UNIVERSIDAD CÉSAR VALLEJO

FACULTAD DE INGENIERÍA

ESCUELA ACADÉMICO PROFESIONAL DE INGENIERÍA DE SISTEMAS

Modelos Del Rendimiento En Gestores De Base De Datos NoSQL

TESIS PARA OBTENER EL TÍTULO PROFESIONAL DE:

Ingeniero De Sistemas

AUTOR:

Br. Purizaga Quiroga Javier Junnior (ORCID : 0000-0003-2362-8370)

ASESOR:

Mg. More Valencia Rubén Alexander Ing. (ORCID: 0000-0002-7496-3792)

LÍNEA DE INVESTIGACIÓN:

Sistema De Información Y Telecomunicaciones

Piura- Perú

2020

Dedicatoria

A Dios por la vida y a mi familia por el incondicional apoyo, especialmente a mis padres por haber deposito la confianza en mí.

A aquellas personas, amigo y hermanos que contribuyeron con comentarios positivos que me ayudaron en este proceso.

Agradecimiento

Infinitamente a Dios por darme la vida y a mis padres que fueron mi motor para seguir adelante con mi vida universitaria.

También agradezco a mis compañeros que con comentarios y consejos positivos que me ayudaron en el proceso, también agradezco al Ingeniero Rubén More quien con sus asesorías, interrogantes y consejos me ayudaron a esclarecer dudas.

Página del Jurado

UCV UNIVERSIDAD CÉSAR VALLEJO	ACTA DE APROBACIÓN DE LA TESIS	Código : F07-PP-PR-02.02 Versión : 09 Fecha : 23-03-2018 Página : 1 de 1

El Jurado en cargado de evaluar la tesis presentada por don (a) Javier Junnior Purizaga Quiroga cuyo título es: "Modelo del Rendimiento En Gestores de Base de Datos de NoSQL"

Reunido en fecha, escucho la sustentación y la resolución de preguntas por es estudiante, otorgándole el calificativo de: 16 (Dieciséis)

Piura 21 de Enero Del 2020

PRESIDENTE
Dr. Irwing Sáenz Seminario

SECRETARIO
Mg. Winner Agurto Marchan

VOCAL
Mg. Jaime Leandro Madrid Casariego

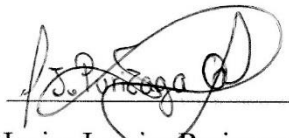
Elaboró	Dirección de Investigación	Revisó	Responsable del SGC	Aprobó	Vicerrectorado de Investigación

Declatoria de Autenticidad

Yo Javier Junnior Purizaga Quiroga, estudiante de la facultad de Ingeniería y de la Escuela Académico Profesional de Ingeniería de Sistemas de la Universidad César Vallejo Filial Piura, identificado con DNI: 71466640, con la tesis titulada "**Modelo de rendimiento de Gestores de Base de Datos NoSQL**".

Por lo tanto asumo totalmente la responsabilidad ante cualquier fraude, encubrimiento o negligencia ante el documento presentado por lo tanto me pongo a disposición de las normas académicas de la Universidad Cesar Vallejo.

Martes, 27 de noviembre del 2018



Javier Junnior Purizaga Quiroga

DNI: 71466640

Índice

Índice de Tablas

Índice de Gráficos

Resumen

La presente investigación tiene como objetivo principal Analizar los resultados de la ejecución del modelo Yahoo Cloud Serving Benchmark, es una investigación de tipo descriptiva cuantitativa no experimental, en la presente vamos a encontrar una definición exacta de Base de Datos NoSQL, además de características y tipos de base de datos NoSQL, también encontraremos un cuadro comparativo entre Apache Phoenix y Yahoo Cloud Serving Benchmark. Por otro lado, hay una muestra de 76 pruebas ejecutadas, estos tiempos fueron recolectados por el instrumento que fue la guía de observación, esto nos conllevo a realizar un Análisis Lineal y los datos fueron procesados en el programa Microsoft Excel 2013. En donde se analizaron los resultados obtenidos de la ejecución de pruebas al modelo en gestores de Base de Datos NoSQL, luego pasamos a discutir los mismos con los antecedentes que tenemos, esta actividad nos conllevo a concluir y dar recomendaciones acerca de líneas futuras.

Palabras claves: Base de datos, rendimiento, tipo de base de datos.

Abstract

The main objective of this research is to Analyze the execution results of the Yahoo Cloud serving model, it is a non-experimental quantitative descriptive type investigation, in this we will find an exact definition of NoSQL database, In addition to features and NoSQL database types, we will also find a cross-table between Apache Phoenix and Yahoo Cloud serving Benchmark. On the other hand, there is a sample of 76 tests executed, these times were collected by the instrument that was the observation guide, this led us to perform a linear analysis and the data were processed in the Microsoft Excel 2013 program. Where we analyzed the results obtained from the execution of tests to the model in database managers NoSQL, then we went on to discuss the same with the background we have, this activity led us to conclude and give recommendations about future lines.

Keywords: database, performance, type of database.

I. INTRODUCCIÓN

La realidad problemática del trabajo de investigación se fundamenta en que, cuando escuchamos gestores de base de datos la mayoría de nosotros lo vinculamos con gestores de base de datos relacionales esto es a que la mayoría hemos escuchado alguna vez o lo hemos utilizado, por otro lado la gran parte no conocemos de este tipo de gestores, porque no se instruyó acerca de estos temas o tuvimos una instrucción muy generalizada acerca del tema y la mayoría de nosotros somos conformistas y no indagamos diferentes gestores de base de datos porque no teníamos el conocimiento necesario para indagar más allá de que se nos instruyó, pero lo que la mayoría no sabemos es que estos gestores de base de datos NoSQL son utilizados por grandes empresas como Amazon, Facebook. (Izquierdo Javier 2015).

Los gestores de base de datos relacionales tienen una forma estructurada de almacenar información y se vinculan entre sí, además las consultas se hacen mediante joins, por consultas multitabla y tienen un esquema claro, además la velocidad de consulta y rendimiento a la hora de trabajar con gran cantidad de datos es lenta, un defecto de estas base de datos es que no son flexible, como por ejemplo todos los valores ingresados deben ser correlativos a los datos pedidos y deben estar validados. La mayoría de base de Datos NoSQL evitando utilizar este lenguaje estructurado o lo utilizan como un lenguaje de apoyo como por ejemplo Casandra utiliza el lenguaje CQL o MongoDB utiliza el lenguaje JSON , a diferencia de las base de datos relacionales que no pueden existir un tipo de relación entre datos o colecciones de documentos, por tanto los gestores de base de datos NoSQL no se usan tablas que utilizan fila y columna aquí se usan modelos de almacenamiento que se utilizan clave-valor, por lo consiguiente existen mayor rapidez a la hora de realizar operaciones de datos, es muy sencillo a la hora de trabajar debido a que tiene un esquema libre, tiene una gran velocidad de consulta y rendimiento a la hora de trabajar gran cantidad de datos, uno de los mayores defectos en esta forma de base de datos es la falta de consistencia a nivel de la base de datos. (Gracia del Busto Hansel y Yanes Enríquez Osmel 2012),(Vergara Jurado Alida 2015).

Si no hubiese este tipo de investigaciones la mayoría de nosotros no tendríamos conocimiento de este tipo de herramientas para medir el rendimiento de base de datos NoSQL o llamadas benchmark es una técnica que sirve para medir el rendimiento de un sistema o de uno de sus componentes, además este tipo de investigaciones generan conocimiento y hacen un gran aporte a la sociedad, por lo consiguiente este tipo de investigaciones nos pueden ayudar a buscar opciones más fáciles para solucionar cierto tipo de problemas, como por ejemplo que modelos utilizar para medir el rendimiento en gestores de base de datos NoSQL, además de saber cuál es el rendimiento de una base de datos NoSQL, que benchmark utilizar para medir el rendimiento de base de datos NoSQL y un conjunto de soluciones que hacen que nuestro día a día sea mucho más sencillo.(Muñoz de Frutos Ana 2016).

Esta investigación tiene como objetivo evaluar el modelo del rendimiento en gestores de base de datos NoSQL, además no existe una respuesta contundente de cuál de los dos gestores de base de datos utilizar, debido a que ambas tienen similares características pero el problema surge a la hora de almacenar información, al respecto estos tipos de base de datos tiene el capacidad para poder adaptarse a los cambios, esto es muy importante debido a que esto va influir a la hora de tomar una decisión. Por otro lado, ambas bases de datos son escalables es decir que nuestro dato tenga el mismo o mejor performance con ciertos problemas que puedan ocurrir, por ejemplo, una base de datos SQL es como una transmisión automática de un vehículo, mientras que las bases de datos NoSQL son como una trasmisión manual. Además, se presume que cualquier cosa que se almacene, tiene que ser una base de datos relacional (SQL), una base de datos relacional puede ser utilizada en diferentes ámbitos como por ejemplo en la Educación: Porque nos a ayuda a organizar la información y también nos ayuda a enseñar el conocimiento lógico al estudiante. Por otro lado, los gestores de base de datos NoSQL se pueden utilizar en redes sociales, el Big Data nos ayuda a gestionar grandes cantidades de información.

Para conocer acerca más del tema, se ha buscado trabajos anteriores, el primero es Raúl, Quispe Yachi (Quispe Yachi, Raúl 2016) en su investigación que lleva como título "Análisis de la bases de datos NOSQL como alternativa a las bases de datos

relacionales", para obtener el grado en de bachiller en Ingeniería de Sistemas en la Universidad Nacional de Ingeniería, Perú, cuyo objetivo es proponer una base de datos adecuada para la gestión de grandes datos y compararlas con las bases de datos relacionales más usadas en las organizaciones mediante volúmenes de información, que a través del método descriptivo propone evaluar las base de datos NoSQL y SQL según criterios a la hora de manejar grandes volúmenes, teniendo como resultado que la mayoría de las pruebas se obtuvo como ganador a Casandra seguido de muy cerca por MongoDB a excepción de una prueba donde MySQL fue ligeramente superior a ambas, concluyendo que as bases de datos NoSQL tienen un mejor rendimiento al manejar grandes volúmenes de información pero eso no quiere decir que es la mejor sal momento de administrar grandes volúmenes de información y es recomendable usar una base de datos NoSQL.(Quispe Yachi, Raúl 2016)

El segundo, Claudiu Barzu (Barzu Claudiu ,2017) en su investigación que lleva como título "Estudio del rendimiento de sistemas de gestión de bases de datos New SQL" para obtener el grado bachiller en Ingeniería Informática, en la Universidad Politécnica de Madrid, España, cuyo objetivo se centra en el estudio del rendimiento de dos nuevos sistemas adaptados a los requerimientos actuales que pretenden ofrecer las funcionalidades de los sistemas tradicionales, como las transacciones y el lenguaje SQL por su facilidad y popularidad, quien a través del método descriptivo propone evaluar el rendimiento con diversas situaciones y con diversos tipos de operaciones, teniendo como resultado que ambos sistemas ofrecen una capa de compatibilidad completa con el lenguaje SQL y un rendimiento similar en situaciones con alta carga de datos, sim embargo el comportamiento entre ambas herramientas es muy diferente, debido a que Apache Phoenix utiliza más tiempo en operaciones de lectura y la herramienta Splice lo utiliza más en operaciones de lectura.(Barzu Claudiu ,2017)

El tercer, Sana Nawazish (Nawazish Ali, Sana 2014a) en su investigación que lleva como título "Benchmarking Distributed System in the Cloud: Yahoo! YCSB" para obtener el grado de bachiller en Ingeniería Informática, en la Universidad Pública de Navarra, España, teniendo como objetivo el estudio, aprendizaje y utilización del modelo

Yahoo Cloud Serving Benchmark, quien a través del método no experimental propone estudiar el rendimiento de base de Datos NoSQL como Casandra y HBase proponiendo ejecutar distintas cargas de trabajo y distintos números de hilos para ver cuál es el rendimiento de las bases de datos ya mencionadas anteriormente ,obteniendo como resultado que algunas pruebas se pudo observar un comportamiento muy similar en la mayoría de las pruebas.(Nawazish Ali, Sana 2014a).

Las teorias Relacionadas al tema , se encontraron diversos conceptos y caracteristicas, NoSQL que significa "Not only SQL" en donde se hace referencia a un conjunto de base de datos no relacionales, teniendo como características que no se encuentran construidas en tablas y no utilizan el lenguaje SQL para la manipulación de Datos. Las bases de datos NoSQL tienen la singular particularidad de poder manejar grandes cantidades de datos, siendo un requerimiento imprescindible para big data(Tovar Ortiz Diego 2017).

Por lo general los Sistemas NoSQL se encuentran diseñados de acuerdo a los parámetros del "ACID (Atomicidad Consistencia Aislamiento Durabilidad), BASE (Basic, Availavility, Soft State, Eventual Consistency), OLTP (Online Transaction Processing) en tiempo real y la solución OLAP (On-line Analytical Processing)" garantizando las necesidades de las organizaciones en sistemas que tienen la capacidad de almacenar y trabajar datos en cantidades descomunales(Cattell Rick 2011).

Las bases de datos NoSQL cuentan con las siguientes características:

- Tienen la capacidad de escalar horizontalmente a largo de varios servidores.
- Tienen la capacidad para replicar y distribuir datos a través de varios servidores.
- Cuentan con una interfaz simple de nivel de llamada o protocolo.
- Modelo de concurrencia débil.
- Apropiado uso de índices distribuidos y memoria RAM.
- Tiene la posibilidad de añadir dinámicamente nuevos atributos a los registros de datos.

