Rendimiento de base de datos relacionales y no relacional
*

María José Párraga[L00392984] and Kevin Norvey Paute Sánchez[L00080134]

Universidad de las Fuerzas Armadas ESPE
mjparraga2@espe.edu.ec
knpaute@espe.edu.ec

Resumen

Las bases de datos son una parte esencial de cualquier producto de software. El rendimiento y la eficiencia de la base de datos se han convertido en uno de los factores clave en el análisis del proceso de selección de tecnologías. Con el desarrollo de nuevos modelos de datos y tecnologías de almacenamiento, existe la necesidad de comparar motores de bases de datos relacionales y no relacionales, especialmente en el campo de la ingeniería de software. Este artículo presenta un análisis comparativo de bases de datos relacionales y no relacionales. Para los propósitos de este documento, se creó una base de datos simple sobre una farmacia. La aplicación soporta tres tipos de bases de datos: SQL en el gestor Sql Server, Mariadb y en No SQL MongoDB. Para cada base de datos se describió el modelo de datos aplicado. El objetivo del análisis fue comparar el rendimiento de estos gestores de bases de datos seleccionadas en el contexto de insertar medio millón de datos masivamente. Las pruebas de rendimiento mostraron en los gestores antes mencionados que Sql server es más rápido para insertar datos que Mongodb en nuestro estudio y a diferencia de estos dos en mariadb la inserción de datos fue más lenta. La aplicación de prueba es completamente funcional, sin embargo, la implementación resultó ser más desafiante para Mariadb al hacer uso de la herramienta Heidi Sql.

1. Introducción

El estudio de datos se torna como un factor clave para todo tipo de empresas. Actualmente, para que una empresa tenga éxito es de vital importancia contar con un gestor de base de datos que tenga la capacidad de procesar grandes cantidades de información de manera eficiente . Hoy en día las bases de datos son una parte esencial de cualquier producto de software. Con énfasis en el rendimiento de las aplicaciones, la eficiencia de la base de datos se convierte en uno de los factores clave a analizar en el proceso de selección de tecnología. Con el desarrollo de nuevos modelos de datos y tecnologías de almacenamiento, la necesidad de una comparación entre motores de bases de datos relacionales y no relacionales es especialmente evidente en el dominio de la ingeniería

*Kevin Paute (Estudiante de la carrera de ingeniería en Tecnologias de la Información de la universidad de la Fuerzas Armadas ESPE, Ecuador)
María José Párraga (Estudiante de la carrera de ingenieria en Tecnologias de la Información de la universidad de la Fuerzas Armadas ESPE, Ecuador)
Repositorio de github—https://github.com/majito02/ArticuloModelado.git

de software [2].

Las bases de datos relacionales existen desde hace muchos años y son la tecnología preferida para la mayoría de sus necesidades de bases de datos, incluyendo la capacidad de almacenamiento y recuperación de datos intensivos para el filtrado, la transformación de datos y la gestión. La recuperación de datos generalmente se realiza utilizando el lenguaje de consulta SQL. Las bases de datos relacionales o los sistemas de bases de datos tradicionales son generalmente eficientes a menos que la base de datos necesite procesar, actualizar o modificar y unir relaciones de tablas de bases de datos grandes. En los últimos años, ha habido mucho interés en la recuperación rápida de datos de los almacenes de datos debido a consultas estructuradas [3].

El sistema de bases de datos relacionales es muy efectivo en términos de robustez y escalabilidad, confiabilidad y requisitos de flexibilidad, pero cuando se trata de las demandas de las aplicaciones modernas que generan big data y, a menudo, datos no estructurados, las bases de datos relacionales fallan. El sistema de base de datos no relacional muestra una verdadera usabilidad en tal caso [4]. El presente artículo tiene como objetivo encontrar las características distintivas del rendimiento de Bases de Datos Relacionales y Orientada a Objetos de una determinada base de datos. La investigación comprende el estudio de varios gestores para analizar el rendimiento al insertar medio millón de datos en varias tablas al mismo tiempo.

