

**Maestría en ingeniería del software**

Trabajo Fin de Maestría

Gestores de bases de datos SQL vs. NoSQL, un análisis comparativo

AUTOR:

Sebastián Núñez

PROFESOR:

Vladimir Valdez

# DEDICATORIA

*Por definir*

# AGRADECIMIENTOS

*Los agradecimientos deben ir en cursiva, alineada a la derecha.*

*Times new Roman 12.*

**CONTENIDOS**

**EJEMPLO:**

[DEDICATORIA ii](#_heading=h.gjdgxs)

[AGRADECIMIENTOS iii](#_heading=h.30j0zll)

[RESUMEN v](#_heading=h.1fob9te)

[ABSTRACT vi](#_heading=h.3znysh7)

[CAPÍTULO 1. MARCO INTRODUCTORIO 1](#_heading=h.2et92p0)

[1.1. ANTECEDENTES 1](#_heading=h.tyjcwt)

[1.2. PROBLEMA DE INVESTIGACION 1](#_heading=h.3dy6vkm)

[1.3. JUSTIFICACIÓN. 1](#_heading=h.1t3h5sf)

[1.4. ESTABLECIMIENTO DE OBJETIVOS E HIPÓTESIS 1](#_heading=h.4d34og8)

[CAPÍTULO 2. MARCO TEÓRICO 2](#_heading=h.2s8eyo1)

[2.1. EJEMPLO DE PUNTO DE DESARROLLO 2](#_heading=h.35nkun2)

[2.2. EJEMPLO 2 DE PUNTO DE DESARROLLO 2](#_heading=h.1ksv4uv)

[CAPÍTULO 3. MARCO EMPÍRICO 3](#_heading=h.3rdcrjn)

[CAPÍTULO 4. CONCLUSIONES, DISCUSION DE RESULTADOS Y RECOMENDACIONES 4](#_heading=h.26in1rg)

[BIBLIOGRAFÍA 5](#_heading=h.lnxbz9)

[ANEXOS 6](#_heading=h.44sinio)

**ÍNDICE DE TABLAS Y FIGURAS**

[Ilustración 1: Logo de unade 2](#_heading=h.17dp8vu)

# Resumen

# Con el Internet, el manejo de información digital el crecimiento exponencial de la información y el aparecimiento de nuevas tecnologías digitales han hecho que surjan nuevos retos de escalabilidad, manejo de los datos y especificidad, para los cuales los gestores de bases de datos relacionales tienen cierta dificultad en su resolución de administración. El presente trabajo busca adentrarse dentro de este tema para explorar bajo una fase inicial de investigación el rendimiento comparativo entre consultas de este tipo de gestores con los no relacionales, es decir un análisis comparativo entre gestores relacionales vs. no relacionales, ya que estos últimos han ido ganando terreno en los recientes años, así se toma en cuenta dentro de esta investigación con lo anteriormente mencionado la cantidad de información que pueden manejar los gestores de bases de datos en una suerte de laboratorio con una muestra de información considerable y aplicando tres de las consultas básicas que se requieren en el manejo de información digital.Abstract

Actually, the exponencial growing of Internet, the digital information managment and new digital technology emergency have became in new challenges about scalability management and specificity in data, about this relacional data manager have kind of dificultry to resolve. This investigation tries to inquire this topic for exploring in an inicial fase a comparative performance between querys in relacional and no relacional data managers. Specially, when the no relacional managers have been earning the attention of informatic community.

## Marco introductorio

### Antecedentes

La expansión a nivel mundial de la Internet ha requerido que nuevas medidas de almacenamiento y gestión de la información digital se tomen en cuenta ya que encarnan una dimensión masiva jamás vista (A.B.M.-Moniruzzaman & Akhter-Hossain, 2013, 2) especialmente cuando especificidades en el manejo de la información han incluido la inserción de imágenes, videos, y metadatos que complejizan las categorías que se manejan en una base de datos tradicional (Morales-Morales & Durán-Cazar, 2019, p.50). Esto ha hecho que modelos antiguos de gestión de la información (como sucede con el modelo relacional) se vuelvan obsoletos en cierto punto, ya que se requieren nuevos sistemas de gestión para afrontar las novedades tecnológicas que van de la mano con el desarrollo de lo que se conoce tecnología web 2.0.

En el caso ecuatoriano no es tendencia encontrar una actualización de los modelos de gestión de información bajo una perspectiva del big data tanto en el empleo como herramienta y tampoco en la presencia de profesionales dentro de la rama que la realicen (Padilla, 2019, p.84). En algunos sectores solo dentro de la industria farmacéutica, o financiera han representado cierta preocupación por el manejo complejo de información en lo que refiere a las bases de datos de tipo no relacional (Cueva & Melo, 2021, p.40). Y las investigaciones dentro de la academia aún se encuentran en una fase inicial que requiere su profundización (Vivanco-Vazquez, 2019; Alvear-Merizalde & Saltos Saá, 2020; Morales-Morales et al., 2019)

Cabe resaltar que cuando mencionamos a un sistema relacional, se hace referencia a un gestor de información que tiene como referencia el lenguaje SQL (Structured Query Language) para el establecimiento de sus transacciones y operaciones, las mismas que siguen el modelo ACID tanto de atomicidad (A), consistencia (C), isolación (I) y durabilidad (D) de los datos. A diferencia de una gestión NoSql (Not Only Sql) que si bien sigue la estructura de *querys* el sacrificio de la consistencia se da con la finalidad de obtener mayor *performance* y disponibilidad de la información (Martín et al., 2013, 168) razón por la cual se habla de una escalabilidad horizontal, a diferencia de una gestión relacional con escalabilidad vertical.

### Problema de investigación

Considerando esto y añadiendo un matiz histórico, el modelo relacional en la gestión de base de datos es el que ha predominado hasta la actualidad desde la década de los 80`s (RDBMS o *Relational Database Management System*) que ha desplazado a otras tipologías de gestión (Mercedes-Marqués, 2011, p.5) Entre los nuevos modelos se merece mencionar al sistema NoSql. Una tipología de gestión basada en *querys* que permite categorizar a mayor detalle la información dentro de lo que se conoce como colecciones, esto está permitiendo ganar popularidad en la comunidad de informáticos y programadores como una manera de a resolver problemas que conllevan un flujo masivo de información (Big Data) con categorías que cada vez más específicas y de mayor escalabilidad en los servidores.

### Justificación

Esta investigación busca realizar un aporte a este contexto actual para entender más el fenómeno informático de las bases de datos, tomando en cuenta el sistema de gestión de bases de datos predominante (relacional) en comparación con un emergente (no relacional). Y obtener información medible que permita saber de forma cuantitativa y a modo exploratorio, las ventajas y desventajas que ambos sistemas presentan, y como su comparación puede permitir que las instituciones y la comunidad de programadores e informáticos, puedan emplear un sistema adecuado a la estructuración de sus proyectos.

La comparación entre ambos gestores de bases de datos es necesaria ya que cuando se trata de un cúmulo masivo, preciso y a tiempo de generación de información en cada unidad de tiempo (big data), cada milisegundo cuenta dentro de la ejecución de cualquier plataforma, framework o app que se esté desarrollando sobre la forma en que se va a almacenar. Y dichas plataformas o gestores no relacionales y relacionales con su manejo, implica saber cómo el tiempo de carga de información que se realiza con determinadas consultas, las más usadas, puede afectar el desempeño dentro de lo que es el desarrollo de un software o plataforma específica, cuando se comparan.

### Establecimiento de hipótesis y objetivos

De acuerdo a la bibliografía revisada se puede plantear como hipótesis nula que los gestores relacionales cuentan con una mejor velocidad de ejecución de las consultas para operación, selección y carga en comparación con los no relacionales. Así para poder demostrar lo que se acaba de manifestar es necesario a nivel global analizar el comportamiento de los gestores de bases de datos relacionales con los no relacionales dentro de un comparativo de eficiencia de ejecución.

Hay que tomar en cuenta que dentro de las gestiones de base de datos las consultas básicas pueden ser definidas en tres grandes categorías. Operación (por ejemplo SUMA), selección (SET o WHERE) y carga. Para poder comparar el rendimiento de estas consultas entre gestores no relacionales y relacionales, la velocidad con la que se realizan dichas consultas es necesario estudiar y saber sus valores, sobre todo ya que cuando se trata de información que cumple característica Big Data, cuando de escalabilidad de servidores se trata.

