INFORME MONGODB

PRESENTADO POR

CRISTIAN CAMILO VALLEJOS BASTIDAS

JOHAN FERNANDO BURBANO CALPA

INSTITUTO TECNOLOGICO PUTUMAYO

TECNOLOGIA EN DESARROLLO DE SOFTWARE

MOCOA – PUTUMAYO

2024

INDICE

RESUMEN EJECUTIVO

En el presente proyecto se realiza una familiarización con las bases de datos NoSQL, a través de MongoDB Compass, una herramienta gráfica utilizada para administrar de forma gráfica y sencilla bases de datos MongoDB. A partir de la implementación de estas herramientas, se realiza un estudio de las operaciones básicas permitidas tales como la creación de base de datos y colecciones, así como la inserción de documentos para adquirir un conocimiento práctico de manejo del modelo de datos NoSQL. La diferencia con las bases de datos relacionales radica en el esquema de almacenamiento flexible, en el formato JSON con el que utiliza documentos que pueden contener información estructurada y no estructurada y se implementan sin la necesidad de previamente definir un esquema de la base de datos.

INTRODUCCION

Contexto y Motivación

El crecimiento exponencial de los datos en la actualidad ha llevado a la adopción de tecnologías más flexibles y escalables para su administración. Las bases de datos relacionales tradicionales no son suficientes para las necesidades de las aplicaciones modernas, que se desempeñan en entornos distribuidos y deben procesar grandes cantidades de datos no estructurados y en tiempo real. Las bases de datos NoSQL se desarrollaron como una solución a la base del problema, ya que permiten gestionar mejor los datos dinámicos y de alta velocidad. Uno de los ejemplos más destacados de la familia NoSQL es MongoDB, un sistema que utiliza datos en JSON documentos, lo que significa que su esquema es mucho más flexible que el de un sistema relacional y se puede ajustar para mejorar la escalabilidad y el rendimiento.

Alcance del Informe

Este informe se centra en conocimiento de las funciones esenciales de **MongoDB Compass** y su aplicación en el manejo de bases de datos NoSQL. Se tomarán aspectos clave como:

1. Creación de bases de datos y colecciones, fundamentales para organizar la información en MongoDB.
2. Inserción de documentos con distintos tipos de datos para mostrar la flexibilidad del esquema.
3. Consultas básicas para recuperar datos de las colecciones.
4. Uso de índices y su importancia para optimizar la búsqueda de datos dentro de la base de datos.
5. Gestión de bases de datos NoSQL y cómo MongoDB maneja operaciones en grandes volúmenes de datos.

Objetivos

El objetivo principal de este informe es proporcionar una introducción práctica a los sistemas de bases de datos NoSQL utilizando MongoDB Compass como herramienta educativa . Los propósitos particulares consisten en:

1. Familiarizarse con MongoDB Compass y como es su uso para gestionar bases de datos NoSQL.

2. Diseñar consultas básicas sobre los documentos almacenados y optimizar eficazmente la búsqueda al aprovechar índices.
3. Comprender las ventajas de NoSQL sobre los sistemas de bases de datos relacionales tradicionales, particularmente en términos de escalabilidad y flexibilidad.

METODOLOGIA

Herramientas Utilizadas

Para llevar a cabo el desarrollo y análisis de este proyecto, se utilizaron las siguientes herramientas:

- MongoDB: Base de datos NoSQL utilizada para almacenar y gestionar los datos de manera flexible y escalable.
- MongoDB Compass: Herramienta gráfica que facilita la interacción visual con las bases de datos MongoDB, permitiendo la creación de colecciones, inserción de documentos y la realización de consultas.
- MongoDB Database Tools: Conjunto de herramientas de línea de comandos utilizadas para la administración avanzada de bases de datos, como la importación y exportación de datos.
- GitHub: Plataforma utilizada para gestionar el control de versiones del proyecto, facilitando la colaboración y el seguimiento de los cambios en los archivos y configuraciones.