2. Materiales y métodos

Existen numerosos métodos para ingresar información de forma masiva en una base de datos. Estos métodos pueden variar dentro de los pasos tomados para ingresar esta información y, por lo tanto, conducen a resultados de extracción completamente diferentes. En el estudio a continuación, cuando el objetivo de la investigación es obtener métodos de medición del desempeño, la elección de la metodología de investigación se limita a unas pocas posibilidades. Nuestro estudio comparativo se lo aplico dividiendo la información en las tablas de la base de datos farmacia. Primero se seleccionaron los gestores de base datos para realizar las pruebas para evaluar el rendimiento. El número de datos que se insertará en la tabla clientes es de 500.000 a diferencia de los productos que será de 1000.000, en factura se ingresaran 600.000. Para proveedor serán 100.000, categoría 100 y farmacéuticos 1000.

Nuestro escenario de pruebas para el estudio de base de datos relacionales constituye el uso de archivos csv. Para los NoSQL se puso en práctica la usabilidad del lenguaje de programación Python. La infraestructura donde se ejecutaran estas pruebas será en una computadora HP con procesador AMD Ryzen que tiene una memoria RAM de 12GB y un sistema operativo de 64bits. Además, como una buena práctica fue conveniente utilizar un repositorio de GitHub para almacenar los archivos. La base de datos a usar será Farmacia, contará con seis tablas relacionadas, en la 1 se puede observar el modelo entidad relación en Sql Server y Mariadb . La información en formato csv se generó con **Mockaroo** que es una herramienta que permite generar datos de forma aleatoria con los diferentes tipos de datos de acuerdo a las tablas de nuestra base de datos.

Los datos se cargaron primeramente en Sql Server, en un nuevo query se establece las estadísticas de tiempo para medir el rendimiento en este gestor utilizando el BULK INSERT. La sintaxis usada para cargar los datos csv se muestra en el Listing 1, se puede denotar que se define el nombre de la tabla seguido de la ubicación del archivo csv con su delimitador. Igualmente, el número de primeras filas e incluso los máximos errores que puede tener, cabe destacar que este

proceso se aplica para cada una de las tablas de la base de datos.

```
    set statistics time on
BULK INSERT
     cliente
        FROM 'C:\registros\cliente.csv'
        WITH (
            FIELDTERMINATOR = ',',
            ROWTERMINATOR = '\n',
            FIRSTROW = 2,
            MAXERRORS = 3
                        );
    set statistics time off
```

Listing 1: Sintaxis en Sql Server para insertar la información en formato csv

Luego para cargar los datos en Mariadb, en un nuevo query en Heidi SQL se establecen las especificaciones del archivo csv y la tabla de destino. La sintaxis usada para cargar los datos se muestra en el Listing 2 en donde se define el delimitador del archivo así como se establece que la primera fila debe ser ignorada por el hecho que contienen cabeceras las columnas. Es importante resaltar que este mismo procedimiento se aplica para todas las tablas.

```
LOAD DATA
    INFILE 'C:/xampp/dataset/CreacionRegistros/producto.csv'
    INTO TABLE producto
    FIELDS TERMINATED BY ';'
    IGNORE 1 ROWS;
```

Listing 2: Sintaxis en Mariadb para insertar la información en formato csv

En mongodb se realizó un algoritmo que permite crear dataset para insertar datos para obtener valores aleatorios de los datos del dataset y guardarlos en una lista por cada columna. En el Listing 3 se puede visualizar la sintaxis en Python que nos permite realizar las acciones ante mencionadas. Parar insertar los datos masivamente se usó otro algoritmo para obtener la información del archivo csv creado. La codificación implementada se puede ver en el Listing 4, se estableció una función para obtener los datos separados en diferentes diccionarios. Después, se instauró una función para generar una n cantidad de usuarios en una colección para luego ser insertada en mongodb y finalmente se imprime el tiempo de ejecución.

```
import pandas as pd
import random as r
data = pd.read_csv("cliente_antiguo.csv", sep=";")
c0 = []
c1 = []
c2 = []
c3 = []
c4 = []
c0 = data["id_cliente"]
c1 = data["nombre"]
c2 = data["apellido"]
c3 = data["telefono"]
c4 = data["direccion"]

# Obtener valores aleatorios de los datos del dataset y los guardar en una lista
#por cada columna
diccionarios = []
for i in range(500000): # Insertar medio millon de clientes
    diccionarios.append({"id_cliente": c0[r.randint(0, 999)], "nombre": c1[r.randint(0, 999)],
    "apellido": c2[r.randint(0, 999)], "telefono": c3[r.randint(0, 999)], "direccion":
    c4[r.randint(0, 999)]})


# Crear un dataframe con los valores aleatorios
df = pd.DataFrame(diccionarios)

# Guardar el dataframe en un archivo csv
df.to_csv("cliente.csv", sep=";")
```

