Sistemas Transaccionales

Ximena Lopez - 202312848

Santiago Pineda - 202023262

Sofia Losada - 202221008

Grupo 9

Documentación Entrega 3

Modelo Embebido:

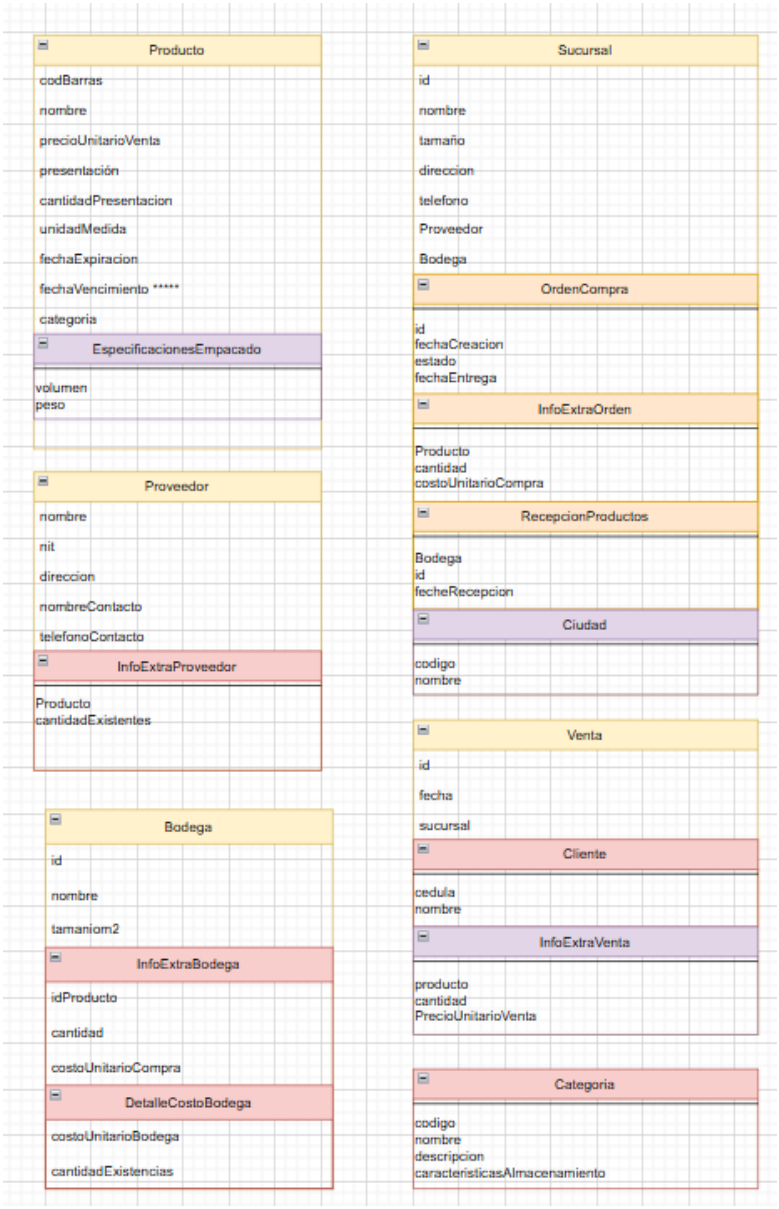
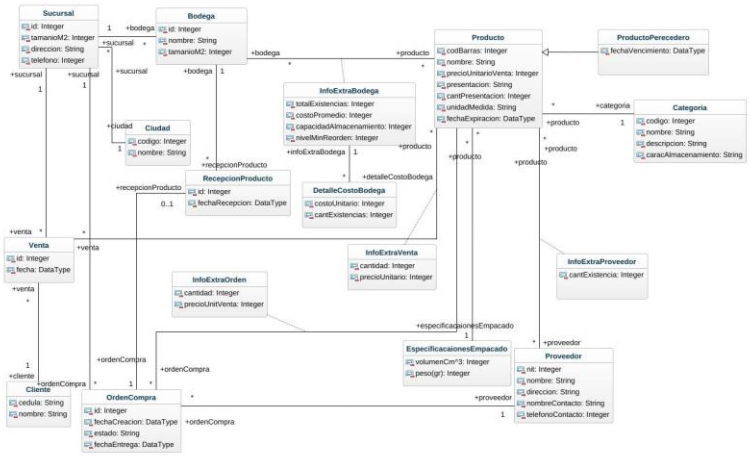


Diagrama UML:



Entidades y Atributos identificados en el modelo embebido:

Entidad	Atributo
Producto	código de barras, nombre, precio unitario de venta, presentación, cantidad de presentación, unidad de medida, fecha de expiración.
Categoria	código, nombre, descripción, características de almacenamiento.
EspecificacionesEmpacado	volumen, peso.
Proveedor	NIT, nombre, dirección, nombre de contacto, teléfono de contacto.
Bodega	ID, nombre, tamaño (m^2).
Sucursal	ID, nombre, tamaño, dirección, teléfono.
OrdenCompra	ID, fecha de creación, estado, fecha de entrega.
RecepciónProd	ID, fecha de recepción.
Ciudad	código, nombre.
Venta	ID, fecha, sucursal
Cliente	cédula, nombre.

Análisis del esquema seleccionado:

Se tuvieron en cuenta cuatro factores para establecer el esquema utilizado:

1. **Relación fuerte vs. débil**:
    - **Relación fuerte**: Si los datos están fuertemente relacionados, es decir, una entidad no tiene sentido fuera del contexto de otra (por ejemplo, InfoExtraOrden dentro de OrdenCompra), lo ideal es embebido.

- **Relación débil**: Si las entidades tienen identidades independientes y pueden reutilizarse en otros contextos. Por ejemplo, Ciudad asociada a múltiples sucursales, es mejor referenciado.

2. **Frecuencia de acceso**:
    - Si los datos relacionados siempre se consultan juntos, es preferible embebido.
    - Si las entidades relacionadas se consultan de forma separada, ya que hay baja frecuencia de acceso conjunto, es mejor referenciado.
3. **Tamaño de los datos**:
    - Si los datos relacionados son grandes o tienen muchos atributos, es mejor referenciado para evitar documentos JSON pesados.
    - Si los datos son pequeños y estáticos, embebido es más eficiente.
4. **Actualización de datos**:
    - Si una relación necesita frecuentes actualizaciones, referenciado es mejor, ya que evita actualizar documentos anidados en múltiples lugares.
    - Si los datos no cambian con frecuencia y están acoplados a la entidad principal, embebido funciona mejor.

Relaciones del modelo y Representación en JSON:

Relación	Descripción	Representación JSON

Producto – EspecificacionesEmpacado
Producto - Categotía
	Las especificaciones son únicas para cada producto y están dentro de categoría
	Esquema: Embebido.
	