Procedimientos

Para el desarrollo del informe y la implementación del proyecto, se siguieron los siguientes métodos y pasos:

Configuración del entorno de trabajo:

1. Instalación de MongoDB como motor de base de datos.
2. Configuración e instalación de MongoDB Compass para interactuar visualmente con la base de datos.
3. Configuración de MongoDB Database Tools para tareas avanzadas como la exportación de datos y la creación de respaldos.
4. Creación de un repositorio en GitHub para gestionar las versiones del proyecto y facilitar la documentación de los avances.

Creación de la base de datos:

5. A través de MongoDB Compass, se creó una base de datos inicial con el nombre correspondiente al proyecto.

6. Se definieron y configuraron las primeras colecciones dentro de la base de datos, considerando la estructura de datos flexible que MongoDB ofrece.

Inserción y manejo de datos:

7. Se insertaron documentos en las colecciones, probando diferentes estructuras de datos para observar cómo MongoDB maneja la flexibilidad en la representación de la información.
8. Se realizaron consultas básicas para recuperar y analizar los datos, incluyendo búsquedas por filtros específicos, agregaciones y operaciones CRUD (Create, Update, Delete).

Uso de MongoDB Database Tools:

9. Se utilizó MongoDB Database Tools para realizar respaldos de la base de datos y exportar colecciones para fines de prueba y migración.
10. Las herramientas de línea de comandos también se emplearon para automatizar algunas tareas de administración de datos.

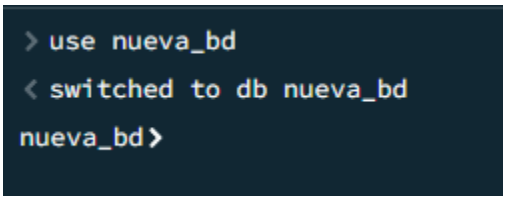
Control de versiones y documentación:

11. Durante todo el proceso, se utilizaron ramas en GitHub para versionar los diferentes avances del proyecto.
12. La documentación del proyecto fue actualizada de forma continua en el repositorio, asegurando un registro claro de los procedimientos seguidos y los resultados obtenidos.

DESARROLLO DEL INFORME

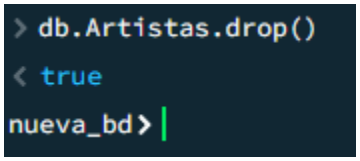
Consultas NoSQL:

CREACION DE BASE DE DATOS EN MONGODB: La creación de un base de datos en MongoDB es muy simple, solo nos dirigimos en nuestro localhost y en el parte superior derecho esta la opción de Open MongoBD Shell ingresamos y con solo poner use y el nombre de la nueva base de datos que quisiéramos crear.



CREACION DE COLECCIONES: Para la creación de colecciones nos situamos en Open MongoBD Shell y con escribimos db.createColletion("nombre_de_coleccion", opciones), ponemos el nombre y las opciones y listo.

ELIMINACION DE COLECCIÓN: Para esto solo bastaría con seleccionar el nombre de la colección y al final poner drop db.nombre_coleccion.drop().



BORRAR BASE DE DATOS: Se debe tener en cuenta el nombre de la base de datos que se quiera eliminar, una vez que tengamos presente el nombre de la base de dato que quisiéramos eliminar, ponemos use nombre_de_la_bd esto para seleccionarla, despues ya podemos proceder a eliminarla con el siguiente comando db.dropDatabase().

```
> db.dropDatabase()
< { ok: 1, dropped: 'nueva_bd' }
nueva_bd>
```

CONSULTA 1: En nuestra primera consulta obtendremos todos los nombres de los artistas en orden alfabético.

```
> use music
< switched to db music
> db.Artistas.find({}, { name: 1, _id: 0 }).sort({ name: 1 })
< {
    name: 'Anuel AA'
  }
  {
    name: 'Bad Bunny'
  }
  {
    name: 'J Balvin'
  }
  {
    name: 'Karol G'
  }
  {
    name: 'Ozuna'
  }
```

