

# INFORME BASE DE DATOS MONGODB

INTEGRANTES:

IVAN BECERRA

JESSICA CORREA

DOCENTE:

BRAYAN ARCOS

INSTITUTO TECNOLOGICO DEL PUTUMAYO  
BASE DE DATOS Y ALMACENAMIENTO MASIVO  
OCTAVO SEMESTRE

9/09/2024

## Tabla de contenido

1. Resumen Ejecutivo .....	3
2. Introducción.....	4
2.1. Contexto y Motivación.....	4
2.2. Alcance del Informe .....	4
2.3. Objetivos .....	5
3. Metodología.....	6
3.1. Herramientas Utilizadas .....	6
3.2. Procedimientos .....	6
4. Desarrollo del Informe.....	7
4.1. Descripción de la Base de Datos .....	7
4.2. Consultas Realizadas.....	8
4.3. Resultados y explicaciones de Consultas .....	11
4.4. Diseño de la Base de Datos .....	13
5. Análisis y Discusión .....	14
5.1. Interpretación de Resultados .....	14
6. Conclusiones.....	15
7. Recomendaciones .....	16
Índices: .....	16
Alertas de inventario y ventas: .....	16
8. Link de repositorio.....	17
9. Referencias .....	17

## **1. Resumen Ejecutivo**

Este informe tiene como objetivo el proceso de diseño, implementación y análisis de una base de datos NoSQL desarrollada en MongoDB, creada para gestionar la información de un supermercado. La base de datos almacena datos de clientes, productos y ventas, permitiendo realizar consultas eficientes para mejorar la gestión operativa y tomar decisiones informadas. A través de esta implementación, se demostró la utilidad de MongoDB para manejar grandes volúmenes de datos de manera flexible, adaptándose a las necesidades del negocio.

Link del repositorio: [https://github.com/jessica123c/BD\\_NOSQL](https://github.com/jessica123c/BD_NOSQL)

## **2. Introducción**

### **2.1.Contexto y Motivación**

En el entorno actual de los supermercados, la gestión eficiente de los datos es fundamental para el éxito operativo. Esta base de datos que realizamos nos permite manejar grandes volúmenes de información, como ventas y productos, con mayor flexibilidad en comparación con bases de datos relacionales tradicionales. Este informe se centra en la implementación de una base de datos NoSQL para un supermercado, diseñada para optimizar la gestión de clientes, productos y transacciones de ventas.

### **2.2.Alcance del Informe**

Este informe cubre los siguientes aspectos clave relacionados con la implementación de una base de datos NoSQL en MongoDB para un supermercado:

**Diseño de la Base de Datos NoSQL:** detallar la estructura de las colecciones usadas que nos permitirán almacenar datos importantes para sus operaciones. Este diseño incluye la definición de las claves primarias, relaciones entre colecciones y los campos necesarios para cumplir con los requisitos operativos.

**Consultas para Extraer Información Clave:** realizaremos consultas con el fin de extraer datos útiles para una mejor gestión como listados disponibles, búsquedas y cálculos de ventas, proporcionando una base sólida para la toma de decisiones

**Justificación del Diseño de las Colecciones:** Se explica por qué se han seleccionado ciertos campos en cada colección y cómo estas decisiones impactan en la eficiencia de la base de datos.

### **2.3.Objetivos**

**Implementar una Base de Datos NoSQL en MongoDB:** Diseñar una base de datos que gestione de manera eficiente los datos de los clientes, productos y ventas. Con el objetivo de facilitar las operaciones del supermercado, mejorando velocidad y flexibilidad en la información

**Ejecutar Consultas Específicas:** Las consultas son fundamentales para obtener la información necesaria para la gestión diaria del supermercado. Se ejecutan consultas como la búsqueda de productos, el cálculo del total de ventas y el seguimiento de clientes, apoyando directamente la toma de decisiones comerciales.

**Analizar el Diseño y su Impacto en la Operatividad:** Evaluar cómo el diseño de la base de datos puede contribuir a la eficiencia operativa. Las consultas realizadas se analizan en términos de rapidez, flexibilidad y capacidad para manejar grandes volúmenes de datos sin afectar rendimiento.

### 3. Metodología

#### 3.1.Herramientas Utilizadas

**MongoDB:** Utilizado para almacenar los datos del supermercado, con un enfoque en la flexibilidad de su estructura NoSQL.

**MongoDB Compass:** Herramienta gráfica que permite visualizar la base de datos y ejecutar consultas, ayudando en el análisis de los resultados.

