Comparativa de gestores de bases de datos no relacionales

Lipa Calabilla, Abraham
Herrera Amezquita, Derian
Paredes Catacora, Randi
Mejia Rodriguez, Julio

30 de Noviembre de 2020

Resumen

Como su propio nombre indica, las bases de datos no relacionales son las que, a diferencia de las relacionales, no tienen un identificador que sirva de relación entre un conjunto de datos y otros. Como veremos, la información se organiza normalmente mediante documentos y es muy útil cuando no tenemos un esquema exacto de lo que se va a almacenar.

Las bases de datos NoSQL están diseñadas específicamente para modelos de datos específicos y tienen esquemas flexibles para crear aplicaciones modernas. Las bases de datos NoSQL son ampliamente reconocidas porque son fáciles de desarrollar, por su funcionalidad y el rendimiento a escala.

Abstract

As the name suggests, databases do not relational are those that, unlike relational, do not have an identifier that serves as a relationship between a data set and others. As we will see, the information is normally organized by documents and is very useful when we do not have an exact schematic of what to store.

NoSQL databases are specifically designed for specific data models and have flexible schemas to create modern applications. NoSQL databases are widely recognized because they are easy to develop, for its functionality and performance at scale.

1. Introducción

Las bases de datos relacionales dominaron el mundo de la gestión de datos desde la década de los 70, pero al surgir el Internet y su auge como plataforma del desarrollo de aplicaciones pusieron a prueba el dominio de las soluciones relacionales.

El volumen de datos al que debe hacer frente una aplicación web ha crecido exponencialmente durante los últimos años, así como el número de usuarios que utiliza las aplicaciones disponibles en Internet, y en consecuencia el volumen de transacciones y la demanda a la que se ven sometidas, ya que los usuarios esperan un tiempo de respuesta inmediato en sus interacciones online con el website. En el presente trabajo desarrollaremos mas a fondo los distintos temas acerca de las bases de datos no relacionales.

2. Bases de datos

Una base de datos no es nada más que una colección de información que existe a lo largo de un periodo de tiempo, generalmente por muchos años.

El término “database” se refiere a una colección de datos que son manejados por un *Sistema gestor de base de datos* (o simplemente, Gestor de base de datos o **DBMS**). [1]

De un DBMS se espera:

- Que los usuarios puedan crear nuevas bases

de datos, consultar y modificar los datos ubicados en éstas, usando un lenguaje de definición, de consultas y de manipulación de datos, respectivamente.

- Que soporten el almacenamiento de grandes cantidades de información, así como la persistencia, el acceso eficiente y la recuperación.
- Que controle el acceso a los datos otorgado a los usuarios y que evite las acciones parciales sobre los datos.

Las bases de datos tradicionales están siendo utilizadas desde los 70's y su trabajo consiste en almacenar una gran cantidad de datos y obteniendo datos de múltiples tablas mediante uniones (*'joins'*) complejas. [2]

Por 1990, las bases de datos relacionales fueron la norma. Las operaciones realizadas sobre los datos de una base de datos relacional se podían realizar mediante SQL (Lenguaje Estructurado de Consultas), el cual fue el más importante basado en el modelo relacional. [1]

3. Bases de datos no relacionales

NoSQL Se refiere a 'No solo SQL' o simplemente 'No SQL'. Describe tecnologías para el almacenamiento de datos que permiten la persistencia de datos con un alto rendimiento necesario para las aplicaciones de la escala de Internet de la actualidad. [3]

Una base de datos no relacional (o bases de datos NoSQL) responde a las necesidades de desarrollo de las aplicaciones modernas. [4]

- Las aplicaciones generan enormes volúmenes de datos nuevos y en constante evolución (estructurados, semiestructurados, no estructurados y polimórficos).
- El trabajo se realiza en equipos pequeños, que realizan 'sprints' de desarrollo ágiles con iteraciones rápidas.
- Las aplicaciones sirven como servicios, que no solo deben funcionar sin interrupción, sino que además tienen que ser accesibles desde muchos dispositivos distintos y deben poder escalarse.
- Las organizaciones ahora están recurriendo a arquitecturas de escalamiento horizontal que utilizan tecnologías de software abierto, servidores básicos y computación en la nube.

3.1. Tipos de bases de datos NoSQL

1. Clave/Valor
   Los datos usan claves que son identificadores y que son similares a una llave primaria en una base de datos relacional. [3] Cada elemento de la base de datos se almacena como un nombre de atributo (o clave), junto con su valor. [4] Algunos ejemplos de este tipo son **Riak** y **Berkeley DB**.

