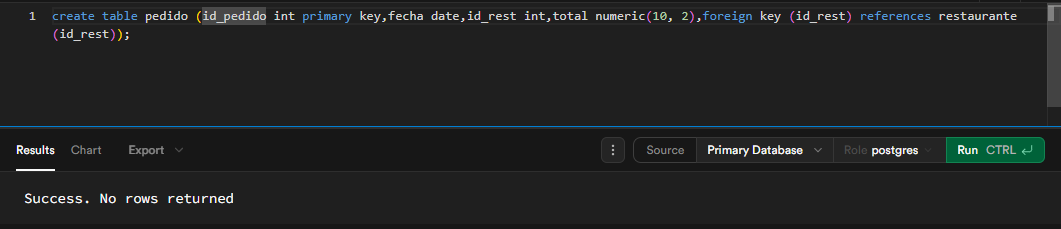


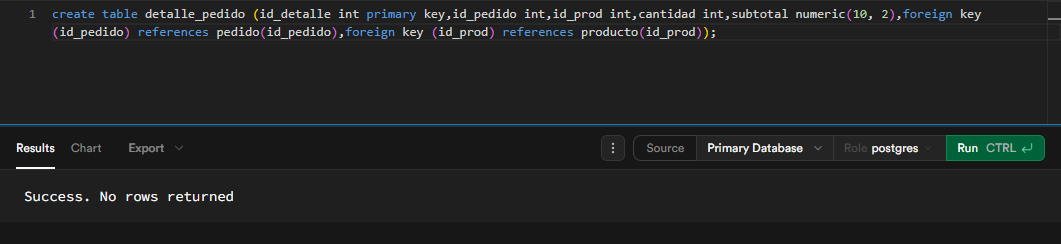
* Se crean las tablas con sus respectivos parámetros y sus respectivas relaciones



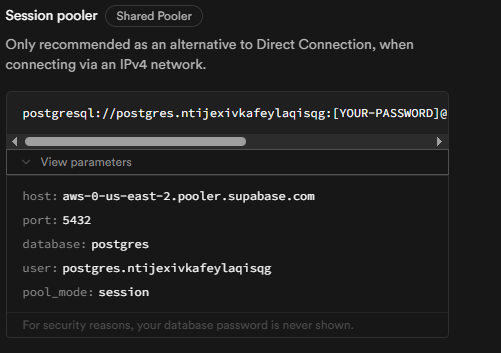




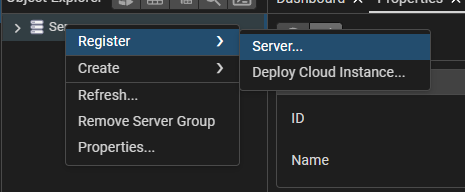




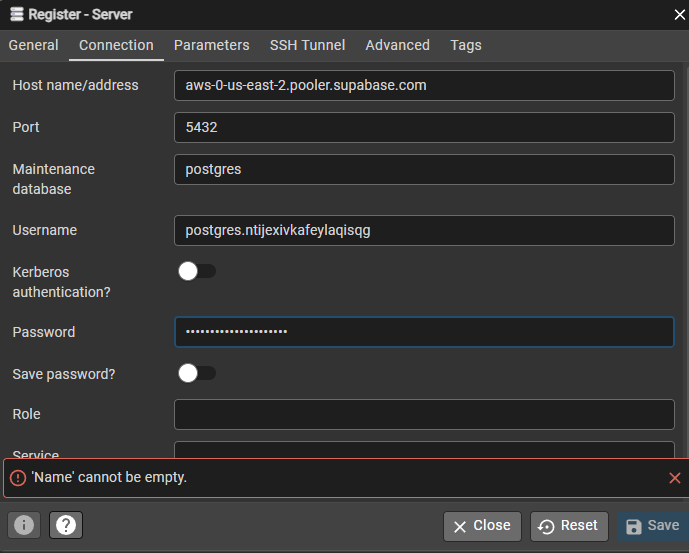
* Se miran los parámetros de la conexión