Listing 3: Sintaxis en Mongodb para crear un dataset con python y guardarlo en formato csv

```
import pandas as pd
import time
import pymongo

inicio=time.time()
# Cargar el dataframe
cli = pd.read_csv(
    "C:/dataset/CreacionRegistros/Procesamiento/csv/cliente.csv", sep=";")

# Crear una conexion con mongo db
client = pymongo.MongoClient("mongodb://localhost:27017/")
# Utilizar la base de datos farmacia
db = client.farmacia
# Crear colecciones
cliente = db.cliente

# Obtener los datos de cliente
nombres_cliente = cli["nombre"]
apellidos_cliente = cli["apellido"]
telefono = cli["telefono"]
direccion = cli["direccion"]
diccionario3 = []

conteoCliente = len(cli)
i = 1  # Contador

# Diccionario para guardar los datos de cliente
for i in range(conteoCliente):
    diccionario3.append(
        {"_id": i, "nombre": nombres_cliente[i], "apellido": apellidos_cliente[i], "telefono":
        telefono[i], "direccion": direccion[i]})

# Insertar los datos en las colecciones
cliente.insert_many(diccionario3)

#Tiempo de ejeucion
final = time.time()
print("\nTiempo de ejecucion: ", final - inicio)

# Terminar la conexion con mongo db
client.close()
```

Listing 4: Sintaxis en Mongodb para insertar los datos masivamente

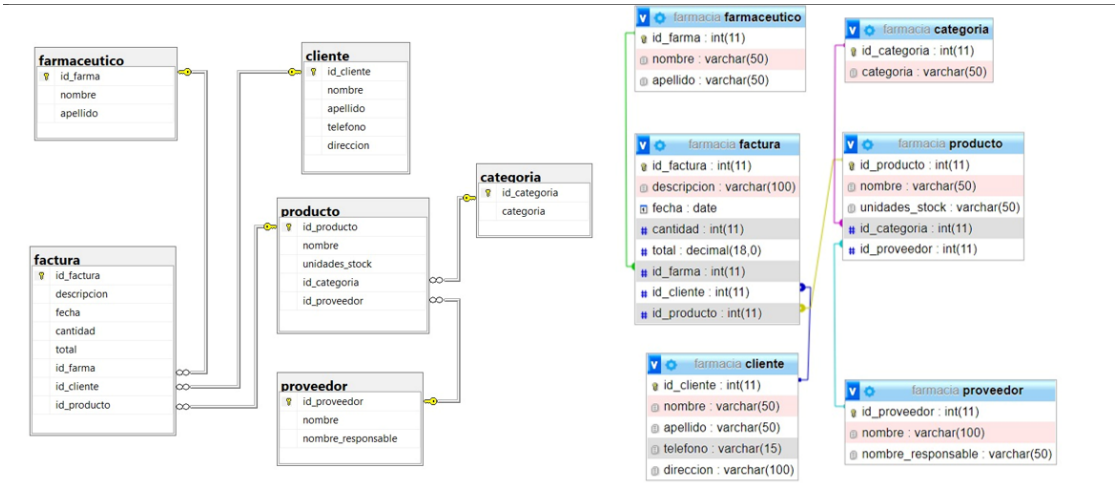

Figura 1: Modelo Entidad/Relación en Sql Server y Mariadb

3. Results and Analysis

Los sistemas de administración de bases de datos relacionales y no relacionales están diseñados con diferentes cualidades en mente. Las bases de datos relacionales se pueden caracterizar por los principios ACID, donde la atomicidad, la coherencia, el aislamiento y la durabilidad deben implementarse mediante cualquier motor de base de datos. En el otro extremo, se encuentran las bases de datos no relacionales, cuya semántica describe propiedades como la disponibilidad, el estado y la consistencia eventual, que son aspectos clave representados por NoSQL.