2. Orientadas a columnas
   Los datos se organizan por columnas en lugar de por filas. El efecto de este diseño arquitectónico es que hace que las consultas agregadas sobre grandes cantidades de datos sean mucho más rápidas de procesar. [3] Algunos ejemplos de este tipo son **Cassandra** y **HBase**.

3. Documentos
   Se empareja cada clave con una estructura de datos compleja que se denomina 'documento'. Los documentos pueden contener muchos pares de clave-valor distintos, o pares de clave-matriz, o incluso documentos anidados. [4] Un ejemplo de este tipo es **MongoDB**.

4. Grafos
   Utilizan la teoría de grafos para almacenar relaciones de datos en una serie de vértices con aristas, lo que hace que las consultas que funcionen con datos de esta manera sean mucho más rápidas. [3] Algunos ejemplos de este tipo son **Neo4J** y **Giraph**.

4. MongoDB

MongoDB es una base de datos NoSQL basado en documentos JSON y de código abierto escrito en C++, que proporciona alto rendimiento, alta disponibilidad y escalabilidad automática. [5]

Las bases de datos como MongoDB proporcionan escalabilidad automática, lo cual hace que esta base de datos sea idónea para grandes cantidades crecientes de información. [5]

El desarrollo de MongoDB comenzó en el año 2007 por la empresa 10gen3, publicando una versión final en el 2009 y actualmente se encuentra en la versión 4.4. [5]

4.1. RDBMS y MongoDB

A continuación se muestra una comparación de los términos utilizados en ambos tipos de bases de datos solo a modo orientativo, dado que no cuentan con la misma estructura. [6]

RDBMS	MongoDB
Database	Database
Table	Collection
Tuple/Row	Document
Column	Field
Table Join	Embedded Documents
Primary Key	Primary Key (Default key _id)

Table 1: Tabla comparativa de terminología RDMS - MongoDB

4.2. Ventajas sobre RDBMS

Algunas de las ventajas que tiene MongoDB sobre los gestores de bases de datos relacionales son: [6]

- Menos esquema
- La estructura de un objeto es clara
- No requiere de uniones complejas
- Consultas dinámicas sobre documentos
- Fácil escalabilidad
- No requiere de conversión o mapeo de objetos
- Rápido acceso a los datos

4.3. Arquitectura

En una base de datos de MongoDB, hay tres partes principales: [8]

mongod (MongoDB server) Es el proceso principal que maneja las solicitudes de datos, administra el formato de los datos y realiza operaciones de administración en segundo plano. Puede haber muchos demonios mongod ejecutándose como instancias primarias secundarias.

mongos Es el servicio de enrutamiento. Este proceso enruta información y datos en el clúster.

MongoDB shell Es la interfaz interactiva. Al usar JavaScript para ordenar, el desarrollador puede examinar los resultados de las consultas y verificar los casos de prueba.

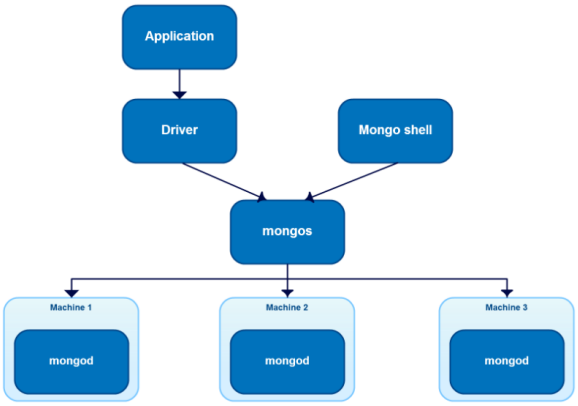

Figure 1: Arquitectura de una base de datos MongoDB

4.4. BSON

Generalmente, los usuarios trabajan con MongoDB mediante el formato JSON, sin embargo, MongoDB usa BSON (o 'Binary JSON'). BSON extiende del modelo JSON para proporcionar "tipos de datos" para una correcta codificación y decodificación en los diferentes lenguajes. [5]

4.4.1. JSON y BSON

JSON proporciona únicamente 6 tipos de datos: [5]

- String
  Este tipo de dato lo compone una cadena de texto normal.
- Number
  Este tipo de dato lo compone un número par o impar, se declara sin comillas dobles.
- Booleans
  Este tipo de dato es utilizado como una opción de tipo true o false.
- null
  Sencillamente este dato es para dejar en blanco los valores de un dato.
- Arrays
  Este tipo de datos es para crear un nuevo subconjunto de datos independientes dentro del JSON.
- Objetos / documentos
  Un archivo JSON puede tener objetos, es decir una nueva fila que dentro de ella hay un conjunto de datos específicos.