Como primero accedemos a nuestra colección **Artistas** mediante **db.Artistas,** después con un **find ()** haremos nuestra búsqueda, con un primer parámetro vacio el cual selecciona todos los documentos de la colección, en un segundo parámetro especificamos que campos se devolverán, en este caso **name: 1** indica que queremos incluir **name** en los resultado y con **_id: 0** indicamos que queremos excluir el campo **_id** que por defecto MongoDB lo incluye, después pasamos a un **.sort()** el cual organiza los documento de acuerdo al campo especificado por el parámetro, con **{ name : 1 }** estamos indicando que se ordenara de orden ascendente por el campo name. Si quisiéramos ordenarlos en orden descendente usaríamos **-1.**

CONSULTA 2: Como siguiente consulta obtendremos la canción con mayor cantidad de reproducciones.

```
> db.Canciones.find().sort({ plays: -1 }).limit(1)
< {
    _id: ObjectId('64e1012a1f4d880001c1a213'),
    title: 'Mi Gente',
    duration: '3:06',
    album_id: ObjectId('64e0012a1f4d880001c1a104'),
    plays: 1600000000
  }
```

Se accede a la colección **Canciones** con un **find()** sin parámetro se seleccionan todos los documentos, al igual que el anterior utilizaremos un **.sort()** donde en el primer parámetro **{ plays : -1}** con el cual indicamos que queremos ordenar los resultados en orden descendente según el campo **plays,** por ultimo utilizamos el método de **limit(1)** para seleccionar el primer documento del conjunto ordenado.

CONSULTA 3: En esta consulta obtendremos los albunes lanzados a partir del año 2020 y en adelante.

```
> use music
< switched to db music
> db.Albums.find({ release_year: { $gte: 2020 } })
< {
    _id: ObjectId('64e0012a1f4d880001c1a101'),
    artist_id: ObjectId('64e0012a1f4d880001c1a001'),
    title: 'YHLQMDLG',
    release_year: 2020,
    songs: [
      ObjectId('64e1012a1f4d880001c1a201'),
      ObjectId('64e1012a1f4d880001c1a202'),
      ObjectId('64e1012a1f4d880001c1a203'),
      ObjectId('64e1012a1f4d880001c1a204')
    ]
  }
  {
    _id: ObjectId('64e0012a1f4d880001c1a102'),
    artist_id: ObjectId('64e0012a1f4d880001c1a001'),
    title: 'El Último Tour del Mundo',
    release_year: 2020,
    songs: [
      ObjectId('64e1012a1f4d880001c1a205'),
      ObjectId('64e1012a1f4d880001c1a206'),
      ObjectId('64e1012a1f4d880001c1a207'),
      ObjectId('64e1012a1f4d880001c1a208')
    ]
```

Traeremos todos los parámetros con **.find()** donde especificaremos que en el campo **reléase_year** ser operado por **$gte** el cual es un operador de comparación y el valor que queremos comparar será **2020.**

CONSULTA 4: En nuestra última consulta buscaremos el valor total de las reproducciones de las canciones:

```
> db.Canciones.aggregate([
    {
      $group: {
        _id: null,
        totalPlays: { $sum: "$plays" }
      }
    }
  ])
< {
    _id: null,
    totalPlays: 9770000000
  }
```

Accedemos a nuestra colección de **Canciones** y utilizaremos el método **.aggregate()** el cual realiza operaciones de agregación permitiéndonos procesar datos de una manera más avanzada, el argumento para esta operación será la etapa **$group** la cual es una etapa de agregación que agrupa los documento en función de dichas especificaciones dadas, al poner el **_id: null** recordemos que esto indica que no estamos agrupando por ningún campo en específico, en el siguiente parámetro **{ $sum: "$plays" }** indicamos que sumamos los valores del campo **plays**

Diseño de Base de Datos

Elección de claves primarias.

El campo "_id" es la clave primaria de cada documento de las colecciones. Se genera automáticamente como un ObjectID ($oid) y garantiza que cada documento tenga un identificador único. Este diseño permite realizar búsquedas y actualizaciones eficientes, ya que cada documento tiene un identificador único en la base de datos.

Definición de los campos de la colección "Albums".