Para poder determinar qué gestor fue más rápido al insertar datos, se inició con MSSQL. En la Figura 2 al ejecutar la sentencia para insertar datos masivamente se obtuvo varios resultados en milisegundos, no obstante, estos valores pertenecen a cada tabla de la base de datos. La Figura 3, se visualiza directamente que al ejecutar las sentencias, hubo demora al procesar los datos, incluso la herramienta Heidi colapsó cuando inició con inserción de productos y facturas, las cuales tienen el mayor número de registros. En última instancia, la Figura 4, muestra que al ser ejecutado mediante Visual Studio Code el proceso de inserción de datos fue exitoso; sin embargo, se destaca que hubo errores de inserción por la cantidad de datos en la colección de productos, teniendo que volver a optimizar código y los datos.

En este aspecto, se determina que la mejor opción para insertar datos de manera masiva es el uso del gestor MSSQL al utilizar la declaración "Bulk Insert", dando un tiempo total de tan solo 8,519 segundos, si se compara con MariaDB y MongoDB, fue el más eficiente porque MariaDB demoró 8 minutos y 31 segundos en insertar los datos, por último MongoDB al insertar con ayuda de Python demoró 41,74 segundos (ver Tabla 1). Por otro lado, se observa la Figura 5 que muestra la comparativa del Gestor que tuvo mejor rendimiento.

Por lo tanto, las comparaciones de rendimiento representadas por las bases de datos relacionales y no relacional son sensatas y parecen justificadas. La aplicación de cualquier motor de base de datos es posible y está asociada con varios beneficios. Un caso de uso de ejemplo podría ser un sitio web minorista grande que use múltiples cualidades de base de datos, factores que impulsan la elección del motor. Algunas de las cuestiones a considerar son la disponibilidad por el hecho de que se puede generar un costo dependiendo de los períodos de uso y el rendimiento,

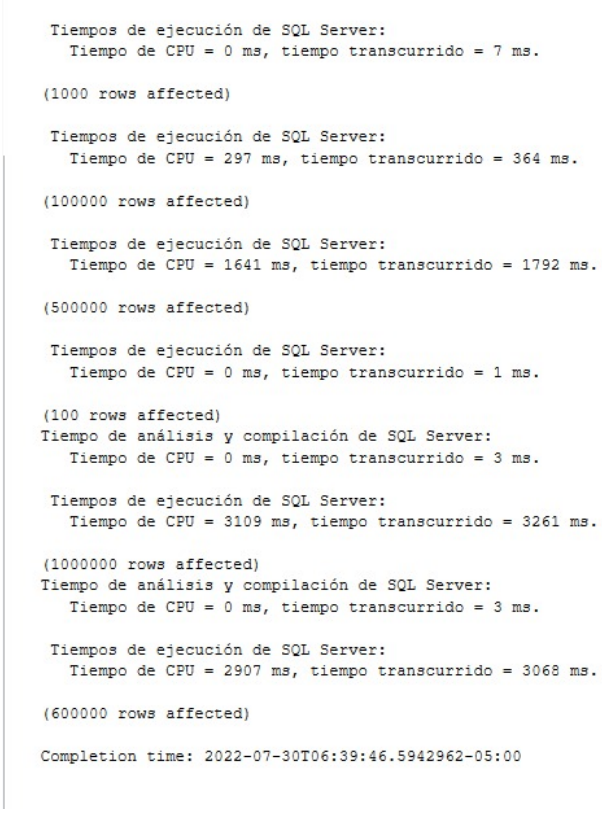

Figura 2: Resultados de tiempo de espera utilizando SQL Server en la herramienta SSMS.