**GitHub:** Se utiliza para almacenar el código y la documentación relacionada con la base de datos, asegurando un fácil acceso y control de versiones para futuras actualizaciones.

**JavaScript:** Lenguaje utilizado para automatizar y ejecutar las consultas, facilitando la interacción con MongoDB a través de código.

#### 3.2.Procedimientos

**Diseño de la Base de Datos:** Se crearon tres colecciones principales:

**Clientes:** Almacena la información básica de los clientes, necesaria para identificar y analizar sus hábitos de compra.

**Productos:** Contiene los detalles de los productos del supermercado, como su nombre, categoría, precio y stock disponible.

**Ventas:** Almacena las transacciones realizadas, conectando a los clientes con los productos comprados, la fecha de la compra y el total de la venta.

**Inserción de Datos:** Se añadieron datos simulados o reales para poblar las colecciones, lo que nos permite realizar pruebas necesarias para verificar el correcto funcionamiento de la base de datos y las consultas.

**Consultas NoSQL:** Se llevaron a cabo consultas sobre clientes, productos y ventas para traer información clave, como el listado, totales acumulados y compras realizadas.

**Análisis de los Resultados:** Los datos obtenidos de las consultas se evaluaron para ver cómo pueden apoyar la operación eficiente del supermercado. Esto analiza la gestión del inventario, el comportamiento de los clientes y el rendimiento financiero.

## **4. Desarrollo del Informe**

### **4.1.Descripción de la Base de Datos**

Esquema de la Base de Datos Exportada

Clientes:

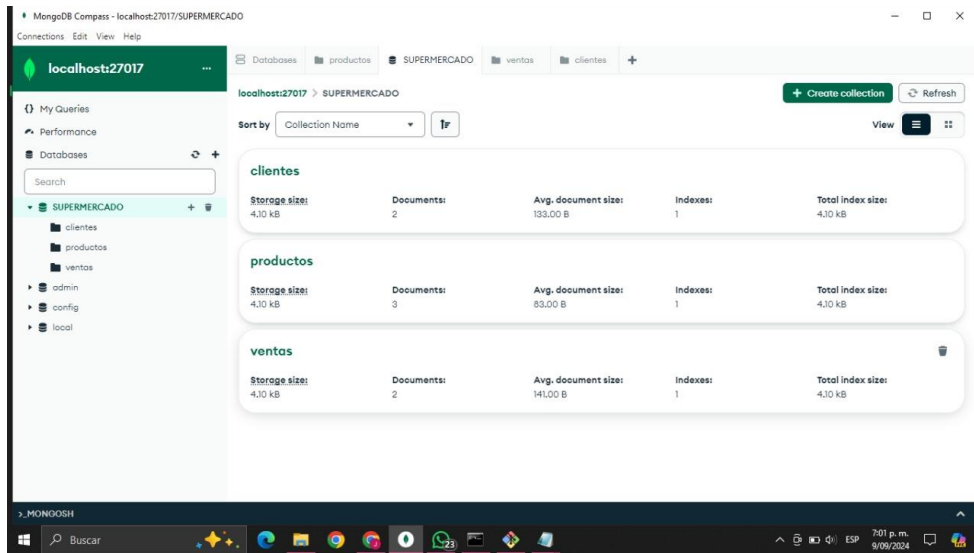
Campos: \_id, nombre, email, teléfono, dirección.

Productos:

Campos: \_id, nombre, categoría, precio, stock.

Ventas:

