

MONGODB

Presentado Por:

Daniel Felipe Mora Rosas

Jhonier Arley Pasos Perengues

Instituto Tecnológico Del Putumayo

Tecnología En Desarrollo De Software

Mocoa – Putumayo

2024

Tabla De Contenido

Resumen Ejecutivo	3
Introducción	4
Contexto y Motivación	4
Alcance del Informe.....	4
Objetivo.....	4
Metodología	5
Herramientas utilizadas.....	5
Procedimiento	6
Desarrollo.....	7
Descripción de la Base de Datos.....	7
Consultas NoSQL	9
Análisis y Diseño	12
Conclusión	13
Recomendaciones	14
REFERENCIAS.....	15

Resumen Ejecutivo

En este informe se expuso sobre el uso del MongoDB en el sistema conceptual de la tienda en línea. Se presentaron los aspectos teóricos en el modelo de la base de datos, ejemplos prácticos de solicitudes de NOSQL, la optimización del sistema de la base de datos, y también la posibilidad de realizar diferentes funciones de mongoDB tanto su versión de comandos como el gráfico “Compass”.

Introducción

Contexto y Motivación

Las bases de datos NoSQL son usadas como una alternativa fuerte a las bases de datos relacionales tradicionales, donde nos ofrecen soluciones más dinámicas y escalables para aplicaciones modernas, aplicaciones web o móviles, y sistemas de análisis en tiempo real. A través del aprendizaje NoSQL con el aplicativo MongoDB, aplicamos nuestros conocimientos para así probar cómo estas tecnologías pueden superar los límites de las estructuras de datos convencionales y mejorar el rendimiento de sistemas complejos.

Alcance del Informe

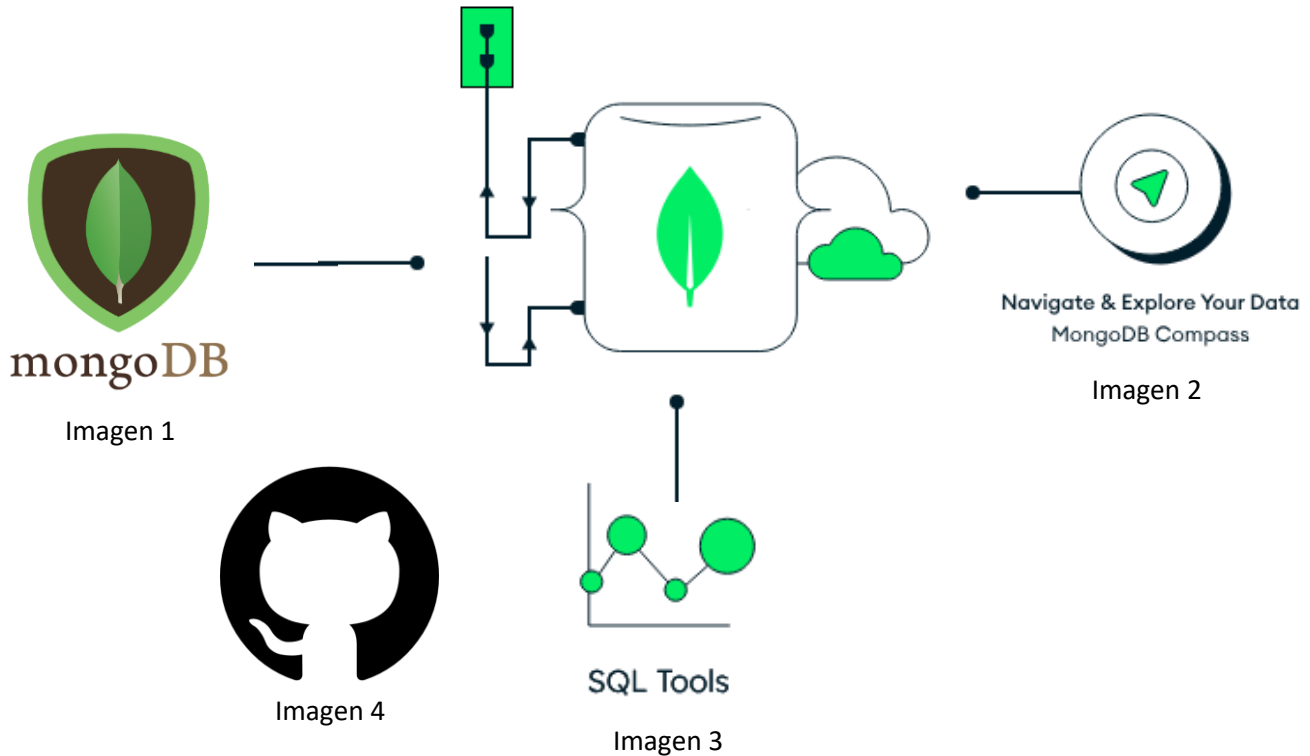
Se cubrirán aspectos esenciales como el diseño de la base de datos, las consultas NoSQL, y la gestión y modelado de datos. El informe incluirá ejemplos prácticos de consultas, análisis de los resultados obtenidos y las ventajas de utilizar NoSQL en comparación con los sistemas relacionales. También se aplicarán conocimientos de las bases de datos NoSQL utilizando un ejemplo práctico de una tienda en línea, con colecciones representativas como clientes, productos, pedidos y transacciones. A lo largo del informe, se detallarán las consultas NoSQL ejecutadas, los criterios utilizados para la selección de datos y las técnicas de agregación aplicadas para extraer información importante.