Los tipos de datos que maneja internamente BSON son los siguientes:

- Double
  Representa un valor flotante.
- String
  Las cadenas BSON son UTF-8. En general, los controladores para cada lenguaje de programación se convierten del formato de cadena del lenguaje a UTF-8 al serializar y deserializar BSON. Esto hace posible almacenar la mayoría de los caracteres internacionales en cadenas BSON con facilidad.
- Object
  Representa un documento incrustado.
- Array
  Los conjuntos o listas de valores se pueden representar como matrices.
- Binary data
  Los datos binarios son una cadena de bytes arbitrarios, no se pueden manipular desde el shell.
- Undefined (deprecated)
- Object id
  Los ObjectIds (identificador de documento MongoDB, equivalente a una clave principal) son: pequeños, probablemente únicos, rápidos de generar y ordenados. Estos valores constan de 12 bytes, donde los primeros cuatro bytes son una marca de tiempo que refleja la creación del ObjectId.
- Boolean
  Un verdadero o falso lógico. Se usa para evaluar si una condición es verdadera o falsa
- Date
  BSON Date es un entero de 64 bits que representa el número de milisegundos desde la época de Unix (1 de enero de 1970).
- Null
  Representa tanto un valor nulo como un campo inexistente.
- Regular Expression
- JavaScript Symbol JavaScript (con scope)
- 32-bit integer
  Los números sin puntos decimales se guardarán como enteros de 32 bits.

- Timestamp
  BSON tiene un tipo de marca de tiempo especial para el uso interno de MongoDB y no está asociado con el tipo de fecha normal.

- 64-bit integer
  Los números sin punto decimal se guardarán y se devolverán como enteros de 64 bits.

- Min key
  MinKey compara menos que todos los demás valores posibles de elementos BSON, respectivamente, y existen principalmente para uso interno.

- Max key
  MaxKey compara más que todos los demás valores posibles de elementos BSON, respectivamente, y existen principalmente para uso interno.s

5. Redis

Redis es una base de datos NoSQL, pero también, es una base de datos multi-modelo que permite la búsqueda, la mensajería, streaming, grafos y otras capacidades más allá de la de un simple almacén de datos. [3]

Redis mantiene los datos en la memoria para un acceso rápido y conserva los datos en el almacenamiento, así como la replicación de los contenidos en la memoria para escenarios de producción de alta disponibilidad. [3]

5.1. Estructura

5.1.1. Bases de datos

En Redis, al igual que en otras DBMS, una base de datos es un conjunto de datos. El caso de uso típico para una base de datos es agrupar los datos de una aplicación y mantenerlos separados de otras aplicaciones. [7]

5.1.2. Clave / Valor

Cada una de las estructuras de datos en Redis tienen al menos una clave y un valor. [7]

Claves Son cómo identificamos piezas de datos.

Valor Representan los datos actualmente asociados con la clave.

5.2. Almacenamiento de estructuras de datos

Las estructuras de datos soportadas que se incluyen: [3]

- Strings
  En Redis un string es una secuencia de bytes, lo que nos permite almacenar cualquier dato, como cadenas de texto, números, imágenes, vídeos, o un objeto serializado.

- Lists
  Las listas permiten almacenar secuencias de strings en el orden en el que fueron insertadas. En Redis las listas están optimizadas para que añadir un elemento al inicio o al final de la lista tarde lo mismo independientemente de si la lista tiene 10 elementos, o 100 millones de elementos.

- Sets
  Los sets son colecciones de strings, sin ningún orden particular, en los que se garantiza que los elementos del mismo son únicos y no pueden estar duplicados. Podemos añadir el mismo elemento 10 veces, pero Redis garantiza que sólo existirá una vez, lo que nos permite poder añadir elementos sin tener que preocuparnos previamente de si ya existen o no.

- Sorted sets
  Permiten la ordenación de los elementos en base a un valor (score) numérico asociado a cada uno de ellos. Es importante aclarar que la unicidad de los elementos se hace sólo teniendo en cuenta los datos almacenados, y no

el valor del score. Si se añade el mismo elemento varias veces con distintos valores del score, el elemento sólo existirá un vez, y el score será el de la última actualización.