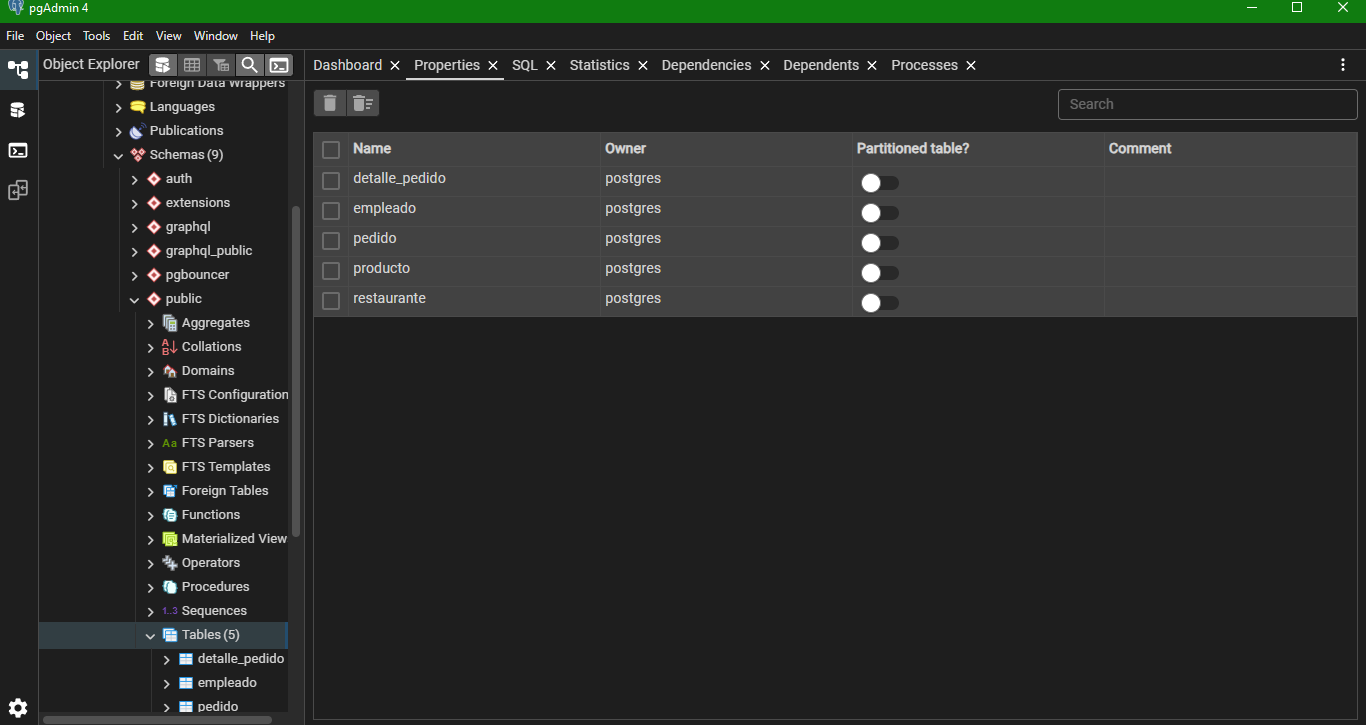


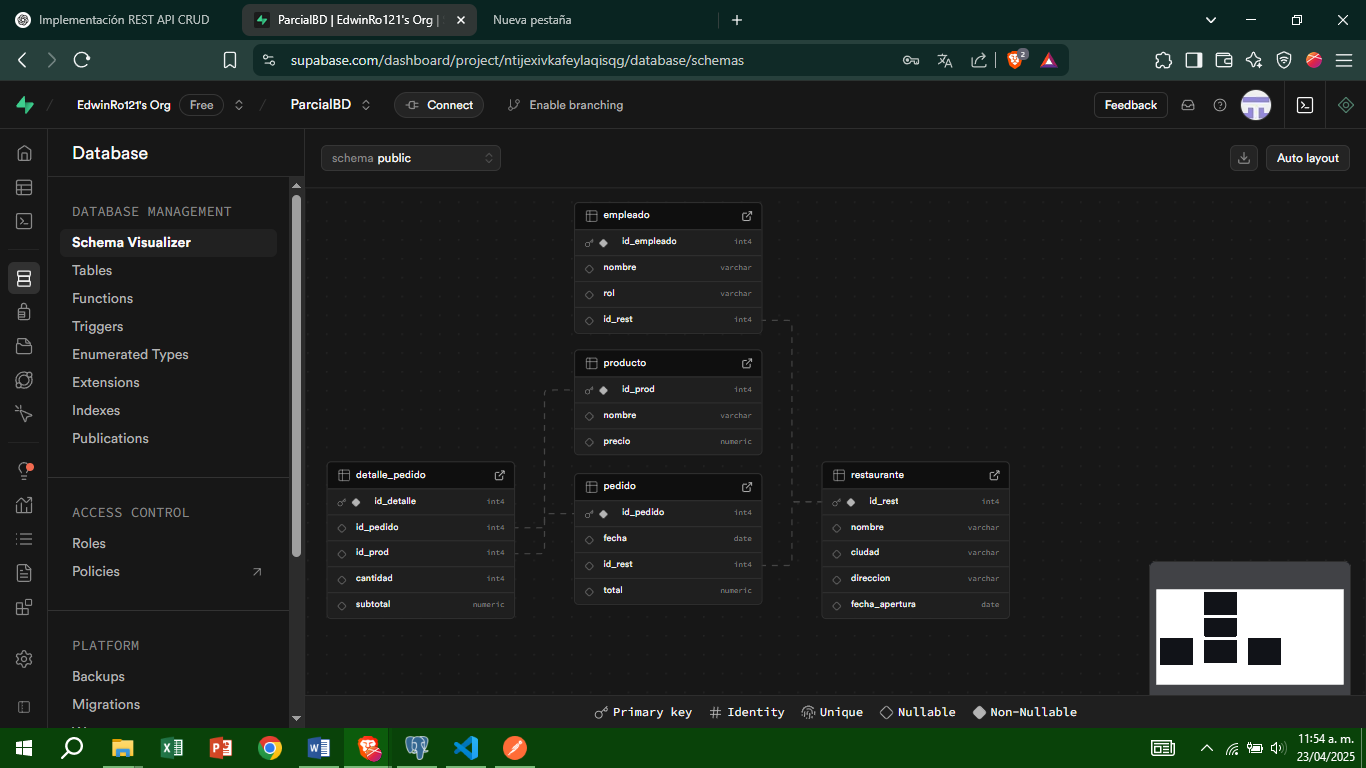
* Se crea un nuevo servidor en pgadmin4



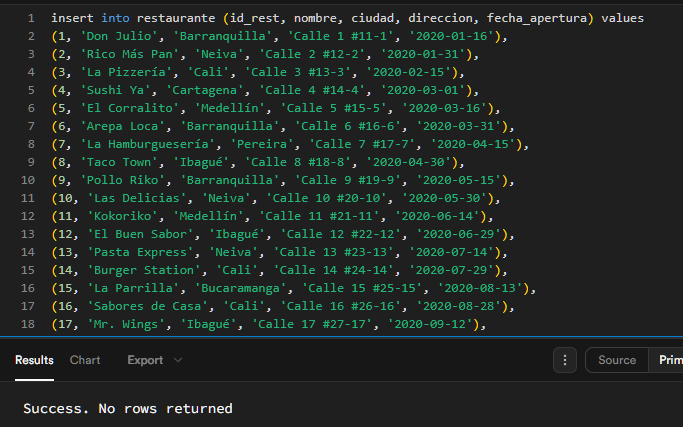
* Se insertan los parámetros de conexión de supabase