## Marco teórico (primer nivel de punto de contenido)

Los gestores de base de datos no relacionales han sido una herramienta útil y desde el aparecimiento del Internet y el manejo de información digital han permitido a varias instituciones y personas adaptarse a los cambios tecnológico informáticos. Sin embargo, el crecimiento exponencial de la información y el aparecimiento de nuevas tecnologías han hecho que surjan nuevos retos de escalabilidad, manejo de los datos y especificidad, para los cuales los gestores de bases de datos relacionales pueden tener cierta dificultad en su resolución. El presente trabajo busca adentrarse dentro de este tema para explorar dentro de una fase inicial de investigación el rendimiento comparativo entre este tipo de gestores tomando en cuenta la cantidad de información que pueden manejar en una suerte de laboratorio con una muestra de datos considerable y relacionada con los gestores de base de datos no relacionales (*Not Only Sql*).

Cabe mencionar que los aportes teóricos sobre las bases de datos relacionales y no relacionales hasta el momento han sido diversos y dispersos, muy certeros en el esclarecimiento de conceptos y variables (A.B.M.-Moniruzzaman & Akhter-Hossain, 2013; Muños-Plascencia, 2020; Mercedes-Marqués, 2011; Binani & Upadhyyay, 2016; Tailor-Hemang et al., 2014), sin embargo en Ecuador el análisis de una comparativa entre ambos gestores, es decir de un estudio más específico sobre ambos tipos de gestores de bases de datos, se  encuentra en una fase inicial, ya que se perciben pocas investigaciones de estudiosos ecuatorianos sobre variantes de este tema en concreto (Vivanco-Vazquez, 2019; Alvear-Merizalde & Saltos Saá, 2020; Morales-Morales et al., 2019)

Para empezar este marco teórico, se merece mencionar que se afrontarán diferentes aportes contextuales, históricos y conceptuales de los autores arriba mencionados, para introducirnos en el análisis de los gestores de bases de datos relacionales y no relacionales para su entendimiento, y finalizar con un diálogo entre los aportes de investigadores ecuatorianos en un estudio del arte comparativo.

**1. Contexto**

La información, desde el aparecimiento de la internet y su expansión a nivel mundial, requiere nuevas medidas de almacenamiento y gestión que toman en cuenta la dimensión masiva (A.B.M.-Moniruzzaman & Akhter-Hossain, 2013, 2). Eso incluye otros tipos de datos que van más allá del simple texto, por ejemplo imágenes, videos, que utilizan las nuevas tecnologías de software o informáticas, así mismo una nueva escalabilidad de un procesamiento o capacidad de almacenamiento que se adecue a necesidades particulares e inmediatas.

Eso implica el planteamiento de un reto informático dentro de la gestión de la información donde existen diferentes tipos de datos,  de similitudes escasas, y con categorizaciones más complejas y específicas (Morales-Morales & Durán-Cazar, 2019, 50). Es decir, plantear el cuestionamiento de si el escalamiento de servidores sería la única solución para poder tratar dicho volumen y tipologías de información se ha convertido en la inquietud que tienen varios profesionales del software y la informática estos días.

Sin embargo desde la década de los 80 ́ s entre todos los modelos de gestión de información, el que ha predominado ha sido el de tipo relacional (RDBMS o Relational Database Management System) desplazando a otras tipologías de gestión (A.B.M.-Moniruzzaman & Akhter-Hossain, 2013, 1) . Pero con las actuales demandas, nuevos modelos se encuentran emergiendo ya que el RDBMS ha tenido que confrontar de forma problemática el incremento exponencial de la información y la compleja interrelación de sistemas que han sido acelerados con el fenómeno de la Internet (Web2.0, redes sociales, apps, etc.)

Haciendo algo de historia en 1970 nacieron las primeras tablas relacionales bajo un esquema conceptual, dentro del cual se merece resaltar el trabajo de Edgar Frank Cobb de los laboratorios de IBM, (Mercedes-Marqués, 2011,5) donde prácticamente estos sistemas siguiendo un modelo ACID empezaron a ofrecer características útiles en lo referente a un entorno transaccional de información. Dicho sistema adquirió un beneficio comercial posterior cuando a través de un lenguaje de gestión conocido como SQL (Structured Query Language) estandarizó la forma de administración de las diferentes unidades categóricas dentro de un modelo relacional que se ha mantenido hasta hoy en día.

El ACID es prácticamente una propiedad definida que tienen las transacciones en torno a la atomicidad, consistencia, isolación y durabilidad de la información y como se encuentran gestionadas en diferentes categorías dentro de una tabla (Binani et al., 2016, 43). Cada operación en torno a la información o data, correspondería a una transacción que bajo el modelo ACID el mínimo cambio se vería reflejado o no en la base de datos (atomicidad), la información se mantendría intacta (consistencia), cada transacción sería independiente una de otra (isolación) y en caso de una daño del sistema informático, los cambios hechos en una base de datos se mantendrían o almacenan (durabilidad) cuando hablamos de RDBMS.

Pero llevando este esquema al presente, se debe considerar que el mundo digital cuenta con nuevos aspectos que antes no existían relacionados al volumen, variedad, velocidad y naturaleza de la información (Morales-Morales & Durán-Cazar, 2019, 43). Algunos expertos han decidido denominar a esto el Big Data, y que debido a características mencionadas aquí se convierte en un reto que requiere más que un esquema bien orientado más allá del el SQL cuando existe la dimensión de una excesiva cantidad de información para su gestión.

El aparecimiento de la Big Data ha creado la necesidad de un sistema de gestión de información horizontal que sea escalable (Binani et al., 2016, 44), en otras palabras también para un sistema NoSQL. Dentro de las cuales se puede encontrar imágenes, XML, native objects, etc; que difieren en varias características de un RDBMS. En este sentido es que se denomina a un sistema de gestión NoSQL como un solucionador de almacenamiento en un tipo de transacción de información horizontalizada que han ido de la mano con innovaciones en el hardware computacional de hoy en día..

**1.1 Nuevas tecnologías en gestión de información**

Los nuevos sistemas de almacenamiento con procesadores multinúcleo están llevando a otro nivel el procesamiento de la información, no solo con lo referente a base de datos, sino también para la toma de decisiones tanto en el ámbito institucional. Existen dos tipologías de arquitectura de la información que siguen parámetros lógicos y físicos respectivamente  (Morales-Morales & Durán-Cazar, 2019, 44). Entre los aspectos lógicos se merece mencionar la forma en que se presenta la información (tablas, vistas, funciones, etc) Y en el físico, al que obedece en gran parte a su almacenamiento (extensiones, transacciones, páginas, etc)

La arquitectura multinúcleo fácilmente está adaptada a dichas arquitecturas para la gestión masiva de información en lo que refiere a base de datos. Un CPU en la actualidad soporta fácilmente los terabytes de información y sus relaciones tomando en cuenta la presente tecnología de 64 bits. (Plattner & Leukert, 2015, 14) Y en lo concerniente a la memoria que ocupan la lectura y escritura participan en niveles diferentes, incluso opuestos (Ávila, 2020) Es decir, las operaciones de lectura son independientes al tipo de almacenamiento en un dispositivo de guardado lento. Mientras que las operaciones de escritura se registran en un almacenamiento no volátil para garantizar la durabilidad de la información al mismo tiempo.

