

ARQUITECTURA FRAGMENTADA DE MONGODB PARA SISTEMA INTEGRADO DE NOTAS UNIVERSITARIO

Jairo Salas, Camilo Ayala, Alejandro Melo
Depto. Ingeniería de sistemas y computación e industrial
Facultad de ingeniería
Bogotá, Colombia

jsalasm@unal.edu.co, cecheverry@unal.edu.co, dmelo@unal.edu.co

ABSTRACT - In today's dynamic educational landscape, efficient management of student records, academic schedules, and performance evaluation is crucial for colleges. This paper presents a comprehensive solution for improving the operational and analytical capabilities of a college management system through the integration of MongoDB's sharded architecture.

The OLTP (Online Transaction Processing) component manages real-time interactions, in this case with students, teachers, enrollments, subjects, campuses, classrooms, and scheduling information. Leveraging the scalability and flexibility of MongoDB, deploying the OLTP system using Docker containers with a router, config, and two nodes, ensuring high availability and fault tolerance. To enable advanced analytics and decision-making, developing a Python-based ETL (Extract, Transform, Load) process that transforms OLTP data into an OLAP (Online Analytical Processing) data warehouse. The OLAP system utilizes key dimensions such as students, time, subjects, teachers, and classrooms, with gradings and enrollments as core facts. This data-driven approach empowers college administrators with deeper insights into student performance, course popularity, and resource utilization.

Through this paper, the implementation in MongoDB sharded will be compared to a past MySQL version.

KEYWORDS: OLTP, OLAP, NoSQL, SQL, MongoDB, Sharding, BI, PowerBI.

1. INTRODUCCIÓN

La eficiente gestión de registros estudiantiles, horarios académicos y evaluación del rendimiento constituye un desafío fundamental en el entorno educativo actual. En este trabajo se presenta una solución integral para mejorar las capacidades operativas y analíticas de un sistema de gestión universitaria, a través de la integración de la arquitectura fragmentada de MongoDB.

El componente OLTP (Procesamiento de Transacciones en Línea) de este sistema se encarga de gestionar las interacciones en tiempo real entre estudiantes, profesores, matrículas, asignaturas, campus, aulas e información de programación académica. Con el fin de aprovechar la escalabilidad y flexibilidad que ofrece MongoDB, se implementa el sistema OLTP utilizando contenedores Docker con un enrutador, configuración y dos nodos, garantizando así alta disponibilidad y tolerancia a fallos.

Para facilitar el análisis avanzado y la toma de decisiones, se emplea un proceso ETL (Extracción, Transformación, Carga) desarrollado en Python, el cual transforma los datos del sistema OLTP en un almacén de datos OLAP (Procesamiento Analítico en Línea). Este sistema OLAP utiliza dimensiones clave, tales como estudiantes, tiempo, asignaturas, profesores y aulas, junto con las colecciones de hechos de calificaciones y matrículas. Gracias a este enfoque basado en datos, los administradores universitarios obtienen una visión más profunda del rendimiento estudiantil, la popularidad de los cursos y la utilización de recursos.

En comparación con una solución basada en SQL y MySQL, la integración de la arquitectura fragmentada de MongoDB ofrece beneficios significativos en términos de escalabilidad y flexibilidad. La arquitectura modular y escalable de MongoDB permite un crecimiento fluido del sistema a medida que aumenta la población estudiantil y el volumen de datos. Así, esta solución se presenta como una alternativa poderosa para la gestión universitaria, permitiendo tomar decisiones basadas en datos y mejorar la eficiencia operativa en general.

2. MARCO TEÓRICO

I. OLTP

El OLTP o OnLine Transaction Processing es un componente que permite el procesamiento de datos rápido y preciso, permitiendo la ejecución en tiempo real de un gran número de transacciones [1]. Estas transacciones

dentro de una base de datos incluye consultas, cambios, inserciones y eliminaciones de datos de una base de datos, o dicho de otra manera, abarca toda interacción que tienen los usuarios de un sistema computacional con acceso a datos.

Este componente se caracteriza por tener una fuerte normalización en la estructura de los datos, generalmente junto con unas garantías ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad) de los datos en aquellos entornos donde se emplean sistemas de gestión de bases de datos relacionales (RDBMS). En este contexto, las transacciones se realizan de forma concurrente por múltiples usuarios deben ser eficientes y consistentes y mantener una alta disponibilidad para asegurar la integridad de los datos suministrados.