A continuación, veremos los modelos más comúnmente utilizados por el sistema de almacenamiento NoSQL:

a. Depósitos de llave-valor: Que por cada llave debemos encontrar un valor asociado, este tipo de valor se almacena de forma binaria (BLOB), este es una de las posibles formas de guardar valores carentes de estructura. Sin embargo, con ayuda de algunas implementaciones de llave-valor pueden soportar listas como valores, algunos beneficios de utilizar llave-valor es su simplicidad y su facilidad para escalar. Es conveniente utilizar los depósitos llave-valor con operaciones simples, basadas en atributos simples, resulta ineficiente a la hora de ejecutar operaciones complejas o por rango. Ejemplos de Sistema de almacenamiento llave-valor son DynamoDB, Redis, SimpleDB(Hecht y Jablonski 2011)

b. Basada en Documentos: Igual manera al sistema de almacenamiento llave-valor hay una clave única y una información, bueno lo que diferencia a este sistema del anterior ya mencionado es que la información pasa a ser un documento con formato JSON o XML, una de la principal limitación de este tipo de datos es que las implementaciones no pueden realizar uniones (JSON) o transacciones que aborden filas o columnas, para resolver este problema recurrimos a las de-normalización. Esta limitación es premeditada, ya que facilita a la base de datos realizar el particionado de forma automática, cabe notar que aunque la de-normalización facilita el escalamiento, pero hace que el costo de actualizaciones de las base de datos incremente y esto puede llevar a la perdida de inconsistencia y propiedades transaccionales.(Hecht y Jablonski 2011)

c. Tabular: Este tipo de sistema en vez de guardar información en filas, las almacenas en columnas, las columnas pueden concentrase en un conjunto llamado familia de columnas, gracias a estos es que estas ganan velocidad no referente a consultas.(Hecht y Jablonski 2011)

d. Orientada a Grafos: Este tipo de base de datos almacenan datos en grafos con nodos y aristas, esto permite elevar la importancia no solo a los datos sino también a la relaciones entre ellos(Hecht y Jablonski 2011)

Dimensión 1 y 2: Rendimiento y latencia

Según (Nawazish Ali, Sana 2014b) “Que para evaluar el rendimiento debemos utilizar los siguientes ítems: Por carga de trabajo y N° Hilos que se ejecutaran”.(p.66)

Según (Barzu Claudiu 2017b) “ Que para medir el rendimiento promedio y latencia promedio debemos utilizar una fórmulas de rendimiento promedio: Throuhput(ops/seg)/ Hilos “

Tabla 1.Cuadro comparativo de Herramientas de Benchmark

MODELO YAHOO CLOUD SERVING BENCHMARK	**APACHE POENIX**
Definición: Es un framework que realiza la medición del rendimiento en sistemas con base de datos que han sido creadas en la nube y para que sean utilizadas. Además realizando estas pruebas obtenemos datos relevantes como por ejemplo el flujo de operaciones, el retraso del procesamiento de las peticiones, la capacidad del procesador a la hora atender las peticiones.	Definición: Es un proyecto que nació con la necesidad de poner devuelta SQL dentro de NoSQL, se puede decir que este es una capa de base de datos relación para HBase , a pesar de ser tan antiguo sigue siendo uno de los lenguajes que más se utiliza a nivel empresarial debido a que es más fácil de utilizar.
Funcionalidad: Por lo general las pruebas suelen ser en dos partes la carga y la ejecución, bueno en el apartado de carga se basa en las inserciones que realizan en la base de datos preparando para la fase de ejecución. La ejecución consta en inserciones, actualizaciones, remociones.	Funcionalidad: Esta herramienta se divide en dos partes, una parte servidora que escucha y la otra que ejecuta querys realizada por la parte del cliente, que instala librerías adicionales en su proyecto
Características: -Que es de código Abierto. -Tiene 06 diferentes pruebas en la herramienta, denominadas carga de trabajo. -Que necesita 06 máquinas tipo servidor	Características: -Que ofrece todas las ventajas de SQL combinadas con las ventajas de NoSQL.

	-Capacidad de Escalar y ofrecer un mayor Throguthput pudiendo llegar a millones de operaciones por segundo. -Trabajo con Nodos

El trabajo de investigación tiene la siguiente formulación del problema

¿Cuál es el rendimiento de base de datos MongoDB utilizando el modelo YCSB?

La justificación del trabajo de investigación se dividió en:

- Justificación Teórica: La investigación se realizó con el propósito de reforzar los conocimientos acerca de la definición de una base de datos NoSQL, sus características y tipos de base de datos NoSQL teniendo como soporte a los antecedentes, en este estudio se aplicó el instrumento que es una guía de observación, cuyos resultados de esta investigación nos sirvieron para ver los puntos fuertes y débiles, además de comparar resultados con la tesis de (Nawazish Ali, Sana 2014b) cuya teoría de que el la latencia de las base de datos va aumentando según aumentan los hilos fue afirmada.
- Justificación Práctica: La investigación se realizó con la necesidad de saber el rendimiento de base de NoSQL, para ver los puntos fuertes y débiles de las base de datos NoSQL, para saber que si estoy utilizando la base de datos adecuada y en qué casos.
- Justificación Metodológica: La investigación es de tipo descriptiva no experimental, se procedió a la elaboración y aplicación de guías de observación, una guía de observación que es para el rendimiento promedio y otra guía de observación para la latencia. Las mismas que fueron validadas por expertos en el tema.

El trabajo de investigación tuvo como objetivo general:

- Analizar los resultados obtenido del modelo Yahoo Cloud Serving Benchmark

El trabajo de investigación tuvo como objetivos específicos:

- Comparar los resultados obtenidos de del modelo de rendimiento en Gestores de Base de Datos NoSQL.
- Evaluar el rendimiento promedio que se obtendrán en la ejecución de pruebas
- Evaluar la latencia promedio que se obtendrán en la ejecución de pruebas.

II. MÉTODO

2.1. Diseño de la Investigación

La presente investigación llegó a un nivel Exploratorio/Descriptiva debido a que en primer lugar haremos un sondeo de los "modelos para evaluar el rendimiento de base de Datos No SQL", que existen en el mercado y poder así describir las virtudes de cada una de estos. El análisis de los datos recolectados será realizado a través de métodos estadísticos por ende la investigación tendrá un enfoque cuantitativo.

Fue de tipo No Experimental dado que únicamente se evaluaran las características encontradas de los modelos más adecuados según las características de base de datos. Tuvo un corte transversal dado que se realizará en un tiempo determinado.

2.2. Variables, Operacionalización

Tabla 2.Matriz de Operacionalización de Variables

Variables	Definición Conceptual	Definición Operacional	Dimensiones	Indicadores	Técnicas	Instrumentos
Bases de Datos NoSQL	NoSQL significa "Not only SQL" en donde se hace referencia a un conjunto de base de datos no relacionales, teniendo como una principal característica que no están construidas en tablas y no utilizan el lenguaje SQL para la manipulación de datos.(Tovar Ortiz Diego 2017)	Lo que procederemos a hacer es realizar pruebas de rendimiento y latencia para encontrar cual es el rendimiento de la base de datos MongoDB utilizando el modelo Yahoo Cloud Serving Benchmark(YCSB)	Rendimiento	Carga de Trabajo Ejecutadas	Observación	Guía de Observación
				Hilos de Ejecución.		
			Latencia	Carga de trabajo Ejecutadas		
				Hilos de Ejecución		

Fuente: Elaborada por Javier J. Purizaga Quiroga

2.3. Población y Muestra

Tabla 3.Matriz de Muestra

Unidad de Análisis	Muestra
N° de hilos ejecutados por prueba	Fichas 76

2.4. Técnicas e Instrumentos de Recolección de Datos, validez y confiabilidad

Tabla 4.Matriz de Instrumentos de Recolección de Datos, validez y confiabilidad

Ítems	Técnica	Instrumento
Carga de Trabajo Ejecutadas	Observación	Guía de Observación
N° Hilos de ejecución	Observación	Guía de Observación
Carga de Trabajo Ejecutadas	Observación	Guía de Observación
N° Hilos de Ejecución	Observación	Guía de Observación

2.5. Método de Análisis

Con respecto a la guía de observación, después de aplicación de la guía de observación al modelo se aplicó la fórmula de rendimiento máximo para analizar, comparar y evaluar los resultados que se reunieron de acuerdo a las guías de observación.

Así mismo se realizar el análisis estadístico.

Para hallar el rendimiento promedio vamos a utilizar la siguiente formula.

Formula de Rendimiento promedio:

$$Rendimiento\ Promedio\ = \frac{Throughput(ops/sec)}{\text{N° Hilos de Carga}}$$

2.6. Aspectos Éticos

La realización del proyecto de investigación ha sido estructurada tomando como referencia el modelo aprobado por la Universidad César Vallejo y se han utilizado las normas ISO 690 para mencionar las bibliografías y referencias de los autores citados.

III. RESULTADOS

3.1. Detalles de Ejecución de Pruebas

Para la realización de este proyecto se utilizó una maquina física con las siguientes características:

- Procesador Intel(R) Core(TM)i5 4550U @3.8GHz
- RAM: 12,00 GB (10,20 GB disponible)
- Disco duro: 1 TB
- Sistema Operativo: Centos 7 Gnone(64bits)
- Tarjeta Red 10/100
- MongoDB v.4.0.0
- Hadoop v.3.5.0
- Tipo de Fichero: Documento

Configuración del Clúster

- Simple 127.0.1.1

Workloads

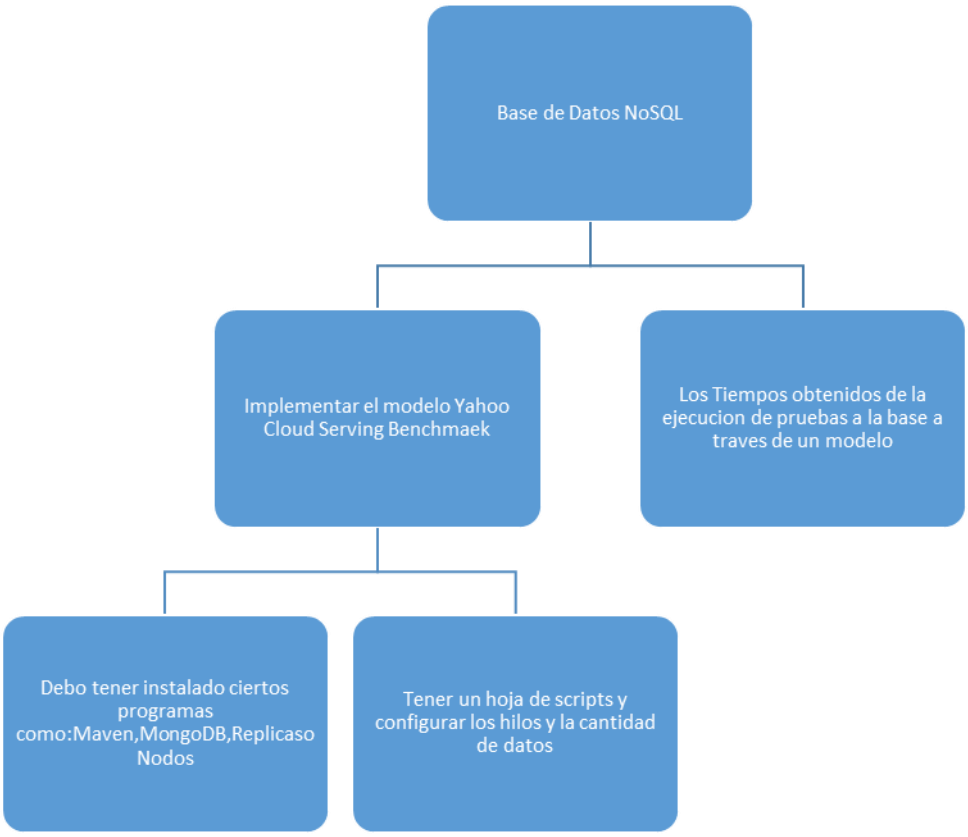
- 700 000 registros de 1KB cada uno en una máquina de 10GB
- 100 threads
- Distribución Zipfian

Tipos de Operaciones

- Insert: Añade un nuevo registro

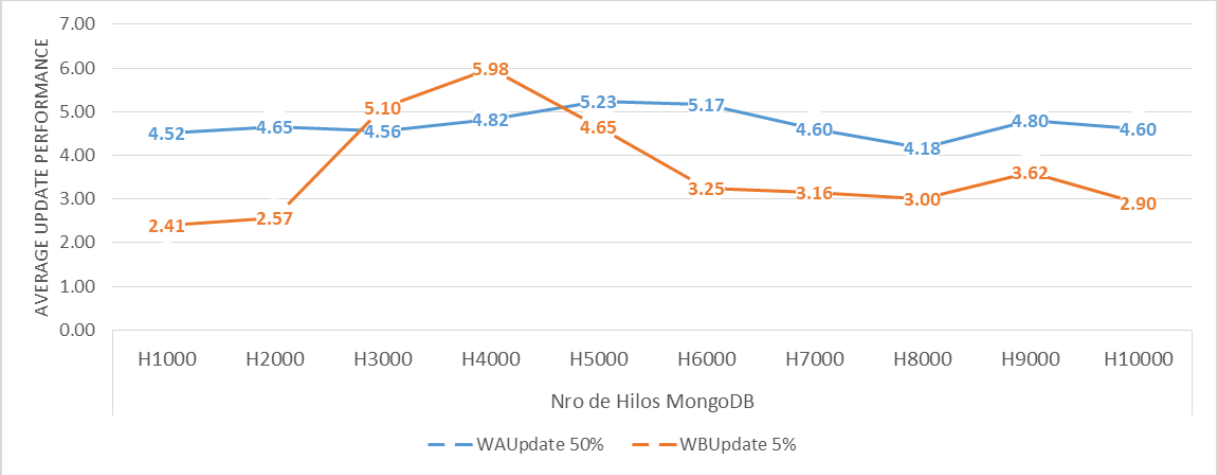
- Update: Cambia el valor de un registro existente en la base de datos.
- Read: Lee un registro de manera aleatoria.
- Scan: Escanea registros en orden, empezando por una clave de registro seleccionada al azar. El número de registros que deben analizarse también es seleccionado al azar entre 1y 100

3.2. Modelo Conceptual de Base de Datos NoSQL



3.3. Análisis de la ejecución del Modelo Yahoo Cloud Serving Benchmark

Gráfico 1..Comparativa de Actualizaciones (Update) de Rendimiento 50% y 5%

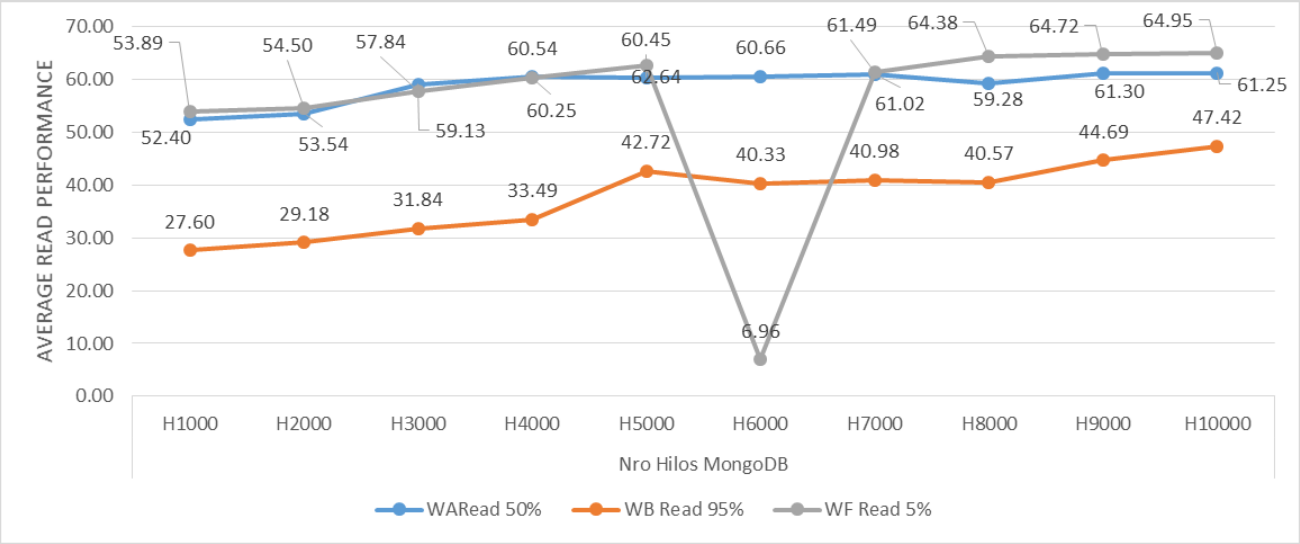


Fuente: Guía de Observación aplicada al modelo
Autor: Javier Junnior Purizaga Quiroga.