Objetivo

Investigar el potencial de MongoDB en la manipulación de grandes volúmenes de datos no estructurados, aplicando técnicas específicas para la organización, consulta y análisis de datos en una base NoSQL. Se busca proporcionar una comprensión práctica de cómo las características únicas de MongoDB, como la flexibilidad en el modelado de datos y la capacidad de escalar horizontalmente, pueden ser aprovechadas para mejorar la eficiencia de los sistemas de bases de datos en aplicaciones reales.

Metodología

Herramientas utilizadas



MongoDB: Utilizado como el sistema de base de datos NoSQL para almacenar y gestionar la información de la tienda en línea. MongoDB fue elegido por su capacidad para manejar grandes volúmenes de datos no estructurados y su flexibilidad en la definición de esquemas. (Imagen 1)

MongoDB Compass: Herramienta gráfica oficial de MongoDB que facilita la administración y visualización de bases de datos. Fue utilizada para crear, modificar y consultar las colecciones de la base de datos, así como para exportar datos de manera gráfica. (Imagen 2)

MongoDB Command Line Database Tools: Empleado para exportar las colecciones en formato JSON para respaldos y migración de datos. (Imagen 3)

GitHub: Utilizado como plataforma de control de versiones para almacenar los backups de la base de datos, permitiendo un historial de cambios y acceso remoto a los respaldos. (Imagen 4)

Procedimiento

1. **Creación de la base de datos:** Se configuraron las colecciones clientes, pedidos, productos, y transacciones en MongoDB. Cada colección fue diseñada para reflejar la estructura de datos de una tienda en línea.
2. **Población de datos:** Los datos se insertaron utilizando las funciones insertOne y insertMany en MongoDB, garantizando la correcta relación entre clientes, productos y pedidos mediante el uso de claves foráneas.
3. **Consultas de datos:** Se realizaron consultas básicas y avanzadas con MongoDB Compass para obtener información detallada sobre los clientes, productos y transacciones.
4. **Exportación de las colecciones:** A través del complemento del Tools, las colecciones fueron exportadas en formato JSON para respaldos y migración de datos.
5. **Subida de Backups a GitHub:** Los backups de las colecciones fueron subidos a GitHub, permitiendo la conservación y acceso seguro a los datos, así como la posibilidad de compartir el historial de cambios con otros colaboradores.
6. **Optimización:** Se aplicaron índices en campos clave como cliente_id y producto_id, mejorando el rendimiento de las consultas y asegurando la eficiencia del sistema.