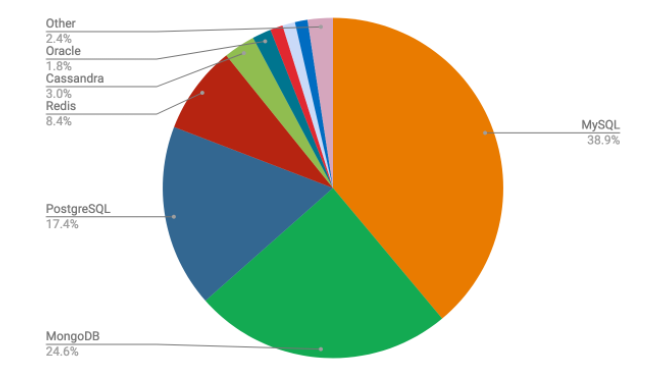
La principal diferencia entre un gestor de base de datos SQL y NoSQL es la forma en la que se encuentran estructurada la información. Un esquema ilustrativo plantea cinco consideraciones donde se resume la comparativa entre el uso de tablas (SQL) y colecciones en lo que respecta a sus unidades de almacenamiento y lectura (NoSql) (Ávila, 2020)

|  |  |
| --- | --- |
| **SQL** | **NoSQL** |
| Base de datos | Base de datos |
| Tablas | Colecciones |
| Columnas | Documentos |
| Index | Index |
| Join | $lookup |
| Foreing key | Reference |

La forma de organizar y consultar la información almacenada en una estructura relacional se puede entender de la siguiente manera, 1) tienen como fundamento el lenguaje SQL (Structured Query Language), siendo los querys cualquier tabla, columna, registro o relación, 2) la manera de usar este lenguaje para estructurar datos se conoce como DDL, 3) las consultas de dichos datos se denomina DML y 4) los sistemas de  gestión de base de datos, son los DBMS (Data Base Management System) (Sena, 2017). Por otro lado en las estructuras no relacionales los registros se estructuran en documentos, especialmente maquetados en el dual clave-valor.

Es así que en la estructura no relacional, los objetos tienen cierto formato. Muy similar a lo que se escribe en javascript donde se puede incluir un valor, o una lista de valores (arrays). La diferencia radica en el empleo de comillas, como sucede en el caso de un formato JSON en lo que respecta a definir una propiedad, y valor de dicha propiedad. En ese sentido, se pueden identificar 1) arreglos 2) documentos 3) arreglos de documentos (Sena, 2017)

Cabe resaltar que dentro de los gestores de base de datos existen preferencias de las cuales se merece mencionar destacan MySQL, MongoDB y PostgreSQL en un sondeo de opinión realizado, y entre otras que se mencionan en la imagen. (Fazt Code, 2020) Eso significa que tras cada una de estas tecnologías existen empresas que las desarrollan presentando particularidades en el manejo del lenguaje SQL y la plataforma que se emplea según la aceptación que tienen en la comunidad informática.

. 

Por otro lado, si nos enfocamos en una referencia en torno a los gestores de base de datos NoSql más populares Apache Cassandra, CouchDB, Redis, Neo4j, Firebase, MongoDB son frecuentemente utilizados. (Ávila, 2020) Sin embargo, los gestores NoSql se encuentran en una fase inicial, lo cual requiere de mayor investigación y esfuerzos, pese a que cuentan con ventajas en torno a su escalabilidad tipo horizontal. Según algunas consideraciones las propiedades BASE (Nosql) no contarían con las capacidades organizativas de las propiedades ACID que se vinculan a la gestión SQL (Richards, 2015).

|  |  |
| --- | --- |
| **Propiedades BASE** | **Propiedades ACID** |
| Consistencia eventual | Atomicidad, consistencia, isolación y durabilidad |
| Tiene un acercamiento optimista | Su acercamiento es pesimista |

En la actualidad también se hacen esfuerzos en aprovechar las potencialidad de ambas tecnologías. Es decir lograr alcanzar la escalabilidad de los gestores noSql  al mismo tiempo con la especificidad que cuentan los gestores SQL. La tecnologìa NewSQL es un intento de gestionar las bases de datos relacionales con la misma escalabilidad que lo hacen las no relacionales. Una escalabilidad horizontal y vertical aplicada a los respectivos sistemas de datos. (Tailor et al., 2014, 6)

**1.1.1 Sistema de base datos**

Uno de los principales retos con los cuales la gestión de la información ha tenido que lidiar ha sido la baja velocidad sobre todo para el desarrollo de aplicaciones. El ingreso de información, uso de analíticas, simulaciones, entre otras operaciones que se dan en la gestión de base de datos requieren de un sistema computacional adecuado (Hulburt et al., 2017, 22). En especial cuando, los servidores almacenan la información de forma local o en una nube, sus procesos requieren de arquitecturas multinúcleo, para alcanzar dicho objetivo, junto a los sistemas de gestión de base de datos.

**1.1.2 Arquitectura con tareas en paralelo**

Los procesadores en la actualidad son multinúcleo, donde las tareas asignadas por el usuario o sistema operativo pueden resolverse de forma simultánea (Stevechambitas, 2021). En lo que respecta a la gestión de base de datos, las operaciones que se realizan se distribuyen hoy en día de forma paralela, optimizando el tiempo de gestión de la información (Morales-Morales & Durán-Cazar, 2019, 52). Así la lectura y escritura de datos tiene lugar en la memoria principal, caracterizada por ser de una capacidad grande y de estilo no volátil en lo que respecta al almacenamiento, para que la información se mantenga registrada  (Muñoz-Plascencias, 2021)

**1.1.3 Memoria principal de almacenamiento o tecnología in memory**

La tecnología in memory se desarrolla ante la urgencia inmediata de acceder a la información almacenada en un servidor sin la necesidad de utilizar el disco duro del mismo. En especial cuando hoy en día hablamos del manejo de grandes volúmenes de información. Esta cualidad, entre otras, permite que bases de datos complejas puedan ser desplazadas de forma permanente en la memoria principal del servidor en lo que forma parte de una gestión de base de datos tipo IMDBMS. (Broke, 2014, 153)

**1.2 Bases de datos no relacionales**

Cuando hablamos de bases de datos no relacionales, se hace referencia a sistemas de almacenamiento de información que no siguen el modelo entidad relación o SQL. Este tipo provee un esquema más flexible en lo que respecta a escalabilidad, donde el sacrificio de la consistencia de la información se traduce en una mayor performance y disponibilidad de la información (Martín et al., 2013, 168). Es fácil detectar su emergente aplicación, ante las nuevas demandas que nacen en el uso de redes sociales, buscadores y aplicaciones que toman en cuenta la nueva ola de la web 2.0.

Algunos expertos han decidido concluir que esto se da como parte de un fenómeno Big Data, donde la cantidad de información que se maneja es compleja. Y en lo que respecta específicamente a la comunidad de desarrolladores donde se requiere que las bases de datos se adapten a la tipología de las unidades de información y que sus alteraciones sean nulas cuando de cambios de estructuras de contenido se trate (Morales-Morales & Durán-Cazar, 2019, 56). En lo que refiere a las plataformas NoSQL según Osmel y Yanes se pueden identificar cinco grupos.

**1.2.1 Por clave - valor**

La base de datos en este tipo de plataformas es simplemente el lugar en el que se almacena la información. En otras palabras la verificación de integridades, la comprobación de referencias tipo cruz se ha de ejecutar a nivel del cliente, en la aplicación que dicho cliente emplee (“Bases de datos Nosql”, 2013, p.8) y no en la base de datos. Así se maneja la información dentro de la base en forma de pares donde una clave cuenta con diferentes valores.

Esta es una de las categorías más sencillas en las bases de datos relacionales. Los valores en este tipo de sistemas son flexibles siguiendo una estructura, desde simples caracteres hasta vectores con diferente tipo de información (Mohammed-Eshtay et al., 2019, p.46). Es por ello que este tipo de bases de datos pueden ser vistas bajo el nombre *key-value*, como los dos campos que estructuran la información que almacenan.

|  |  |
| --- | --- |
| **clave** | **valor** |
| Cliente 1:Nombre | “Victor” |
| Cliente 2: Nombre | “Carlos” |
| Cliente 1: Funciones | {Marketing, Diseño, web} |
| Cliente 2:Teléfonos | {0995479142, 099545612} |

La simplicidad con la que se estructura la información en las bases *key-value* las convierte en un modelo óptimo para el tratamiento de información que sigue los esquemas de una *big data*.  En otros términos, este tipo de modelo de base de datos se enfoca más en la escalabilidad antes que en la consistencia de la información. Entre los almacenamientos más populares hasta la fecha que emplean un modelo key-value se pueden mencionar a Amazon dynamo, LinkedIn, Voldemort, Redis y Basho Riak (Mohammed-Eshtay et al., 2019, p.46)

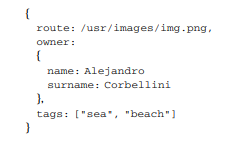
**1.2.2 Por documentos**

Al ser una variante de la anterior, los valores almacenan documentos. Esto genera una enorme ventaja ya que se puede almacenar y recuperar datos en una sola unidad (Nayak et al., 2013, 17) .Los documentos que recepta este tipo de base están en torno a algunos ejemplos entre los documentos más usados hoy en día. PDF, JSON, XML, BSON etc.