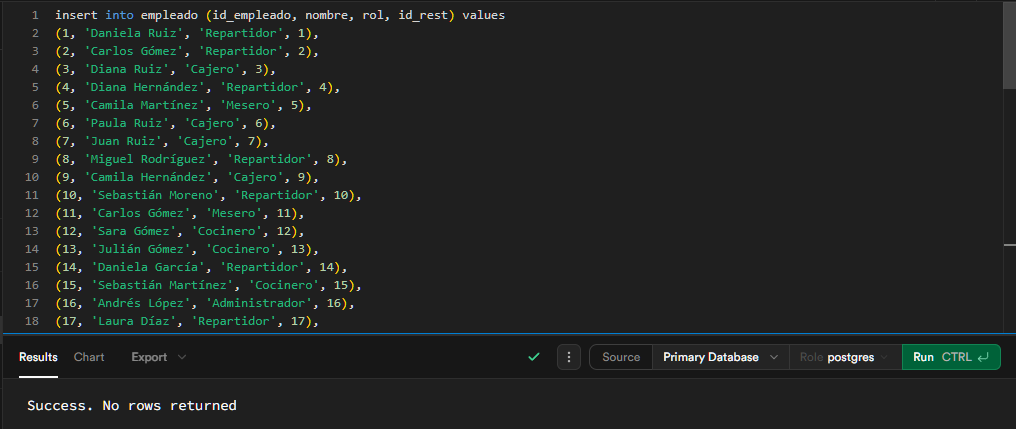




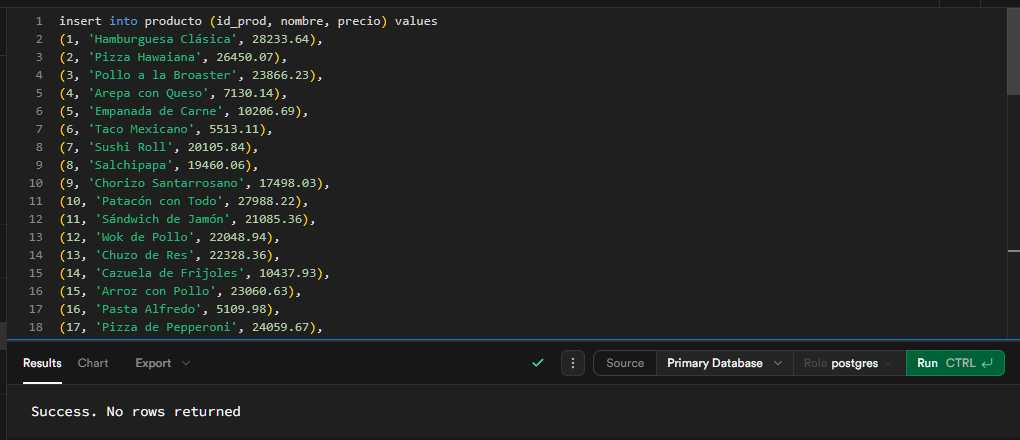
* Se insertan los 50 registros de restaurante



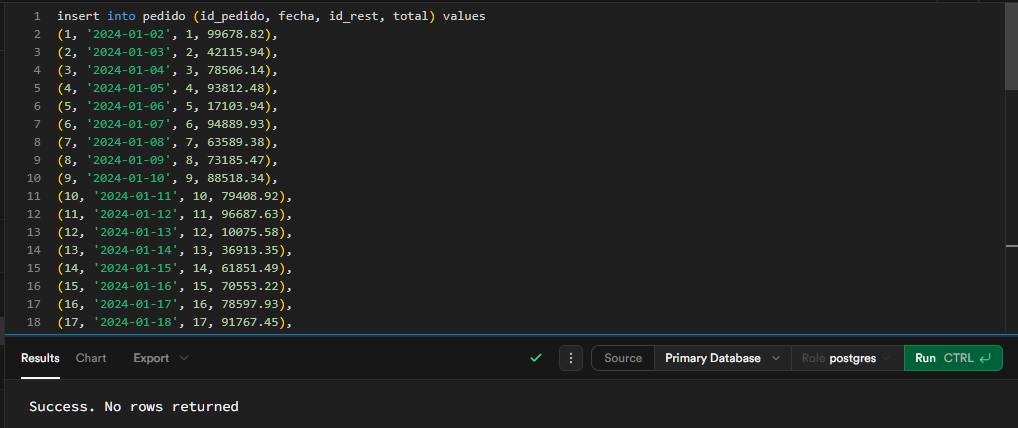
* Se insertan los 50 de empleado



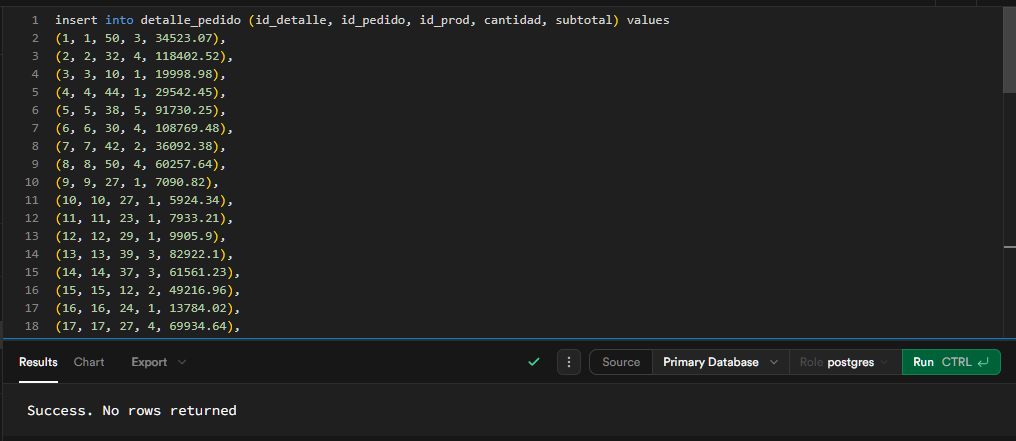
* Se insertan los 50 registros de producto



