Comparativa de gestores de bases de datos no relacionales

Mejía Rodriguez, Julio Oliver
(2010037899)

Paredes Catacora, Randi Angel
(2013047246)

Herrera Amezquita, Derian Francisco
(2017059489)

Lipa Calabilla, Abraham
(2019064039)

30 de Noviembre de 2020

Resumen

En el mundo de las bases de datos, es importante conocer las diferentes alternativas que están disponibles para tomar la mejor decisión de acuerdo al problema que queremos dar solución. Uno de los puntos de discusión más importantes es el del rendimiento, facilidad de uso y la durabilidad del gestor de base de datos.

En el presente trabajo se van a ver las características más importantes de los gestores de bases de datos no relacionales MongoDB y Redis, siendo estos los más populares. Veremos una comparación de sus características en común y los puntos en los que cada uno destaca más. Se van a comparar en rendimiento, facilidad de uso y la durabilidad del gestor de base de datos. Finalmente vamos a determinar los casos de uso de estos dos gestores de bases de datos.

Abstract

In the world of databases, it is important to know the different alternatives that are available to make the best decision according to the problem we want to solve. One of the most important talking points is the performance, ease of use, and durability of the database manager.

In this work, the most important characteristics of the non-relational database managers MongoDB and Redis will be seen, these being the most popular. We will see a comparison of their common characteristics and the points in which each one stands out the most. They will be compared in performance, ease of use and the durability of the database manager. Finally we are going to determine the use cases of these two database managers.

1. Introducción

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.

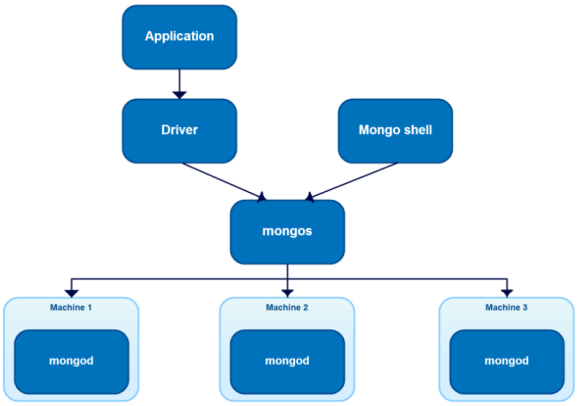

Figure 1: Arquitectura de una base de datos MongoDB

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
- Number
- Booleans
- null
- Arrays
- Objetos / documentos

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

- Double
- String
- Object
- Array
- Binary data
- Undefined (deprecated)
- Object id
- Boolean
- Date
- Null
- Regular Expression
- JavaScript Symbol JavaScript (con scope)
- 32-bit integer
- Timestamp
- 64-bit integer
- Min key
- Max key

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
- Lists
- Sets
- Sorted sets
- Hashes
- Bit arrays
- Streams
- HyperLogLogs

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. Comparación

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

Redis 2.8.15

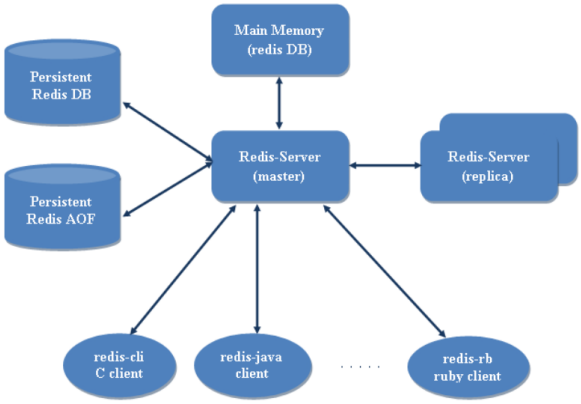

Figure 2: Arquitectura de una base de datos Redis

6.1. Rendimiento

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.