¿En qué consiste el concepto de documento dentro de este tipo de base de datos? La idea principal es proveer un desempeño óptimo en lo que respecta al almacenamiento y al manejo de querys (librerías que permiten ejecutar más rápido determinadas funciones)(Mohammed-Eshtay et al., 2019, p.47). Usualmente los documentos contienen información semi estructurada bajo los atributos nombre-valor. Es así que cada uno de los documentos almacena un conglomerado de información con campos interconectados junto a una lista de atributos que permiten una adecuada indexación.

Bajo esta característica, las bases no relacionales por documentos no tienen una estructura completamente relacional, su principal ventaja es que nuevas características pueden ser añadidas siguiendo determinadas reglas (Corbellini et al., 2017, p.14). Las reglas vienen especificadas según la tipología de los documentos, por ejemplo para citas los casos más populares, XML, JSON Y BSON. Esto permite tener una ventaja adicional con respecto a las base de datos *key-value*, como sucede con las operaciones de  adición, modificación de datos y sobre determinados *querys*.

 Por ejemplo, se pueden añadir nuevas características sobre todo cuando todas las características en diseño no han sido aún llenadas(Corbellini et al., 2017, p.14) como se muestra en el siguiente ejemplo:



Así si se suman al anterior documento la potestad de acceder a la imagen de un sitio Web, una comprobación y valoraciones de usuario, el documento se podría mostrar de la siguiente manera.



**1.2.3 Por columna**

La estructura principal de este tipo de plataforma radica en la serie de conjunto en base a columna-columna donde se puede almacenar la información. Así que se puede pensar en forma de un atributo vectorial cuyos valores representan cada característica que corresponden al atributo vector (Nayak et al., 2013, 17). Esto vuelve a las plataformas columnares ideales para organizar analíticas y datos estadísticos complejos. Es así que la información se enfoca en un almacenamiento de la información netamente columnar y por filas (Eshtay et al., 2019, p.46)

En este modelo de base de datos cada columna tiene una indexación única por ello es que minimiza el costo I/O al máximo. Así los componentes de fila son un conglomerado de familias de columnas que contienen los valores de las filas dadas (Eshtay et al., 2019, p.47).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Clave Fila 1 | Clave Columna1 | Clave Columna2 | Clave Columna 3 | ... |
| Valor Columna 1 | Valor Columna 2 | Valor Columna 3 |  |
|  | | | | |

A diferencia de los otros modelos de base de datos no relacionales las categorías y el almacenamiento de la información de manera columnar es óptima en indexación y querying.

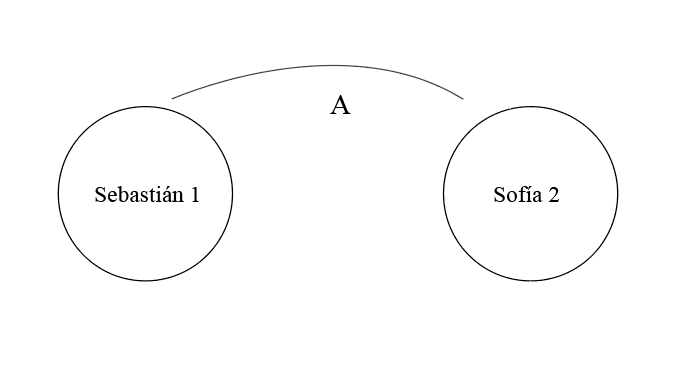
**1.2.4 Por grafo**

Son ideales para el análisis complejo de redes de información no lineales. (Hurlburt, 2017, 22) El objetivo principal es poder gestionar información relacionada con distintos nodos de interconexión. Por ejemplo se usan en la cotidianeidad de estudios transporte público, mapas, topología, sistemas de alimentación eléctrica, etc. La forma en que este modelo de base de datos funciona es direccionando el almacenamiento y sus enlaces de manera gráfica (Corbellini et al., 2017, p. 16).

Para el caso de base datos relacionales llevar de manera gráfica cada vinculación requeriría una gran inversión de almacenamiento. La fortaleza de este tipo de bases es que las relaciones entre los diferentes datos se estructuran en vértices que contienen información y listas de indicadores con otro tipo de vértices (Corbellini et al., 2017, p. 17).

|  |  |  |
| --- | --- | --- |
| Vértice | Nombre | Email |
| 1 | Sebastián | kyriosbastiano@gmail.com |
| 2 | Sofía | snunez@gmail.com |

|  |  |
| --- | --- |
| Vértices | Data |
| 1-2 | A |



**1.2.5 Por objetos**

Las unidades de almacenamiento en este tipo de gestor de base datos se representan como un objeto (Nayak et al., 2013, 18) No se emplean cuando de relaciones simples se trata ya que manejan los objetos, sus atributos de clases, y clases en torno a tuplas con un id único para cada objeto.

**1.3 Teorema CAP**

Al teorema CAP se le conoce como la teoría de Brewers, un término que nació durante la ponencia del profesor Erick Brewer sobre sistemas distribuidos computacionales. (IBM Cloud Education, 2019) La idea principal de esta teoría radica en la imposibilidad para el diseño de una plataforma de gestión de información perfecta que provea al mismo tiempo, consistencia, alta disponibilidad y partición  en lo que respecta a un servidor web.

El teorema CAP permite responder cuando es más adecuado utilizar un gestor de base de datos relacional y no relacional (Zacavix Tech, 2021) tomando en cuenta estas tres asignaciones. Y esto se justifica ya que las aplicaciones modernas de hoy en día es común que usen más de un tipo de gestor de base de datos que otros. El teorema CAP aplica la lógica de los sistemas distribuidos, siendo cada unidad de almacenamiento de información un posible nodo que procesa la información y que corresponde a una amplia red o networking de información (IBM Cloud Education, 2019). La descripción más utilizada es que todo sistema cumpla el máximo de las cualidades mencionadas (consistencia, disponibilidad alta y partición.)

Consistencia, implica que para toda consulta se debe recibir la misma información de forma independiente del servidor nodo que haga el pedido.

Alta disponibilidad, que todos los clientes puedan leer la información pese a que uno de los nodos, no esté funcionando.

Partición, el sistema debe seguir activo aún si uno de los nodos ubicados geográficamente en una zona distinta, deja de funcionar.(Gilbert & Lynch, 2012, 3)

**2. Estudio del arte**

El problema generalizado de los autores revisados tiene en consideración al big data como asunto tanto en el ámbito empresarial como institucional que debe ser revisado por los profesionales en el área (Vivanco-Vazquez, 2019; Alvear-Merizalde & Saltos Saá, 2020; Morales-Morales et al., 2019). Las investigaciones tienen como punto problemático la necesidad de referentes entre las bases de datos para posibles aplicaciones. Al respecto, Vivanco profundiza más en lo que implica un sistema de servidores (nodos) que se relacionan para saber el tiempo que demoran realizar consultas de creación, lectura, actualización y borrado (CRUD) (Vivanco-vásquez, 2019, pp.8-55).

Alvear y Saltos (2020, pp.11-25) por otro lado establecen un análisis para extracción y almacenamiento tomando en cuenta gestores como Elasticsearch y JVM en lo que corresponde a extracción, y para almacenamiento, entre MongoDB y PostgreSql. Finalmente, Tandazo y Duran (2018, pp. 5-50) investigan los tiempos de carga con relación a tres consultas, clave-valor, setting (where) y de una función suma cómo se compara a continuación.