II. ETL

Es un proceso clave que se encarga de extraer datos de múltiples fuentes diferentes y estandarizarla, luego la transforma para generar conjuntos de datos más valiosos y finalmente carga todo en alguna estructura dada que permite acceder rápidamente a los datos para generar estructuras de presentación y análisis más claras

III. MODELO DIMENSIONAL

A diferencia del modelo relacional usado en un paradigma SQL, el cual es un modelo convencional para dar estructura a los datos de una base de datos, el modelo dimensional consta de una tabla de hechos y varias tablas de dimensiones. La tabla de hechos contiene los valores numéricos relacionados con un proceso empresarial mientras que la tabla de dimensiones contiene los datos que describen cada atributo presente en la tabla de hechos [2].

Este modelo se caracteriza por su enfoque en consultas, bajos tiempos de respuesta, baja normalización dado que se busca reducir el tiempo de consulta y en este sentido se requiere de una baja cantidad de operaciones join, por ejemplo, con el fin de no aumentar la complejidad de la consulta debido a los grandes volumen de datos que se manejan. Por último también presentan gran cantidad de datos provenientes de agregaciones.

IV. OLAP

El OLAP o OnLine Analytical Processing es un componente que permite realizar análisis multidimensional a altas velocidades en grandes volúmenes de datos [3] que pueden provenir de distintas fuentes de datos, los cuales se adecuan a la estructura existente por medio de una ETL.

Este componente permite realizar consultas complejas sobre datos agregados y resumidos, lo que permite obtener una vista panorámica de los datos con el fin de evidenciar tendencias que permitan una toma de decisiones informada por parte de las personas de negocio.

A diferencia del OLTP, que se enfoca en operaciones de escritura y actualización en tiempo real, el OLAP está orientado principalmente a operaciones de lectura y análisis, en consecuencia implica una gran carga en términos de consultas y análisis de datos complejos.

V. MYSQL

Es un sistema de gestión de bases de datos relacionales ampliamente utilizado en múltiples contextos; Es una opción popular y de código abierto que ofrece una combinación de rendimiento, confiabilidad y flexibilidad par cualquier organización, siguiendo un enfoque en múltiple tablas que estructuran los datos y que tiene relaciones entre sí para lograr interacciones más complejas. Es una herramienta con mucha historia en el mundo del desarrollo, por lo que cuenta con un montón de soporte e integración con diferentes sistemas además de poseer una comunidad activa y madura en su uso.

VI. MONGODB

Es una base de datos Not Only SQL enfocada en documentos altamente escalable y flexible que ha ganado gran relevancia en múltiples campos de implementación

A. ARQUITECTURA FRAGMENTADA

MongoDB tiene la posibilidad de generar una arquitectura fragmentada sobre múltiples servidores para generar un set de replicación que se divide, respalda y balancea los datos en múltiples

nodos, consta de un enrutador llamado Mongos que permite un punto de acceso único a toda la estructura, un nodo de configuración, que distribuye variables de entorno claves y configuraciones como control de acceso a todos los nodos, así como generar el cambio de nodo maestro cuando este no está disponible, y finalmente una cantidad indefinida de nodos en un arreglo de maestro esclavo que manejan todos los datos y operaciones de agregación, consulta, actualización y eliminación.

VII. DOCKER

Es una plataforma de virtualización que emplea contenedores como mecanismo principal, permite la creación de entornos aislados, portables, ligeros y eficientes para el empaquetamiento y despliegue de soluciones de software, es una herramienta extremadamente valiosa usada en entornos productivos para construir infraestructuras robustas compuestas de varios microservicios independientes que se comunican entre sí que tienen la capacidad de replicar a demanda según las cargas de trabajo.

En este aspecto docker se convierte en un pilar fundamental para el despliegue de una solución fragmentada de docker porque facilita aumentar la cantidad de nodos en el conjunto de replicación según haya demanda de estos, mantener un esquema de recuperación a fallos y administrar rápidamente el estado de cada contenedor independientemente

VIII. BUSINESS INTELLIGENCE

Son un conjunto de procesos, tecnologías y herramientas utilizadas para recopilar, analizar y presentar datos críticos con el fin de generar una toma de decisiones más efectiva, estratégica e informada, por lo tanto su principal objetivo es convertir datos crudos en información valiosa, e incluye procesos de extracción, análisis, modelado, distribución y presentación de información que puede tener diferentes niveles de complejidad según los requerimientos de la organización.