	En cuanto a categoría, está de manera referenciada en producto
		{ "codigoBarras": "123456789", "nombre": "Producto A", "precioUnitarioVenta": 50, "especificacionesEmpacado": { "volumen": 1.5, "peso": 500 } },
		Categoria [{
		“codigo”: ,
		“nombre”: ,
		“especificacionesEmpacado” :
		}
		]

Proveedor - InfoExtraProveedor	Cada proveedor tiene su respectiva información de contacto y además que producto y cantidad aporta. Esquema: Embebido	{ "nit": "PROV001", "nombre": "Proveedor A", "direccion": "Calle 123", "nombreContacto": "Juan Pérez", "telefonoContacto": "123456789", "productos": [{ "codigoBarras": "123456789",

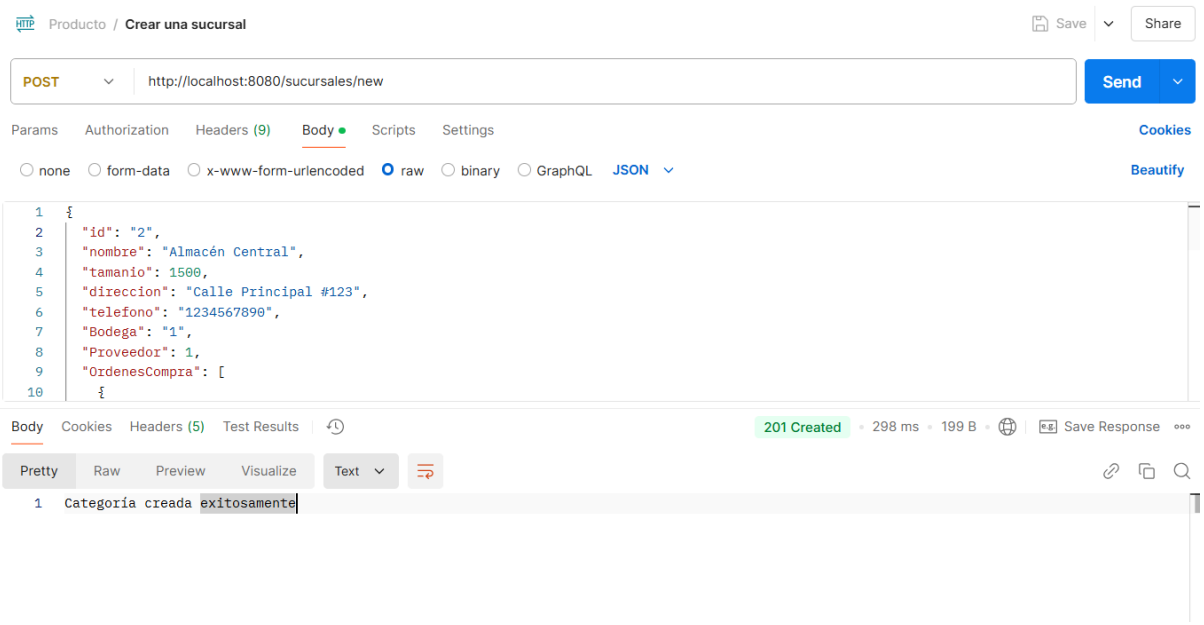
		"nombre": "Producto A", "cantidadExistentes": 150 }, { "codigoBarras": "987654321", "nombre": "Producto B", "cantidadExistentes": 200 }] }
Bodega – InfoExtraBodega Bodega - DetalleCostoBodega	Cada bodega tiene sus respectivos atributos, pero además debe tener la información extra respectiva en donde se especifican productos y su respectiva cantidad. Esquema: Embebido De igual manera, se incluye el detalle del costo de la bodega en donde se especifica el costo unitario de los productos en la respectiva bodega y la cantidad de existencias. Esquema: Embebido	
Sucursal-OrdenCompra Sucursal – InfoExtraOrden Sucursal - RecepcionProductos	Una sucursal tiene sus propios atributos y de manera embebida contiene los datos de la respectiva orden de compra, información extra de la orden y la recepción de productos Esquema: Embebido	{ "id": "SUC001", "nombre": "Sucursal A", "ciudad": { "$ref": "ciudades", "codigo": "CIU001" } }
RecepciónProductos - Ciudad	La ciudad la contiene su respectivo nombre y código y está embebida en RecepciónProductos ya que los productos son recibidos en una ciudad específica, pues las sucursales se encuentran en ciudades específicas.	{ "id": "PROV003", "nombre": "Proveedor 3", "direccion": "Avenida 15 #45-67", "nombreContacto": "Carlos López", "telefonoContacto": "4567891234", "sucursales": [{ "$ref": "sucursales", "id": "SUC001" }, { "$ref": "sucursales", "id": "SUC002" }] }
Venta - Cliente	InfoExtraOrden describe detalles particulares de una	{ "id": "ORD001", "fechaCreacion": "2024-12-01",

	orden, como productos y cantidades. Esquema: Embebido.	"estado": "vigente", "infoExtraOrden": [{ "producto": "123456789", "cantidad": 10, "costoUnitarioCompra": 45 }] }
Cliente - InfoExtraVenta	La información de cliente contiene de manera embedida InfoExtraVentas	{ "id": "VEN001", "fecha": "2024-12-01", "sucursal": "SUC001", "infoExtraVenta": [{ "producto": "123456789", "cantidad": 2, "precioUnitarioVenta": 50 }] }

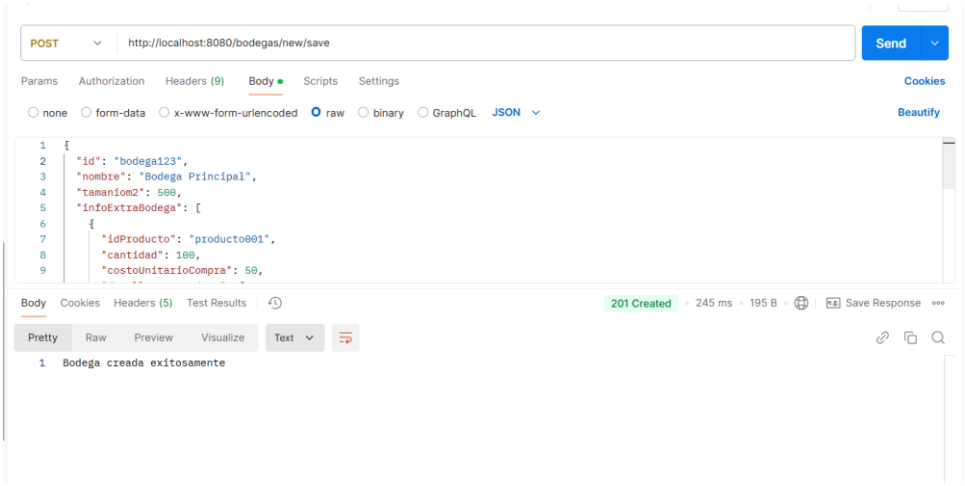
UML vs Embebido:

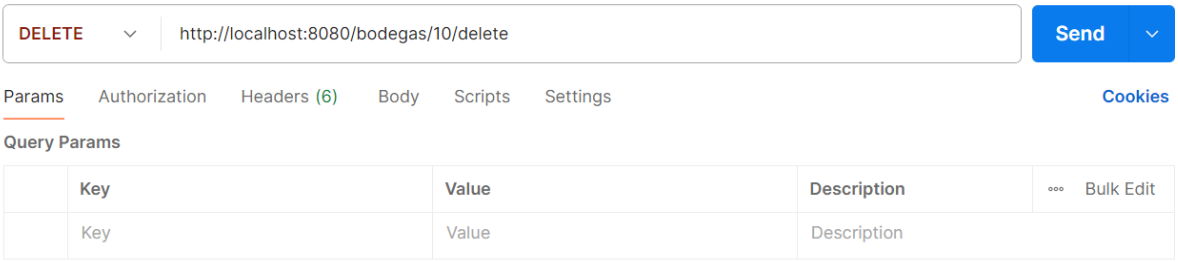
Relación de entidades	Cardinalidad UML	Modelo embebido
Sucursal - Proveedor	1-1	Referenciada
Sucursal - Ciudad	1-N	Referenciado
Producto - Categoría	N-1	Referenciado
OrdenCompra - InfoExtraOrden	1-N	Embebido
Cliente - Venta	1-N	Embebido
Bodega - InfoExtraBodega	1-N	Embebido