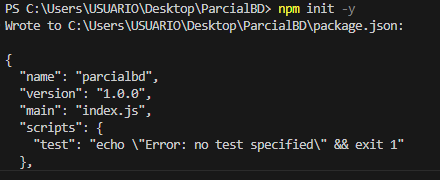
* Se insertan 50 registros de pedido



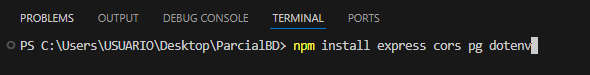
* Se insertan 50 registros de detalle\_pedido



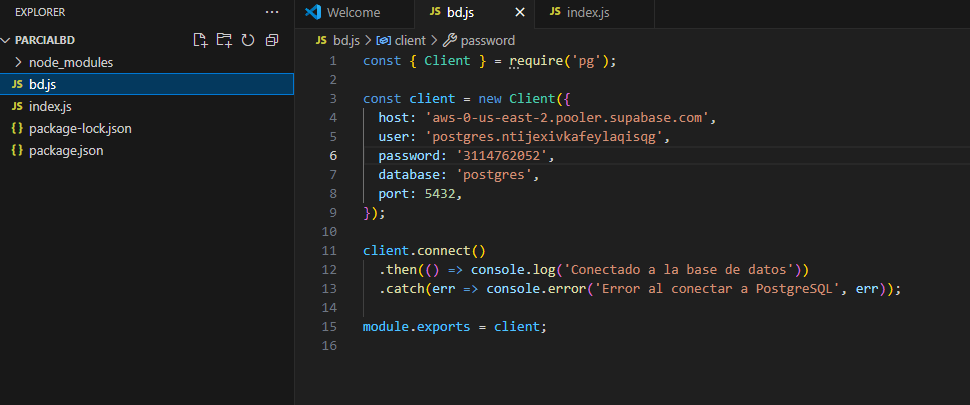
* Se abre el visual studio y se asigna una carpeta después se inicia un repositorio



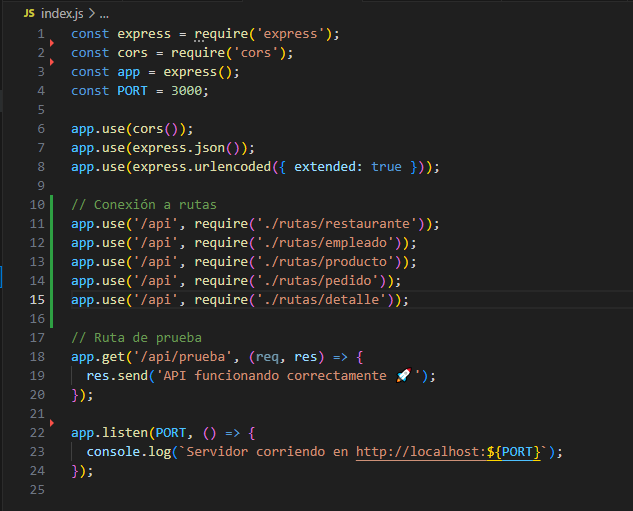
* Se instalan las respectivas las respectivas herramientas que usaremos



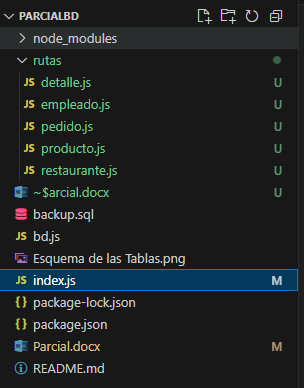
* Se crea un nuevo archivo y se crea la conexión con supabase



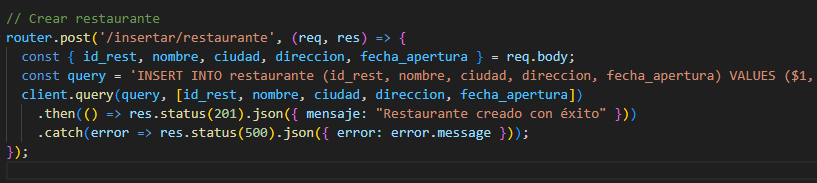
* Se crea otro archivo llamado index.js y se crea las conexiones de cada tabla con sus respectivas apis



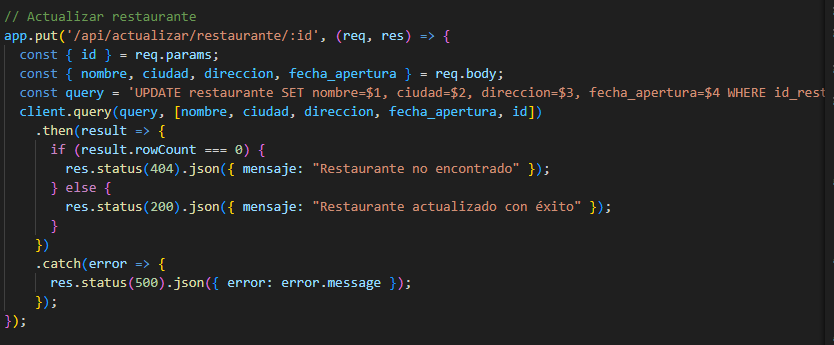
* Se crea un nuevo archivo para cada tabla