3. ARQUITECTURA Y DISEÑO

La arquitectura propuesta para el sistema de gestión universitaria se compone de varios componentes interrelacionados, diseñados para proporcionar una plataforma eficiente y escalable para el procesamiento de datos y la generación de información relevante. A continuación, se detallan los principales elementos de la arquitectura:

Base de datos OLTP: La capa de base de datos OLTP se encarga de almacenar y gestionar los datos transaccionales relacionados con estudiantes, profesores, matrículas, asignaturas, programas, campus y aulas. Esta base de datos se utiliza para el registro y seguimiento de las interacciones diarias con el sistema, como matriculaciones, calificaciones y cambios de horario.

Proceso ETL: Se ha desarrollado un proceso de Extracción, Transformación y Carga (ETL) en Python para extraer datos de la base de datos OLTP y transformarlos en un formato adecuado para su posterior carga en el almacén de datos OLAP. Este proceso incluye la limpieza y agregación de datos, así como la aplicación de reglas de negocio específicas.

Almacén de datos OLAP: El almacén de datos OLAP almacena datos estructurados de manera dimensional utilizando las dimensiones de estudiante, tiempo, asignatura, profesores y aulas. También incluye dos hechos principales: calificaciones y matrículas. Este almacén de datos proporciona una visión consolidada y optimizada para el análisis y la generación de informes.

Contenedores Docker: Para garantizar la portabilidad y escalabilidad de los componentes de la arquitectura, tanto la base de datos OLTP como el almacén de datos OLAP se han desplegado utilizando contenedores Docker. Cada componente se ejecuta en un contenedor independiente, lo que facilita la gestión de la infraestructura y permite una mayor flexibilidad en el escalado horizontal, se generan por tanto dos fragmentos de mongo cada cual con su enrutador, configuración y nodos.

Instancia de Power BI: Para visualizar y representar la información almacenada en el almacén de datos OLAP, se utiliza una instancia de Power BI. Esta herramienta de visualización de datos permite crear paneles interactivos, informes y gráficos que facilitan

la comprensión y el análisis de los datos por parte de los usuarios finales.

4. ESCENARIO DE COMPARACIÓN SOLUCIÓN MySQL Y MongoDB

Se propone realizar una comparativa entre las soluciones desarrolladas utilizando los motores de bases de datos MySQL y MongoDB. El objetivo es evaluar el desempeño de ambas bases de datos al ejecutar una consulta que busca satisfacer indicadores clave definidos en el proceso de Business Intelligence. El escenario de prueba consiste en dos máquinas virtuales desplegadas en distintas regiones utilizando Google Cloud Platform. Cada máquina virtual cuenta con 4 núcleos de CPU, 16 GB de RAM y 30 GB de almacenamiento, utilizando el tipo e2-standard-4.

Tanto MySQL como MongoDB se ejecutan dentro de contenedores de Docker. En el caso de MySQL, se utiliza una única instancia que almacena todos los datos. Por otro lado, MongoDB utiliza una arquitectura fragmentada con dos instancias, como se describió en la sección de Arquitectura y Diseño. La comparativa se basa en el tiempo de consulta, el uso de memoria y el uso de CPU. Para medir estos indicadores, se utilizan el *performance_schema* de MySQL, el *executionStats* de MongoDB y los datos de observabilidad proporcionados por el proveedor de la nube mencionado.

Los indicadores específicos que se evalúan en la comparativa son los siguientes: Créditos inscritos por un estudiante en un semestre académico, Promedio de créditos inscritos por un estudiante durante su vida universitaria. Mediante la comparación de estos indicadores en ambos motores de bases de datos, se busca determinar cuál de las soluciones proporciona un mejor desempeño y eficiencia en términos de tiempo de consulta y recursos utilizados. Esto permitirá tomar decisiones informadas sobre la elección del motor de base de datos más adecuado para el caso de uso en particular.

5. RESULTADOS

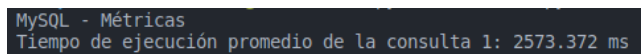
En la comparativa realizada se tienen en cuenta principalmente las métricas de utilización de CPU, memoria RAM y tiempo promedio en la ejecución de las consultas o agregaciones, para ello se hizo uso de un script hecho en el lenguaje de programación Python sobre el cual se realiza la ejecución de las consultas o agregaciones de los indicadores mencionados en la Sección Escenario de comparación solución MySQL y MongoDB, estas se ejecutan en 10 ocasiones con el fin de obtener un valor promedio en cuanto al tiempo de ejecución.