**2.1 Comparativa entre almacenamiento y extracción en gestores de bases de datos**

Se aborda el problema del Big Data en el manejo de información compleja bajo un enfoque empresarial, destacando la necesidad que tiene este sector y las ventajas que de ello pueden surgir, sobre la administración de una gran cantidad de información (Alvear-Merizalde & Saltos-Saá, 2020, p.3) Las autoras se proponen realizar una comparativa entre un representante de los gestores de bases de datos no lineales, con los lineales. En especial en lo que corresponde a sugerir una arquitectura de software en lo que corresponde a la extracción y almacenamiento de grandes cantidades de información. Para ello se enfocan en los tiempos de ejecución de las operaciones de extracción y almacenamiento. Alvear (2020, pp.10-8) toma en cuenta datos

La interesante propuesta de las autoras es que se adentran en las ventajas que ofrece *Elasticsearch* como una herramienta de almacenamiento de información. En sus palabras “un sistema analítico donde se almacenan datos en formato índice inverso Apache Lucene” (Alvear-Merizalde & Saltos-Saá, 2020, p.3). En lo que corresponde al almacenamiento, dicha herramienta, cuenta con administrador master quien puede realizar dos tipos de operaciones: lectura y escritura.

También se pueden realizar réplicas del máster que tiene la información almacenada, y lo que llama la atención es la capacidad que Elasticsearch cuenta para procesar distintas consultas al mismo tiempo. La información que planean realizar las mediciones dentro de esta investigación toma en cuenta las API o información pública de dos importantes redes sociales. Por un lado Twitter Api con 328 millones de usuarios se convierte en una plataforma interesante de almacenamiento de información (Alvear-Merizalde & Saltos-Saá, 2020, p.23)

Por otro lado, YouTube Data Api V3 desarrollado por la multinacional Google. Esta app de acceso público forma parte de una de las tres app que cuenta la plataforma de Youtube. Accediendo a la información de uno de los canales de Youtube, con el ID del canal las autoras analizan la información de comentarios, likes, dislikes.

El tipo de almacenamiento que se realiza en esta investigación sobre la información obtenida en las redes sociales mencionadas cubre dos rubros, una comparativa entre memoria JVM (Java) con relación al almacenamiento en memoria de *Elasticsearch*. Y una segunda es la comparativa entre un gestor Sql con uno no SQL. Siendo los tiempos de almacenamiento almacenados medidos en milisegundos a través del método currentTimeMillis de Java (Alvear-Merizalde & Saltos-Saá, 2020, p.29), para el caso de las primeras mediciones las autoras toman en cuenta muestras de 1000, 5000, 10000, 20000 y 50000, y para el segundo caso las muestras varían entre 1000, 5000, 10000, 20000, 50000 y 100000.

Los resultados obtenidos indican que para la primera comparativa a medida que el volumen de datos aumenta la memoria también lo hace para el caso de JVM. En palabras de las investigadoras por el otro lado “Elasticsearch tiene una respuesta lenta, no necesita más de 2gb para trabajar con todos los volúmenes” (Alvear-Merizalde & Saltos-Saá, 2020, p.29). Lo que significa menos consumo de memoria para el almacenamiento Elasticsearch.

En el caso de los gestores de base de datos, dichos gestores se seleccionaron tomando en cuenta su condición de código abierto, que sean conocidas en el mercado, y que sean de conocimiento previo de los investigadores. Así las comparativas en este estudio se basaron entre PostgreSQL (lineal) y MongoDB (no lineal). Demostrando los resultados que MongoDB a partir de volúmenes superiores a los 4000 saca una mayor ventaja en tiempo de ejecución en comparación con PostgreSQL (Alvear-Merizalde & Saltos-Saá, 2020, pp.46-47).

**2.2 Estudio del rendimiento de una base columnar**

Según Tandazo-Gaona y Durán-Cazar (2018, p.4) el principal problema que se identifica cuando se realiza una comparativa entre base de datos relacionales y no relacionales es que las primeras (Sql) pierden rendimiento cuando de grandes volúmenes de información se trata. Y se convierte en un reto para los manejadores de base datos elegir la mejor elección en lo que respecta a sus proyectos. Las empresas ante ello estarían en una situación vulnerable al no poder darse abasto al manejar grandes cúmulos de información que se dan con los nuevos avances tecnológicos.

Los autores prácticamente buscan encontrar un límite para establecer el rendimiento de una base de datos lineal en un entorno operativo. En ello evalúan diferentes escenarios para una muestra de bases relacional y no relacional de manera comparativa. Y la principal evaluación se centra en el tiempo de ejecución de determinadas consultas.

La elección de los gestores de base de datos en esta investigación fueron MySql y PostgreSql (relacional) y Cassandra v.3.1.0., MonetDB y MongoDB (no relacional). Para la selección se tomaron en cuenta versiones no comerciales, que no necesiten licencia y que se manejen en sistema multiplataforma (Tandazo-Gaona y Durán-Cazar, 2018, p.29). Además, dichas bases de datos seleccionadas, forman parte de un ranking internacional que es de libre acceso a su información.

La consulta se realizó a través de una base de datos con 125 millones de registros. Mismo registro que es cargado en 4 cargas incrementales de 1 millón, 10 millones, 25 millones y 50 millones. Haciendo el análisis respectivo en cuatro escenarios y con tres tipos de consultas aplicadas tanto a los gestores de bases de datos lineales y no lineales (Tandazo-Gaona y Durán-Cazar, 2018, p.33).

Entre las consultas que se mencionan dentro de este trabajo para la aplicación de las mediciones están distribuidas: por clave-valor en pares, por conglomerado de datos (setting) y ejecución de operación (sumatoria). La herramienta utilizada para realizar la carga en las distintas bases es *Pentaho Data Integration* que permite realizar la carga de archivos CSV en los gestores arriba mencionados(Tandazo-Gaona y Durán-Cazar, 2018, p.36). Y los valores obtenidos del tiempo, se reflejaron en un promedio de cinco mediciones.

**2.2.1 Mediciones y resultados de eficiencia relacional y no relacional**

Los resultados en torno a valores muestrales de 1 millón en la consulta (clave-valor) se revela la constante en los tiempos de carga. Es decir que el tiempo de carga es semejante en los gestores de base de datos relacionales y no relacionales. Por otro lado, en la segunda consulta (where) el gestor MySql muestra un rendimiento deficiente en comparación al resto con una respuesta de 531.2 milisegundos (Tandazo-Gaona & Durán Cazar, 2018, p.44). Siendo Cassandra con un tiempo de 9.4 milisegundos la más óptima en lo que corresponde a la selección de un setting de información.

En lo referente a la función suma (SUM) en este método de agregación, los tiempos óptimos corresponden a los gestores de datos : Cassandra, MonetDB y MongoDB, destacando Cassandra con un tiempo de 62.6 milisegundos (Tandazo-Gaona & Durán Cazar, 2018, p.44). En el otro extremo, con el peor rendimiento en la muestra de 1 millón de datos, MySql registra un tiempo de 353 milisegundos. En otros términos es aproximadamente 6 veces menos eficiente que un gestor de base de datos no relacional, tomando en cuenta los resultados de Cassandra.

En el segundo escenario, en una muestra de un total de 10 millones de datos, los tiempos son similares para todas los gestores de bases de datos. Sin embargo, las diferencias aparecen en lo que respecta a la segunda y tercera consulta (Tandazo-Gaona & Durán Cazar, 2018, p.44). En la segunda consulta, el gestor MySql tiene los valores más bajos, 6422 milisegundos, a diferencia de Cassandra con los mejores valores 20.4 milisegundos, le sigue MonetDB con 268.4 milisegundos y en tercer posición MongoDB con 251.4 milisegundos. La diferencia es grande, ya que con MySql la relación es de aproximadamente 314 veces en contra con lo que respecta al mejor representante de los gestores de bases de datos no relacionales.

Para la tercera consulta, los valores son aproximados en lo que respecta a los gestores de bases de datos no relacionales (MongoDB, MonetDB y Cassandra). También se merece resaltar que existen unas referencias interesantes en lo que respecta a las mediciones columnares. Entre MySql y PostgreSql, la de menor rendimiento es MySql. La relación de tiempo está entre dos valores, 3784.4 milisegundos y 148. milisegundos para ambos gestores de base de datos respectivamente.