Campo	Definición
_id	Este campo almacena el identificador único del álbum, lo que permite realizar consultas rápidas y precisas. Al utilizar un ObjectID, MongoDB asegura que la generación de esta clave sea eficiente y única en toda la base de datos.
artist_id	Este campo almacena el identificador del artista asociado con el álbum. Se utiliza para crear una referencia entre la colección de álbumes y la colección de artistas. También es un ObjectID, lo que sugiere que se establece una relación entre colecciones, aunque MongoDB no tiene restricciones formales de claves foráneas como los sistemas relacionales. En un esquema relacional, este campo sería equivalente a una clave foránea.
title	El título del álbum, representado como una cadena de texto (String). Este campo es fundamental para identificar el álbum de manera legible por los usuarios.
release_year	Almacena el año de lanzamiento del álbum como un valor numérico (Number). Este campo permite hacer consultas sobre los álbumes lanzados en un período específico o realizar análisis históricos.
songs:	Este es un campo que contiene una lista de identificadores de las canciones asociadas al álbum. Los identificadores son ObjectID que permiten referenciar los documentos en la colección de canciones. Este diseño permite mantener una relación "uno a muchos", donde un álbum puede tener varias canciones asociadas.

Definición de los campos de la colección “Canciones”.

Campo	Definición
_id	El identificador único de la canción, que es un ObjectID. Este campo asegura que cada documento en la colección “Canciones” sea distinto y permite un acceso eficiente a los datos de cada canción.
title	El título de la canción, almacenado como un campo de texto (String). Este campo permite identificar y mostrar la canción en las interfaces de usuario y realizar búsquedas basadas en el nombre de la canción.
duration	La duración de la canción se almacena como una cadena de texto (String), representando el tiempo en formato "minutos

	" ("4:55"). Aunque podría almacenarse como un número de segundos, el formato de texto puede ser más intuitivo para mostrar al usuario, aunque debes considerar si es más práctico almacenarlo en segundos para cálculos o análisis.
album_id	Este campo almacena el ObjectID del álbum al que pertenece la canción. Esto crea una relación entre las colecciones songs y albums. Aunque MongoDB no tiene restricciones formales de claves foráneas, este campo funciona como una referencia a la colección de albums, permitiendo consultas sobre las canciones de un álbum específico.
plays	Almacena el número de reproducciones de la canción como un valor numérico (Number). Este campo es útil para realizar análisis de popularidad, calcular estadísticas o generar recomendaciones basadas en la cantidad de veces que una canción ha sido escuchada.

Definición de los campos de la colección "Artistas".

Campo	Definición
_id	El identificador único del artista. Este campo es un ObjectID y sirve para diferenciar a cada artista en la base de datos, garantizando unicidad en las operaciones CRUD.
name	El nombre del artista, representado como una cadena de texto (String). Este campo es fundamental para identificar al artista de manera legible y realizar búsquedas por nombre.
genre	El género musical del artista, almacenado también como una cadena de texto (String). Este campo permite clasificar a los artistas según su estilo musical, facilitando búsquedas por género o análisis de tendencias musicales.
albums	Este es un campo que almacena una lista de identificadores (ObjectID) de los álbumes creados por el artista. Cada identificador en la lista referencia un documento en la colección albums, creando una relación "uno a muchos" entre un artista y sus álbumes. MongoDB no impone restricciones de claves foráneas, pero esta estructura ayuda a mantener las relaciones de manera flexible entre colecciones.

ANÁLISIS Y DISCUSIÓN

Flexibilidad en la gestión de datos: Para la gestión de muchos tipos de datos sin necesidad de un marco predefinido como en las bases de datos relacionales. Facilita la adaptación a aplicaciones que necesitan manejar datos dinámicos o no estructurados, logrando el objetivo de mostrar la flexibilidad del sistema NoSQL en comparación con las bases de datos relacionales. El informe muestra la facilidad de uso para administrar grandes volúmenes de datos, ya que muestra cómo insertar documentos de diversas estructuras y realizar consultas efectivas.

Optimización de consultas mediante índices: Un aspecto clave del análisis es la utilización de índices para optimizar las búsquedas dentro de las colecciones. El uso de índices fue probado en el desarrollo del informe, lo que demuestra cómo MongoDB maneja eficientemente los datos almacenados, alineándose con el objetivo de optimizar las consultas en una base de datos NoSQL.