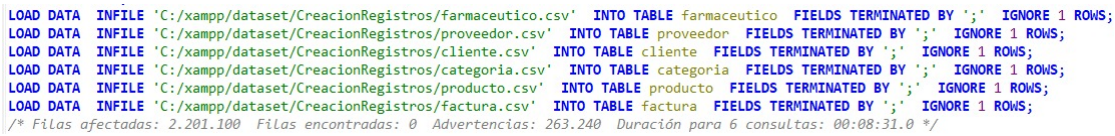

Figura 3: Resultado de tiempo de ejecución utilizando MariaDB en la herramienta Heidi

el cual es un factor clave para las empresas. Estos aspectos pueden cumplirse aplicando ambas bases de datos relacionales y no relacionales, mientras que el desempeño puede verse como un factor decisivo que influye en la decisión.

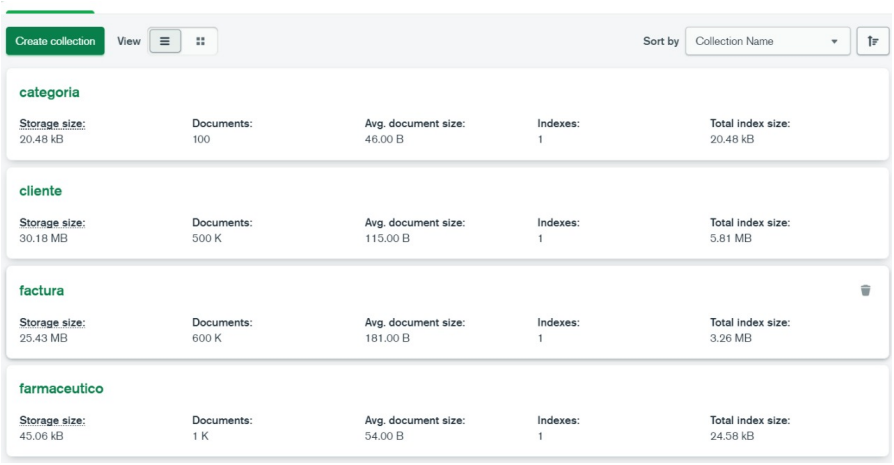

Figura 4: Resultado de tiempo de ejecución para MongoDB utilizando Python.

Cuadro 1: Tiempo de inserción de datos masivos

Tiempo de ejecución de cada Gestor de BDD				
Tabla	**N° Registros**	**SQL Server**	**MariaDB**	**MongoDB**
farmaceutico	1.000	0,007	8 min 31 s	41, 74 s
proveedor	100.000	0,364		
cliente	500.000	1,792		
categoria	100	0,001		
producto	1'000.000	3,264		
factura	600.000	3,091		
Total	2'201.100	8,519 s	8 min 31 s	41,74 s

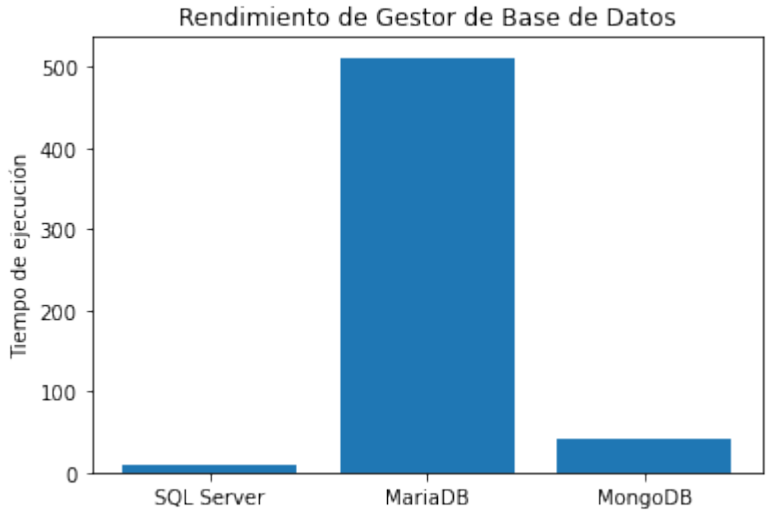

Figura 5: Gráfica de barras del rendimiento de los gestores.