Interpretación

En el grafico 1 se pudo notar que un pico ascendiente entre los puntos 4.56 a 5.10 y que decae entre los puntos 5.98 a 3.25 debido a que algún fichero que fue seleccionado en la prueba contenía una serie de colecciones y cada colección puede obtener diferentes objetivos, esto hizo que hubiera mayor demanda de recursos y luego denotamos descenso esto significa que los demás ficheros que fueron seleccionados no tuvieron este tipo de problema.

Gráfico 2.Comparativa de Lectura (Read) de Rendimiento 50%,95% y 5%.

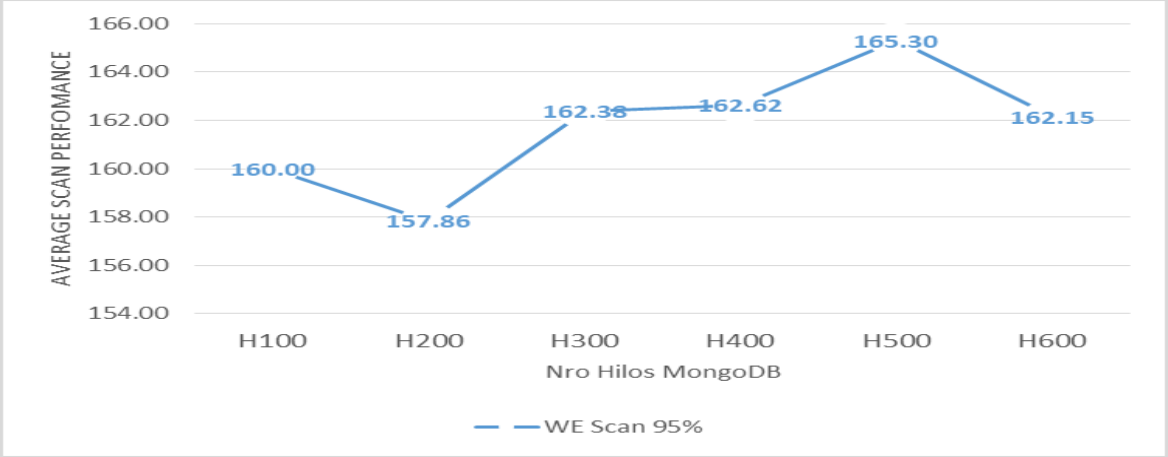


Fuente: Guía de Observación N°1 aplicada al modelo
Autor: Javier Junnior Purizaga Quiroga

Interpretación

Como se logró apreciar en el grafico N°2 se muestra que entre los puntos 62.64 y 6.96 notamos un descenso perpendicular, este tipo de problemas suelen pasar porque a la hora de filtrar el fichero hubo alguna dificultad ya que hubo diferentes colecciones en el fichero o porque hubo varios objetivos. Además, también podemos percibir que hay un ascenso perpendicular en donde la prueba sigue con normalidad esto hace referencia a que se pudo procesar la información que el fichero contenía y por otra parte se pudo observar si la información que contenía el fichero iba dirigido a un objetivo a varios Además las pruebas READ al 95% no presentaron este tipo de problemas, debido a que toda la data que contenían los ficheros iba dirigida a un solo objetivo.

Gráfico 3.Comparativa de Rendimiento en Escaneo (Scan) al 95%



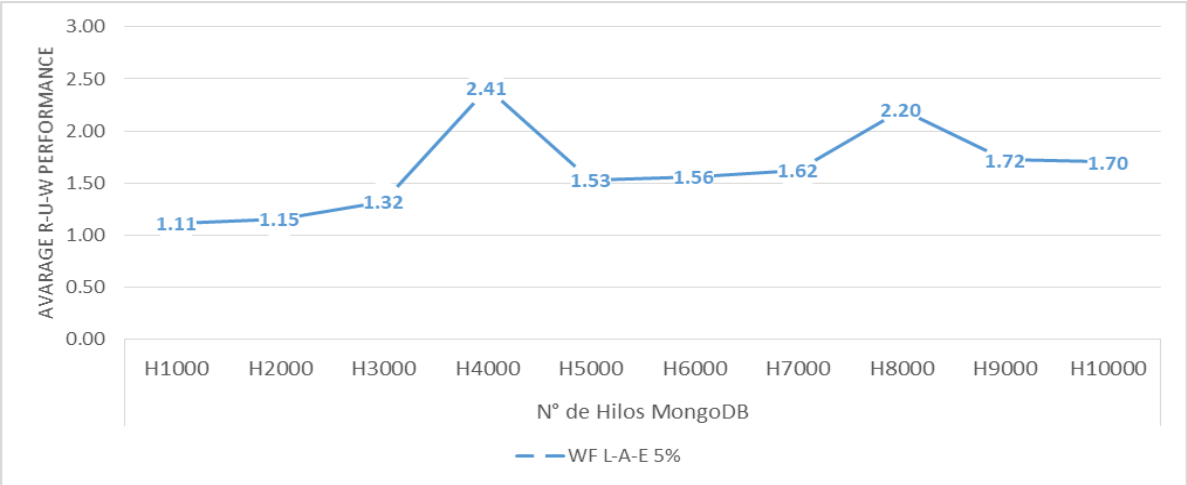
Fuente: Guía de Observación N°1 aplicada al modelo

Autor: Javier Junnior Purizaga Quiroga

Interpretación

En el grafico N°3 se pudo percibir que hay desniveles uno está entre los puntos 160.0 a 157.86 se puede explicar de la siguiente forma que hubo algunos ficheros que contenían datos precisos y con un solo objetivo, además se debe tomar en cuenta que los ficheros para esta prueba se seleccionaran de forma aleatoria de 1 al 100.

Gráfico 4.Comparativa Lectura, Actualización y Escaneo 5%



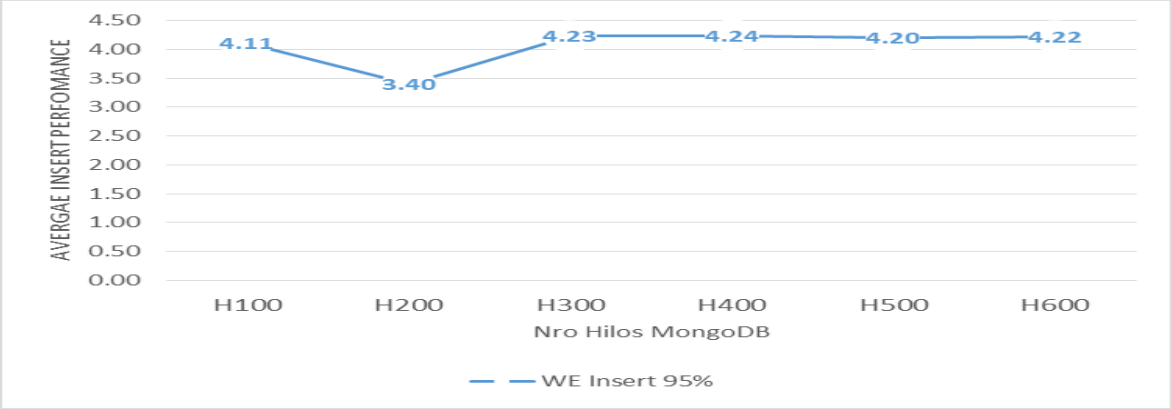
Fuente: Guía de Observación N°1 aplicada al modelo

Autor: Javier Junnior Purizaga Quiroga

Interpretación

Como se denoto que el grafico N°4 de Lectura, Actualización y Escaneo, se pudo percibir que hay varios desniveles, uno de ellos se encuentra citado entre los puntos entre los puntos 1.32 y 2.41 este puede pasar porque hubo incongruencias a la hora filtrar datos en el fichero o porque hubo ficheros que contenían un conjunto de colecciones que iban dirigidas a varios objetivos y a consecuencia de esto se requirió de mayor consumo de recursos. Además, también pudimos percibir que entre los puntos 4.62 y 2.20 también encontramos un desnivel esto se puede explicar con lo mencionado en líneas anteriores.

Gráfico 5.Comparativa de Insert 95%



Fuente: Guía de Observación aplicada al modelo N°1
Autor: Javier Junnior Purizaga Quiroga

Interpretación

Como denotamos en el grafico N°5 podemos percibir que hay un descenso entre los 4.11 al punto 3.40 esto se debe a que en el momento que se ejecutó la consulta INSERT hubo algún fichero que contenía data para una sola colección y esta iba dirigida hacia un solo objetivo y se observó ese desnivel, además la prueba sigue con normalidad también debemos tomar en cuenta que los ficheros fueron seleccionados aleatoriamente de 1 al 100.

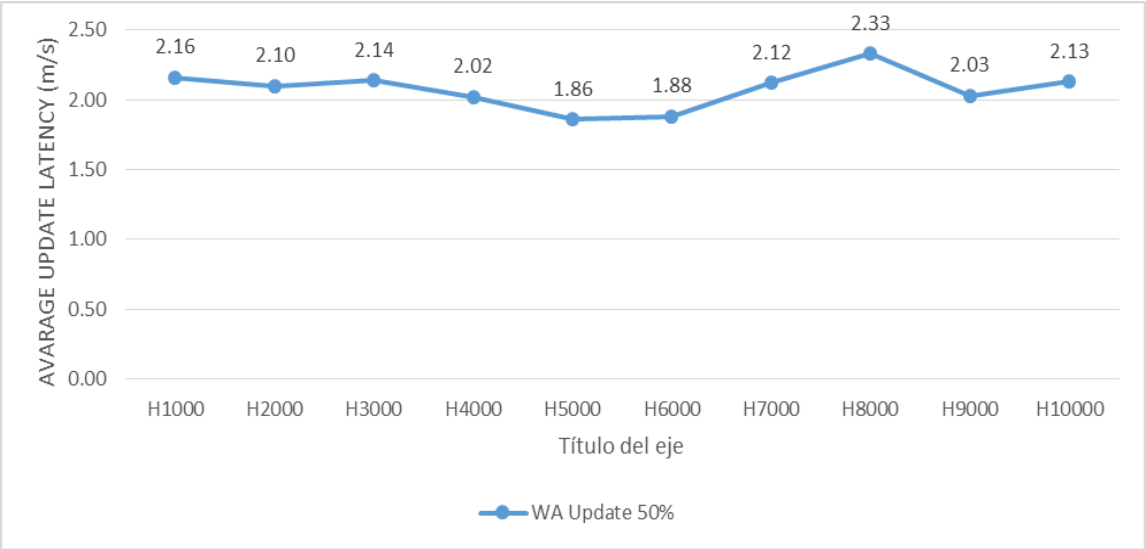
Gráfico 6.Comparativa de Latencia Promedio al ejecutar Consulta READ 50%

Fuente: Guía de Observación Aplicada N°2 al modelo
Autor: Javier Junnior Purizaga Quiroga

Interpretación

Como se observó en el grafico N°6 la latencia promedio obtenida de la ejecución UPDATE la gráfica tiene una forma que va de ascendiente hacia descendiente, este paradigma se puede explicar debido a que para ejecutar la sentencia UPDATE se ejecutaran dos órdenes implícitas, una de ella es para filtrar las colecciones y la otra consulta es la de modificar el fichero que puede contener una serie de colecciones.

Gráfico 7.Comparativa de Actualizaciones de Latencia Promedio 50%



Fuente: Guía de Observación N°2 aplicada al modelo

Autor: Javier Junnior Purizaga Quiroga

Interpretación

Como se pudo apreciar en el grafico N°7 la latencia obtenida de la ejecución UPDATE la gráfica tiene desnivel esto se debe que para que MongoDB ejecute la orden READ primero tiene que filtrar que fichero que puede contener una o varias colecciones y por otro lado la segunda orden que está implícita es que una vez que se encontró el fichero se puede leer lo que contiene el fichero o modificar.

IV. DISCUSIÓN

De acuerdo al objetivo el cual fue analizar el modelo de rendimiento en gestores de base de Datos NoSQL .A partir de los datos obtenidos de la aplicación de la guía de observación al modelo Yahoo Cloud Serving Benchmark en donde se obtuvo una comparativa del rendimiento en las actualizaciones de 50% y 5 % se puede apreciar un notorio desnivel en la curvatura que comprende entre los puntos 4.56 al 5.98 esto es debido a que en la ejecución de este consulta que es la Actualización , se hizo la búsqueda del fichero o colecciones que fue seleccionada de forma aleatoria entre 1 al 100 y luego de encontrado el fichero se obtiene los elementos que contiene el mismo y se procede a la actualización del fichero o colección. A pesar de no haber los mismos tiempos que mi antecedente de (Nawazish Ali, Sana 2014c) que encuentra una latencia con distintos valores pero se afirma que la Latencia entre

Casandra y MongoDB aumenta conforme aumentan el número de hilos , aquí e pude observar que Casandra no obtiene resultados tan optimizado para realizar consultas de actualización con respecto a HBase que obtiene mejores resultados en latencia ya que esta cuenta con una memoria volátil en donde se almacenan las actualizaciones y se puede observar los valores que están por debajo de 1 , muy diferente a Casandra que obtiene valores por encima de 1 hasta 3 por otra parte MongoDB obtiene resultados posterior de 1 hasta 2.33 teniendo resultados más óptimos que Casandra.

Se Analizó los resultados obtenidos de la ejecución de la consulta de Scan al 95%con los resultados obtenidos de la misma ejecución de consulta con el antecedente ya antes mencionado, bueno los resultados de la ejecución de la consulta de Scan por del antecedente fueron la latencia en HBase fue entre los puntos 34,90 a 56,43 teniendo como pico más alto a 56,43 y Cassandra con resultados entre los puntos 49,36 a 31,93 teniendo como pico alto a 49,36, por otro lado MongoDB o tuvo resultados entre los puntos de 160,00 a 162,15 teniendo como pico más alto a 165,30, en donde se puede decir que HBase es la que ha tenido un comportamiento mejor que las demás.

