Comparación entre las bases de datos de Documentos y de Columnas

Damián Sancristobal
C.I.: 4.955.160-2
Facultad de Ingeniería, Universidad de la República
Montevideo, Uruguay
damiansancristobal@gmail.com

Santaigo Alles Conde
C.I.: 4.853.251-8
Facultad de Ingeniería, Universidad de la República
Montevideo, Uruguay
santiagoallesconde@gmail.com

***Resumen*—En la actualidad existe un amplio uso de las bases de datos no relacionales (BDNR). Debido a esto se considera relevante estudiar el estado del arte en este tema. En este artículo se realizara una comparación entre dos de las BDNR mas populares de la actualidad la base de datos documental MongoDB y la base de datos de columnas Cassandra. Esto se realizara siguiendo los pasos del artículo Which NoSQL Database? A Performance Overview [Abramova et al.(2014)Abramova, Bernardino, and Furtado]. En este artículo se realizara el mismo experimento de carga que en el artículo mencionado pero con versiones mas modernas de las bases de datos(BD). Con el fin de detectar cambios significativos en el rendimiento de las BD.**

I. Introducción

En la actualidad dada la gran popularidad con la que gozan las BDNR, se considera de valor comprobar el rendimiento de algunas de las mas populares bajo situaciones de estrés.

En este artículo se busca comparar las bases de datos documentales y las bases de datos de columnas.

Como se menciona en el resumen este artículo esta basado en Which NoSQL Database? A Performance Overview [Abramova et al.(2014)Abramova, Bernardino, and Furtado] de donde se toma el experimento de carga para replicarlo pero con versiones modernas de las bases de datos. Lo que se busca con esto es contrastar los resultados y comprobar la existencia de diferencias significativas entre los resultados originales y los obtenidos en la realización de este artículo. Documentando así el estado del arte de estas BDNR.

II. Trabajos relacionados

El principal trabajo relacionado con este artículo es Which NoSQL Database? A Performance Overview [Abramova et al.(2014)Abramova, Bernardino, and Furtado] en el cual se comparan diferentes tipos de BDNR. De este trabajo se obtuvo el experimento a replicar y también es con el cual se compararon los resultados obtenidos.

Además se utilizo como referencia al artículo: Type of nosql databases and its comparison with relational databases [Ameya et al.(2013)Ameya, Anil, and Dikshay]. El cual se utilizo para contextualizar y dar marco teórico a los experimentos de carga. Este artículo es un análisis de las ventajas y desventajas de las bases de datos NoSQL junto con una descripción de los principales tipos existentes.

III. Bases de datos no relacionales

Continuamente en la actualidad, muchas bases de datos NoSQL y productos relacionados se desarrollan y actualizan.

Hoy en día existen cinco grandes tipos de Base de Datos NoSQL, A continuación se describen brevemente cada tipo, basados en el artículo [Ameya et al.(2013)Ameya, Anil, and Dikshay]:

III-A. Base de Datos Clave Valor

En las Bases de Datos Clave Valor, los datos se almacenan en pares clave / valor. Está diseñado de tal manera que maneja muchos datos y cargas pesadas. Son altamente divisibles y permiten el escalado horizontal a escalas que otros tipos de bases de datos no pueden alcanzar.

En la actualidad, existen distintos sistemas de gestión de bases de datos basados en el almacenamiento con claves y valores, como por ejemplo Amazon DynamoDB y Redis, siendo esta última la más usada en la actualidad.

III-B. Base de Datos orientadas a Columnas

Las Base de Datos orientadas a Columnas almacenan tablas de datos por columnas en lugar de por fila. Normalmente debido a su rápida recuperación de columnas de datos, son usadas en aplicaciones analíticas.También reducen notablemente los requisitos globales de E/S del disco, y disminuye el volumen de datos que hay que cargar desde él.

Es en este tipo de base de datos donde se encuentra una de las que se tiene como objetivo analizar en el presente informe, Cassandra. En estos últimos años su popularidad se ha mantenido en los primeros lugares por amplia diferencia, pero el crecimiento de algunas otras han abierto el abanico de posibilidades a la hora de usar este tipo de BDs, como por ejemplo HBase o Microsoft Azure Cosmos DB.