Consultas específicas y agregaciones: Se llevaron a cabo diversas consultas y agregaciones (como la consulta para obtener la canción con mayor cantidad de reproducciones o los álbumes lanzados a partir de 2020), lo que permitió evidenciar la capacidad de MongoDB para realizar análisis de los datos. Este resultado está en línea con el objetivo de comprender cómo se pueden realizar operaciones complejas y cómo MongoDB es adecuado para estas tareas.

Comparación con bases de datos relacionales: Se demuestra que MongoDB puede manejar grandes volúmenes de datos no estructurados y estructurados, lo que es crucial en entornos distribuidos y en tiempo real, cumpliendo así con el objetivo de resaltar las ventajas del enfoque NoSQL en comparación con las bases de datos relacionales.

Relación con los Objetivos

Familiarización con MongoDB Compass: La interacción con MongoDB Compass facilitó la creación de bases de datos y colecciones, así como la inserción y consulta de documentos. Esto cumple directamente con el objetivo principal de familiarizarse con esta herramienta y sus funciones clave.

Optimización de consultas: La implementación de índices y la ejecución de consultas agregadas demostraron cómo se pueden optimizar las búsquedas, lo cual es un claro reflejo de los objetivos relacionados con el uso eficiente de índices.

Escalabilidad y flexibilidad de NoSQL: Los resultados obtenidos a lo largo del informe muestran que MongoDB es una solución escalable y flexible, destacando cómo este sistema es ideal para manejar datos de diferentes formatos sin requerir un esquema predefinido.

CONCLUSIONES

1. MongoDB ofrece una flexibilidad en el manejo de datos no estructurados, permitiendo la inserción de documentos sin necesidad de definir previamente un esquema rígido. Esta característica lo convierte en una herramienta ideal para aplicaciones que requieren manejar grandes volúmenes de datos.
2. MongoDB Compass se destacó como una herramienta eficiente y amigable para la gestión de bases de datos NoSQL. Su interfaz gráfica simplificó la creación de bases de datos, colecciones y la ejecución de consultas, permitiendo un manejo intuitivo de las operaciones básicas en MongoDB.
3. La comparación entre MongoDB y las bases de datos relacionales tradicionales mostró que, si bien ambas tienen ventajas específicas, MongoDB sobresale en entornos donde se requieren escalabilidad y flexibilidad. Su capacidad para gestionar datos dinámicos y de alto volumen es un factor clave que lo diferencia de los sistemas relacionales.

RECOMENDACIONES

Mejorar la documentación: El control de versiones y la documentación han sido parte de este proceso, es importante continuar documentando cada cambio significativo en la estructura de la base de datos o en los procedimientos utilizados. Esto facilitará el trabajo colaborativo y la futura escalabilidad del sistema.

REFERENCIAS

¿Cómo utilizar MongoDB Compass? Brújula MongoDB. (s. f.). Recuperado 11 de septiembre de 2024, de https://serverspace.io/es/support/help/how-to-use-mongodb-compass/

MongoDB Atlas: Cloud Document Database | MongoDB. (s. f.). Recuperado 11 de septiembre de 2024, de https://www.mongodb.com/es/lp/cloud/atlas/try4?utm_source=google&utm_campaign=search_gs_pl_evergreen_atlas_core_prosp-brand_gic-null_amers-co_ps-all_desktop_es-la_lead&utm_term=mongo&utm_medium=cpc_paid_search&utm_ad=p&utm_ad_campaign_id=20745580680&adgroup=156970815322&cq_cmp=20745580680&gad_source=1&gclid=CjwKCAjw_4S3BhAAEiwA_64YhksVx8qhuwgsCofV7-_Gb8ziGb7R3sUc4McAMoGNH9Rc38Ukr2uEGBoCjP4QAvD_BwE

MongoDB create collection para crear tus propias colecciones. (2022, diciembre 21). IONOS Digital Guide. https://www.ionos.com/es-us/digitalguide/paginas-web/desarrollo-web/mongodb-create-collection/

MongoDB Documentation. (s. f.). Recuperado 11 de septiembre de 2024, de https://www.mongodb.com/docs/