Con respecto al segundo objetivo que fue comparar el modelo en gestores de base de datos NoSQL, considerando que se aplicó la guía de observación al modelo Yahoo Cloud Serving Benchmark en donde se encontró que en la comparativa de Rendimiento en Lecturas al 50%,95% y 5% pudimos contemplar que entre la lecturas 50% y 5 % se encontró una similitud en la curvatura excepto por un desnivel hacia abajo dado que para ejecutar la consulta READ primero se tiene que hacer una búsqueda de la colección que fue seleccionada de forma aleatoria entre 1 y 100 esto hizo que haya un descenso en la prueba porque se tuvo que buscar una colección que no estaba al alcance de la prueba, además luego miramos que hay un ascenso eso significa que hubo una búsqueda de la colección donde este estuvo al alcance de la prueba y se procedió a ejecutar la consulta READ lo que contenía cierta colección esto es con respecto a la comparativa de Rendimiento en Actualizaciones al 50% y 5% con respecto a la actualización del 95% se pude observar que por debajo de la actualización del 50% esto es debido a que todas las colecciones estuvieron a la mano de la prueba. Por otro lado en comparativa de Latencia en Read al 50 % obteniendo resultado como

18.66 desciendo a 16.10 muy diferente al resultado obtenido por (Nawazish Ali, Sana 2014c) que la latencia en la base de datos HBase un poco más alta debido a que este tipo de base de datos tiene que reconstruir datos, por otra parte la base de datos Casandra obtuvo entre 12.23 hasta 19.56 siendo el pico más alto 19.56 y en MongoDB dando como resultado los valores entre 18.66 hasta 16.10,en conclusión MongoDB es la base de Datos más optimizado para la ejecución de Lectura esto se debe a que mongo hace consultas simples y que una colección tiene un conjunto de elementos o un elemento.

Así mismo en la investigación ya antes mencionada donde se aplicó la guía de observación al modelo en donde se ejecutó consulta de Read-update-write se hallaron resultados que iban por debajo de 1 en ambas bases de datos, esta pruebas se ejecutaron througputs bajos para observar más de cerca el comportamiento de ambas bases, se puede observar que en ambas base de datos el comportamiento es similar y la latencia es muy bajo todo lo contrario a la base de datos MongoDB los resultados que se obtuvieron fueron por encima 1 hasta 1.70 teniendo con pico más alto al valor 2.41 esto nos conlleva a expresar que las base de datos HBase y Casandra son las más óptimas para ejecutar esta consulta muy diferente a MongoDB.

Con respecto a los tiempos de la ejecución de la consulta de Lectura al 5% y 95% obteniendo como resultado observamos un desnivel en la curvatura de ejecución de la consulta al 5% entre los puntos 4,56 a 5,98esto a causa de que hubo algún fichero que tuvo que ser buscado porque no estaba al alcance del cloud por parte de la ejecución de la consulta lectura al 50% no se denoto ningún desnivel por otro lado según el autor (Barzu Claudiu 2017c) no dice que se puede observar como el tiempo de lectura no aumenta tanto al variar el tamaño de la base esto debido a que las curvaturas son semejantes , muy contrario a MongoDB que obtuvimos graficas no tan semejantes y además con un desnivel.

V. CONCLUSIONES

- Tras haber Analizado la ejecución del modelo Yahoo Cloud Serving Benchmark pudimos notar que hubo un desnivel de las actualizaciones al 5% y 95% esto se debe a que en la prueba hubo una búsqueda de un fichero que no estaba al alcance de la prueba y a causa de esto hizo que hubiera un mayor consumo de recursos por parte de la prueba y además pudimos observar un punto débil para este modelo al ejecutar una Actualización al 5%.
- Después de haber comparado los resultados de la ejecución del modelo Yahoo Cloud Serving Benchmark pudimos notar que un desnivel de la lectura al 5% esto porque hubo una dificultad para buscar este fichero que fue elegido al azar entre 1 al 100 y a causa de esto hubo un descenso perpendicular.
- Al comparar los tiempos de ejecución obtenidos del modelo Yahoo Cloud Serving Benchmark se pudo deducir que la base de Datos MongoDB no tiene los mejores tiempos a la hora de hacer un escaneo de ficheros.
- Al evaluar latencia pudimos observar que la latencia obtenida de la ejecución de la sentencia READ tiene varios desniveles esto se debe a que primero se tuvo que filtrar el fichero que se desee actualizar, para esto se utilizó WHERE Y SET.
- Al evaluar la latencia se pudo percatar que la latencia obtenida de la ejecución de la sentencia UPDATE tiene un desnivel que asciende esto se debe a que los ficheros estuvieron al alcance de la prueba.

VI. RECOMENDACIONES

- Implementar un Análisis de Herramienta de Benchmark en Base de Datos NoSQL p para esto se debe contar con datos reales de una organización, contar con un servidor en donde poder ejecutar la herramienta, y una base de datos NoSQL que contenga más datos.
- Se utilizar la base de Datos NoSQL MongoDB para la implementación de un sistema de biblioteca digital porque los mejores tiempos a la hora de ejecutar estas tres operaciones juntas que son Leer-Actualizar-Escribir.
- Se recomienda la base de datos NoSQL MongoDB en la implementación de citas médicas ya que este tipo de base de base tiene un tiempo que es destaco a la hora de ejecutar tres operaciones a la vez Leer-Actualizar-Escribir, esto es crucial para este tipo de problemas.
- A pesar de que este tipo de trabajos está en pañales se puede hacer comparaciones con otras bases de datos como, por ejemplo: MySQL-MongoDB, CouchDB-MongoDB.

REFERENCIAS

1. BARZU CLAUDIU, 2017. *Estudio del rendimiento de sistemas de gestión de bases de datos New SQL* [en línea]. Masters. S.l.: E.T.S. de Ingenieros Informáticos (UPM). [Consulta: 12 septiembre 2018]. Disponible en: http://oa.upm.es/47291/.
2. BAS ABAD VICENTE JESÚS, 2015. Estudio comparativo de BBDD relacionales y NoSQL en un entorno industrial. [en línea]. [Consulta: 1 octubre 2018]. Disponible en: https://riunet.upv.es/handle/10251/55530.
3. CAJAS MALAVÉ EDUARDO FABIAN, 2011. *Estudio sobre las Bases de Datos NOSQL Orientado a los Servicios Web 2.0* [en línea]. Thesis. S.l.: Universidad de Guayaquil. Facultad de Ciencias Matematicas y Fisicas. Carrera de Ingeniería en Sistemas Computacionales. Carrera de Ingeniería en Networking y Telecomunicaciones. [Consulta: 1 octubre 2018]. Disponible en: http://repositorio.ug.edu.ec/handle/redug/6645.
4. CAMARGO-VEGA, J.J., CAMARGO-ORTEGA, J.F. y JOYANES-AGUILAR, L., 2015. Knowing the Big Data. *Facultad de Ingeniería*, vol. 24, no. 38, pp. 63-77. ISSN 0121-1129.
5. CARABIO, A.L.R., BENEDETTO, M.G. y FALAPPA, M.A., 2016. Comportamiento de bases de datos no relacionales en entornos distribuidos. *XVIII Workshop de Investigadores en Ciencias de la Computación (WICC 2016, Entre Ríos, Argentina)* [en línea]. S.l.: s.n., [Consulta: 27 noviembre 2018]. ISBN 978-950-698-377-2. Disponible en: http://hdl.handle.net/10915/52954.
6. CARDENAS, J.E.S., 2014. ANÁLISIS COMPARATIVO DE DOS BASES DE DATOS SQL Y DOS BASES DE DATOS NO SQL. , no. 2014, pp. 80.
7. CATTANEO, M.F.P., NOCERA, M.L. y ROTTOLI, G.D., 2014. Rendimiento de tecnologías NoSQL sobre cantidades masivas de datos. *CUADERNO ACTIVA*, no. 6, pp. 11-17. ISSN 2619-5232.
8. CATTELL, R., 2011. Scalable SQL and NoSQL data stores. *ACM SIGMOD Record*, vol. 39, no. 4, pp. 12. ISSN 01635808. DOI 10.1145/1978915.1978919.
9. CHACALTANA GONZALO, 2017. El poder de las bases de datos NoSQL | $> SoloCodigoWeb. *Solo Codigo Web* [en línea]. [Consulta: 22 noviembre 2018]. Disponible en: http://www.solocodigoweb.com/blog/2017/05/22/el-poder-de-las-bases-de-datos-nosql/.
10. CHAVEZ, O., 2013. Administracion De Base De Datos: Rendimiento de una base de datos. *Administracion De Base De Datos* [en línea]. [Consulta: 12 septiembre 2018]. Disponible en: http://chavez-atienzo-2013.blogspot.com/2013/04/rendimiento-de-una-base-de-datos.html.
11. GÓMEZ, R.H. y MAQUEDA, A.M.I., [sin fecha]. BASES DE DATOS NOSQL: ARQUITECTURA Y EJEMPLOS DE APLICACIÓN. , pp. 159.
12. GRACIA DEL BUSTO HANSEL y YANES ENRÍQUEZ OSMEL, 2012. Bases de datos NoSQL. *Telem@tica (La Habana)*, vol. 11, no. 3, pp. 21-33. ISSN 1729-3804.
13. HECHT, R. y JABLONSKI, S., 2011. NoSQL evaluation: A use case oriented survey. *2011 International Conference on Cloud and Service Computing*. S.l.: s.n., pp. 336-341. DOI 10.1109/CSC.2011.6138544.
14. Introducción a las bases de datos NoSQL usando MongoDB - Free Download PDF. [en línea], 2018. [Consulta: 1 octubre 2018]. Disponible en:

https://kupdf.net/download/introduccion-a-las-bases-de-datos-nosql-usando-mongodb_59d0347908bbc53e5168701b_pdf.
15. IZQUIERDO JAVIER, 2015. NoSQL vs SQL: Principales diferencias y cuándo elegir cada una de ellas. [en línea]. [Consulta: 1 octubre 2018]. Disponible en: https://blog.pandorafms.org/es/nosql-vs-sql-diferencias-y-cuando-elegir-cada-una/.
16. LOPEZ PEÑA, CARLOS ANDRES, 2012. *Análisis de las bases de datos NOSQL como alternativa a las bases de datos SQL* [en línea]. Thesis. S.l.: Administrativa, Financiera, Sistemas y Computación. [Consulta: 1 octubre 2018]. Disponible en: http://repository.eia.edu.co/handle/11190/411.
17. MARTÍN, A.E., CHÁVEZ, S.B., RODRÍGUEZ, N.R., VALENZUELA, A. y MURAZZO, M.A., 2013. Bases de datos NoSQL en cloud computing. *XV Workshop de Investigadores en Ciencias de la Computación* [en línea]. S.l.: s.n., [Consulta: 12 septiembre 2018]. Disponible en: http://hdl.handle.net/10915/27121.
18. MARTÍN, S., 2017. Bases de datos NoSQL : Guía definitiva. [en línea]. [Consulta: 1 octubre 2018]. Disponible en: https://blog.pandorafms.org/es/bases-de-datos-nosql/.
19. MATÉ JIMÉNEZ, C., 2014. Big data. Un nuevo paradigma de análisis de datos. *Revista: Anales de Mecánica y Electricidad, Periodo: 1, Volumen: XCI, Número: VI, Página inicial: 10, Página final: 16* [en línea], [Consulta: 27 noviembre 2018]. ISSN 0003-2506. Disponible en: https://repositorio.comillas.edu/xmlui/handle/11531/4873.
20. MENDOZA, E., 2017. ¿Cómo saber si necesitas una Base de Datos NoSQL? *Eugenio Mendoza* [en línea]. [Consulta: 22 noviembre 2018]. Disponible en: https://medium.com/@eugeniomendoza/c%C3%B3mo-saber-si-necesitas-una-base-de-datos-nosql-b6cfd5bb7d9b.
21. MIGANI, S., VERA, C. y LUND, M.I., 2018. NoSQL: modelos de datos y sistemas de gestión de bases de datos. *XX Workshop de Investigadores en Ciencias de la Computación (WICC 2018, Universidad Nacional del Nordeste).* [en línea]. S.l.: s.n., [Consulta: 22 noviembre 2018]. ISBN 978-987-3619-27-4. Disponible en: http://hdl.handle.net/10915/67258.
22. MUÑOZ DE FRUTOS ANA, 2016. ¿Qué es un benchmark y para qué sirve? *ComputerHoy* [en línea]. [Consulta: 7 octubre 2018]. Disponible en: https://computerhoy.com/noticias/moviles/que-es-benchmark-que-sirve-40273.
23. NAWAZISH ALI, SANA, 2014. Benchmarking distributed systems in the cloud: Yahoo! YCSB. [en línea], [Consulta: 7 octubre 2018]. Disponible en: https://academica-e.unavarra.es/xmlui/handle/2454/10203.
24. QUISPE YACHI, RAÚL, 2016. Análisis de la bases de datos NOSQL como alternativa a las bases de datos relacionales. *Universidad Nacional de Ingeniería* [en línea], [Consulta: 1 octubre 2018]. Disponible en: http://cybertesis.uni.edu.pe/handle/uni/4017.
25. REA PEÑAFIEL XAVIER MAURICIO, 2016. Análisis comparativo entre la base de datos no relacional Mongodb con la base de datos Postgresql, sistema para la gestión de clientes y registro de pagos de la Clínica Odontológica Ortho Dent. [en línea], [Consulta: 1 octubre 2018]. Disponible en: http://repositorio.utn.edu.ec/handle/123456789/4661.
26. RODRÍGUEZ PÉREZ, A., RODRÍGUEZ HERNÁNDEZ, D. y DÍAZ MARTÍNEZ, E., 2016. Selección de Base de Datos No SQL para almacenamiento de Históricos

en Sistemas de Supervisión. *Revista Cubana de Ciencias Informáticas*, vol. 10, no. 3, pp. 159-170. ISSN 2227-1899.
27. ROMERO, A.C., SANABRIA, J.S.G. y CUERVO, M.C., 2012. Utilidad y funcionamiento de las bases de datos NoSQL. *Revista Facultad de Ingeniería*, vol. 21, no. 33, pp. 21-32. ISSN 2357-5328.
28. RÓTTOLI, G., LÓPEZ NOCERA, M., CATTANEO, P. y FLORENCIA, M., 2015. Utilización de NoSQL para resolución de problemas al trabajar con cantidades masivas de datos. *XVII Workshop de Investigadores en Ciencias de la Computación (Salta, 2015)* [en línea]. S.l.: s.n., [Consulta: 1 octubre 2018]. Disponible en: http://hdl.handle.net/10915/45514.
29. SARASA CABEZUELO ANTONIO, 2018. Introducción a las bases de datos NoSQL usando MongoDB :: Editorial UOC - Editorial de la Universitat Oberta de Catalunya. [en línea]. [Consulta: 1 octubre 2018]. Disponible en: http://www.editorialuoc.com/introduccion-a-las-bases-de-datos-nosql-usando-mongodb.
30. TOVAR ORTIZ DIEGO, 2017. Bases de datos NOSQL en Big Data. [en línea]. [Consulta: 22 noviembre 2018]. Disponible en: https://repository.unilibre.edu.co/handle/10901/11213.
31. VALBUENA, S.J. y LONDOÑO, J.M., 2014. SISTEMAS PARA ALMACENAR GRANDES VOLÚMENES DE DATOS. *Revista GTI*, vol. 13, no. 37, pp. 17-28. ISSN 2027-8330.
32. VERGARA JURADO ALIDA, 2015. SQL vs NoSQL ¿Cuál debo usar? *Tech blog for developers | Facilcloud* [en línea]. [Consulta: 7 Mayo 2018]. Disponible en: https://www.facilcloud.com/noticias/sql-vs-nosql-which-one-should-i-use/.