III-C. Base de Datos orientadas a Documentos

Consideradas una de las más importantes dentro de los modelos no relacionales. Estas se han convertido en un elemento indispensable para gestionar información. Han sido diseñada para almacenar y consultar datos como documentos de tipo JSON.

Facilitan la consulta de datos en una base de datos mediante el mismo formato de modelo de documentos que emplean en el código de aplicación.

La naturaleza flexible, semi estructurada y jerárquica de los documentos y las bases de datos de documentos permite que evolucionen según las necesidades de las aplicaciones. Un ejemplo de esta tecnología es MongoDB, la otra base de datos que se analizará en este informe. Ha incrementado notoriamente su popularidad en los últimos tiempos, tanto que la tecnología que le sigue los pasos, Amazon DynamoDB, es siete veces menos popular.

III-D. Base de Datos orientadas a Grafos

Son bases de datos que almacenan información en forma de grafo. El gráfico consta de nodos y vértices, donde los nodos actúan como los objetos y los vértices representan relaciones.

En una base de datos de grafos, el énfasis principal está en la relación entre datos. Las consultas se expresan como recorridas del grafo, lo que hace que las bases de datos de gráficos sean mucho más rápidas que las bases de datos relacionales, y brindan una gran escalabilidad.

Las bases de datos de gráficos se pueden utilizar para una variedad de aplicaciones, como redes sociales, software de recomendación y publicidad, bio-informática, gestión de contenidos, seguridad y acceso, control, gestión de redes y nube, etc.

III-E. Base de Datos orientadas a Objetos

Las bases de datos orientadas a Objetos representan la información almacenada en Objetos (en el sentido de la programación orientada a Objetos).

Este tipo de bases de datos proveen todas las propiedades de la programación orientada a objetos, como la encapsulación de la información, el polimorfismo y herencia.

El acceso a la información más rápida que en bases SQL, ya que la referencia a la información se realiza directamente en base a punteros. Sin embargo, tienen la desventaja de estar atadas a un lenguaje de programación específico.

IV. Configuración del Experimento

En esta sección se describirá tanto el ambiente utilizado para el experimento como las cargas de trabajo a utilizar.

IV-A. Ambiente

Para la replicación del experimento de carga se utilizaron las siguientes herramientas:

- VM Ubuntu Server[1] 20.04 64-bits
- Virtualizador Oracle VM VirtualBox[2] 6.1
- Yahoo! Cloud Serving Benchmark 0.17.4
- Memoria de 4GB RAM y 10GB disco duro dinámico

Yahoo! Cloud Serving Benchmark[3] (YCSB), es una aplicación que permite la generación y carga de datos de prueba, así como la definición de pruebas de rendimiento basadas en operaciones de lectura y escritura de datos.

Durante el experimento se utilizaron las siguientes bases de datos:

- MongoDB 4.4.6 [4]
- Cassandra 4.0 [5]

IV-B. Cargas de trabajo

Los distintos escenarios de pruebas se definen a través de cargas de trabajo, que corresponden a una configuración que incluye: cantidad de operaciones, porcentajes de lecturas y escrituras y cantidad de datos utilizados.

El Cuadro I indica las distintas cargas de trabajo utilizadas en el experimento y la proporción de lecturas y actualizaciones.

Carga de Trabajo	Lectura	Actualizaciones
A	50 %	50 %
C	100 %	0 %
H	0 %	100 %

Cuadro I: Distribución de tipo de operaciones en las cargas de trabajo

Todas las cargas de trabajo consistieron de 1000 operaciones sobre un conjunto de datos de 600.000 registros con 10 campos cada registro y el tamaño de los campos es de 100 *bites*.

V. Resultados de los experimentos

Se ejecutaron 3 veces cada carga de trabajo y luego se realizo un promedio de los resultados obtenidos. Hay que mencionar que en la carga de trabajo A se ejecuto 6 veces debido al impacto que se observo del uso de cache.

V-A. Carga de trabajo A

Esta carga de trabajo se ejecuto el doble de veces que las demás, esto fue debido a que se vieron mejoras en el tiempos de ejecución de MongoDB gracias a las técnicas de uso de cache. En la Figura 1 logramos ver dicha evolución.

