UNIVERSIDAD AUTÓNOMA METROPOLITANA
Casa abierta al tiempo

EDUCACIÓN IT

MongoDB Fundamentos

Motores NoSQL

Muchos de ustedes probablemente se estarán preguntando por qué necesitamos otra base de datos cuando ya contamos con las Relacionales (MySQL, MS SQL Server, Oracle, DB2, etc.). Las bases de datos relacionales son extremadamente ricas en características. Pero estas características no vienen gratis; hay un precio a pagar y se realiza comprometiendo la escalabilidad y flexibilidad.

NoSQL es una nueva forma de diseñar soluciones de base de datos a escala de Internet. No es un producto o tecnología, sino un término que define un conjunto de tecnologías de base de datos que no se basan en los principios tradicionales de los motores Relacionales.

No cabe duda de que la forma en que las aplicaciones web tratan los datos ha cambiado de forma significativa durante la última década. Cada vez se recopilan más datos y cada vez son más los usuarios que acceden a estos datos al mismo tiempo. Esto significa que la escalabilidad y el rendimiento se han convertido en auténticos retos para las bases de datos relacionales basadas en esquemas.

En primer lugar, había tipos de bases de datos NoSQL (de origen cerrado), desarrolladas por grandes empresas para satisfacer sus necesidades específicas, como BigTable de Google, que se cree es el primer sistema NoSQL y DynamoDB de Amazon.
El éxito de estos sistemas patentados, inició el desarrollo de varios sistemas de bases de datos de código abierto siendo los más populares Hypertable, Cassandra, MongoDB, DynamoDB, HBase y Redis.

Características

- Alta escalabilidad
- Fácil gestión y administración
- Bajo costo
- Modelos de datos flexibles

MongoDB

Qué es MongoDB? simplemente es una base de datos **orientada a documentos.**

Está diseñado para trabajar con documentos sin necesidad de tener columnas predefinidas o tipos de datos (a diferencia de las bases de datos relacionales), lo que hace que el modelo de datos sea extremadamente flexible

Estos documentos se almacenan en un formato llamado JSON binario (también conocido como BSON). Los documentos pueden tener diferentes esquemas, lo que significa que el esquema puede cambiar a medida que la aplicación evoluciona. MongoDB está construido Para escalabilidad, rendimiento y alta disponibilidad.

Cómo es un documento JSON?

Como podemos ver, un documento JSON siempre comienza y termina con llaves. Los campos y sus valores están separados por comas, con el nombre de campo siendo siempre un string y el valor de cualquier tipo de data que va de string, number, date, array, otro documento JSON, etc.

No proporciona soporte para operaciones JOIN, como en SQL. De todos modos, permite al usuario almacenar todos los datos relevantes juntos en un solo documento, evitando el uso de JOINs.

MongoDB no proporciona soporte para transacciones de la misma manera que SQL.
Sin embargo, garantiza la atomicidad a nivel de documento. Además, utiliza un operador de aislamiento para operaciones de escritura que afectan a varios documentos, pero no proporciona atomicidad de "todo o nada" para operaciones de escritura de varios documentos.

¿Por qué usar MongoDB?