ANEXOS

Instrumento N°1

Guía de observación N°1

En la siguiente guía de observación sirve para anotar datos obtenidos de pruebas ejecutadas a distintas bases de datos No SQL con el fin hallar su rendimiento.

Modelo:
Lugar de ejecución:
Fecha de ejecución:
Responsable:

Carga de trabajo	H1000	H2000	H3000	H4000	H5000	H6000	H7000	H8000	H9000	H10000
Workload A Read 50%										

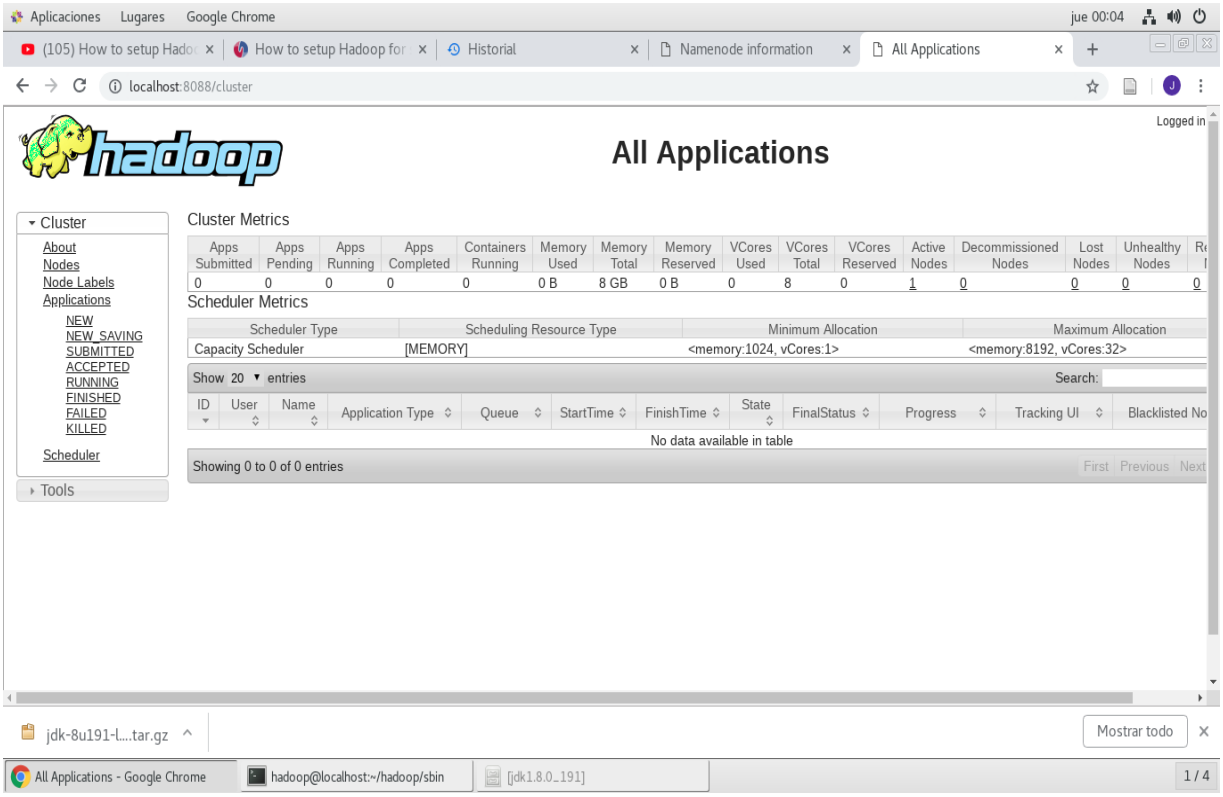
Consideraciones:

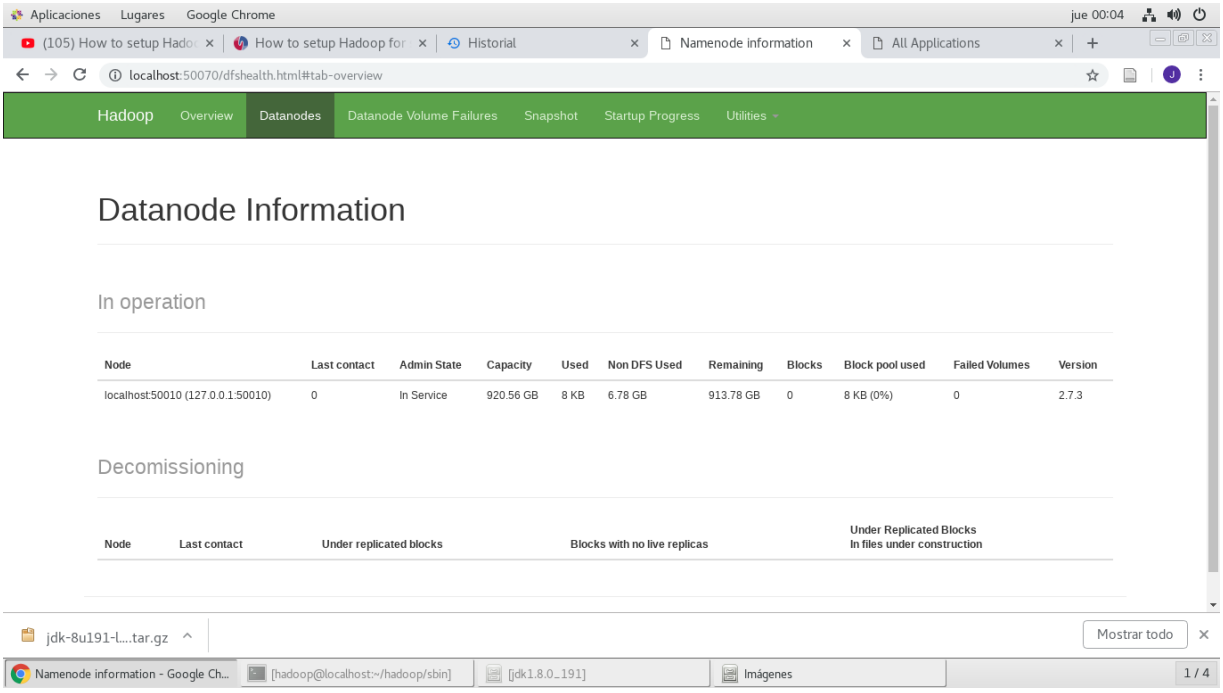
Guía de observación N°2

Modelo:
Lugar de ejecución:
Fecha de ejecución:
Responsable:

Carga de trabajo	H1000	H2000	H3000	H4000	H5000	H6000	H7000	H8000	H9000	H10000

Consideraciones


Aplicaciones Lugares Google Chrome
jue 00:04
(105) How to setup Hado
How to setup Hadoop for
Historial
Namenode information
All Applications
localhost:8088/cluster
Logged in
All Applications
Cluster
About
Nodes
Node Labels
Applications
NEW
NEW_SAVING
SUBMITTED
ACCEPTED
RUNNING
FINISHED
FAILED
KILLED
Scheduler
Tools
Cluster Metrics
Apps Submitted
Apps Pending
Apps Running
Apps Completed
Containers Running
Memory Used
Memory Total
Memory Reserved
VCores Used
VCores Total
VCores Reserved
Active Nodes
Decommissioned Nodes
Lost Nodes
Unhealthy Nodes
0 0 0 0 0 0 B 8 GB 0 B 0 8 0 1 0 0 0 0
Scheduler Metrics
Scheduler Type
Scheduling Resource Type
Minimum Allocation
Maximum Allocation
Capacity Scheduler
[MEMORY]
<memory:1024, vCores:1>
<memory:8192, vCores:32>
Show 20 entries
Search:
ID
User
Name
Application Type
Queue
StartTime
FinishTime
State
FinalStatus
Progress
Tracking UI
Blacklisted No
No data available in table
Showing 0 to 0 of 0 entries
First
Previous
Next
jdk-8u191-l....tar.gz
Mostrar todo
All Applications - Google Chrome
hadoop@localhost:~/hadoop/sbin
[jdk1.8.0_191]
1 / 4