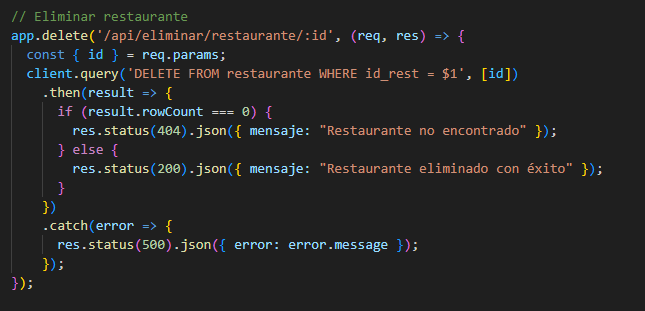
* Se crea el api de insertar



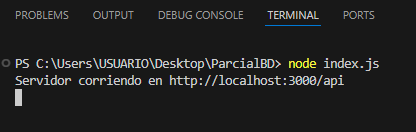
* app.post() crea una ruta POST.
* req.body obtiene los datos enviados por el cliente (en JSON).
* Se arma una consulta SQL con INSERT INTO para guardar el restaurante.
* Se usa client.query() (desde tu conexión a PostgreSQL) para ejecutar la consulta.
* Si todo va bien, responde con 201 (Created).
* Si falla, responde con 500 (Internal Server Error).
* GET para consultar todos los restaurantes.
* Ejecuta SELECT \* FROM restaurante para obtener todos los registros.
* result.rows es un array con todos los resultados.
* Responde con un objeto JSON que contiene la información
* Se crea la api de actualizar



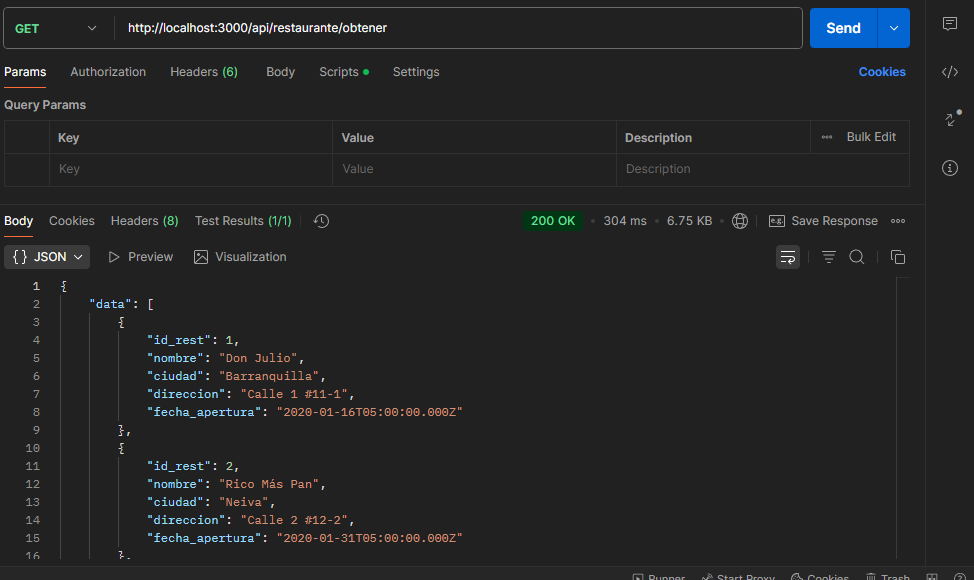
* Ruta PUT para actualizar un restaurante.
* Usa :id como parámetro en la URL req.params.id
* Actualiza los campos especificados en el body.
* rowCount verifica si se actualizó al menos un registro. Si no, muestra 404.
* se crea la api de eliminar



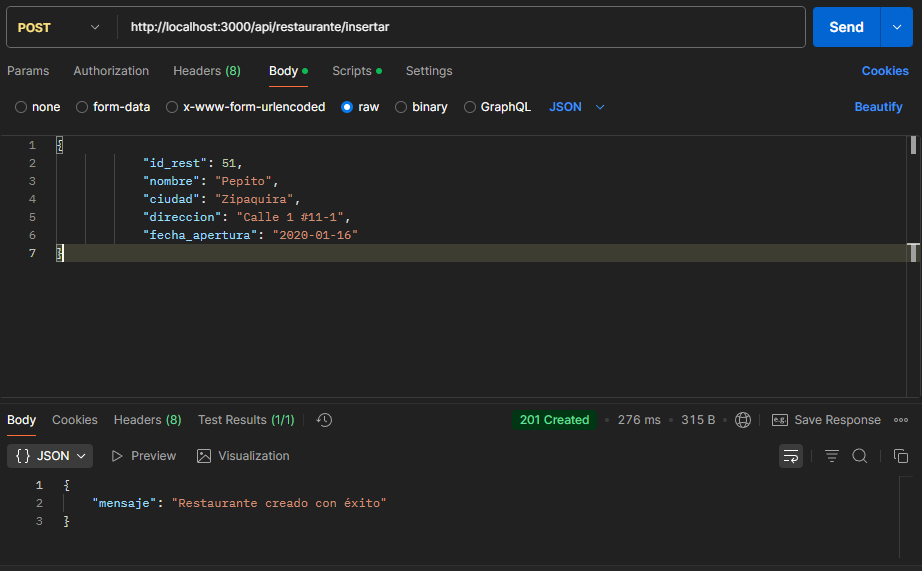
* Ruta DELETE para eliminar un restaurante por ID.
* Ejecuta DELETE FROM restaurante WHERE id\_rest = $1 ($1 significa el dato que se le ingrese por ejemplo 4 , Ejecuta DELETE FROM restaurante WHERE id\_rest = 4)
* Si no encuentra el restaurante, rowCount es 0, así que responde con 404
* Se corre el servidor