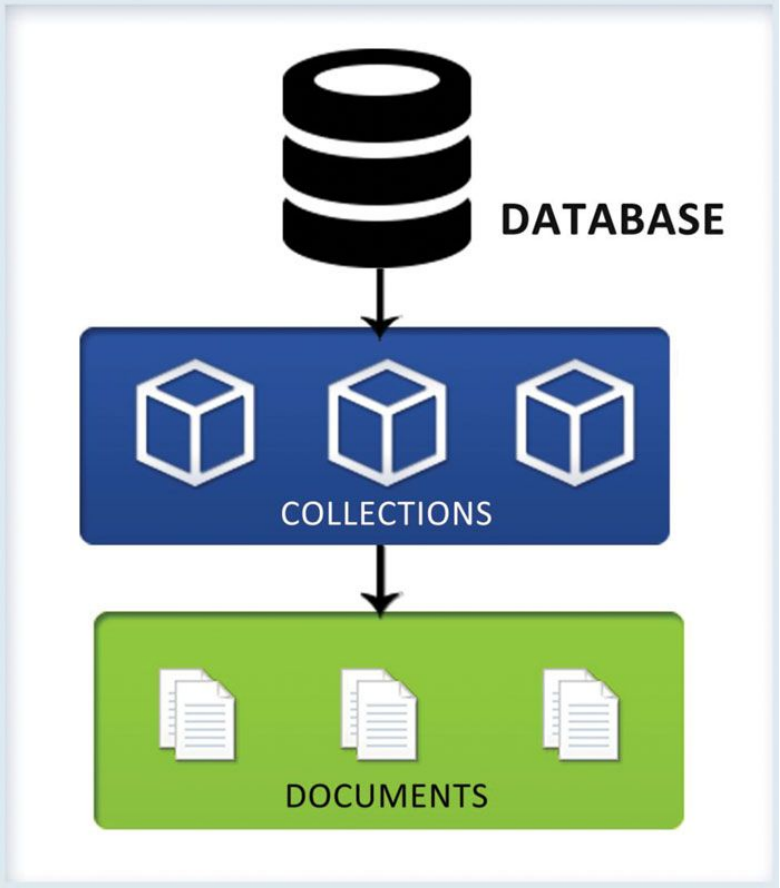
- Almacenamiento orientado a documentos: los datos se almacenan en forma de documentos de estilo JSON.
- Índice sobre cualquier atributo.
- Replicación y alta disponibilidad.
- Auto-sharding
- Consultas enriquecidas

¿Dónde usar MongoDB?

- Big Data
- Gestión de contenido y entrega
- Infraestructura móvil y social
- Gestión de datos de usuario
- Centro de datos

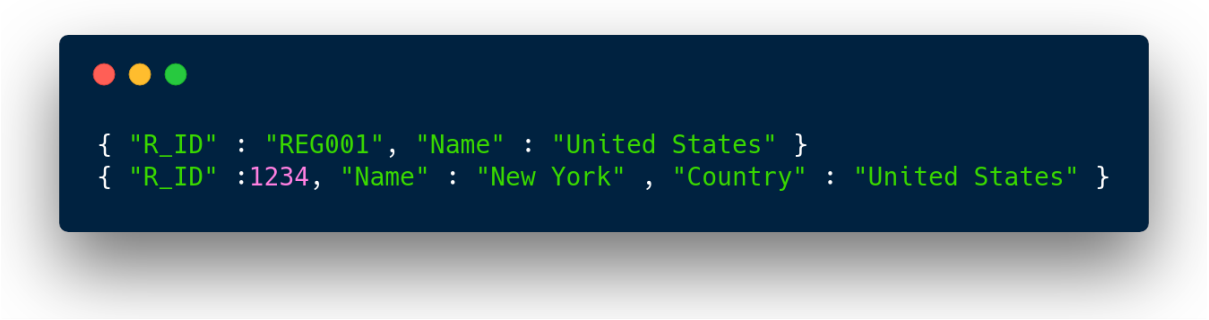
Modelado de Datos en MongoDB

Una implementación de MongoDB puede tener muchas bases de datos. Cada base de datos es un conjunto de colecciones. Las colecciones son similares al concepto de tablas en RDBMS; Sin embargo, no necesitan conocer con anterioridad el esquema (modelo de datos) y el mismo es dinámico. Cada colección puede tener múltiples documentos. Piense en un documento como una fila en RDBMS.



Esquema dinámico significa que los documentos dentro de la misma colección pueden tener conjuntos iguales o diferentes de campos o estructura, e incluso campos comunes pueden almacenar diferentes tipos de valores en los documentos. No hay rigidez en la forma en que los datos se almacenan en los documentos de una colección.

Veamos este ejemplo,



En este código, tenemos dos documentos en la colección Región. Aunque ambos documentos forman parte de la misma colección, tienen diferentes estructuras: la segunda colección tiene un campo de información adicional, que es país. De hecho, si observamos el campo "R_ID", almacena un valor de STRING en el primer documento mientras que es un número en el segundo documento. Por lo tanto, los documentos de una misma colección pueden tener esquemas completamente diferentes. Le corresponde a nuestra aplicación almacenar documentos en una colección particular juntos o para tener múltiples colecciones.