- Hashes
  Permite asociar a una clave una colección de pares clave-valor, habitualmente usado para representar objetos.
- Bit arrays
- Streams
- HyperLogLogs
  Es una estructura de datos probabilística que se utiliza para contar valores únicos, o como se le conoce en matemáticas: calcular la cardinalidad de un conjunto.

5.3. Arquitectura

La arquitectura de Redis está basada en el enfoque BASE (Disponible básicamente, Soft-state y eventualmente consistente) mientras se elimina el enfoque ACID (atomicidad, consistencia, durabilidad y aislamiento) en RDBMS. Redis se basa en la arquitectura cliente / servidor y consta de los siguientes componentes: [8]

- Servidor Redis
- Servidores réplica de Redis (opcionales)
- Cliente Redis

6. Rendimiento

Para la comparación de rendimiento entre MongoDB y Redis, se ha realizado un proceso de benchmarking en una sola máquina (con las mismas características). La herramienta utilizada es YCSB (Yahoo Cloud Serving Benchmark).

Versiones de las bases de datos:

MongoDB 2.7.4

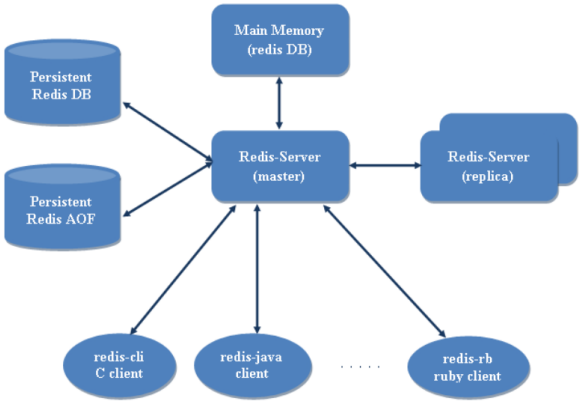

Figure 2: Arquitectura de una base de datos Redis

Redis 2.8.15

Se muestran los datos correspondientes a la relación entre el número de hilos de la máquina (los cuales son 1, 2, 4 y 8) y las operaciones por segundo, dada cierta cantidad de registros (1000, 5000 y 10000) y 1000 operaciones realizadas.

