

Sistemas transaccionales

Proyecto grupo 6

Documentación con imágenes

Integrantes:

- María Lucia Benavides
- Ángela Jiménez
- Kevin Castillo

Notas:

Los cambios hechos para que funcionaran todas las pruebas de la entrega 1 eran más que todo de sintaxis y de conexión con bases de datos, además se crearon pruebas para los requerimientos de consulta de la entrega 1 y quedaron funcionando perfectamente.

1. Creación de tablas:

```
Table CIUDADES created.
```

```
Table SUCURSALES created.
```

```
Table PROVEEDORES created.
```

```
Table CATEGORIAS created.
```

```
Table BODEGAS created.
```

```
Table CLIENTES created.
```

```
Table PRODUCTOS created.
```

```
Table RECEPCIONES created.
```

Table RECEPCIONES created.

Table INFO_RECEPCION created.

Table NIVELES_REORDEN created.

Table ORDENES_COMPRA created.

Table PRODUCTO_BODEGA created.

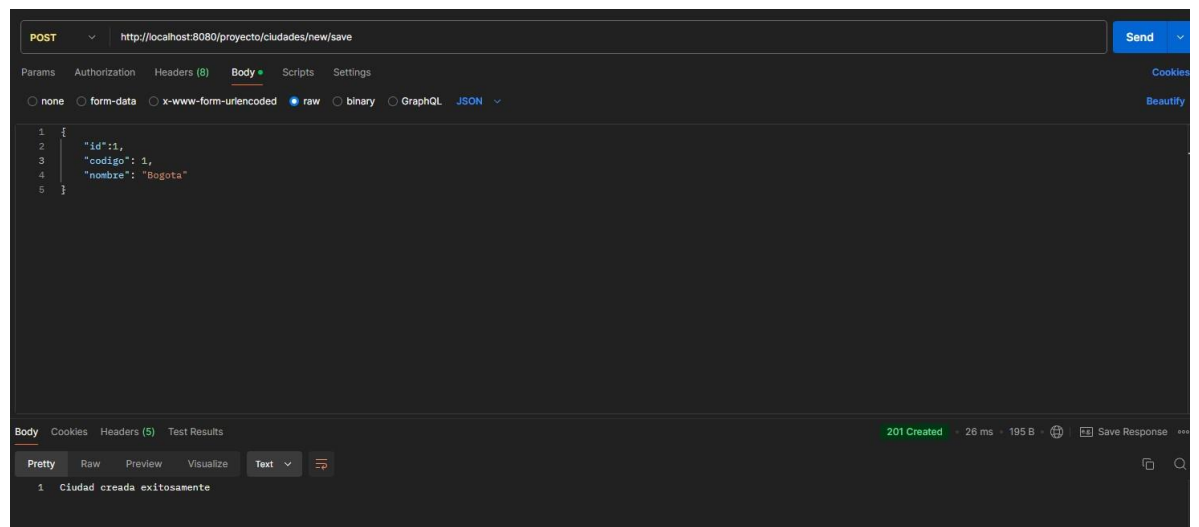
Table PRODUCTO_ORDEN created.

Table VENTAS created.

2. Pruebas en PostMan de requerimientos:

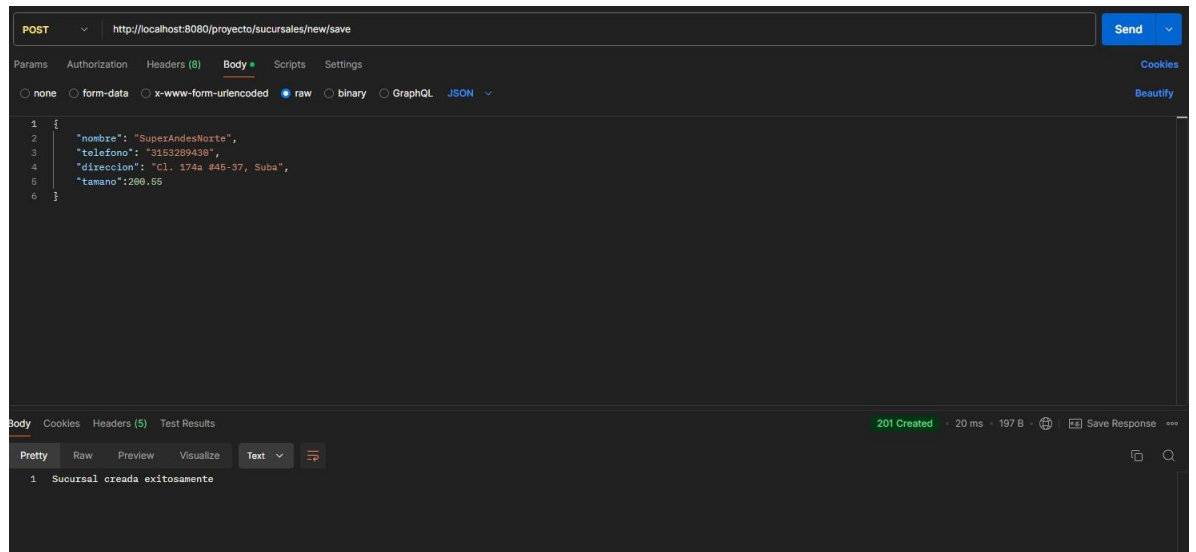
Requerimientos funcionales:

-Requerimiento funcional 1: Crear una ciudad:



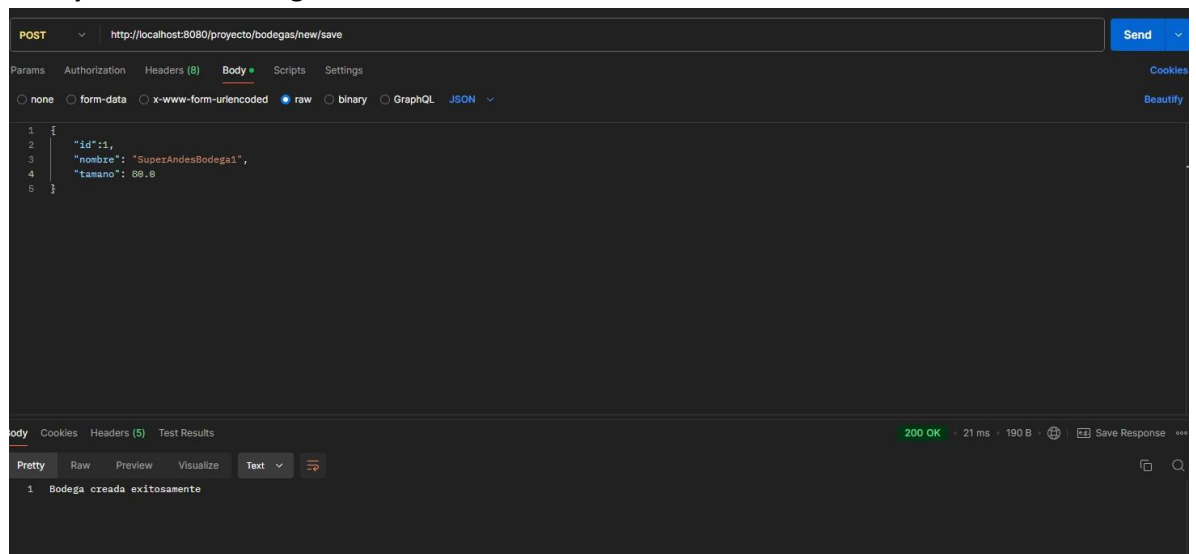
-Requerimiento funcional 2:

Crear una sucursal:

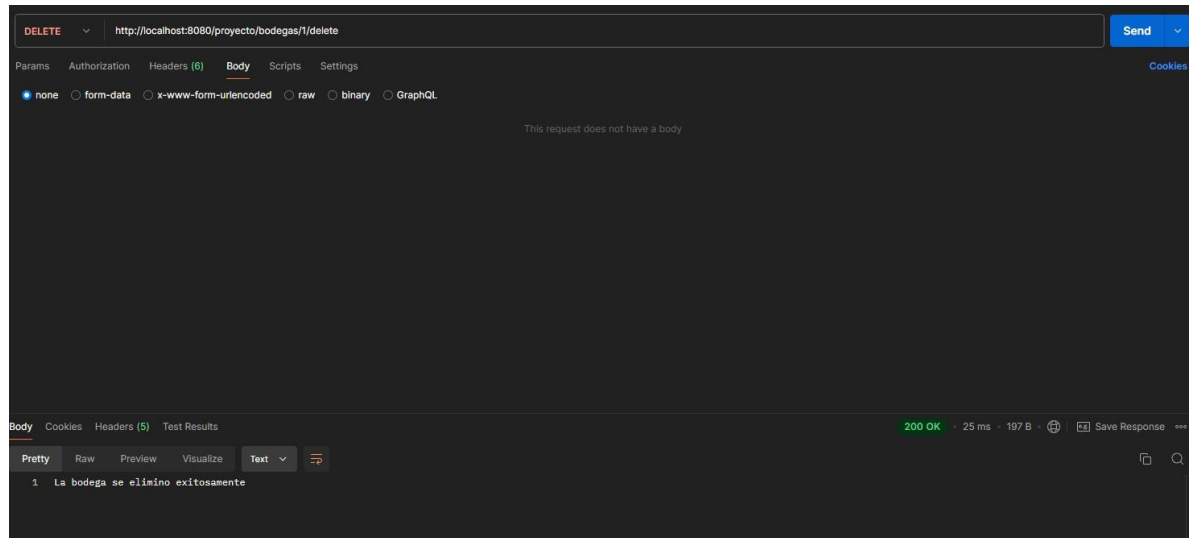


-Requerimiento funcional 3:

Crear y borrar una bodega: Crear:

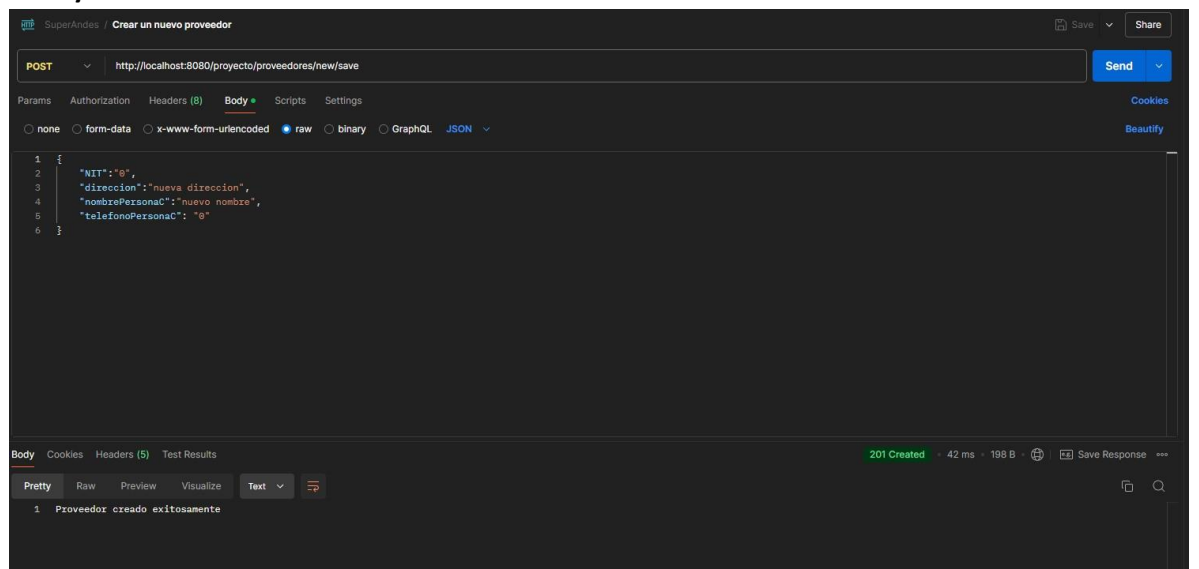


Borrar:

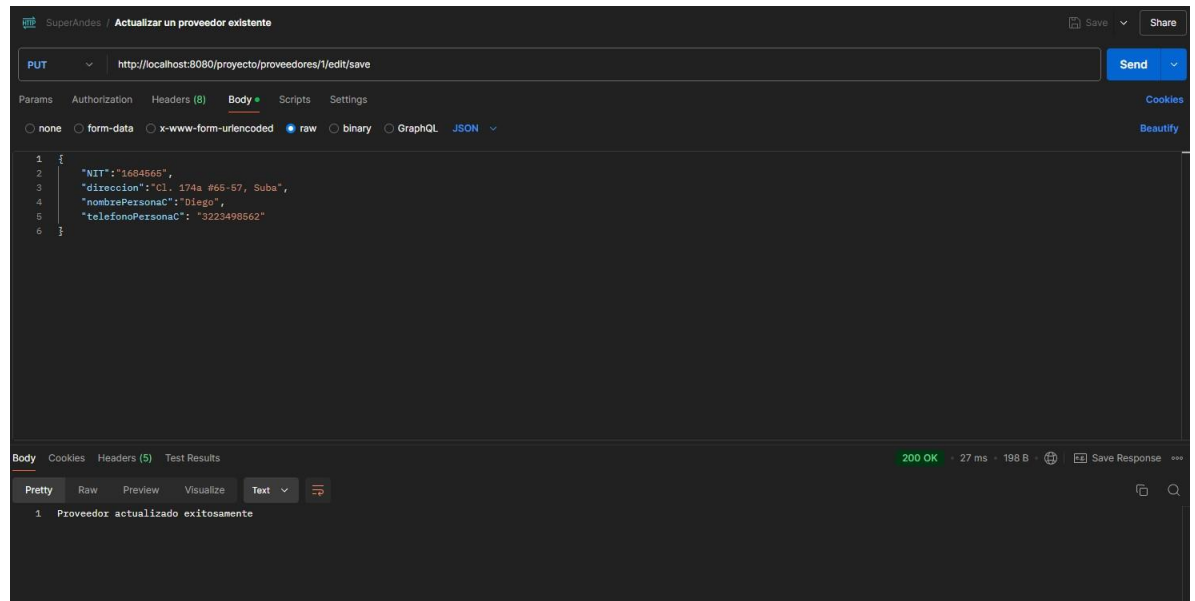


-Requerimiento funcional 4:

Crear y Actualizar Proveedores: Crear:



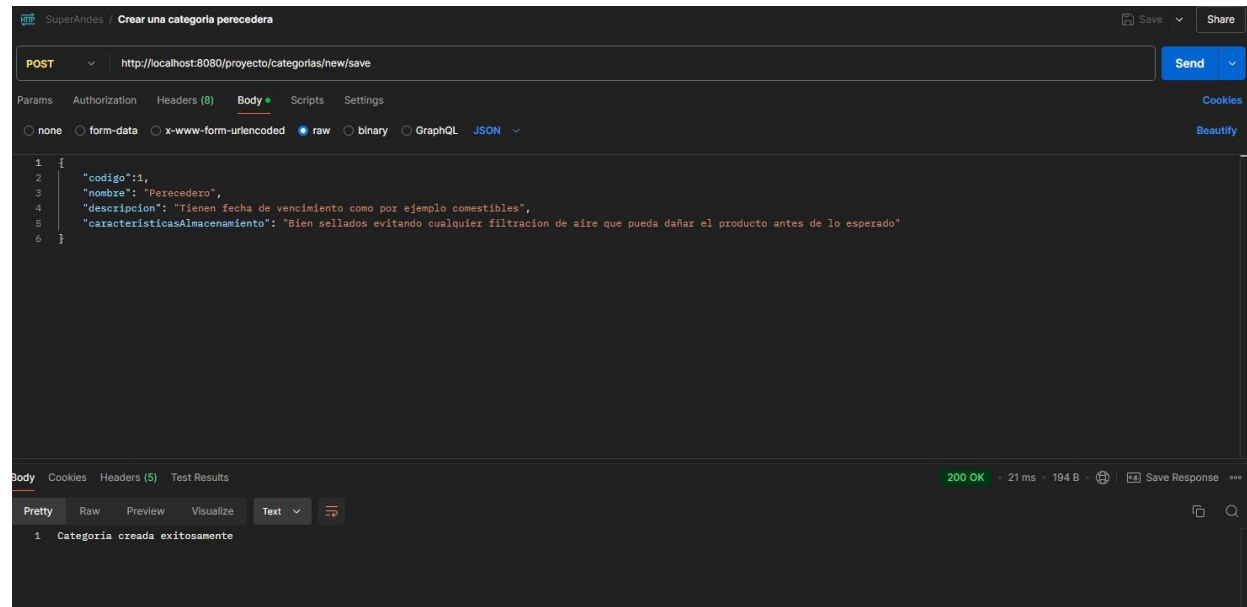
Actualizar:

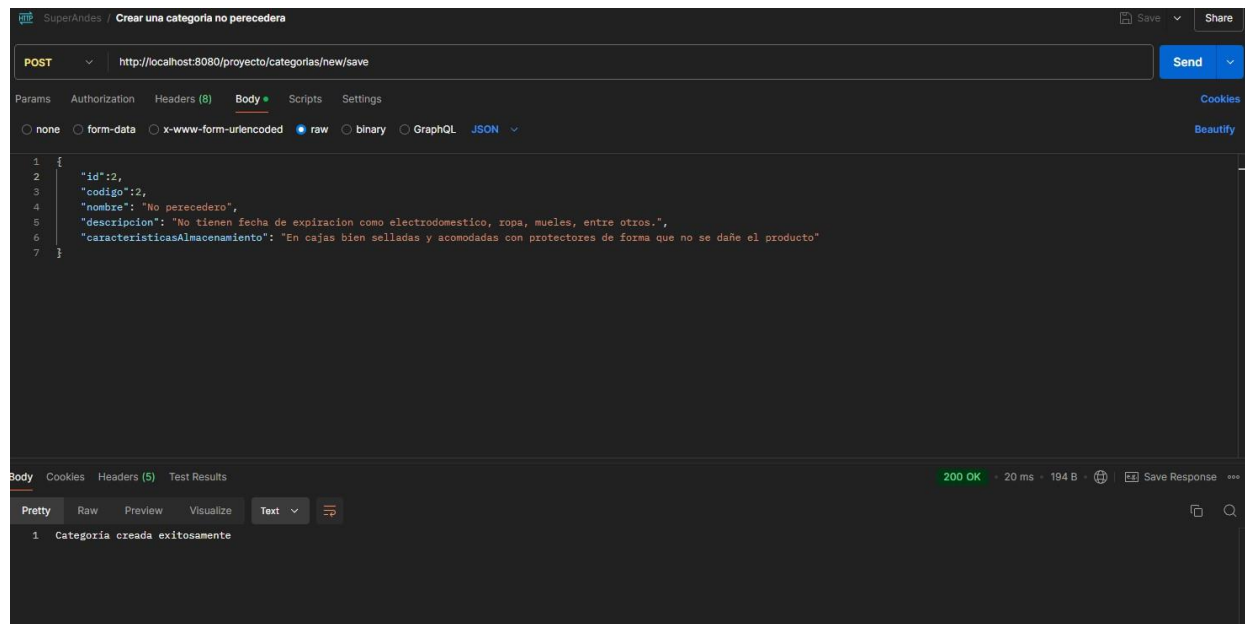


-Requerimiento funcional 5:

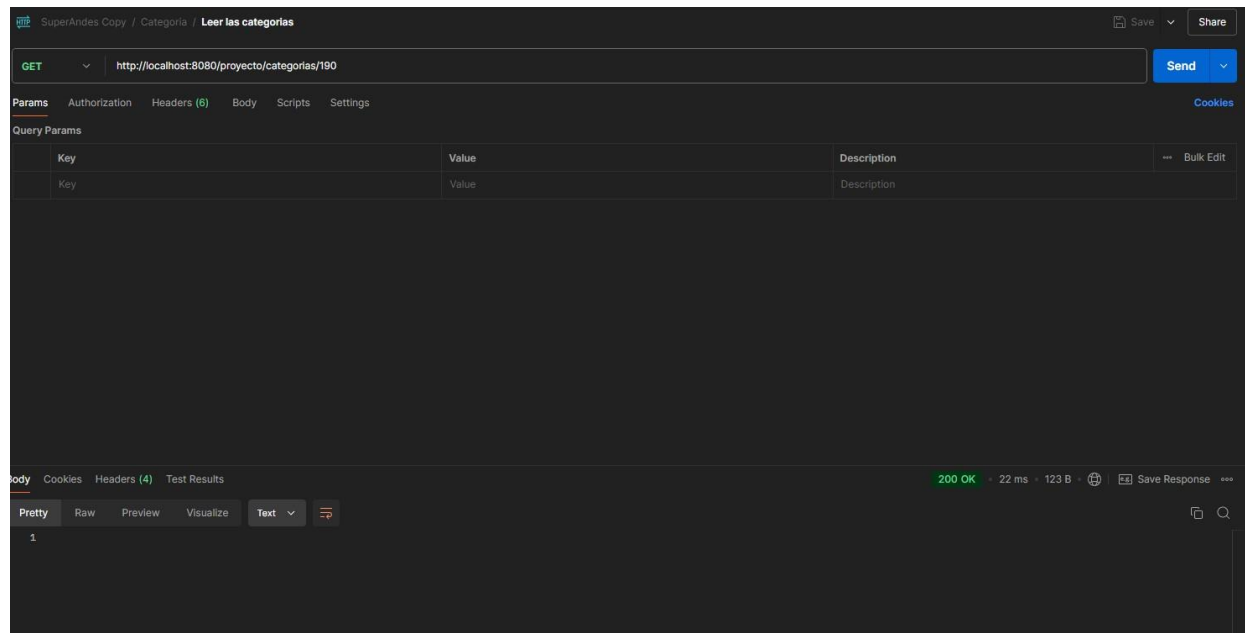
Crear y leer una categoría de producto:

Crear:





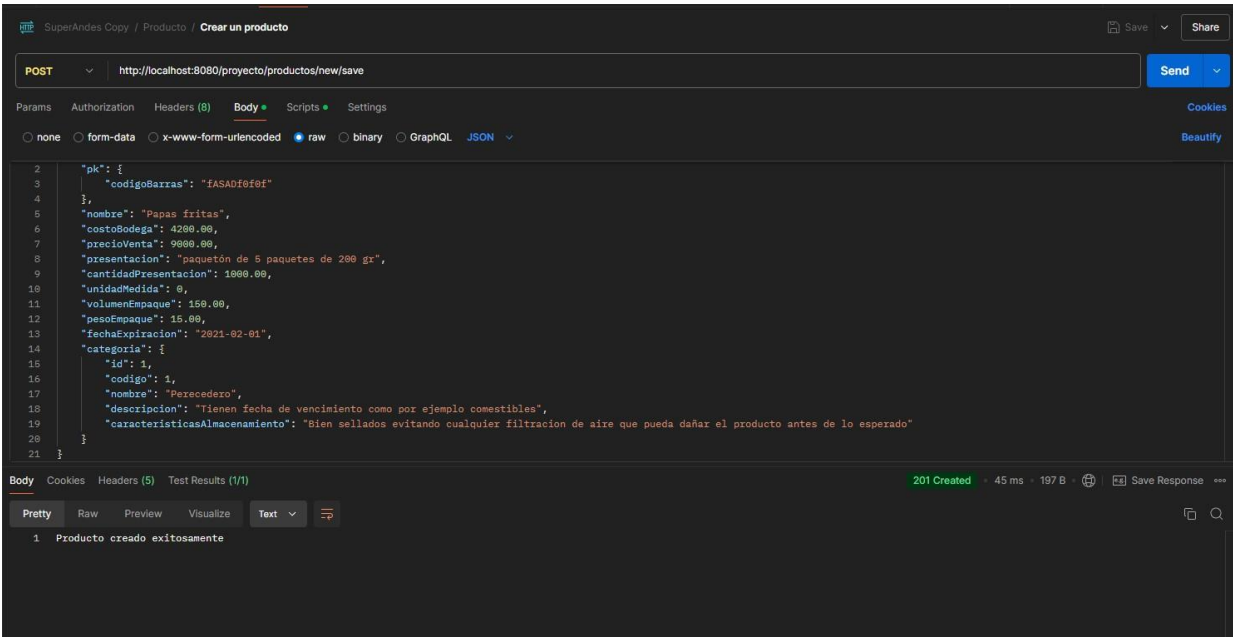
Leer:



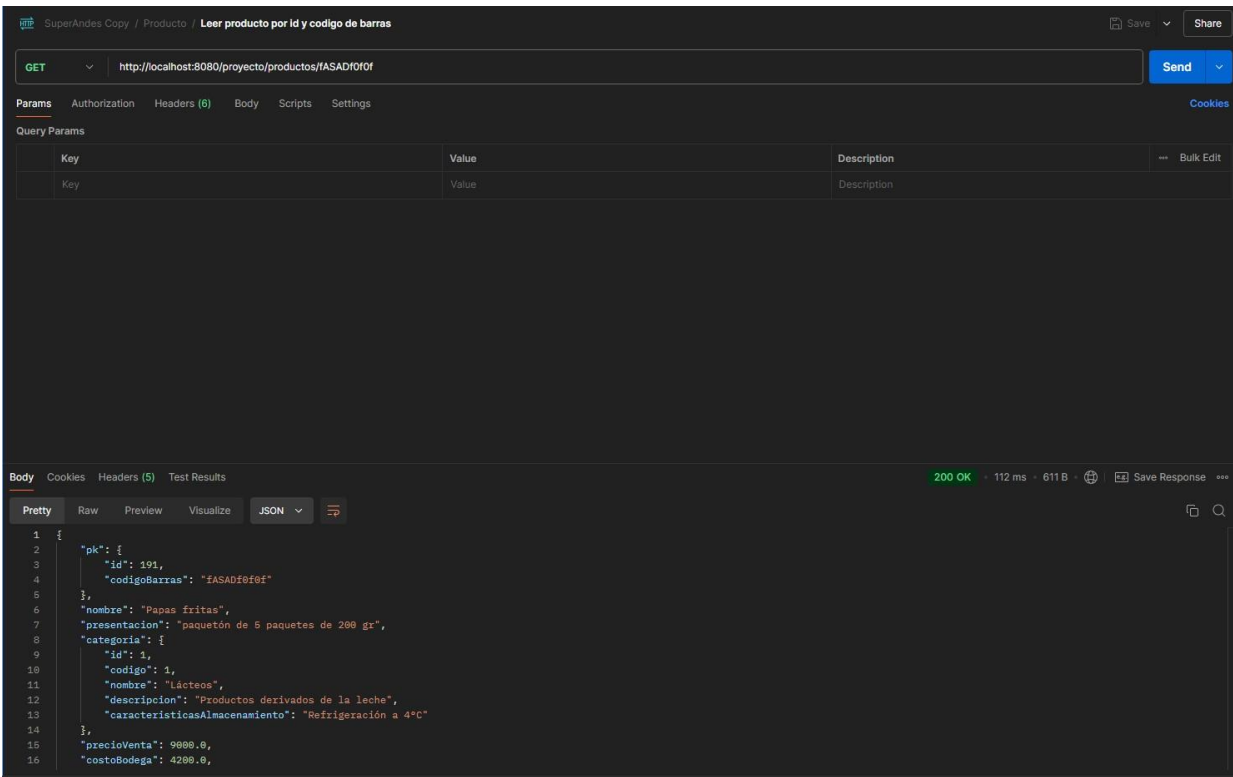
-Requerimiento funcional 6:

Crear y leer y actualizar un producto:

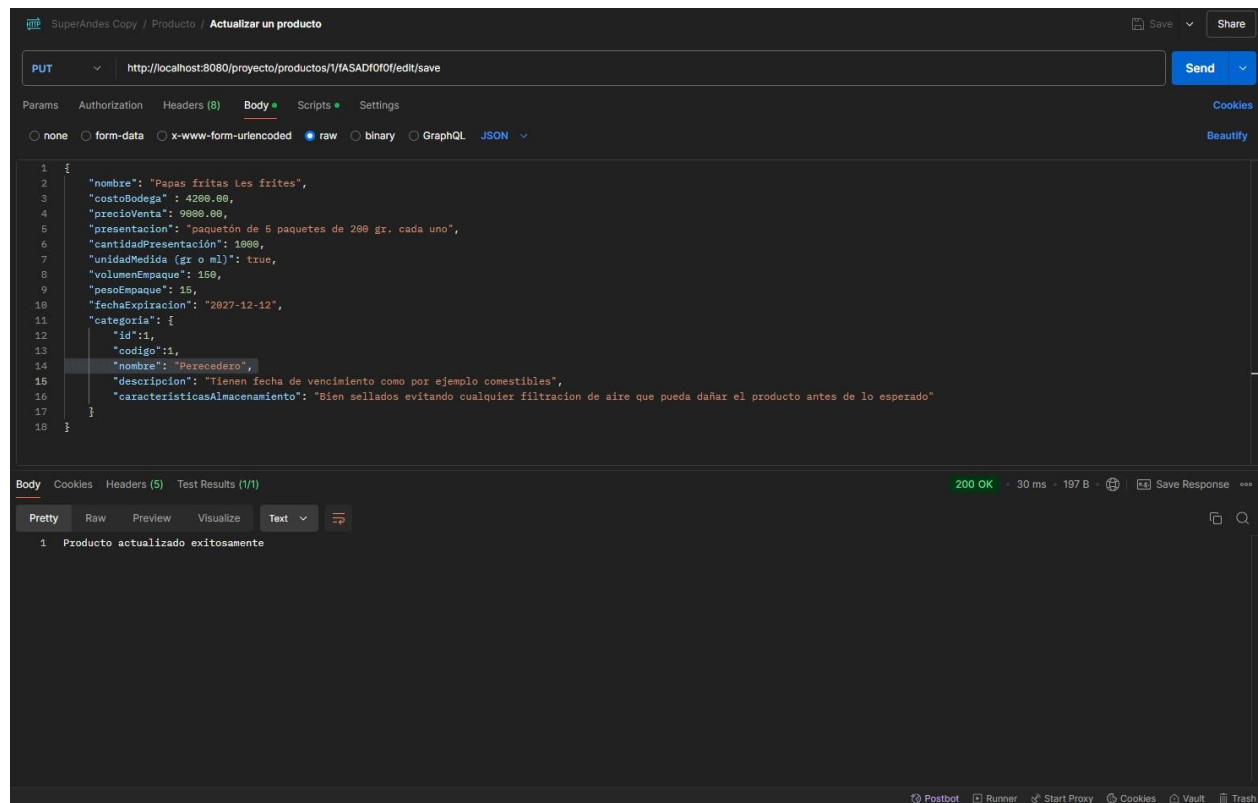
Crear:



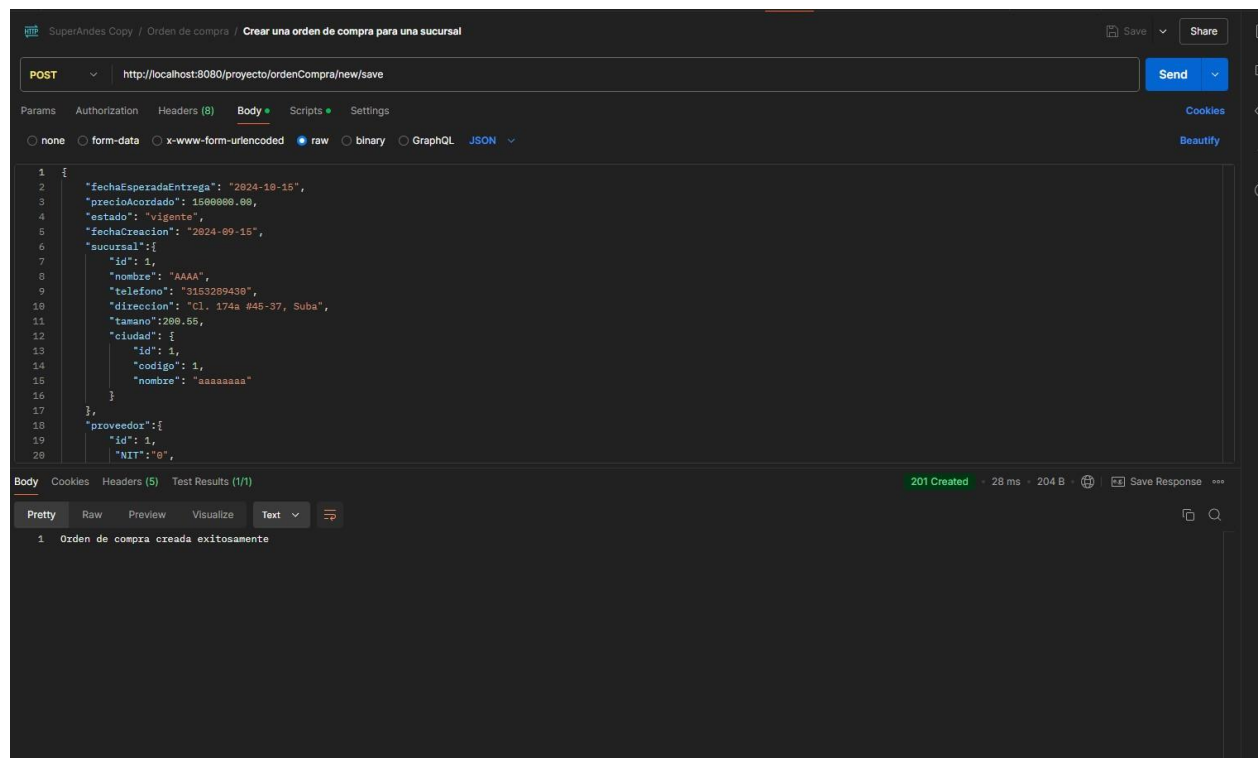
Leer:



Actualizar: -Requerimiento funcional 7:



Crear una orden de compra para una sucursal: -Requerimiento funcional 8:



Actualizar una orden de compra cambiando su estado a anulada:

SuperAndes Copy / Orden de compra / Actualizar una orden de compra

PUThttp://localhost:8080/proyecto/ordenCompra/edit/save

Send

ParamsAuthorizationHeaders (8)BodyScriptsSettings

noneform-datax-www-form-urlencodedrawbinaryGraphQLJSON

```
1 {
2   "id":1,
3   "fechaEsperadaEntrega": "2024-10-15",
4   "precioAcordado": 1500000.00,
5   "estado": "anulada",
6   "fechaCreacion": "2024-09-15",
7   "sucursal":{"id":1,
8     "nombre": "SuperAndesNorte",
9     "telefono": 3153289430,
10    "direccion": "Cl. 174a #45-37, Suba, Bogotá",
11    "tamano":200.55,
12    "id_ciudad":{"id":1,
13      "codigo": 1,
14      "nombre": "Bogota"
15    }
16  },
17   "proveedor":{"id":1,
18     "NIT":"0",
19     "direccion":"nueva direccion",
20   }
21 }
```

BodyCookiesHeaders (5)Test Results (1/1)

200 OK · 24 ms · 204 B · Save Response

PrettyRawPreviewVisualizeText

1 Orden de compra actualizada exitosamente

-Requerimiento funcional 9:
Mostrar todas las órdenes de compra:

SuperAndes Copy / Orden de compra / Obtener todas las ordenes de compra

GEThttp://localhost:8080/proyecto/ordenesCompra

Send

ParamsAuthorizationHeaders (6)BodyScriptsSettings

Query Params

Key	Value	Description
Key	Value	Description

BodyCookiesHeaders (5)Test Results

200 OK · 109 ms · 6.13 KB · Save Response

PrettyRawPreviewVisualizeJSON

```
1 [
2   {
3     "id": 211,
4     "fechaEsperadaEntrega": "2024-10-15 00:00:00",
5     "estado": "vigente",
6     "fechaCreacion": "2024-09-15 00:00:00",
7     "sucursal": {
8       "id": 1,
9       "nombre": "Sucursal Centro",
10      "telefono": "3200000001",
11      "direccion": "Carrera 7 #34-15",
12      "tamano": 500.0,
13      "ciudad": {
14        "id": 1,
15        "codigo": 1001,
16        "nombre": "Bogotá"
17      }
18    }
19  }
20 ]
```

-Requerimiento funcional 10:

Registrar ingreso de productos a bodega:

Para este requerimiento hicimos dos pruebas, una es recepción hoy la cual prueba una parte importante del requerimiento (principalmente para detectar errores y verificar la respuesta que da) y el otro es el requerimiento completo y se llama registrar productos en bodega. **Recepción hoy:**

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

- Request:** Method: GET, URL: `http://localhost:8080/proyecto/recepciones/hoy?bodega_id=1&proveedor_id=1&fechaString=2024-11-3`
- Query Params:**

Key	Value	Description
bodega_id	1	
proveedor_id	1	
fechaString	2024-11-3	
- Response:** Status: 200 OK, Time: 67 ms, Size: 571 B. The response body is a JSON object:

```
1 {
2   "id": 22,
3   "id_bodega": {
4     "id": 1,
5     "nombre": "Bodega Central",
6     "tamano": 1000.0,
7     "sucursal": {
8       "id": 1,
9       "nombre": "Sucursal Centro",
10      "telefono": "3200000001",
11      "direccion": "Carrera 7 #34-16",
12      "tamano": 900.0,
13      "ciudad": {
14        "id": 1,
15        "codigo": 1004,
16        "nombre": "Bogotá"
```

Registrar producto en bodega:

Requerimientos funcionales de consulta:

-Requerimiento funcional de consulta 1:

SuperAndes Copy 2 / Bodega / Registrar producto en bodega

GET http://localhost:8080/proyecto/bodegas/registroProductoBodega?orden_compra_id=1&bodega_id=1 Send

Params Authorization Headers (6) Body Scripts Settings

Query Params

<input checked="" type="checkbox"/> Key	Value	Description
<input checked="" type="checkbox"/> orden_compra_id	1	
<input checked="" type="checkbox"/> bodega_id	1	
<input type="checkbox"/> Key	Value	Description

Body Cookies Headers (5) Test Results

200 OK · 373 ms · 206 B Save Response

Pretty Raw Preview Visualize Text

```
1 Producto registrado en bodega exitosamente
```

Mostrar índice de ocupación de cada una de las bodegas de una sucursal: -Requerimiento funcional de consulta 2:

SuperAndes Copy / Sucursal / Índice de ocupación

GET <http://localhost:8080/proyecto/sucursales/consulta> Send

Params Authorization Headers (6) Body Scripts Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL ☒ JSON Beautify

```
1 [1, 2, 3]
2
```

Body Cookies Headers (5) Test Results

200 OK · 103 ms · 192 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "porcentajeOcupacion": 66.6
3 }
```

Mostrar los productos que cumplen con cierta característica:

SuperAndes Copy / Producto

Consultar producto por característica

SaveShare

GEThttp://localhost:8080/proyecto/productos/consultaCaracteristica?precioMinimo=10&precioMaximo=30000&fechaInicio=2001-01-01&fechaFin=2040-01-01&idCategoria=1Send

ParamsAuthorizationHeaders (6)BodyScriptsSettingsCookie

Query Params

<input checked="" type="checkbox"/>	Key	Value	Description	Bulk Edit
<input checked="" type="checkbox"/>	precioMinimo	10		
<input checked="" type="checkbox"/>	precioMaximo	30000		
<input checked="" type="checkbox"/>	fechaInicio	2001-01-01		
<input checked="" type="checkbox"/>	fechaFin	2040-01-01		
<input checked="" type="checkbox"/>	idCategoria	1		
<input type="checkbox"/>	Key	Value	Description	

BodyCookiesHeaders (5)Test Results200 OK51 ms466 BSave Response

PrettyRawPreviewVisualizeJSON

```
1 {
2   "costo_bodega": 4200.0,
3   "peso_empaque": 15.0,
4   "cantidad_presentacion": 1000.0,
5   "unidad_medida": 0,
6   "presentacion": "paquetón de 6 paquetes de 200 gr",
7   "fecha_expiration": "2021-02-01 00:00:00.0",
8   "precio_venta": 9000.0,
9   "categoria_id": 1,
10  "nombre": "Papas fritas",
11  "codigo_barras": "4ASADf0f0f",
12  "volumen_empaque": 150.0
13 }
```

-Requerimiento funcional de consulta 3:
Inventario de productos en bodega:

-Requerimiento funcional de consulta 4:

SuperAndes Copy 2 / Bodega / Consulta productos de bodega

GET http://localhost:8080/proyecto/bodegas/consultaBodega?bodega_id=2&sucursal_id=2 Send

Params • Authorization Headers (6) Body Scripts Settings Cookies

Query Params

<input checked="" type="checkbox"/>	Key	Value	Description
<input checked="" type="checkbox"/>	bodega_id	2	
<input checked="" type="checkbox"/>	sucursal_id	2	
	Key	Value	Description

Body Cookies Headers (5) Test Results 200 OK • 273 ms • 257 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "nivel_minimo": 30,
3   "costo_promedio": 8800.0,
4   "producto": "Carne de Res",
5   "cantidad_existente": 60
6 }
```

Mostrar sucursales en las que hay disponibilidad de un producto:

SuperAndes Copy / Bodega / Consulta de sucursal producto

GET http://localhost:8080/proyecto/bodegas/consultarSucursalProducto?producto_id=2 Send

Params • Authorization Headers (6) Body Scripts Settings Cookies

Query Params

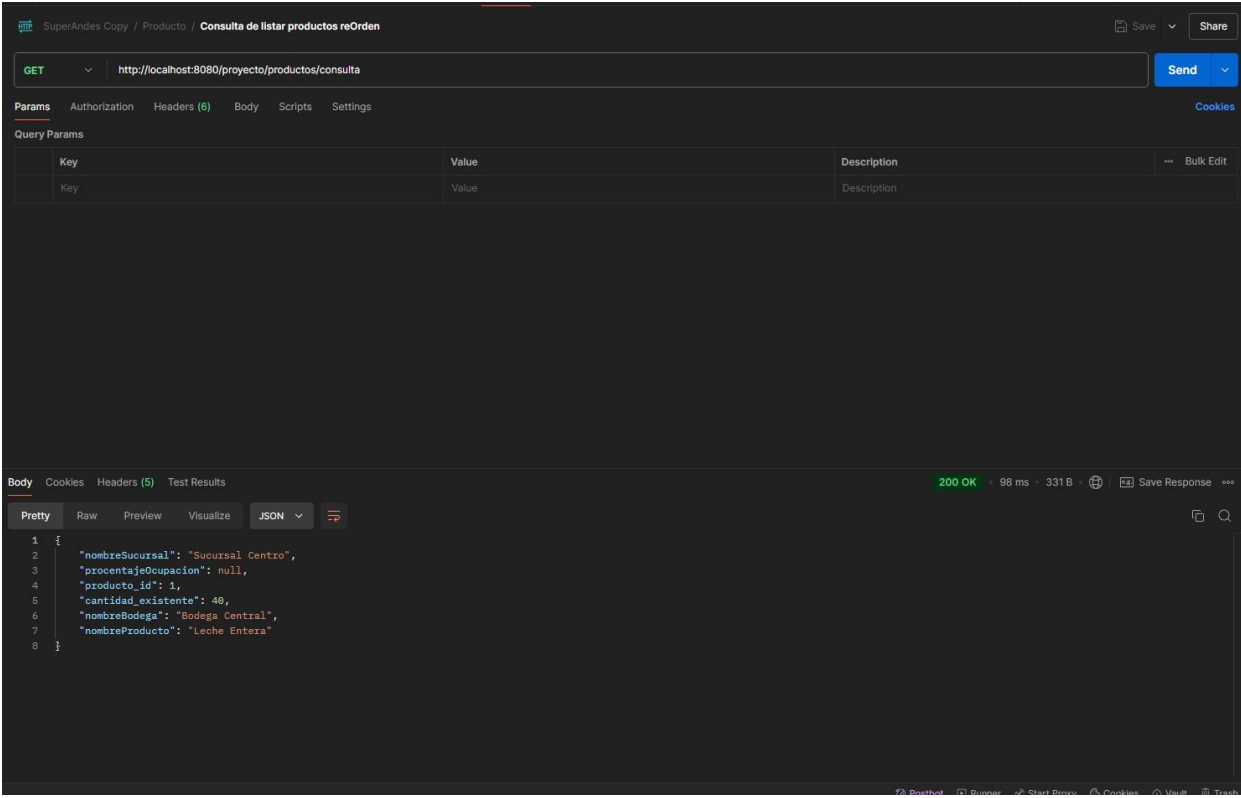
<input checked="" type="checkbox"/>	Key	Value	Description
<input checked="" type="checkbox"/>	producto_id	2	
	Key	Value	Description