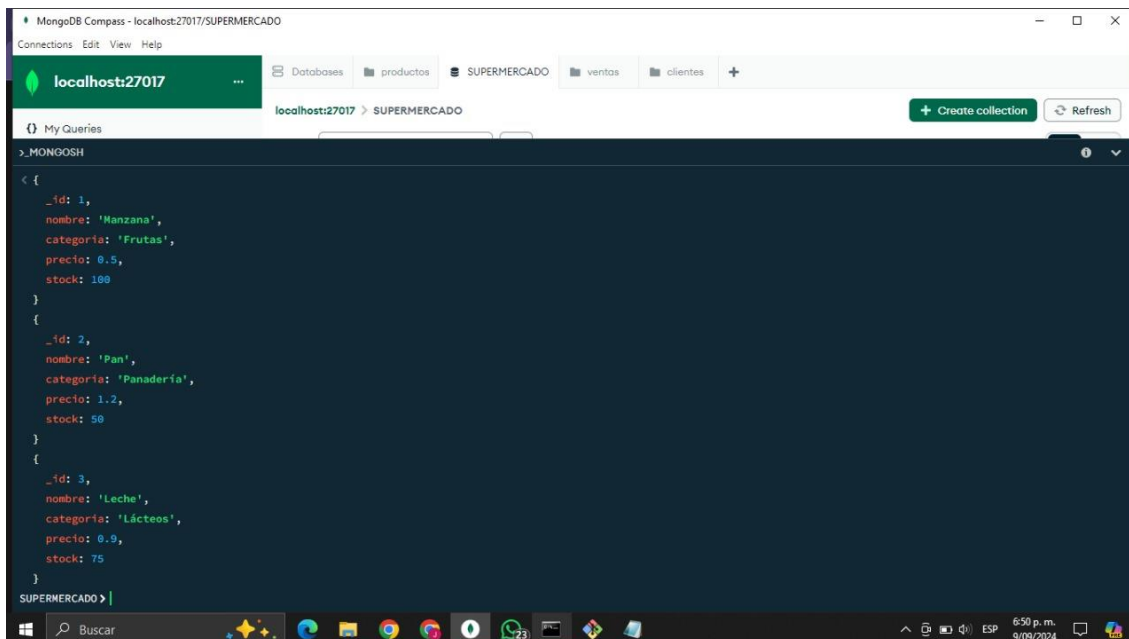
Campos: \_id, cliente \_id, productos (lista con ids), total, fecha.



## Consultas NoSQL

### 4.2.Consultas Realizadas

Consulta 1: Obtener todos los productos disponibles



Consulta 2:



## Buscar un producto específico por nombre

The screenshot shows the MongoDB Compass interface for the 'localhost:27017/SUPERMERCADO' database. The 'productos' collection is selected. The query bar contains the text: 'Type a query: { field: 'value' } or [Generate query](#)'. Below the query bar, the 'Documents' tab shows a single document with the following fields: `_id: 3`, `nombre: "Leche"`, `categoría: "Lácteos"`, `precio: 8.9`, and `stock: 75`. The bottom terminal window shows the following command and result:

```
> db.productos.find({ nombre: "Leche" })
< {
  _id: 3,
  nombre: 'Leche',
  categoría: 'Lácteos',
  precio: 8.9,
  stock: 75
}
```

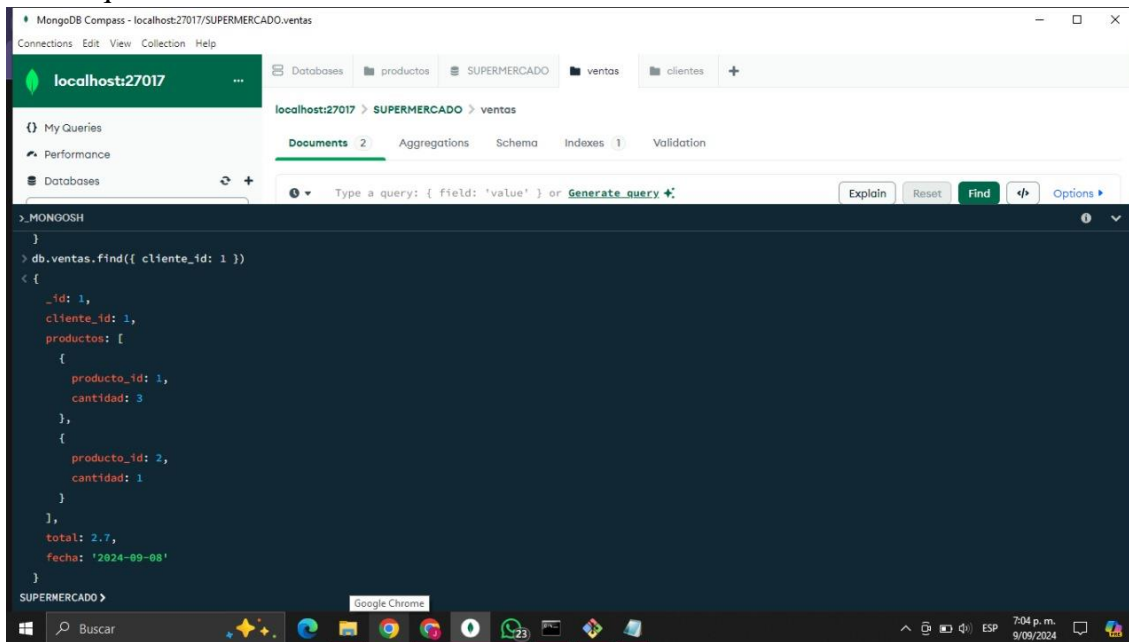
## Consulta 3 : Busca todos los clientes