4. Discusión

Los avances en la tecnología actual han aumentado la generación de datos, lo que ha creado la necesidad de procesar rápidamente grandes cargas de datos, lo que lleva a observaciones del rendimiento del sistema de base de datos. Se convierte en un punto clave en su evaluación de la calidad, mantener el rendimiento de DBMS es una prioridad principal para las organizaciones basadas en datos de hoy. Según [5] se estima que los volúmenes de datos crecerán exponencialmente para 2030, mientras que la capacidad de la CPU solo crecerá exponencialmente, lo que requerirá soluciones para mejorar el rendimiento de la base de datos.

Para cumplir con los requisitos enumerados anteriormente, los ingenieros de software buscan sistemas de bases de datos que brinden el mejor rendimiento.Existe una clara necesidad de implementar sistemas de bases de datos de mayor rendimiento en el software informático. Esto crea la necesidad de comparaciones de eficiencia entre varios motores de administración de bases de datos. Con el advenimiento de nuevas tecnologías en el campo de las bases de datos, como las bases de datos no relacionales, se hace evidente la necesidad de compararlas. Por lo tanto, el enfoque de este estudio es medir la eficiencia cuantitativa de los dos motores de bases de datos relacionales y no relacionales más populares, que satisfarían las necesidades de comparación de rendimiento de bases de datos descritas anteriormente.

5. Conclusión

A través de este estudio, se determinó que la gestión de bases de datos requiere una clasificación adecuada de diferentes sistemas de administración de base de datos. La división básica de los tipos de DBMS se puede basar en el modelo de datos relacionales, que agrupa los objetos de interés en conjuntos relacionales y no relacionales. Por lo tanto, comparar diferentes sistemas de bases de datos en términos de rendimiento requiere mucho esfuerzo e investigación; en el proceso, se han descubierto diferentes formas de medir el rendimiento de las bases de datos para cumplir con el propósito de nuestra investigación que es determinar la productividad de los gestores de base de datos al momento de ingresar grandes cantidades de información.

Cabe destacar que la comparación del rendimiento entre bases de datos es un tema interesante y mutuamente beneficioso. La actividad de comparar bases de datos de diferentes familias y estructuras de modelos de datos ofrece un valor práctico adicional, ya que los resultados ayudan a visualizar las ventajas y desventajas tecnológicas de cada gestor. El objetivo de esta investigación fue medir y comparar el desempeño de dos sistemas de gestión de bases de datos. Se eligieron tres gestores dos relacionales y uno no relacional. Como resultado de la revisión de los sistemas de base de datos SQL y NoSQL más populares, se eligieron Mariadb, Sql Server y Mongo DB como ejemplos en nuestro estudio.

Referencias

[1] Jhonatan Wilson Durán Cazar and Eduardo Javier Tandazo Gaona. Estudio del rendimiento de una base de datos columnar en el análisis de datos. B.S. thesis, Quito: UCE, 2018.

[2] Nishtha Jatana, Sahil Puri, Mehak Ahuja, Ishita Kathuria, and Dishant Gosain. A survey and comparison of relational and non-relational database. *International Journal of Engineering Research & Technology*, 1(6):1–5, 2012.

[3] Navneet Kumar Kashyap, Binay Kumar Pandey, HL Mandoria, and Ashok Kumar. A review of leading databases: Relational & non-relational database. *i-Manager's Journal on Information Technology*, 5(2):34, 2016.

[4] Kamil Kolonko. Performance comparison of the most popular relational and non-relational database management systems, 2018.

[5] Vanessa Valverde, Narcisa Portalanza, Paulina Mora, et al. Análisis descriptivo de base de datos relacional y no relacional. *Revista Atlante: Cuadernos de Educación y Desarrollo*, 3, 2019.