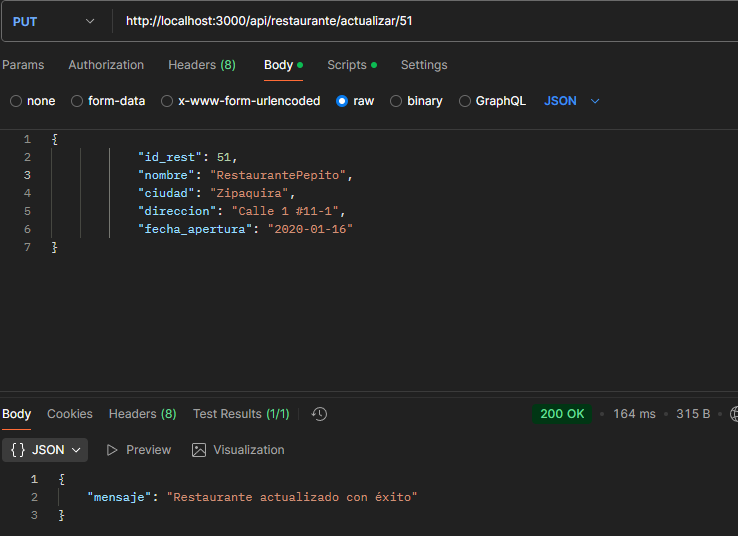
* Probamos las peticiones para comprobar que están funcionando correctamente haremos la prueba con la tabla de restaurante en la aplicación postman
* Se ejecuta en get para traer los registros con la api de obtener



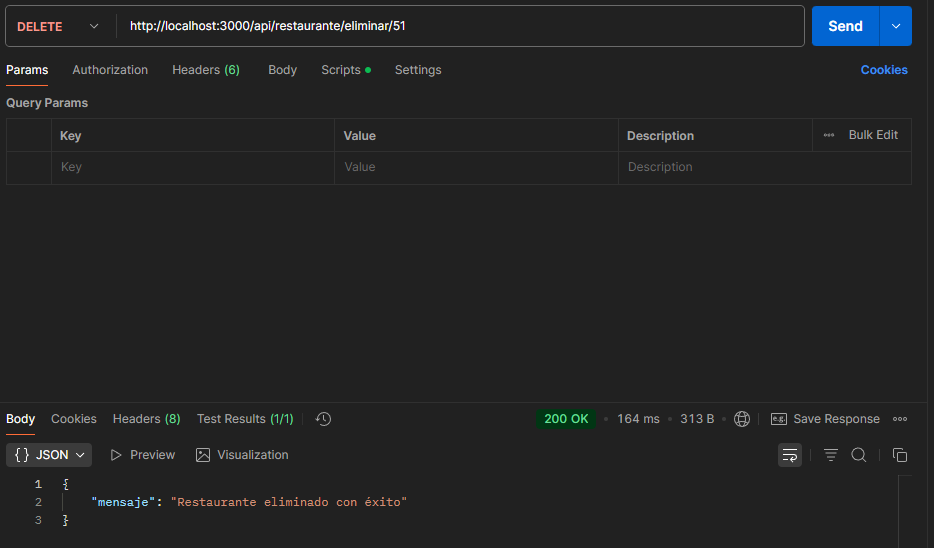
* Se usa el post para insertar un nuevo registro con la api de insertar , para esto toca colocar los parametos en el apartado de body



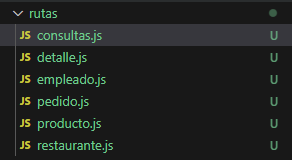
* Se usa put para modificar algún parámetro de un registro , para esto se tiene q escribir el parámetro que se desea cambiar con la api de actualizar



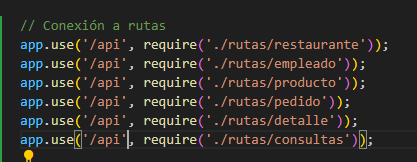
* Se usa del para eliminar un registro, para esto se tiene que ingresar la id del registro que se desea eliminar con la api de eliminar