Desarrollo

Descripción de la Base de Datos

La base de datos para la tienda en línea está compuesta por cuatro colecciones principales: clientes, pedidos, productos, y transacciones. A continuación, primero hay una tabla donde muestras los comandos principales para agregar datos, obtener datos para su lectura, actualizar datos y eliminar datos

Operación	Comando Mongo	Descripción
CREATE (INSERTAR)	db.collection.insertOne()	Insertar un solo dato
	db.collection.insertMany()	Para varios valores
READ (LEER/BUSCAR)	db.collection.find()	Para mostrar los datos de esa colección
	db.collection.findOne()	Busca un único documento que cumpla los criterios
UPDATE	db.collection.updateOne(<filter>, <update>, <options>)	Actualiza el primer documento que coincide con el filtro
	db.collection.updateMany(<filter>, <update>, <options>)	Actualiza todos los documentos que coinciden con el filtro
	db.collection.replaceOne(<filter>, <update>, <options>)	Reemplaza completamente el primer documento que coincide con el filtro con un nuevo documento
DELETE	db.collection.deleteOne()	Elimina un solo documento que coincida con el filtro.
	db.collection.deleteMany()	Elimina todos los documentos que coincidan con el filtro.

Campo	Tipo de dato	Descripción
Idcliente	objectID	Identificador único del cliente.
Nombre	String	Nombre completo del cliente.
Email	String	Correo electrónico del cliente.
Dirección	String	Dirección física del cliente.
Teléfono	String	Número de teléfono del cliente.
Fecha_registro	Date	Fecha de registro del cliente

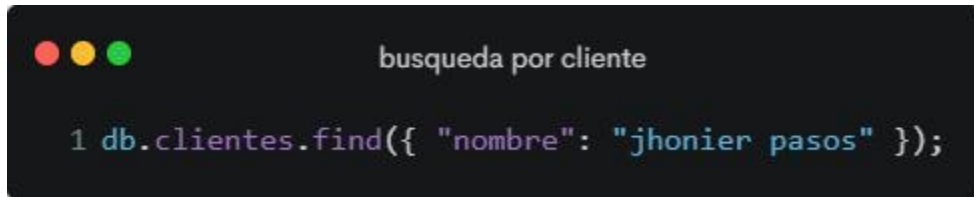
Campo	Tipo de dato	Descripción
Idproducto	objectID	Identificador único del producto.
Nombre	String	Nombre del producto
Categoría	String	Categoría del producto (e.g., "Electrónica").
Precio	Decimal	Precio unitario del producto.
Stock	Int	Cantidad de unidades en el inventario
FechaCreacion	Date	Fecha cuando se agregó al catálogo un producto

Campo	Tipo de dato	Descripción
Idtransaccion	objectID	Identificador único de la transacción.
PedidoID	objectID	Identificador del pedido con la transacción.
Monto	Decimal	Monto total de la transacción.
MetodoPago	String	Método de pago utilizado (e.g., "Tarjeta de crédito").
FechaTransaccion	Date	Fecha de la transacción.

Campo	Tipo de dato	Descripción
Idpedido	objectID	Identificador único del pedido.
ClienteID	objectID	Identificador de quien hace el pedido (cliente)
Productos	Array	Product.comprados, con cantidad y precio unitario.
Productos. productosID	objectID	Identificador del producto en la colección productos.
Productos. Cantidad	Int	Cantidad de productos en el pedido.
Productos. PrecioUnitario	Decimal	Precio unitario del producto.
Total	Decimal	Total del pedido calculado en base a productos.
TechaPedido	Date	Fecha en la que se realizó el pedido.
Estado	String	Estado actual del pedido ("Procesado", Etc....).

Consultas NoSQL

1. Buscar un cliente por nombre:



```
1 db.clientes.find({ "nombre": "jhonier pasos" });
```

Esta consulta devuelve el cliente cuyo nombre es "jhonier pasos". Es útil para buscar clientes por su nombre dentro de la colección llamada clientes.