The screenshot shows the MongoDB Compass interface for the 'localhost:27017/SUPERMERCADO' database. The 'clientes' collection is selected. The query bar contains the text: 'Type a query: { field: 'value' } or [Generate query](#)'. Below the query bar, the 'Documents' tab shows two documents with the following fields: `_id: 1`, `nombre: 'Juan Pérez'`, `email: 'juan.perez@example.com'`, `telefono: '123456789'`, and `direccion: 'Calle Falsa 123'`; and `_id: 2`, `nombre: 'María López'`, `email: 'maria.lopez@example.com'`, `telefono: '987654321'`, and `direccion: 'Avenida Siempre Viva 456'`. The bottom terminal window shows the following command and result:

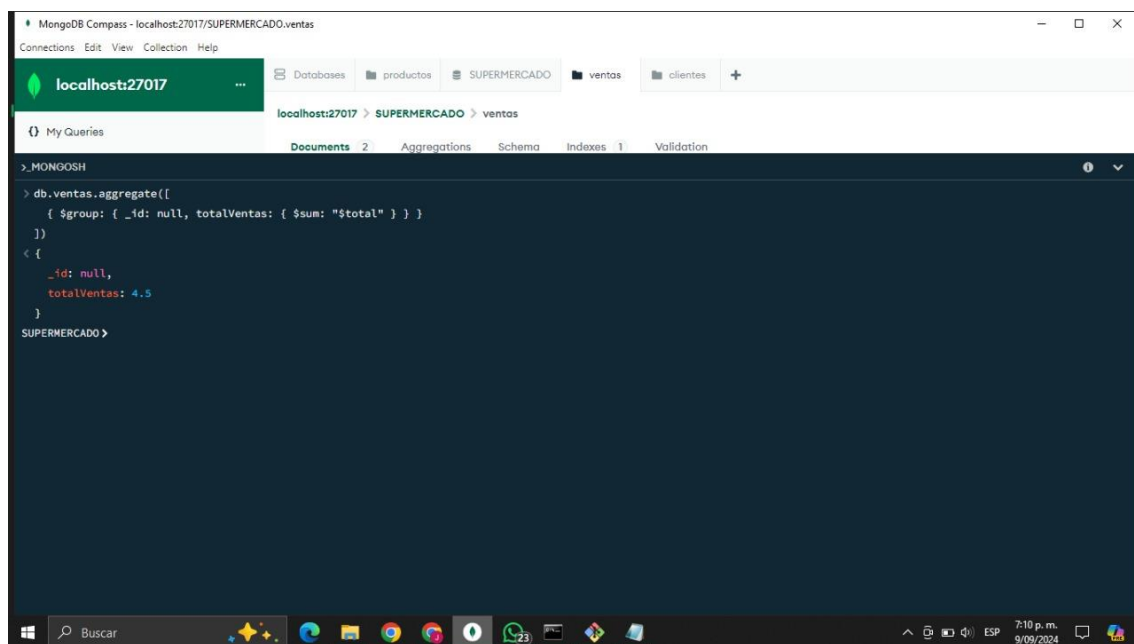
```
> db.clientes.find()
< {
  _id: 1,
  nombre: 'Juan Pérez',
  email: 'juan.perez@example.com',
  telefono: '123456789',
  direccion: 'Calle Falsa 123'
}
{
  _id: 2,
  nombre: 'María López',
  email: 'maria.lopez@example.com',
  telefono: '987654321',
  direccion: 'Avenida Siempre Viva 456'
}
```