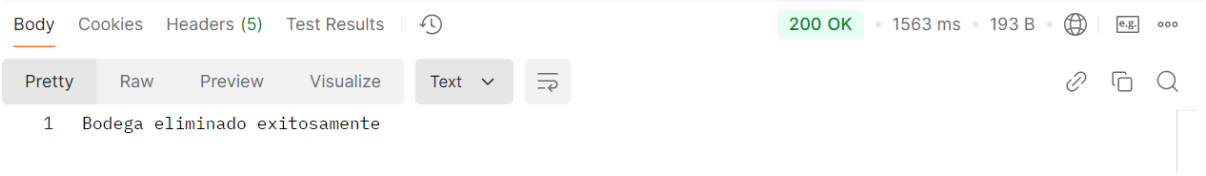
RF1



RF2

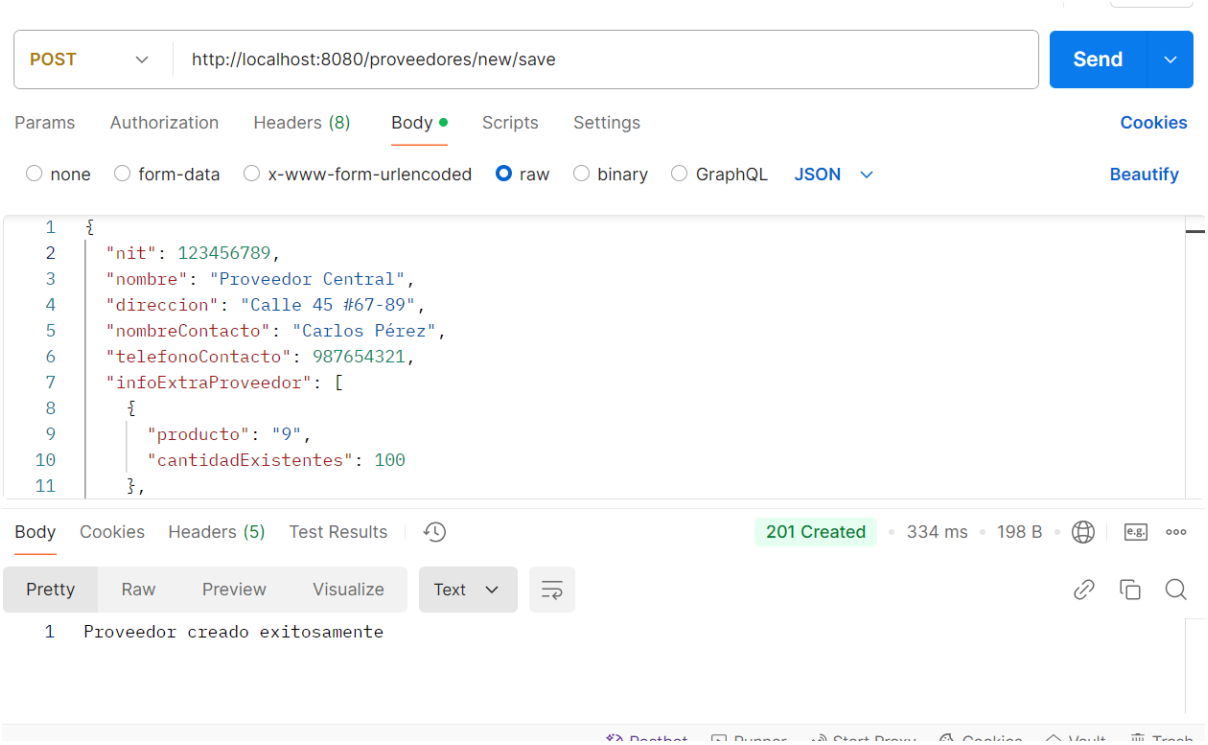




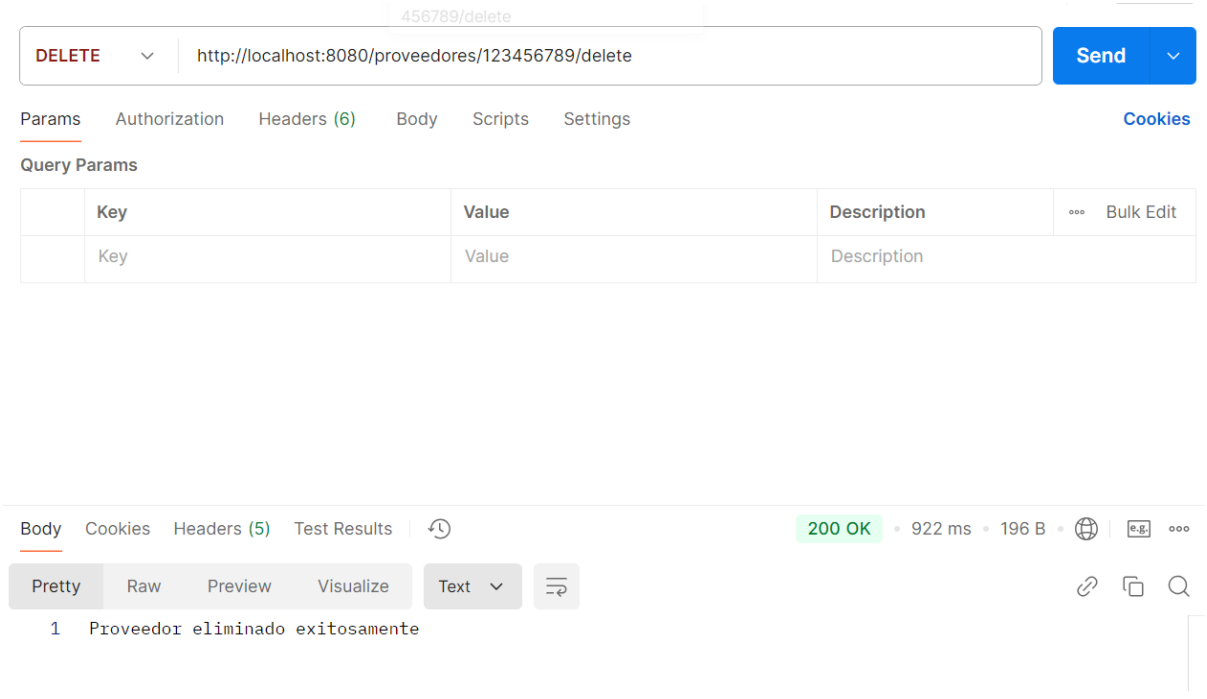


RF3:

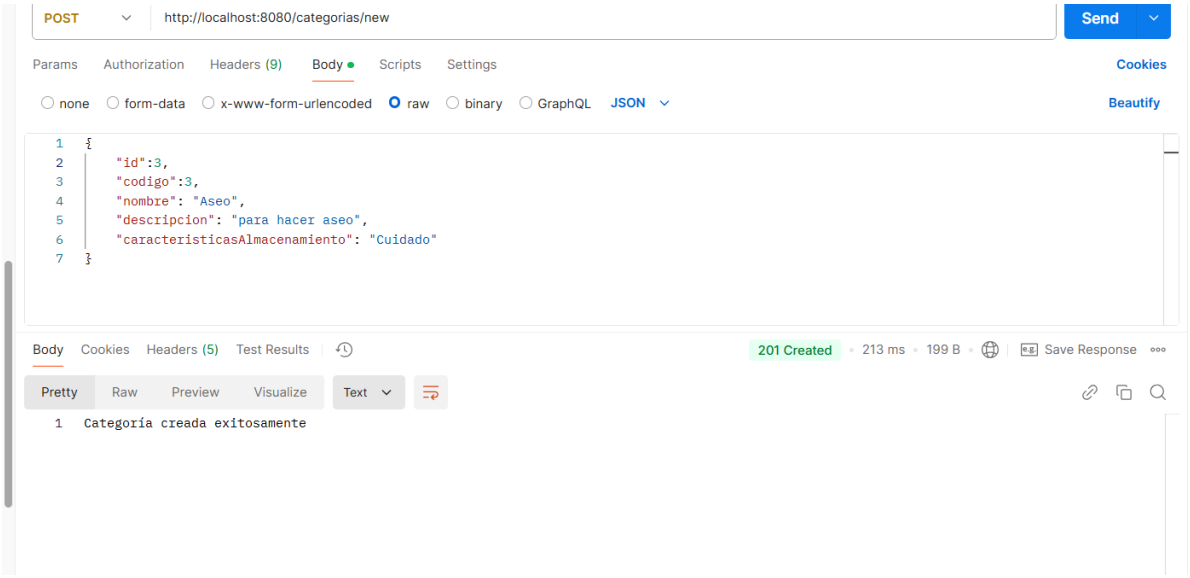
Crear un proveedor:



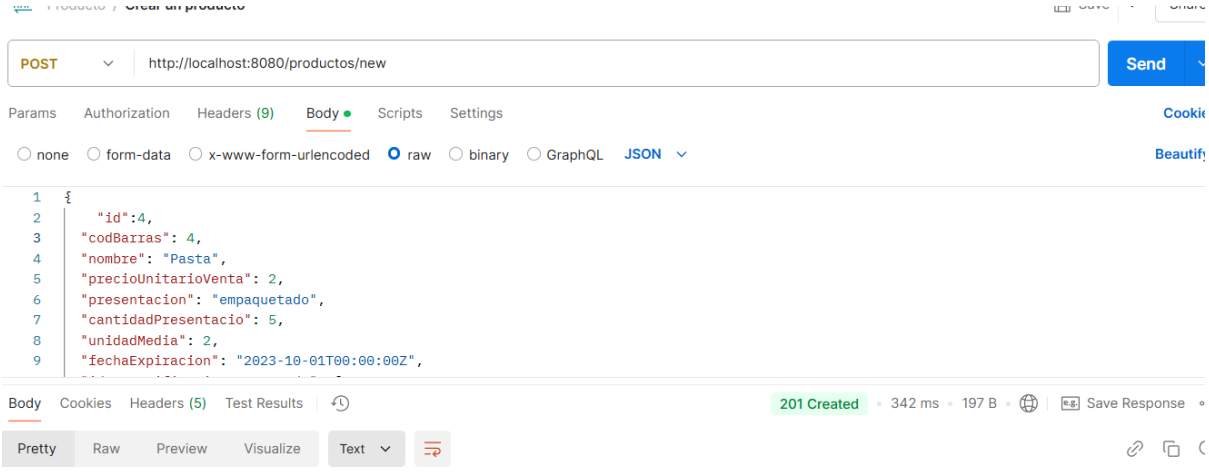
Eliminar un proveedor:

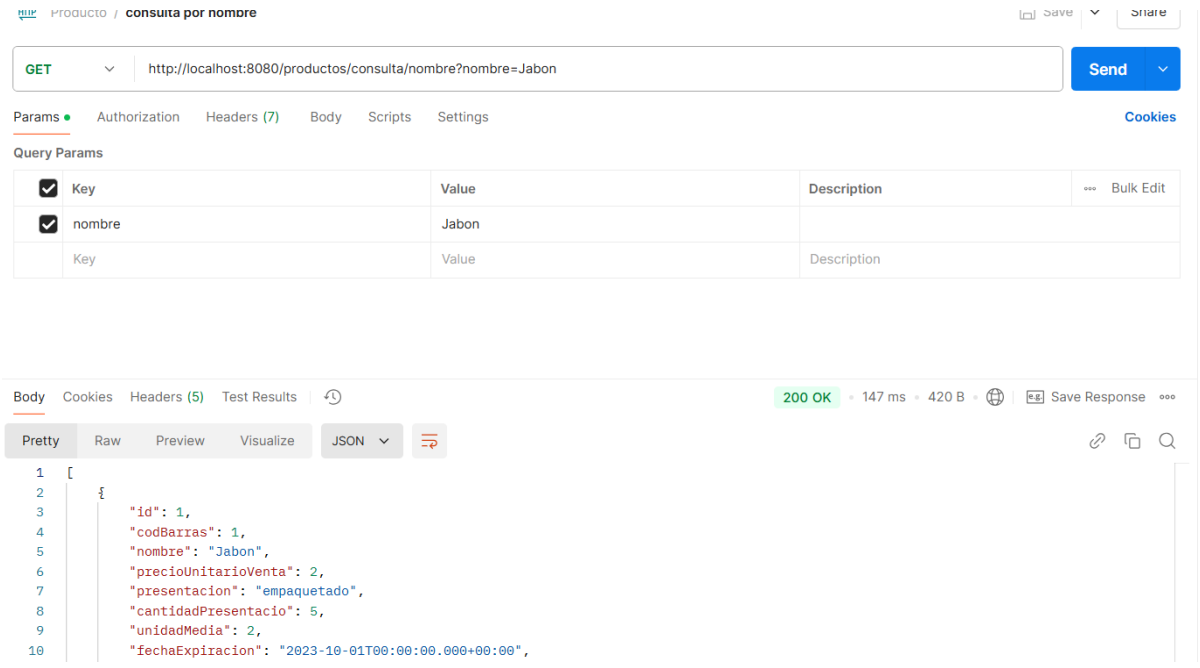


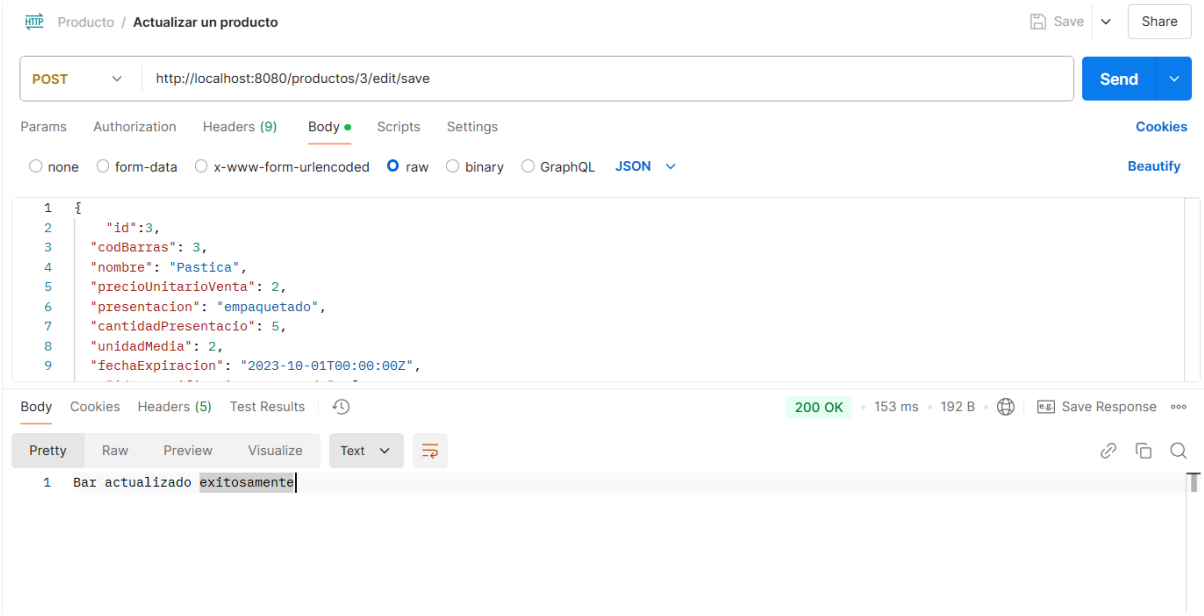
RF4:


POST
http://localhost:8080/categorias/new
Send
Params
Authorization
Headers (9)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"id":3,
"codigo":3,
"nombre": "Aseo",
"descripcion": "para hacer aseo",
"caracteristicasAlmacenamiento": "Cuidado"
}
Body
Cookies
Headers (5)
Test Results
201 Created
213 ms
199 B
Save Response
Pretty
Raw
Preview
Visualize
Text
Categoría creada exitosamente

RF5:


POST
http://localhost:8080/productos/new
Send
Params
Authorization
Headers (9)
Body
Scripts
Settings
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
{
"id":4,
"codBarras": 4,
"nombre": "Pasta",
"precioUnitarioVenta": 2,
"presentacion": "empaquetado",
"cantidadPresentacio": 5,
"unidadMedia": 2,
"fechaExpiracion": "2023-10-01T00:00:00Z",
Body
Cookies
Headers (5)
Test Results
201 Created
342 ms
197 B
Save Response
Pretty
Raw
Preview
Visualize
Text
Producto creado exitosamente


Producto / consulta por nombre
GET
http://localhost:8080/productos/consulta/nombre?nombre=Jabon
Send
Params
Authorization
Headers (7)
Body
Scripts
Settings
Cookies
Query Params
Key
Value
Description
Bulk Edit
nombre
Jabon
Body
Cookies
Headers (5)
Test Results
200 OK
147 ms
420 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
[
{
"id": 1,
"codBarras": 1,
"nombre": "Jabon",
"precioUnitarioVenta": 2,
"presentacion": "empaquetado",
"cantidadPresentacio": 5,
"unidadMedia": 2,
"fechaExpiracion": "2023-10-01T00:00:00.000+00:00",