A su vez también se pueden apreciar los tiempos de ejecución de Cassandra los cuales no evolucionan con el uso de cache pero son mejores que los de su contra parte en todas las ejecuciones realizadas.

Cabe mencionar que en la cuarta ejecución los tiempos fueron muy similares.

En la Figura2 se pueden apreciar los promedios de las seis ejecuciones para cada base de datos, donde se confirma que Cassandra tuvo un mejor desempeño en esta carga de trabajo.

[1]Ubuntu Server https://ubuntu.com/download/server
[2]VM VirtualBox https://www.virtualbox.org
[3]YCSB wiki https://github.com/brianfrankcooper/YCSB/wiki
[4]MongoDB https://www.mongodb.com
[5]Cassandra https://cassandra.apache.org

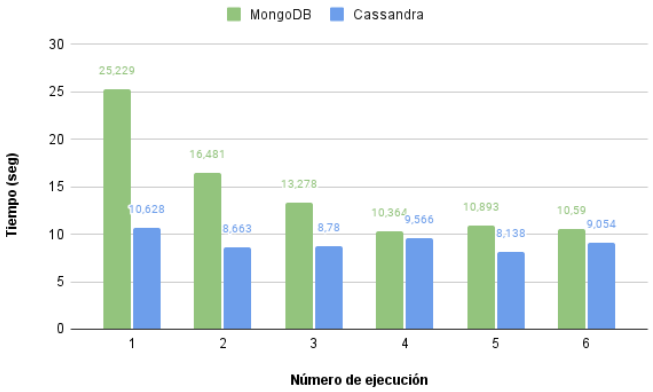

Figura 1: Ejecuciones de carga de trabajo A.

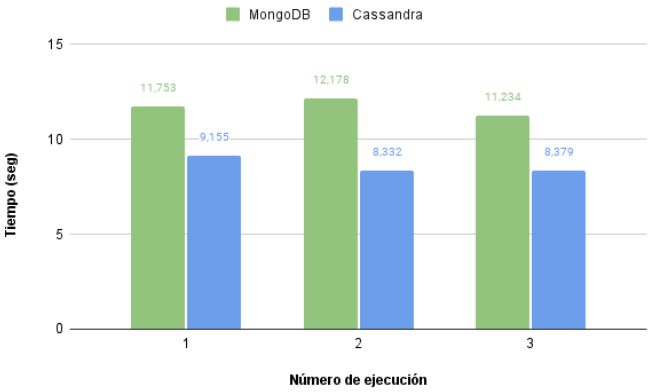

Figura 3: Ejecuciones de carga de trabajo C.

V-B. Carga de trabajo C

En la Figura 3 se ven los tiempos de ejecución de cada BD para la carga de trabajo C, donde no se vio un impacto significativo de las técnicas de uso de cache. Se empieza a dilucidar un patrón en cuanto al rendimiento de las bases de datos, donde Cassandra resulta ser unos segundos mas rápida que MongoDB.

En la Figura 4 no hay grandes observaciones que realizar, esto se debe a que las ejecuciones sobre esta carga fueron muy similares. Pero nos sera útil mas adelante para comparar con los resultados obtenidos por el artículo [Abramova et al.(2014)Abramova, Bernardino, and Furtado].

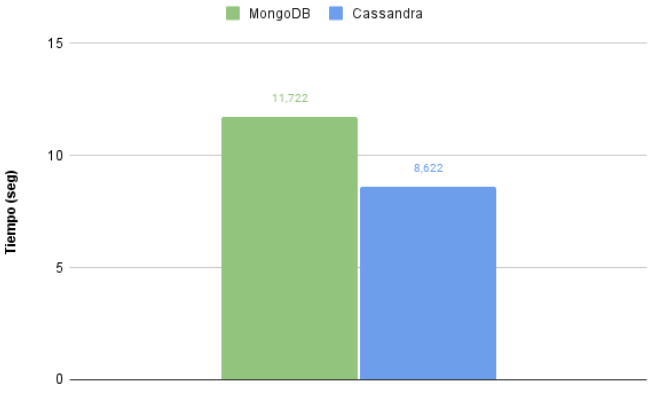

Figura 4: Promedios carga de trabajo C.