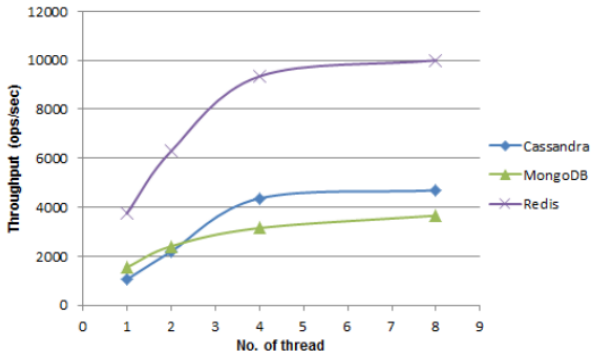

Figure 3: Gráfico comparativo con 1000 registros y 1000 operaciones

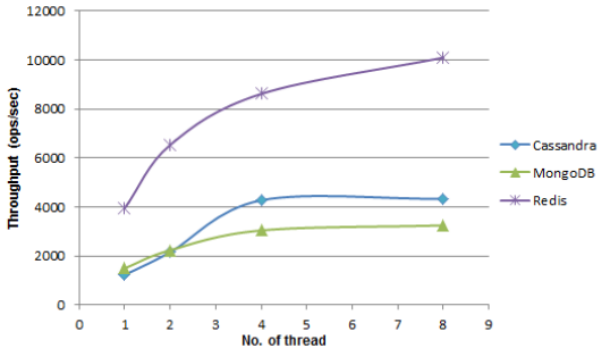

Figure 4: Gráfico comparativo con 5000 registros y 1000 operaciones

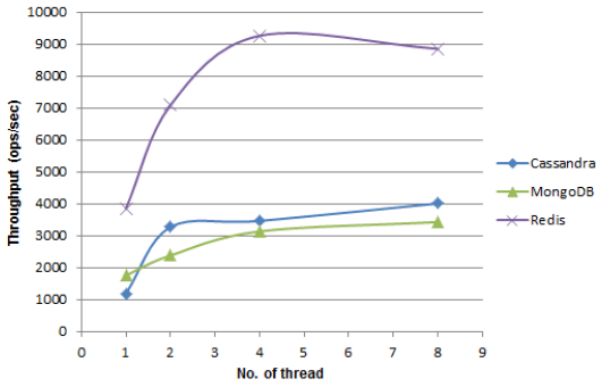

Figure 5: Gráfico comparativo con 10000 registros y 1000 operaciones

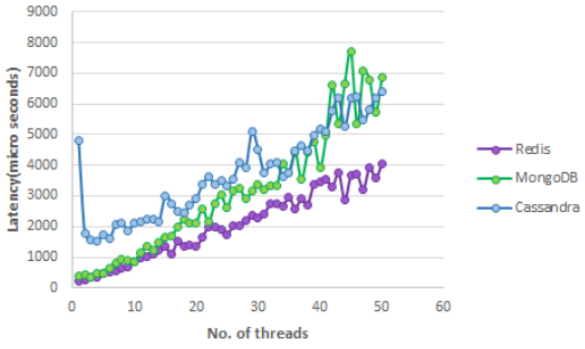

Figure 6: Latencia de lectura

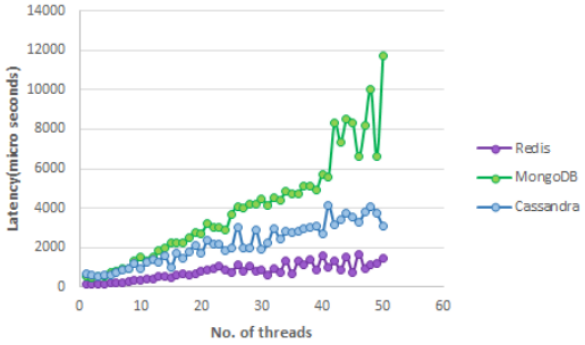

Figure 7: Latencia de actualización

6.2. Latencia y rendimiento

Se muestran los datos correspondientes a la relación entre el número de hilos de la máquina (los cuales van desde 1 a 50) y la latencia en microsegundos, dados 100000 registros.

Se muestran los datos correspondientes a la relación entre el número de hilos de la máquina (los cuales van desde 1 a 50) y las operaciones por segundo, dados 100000 registros.

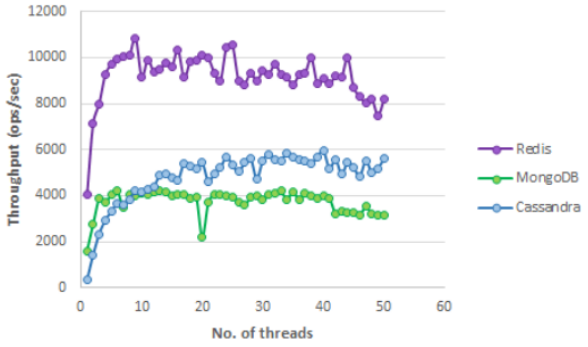

Figure 8: Rendimiento con 100000 registros

7. Conclusiones

Como se ha visto en la comparación de rendimiento, Redis tiene una gran ventaja en la acceso a los datos debido a que ofrece replicación de datos en memoria. También a las estructuras de datos que son muy simples. En cambio, MongoDB de cierta forma puede llegar a tener estructuras mucho más complejas, sin considerar la cantidad de estructuras de datos incluidos en BSON.

En el apartado de facilidad de uso, hemos visto que MongoDB usa una sintaxis similar a Javascript en el lenguaje de consultas, lo cual puede resultar familiar para los desarrolladores web Frontend o Backend (que hacen uso de Node). En cambio, Redis usa una sintaxis similar a bash con la ejecución de comandos que pueden argumentos separados por espacios.

8. Recomendaciones

9. Bibliografía