Body Cookies Headers (5) Test Results 200 OK • 33 ms • 261 B Save Response

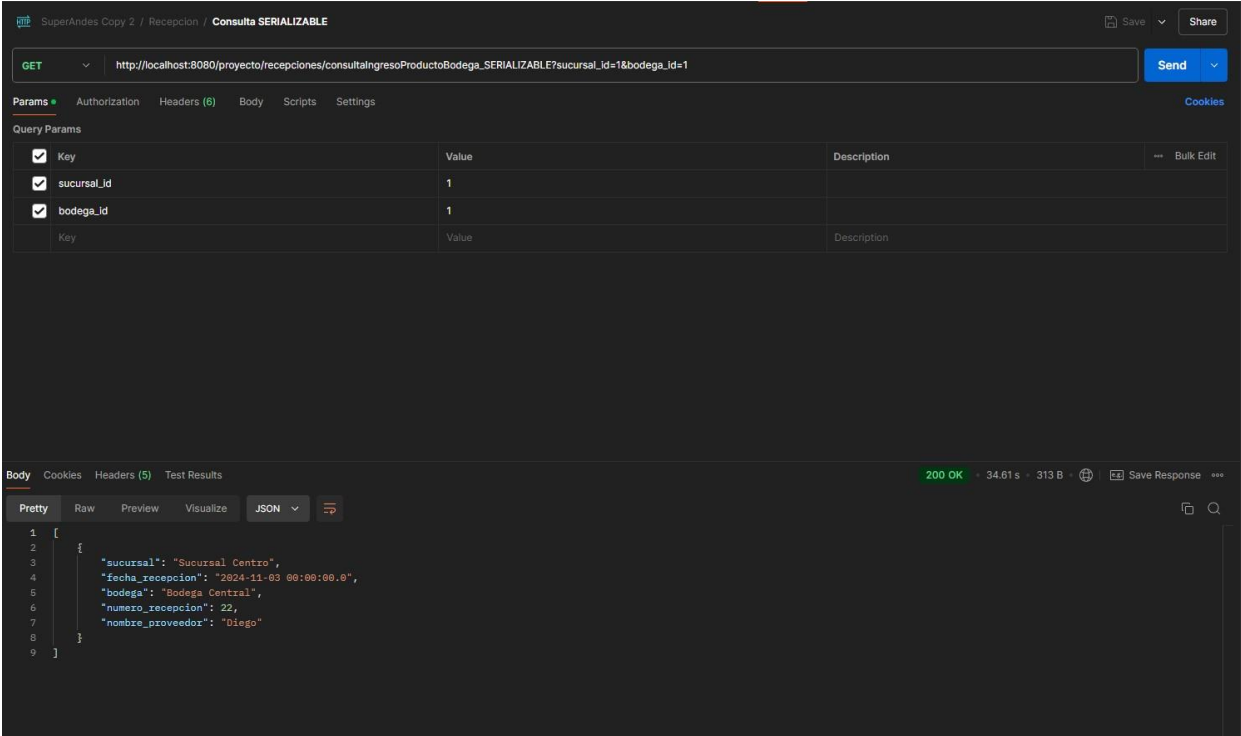
Pretty Raw Preview Visualize JSON

```
1 {
2   "direccion": "Avenida 19 #153-30",
3   "telefono": "3100000002",
4   "nombre": "Bodega Norte",
5   "tamano": 800.0
6 }
```

-Requerimiento funcional de consulta 5:



Mostrar todos los productos que requieren una orden de compra: -Requerimiento funcional de consulta 6:
Consulta de documentos de ingreso de productos a bodega - Serializable:



-Requerimiento funcional de consulta 7:

Consulta de documentos de ingreso de productos a bodega – Read Committed:

SuperAndes Copy 2 / Recepcion / Consulta READ_COMMITED

GET

http://localhost:8080/proyecto/recepciones/consultaIngresoProductoBodega_READ_COMMITTED?bodega_id=1&sucursal_id=1

Send

Params

Authorization

Headers (6)

Body

Scripts

Settings

Cookies

Query Params

Key	Value	Description
bodega_id	1	
sucursal_id	1	

Body

Cookies

Headers (5)

Test Results

200 OK

30.04 s

313 B

Save Response

Pretty

Raw

Preview

Visualize

JSON

```
1 [
2   {
3     "bodega": "Bodega Central",
4     "fecha_recepcion": "2024-11-03 00:00:00.0",
5     "nombre_proveedor": "Diego",
6     "numero_recepcion": 22,
7     "sucursal": "Sucursal Centro"
8   }
9 ]
```

3. Escenarios de prueba

```
1 row inserted.

Error starting at line : 8 in command -
INSERT INTO proveedores (id, NIT, direccion, nombrePersonaC, telefonoPersonaC)
VALUES (1, '987654321', 'Av. Secundaria 200', 'Maria Garcia', '555-5678')
Error report -
ORA-00001: unique constraint (ISIS2304A01202420.SYS_C001329426) violated

1 row inserted.

Error starting at line : 16 in command -
INSERT INTO productos (id, codigoBarras, nombre, costoBodega, precioVenta, presentacion, cantidadPresentacion, unidadMedida, volumenEmpaque, pesoEmpaque)
VALUES (1, 'a0a0a0a0a0', 'Cereal Chocoflakes', 3000, 5000, 'Caja', 500, 1, 100, 10, TO_DATE('2026-12-12', 'YYYY-MM-DD'), 2)
Error report -
ORA-00001: unique constraint (ISIS2304A01202420.SYS_C001329441) violated

1 row inserted.
```

```

1 row inserted.

1 row inserted.

1 row inserted.

Error starting at line : 31 in command -
INSERT INTO bodegas (nombre, tamano, id_sucursal)
VALUES ('Bodega Secundaria', 150, 999)
Error report -
ORA-02291: integrity constraint (ISIS2304A01202420.FK_BODEGA_SUCURSAL) violated - parent key not found

1 row inserted.

1 row inserted.

1 row inserted.

Error starting at line : 50 in command -
INSERT INTO info_recepcion (id_recepcion, id_producto, cantidadRecibida, costoRecibido)
VALUES (1, 1, 10, 9000)
Error report -
ORA-00001: unique constraint (ISIS2304A01202420.PK_INFO_RECEPCION) violated

1 row inserted.

1 row inserted.

1 row inserted.

1 row updated.

1 row inserted.

```

ESCENARIOS DE PRUEBAS DE CONCURRENCIA:

Escenario 1

Para el escenario 1 llegamos a las siguientes conclusiones:

Se lanza la consulta `consultaIngresoProductoBodega_SERIALIZABLE`, que busca obtener los documentos de ingreso de productos en una bodega específica.

Esta operación es transaccional y se ejecuta en un nivel de aislamiento `SERIALIZABLE`, lo cual es el nivel más estricto. Este nivel de aislamiento garantiza que no habrá interferencias de otras transacciones y asegura la consistencia en los datos durante la ejecución de RFC6.

La consulta realiza una primera obtención de los datos (almacenados en `respuestaConsultaIngreso`), espera 30 segundos (`Thread.sleep(30000)`), y luego vuelve a realizar la consulta en el repositorio antes de retornar los datos.

Ejecución Concurrente de RF10:

Antes de que transcurran los 30 segundos de espera en RFC6, se ejecuta la operación registroProductoBodega (RF10), cuyo propósito es registrar el ingreso de productos en la bodega. Esta operación también es transaccional, con un nivel READ COMMITTED.

RF10 busca actualizar el estado del inventario en la bodega y marcar la orden de compra como "ENTREGADA".

Paso a Paso en la Línea de Tiempo

T=0 segundos:

RFC6 comienza a ejecutarse y realiza la primera consulta en el repositorio para obtener los documentos de ingreso de productos. Como está en modo SERIALIZABLE, se "bloquea" un conjunto de datos, evitando que otros procesos modifiquen el mismo conjunto durante la transacción.