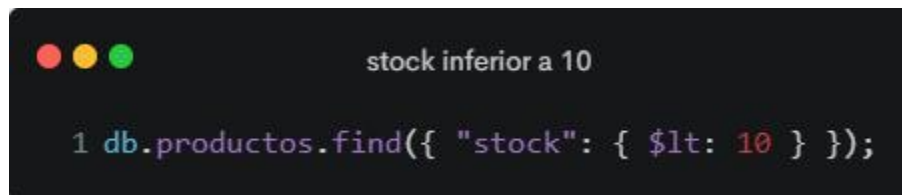
Cabe resaltar que con la función `find()` no solo podemos buscar por nombre si no que también por cualquier otro campo de nuestra colección.

si solo utilizáramos la función `db.clientes.find({})` sin ningún parámetro nos mostraría en pantalla todos los datos de esa colección, lo cual sería equivalente a una consulta de este tipo `SELECT * FROM clientes` en mysql

como salida de nuestra consulta de la persona jhonier pasos se debería evidenciar en pantalla todos los datos del cliente con ese nombre asignado

```
> db.clientes.find({"nombre":"jhonier pasos"})
< {
  _id: ObjectId('66df6e363005585b151f64f0'),
  nombre: 'jhonier pasos',
  email: 'jp@gmail.com',
  direccion: 'Calle 123, Ciudad',
  telefono: '555-1234',
  fecha_registro: 2024-08-25T00:00:00.000Z
}
```

2. Buscar stock cerca a terminar:



```
1 db.productos.find({ "stock": { $lt: 10 } });
```

En esta consulta lo que estamos buscando dentro de productos en el campo de stock pero estamos utilizando el operador de comparación `$lt` el cual compara dos valores y devuelve true si el primero es menor que el segundo indicando como resulta de la consulta a todos los productos con stock inferior al número que se le defina.

```
> db.productos.find({ "stock": { $lt: 10 } });
< {
  _id: ObjectId('66df6f985619b93141fc367f'),
  nombre: 'Mac book',
  categoria: 'Electrónica',
  precio: 23233,
  stock: 8,
  fecha_creacion: 2024-09-01T00:00:00.000Z
}
```

3. buscar pedidos por un cliente en particular:

```

● ● ● pedidos por cliente

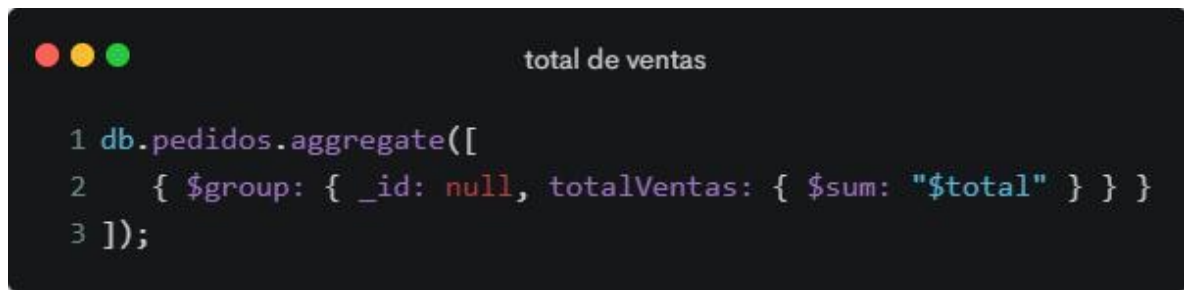
1 db.pedidos.find({ "cliente_id": ObjectId("66df6e363005585b151f64f0") });

```

Se busca dentro de la coleccion de transacciones todos los pedidos que esa persona ha realizado mediante el identificador del cliente.