Aplicaciones Lugares Google Chrome
jue 00:04
(105) How to setup Hado
How to setup Hadoop for
Historial
Namenode information
All Applications
localhost:50070/dfshealth.html#tab-overview
Hadoop
Overview
Datanodes
Datanode Volume Failures
Snapshot
Startup Progress
Utilities
Datanode Information
In operation
Node
Last contact
Admin State
Capacity
Used
Non DFS Used
Remaining
Blocks
Block pool used
Failed Volumes
Version
localhost:50010 (127.0.0.1:50010) 0 In Service 920.56 GB 8 KB 6.78 GB 913.78 GB 0 8 KB (0%) 0 2.7.3
Decomissioning
Node
Last contact
Under replicated blocks
Blocks with no live replicas
Under Replicated Blocks
In files under construction
jdk-8u191-l....tar.gz
Mostrar todo
Namenode information - Google Ch...
[hadoop@localhost:~/hadoop/sbin]
[jdk1.8.0_191]
Imágenes
1 / 4

```
YCSB Client 0.1
Command line: -db com.yahoo.ycsb.db.MongoDbClient -s -P workloads/workloade -t
[WORKLOAD], SCAN, 0.95%.
mongo connection created with localhost:27017
[OVERALL], RunTime(ms), 437259.0
[OVERALL], Throughput(ops/sec), 1600.8818572059122
[CLEANUP], Operations, 1
[CLEANUP], AverageLatency(us), 2723.0
[CLEANUP], MinLatency(us), 2723
[CLEANUP], MaxLatency(us), 2723
[CLEANUP], 95thPercentileLatency(us), 2720
[CLEANUP], 99thPercentileLatency(us), 2720
[CLEANUP], 0us, 0
[CLEANUP], 10us, 0
[CLEANUP], 20us, 0
[CLEANUP], 30us, 0
[CLEANUP], 40us, 0
[CLEANUP], 50us, 0
[CLEANUP], 60us, 0
[CLEANUP], 70us, 0
[CLEANUP], 80us, 0
[CLEANUP], 90us, 0
[CLEANUP], 100us, 0
[CLEANUP], 110us, 0
[CLEANUP], 120us, 0
[CLEANUP], 130us, 0
[CLEANUP], 140us, 0
[CLEANUP], 150us, 0
[CLEANUP], 160us, 0
[CLEANUP], 170us, 0
[CLEANUP], 180us, 0
[CLEANUP], 190us, 0
[CLEANUP], 200us, 0
```

```
[OVERALL], RunTime(ms), 443430.0
[OVERALL], Throughput(ops/sec), 1578.6031617166182
[CLEANUP], Operations, 1
[CLEANUP], AverageLatency(us), 2464.0
[CLEANUP], MinLatency(us), 2464
[CLEANUP], MaxLatency(us), 2464
[CLEANUP], 95thPercentileLatency(us), 2460
[CLEANUP], 99thPercentileLatency(us), 2460
[CLEANUP], 0us, 0
[CLEANUP], 10us, 0
[CLEANUP], 20us, 0
[CLEANUP], 30us, 0
[CLEANUP], 40us, 0
[CLEANUP], 50us, 0
[CLEANUP], 60us, 0
[CLEANUP], 70us, 0
[CLEANUP], 80us, 0
[CLEANUP], 90us, 0
[CLEANUP], 100us, 0
[CLEANUP], 110us, 0
[CLEANUP], 120us, 0
[CLEANUP], 130us, 0
[CLEANUP], 140us, 0
[CLEANUP], 150us, 0
[CLEANUP], 160us, 0
[CLEANUP], 170us, 0
[CLEANUP], 180us, 0
[CLEANUP], 190us, 0
[CLEANUP], 200us, 0
[CLEANUP], 210us, 0
[CLEANUP], 220us, 0
[CLEANUP], 230us, 0
[CLEANUP], 240us, 0
[CLEANUP], 250us, 0
[CLEANUP], 260us, 0
```

```
[OVERALL], RunTime(ms), 431062.0
[OVERALL], Throughput(ops/sec), 1623.8963304582635
[CLEANUP], Operations, 1
[CLEANUP], AverageLatency(us), 2521.0
[CLEANUP], MinLatency(us), 2521
[CLEANUP], MaxLatency(us), 2521
[CLEANUP], 95thPercentileLatency(us), 2520
[CLEANUP], 99thPercentileLatency(us), 2520
[CLEANUP], 0us, 0
[CLEANUP], 10us, 0
[CLEANUP], 20us, 0
[CLEANUP], 30us, 0
[CLEANUP], 40us, 0
[CLEANUP], 50us, 0
[CLEANUP], 60us, 0
[CLEANUP], 70us, 0
[CLEANUP], 80us, 0
[CLEANUP], 90us, 0
[CLEANUP], 100us, 0
[CLEANUP], 110us, 0
[CLEANUP], 120us, 0
[CLEANUP], 130us, 0
[CLEANUP], 140us, 0
[CLEANUP], 150us, 0
[CLEANUP], 160us, 0
[CLEANUP], 170us, 0
[CLEANUP], 180us, 0
[CLEANUP], 190us, 0
[CLEANUP], 200us, 0
[CLEANUP], 210us, 0
[CLEANUP], 220us, 0
[CLEANUP], 230us, 0
[CLEANUP], 240us, 0
[CLEANUP], 250us, 0
[CLEANUP], 260us, 0
```

```
[CLEANUP], 9980us, 0
[CLEANUP], 9990us, 0
[CLEANUP], >1000, 0
[SCAN], Operations, 700000
[SCAN], AverageLatency(us), 612.3198742857143
[SCAN], MinLatency(us), 175
[SCAN], MaxLatency(us), 72523
[SCAN], 95thPercentileLatency(us), 990
[SCAN], 99thPercentileLatency(us), 1290
[SCAN], Return=0, 700000
[SCAN], 0us, 0
[SCAN], 10us, 0
[SCAN], 20us, 0
[SCAN], 30us, 0
[SCAN], 40us, 0
[SCAN], 50us, 0
[SCAN], 60us, 0
[SCAN], 70us, 0
[SCAN], 80us, 0
[SCAN], 90us, 0
[SCAN], 100us, 0
[SCAN], 110us, 0
[SCAN], 120us, 0
[SCAN], 130us, 0
[SCAN], 140us, 0
[SCAN], 150us, 0
[SCAN], 160us, 0
[SCAN], 170us, 11
[SCAN], 180us, 488
[SCAN], 190us, 1438
[SCAN], 200us, 2801
[SCAN], 210us, 4071
[SCAN], 220us, 5262
[SCAN], 230us, 6149
[SCAN], 240us, 6846
```

```
[OVERALL], RunTime(ms), 430439.0
[OVERALL], Throughput(ops/sec), 1626.2466923303882
[CLEANUP], Operations, 1
[CLEANUP], AverageLatency(us), 2536.0
[CLEANUP], MinLatency(us), 2536
[CLEANUP], MaxLatency(us), 2536
[CLEANUP], 95thPercentileLatency(us), 2530
[CLEANUP], 99thPercentileLatency(us), 2530
[CLEANUP], 0us, 0
[CLEANUP], 10us, 0
[CLEANUP], 20us, 0
[CLEANUP], 30us, 0
[CLEANUP], 40us, 0
[CLEANUP], 50us, 0
[CLEANUP], 60us, 0
[CLEANUP], 70us, 0
[CLEANUP], 80us, 0
[CLEANUP], 90us, 0
[CLEANUP], 100us, 0
[CLEANUP], 110us, 0
[CLEANUP], 120us, 0
[CLEANUP], 130us, 0
[CLEANUP], 140us, 0
[CLEANUP], 150us, 0
[CLEANUP], 160us, 0
[CLEANUP], 170us, 0
[CLEANUP], 180us, 0
[CLEANUP], 190us, 0
[CLEANUP], 200us, 0
[CLEANUP], 210us, 0
[CLEANUP], 220us, 0
[CLEANUP], 230us, 0
[CLEANUP], 240us, 0
[CLEANUP], 250us, 0
[CLEANUP], 260us, 0
```

```
[OVERALL], RunTime(ms), 431696.0
[OVERALL], Throughput(ops/sec), 1621.5114339720544
[CLEANUP], Operations, 1
[CLEANUP], AverageLatency(us), 2805.0
[CLEANUP], MinLatency(us), 2805
[CLEANUP], MaxLatency(us), 2805
[CLEANUP], 95thPercentileLatency(us), 2800
[CLEANUP], 99thPercentileLatency(us), 2800
[CLEANUP], 0us, 0
[CLEANUP], 10us, 0
[CLEANUP], 20us, 0
[CLEANUP], 30us, 0
[CLEANUP], 40us, 0
[CLEANUP], 50us, 0
[CLEANUP], 60us, 0
[CLEANUP], 70us, 0
[CLEANUP], 80us, 0
[CLEANUP], 90us, 0
[CLEANUP], 100us, 0
[CLEANUP], 110us, 0
[CLEANUP], 120us, 0
[CLEANUP], 130us, 0
[CLEANUP], 140us, 0
[CLEANUP], 150us, 0
[CLEANUP], 160us, 0
[CLEANUP], 170us, 0
[CLEANUP], 180us, 0
[CLEANUP], 190us, 0
[CLEANUP], 200us, 0
[CLEANUP], 210us, 0
[CLEANUP], 220us, 0
[CLEANUP], 230us, 0
[CLEANUP], 240us, 0
[CLEANUP], 250us, 0
[CLEANUP], 260us, 0
```

```
 90,0 sec: 362853 operations; 4097,9 current ops/sec; [INSERT AverageLatency(us)=239,41]
 100,0 sec: 407112 operations; 4425,9 current ops/sec; [INSERT AverageLatency(us)=222,8]
 110,0 sec: 448309 operations; 4119,7 current ops/sec; [INSERT AverageLatency(us)=238,5]
 120,0 sec: 487282 operations; 3888,91 current ops/sec; [INSERT AverageLatency(us)=253,3]
 130,0 sec: 529363 operations; 4216,1 current ops/sec; [INSERT AverageLatency(us)=233,23]
 140,0 sec: 568964 operations; 3960,1 current ops/sec; [INSERT AverageLatency(us)=247,7]
 150,0 sec: 612470 operations; 4350,16 current ops/sec; [INSERT AverageLatency(us)=226,73]
 160,0 sec: 656994 operations; 4452,4 current ops/sec; [INSERT AverageLatency(us)=220,85]
 170,0 sec: 698303 operations; 4130,9 current ops/sec; [INSERT AverageLatency(us)=238,12]
nov 15, 2018 4:29:58 PM com.mongodb.diagnostics.logging.JULLogger log
INFORMACIÓN: Closed connection [connectionId{localValue:2, serverValue:32}] to localhost:27017 because the pool has been closed.
 170,5 sec: 700000 operations; 4021,33 current ops/sec; [CLEANUP AverageLatency(us)=3003] [INSERT AverageLatency(us)=242,81]
[OVERALL], RunTime(ms), 170451.0
[OVERALL], Throughput(ops/sec), 4106.752087110079
[CLEANUP], Operations, 1
[CLEANUP], AverageLatency(us), 3003.0
[CLEANUP], MinLatency(us), 3003
[CLEANUP], MaxLatency(us), 3003
[CLEANUP], 95thPercentileLatency(us), 3000
[CLEANUP], 99thPercentileLatency(us), 3000
[CLEANUP], 0us, 0
[CLEANUP], 10us, 0
[CLEANUP], 20us, 0
[CLEANUP], 30us, 0
[CLEANUP], 40us, 0
[CLEANUP], 50us, 0
[CLEANUP], 60us, 0
[CLEANUP], 70us, 0
[CLEANUP], 80us, 0
[CLEANUP], 90us, 0
[CLEANUP], 100us, 0
[CLEANUP], 110us, 0
[CLEANUP], 120us, 0
[CLEANUP], 130us, 0
[CLEANUP], 140us, 0
```

```
[OVERALL], Throughput(ops/sec), 3396.4259894516713
[CLEANUP], Operations, 1
[CLEANUP], AverageLatency(us), 3031.0
[CLEANUP], MinLatency(us), 3031
[CLEANUP], MaxLatency(us), 3031
[CLEANUP], 95thPercentileLatency(us), 3030
[CLEANUP], 99thPercentileLatency(us), 3030
[CLEANUP], 0us, 0
[CLEANUP], 10us, 0
[CLEANUP], 20us, 0
[CLEANUP], 30us, 0
[CLEANUP], 40us, 0
[CLEANUP], 50us, 0
[CLEANUP], 60us, 0
[CLEANUP], 70us, 0
[CLEANUP], 80us, 0
[CLEANUP], 90us, 0
[CLEANUP], 100us, 0
[CLEANUP], 110us, 0
[CLEANUP], 120us, 0
[CLEANUP], 130us, 0
[CLEANUP], 140us, 0
[CLEANUP], 150us, 0
[CLEANUP], 160us, 0
[CLEANUP], 170us, 0
[CLEANUP], 180us, 0
[CLEANUP], 190us, 0
[CLEANUP], 200us, 0
[CLEANUP], 210us, 0
[CLEANUP], 220us, 0
[CLEANUP], 230us, 0
[CLEANUP], 240us, 0
[CLEANUP], 250us, 0
[CLEANUP], 260us, 0
[CLEANUP], 270us, 0
[CLEANUP], 280us, 0
```

```
INFORMACIÓN: Closed connection [connectionId{localVal
 165,4 sec: 700000 operations; 4251,86 current ops/se
[OVERALL], RunTime(ms), 165394.0
[OVERALL], Throughput(ops/sec), 4232.317980096013
[CLEANUP], Operations, 1
[CLEANUP], AverageLatency(us), 3296.0
[CLEANUP], MinLatency(us), 3296
[CLEANUP], MaxLatency(us), 3296
[CLEANUP], 95thPercentileLatency(us), 3290
[CLEANUP], 99thPercentileLatency(us), 3290
[CLEANUP], 0us, 0
[CLEANUP], 10us, 0
[CLEANUP], 20us, 0
[CLEANUP], 30us, 0
[CLEANUP], 40us, 0
[CLEANUP], 50us, 0
[CLEANUP], 60us, 0
[CLEANUP], 70us, 0
[CLEANUP], 80us, 0
[CLEANUP], 90us, 0
[CLEANUP], 100us, 0
[CLEANUP], 110us, 0
[CLEANUP], 120us, 0
[CLEANUP], 130us, 0
[CLEANUP], 140us, 0
[CLEANUP], 150us, 0
[CLEANUP], 160us, 0
[CLEANUP], 170us, 0
[CLEANUP], 180us, 0
[CLEANUP], 190us, 0
[CLEANUP], 200us, 0
[CLEANUP], 210us, 0
[CLEANUP], 220us, 0
[CLEANUP], 230us, 0
[CLEANUP], 240us, 0
```

```
INFORMACIÓN: Closed connection [connectionId{local
 164,9 sec: 700000 operations; 4357,96 current ops
[OVERALL], RunTime(ms), 164941.0
[OVERALL], Throughput(ops/sec), 4243.941773118873
[CLEANUP], Operations, 1
[CLEANUP], AverageLatency(us), 2988.0
[CLEANUP], MinLatency(us), 2988
[CLEANUP], MaxLatency(us), 2988
[CLEANUP], 95thPercentileLatency(us), 2980
[CLEANUP], 99thPercentileLatency(us), 2980
[CLEANUP], 0us, 0
[CLEANUP], 10us, 0
[CLEANUP], 20us, 0
[CLEANUP], 30us, 0
[CLEANUP], 40us, 0
[CLEANUP], 50us, 0
[CLEANUP], 60us, 0
[CLEANUP], 70us, 0
[CLEANUP], 80us, 0
[CLEANUP], 90us, 0
[CLEANUP], 100us, 0
[CLEANUP], 110us, 0
[CLEANUP], 120us, 0
[CLEANUP], 130us, 0
[CLEANUP], 140us, 0
[CLEANUP], 150us, 0
[CLEANUP], 160us, 0
[CLEANUP], 170us, 0
[CLEANUP], 180us, 0
[CLEANUP], 190us, 0
[CLEANUP], 200us, 0
[CLEANUP], 210us, 0
[CLEANUP], 220us, 0
[CLEANUP], 230us, 0
[CLEANUP], 240us, 0
```

```
[OVERALL], Throughput(ops/sec), 4203.774989940967
[CLEANUP], Operations, 1
[CLEANUP], AverageLatency(us), 3114.0
[CLEANUP], MinLatency(us), 3114
[CLEANUP], MaxLatency(us), 3114
[CLEANUP], 95thPercentileLatency(us), 3110
[CLEANUP], 99thPercentileLatency(us), 3110
[CLEANUP], 0us, 0
[CLEANUP], 10us, 0
[CLEANUP], 20us, 0
[CLEANUP], 30us, 0
[CLEANUP], 40us, 0
[CLEANUP], 50us, 0
[CLEANUP], 60us, 0
[CLEANUP], 70us, 0
[CLEANUP], 80us, 0
[CLEANUP], 90us, 0
[CLEANUP], 100us, 0
[CLEANUP], 110us, 0
[CLEANUP], 120us, 0
[CLEANUP], 130us, 0
[CLEANUP], 140us, 0
[CLEANUP], 150us, 0
[CLEANUP], 160us, 0
[CLEANUP], 170us, 0
[CLEANUP], 180us, 0
[CLEANUP], 190us, 0
[CLEANUP], 200us, 0
[CLEANUP], 210us, 0
[CLEANUP], 220us, 0
[CLEANUP], 230us, 0
[CLEANUP], 240us, 0
[CLEANUP], 250us, 0
[CLEANUP], 260us, 0
[CLEANUP], 270us, 0
```

```
[OVERALL], RunTime(ms), 166558.0
[OVERALL], Throughput(ops/sec), 4202.740186601664
[CLEANUP], Operations, 1
[CLEANUP], AverageLatency(us), 3674.0
[CLEANUP], MinLatency(us), 3674
[CLEANUP], MaxLatency(us), 3674
[CLEANUP], 95thPercentileLatency(us), 3670
[CLEANUP], 99thPercentileLatency(us), 3670
[CLEANUP], 0us, 0
[CLEANUP], 10us, 0
[CLEANUP], 20us, 0
[CLEANUP], 30us, 0
[CLEANUP], 40us, 0
[CLEANUP], 50us, 0
[CLEANUP], 60us, 0
[CLEANUP], 70us, 0
[CLEANUP], 80us, 0
[CLEANUP], 90us, 0
[CLEANUP], 100us, 0
[CLEANUP], 110us, 0
[CLEANUP], 120us, 0
[CLEANUP], 130us, 0
[CLEANUP], 140us, 0
[CLEANUP], 150us, 0
[CLEANUP], 160us, 0
[CLEANUP], 170us, 0
[CLEANUP], 180us, 0
[CLEANUP], 190us, 0
[CLEANUP], 200us, 0
[CLEANUP], 210us, 0
[CLEANUP], 220us, 0
[CLEANUP], 230us, 0
[CLEANUP], 240us, 0
[CLEANUP], 250us, 0
[CLEANUP], 260us, 0
```

```
YCSB Client 0.1
Command line: -db com.yahoo.ycsb.db.MongoDbClient -s -P workloads/workloada -t
[WORKLOAD], UPDATE, 0.5%.
mongo connection created with localhost:27017
[OVERALL], RunTime(ms), 154652.0
[OVERALL], Throughput(ops/sec), 4526.291286242661
[UPDATE], Operations, 700000
[UPDATE], AverageLatency(us), 216.45875714285714
[UPDATE], MinLatency(us), 146
[UPDATE], MaxLatency(us), 487721
[UPDATE], 95thPercentileLatency(us), 260
[UPDATE], 99thPercentileLatency(us), 420
[UPDATE] 700000
[UPDATE], 0us, 0
[UPDATE], 10us, 0
[UPDATE], 20us, 0
[UPDATE], 30us, 0
[UPDATE], 40us, 0
[UPDATE], 50us, 0
[UPDATE], 60us, 0
[UPDATE], 70us, 0
[UPDATE], 80us, 0
[UPDATE], 90us, 0
[UPDATE], 100us, 0
[UPDATE], 110us, 0
[UPDATE], 120us, 0
[UPDATE], 130us, 0
[UPDATE], 140us, 794
[UPDATE], 150us, 23391
[UPDATE], 160us, 72390
[UPDATE], 170us, 174548
[UPDATE], 180us, 118895
[UPDATE], 190us, 89176
[UPDATE], 200us, 62345
[UPDATE], 210us, 43229
[UPDATE], 220us, 31841
```

```
YCSB Client 0.1
Command line: -db com.yahoo.ycsb.db.MongoDbClient -s -P workloads/workloada -t
[WORKLOAD], UPDATE, 0.5%.
mongo connection created with localhost:27017
[OVERALL], RunTime(ms), 150265.0
[OVERALL], Throughput(ops/sec), 4658.43676172096
[UPDATE], Operations, 700000
[UPDATE], AverageLatency(us), 210.35433857142857
[UPDATE], MinLatency(us), 146
[UPDATE], MaxLatency(us), 592818
[UPDATE], 95thPercentileLatency(us), 250
[UPDATE], 99thPercentileLatency(us), 380
[UPDATE], Return=0, 700000
[UPDATE], 0us, 0
[UPDATE], 10us, 0
[UPDATE], 20us, 0
[UPDATE], 30us, 0
[UPDATE], 40us, 0
[UPDATE], 50us, 0
[UPDATE], 60us, 0
[UPDATE], 70us, 0
[UPDATE], 80us, 0
[UPDATE], 90us, 0
[UPDATE], 100us, 0