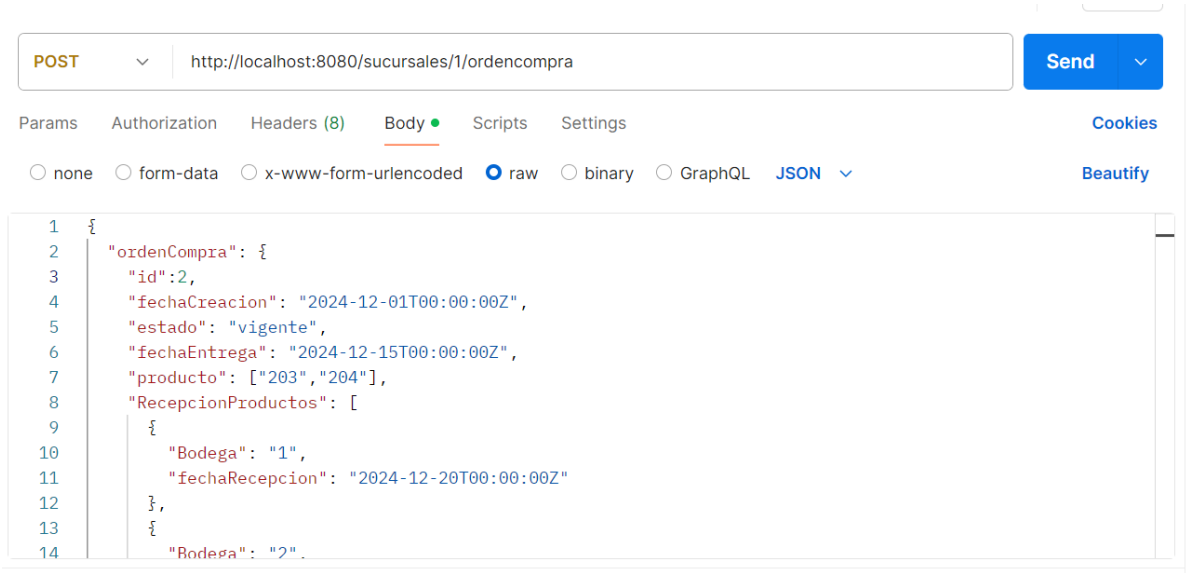


RF6

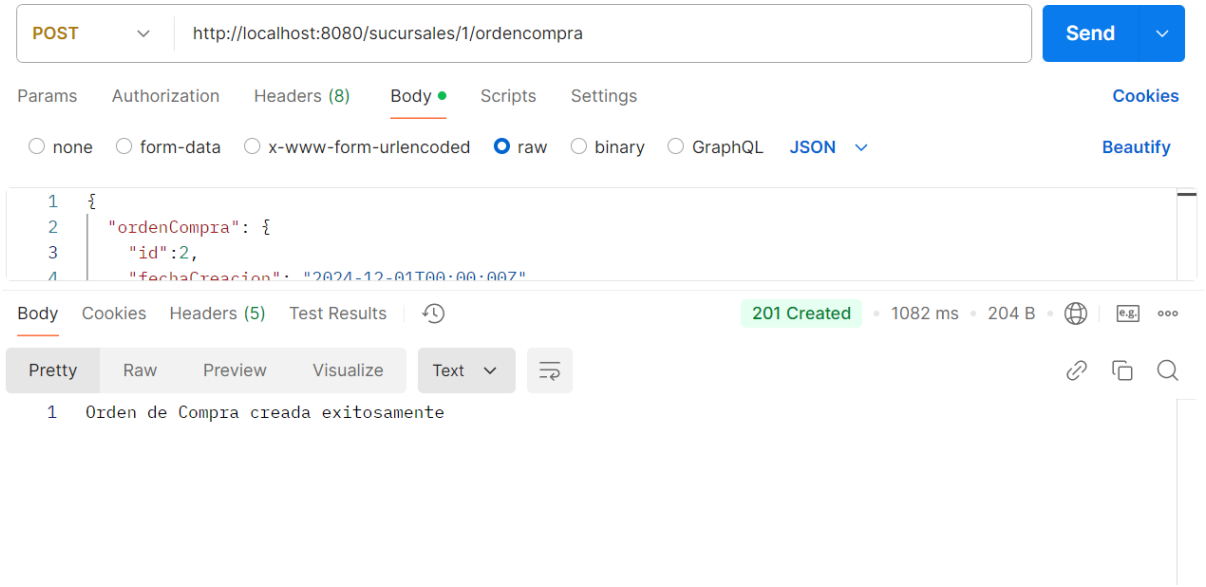
Se debe crear una Orden de Compra para una sucursal, por lo que se recibe: El encabezado con: La fecha de creación de la orden, el id de la sucursal, el proveedor al que se le quiere comprar, el id del proveedor, la fecha esperada de entrega de los productos, la cual es ingresada por el usuario.

Y además también se recibe el detalle, el cual por su lado contiene: Uno o más productos que se quieren comprar, se recibiría el id de cada producto. Las cantidades de los productos que se desean comprar. Los precios de los productos que se desean comprar, los cuales son ingresados por el usuario. Al ser creada, la orden de compra queda en estado “vigente”.

Por lo que así se vería la entrada en postman:



Y cuando se ejecuta responde que se ha creado la orden de compra:



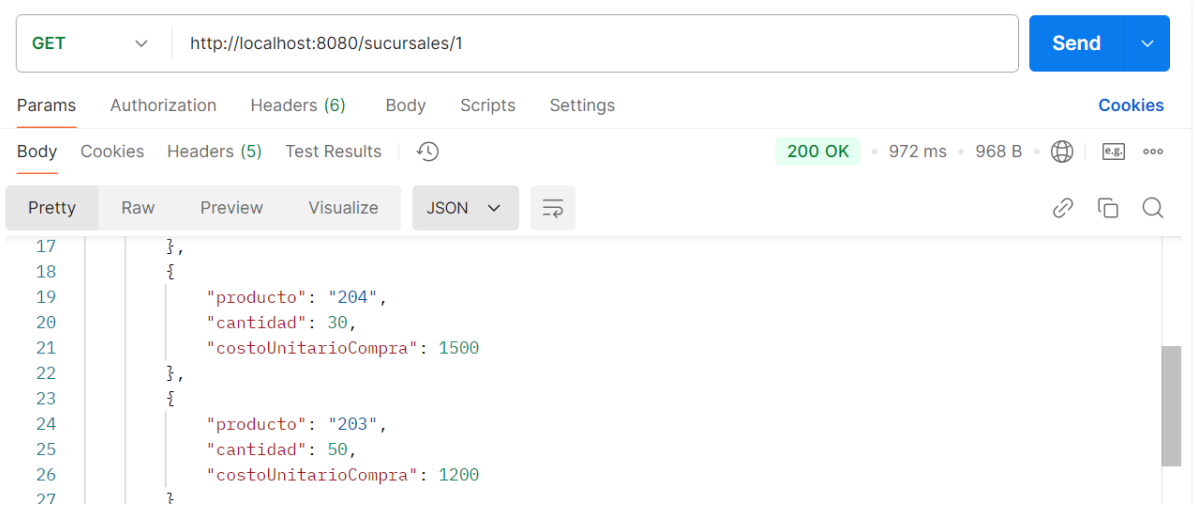
Y acá se puede ver cómo queda en la base de datos:

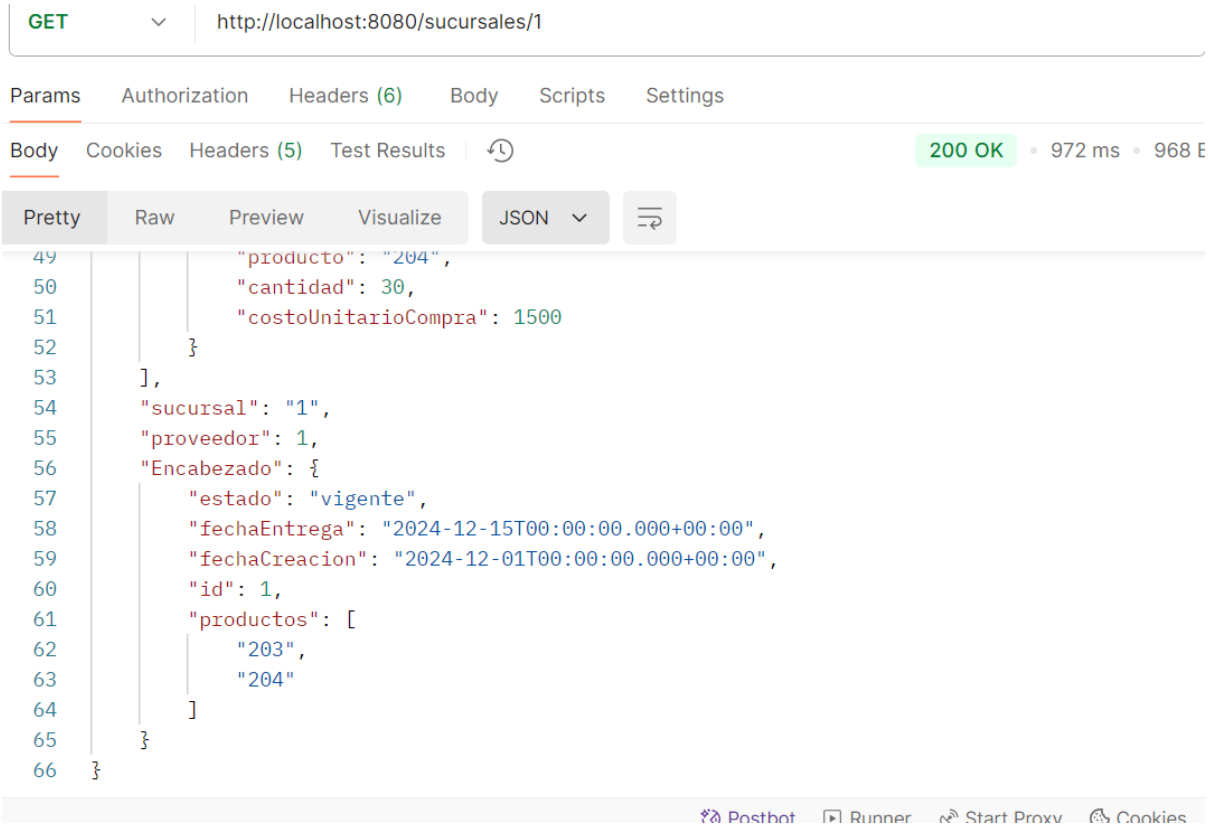
```
_id: "1"
nombre : "Sucursal Norte"
tamanio : 1000
direccion : "Av. Principal #45-67"
telefono : "123456789"
Proveedor : 1
Bodega : "1"
▸ infoExtraOrden : Array (10)
▾ ordenesCompra : Array (3)
  ▸ 0: Object
  ▸ 1: Object
  ▾ 2: Object
      _id : 2
      fechaCreacion : 2024-12-01T00:00:00.000+00:00
      estado : "vigente"
      fechaEntrega : 2024-12-15T00:00:00.000+00:00
    ▸ productos : Array (2)
```

RF7

Se quiere leer una Orden de Compra, así que se recibe el número de la orden de compra. Como resultado se obtiene la información del encabezado y del detalle descritos antes.

Acá se ve la entrada en postman y como da la respuesta:

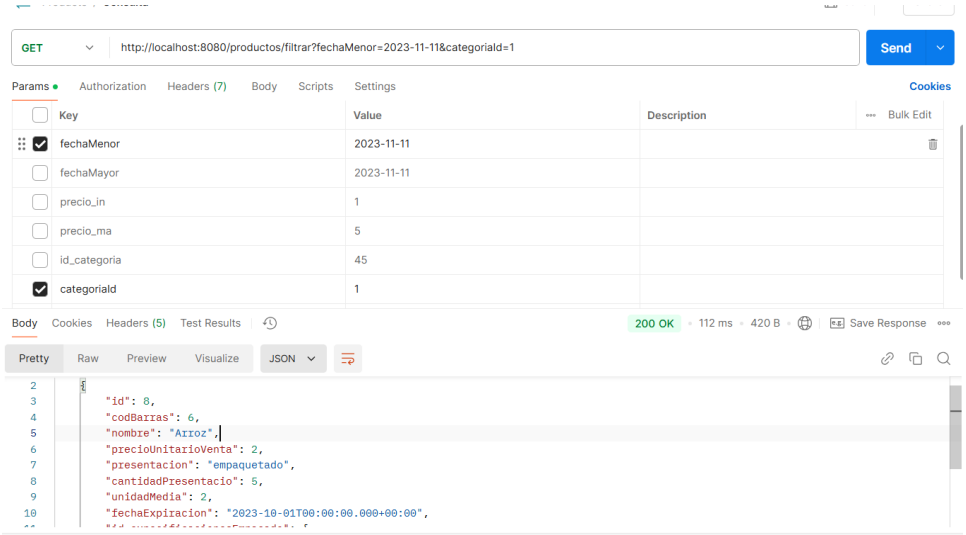




Requerimientos Funcionales de Consulta

RFC1

Consultar productos con unas caracteristicas

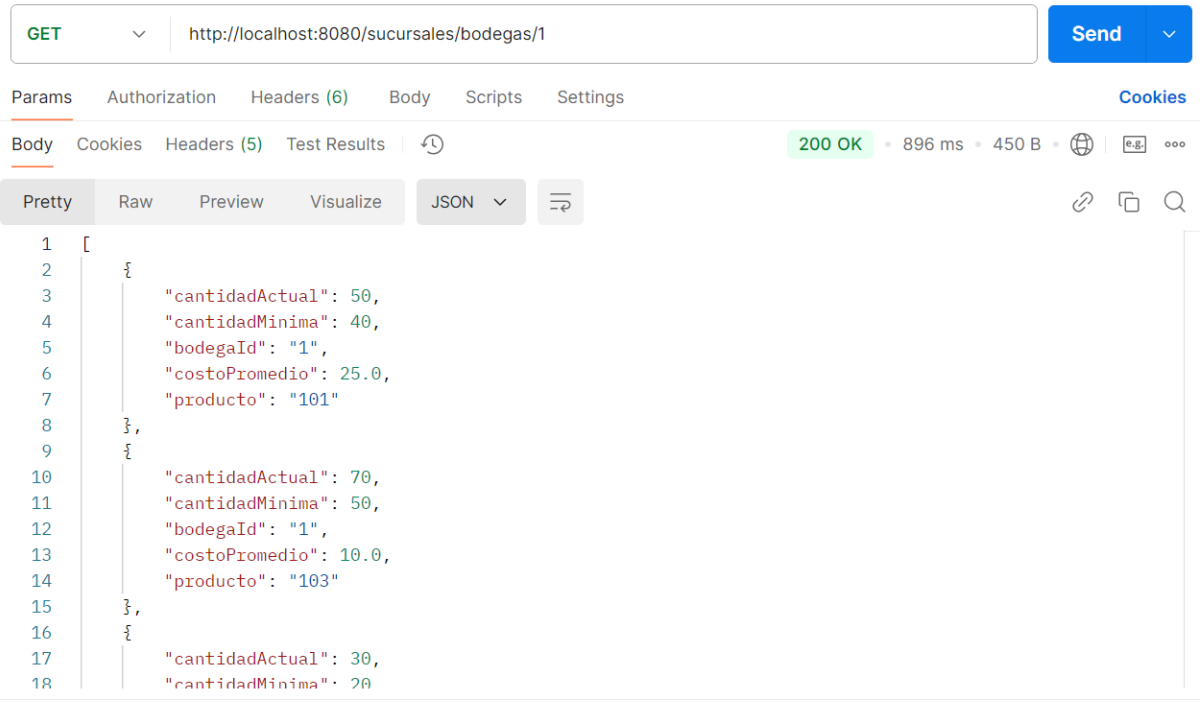


RFC2

Se quiere saber el inventario de productos en una sucursal. Entonces se ingresa la sucursal para la cual se quiere el reporte. El reporte arroja, para cada bodega de la

sucursal ingresada, el nombre de los productos que contiene, su cantidad actual disponible, la cantidad mínima que se requiere en inventario para esa bodega y su costo promedio.

Acá se muestra un ejemplo de la entrada en postman con el id de la sucursal y la salida:



Anexos

ESCENARIOS DE PRUEBA

Caso que no pasan validación para RF3:

db.proveedores.insertOne({

nit: "123456789",

direccion: "Calle 45 #67-89",

nombreContacto: "Carlos Pérez",

telefonoContacto: 987654321,

infoExtraProveedor: [

{

```
        producto:                                                    123,
        cantidadExistentes:                                          "100"
      }
   ]
});
```

Script utilizado para insertar bodegas:

```
[
 {
  "_id": "1",
  "nombre": "Bodega Central",
  "tamaniom2": 500,
  "infoExtraBodega": [
   {
    "idProducto": "101",
    "cantidad": 50,
    "costoUnitarioCompra": 100,
    "detalleCostoBodega": {
     "costoUnitarioVenta": 150,
     "cantidadExistencias": 40
    }
   },
   {
```

```
    "idProducto": "102",
    "cantidad": 30,
    "costoUnitarioCompra": 80,
    "detalleCostoBodega": {
     "costoUnitarioVenta": 120,
     "cantidadExistencias": 20
    }
   }
 ]
},
{
 "_id": "2",
 "nombre": "Bodega Norte",
 "tamaniom2": 300,
 "infoExtraBodega": [
  {
    "idProducto": "103",
    "cantidad": 40,
    "costoUnitarioCompra": 60,
    "detalleCostoBodega": {
     "costoUnitarioVenta": 90,
     "cantidadExistencias": 30
    }
```

```
    }
  ]
 },
 {
  "_id": "3",
  "nombre": "Bodega Sur",
  "tamaniom2": 450,
  "infoExtraBodega": [
   {
    "idProducto": "104",
    "cantidad": 70,
    "costoUnitarioCompra": 200,
    "detalleCostoBodega": {
     "costoUnitarioVenta": 250,
     "cantidadExistencias": 60
    }
   }
  ]
 },
 {
  "_id": "4",
  "nombre": "Bodega Este",
  "tamaniom2": 400,
```

```
 "infoExtraBodega": [
  {
   "idProducto": "105",
   "cantidad": 25,
   "costoUnitarioCompra": 50,
   "detalleCostoBodega": {
    "costoUnitarioVenta": 70,
    "cantidadExistencias": 15
   }
  }
 ]
},
{
 "_id": "5",
 "nombre": "Bodega Oeste",
 "tamaniom2": 350,
 "infoExtraBodega": [
  {
   "idProducto": "106",
   "cantidad": 90,
   "costoUnitarioCompra": 300,
   "detalleCostoBodega": {
    "costoUnitarioVenta": 400,
```

```
      "cantidadExistencias": 80
     }
    }
  ]
},
{
  "_id": "6",
  "nombre": "Bodega Principal",
  "tamaniom2": 600,
  "infoExtraBodega": [
   {
    "idProducto": "107",
    "cantidad": 45,
    "costoUnitarioCompra": 120,
    "detalleCostoBodega": {
     "costoUnitarioVenta": 180,
     "cantidadExistencias": 35
    }
   }
  ]
},
{
  "_id": "7",
```

```
 "nombre": "Bodega Auxiliar",
 "tamaniom2": 250,
 "infoExtraBodega": [
  {
   "idProducto": "108",
   "cantidad": 55,
   "costoUnitarioCompra": 140,
   "detalleCostoBodega": {
    "costoUnitarioVenta": 190,
    "cantidadExistencias": 45
   }
  }
 ]
},
{
 "_id": "8",
 "nombre": "Bodega Secundaria",
 "tamaniom2": 320,
 "infoExtraBodega": [
  {
   "idProducto": "109",
   "cantidad": 35,
   "costoUnitarioCompra": 60,
```

```
    "detalleCostoBodega": {
      "costoUnitarioVenta": 80,
      "cantidadExistencias": 25
    }
   }
 ]
},
{
 "_id": "9",
 "nombre": "Bodega Adicional",
 "tamaniom2": 420,
 "infoExtraBodega": [
  {
    "idProducto": "110",
    "cantidad": 75,
    "costoUnitarioCompra": 220,
    "detalleCostoBodega": {
      "costoUnitarioVenta": 300,
      "cantidadExistencias": 65
    }
  }
 ]
},
```

```
 {
  "_id": "10",
  "nombre": "Bodega de Respaldo",
  "tamaniom2": 480,
  "infoExtraBodega": [
   {
    "idProducto": "111",
    "cantidad": 60,
    "costoUnitarioCompra": 180,
    "detalleCostoBodega": {
     "costoUnitarioVenta": 250,
     "cantidadExistencias": 50
    }
   }
  ]
 }
]
```