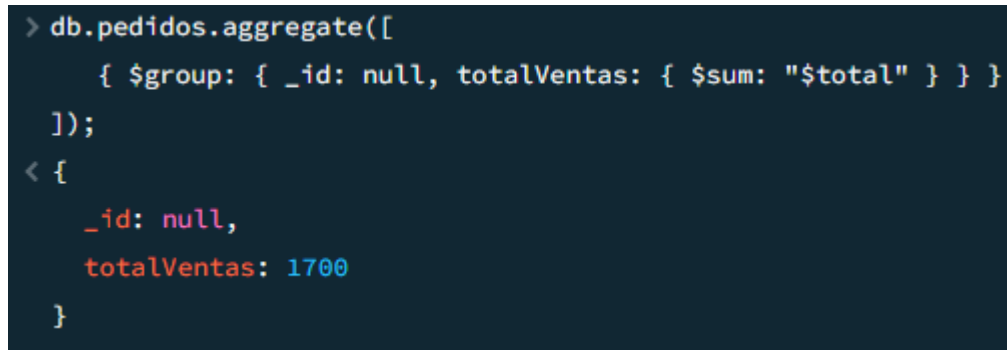
```
> db.pedidos.find({ "cliente_id": ObjectId("66df6e363005585b151f64f0") });
< {
  _id: ObjectId('66df6fad3005585b151f64f3'),
  cliente_id: ObjectId('66df6e363005585b151f64f0'),
  productos: [
    {
      producto_id: ObjectId('66df6f985619b93141fc367f'),
      cantidad: 2,
      precio_unitario: 850
    }
  ],
  total: 1700,
  fecha_pedido: 2024-09-01T00:00:00.000Z,
  estado: 'Procesado'
}
```

4. Total de ventas realizadas en la tienda:



```
1 db.pedidos.aggregate([
2   { $group: { _id: null, totalVentas: { $sum: "$total" } } }
3 ]);
```

Esta consulta realiza una agregación para calcular el total de todas las ventas sumando el campo total en la colección pedidos, dando resultado a una sumatoria de todas las ventas



```
> db.pedidos.aggregate([
  { $group: { _id: null, totalVentas: { $sum: "$total" } } }
]);
< {
  _id: null,
  totalVentas: 1700
}
```

Análisis y Diseño

La implementación de MongoDB en la base de datos de una tienda, con la gestión de datos no estructurados trabajando a través de consultas, logramos resultados acertados para la operación de la tienda. Con la búsqueda de clientes por nombre, mostramos cómo MongoDB maneja eficientemente los datos con una sintaxis simplificada en comparación con SQL, también tenemos la consulta de los pedidos de clientes específicos, donde nos damos cuenta la eficiencia de las relaciones mediante identificadores únicos, eliminando la necesidad de joins y mejorando la velocidad de las consultas; Además, el cálculo del total de ventas utilizando agregaciones notamos la flexibilidad de mongo para realizar cálculos, optimizando la generación de informes financieros. En conjunto, estos resultados no solo reflejan la eficiencia y escalabilidad de MongoDB, sino que también validan su adaptabilidad para gestionar datos en escenarios reales.

Conclusión

Usar MongoDB para nuestra tienda en línea ha sido muy útil para manejar datos flexibles y realizar consultas. Vimos cómo MongoDB facilita la organización y el análisis de la información sin complicaciones. Este tipo de base de datos no solo hace que trabajar con datos sea más fácil, sino que también ofrece herramientas para entender y visualizar la información. En pocas palabras, MongoDB ha demostrado ser una muy buena opción para adaptarse a las necesidades de la tienda en un futuro de forma escalable.

Recomendaciones

Procedimientos y Estándares de Estandarización Típica: definir y documentar un conjunto de estándares y convenciones en relación con la creación y modificación de colecciones, documentos y consultas. Esto implica en particular utilizar los mismos nombres de campos y colecciones y seguir buenas prácticas al formular una consulta. Definiendo y siguiendo Procedures and Typical Standardization Standards se facilita mantener el sistema en el futuro, integrar nuevos desarrolladores y mantener una base de datos ordenada y organizada.

REFERENCIAS

MongoDB, Inc. (2024). *MongoDB Manual: The official documentation for MongoDB 6.x*. Retrieved from <https://docs.mongodb.com>

Faura, R. (2013). *Tutorial MongoDB Operadores Expresión II*. Retrieved from <https://blog.rubenfa.me>

Jhonier Pasos (2024). *Repository for SQL and MongoDB* [GitHub Repository]. Retrieved from https://github.com/jhonierp/repository_sql.git