[UPDATE], 110us, 0
[UPDATE], 120us, 0
[UPDATE], 130us, 0
[UPDATE], 140us, 835
[UPDATE], 150us, 25548
[UPDATE], 160us, 103113
[UPDATE], 170us, 193609
[UPDATE], 180us, 112334
[UPDATE], 190us, 88465
[UPDATE], 200us, 55008
[UPDATE], 210us, 35152
[UPDATE], 220us, 23560
```

```
YCSB Client 0.1
Command line: -db com.yahoo.ycsb.db.MongoDbClient -s -P workloads/workloada -t
[WORKLOAD], UPDATE, 0.5%.
mongo connection created with localhost:27017
[OVERALL], RunTime(ms), 153372.0
[OVERALL], Throughput(ops/sec), 4564.066452807553
[UPDATE], Operations, 700000
[UPDATE], AverageLatency(us), 214.48632857142857
[UPDATE], MinLatency(us), 145
[UPDATE], MaxLatency(us), 446145
[UPDATE], 95thPercentileLatency(us), 270
[UPDATE], 99thPercentileLatency(us), 480
[UPDATE], Return=0, 700000
[UPDATE], 0us, 0
[UPDATE], 10us, 0
[UPDATE], 20us, 0
[UPDATE], 30us, 0
[UPDATE], 40us, 0
[UPDATE], 50us, 0
[UPDATE], 60us, 0
[UPDATE], 70us, 0
[UPDATE], 80us, 0
[UPDATE], 90us, 0
[UPDATE], 100us, 0
[UPDATE], 110us, 0
[UPDATE], 120us, 0
[UPDATE], 130us, 0
[UPDATE], 140us, 773
[UPDATE], 150us, 33399
[UPDATE], 160us, 64282
[UPDATE], 170us, 174298
[UPDATE], 180us, 111857
[UPDATE], 190us, 82826
[UPDATE], 200us, 60776
[UPDATE], 210us, 43192
[UPDATE], 220us, 34534
```

```
YCSB Client 0.1
Command line: -db com.yahoo.ycsb.db.MongoDbClient -s -P workloads/workloada -t
[WORKLOAD], UPDATE, 0.5%.
mongo connection created with localhost:27017
[OVERALL], RunTime(ms), 144949.0
[OVERALL], Throughput(ops/sec), 4829.284782923649
[UPDATE], Operations, 700000
[UPDATE], AverageLatency(us), 202.72824857142857
[UPDATE], MinLatency(us), 145
[UPDATE], MaxLatency(us), 408440
[UPDATE], 95thPercentileLatency(us), 260
[UPDATE], 99thPercentileLatency(us), 410
[UPDATE], Return=0, 700000
[UPDATE], 0us, 0
[UPDATE], 10us, 0
[UPDATE], 20us, 0
[UPDATE], 30us, 0
[UPDATE], 40us, 0
[UPDATE], 50us, 0
[UPDATE], 60us, 0
[UPDATE], 70us, 0
[UPDATE], 80us, 0
[UPDATE], 90us, 0
[UPDATE], 100us, 0
[UPDATE], 110us, 0
[UPDATE], 120us, 0
[UPDATE], 130us, 0
[UPDATE], 140us, 3259
[UPDATE], 150us, 29843
[UPDATE], 160us, 86989
[UPDATE], 170us, 174865
[UPDATE], 180us, 104068
[UPDATE], 190us, 93665
[UPDATE], 200us, 56332
[UPDATE], 210us, 41562
[UPDATE], 220us, 29726
```

```
YCSB Client 0.1
Command line: -db com.yahoo.ycsb.db.MongoDbClient -s -P workloads/workloada -t
[WORKLOAD], UPDATE, 0.5%.
mongo connection created with localhost:27017
[OVERALL], RunTime(ms), 133818.0
[OVERALL], Throughput(ops/sec), 5230.985368186642
[UPDATE], Operations, 700000
[UPDATE], AverageLatency(us), 186.9748157142857
[UPDATE], MinLatency(us), 145
[UPDATE], MaxLatency(us), 264453
[UPDATE], 95thPercentileLatency(us), 230
[UPDATE], 99thPercentileLatency(us), 300
[UPDATE], Return= 700000
[UPDATE], 0us, 0
[UPDATE], 10us, 0
[UPDATE], 20us, 0
[UPDATE], 30us, 0
[UPDATE], 40us, 0
[UPDATE], 50us, 0
[UPDATE], 60us, 0
[UPDATE], 70us, 0
[UPDATE], 80us, 0
[UPDATE], 90us, 0
[UPDATE], 100us, 0
[UPDATE], 110us, 0
[UPDATE], 120us, 0
[UPDATE], 130us, 0
[UPDATE], 140us, 9621
[UPDATE], 150us, 49468
[UPDATE], 160us, 175178
[UPDATE], 170us, 186881
[UPDATE], 180us, 84814
[UPDATE], 190us, 82470
[UPDATE], 200us, 40570
[UPDATE], 210us, 19157
[UPDATE], 220us, 11934
```

```
YCSB Client 0.1
Command line: -db com.yahoo.ycsb.db.MongoDbClient -s -P workloads/workloada -t
[WORKLOAD], UPDATE, 0.5%.
mongo connection created with localhost:27017
[OVERALL], RunTime(ms), 135236.0
[OVERALL], Throughput(ops/sec), 5176.136531692745
[UPDATE], Operations, 700000
[UPDATE], AverageLatency(us), 188.81322
[UPDATE], MinLatency(us), 144
[UPDATE], MaxLatency(us), 1135801
[UPDATE], 95thPercentileLatency(us), 230
[UPDATE], 99thPercentileLatency(us), 300
[UPDATE], Return=700000
[UPDATE], 0us, 0
[UPDATE], 10us, 0
[UPDATE], 20us, 0
[UPDATE], 30us, 0
[UPDATE], 40us, 0
[UPDATE], 50us, 0
[UPDATE], 60us, 0
[UPDATE], 70us, 0
[UPDATE], 80us, 0
[UPDATE], 90us, 0
[UPDATE], 100us, 0
[UPDATE], 110us, 0
[UPDATE], 120us, 0
[UPDATE], 130us, 0
[UPDATE], 140us, 12467
[UPDATE], 150us, 50827
[UPDATE], 160us, 174925
[UPDATE], 170us, 194041
[UPDATE], 180us, 85044
[UPDATE], 190us, 83235
[UPDATE], 200us, 36063
[UPDATE], 210us, 16832
[UPDATE], 220us, 10168
```

```
YCSB Client 0.1
Command line: -db com.yahoo.ycsb.db.MongoDbClient -s -P workloads/workloada -t
[WORKLOAD], UPDATE, 0.5%.
mongo connection created with localhost:27017
[OVERALL], RunTime(ms), 152085.0
[OVERALL], Throughput(ops/sec), 4602.68928559687
[UPDATE], Operations, 700000
[UPDATE], AverageLatency(us), 212.74674142857143
[UPDATE], MinLatency(us), 145
[UPDATE], MaxLatency(us), 526657
[UPDATE], 95thPercentileLatency(us), 260
[UPDATE], 99thPercentileLatency(us), 430
[UPDATE], Return=700000
[UPDATE], 0us, 0
[UPDATE], 10us, 0
[UPDATE], 20us, 0
[UPDATE], 30us, 0
[UPDATE], 40us, 0
[UPDATE], 50us, 0
[UPDATE], 60us, 0
[UPDATE], 70us, 0
[UPDATE], 80us, 0
[UPDATE], 90us, 0
[UPDATE], 100us, 0
[UPDATE], 110us, 0
[UPDATE], 120us, 0
[UPDATE], 130us, 0
[UPDATE], 140us, 829
[UPDATE], 150us, 34153
[UPDATE], 160us, 69084
[UPDATE], 170us, 200802
[UPDATE], 180us, 115458
[UPDATE], 190us, 79311
[UPDATE], 200us, 53429
[UPDATE], 210us, 38827
[UPDATE], 220us, 28913
```

```
YCSB Client 0.1
Command line: -db com.yahoo.ycsb.db.MongoDbClient -s -P workloads/workloada -t
[WORKLOAD], UPDATE, 0.5%.
mongo connection created with localhost:27017
[OVERALL], RunTime(ms), 167183.0
[OVERALL], Throughput(ops/sec), 4187.0285854423
[UPDATE], Operations, 700000
[UPDATE], AverageLatency(us), 233.70578714285713
[UPDATE], MinLatency(us), 147
[UPDATE], MaxLatency(us), 426128
[UPDATE], 95thPercentileLatency(us), 330
[UPDATE], 99thPercentileLatency(us), 710
[UPDATE], Return= 700000
[UPDATE], 0us, 0
[UPDATE], 10us, 0
[UPDATE], 20us, 0
[UPDATE], 30us, 0
[UPDATE], 40us, 0
[UPDATE], 50us, 0
[UPDATE], 60us, 0
[UPDATE], 70us, 0
[UPDATE], 80us, 0
[UPDATE], 90us, 0
[UPDATE], 100us, 0
[UPDATE], 110us, 0
[UPDATE], 120us, 0
[UPDATE], 130us, 0
[UPDATE], 140us, 25
[UPDATE], 150us, 7441
[UPDATE], 160us, 26679
[UPDATE], 170us, 101647
[UPDATE], 180us, 119451
[UPDATE], 190us, 85795
[UPDATE], 200us, 75397
[UPDATE], 210us, 57491
[UPDATE], 220us, 54094
```

```
YCSB Client 0.1
Command line: -db com.yahoo.ycsb.db.MongoDbClient -s -P workloads/workloada -t
[WORKLOAD], UPDATE, 0.5%.
mongo connection created with localhost:27017
[OVERALL], RunTime(ms), 145656.0
[OVERALL], Throughput(ops/sec), 4805.843906189927
CLEANUP], >1000, 0
[UPDATE], Operations, 700000
[UPDATE], AverageLatency(us), 203.69679714285715
[UPDATE], MinLatency(us), 146
[UPDATE], MaxLatency(us), 495717
[UPDATE], 95thPercentileLatency(us), 240
[UPDATE], 99thPercentileLatency(us), 360
[UPDATE], Return=700000
[UPDATE], 0us, 0
[UPDATE], 10us, 0
[UPDATE], 20us, 0
[UPDATE], 30us, 0
[UPDATE], 40us, 0
[UPDATE], 50us, 0
[UPDATE], 60us, 0
[UPDATE], 70us, 0
[UPDATE], 80us, 0
[UPDATE], 90us, 0
[UPDATE], 100us, 0
[UPDATE], 110us, 0
[UPDATE], 120us, 0
[UPDATE], 130us, 0
[UPDATE], 140us, 2019
[UPDATE], 150us, 40018
[UPDATE], 160us, 92604
[UPDATE], 170us, 196674
[UPDATE], 180us, 116027
[UPDATE], 190us, 80237
[UPDATE], 200us, 53613
[UPDATE], 210us, 33655
```

```
YCSB Client 0.1
Command line: -db com.yahoo.ycsb.db.MongoDbClient -s -P workloads/workloada -t
[WORKLOAD], UPDATE, 0.5%.
mongo connection created with localhost:27017
[OVERALL], RunTime(ms), 152314.0
[OVERALL], Throughput(ops/sec), 4595.769266121302
[UPDATE], Operations, 700000
[UPDATE], AverageLatency(us), 212.91871
[UPDATE], MinLatency(us), 147
[UPDATE], MaxLatency(us), 313472
[UPDATE], 95thPercentileLatency(us), 270
[UPDATE], 99thPercentileLatency(us), 410
[UPDATE], Return=700000
[UPDATE], 0us, 0
[UPDATE], 10us, 0
[UPDATE], 20us, 0
[UPDATE], 30us, 0
[UPDATE], 40us, 0
[UPDATE], 50us, 0
[UPDATE], 60us, 0
[UPDATE], 70us, 0
[UPDATE], 80us, 0
[UPDATE], 90us, 0
[UPDATE], 100us, 0
[UPDATE], 110us, 0
[UPDATE], 120us, 0
[UPDATE], 130us, 0
[UPDATE], 140us, 93
[UPDATE], 150us, 18484
[UPDATE], 160us, 41267
[UPDATE], 170us, 139812
[UPDATE], 180us, 133534
[UPDATE], 190us, 86052
[UPDATE], 200us, 75146
[UPDATE], 210us, 50805
[UPDATE], 220us, 46450
```

```
YCSB Client 0.1
Command line: -db com.yahoo.ycsb.db.MongoDbClient -s -P workloads/workloada -t
[WORKLOAD], READ, 0.5%.
mongo connection created with localhost:27017
[OVERALL], RunTime(ms), 133572.0
[OVERALL], Throughput(ops/sec), 5240.6192914682715
[READ], Operations, 700000
[READ], AverageLatency(us), 18.669288857142857
[READ], MinLatency(us), 139
[READ], MaxLatency(us), 64467
[READ], 95thPercentileLatency(us), 250
[READ], 99thPercentileLatency(us), 400
[READ], Return=0, 700000
[READ], 0us, 0
[READ], 10us, 0
[READ], 20us, 0
[READ], 30us, 0
[READ], 40us, 0
[READ], 50us, 0
[READ], 60us, 0
[READ], 70us, 0
[READ], 80us, 0
[READ], 90us, 0
[READ], 100us, 0
[READ], 110us, 0
[READ], 120us, 0
[READ], 130us, 2
[READ], 140us, 2100
[READ], 150us, 174826
[READ], 160us, 184951
[READ], 170us, 86411
[READ], 180us, 60523
[READ], 190us, 41029
[READ], 200us, 36857
[READ], 210us, 27722
[READ], 220us, 21626
```

```
YCSB Client 0.1
Command line: -db com.yahoo.ycsb.db.MongoDbClient -s -P workloads/workloada -t
[WORKLOAD], READ, 0.5%.
mongo connection created with localhost:27017
[OVERALL], RunTime(ms), 130726.0
[OVERALL], Throughput(ops/sec), 5354.711381056561
[READ], Operations, 700000
[READ], AverageLatency(us), 18.259859142857144
[READ], MinLatency(us), 136
[READ], MaxLatency(us), 66943
[READ], 95thPercentileLatency(us), 240
[READ], 99thPercentileLatency(us), 380
[READ], Return=700000
[READ], 0us, 0
[READ], 10us, 0
[READ], 20us, 0
[READ], 30us, 0
[READ], 40us, 0
[READ], 50us, 0
[READ], 60us, 0
[READ], 70us, 0
[READ], 80us, 0
[READ], 90us, 0
[READ], 100us, 0
[READ], 110us, 0
[READ], 120us, 0
[READ], 130us, 176
[READ], 140us, 7733
[READ], 150us, 225778
[READ], 160us, 169198
[READ], 170us, 79147
[READ], 180us, 50921
[READ], 190us, 33884
[READ], 200us, 32885
[READ], 210us, 24913
[READ], 220us, 19435
```

```
YCSB Client 0.1
Command line: -db com.yahoo.ycsb.db.MongoDbClient -s -P workloads/workloada -t
[WORKLOAD], READ, 0.5%.
mongo connection created with localhost:27017
[OVERALL], RunTime(ms), 118382.0
[OVERALL], Throughput(ops/sec), 5913.061107262928
[READ], Operations, 700000
[READ], AverageLatency(us), 16.502284
[READ], MinLatency(us), 135
[READ], MaxLatency(us), 63148
[READ], 95thPercentileLatency(us), 210
[READ], 99thPercentileLatency(us), 260
[READ], Return=0, 700000
[READ], 0us, 0
[READ], 10us, 0
[READ], 20us, 0
[READ], 30us, 0
[READ], 40us, 0
[READ], 50us, 0
[READ], 60us, 0
[READ], 70us, 0
[READ], 80us, 0
[READ], 90us, 0
[READ], 100us, 0
[READ], 110us, 0
[READ], 120us, 0
[READ], 130us, 10
[READ], 140us, 47251
[READ], 150us, 400232
[READ], 160us, 105013
[READ], 170us, 50773
[READ], 180us, 30274
[READ], 190us, 16438
[READ], 200us, 14542
[READ], 210us, 9627
[READ], 220us, 6986
```

```
YCSB Client 0.1
Command line: -db com.yahoo.ycsb.db.MongoDbClient -s -P workloads/workloada -t
[WORKLOAD], READ, 0.5%.
mongo connection created with localhost:27017
[OVERALL], RunTime(ms), 115607.0
[OVERALL], Throughput(ops/sec), 6054.996669751832
[READ], Operations, 700000
[READ], AverageLatency(us), 16.130455
[READ], MinLatency(us), 136
[READ], MaxLatency(us), 62399
[READ], 95thPercentileLatency(us), 200
[READ], 99thPercentileLatency(us), 250
[READ], Return=0, 700000
[READ], 0us, 0
[READ], 10us, 0
[READ], 20us, 0
[READ], 30us, 0
[READ], 40us, 0
[READ], 50us, 0
[READ], 60us, 0
[READ], 70us, 0
[READ], 80us, 0
[READ], 90us, 0
[READ], 100us, 0
[READ], 110us, 0
[READ], 120us, 0
[READ], 130us, 43
[READ], 140us, 141047
[READ], 150us, 362850
[READ], 160us, 82401
[READ], 170us, 45636
[READ], 180us, 20154
[READ], 190us, 12184
[READ], 200us, 10288
[READ], 210us, 6566
[READ], 220us, 4752
```

```
YCSB Client 0.1
Command line: -db com.yahoo.ycsb.db.MongoDbClient -s -P workloads/workloada -t
[WORKLOAD], READ, 0.5%.
mongo connection created with localhost:27017
[OVERALL], RunTime(ms), 115781.0
[OVERALL], Throughput(ops/sec), 6045.896995189193
[READ], Operations, 700000
[READ], AverageLatency(us), 16.147814571428572
[READ], MinLatency(us), 137
[READ], MaxLatency(us), 63440
[READ], 95thPercentileLatency(us), 200
[READ], 99thPercentileLatency(us), 250
[READ], Return=0, 700000
[READ], 0us, 0
[READ], 10us, 0
[READ], 20us, 0
[READ], 30us, 0
[READ], 40us, 0
[READ], 50us, 0
[READ], 60us, 0
[READ], 70us, 0
[READ], 80us, 0
[READ], 90us, 0
[READ], 100us, 0
[READ], 110us, 0
[READ], 120us, 0
[READ], 130us, 45
[READ], 140us, 145719
[READ], 150us, 356183
[READ], 160us, 82870
[READ], 170us, 45963
[READ], 180us, 20038
[READ], 190us, 12358
[READ], 200us, 10175
[READ], 210us, 6774
[READ], 220us, 4858
```

```
YCSB Client 0.1
Command line: -db com.yahoo.ycsb.db.MongoDbClient -s -P workloads/workloada -t
[WORKLOAD], READ, 0.5%.
mongo connection created with localhost:27017
[OVERALL], RunTime(ms), 115386.0
[OVERALL], Throughput(ops/sec), 6066.593867540256
[READ], Operations, 700000
[READ], AverageLatency(us), 16.089626
[READ], MinLatency(us), 137
[READ], MaxLatency(us), 62692
[READ], 95thPercentileLatency(us), 200
[READ], 99thPercentileLatency(us), 250
[READ], Return=700000
[READ], 0us, 0
[READ], 10us, 0
[READ], 20us, 0
[READ], 30us, 0
[READ], 40us, 0
[READ], 50us, 0
[READ], 60us, 0
[READ], 70us, 0
[READ], 80us, 0
[READ], 90us, 0
[READ], 100us, 0
[READ], 110us, 0
[READ], 120us, 0
[READ], 130us, 39
[READ], 140us, 167094
[READ], 150us, 344291
[READ], 160us, 78865
[READ], 170us, 43667
[READ], 180us, 18920
[READ], 190us, 11582
[READ], 200us, 10191
[READ], 210us, 6670
[READ], 220us, 4662
```