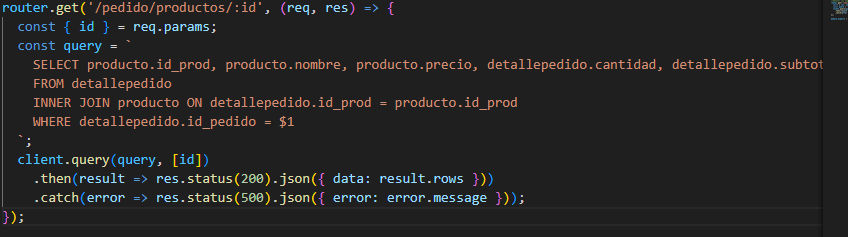
* Se crea el archivo de consultas



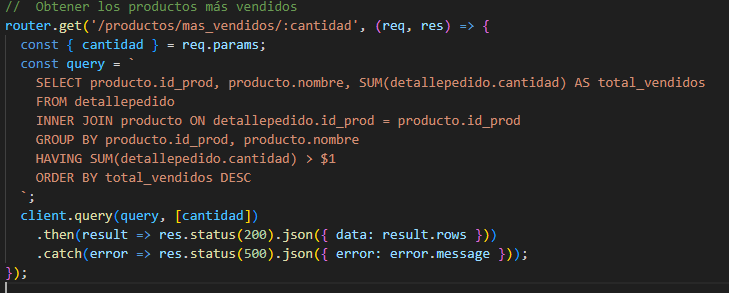
* Se agrega la conexión al archivo



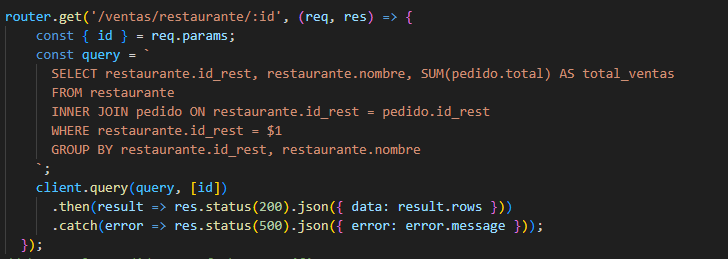
* S e crea la api para obtener productos de un pedido específico



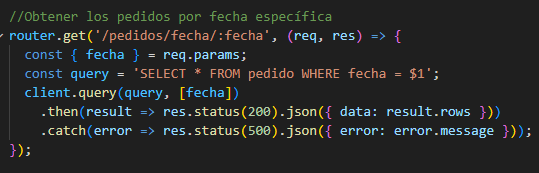
* router.get() crea una ruta GET
* req.params.id se pide el parámetro del ID del pedido desde la URL.
* INNER JOIN vincula productos con su detalle en el pedido haciendo una consulta SQL
* Se usa client.query() para ejecutar la consulta en PostgreSQL.
* Si va bien, responde con 200 OK y un JSON con los productos del pedido.
* Si ocurre un error, devuelve 500 Internal Server Error
* Se crea la api de Obtener los productos más vendidos



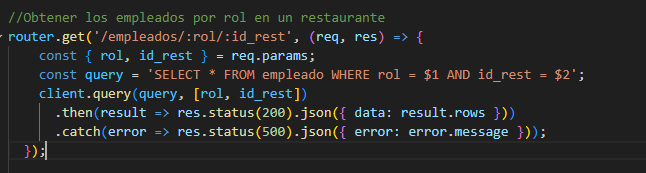
* router.get() se utiliza para usar la ruta de GET.
* req.params.cantidad indica que requiere el parámetro de cantidad ,el cual debe indicar el mínimo de unidades vendidas
* La consulta se agrupa por producto y suma cuántas veces se ha vendido.
* Solo se devuelven productos con ventas mayores a cantidad
* Si la consulta salió correctamente responde con el codigo 200 con los productos más vendidos, ordenados de mayor a menor.
* Si hay error, se devuelve 500.
* Se crea la api de Obtener el total de ventas por restaurante



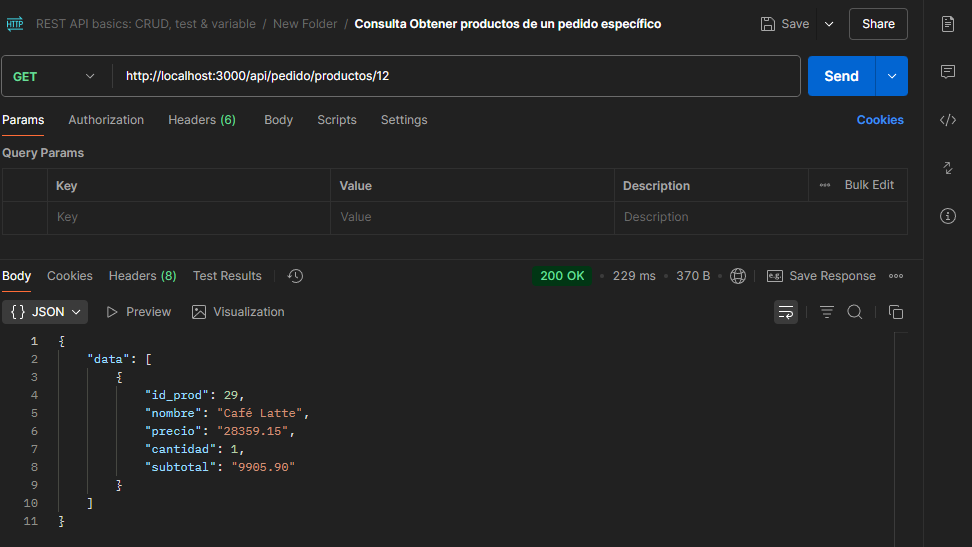
* req.params.id toma el ID del restaurante enviado en la URL.
* Suma (SUM) el total de todos los pedidos asociados a ese restaurante.
* Usa INNER JOIN para conectar restaurante con pedido.
* Agrupa por restaurante.id\_rest y nombre.
* client.query() ejecuta la consulta con el ID como parámetro.
* Si todo va bien, responde con 200 OK y el total de ventas.
* Si hay un error, responde con 500 Internal Server Error.
* Obtener los pedidos por fecha específica