Finalmente, en lo referente a la tercera muestra, para mencionar, 25 millones de datos, al igual que en las dos muestras anteriores, en lo que respecta a la primera consulta, los valores son similares (Tandazo-Gaona & Durán Cazar, 2018, p.46). Entre las bases de datos no relacionales Cassandra y Monet tienen valores de respuesta parecidos en las consultas seteado y de adición. Y para los tres casos de consulta, MySql mantiene las referencias más lentas en comparación a los gestores de bases de datos analizados.

**2.3 Análisis de desempeño computacional entre gestores de bases de datos no relacionales.**

El análisis toma en cuenta gestores de base de datos no tan conocidos según el ranking que en nuestro estudio se emplea. Vivanco (2019, p. 2) realiza una investigación introductoria sobre una comparativa más específica entre base de datos no relacionales. Para ello utiliza como muestra referencial de información a la base UniProt/Swiss-Prot la misma que posee cerca de 550.950 entradas sobre proteínas. Su aporte se centra en la importancia que deben tener herramientas como Elasticsearch y Hadoop (gestores no Sql) en lo que respecta a consultas CRUD para la bioinformática.

En análisis de consulta CRUD corresponde a creación, lectura, actualización y borrado de datos aplicado a la base UniPrt/Swiss-Prot que cuenta con 4 atributos, *accesion*, *name*, *gene* y *organism*.  Y la cualidad destacada de los gestores de bases de datos donde se aplican las consultas es que manejan un modelo de arquitectura, maestro-esclavo. En lo que respecta a la metodología seguida por la autora, sigue dos métodos. Una de ciclo para preparar los datos y otra de evaluación que dirige los pasos para la medición (Vivando-vásquez, 2019, p.8). Para la preparación de los datos, los mismos se convierten de formato XML a objetos JSON, que son reconocidos por las plataformas no relacionales.

En lo referente a la evaluación se utilizan nodos para el levantamiento de los procesos respectivos. Tanto para Hadoop y Elasticsearch desde el lado del cliente se envían las peticiones. Es decir, el cliente ejecuta los scripts con las consultas mencionadas arriba en formato Python para que se redireccione a los servidores maestro y esclavo tanto desde las plataformas Hadoop y Elasticsearch, respectivamente (Vivando-vásquez, 2019, p.23). Las mediciones se realizan tomando en cuenta el tiempo que emplea el retorno de una salida respectiva usando un cuadro comparativo con estándares de medición utilizando prototipos y criterios: Escenario, tiempo requerido, herramientas, exactitud, costo y confiabilidad.

**2.31. Mediciones por atributos y no por carga total**

Cuando se realiza la comparativa entre ambos gestores de bases de datos con la opción de crear, en Hadoop el 70% del ingreso de cada uno de los atributos se registra entre los 0 a 100 segundos. A diferencia de los resultados obtenidos en Elasticsearch, donde los resultados se encontraron en el rango de los 0 a 0.03 segundos. En lo que respecta a la consulta read, los valores deben ser categorizados en cada uno de los atributos *name*, *accession, organism, gene* respectivamente.

Es así que en *Elasticsearch* consulta *read* en atributo name, los valores corresponden a 16 771 499 donde los tiempos de respuesta del 99.45% de dichos valores obtenidos están entre 0 y 0.03 segundos. Similar sucede con *accession* de los 23 593 023 valores medidos el 99.24% de los resultados tienen un tiempo de respuesta entre 0 y 0.03 segundos. En lo que corresponde a *organism*, el 98.53% de los resultados obtenidos en la búsqueda están entre los 0 y 0.03 segundos (Vivanco-Vásquez, 2019, p.50). En el caso del atributo *gene*, el 99.47% de los datos obtenidos en la consulta también se encuentran entre los tiempos 0 y 0.03 segundos.

Cuando se compara con *Hadoop* con la consulta read para el atributo name, se obtienen 96 341 datos de los cuales el 98.92% tienen un tiempo de consulta que va entre los 0 y 10 segundos. Cuando se realiza en la consulta *accession* se generan 79 679 resultados entre los cuales su 99.22% también tiene su tiempo de consulta entre los 0 y 10 segundos. En lo que corresponde a *organism*, los valores se favorecen ya que de los 106 571 valores generados, el 85% cuenta con un tiempo entre 0 a 2 segundos. Y para el atributo de *gene* de 1100 719, su 83% demoran también entre 0 a 2 segundos.

En lo relacionado a la carga Update, los cambios son más diferenciados. Ya que de los valores generados en Elasticsearch (120) el 96.67% tienen un valor de 0 a 1 segundo. En Hadoop, de los 120 resultados, el 68.33% tienen un tiempo de carga entre 0 a 10 segundos. Para la consulta delete se puede manifestar lo siguiente.

En *Elasticsearch delete*, (120 resultados obtenidos) en la eliminación se registra un rango de tiempo entre 0 y 2 segundos para el 100% de los datos.  En comparación con Hadoop cuyo valor está entre los 0 y 10 segundos en lo que corresponde al 95.83% de los resultados obtenidos (120).

**2.4 Referencias concluyentes o posible hipótesis**

Es necesario mencionar que todas las investigaciones toman en cuenta que la velocidad en la que se ejecutan las operaciones bajo distintos métodos, indica que los gestores de las bases de datos no relacionales tienen un menor tiempo. Es por ello que se justifica su eficiencia cuando de big data se trata. Además, también concluyen que el gestor de datos MySql puede ser el que menor rendimiento tenga al momento de ejecutar una operación. Por otro lado, dentro de las bases no relacionales, los valores pueden ser bastante similares.

Esta referencia se aplicaría tanto en las cuatro dimensiones que CRUD (creación, lectura, actualización y borrado) establece para una arquitectura informática. Ya que las tres investigaciones separadamente abordan tanto el lado del cliente y del servidor para realizar un análisis amplio.

Objetivo general

Tomando en cuenta las consideraciones anteriores, en especial, con la posible ventaja en ejecutar consultas que cuentan los gestores de bases de datos no relacionales Vivanco-Vazquez, 2019; Alvear-Merizalde & Saltos Saá, 2020; Morales-Morales et al., 2019) esta investigación tiene como objetivo general identificar la relación que existe entre los gestores de bases no relacionales con los gestores de bases de datos relacionales de forma exploratoria. Para poder llegar a este objetivo se establecen tres objetivos secundarios. El primero radica en comparar las velocidades de tres consultas base aleatorias (carga, selección y operación) entre ambos gestores (relacional vs no relacional).

También se plantea analizar 3 gestores de bases de datos populares dentro de la comunidad de informáticos y programadores tanto relacionales como no relacionales considerando, primero, según ranking de la encuestadora DB-engines (2022*, DB-Engines Ranking of Relational DBMS*), y segundo con la información obtenida en la consultora G2 (https://www.g2.com/). Dentro de estas dos consideraciones los criterios de selección de las muestras dependen del acceso a la información sobre el uso de las bases de datos que haya disponible y la gratuidad de las mismas. Es decir, las muestras cumplen criterios no probabilísticos.

Así entre la muestra de gestores relacionales se merece mencionar MySQL y PostgreSQL y entre los no relacionales a MogoDB y Casandra por ser de código abierto y cumplir las características mencionadas. También para ello se plantea medir las velocidades en milisegundos para tener los datos más precisos que permitan diferenciar cambios imperceptibles. Y tercero, comparar las mediciones en dos etapas, la primera para 250 mil y la segunda para 500 mil y la tercer para un millón.

Metodología

La investigación cumple características exploratorias, ya que existe muy poca información desarrollada en lo que respecta al *big data* en el Ecuador, y se desea generar una contribución a este tema con una metodología cuantitativa. Las mediciones se basarán en una muestra de 1 millón de datos. Esto cumple características estadísticamente aceptables para lo que se entendería como big data. Se utilizará la herramienta Aqua Studio para poder medir en milisegundos las consultas planteadas y que han sido investigadas por el autor dentro de los gestores que se encuentran en el ranking considerado de bases lineales y no lineales.

Además, este trabajo académico representa la fase uno de una serie de contribuciones que permitan entender más el asunto sobre las bases de datos. Los resultados están enfocados en la utilidad para otras investigaciones por ejemplo aquellas que se enfoquen o consideren una fase de extracción de información, por ejemplo.