## Consulta 4:

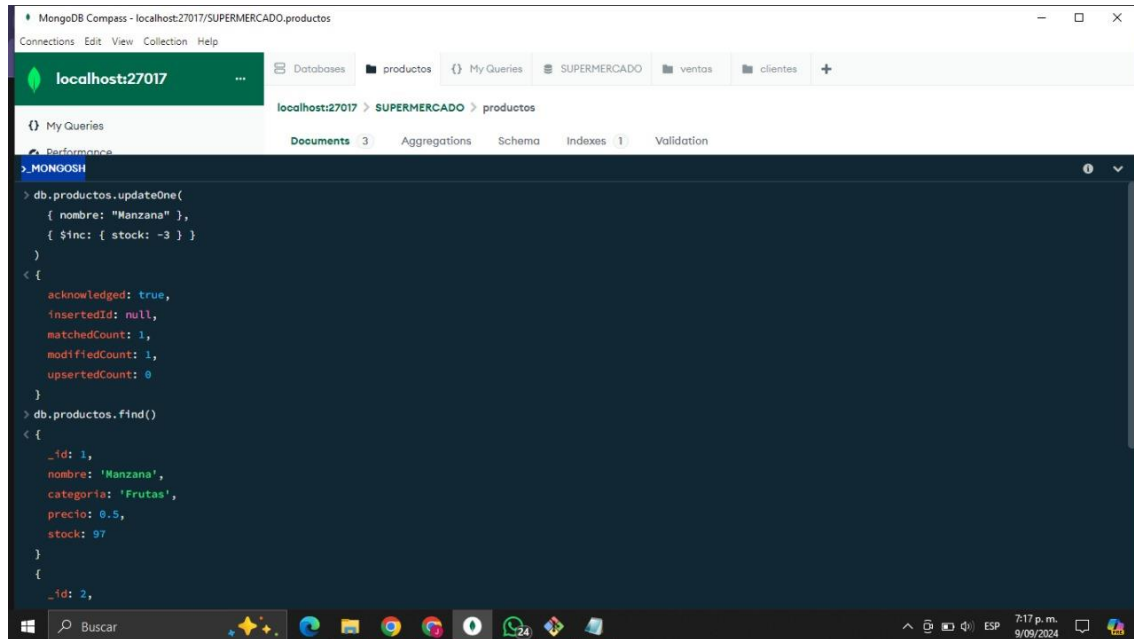
ventas que hizo un cliente



Consulta 5: Calcula el total de las ventas



Consulta 6 :Actualizar el stock de un producto (por ejemplo, reducir el stock de la manzana)

The screenshot shows the MongoDB Compass interface. The top bar indicates the connection to 'localhost:27017/SUPERMERCADO'. The left sidebar shows the 'productos' collection selected. The main panel displays the 'Documents' tab for the 'productos' collection. A JavaScript query is entered in the command line: 

```
> db.productos.updateOne(
  { nombre: "Manzana" },
  { $inc: { stock: -3 } }
)
< {
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Below the command, the result of the update is shown, followed by a `db.productos.find()` query. The output shows two documents: 