V-C. Carga de trabajo H

En la Figura 5 se confirma el patrón mencionado anteriormente donde Cassandra es algunos segundos mas rápida que MongoDB. A su vez se puede apreciar una pequeña mejora en el tiempo de ejecución de MongoDB entre la primera y la segunda ejecución de la carga de trabajo. Mejora que es atribuida al uso de cache.

En la Figura 6 muy similar a lo que sucedió con la Figura 4 no se aprecian grandes sorpresas en cuanto a los valores obtenidos ya que al igual que en las cargas anteriores Cassandra logro mejores tiempos que MongoDB en todas las ejecuciones.

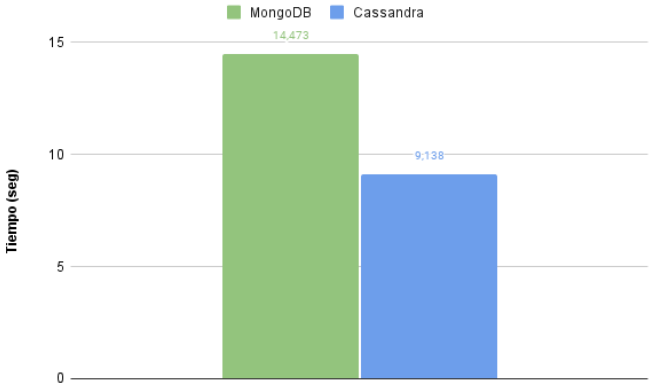

Figura 2: Promedios carga de trabajo A.

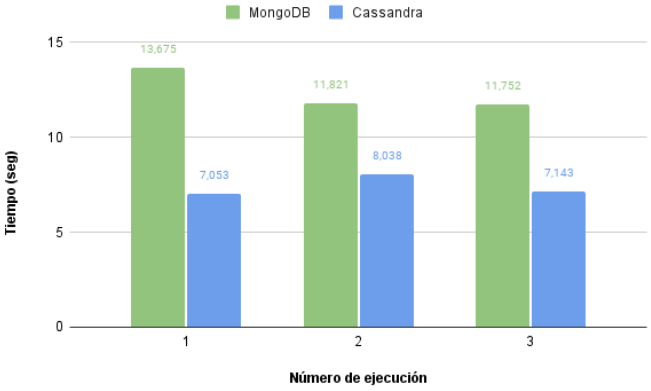

Figura 5: Ejecuciones de carga de trabajo H.

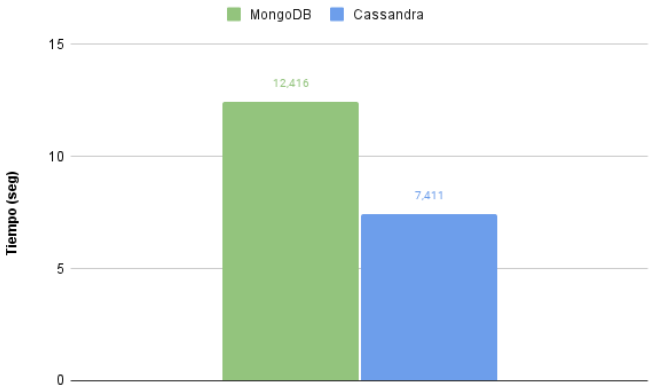

Figura 6: Promedios carga de trabajo H.

VI. Comparativa de resultados

En esta sección se realizara la comparativa de los resultados obtenidos en este artículo y los del artículo [Abramova et al.(2014)Abramova, Bernardino, and Furtado]. Para la comparativa se utilizaran los promedios obtenidos en las ejecuciones de este artículo contra los valores del artículo [Abramova et al.(2014)Abramova, Bernardino, and Furtado].

Como se observa en la Figura 7 los resultados en las cargas de trabajo A y C son consistentes y muestran una mejora en el desempeño de las versiones actuales de MongoDB. Por su parte Cassandra no muestra una mejora consistente en las versiones mas modernas en cuanto a tiempos de ejecución.