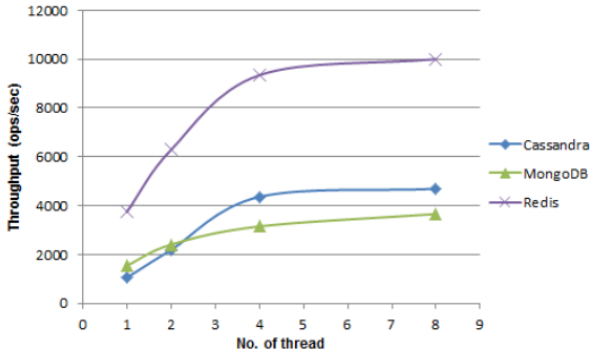

Figure 3: Gráfico comparativo con 1000 registros y 1000 operaciones

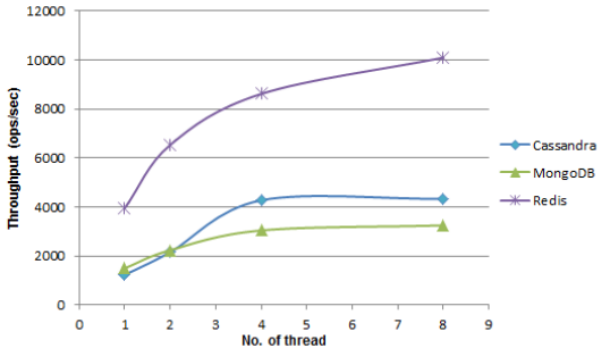

Figure 4: Gráfico comparativo con 5000 registros y 1000 operaciones

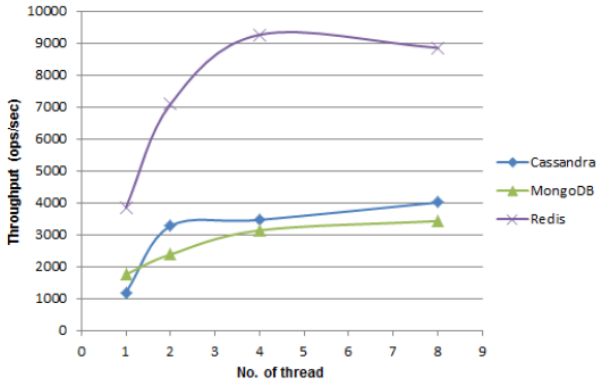

Figure 5: Gráfico comparativo con 10000 registros y 1000 operaciones

7. Conclusiones

A partir de la definición, requisitos, ventajas y características de la base de datos no relacionales, podemos conocer algunos aspectos dignos de atención, entre ellos podemos decir que la base de datos: una base de datos es un conjunto de datos o información que se utiliza para brindar servicios para muchas aplicaciones al mismo tiempo. En cuanto a los requisitos, se puede decir que ha realizado las mismas tareas de análisis que el software, y tiene las características de asociar información en organizaciones y asociaciones La ventaja de la base de datos es que la plataforma se puede utilizar para desarrollar aplicaciones. organización. Otro aspecto importante es el diseño y creación de la base de datos, existen muchas formas de organizar la información y expresar la relación entre los datos, los tres modelos lógicos principales en la base de datos son jerárquico, de red y relacional. Tiene ciertas ventajas comerciales y de procesamiento.

8. Recomendaciones

La elección de la tecnología de almacenamiento adecuada implica muchas consideraciones. Aunque el rendimiento suele ser el factor más importante, se deben considerar factores como la funcionalidad, la facilidad de operación, la facilidad de uso, la disponibilidad de profesionales conocedores, la seguridad y otros factores (como la existencia de herramientas y las comunidades que respaldan el producto). Gracias a los beneficios como el aumento de la productividad del equipo de desarrollo, la capacidad de ingresar al mercado antes y reducir el costo total de propiedad, vemos cómo la tecnología NoSQL se convierte cada vez más en parte de las soluciones de proyectos comerciales. Es importante tener en cuenta que, como dice el nombre ”NoSQL”, no solo SQL (”no solo SQL”), las tecnologías NoSQL no son necesariamente la única parte del almacenamiento de datos de la solución, pero a menudo acompañan a las bases de datos SQL, incluso si siguen siendo seguras en el futuro Úselo, incluso en combinación con otras bases de datos NoSQL. Con el reciente crecimiento explosivo de las arquitecturas basadas en microservicios, veremos cada vez más cómo cada servicio encapsula su propia solución de gestión de datos y, en la mayoría de los casos, utiliza ciertas tecnologías NoSQL disponibles.

9. Bibliografía

References

[1] GARCIAMOLINA, H., ULLMAN, J., & WIDOM, J. , (2008),Database Systems: The Complete Book (2.a ed.). Pearson.

[2] PUNIA, Y., & AGGARWAL, R. , (2014),Implementing Information System Using MongoDB and Redis. International Journal of Advanced Trends in Computer Science and Engineering,3(2), 16-20.

[3] JOHN WILEY & SONS, INC. , (2019),Redis For Dummies(Limited Edition) [Libro electrónico]. John Wiley & Sons, Inc. https://redislabs.com/redis-for-dummies/

[4] MONGODB. (S. F.). , Explicación sobre las bases de datos NoSQL. Recuperado 1 de diciembre de 2020, de:https://www.mongodb.com/es/nosql-explained

[5] MADRIGAL MARINAS, J. M. M. , (2020), MongoDB en Castellano (2020.a-07-13 ed. ed.) [Libro electrónico]. Leanpub. https://leanpub.com/mongodbcastellano

[6] TUTORIALS POINT (I) PVT. LTD. , (2018),MongoDB Tutorial. Tutorials Point (I) Pvt. Ltd. https://www.tutorialspoint.com/mongodb/mongodb_tutorial.pdf

[7] KARL SEGUIN, K. S. , (2020),The Little Redis Book (1.a ed.) [Libro electrónico]. Github. https://github.com/karlseguin/the-little-redis-book

[8] KUMARASINGHE, C. U., LIYANAGE, K. L. D. U., MADUSHANKA, W. A. T., & MENDIS, R. A. C. L. , (2016),Performance Comparison of NoSQL Databases in Pseudo Distributed Mode: Cassandra, MongoDB & Redis.

[9] GOALKICKER.COM. , (2017),MongoDB Notes for Professionals (3.6.1 ed.) [Libro electrónico]. GoalKicker.com. https://goalkicker.com/MongoDBBook/MongoDBNotesForProfessionals.pdf