UCV UNIVERSIDAD CÉSAR VALLEJO

DIRECCIÓN DE INVESTIGACIÓN

CONSTANCIA DE VALIDACIÓN

Quien suscribe, Jaime Leandro Madrid Casariego,
con documento nacional de identidad N° 02773138, de profesión Ingeniero de Sistemas, con Grado de magister, ejerciendo actualmente como Docente Tiempo C. de la institución Universidad Cesar Vallejo;
hago constar, por medio de la presente, que he revisado con fines de validación el instrumento Guía de Observación para medir el rendimiento, para su aplicación en el trabajo de investigación titulado: Modelos de rendimiento de Gestores de Base de Datos NoSql

Luego, de haber realizado las observaciones pertinentes, puedo formular las siguientes apreciaciones según lista de cotejo:

N°	CRITERIOS	SI	NO
1	El instrumento recoge información que permite dar respuesta al problema de investigación	/	
2	El instrumento propuesto responde a los objetivos del estudio	/	
3	La estructura del instrumento es adecuada	/	
4	Los ítems del instrumento son claros y presentan coherencia	/	
5	Los ítems están correctamente secuenciados.	/	
6	La cantidad de ítems es adecuada para su aplicación	/	

Piura, 11 de julio de 2018.

Firma
DNI N° 02773132

UCV UNIVERSIDAD CÉSAR VALLEJO

DIRECCIÓN DE INVESTIGACIÓN

CONSTANCIA DE VALIDACIÓN

Quien suscribe, Rubén Alexander Mora Valenza,
con documento nacional de identidad N° 02897931, de
profesión Ingeniero Informático, con Grado de
Magister, ejerciendo actualmente como Ingeniero
de la institución UNIVERSIDAD CESAR VALLEJO;
hago constar, por medio de la presente, que he revisado con fines de validación el
instrumento GUIA DE OBSERVACION PARA MEDIR RENDIMIENTO
para su aplicación en el trabajo de investigación titulado:
MODELO DE RENDIMIENTO EN GESTORES DE
BASE DE DATOS NOSQL

Luego, de haber realizado las observaciones pertinentes, puedo formular las siguientes apreciaciones según lista de cotejo:

N°	CRITERIOS	SI	NO
1	El instrumento recoge información que permite dar respuesta al problema de investigación	✓	
2	El instrumento propuesto responde a los objetivos del estudio	✓	
3	La estructura del instrumento es adecuada	✓	
4	Los ítems del instrumento son claros y presentan coherencia	✓	
5	Los ítems están correctamente secuenciados.	✓	
6	La cantidad de ítems es adecuada para su aplicación	✓	

Piura, 11 de Julio de 2018

Firma

DNI N° 02897931

CIP: 141461

UCV UNIVERSIDAD CÉSAR VALLEJO

DIRECCIÓN DE INVESTIGACIÓN

CONSTANCIA DE VALIDACIÓN

Quien suscribe, Marlon Nelson Martinez Sernaque, con documento nacional de identidad N° 40415866, de profesión Ingemiero Informatico, con Grado de Magister, ejerciendo actualmente como DIRECTOR DE ESCUELA de la institución UNIVERSIDAD CESAR VALLEJO;

hago constar, por medio de la presente, que he revisado con fines de validación el instrumento GUIA DE OBSERVACION PARA MEDIR EL RENDIMIENTO, para su aplicación en el trabajo de investigación titulado: MODELOS DE RENDIMIENTO DE GESTORES DE BASE DE DATOS NOSQL

Luego, de haber realizado las observaciones pertinentes, puedo formular las siguientes apreciaciones según lista de cotejo:

N°	CRITERIOS	SI	NO
1	El instrumento recoge información que permite dar respuesta al problema de investigación	✓	
2	El instrumento propuesto responde a los objetivos del estudio	✓	
3	La estructura del instrumento es adecuada	✓	
4	Los ítems del instrumento son claros y presentan coherencia	✓	
5	Los ítems están correctamente secuenciados.	✓	
6	La cantidad de ítems es adecuada para su aplicación	✓	

Piura,..... de..... de 20....

Firma

DNI N°.....40415866.....

UCV UNIVERSIDAD CÉSAR VALLEJO

DIRECCIÓN DE INVESTIGACIÓN

CONSTANCIA DE VALIDACIÓN

Quien suscribe, Ruben Alexander More Valencia,
con documento nacional de identidad N° 02897931, de profesión Ingeniero Informatico, con Grado de Magister, ejerciendo actualmente como Ingeniero de la institución Universidad Cesar Vallejo;
hago constar, por medio de la presente, que he revisado con fines de validación el instrumento Guía de Observacion para medir la Latencia,
para su aplicación en el trabajo de investigación titulado: Modelos de rendimiento de Gestores de Base de Datos NoSQL

Luego, de haber realizado las observaciones pertinentes, puedo formular las siguientes apreciaciones según lista de cotejo:

N°	CRITERIOS	SI	NO
1	El instrumento recoge información que permite dar respuesta al problema de investigación	✓	
2	El instrumento propuesto responde a los objetivos del estudio	✓	
3	La estructura del instrumento es adecuada	✓	
4	Los ítems del instrumento son claros y presentan coherencia	✓	
5	Los ítems están correctamente secuenciados.	✓	
6	La cantidad de ítems es adecuada para su aplicación	✓	

Piura 11 de Julio de 2018

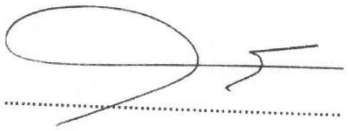
Firma
DNI N° 02897931
CIP: 141461

UCV UNIVERSIDAD CÉSAR VALLEJO	ACTA DE APROBACIÓN DE ORIGINALIDAD DE TESIS	Código : F06-PP-PR-02.02 Versión : 09 Fecha : 23-03-2018 Página : 1 de 1

Yo, **Rubén Alexander More Valencia** docente de la Facultad de Ingeniería y Escuela Profesional de Sistemas de la Universidad César Vallejo Piura, revisor (a) de la tesis titulada "**Modelos del Rendimiento En Gestores de Base de Datos NoSQL**", del estudiante **Purizaga Quiroga Javier Junnior,** constato que la investigación tiene un índice de similitud de **25 %** verificable en el reporte de originalidad del programa Turnitin.

El suscrito analizó dicho reporte y concluyó que cada una de las coincidencias detectadas no constituyen plagio. A mi leal saber y entender la tesis cumple con todas las normas para el uso de citas y referencias establecidas por la Universidad César Vallejo.

Piura 20 de marzo del 2020



Firma

More Valencia, Rubén Alexander

DNI: 02897931

Elaboró	Dirección de Investigación	Revisó	Responsable del SGC	Aprobó	Vicerrectorado de Investigación

UNIVERSIDAD CÉSAR VALLEJO

FACULTAD DE INGENIERÍA

ESCUELA ACADEMICO PROFESIONAL DE INGENIERÍA DE SISTEMAS

Modelos Del Rendimiento En Gestores De Base De Datos NoSQL

TESIS PARA OBTENER EL TÍTULO PROFESIONAL DE:

Ingeniero De Sistemas

AUTOR:

Br. Purizaga Quiroga Javier Junnior (ORCID: 0000-0003-2362-8370)

ASESOR:

Mg. More Valencia Rubén Alexander Ing. (ORCID: 0000-0002-7496-3792)

LÍNEA DE INVESTIGACIÓN:

Sistema De Información Y Telecomunicaciones

Piura- Perú

2018

UNIVERSIDAD CÉSAR VALLEJO

Centro de Recursos para el Aprendizaje y la Investigación (CRAI)
"César Acuña Peralta"

FORMULARIO DE AUTORIZACIÓN PARA LA PUBLICACIÓN ELECTRÓNICA DE LAS TESIS

1. **DATOS PERSONALES**
   Apellidos y Nombres: Purizaga Quiroga Javier Junnior
   D.N.I. : 71466640
   Domicilio : Mz. Qb Lote 22 Urb. Santa Margarita – 26 de Octubre
   Teléfono : Fijo : ..... Móvil :986735014
   E-mail : purizagaquirogaj@gmail.com

2. **IDENTIFICACIÓN DE LA TESIS**
   Modalidad:
   ☒ Tesis de Pregrado
   Facultad : Ingeniería
   Escuela : Ingeniería de Sistemas
   Carrera : Ingeniería de Sistemas
   Título : Ingeniero de Sistema

   ☐ Tesis de Post Grado
   ☐ Maestría ☐ Doctorado
   Grado : .....
   Mención : .....

3. **DATOS DE LA TESIS**
   Autor (es) Apellidos y Nombres:
   Purizaga Quiroga Javier Junnior

   Título de la tesis:
   "Modelos Del Rendimiento En Gestores De Base De Datos NoSQL".

   Año de publicación : 2020

4. **AUTORIZACIÓN DE PUBLICACIÓN DE LA TESIS EN VERSIÓN ELECTRÓNICA:**
   A través del presente documento,
   Si autorizo a publicar en texto completo mi tesis. ☒
   No autorizo a publicar en texto completo mi tesis. ☐

   Firma : ..... Fecha: 21/01/2020
   Purizaga Quiroga Javier Junnior

UNIVERSIDAD CÉSAR VALLEJO

AUTORIZACIÓN DE LA VERSIÓN FINAL DEL TRABAJO DE INVESTIGACIÓN

CONSTE POR EL PRESENTE EL VISTO BUENO QUE OTORGA EL ENCARGADO DE INVESTIGACIÓN DE

LA ESCUELA PROFESIONAL DE INGENIERIA DE SISTEMAS

A LA VERSIÓN FINAL DEL TRABAJO DE INVESTIGACIÓN QUE PRESENTA:

Br. PURIZAGA QUIROGA JAVIER JUNNIOR

INFORME TITULADO:

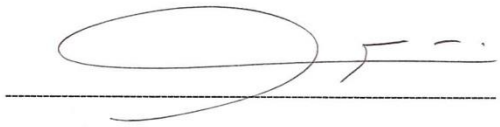
MODELOS DEL RENDIMIENTO EN GESTORES DE BASE DE DATOS NOSQL

PARA OBTENER EL GRADO O TÍTULO DE:

INGENIERO DE SISTEMAS

SUSTENTADO EN FECHA: 21/01/2020

NOTA O MENCIÓN: 16(Unanimidad)



MG.RUBÉN ALEXANDER MORE VALENCIA

COORDINADOR INVESTIGACIÓN EAP INGENIERÍA SISTEMAS UCV PIURA

FIRMA DEL ENCARGADO DE INVESTIGACIÓN