Los gestores que se plantean relacionar son Postgre SQL, MySQL, María Db en los relacionales y no relacionales MongoDB, Cassandra y Neo4j considerando una base un millón de datos. Los datos cumplen con las características de clave valor, tomando en cuenta sólo el campo nominación (VARCHAR), valor (INT) y true/false (Boleann) bajo esquemas cuantitativos de selección. El servidor de almacenamiento será local.



**Figura 1: Logo de Unade (ejemplo)**

### Antecedentes

## MARCO EMPÍRICO

Realizar el desarrollo de todos los puntos mencionados en capítulo primero (marco introductorio) dícese:

* Hipótesis (si las hay).
* Objetivos generales.
* Objetivos específicos.

También debe incluir como **mínimo** los siguientes puntos:

* Diseño y tipo de investigación.
* Población y muestra.
* Instrumentos de evaluación.
* Resultados.

De acuerdo con la bibliografía mencionada los gestores de bases de datos no relacionales cumplen en menor tiempo la ejecución de las consultas cuando de una administración de información que cumple las características de big data se trata (A.B.M.-Moniruzzaman & Akhter-Hossain, 2013; Muños-Plascencia, 2020; Mercedes-Marqués, 2011; Binani & Upadhyyay, 2016; Tailor-Hemang et al., 2014). Esto indicaría que la hipótesis con la que se aproxima esta investigación consiste en demostrar la validez de lo arriba mencionado al expresar dentro de una hipótesis nula, que los gestores de bases de datos no lineales son poco efectivos en realizar consultas en comparación a los lineales tomando en cuenta su tiempo de ejecución.

Para lograr el acometido es plausible plantear los objetivos que guían esta iniciativa. Se recuerda al lector que esta investigación es un primer paso y una contribución inicial que permita comprender de mejor manera el amplio mundo de los gestores de bases de datos dentro de un escenario que maneja a la big data como dimensión.

Es por ello que el objetivo primordial se basa en comparar una muestra de ambos gestores de bases de datos tomando como referencias a los gestores según la disponibilidad de la información sobre determinado gestor, la gratuidad del mismo, y la popularidad que se manifiesta dentro de las encuestadoras DB-engines (2022*, DB-Engines Ranking of Relational DBMS*) y G2 (<https://www.g2.com/>).

Así se plantean tres objetivos específicos siendo el primero comparar las velocidades de tres consultas de base aleatorias (carga, selección y operación) entre ambos gestores (relacional vs no relacional). Para el primer objetivo las referencias tomadas de carga CSV, cuentan con un millón de datos tomando en cuenta una atomicidad inicial. Es decir bajo nominalidad y valor vinculado a cada campo con nombre (string), valor (int) y valor\_comun (int o bolean). Estos tipos de datos tienen también en cuenta el registro valor clave-valor con los que trabajan los gestores no lineales.

El segundo objetivo es analizar 3 gestores de bases de datos populares dentro de la comunidad de informáticos y programadores tanto relacionales como no relacionales considerando, primero, según ranking de la encuestadora DB-engines (2022*, DB-Engines Ranking of Relational DBMS.* Y también tomado en cuenta disponibilidad de la información sobre determinado gestor, la gratuidad del mismo, su capacidad de contar con medidores de tiempo de ejecución de consultas o su vinculación con la plataforma Aqua Data Studio.

Y finalmente medir las velocidades de las diferentes consultas mencionadas para tener referentes de comparación que permitan verificar la hipótesis nula planteada. Las mismas se realizan en milisegundos y se transforman a segundos para tomar en cuenta gráficamente la brecha en tiempos que pueden existir entre los gestores de bases relacionales en comparación con los no relacionales. Incluida también una comparativa entre los ejemplos mencionados de cada grupo, para también hacer un breve modelo interno.

Es así que la estructura del *csv* referencial se configuró de la siguiente manera. Con una tabla (documento) con los siguientes campos (valores), como se indica en la figura.

|  |  |  |
| --- | --- | --- |
| Nombre | Valor | Valor\_común |
| (*String*) | (*Integer*) | (*Integer*) |

Dentro de la cual se registraron 1048576 datos como una pauta referencial de lo que correspondería una base con una de las características que se consideran bajo el *big data*.

En lo que corresponde a las consultas se investigan los diferentes métodos que cada uno de los grupos de gestores de bases cumple en lo que corresponde a una carga (*load*), operación de suma (*sum*) y una selección (*select*) con sus diferentes adaptaciones y similitudes, similitudes en especial que se observan en la bases de datos relacionales que siguen una nomenclatura Sql.

En lo que respecta a Mysql y Maríadb en carga de información con el método LOAD DATA INFILE. En Postgresql con COPY PUBLIC. FROM. Para la consulta de selección en los tres gestores con SELECT \* FROM desde la consola, y finalmente con relación a la operación, SELECT SUM (). Para lo referente a la selección se tomo en cuenta el método LIMIT 100 para solo mostrar los 100 primeros datos cumpliendo una condición WHERE.

De forma análoga se investiga las consultas que corresponden en las bases de datos no relacionales (*Not only Sql*). Donde cada una tiene sus propios mecanismos o métodos unos más alejados a la nomenclatura Sql. Siendo así para MongoDb la carga con MONGOIMPORT, la selección con el método junto a FIND y MATCH como condicional, y en el caso de operación utilizando AGGREGATE y $SUM. En lo referente a NEO4J, las consultas con relación a carga, selección y operación respectivamente se basan empleando LOAD CSV WITH HEADER, MATCH RETURN y MATCH RETURN SUM. Y con lo que respecta a Apache Cassandra, se emplea COPY FROM (carga), SELECT FROM y SELECT SUM (operación).

La investigación que se realiza es cuantitativa tomando en cuenta valores medidos de tiempo de ejecución de las consultas mencionadas. La frecuencia de medición se basa en una toma de tres valores para tener una media referencial de tiempo y realizar una aproximación de la comparación que se busca realizar. Se recuerda que esta investigación es de carácter exploratorio y bajo las condiciones de diseño se obtuvieron los siguientes resultados en las mediciones.

|  |  |  |  |
| --- | --- | --- | --- |
|  | CONSULTAS (tiempo en segundos) | | |
|  | carga | operación | selección |
| Mysql | 4 | 0,822 | 0,075 |
| 4,25 | 0,812 | 0,086 |
| 4,28 | 0,802 | 0,077 |
| MONGO | 0,03 | 1 | 0,092 |
| 0,03 | 1 | 0,089 |
| 0,04 | 1 | 0,078 |
| POSTRESQL | 0,89 | 0,191 | 0,072 |
| 0,988 | 0,119 | 0,073 |
| 0,881 | 0,118 | 0,071 |
| MARIA DB | 2 | 0,368 | 0,074 |
| 2 | 0,634 | 0,076 |
| 2 | 0,35 | 0,077 |
| NEO4J | 2,917 | 0,251 | 0,01 |
| 2,212 | 0,242 | 0,01 |
| 2,176 | 0,204 | 0,01 |
| Cassandra | 20,953 | - | 0,063 |
| 21,912 | - | 0,065 |
| 20,277 | - | 0,059 |

Aclarando que en Apache Cassandra la operación no soporta la cantidad de 1048576. Es decir no cumpliría la condición de ser un gestor de big data cuando de una suma se trata. En referencia a la misma información

Los gestores que se plantean relacionar son Postgre SQL, MySQL, MongoDB y Cassandra considerando una base un millón de datos. Los datos cumplen con las características de clave valor, tomando en cuenta sólo el campo nominación (VARCHAR), id y valor (INT) y true/false (Boleann) bajo esquemas cuantitativos de selección. El servidor de almacenamiento será local.

## CONCLUSIONES, DISCUSION DE RESULTADOS Y RECOMENDACIONES

Interpretar los resultados, situarlos en el contexto de los hallazgos anteriores y explicar lo que significan para futuras investigaciones, así como para posibles aplicaciones a la vida real.