La mayor diferencia que se presento fue en los tiempos de la carga de trabajo H donde según [Abramova et al.(2014)Abramova, Bernardino, and Furtado] Cassandra completo la carga en menos de un segundo. Mientras que en el trabajo actual como se ve en las Figuras 5, 6 y 7 le tomo un promedio de siete segundos completar la carga de trabajo.

No se encontraron errores de configuración o de procedimiento en el presente artículo al revisar los resultados sobre la carga de trabajo H. Tampoco existe una diferencia significativa en el hardware empleado, ya que el mismo fue seleccionado con el objetivo de ser lo mas similar posible al empleado en el artículo de referencia. Lo que nos deja como interrogante como lograron ese tiempo de ejecución en dicho artículo.

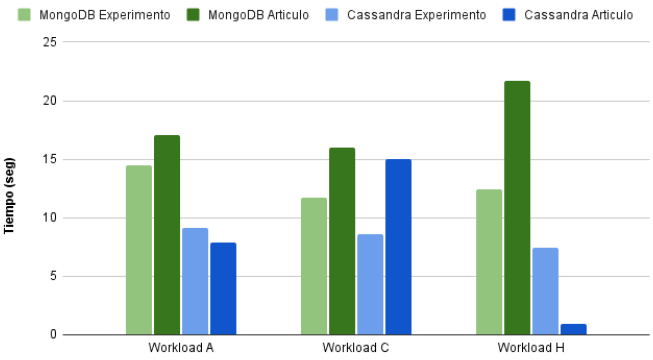

Figura 7: Comparativa de tiempos entre artículo y experimento

VII. Conclusiones

Luego de obtener los resultados de las ejecuciones sobre las tres cargas de trabajo se ve un mejor rendimiento por parte de Cassandra en todos los casos. Si bien esto podría hacer pensar que es la mejor opción en todos los escenarios posibles, esto no es así.

Tanto Cassandra como MongoDB tienen muchas funcionalidades que le dan ventaja a una sobre la otra. Ya sean ventajas inherentes al tipo de base de datos NoSQL que pertenecen o a las implementaciones puntuales.

Destacar que como ya se menciono, MongoDB al pertenecer a las bases de datos documentales suelen ser útiles para manejar grandes conjuntos de datos dinámicos, además de ofrecer una gran experiencia de usuario.

Mientras que Cassandra tiene mas utilidad a la hora de manejar grandes conjuntos de datos con gran velocidad, los mismos no serian tan dinámicos como en el caso documental, pero seria muy beneficioso su uso en escenarios donde la velocidad es importante y el dinamismo de los datos es bajo.

En cuanto a la comparativa de los resultados obtenidos en este artículo y los del artículo [Abramova et al.(2014)Abramova, Bernardino, and Furtado], se concluye que los resultados son consistentes, Cassandra tiene mejores tiempos en todas las pruebas. Dicho esto, en ambas BD se ve importante mejora en los tiempos de ejecución de las BD con respecto a versiones previas.

VIII. Trabajo a futuro

Siguiendo la linea del artículo actual queda por hacer la comparativa con las otras bases de datos presentes en [Abramova et al.(2014)Abramova, Bernardino, and Furtado].

Otro camino a seguir seria definir mejores experimentos de carga que logren ser mas extensivas con las funcionalidades de las bases de datos analizadas en este artículo. Ya que solo tener en cuenta al rendimiento sobre datos aleatorios de las bases de datos no una medida completa para determinar que una base de datos es mejor que la otra.

Un ejemplo de experimento mas real seria medir el rendimiento de las bases de datos en sistemas distribuidos. Donde las cargas de trabajo podrían ser muy similares pero la infraestructura a donde se encontrarían las bases de datos constaría de varios servidores.

Referencias

[Abramova et al.(2014)Abramova, Bernardino, and Furtado] Veronika Abramova, Jorge Bernardino, and Pedro Furtado. Which NoSQL Database? A Performance Overview. *Open Journal of Databases (OJDB)*, 1(2):17–24, 2014.

[Ameya et al.(2013)Ameya, Anil, and Dikshay] Nayak Ameya, Poriya Anil, and Poojary Dikshay. Type of NOSQL databases and its comparison with relational databases. *International Journal of Applied Information Systems*, 5(January 2013):16–19, 2013.