Las métricas provienen, en el caso de MySQL de la librería *time*, la cual es nativa de Python, sin embargo, cabe mencionar que también se puede considerar el uso de la base de datos de *performance_schema* la cual hay que habilitar dentro de la configuración del RDBMS en cuestión y más concretamente de la tabla *events_statements_summary_by_digest*, la cual lleva un registro histórico de las métricas allí almacenadas las cuales se actualizan cada vez que se realiza una consulta.

Por otra parte, para el caso de MongoDB las métricas se obtienen a partir de los planes de ejecución, concretamente del *executionStats*, el cual se ejecuta junto con la agregación, esto con el fin de obtener datos precisos sobre la ejecución de esta y evitar manejar datos aproximados o estimados.

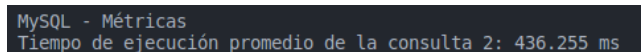
En cuanto al uso de CPU y memoria RAM, estas se obtienen por medio de las herramientas de observabilidad que ofrece Google Cloud Platform para las instancias del servicio de Compute Engine.

Una vez detallado el origen de los datos, se obtiene que en esta caso la solución de MySQL ofrece mejores resultados en comparación con la solución implementada con MongoDB, debido a que el promedio de ejecución del primer indicador es de 2573.37 ms contra 46772 ms, y para el segundo indicador es de 436.25 ms contra 31297 ms.



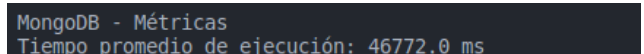
```
MySQL - Métricas
Tiempo de ejecución promedio de la consulta 1: 2573.372 ms
```

Figura 1. Tiempos de ejecución del primer indicador en MySQL.



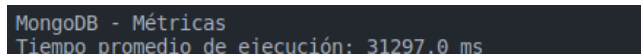
```
MySQL - Métricas
Tiempo de ejecución promedio de la consulta 2: 436.255 ms
```

Figura 2. Tiempos de ejecución del segundo indicador en MySQL.



```
MongoDB - Métricas
Tiempo promedio de ejecución: 46772.0 ms
```

Figura 3. Tiempo de ejecución del primer indicador en MongoDB



```
MongoDB - Métricas
Tiempo promedio de ejecución: 31297.0 ms
```

Figura 4. Tiempo de ejecución del segundo indicador en MongoDB.

En cuanto al uso de recursos computacionales por parte de ambas soluciones, se observa de forma considerable un mayor uso de recursos por parte la instancia de MongoDB, el cual en promedio usó un 27% de CPU y 20% de memoria RAM mientras que la instancia de MySQL tuvo un uso menor de CPU

llegando incluso a tener un pico temporal de 5.56% mientras que el uso de memoria RAM se mantuvo en un intervalo de 6% a 6.5% de uso, como se puede observar a continuación:

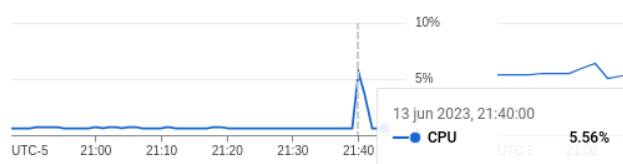


Figura 5. Utilización de CPU en la ejecución de los indicadores usando MySQL.



Figura 6. Utilización de memoria RAM en la ejecución de los indicadores usando MySQL.



Figura 7. Utilización de CPU en la ejecución de los indicadores usando MongoDB.



Figura 8. Utilización de memoria RAM en la ejecución de los indicadores usando MongoDB.

Cabe destacar que MongoDB, como se mencionó anteriormente, es una base de datos NoSQL orientada a documentos lo que permite tener una estructura de los datos más complejas en comparación con MySQL. Sin embargo, debido a este enfoque este tipo de bases de datos no están optimizadas para operaciones de JOIN. Por lo tanto, el modelo de datos planteado desempeña un papel fundamental en el rendimiento de las consultas, especialmente en términos del volumen de datos y la optimización de consultas, el modelo de datos se pueden encontrar en los Anexos.