Incluir:

* Limitantes de investigación que se identifican (o identificaron si se aplicó la propuesta).
* Sugerencias de continuidad del proyecto de forma descriptiva (acciones, metas), retomando la línea de investigación (en relación al punto que señala la posible aplicabilidad en la vida real y futuras investigaciones)
* Recomendaciones: partiendo de la contrastación de la tesis con el marco teórico propuesto (o resultados si se presentan)

# REFERENCIAS BIBLIOGRÁFICAS

Citar según Normativa APA. Incluir un mínimo de **35 referencias bibliográficas.**

(A.B.M.-Moniruzzaman & Akhter-Hossain, 2013, 2) A. B. M. Moniruzzaman and S. A. Hossain, “NoSQL database: New era of databases for big data analytics - classification, characteristics and comparison,” International Journal of Database Theory and Application, vol. 6, no. 4, pp. 1–5, 2013. [Online]. Available: <http://bit.ly/2XaKoPK>

Avila, J.  [1 Bases de Datos NoSql]. (15 de junio 2020). *Bases de datos NoSql*

(tutorial) [Video]. Youtube.<https://www.youtube.com/watch?v=caS51djbuk4&t=46s>

Morales-Morales, M. R., Durán-Cazar, J. W., Tandazo-Gaona, E. J., & Morales Cardoso, S. (2019). Rendimiento de bases de datos columnares. *Ingenius*, (22), 47–58. https://doi.org/10.17163/ings.n22.2019.05

(Mercedes-Marqués, 2011,5) M. Marqués, Bases de datos. Universitat Jaume, 2011. [Online]. Available: <http://bit.ly/2RcPtS9> (pagina 5)

Binani, S., Gutti, A., & Upadhyay, S. (2016). SQL vs. NoSQL vs. NewSQL- A Comparative Study. *Communications on Applied Electronics*, *6*(1), 43–46. <https://doi.org/10.5120/cae2016652418> (<https://www.mendeley.com/catalogue/8dcd3fec-9d59-3c0c-aad9-697f3910ea33/> + 43)

C

Corbellini, A., Mateos, C., Zunino, A., Godoy, D., & Schiaffino, S. (2017). Persisting big-data: The NoSQL landscape. *Information Systems*, *63*, 1–23. <https://doi.org/10.1016/j.is.2016.07.009>

Cueva y Melo https://repositorio.uasb.edu.ec/handle/10644/8325

D

Db-engines.(2022, 8 de marzo). *Db-engines ranking*. https://db-engines.com/en/

E

Eshtay, M., Sleit, A., & Aldwairi, M. (2019). Implementing bi-temporal properties into various NoSQL database categories. *International Journal of Computing*, *18*(1), 45–52. <https://doi.org/10.47839/ijc.18.1.1272>

G

G2. (2022, 8 de marzo).Best Relacional Databases. *Compare Relational Databases*. <https://www.g2.com/categories/relational-databases>

H. Plattner and B. Leukert, The In-Memory Revolution. Springer, 2015. [Online]. Available: http://bit.ly/2F3ezhO (<https://books.google.com.ec/books?id=6IdNCwAAQBAJ&pg=PA13&hl=es&source=gbs_toc_r&cad=3#v=onepage&q&f=false> )

Fazt Code. [Hablemos de Sql vs NoSql]. (26 de marzo 2020). *Hablemos de Sql y NoSql*

(tutorial) [Video]. Youtube.<https://www.youtube.com/watch?v=0xx3fcYcb28>

Avila, J.  [1 Bases de Datos NoSql]. (15 de junio 2020). *Bases de datos NoSql*

(tutorial) [Video]. Youtube.<https://www.youtube.com/watch?v=caS51djbuk4&t=46s>

Jenny Richards, Advantages and Disadvantages of NoSQL databases – what you should know, Hadoop360, September 24, 2015, <http://www.hadoop360.com/blog/advantages-anddisadvantages-of-nosql-databases-what-you-should-k>

Tailor, Hemang & Choudhary, Sushant & Jain, Vinay. (2014). Rise Of NewSql. 10.13140/2.1.4756.1608. <https://www.researchgate.net/publication/264744442_Rise_Of_NewSql>

Hurlburt, G. F., Thiruvathukal, G. K., & Lee, M. R. (2017, November 1). The Graph Database: Jack of All Trades or Just Not SQL? *IT Professional*. IEEE Computer Society. <https://doi.org/10.1109/MITP.2017.4241475>

S

Sena. [Cómo pasar de SQL a NoSQL sin sufrir]. (28 de marzo 2017). Conceptos de MongoDev para DEVS - Primera Parte (tutorial) [Blog]. <https://medium.com/techwomenc/como-pasar-de-sql-a-nosql-sin-sufrir-e34dd22349e5>

Stevechambitas.  [Procesador multinúcleo explicado]. (26 de enero 2021). *Procesador multinúcleo explicado* (tutorial) [Video]. Youtube.<https://www.youtube.com/watch?v=AuPJ8kL-iLs&list=PLXPZ__GYSXuEVOm40S0ZqtNHJPxuwzhL8&index=2&t=14s>

Muñoz-Plascencia, A. [Sistemas operativos multiprocesamiento]. (10 de junio 2020). *Sistemas operativos multiprocesamiento* (tutorial) [Video]. Youtube.<https://www.youtube.com/watch?v=fPozCdDjVg8>  (Muñoz-Plascencias, 2021)

vom Brocke, J., Debortoli, S., Müller, O., & Reuter, N. (2014). How in-memory technology can create business value: Insights from the hilti case. *Communications of the Association for Information Systems*, *34*(1), 151–168. <https://doi.org/10.17705/1cais.03407>

Martín, A., Chavez, S., Rodriguez, N., Valenzuela, A., & Murazzo, M. (2013). Bases de datos NoSql en cloud computing. *XV Workshop de …*, 166–170. Retrieved from <http://sedici.unlp.edu.ar/handle/10915/27121>

BASES DE DATOS NoSQL. (2013). *Revista Telemática*, *11*(3), 21–33. (pagina 8)

A. Nayak, A. Poriya, and D. Poojary, “Type of NoSQL databases and its comparison with relational databases,” International Journal of Applied Information Systems (IJAIS), vol. 5, no. 4, pp. 16–19, 2013. [Online]. Available: <http://bit.ly/2X2fIQQ> (pagina 17 )

Hurlburt, G. F., Thiruvathukal, G. K., & Lee, M. R. (2017, November 1). The Graph Database: Jack of All Trades or Just Not SQL? *IT Professional*. IEEE Computer Society. <https://doi.org/10.1109/MITP.2017.4241475> (PAGINA 22)

IBM Cloud Education. (2019). What is the CAP Theorem? | IBM. Retrieved from <https://www.ibm.com/cloud/learn/cap-theorem>

Zacavix Tech. [¿Qué gestor de base de datos elegir para un proyecto?]. (2 de mayo 2021). *¿Qué gestor de base de datos elegir para un proyecto?* (tutorial) [Video]. Youtube.<https://www.youtube.com/watch?v=MSSf954u_jU>

Gilbert, S., & Lynch, N. (2012). Perspectives on the CAP Theorem. *Computer*, *45*(2), 30–36. <https://doi.org/10.1109/mc.2011.389>

Crowd, Best Relational Databases Software. Crowd. Inc, 2019. [Online]. Available: http://bit.ly/2RbQPge

DB-Engines. (2019) Db-engines ranking of wide column stores. [Online]. Available: http://bit.ly/2KOBYHs

vom Brocke, J., Debortoli, S., Müller, O., & Reuter, N. (2014). How in-memory technology can create business value: Insights from the hilti case. *Communications of the Association for Information Systems*, *34*(1), 151–168. <https://doi.org/10.17705/1cais.03407>

Meena, J. S., Sze, S. M., Chand, U., & Tseng, T. Y. (2014). Overview of emerging nonvolatile memory technologies. *Nanoscale Research Letters*, *9*(1), 1–33. https://doi.org/10.1186/1556-276X-9-526

Hurlburt, G. F., Thiruvathukal, G. K., & Lee, M. R. (2017, November 1). The Graph Database: Jack of All Trades or Just Not SQL? *IT Professional*. IEEE Computer Society. <https://doi.org/10.1109/MITP.2017.4241475>

# APÉNDICES

Incluir toda la documentación utilizada durante el transcurso de la investigación. Ejemplos: entrevistas, cuestionarios, resultados de encuestas, formularios de encuestas, etc.