```
< {
  _id: 1,
  nombre: 'Manzana',
  categoria: 'Frutas',
  precio: 0.5,
  stock: 97
}
{
  _id: 2,
```

### 4.3.Resultados y explicaciones de Consultas

#### Consulta 1

Resultado: Esta consulta devuelve todos los productos que están almacenados en la colección "productos". Se muestran todos los campos (`_id`, nombre, categoría, precio, stock) de cada producto en la base de datos.

Explicación: La función `find()` sin parámetros busca y devuelve todos los documentos en la colección "productos". Esto te permite obtener una lista completa del inventario disponible.

#### Consulta 2

Resultado: Esta consulta devolvería un documento con la información del producto llamado "Manzana", incluyendo su `_id`, categoría, precio y stock.

Explicación: El filtro {nombre: "Manzana"} busca todos los productos cuyo campo nombre sea exactamente igual a "Manzana". Si existen varias coincidencias, todas serán retornadas.

#### Consulta 3

Resultado: Se devuelve una lista de todos los clientes almacenados en la colección "clientes", con sus campos (\_id, nombre, email, teléfono, dirección).

Explicación: Similar a la consulta 1, find() sin ningún filtro devuelve todos los clientes registrados en la base de datos.

#### Consulta 4

Resultado: Esta consulta devuelve un solo documento con el total acumulado de todas las ventas realizadas, sumando el valor del campo total de cada venta.

Explicación: La etapa \$group en la consulta de agregación agrupa todos los documentos de la colección "ventas" (al usar \_id: null, no agrupa por ningún campo específico). Luego, usa \$sum para calcular la suma de los valores del campo total en todas las ventas.

#### Consulta 5

Resultado: Se devuelve una lista de todas las ventas realizadas por el cliente con el cliente\_id igual a 1. Los documentos incluirán los productos comprados, la fecha de la venta y el total.

Explicación: El filtro {cliente\_id: 1 } busca todas las ventas donde el campo cliente\_id sea igual a 1. Esto permite consultar todas las transacciones asociadas a un cliente específico.

## Consulta 6

Resultado: El stock del producto "Manzana" se reducirá en 3 unidades.

Explicación: La función `updateOne()` actualiza un solo documento en la colección "productos". El filtro `{nombre: "Manzana" }` selecciona el documento del producto llamado "Manzana". El operador `$inc` reduce el valor del campo stock en 3. Esto es útil para actualizar el inventario después de una venta.

### **4.4.Diseño de la Base de Datos**

#### 1. Consideraciones de Diseño

Claves Primarias: En MongoDB, el campo `_id` se utiliza como clave primaria de manera predeterminada, esto garantiza que cada documento tenga un identificador único. siendo útil para realizar búsquedas rápidas y asegura la integridad de los datos. Como MongoDB genera automáticamente estos campos, no es necesario definir claves manualmente, pero si se pueden personalizar si es necesario usar una clave generada por el usuario (por ejemplo, un código de producto).

Elección de Campos: Los campos seleccionados en cada colección corresponden a las necesidades operativas del supermercado. Como lo puede ser:

Clientes: Los campos como nombre, email y teléfono importantes para gestionar las relaciones con los clientes

Productos: Los campos como nombre, categoría, precio y stock son necesarios para una mejor gestión de inventario y fijación de precios. El campo stock es importante para llevar un control en tiempo real de las existencias y evitar la sobreventa.

Ventas: Los campos cliente \_id y productos (lista de IDs de productos) conectan las ventas con los clientes y los productos específicos comprados. El campo total y la fecha permiten realizar análisis financieros y de temporalidad.

## **5. Análisis y Discusión**

### **5.1.**

### **5.2. Interpretación de Resultados**

Los resultados obtenidos a partir de las consultas realizadas en MongoDB muestran cómo se optimizan las operaciones del supermercado y se cumplen los objetivos planteados:

**Gestión del Inventario:** Las consultas sobre productos disponibles y la actualización automática del stock permiten un control preciso del inventario en tiempo real, asegurando que los productos estén siempre disponibles para los clientes. Esto mejora la eficiencia en la gestión de existencias.

**Atención al Cliente:** La posibilidad de buscar productos específicos y consultar ventas por cliente permite ofrecer un servicio más personalizado. Esto ayuda a identificar las preferencias de los clientes y a mejorar su experiencia de compra, contribuyendo a la fidelización.

**Análisis Financiero:** El cálculo del total de ventas y el seguimiento de compras por cliente proporciona información clave para evaluar el rendimiento financiero y tomar decisiones estratégicas basadas en datos, como identificar a los mejores clientes y ajustar las estrategias de ventas.

En conjunto, estas consultas nos permiten una mayor eficiencia operativa, mejoran la gestión del inventario y optimizan la atención al cliente, cumpliendo con los objetivos del informe y mostrando la capacidad de MongoDB para manejar datos complejos de manera flexible y eficiente.

## 6. Conclusiones

MongoDB ha demostrado ser una solución eficiente para la gestión de datos en un entorno como el de un supermercado. Con una capacidad de manejar grandes volúmenes de datos y permite realizar consultas flexibles lo hace ideal para operaciones cotidianas y análisis avanzados. Además, la flexibilidad de su esquema sin necesidad de estructuras rígidas permite adaptarse a las necesidades cambiantes del negocio.

El uso de MongoDB ha simplificado el proceso de consultas, permitiendo obtener información clave sobre clientes, productos y ventas de forma rápida. La posibilidad de hacer operaciones de agregación, como el cálculo del total de ventas, es muy útil para el análisis financiero del negocio.

## **7. Recomendaciones**

### **Índices:**

Implementar índices adicionales en campos clave (como nombre en productos o cliente\_id en ventas) puede mejorar el rendimiento de las consultas más frecuentes. Los índices en MongoDB permiten que las búsquedas sean más rápidas, especialmente cuando se trata de grandes volúmenes de datos.

### **Alertas de inventario y ventas:**

Desarrollar un sistema de alertas basado en el inventario podría ser una mejora útil. Cuando el stock de un producto está por debajo de un umbral determinado, se puede generar una alerta para que el personal de compras realice un reabastecimiento a tiempo. De manera similar, un sistema de alertas basado en ventas puede detectar picos en la demanda de ciertos productos y ajustar la oferta de manera proactiva.



## **8. Link de repositorio**

[https://github.com/jessica123c/BD\\_NOSQL](https://github.com/jessica123c/BD_NOSQL)

## **9. Referencias**

Documentación de MongoDB.

Artículos y tutoriales sobre el diseño de bases de datos NoSQL.