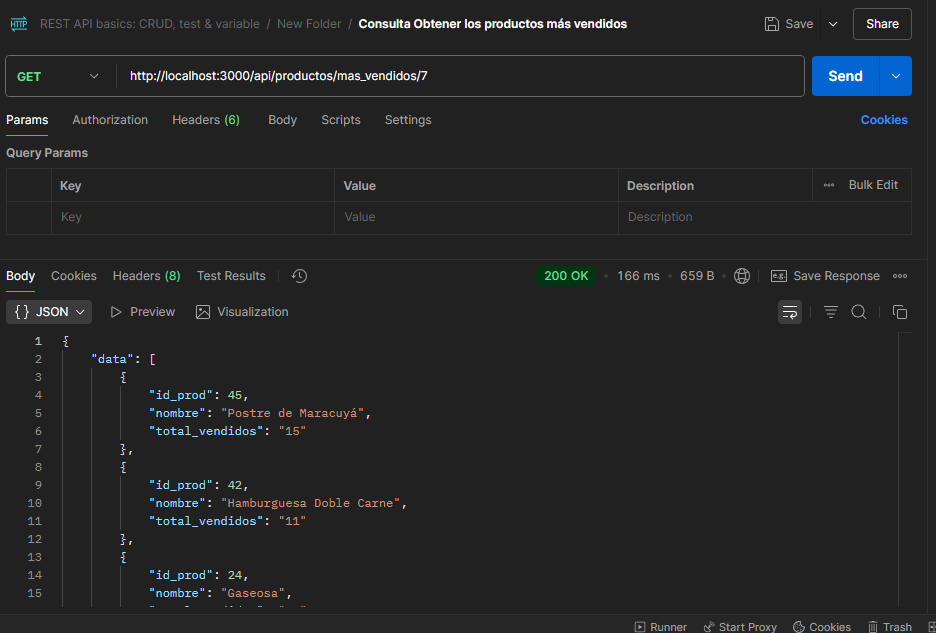
* Ruta GET con fecha como parámetro.
* Se buscan todos los pedidos con la fecha exacta proporcionada.
* Devuelve todos los campos de la tabla pedido.
* Respuesta 200 con los pedidos encontrados.
* Si falla, se devuelve 500.
* Obtener los empleados por rol en un restaurante



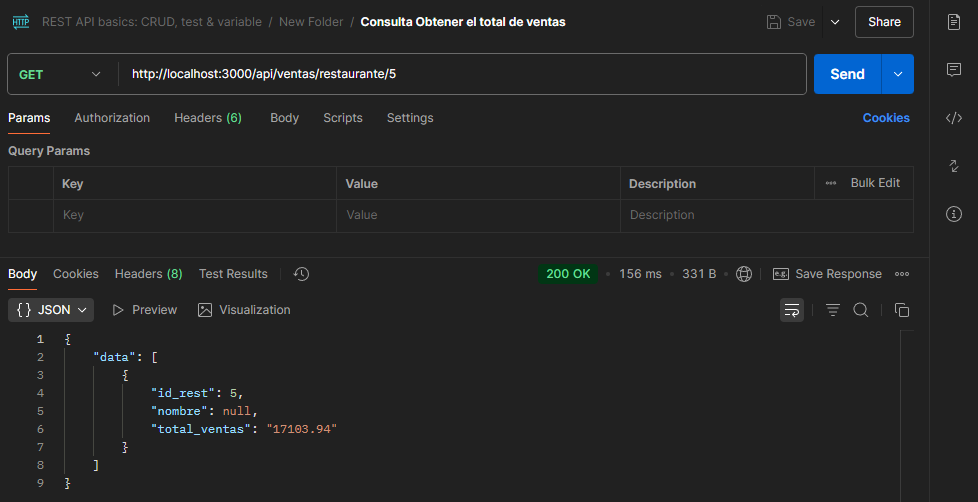
* router.get() define la ruta GET /empleados/:rol/:id\_rest.
* req.params.rol y req.params.id\_rest capturan el rol y restaurante de la URL.
* La consulta busca empleados que coincidan con ese rol y pertenezcan a ese restaurante.
* client.query() ejecuta la consulta con ambos valores.
* Si encuentra coincidencias, responde con 200 OK y los datos de los empleados.
* Si falla, responde con 500 Internal Server Error.
* Probamos las consultas para comprobar que están funcionando correctamente
* Consulta Obtener productos de un pedido especifico



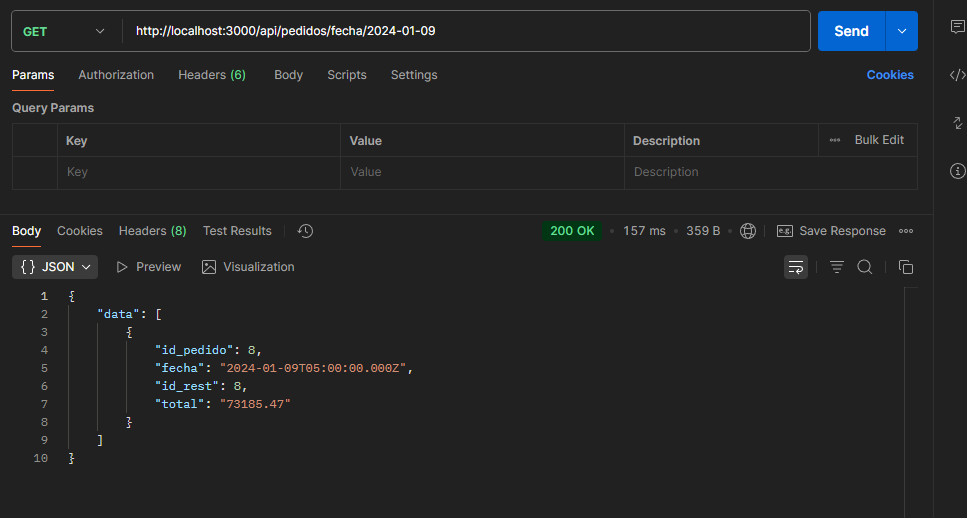
* Consulta obtener los productos mas vendidos



* Consulta obtener el total de ventas de un restaurante



* Consulta Obtener los pedidos por fecha específica



* Consulta Obtener los empleados por rol en un restaurante