T=15 segundos:

Se lanza RF10 mientras RFC6 aún está en espera. Debido a la serialización de RFC6, RF10 no puede modificar o registrar ningún ingreso de producto en la bodega hasta que RFC6 termine su ejecución.

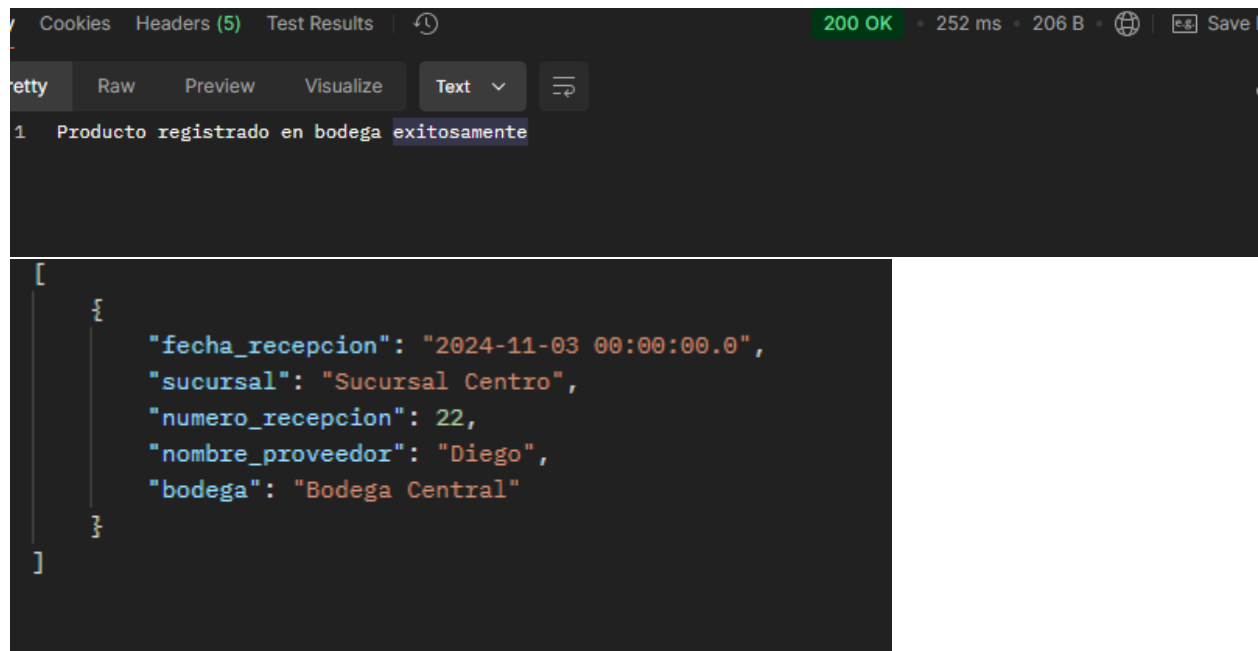
T=30 segundos:

RFC6 realiza la segunda consulta en el repositorio y finaliza, liberando el bloqueo serializable.

T=31 segundos:

RF10 puede ahora registrar los productos en la bodega y finalizar su proceso.

Resultado y Observaciones



```
1 Producto registrado en bodega exitosamente

[
  {
    "fecha_recepcion": "2024-11-03 00:00:00.0",
    "sucursal": "Sucursal Centro",
    "numero_recepcion": 22,
    "nombre_proveedor": "Diego",
    "bodega": "Bodega Central"
  }
]
```

Descripción de lo sucedido:

El componente RF10 no pudo registrar el ingreso de productos en la bodega hasta que finalizó la ejecución de RFC6. Esto se debe a que el nivel de aislamiento SERIALIZABLE de RFC6 bloquea las modificaciones concurrentes de los datos que está leyendo.

Al estar en aislamiento serializable, RFC6 garantiza que los datos que leyó inicialmente permanecen inalterados hasta el final de la transacción. RF10, que requiere modificar estos datos, tiene que esperar a que RFC6 libere el bloqueo.

Escenario 2

1. Pasos para la Ejecución Concurrente de RFC7 y RF10 a través de la Línea de Tiempo

- Paso 1: Se inició RFC7 con el nivel de aislamiento READ COMMITTED, lo cual comenzó una consulta de documentos de ingreso de productos a bodega, incluyendo un temporizador de 30 segundos.
- Paso 2: Mientras RFC7 estaba en ejecución, se inició RF10 para registrar un nuevo ingreso de producto en la bodega.
- Paso 3: RF10 finalizó rápidamente y confirmó la transacción de inserción, mientras RFC7 aún estaba en el proceso de espera.
- Paso 4: Al finalizar los 30 segundos, RFC7 completó su consulta y devolvió los documentos de ingreso de los últimos 30 días.

2. Descripción de lo Sucedido

- ¿RF10 Debíó Esperar a RFC7?

No, RF10 no tuvo que esperar a que terminara RFC7. Dado que el nivel de aislamiento de RFC7 es READ COMMITTED, no bloquea la escritura concurrente de RF10. Esto significa que RF10 pudo registrar el ingreso del producto de manera simultánea sin interferir con la consulta en curso de RFC7.

3. Resultado Presentado por RFC7

- ¿Apareció el Nuevo Documento de Ingreso?

Sí, el nuevo documento de ingreso creado por RF10 apareció en los resultados de la consulta de RFC7. Esto se debe a que RFC7, con el nivel de aislamiento READ COMMITTED, puede ver las transacciones confirmadas durante su ejecución. Al completar el temporizador de 30 segundos, RFC7 incluyó el nuevo documento de ingreso registrado por RF10 en sus resultados.

Conclusión

El nivel de aislamiento READ COMMITTED permitió que la consulta de RFC7 incluyera el documento de ingreso de productos realizado por RF10 de manera simultánea, ya que la transacción de RF10 se confirmó antes de que finalizara la ejecución de RFC7. Esto demuestra que READ COMMITTED permite ver los datos confirmados sin bloquear las transacciones de escritura concurrentes.

SuperAndes / Recepcion / Consulta READ_COMMITTED

GET

http://localhost:8080/proyecto/recepciones/consultaIngresoProductoBodega_READ_COMMITTED?bodega_id=1&sucursal_id=1

Send

Params

Authorization

Headers (6)

Body

Scripts

Settings

Cookies

Query Params

<input checked="" type="checkbox"/> Key	Value	Description	Bulk Edit
<input checked="" type="checkbox"/> bodega_id	1		
<input checked="" type="checkbox"/> sucursal_id	1		
Key	Value	Description	

Body

Cookies

Headers (5)

Test Results

200 OK

30.63 s

313 B

Save Response

Pretty

Raw

Preview

Visualize

JSON

```
1 [
2   {
3     "sucursal": "Sucursal Centro",
4     "fecha_recepcion": "2024-11-03 00:00:00.0",
5     "bodega": "Bodega Central",
6     "nombre_proveedor": "Diego",
7     "numero_recepcion": 22
8   }
9 ]
```

Postbot
Ctrl Alt P

SuperAndes / Bodega / Registrar producto en bodega

GET

http://localhost:8080/proyecto/bodegas/registroProductoBodega?orden_compra_id=1&bodega_id=1

Send

Params

Authorization

Headers (6)

Body

Scripts

Settings

Cookies

Query Params

<input checked="" type="checkbox"/> Key	Value	Description	Bulk Edit
<input checked="" type="checkbox"/> orden_compra_id	1		
<input checked="" type="checkbox"/> bodega_id	1		
Key	Value	Description	

Body

Cookies

Headers (5)

Test Results

200 OK

171 ms

206 B

Save Response

Pretty

Raw

Preview

Visualize

Text

```
1 Producto registrado en bodega exitosamente
```