También se debe considerar que la arquitectura implementada en el caso de MongoDB es una arquitectura fragmentada en dos nodos, de los cuales uno solo soporta operaciones de escritura, por lo que al momento de realizar las agregaciones necesarias para la obtención de los indicadores debe realizar consultas en ambos nodos, ya que ambos contienen una parte del conjunto total de datos del que se compone el OLAP. Por otro lado, la arquitectura implementada en el caso de MySQL no cuenta con replicación de ningún tipo, por lo que todos los datos de la OLAP se encuentran dentro de la misma instancia de la base de datos.

Otro punto crucial es el conjunto de datos utilizado, el cual ha sido generado de manera aleatoria mediante diversos scripts de Python. Como resultado, es importante tener en cuenta que los resultados obtenidos pueden variar lo cual repercute directamente en el desempeño de las consultas.

6. CONCLUSIONES

En este paper, se ha realizado una comparación exhaustiva entre una versión NoSQL y una versión SQL de un sistema de gestión universitaria, utilizando MongoDB y MySQL respectivamente, buscando evaluar la eficiencia y escalabilidad de ambas soluciones, considerando su estructura y las operaciones que se llevan a cabo en el contexto de la gestión universitaria.

Se encontró que la implementación NoSQL utilizando MongoDB ofrece ventajas significativas en términos de flexibilidad y rendimiento en comparación con la solución SQL basada en MySQL. MongoDB, al ser una base de datos NoSQL, permite una estructura flexible y no requiere un esquema predefinido. Esto resulta especialmente beneficioso en un entorno universitario donde los datos pueden variar y evolucionar con el tiempo. Además, MongoDB ofrece capacidades avanzadas de agregación y consultas flexibles que permiten un procesamiento eficiente de los datos. Estas funcionalidades son especialmente útiles en escenarios de alto rendimiento y para realizar operaciones complejas como la agregación de datos, que son frecuentes en el contexto de la gestión universitaria.

Sin embargo, a pesar de la eficiencia y flexibilidad de MongoDB, se observó que MySQL con su enfoque SQL y optimización en álgebra relacional y otras operaciones que consumen más tiempo, demostró ser más eficiente en términos de rendimiento para ciertas consultas y operaciones específicas. La fragmentación de datos en MongoDB juega un papel fundamental en la implementación realizada,

implicando que los datos se distribuyen en la cantidad de nodos disponibles y requiere de una búsqueda previa de todos en cada uno de estos

En resumen, este estudio demuestra que tanto MongoDB como MySQL tienen sus ventajas y desventajas en el contexto de un sistema de gestión universitaria. MongoDB ofrece flexibilidad y escalabilidad, especialmente para muchos datos que necesitan tolerancia a fallos y consultas flexibles. Por otro lado, MySQL con su enfoque SQL y su capacidad para realizar operaciones más eficientes en ciertos casos específicos muestra ser más adecuado en términos de rendimiento. La elección entre MongoDB y MySQL dependerá de las necesidades específicas de la institución educativa. Si la flexibilidad y el rendimiento en operaciones complejas son prioritarios, MongoDB es una opción sólida. Sin embargo, si el rendimiento en consultas específicas es fundamental, MySQL puede ser la mejor opción. En última instancia, la decisión debe basarse en un análisis exhaustivo de los requisitos y objetivos del sistema de gestión universitaria.

7. REFERENCIAS

- [1] Martin Ekuan, «Procesamiento de transacciones en línea (OLTP) - Azure Architecture Center», *Microsoft Learn*, 18 de febrero de 2023. <https://learn.microsoft.com/es-es/azure/architecture/data-guide/relational-data/online-transaction-processing>
- [2] «¿Qué es el OLAP? - Explicación del procesamiento analítico en línea - AWS», *Amazon Web Services, Inc.* <https://aws.amazon.com/es/what-is/olap/>
- [3] «What is OLAP? | IBM». <https://www.ibm.com/topics/olap>
- [4] MongoDB, "MongoDB Documentation," MongoDB, 2022. [Online]. Available: <https://docs.mongodb.com/>. [Accessed: June 12, 2023].
- [5] Oracle Corporation, "MySQL Documentation," MySQL, 2022. [Online]. Available: <https://dev.mysql.com/doc/>.
- [6] Microsoft, "Power BI Documentation," Microsoft Power BI, 2022. [Online]. Available: <https://docs.microsoft.com/power-bi/>.
- [7] Docker, "Docker Documentation," Docker, 2022. [Online]. Available: <https://docs.docker.com/>.