Identificador _id

Hemos visto que MongoDB almacena datos en documentos. Los documentos se componen de pares clave-valor. Aunque un documento puede compararse con una fila en RDBMS, a diferencia de una fila, los documentos tienen un esquema flexible. Una clave, que no es más que una etiqueta, puede compararse aproximadamente con el nombre de la columna en RDBMS. Por lo tanto, como una clave primaria RDBMS (utilizada para identificar de forma única cada fila), una colección debe tener una clave que identifique de forma única cada documento. A esto nos referimos como identificador _id en MongoDB. Si no ha sido especificado explícitamente ningún valor para una clave, se generará automáticamente un valor único y asignado a ella por MongoDB. Este valor clave es inmutable y puede ser de cualquier tipo de datos, excepto los arrays.

Algunas consideraciones al diseñar el esquema en MongoDB

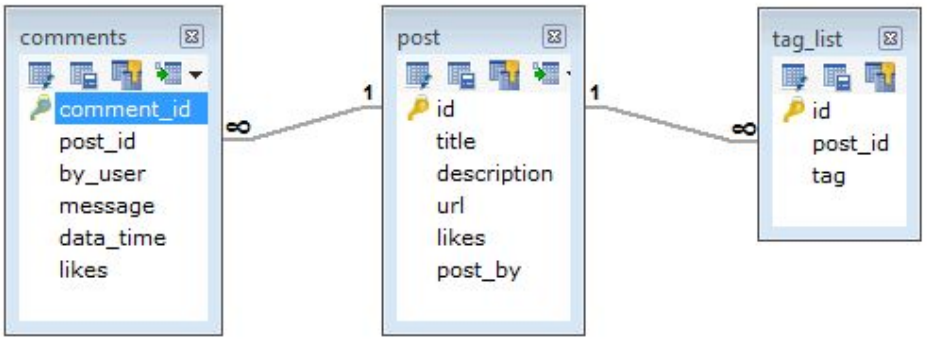
- Diseña tu esquema de acuerdo a los requerimientos del usuario.
- Combine los objetos en un solo documento si los usará juntos. De lo contrario, sepárelos (pero asegúrese de que no haya necesidad de uniones).
- Duplique los datos (pero limitado) porque el espacio en disco es barato en comparación con el tiempo de cálculo.
- Optimice su esquema para los casos de uso más frecuentes.

Ejemplo de caso de uso: Blog

Supongamos que un cliente necesita un diseño de base de datos para su blog / sitio web y ve las diferencias entre RDBMS y el diseño de esquema de MongoDB. El sitio web tiene los siguientes requisitos:

- Cada publicación tiene el título único, descripción y url.
- Cada publicación puede tener una o más etiquetas.
- Cada publicación tiene el nombre de su editor y el número total de "me gusta".
- Cada publicación tiene comentarios proporcionados por los usuarios junto con su nombre, mensaje, tiempo de datos y me gusta.
- En cada publicación, puede haber cero o más comentarios.

En el esquema RDBMS, el diseño para los requisitos anteriores tendrá un mínimo de tres tablas.



RDBMS vs. MongoDB

RDBMS	MongoDB
Base de datos	Base de datos
Tabla	Colección
Fila/Tupla	Documento
Columna	Campo
Join	Documentos Embebidos
Primary Key	Mongo campo _id automático

Instalar MongoDB

Para instalar el motor de MongoDB ir al sitio oficial https://www.mongodb.com/download-center/community y descargar la versión correspondiente a nuestro sistema operativo.

Luego de instalarlo nos encontraremos con dos archivos ejecutables, mongod (motor de base datos) y mongo (cliente de línea de comandos).
Primero tenemos que ejecutar “mongod” para levantar el motor, el administrador de los datos. Cuando logremos levantarlo podemos conectarnos con “mongo” el cliente de línea de comandos para realizar operaciones en la base de datos.

Clientes Gráficos

Al ya mencionado cliente de línea de comandos, podemos agregarle clientes gráficos para poder realizar operaciones de forma visual y más amigable al usuario. De todas formas te recomendamos familiarizarte con el cliente de línea de comandos ya que su uso es más extendido.
Mongo Compass: https://www.mongodb.com/download-center/compass
Robo 3T: https://robomongo.org/download

Fuentes

https://docs.mongodb.com/manual/
Apress - Practical MongoDB