Análisis comparativo de base de datos relacionales y no relacionales en la inserción masiva de datos*

Anthony Goyes[1] and Génesis Heredia[1]

Universidad de las Fuerzas Armadas ESPE
{amgoyes,gbheredia}@espe.edu.ec

Resumen Actualmente, el procesamiento de datos masivos en bases de datos relacionales y no relacionales viene dado por el comportamiento de los datos durante dicho proceso, el presente artículo consiste en determinar el rendimiento de una base de datos al realizar un procesamiento de datos almacenado. Los motores de bases de datos que serán utilizados son tres: SQL Server 2019 y Postgres para las bases de datos relacionales y MongoDB para la base de datos no relacional. Este procedimiento se evaluaó en dos equipos con diferentes características con la finalidad de obtener datos conclusivos en relación a los requisitos de cada equipo lo que permitió determinar cual es el motor de base de datos más efectivo en el escenario de insertar registros de manera masiva en una base de datos con tablas relacionadas. Mediante la recopilación de datos en el timpo de ejecución se logró concluir que el motor de SQL Server presenta el peor rendimiento en comparación con MongoDB y Postgres que tienen un rendimiento superior.

Keywords: SQL Server · MongoDB· Postgres · Rendimiento de motores de base de datos · SQL · NoSQL.

1. Introducción

Cada día se generan enormes cantidades de datos y realizar un análisis útil de los mismos es una tarea que requiere muchos recursos. Encontrar formas eficientes de manejarlos es un área de investigación importante dentro de la informática, especialmente en sistemas de gestión de bases de datos (SGBD) [1]. Nos proponemos investigar los problemas relacionados con el manejo de grandes volúmenes de datos generados de manera aleatoria, ya sea por medio de variables cambiantes por ciclo o funciones para obtener valores aleatorios. La inserción de big data es un importante cuello de botella de rendimiento en los sistemas intensivos de datos y en el proceso de análisis de datos [6].

Las bases de datos tradicionales se basan en el modelo relacional para almacenamiento de datos. Recibieron el nombre de bases de datos SQL después de que el lenguaje de consulta se usara para definir, consultar, modificar y controlar los datos en una base de datos relacional [9]. Sin embargo, en los últimos

* Goyes A. y Heredia G.

años las bases de datos no relacionales conocidas como bases de datos NoSQL, han sido muy apreciadas. Con el creciente uso de Internet y la disponibilidad de almacenamiento de almacenamiento barato, se crean y almacenan cantidades masivas de datos estructurados, emiestructurados y no estructurados por una variedad de aplicaciones. Por lo general, estos datos a gran escala se conocidos como big data [10].

El procesamiento de grandes cantidades de datos requiere máquinas rápidas, esquemas de bases de datos flexibles y arquitecturas distribuidas que no caben en las bases de datos relacionales. Las bases de datos NoSQL afirman que proporcionan un fácil acceso, alta velocidad y capacidades de desarrollo para trabajar con grandes datos. Por otro lado, hay muchas opciones propuestas como bases de datos NoSQL comerciales y de código abierto [4]. La variedad de opciones disponibles, tanto para bases de datos relacionales como no relaciones, nos lleva a la cuestión de conocer la diferencia entre ambos paradigmas, estructurado y no estructurado, seleccionando SQL Server y Postgres en relación a bases de datos SQL; y MongoDB como base de datos NoSQL. Se comparará el tiempo de procesamiento para la inserción de datos masivos, un total de medio millón de registros, en el esquema de base de datos propuesto por Microsoft denominado "School Sample Database".

2. Trabajos relacionados

Anteriormente, otras investigaciones han demostrado que el multihilo puede mejorar el rendimiento de la inserción en la base de datos. Esto ha marcado la tendencia de utilizar el paralelismo a nivel de hilo paralelismo y escalabilidad de rendimiento en el desarrollo de software moderno [7]. Además, los artículos [8] y [3] investigan el rendimiento de la inserción en general utilizando técnicas de técnicas de computación paralela y multihilo para lograr la mejor velocidad de inserción posible. Un estudio demostró que MongoDB supera significativamente a PostgreSQL y MySQL en la ejecución de transacciones de inserción con menos de 30 consultas de inserción [2].

DeWitt y Gray [5] muestran que el procesamiento paralelo es una forma barata y rápida de ganar significativamente en rendimiento en los sistemas de bases de datos. Las técnicas de software, como la partición de datos datos, el flujo de datos y el paralelismo intra-operador son necesarias para tener una fácil migración al procesamiento paralelo. La disponibilidad de procesadores rápidos y paquetes de discos baratos es una plataforma ideal para los sistemas de bases de datos paralelas.

3. Metodología

3.1. Selección de base de datos

Para el proceso de selección de la base de datos, se eligió la base de datos School que cuenta con algunas tablas y tomaremos 4 tablas para poder realizar las respectivas pruebas, las tablas se denominan: course, departament, courseInstructor y person. Cada una de las tablas cuentan con diferentes campos que permitirán insertar los datos acorde a los campos requeridos. A su vez, se realizará el diagrama entidad-relación en donde se visualizará en la Figura 1 la relación existente en cada una de las tablas permitiendo así poder considerar en base a la base de datos seleccionada, cual es el método de inserción de datos masiva más efectivo en base al tiempo de ejecución en cada motor de base de datos.

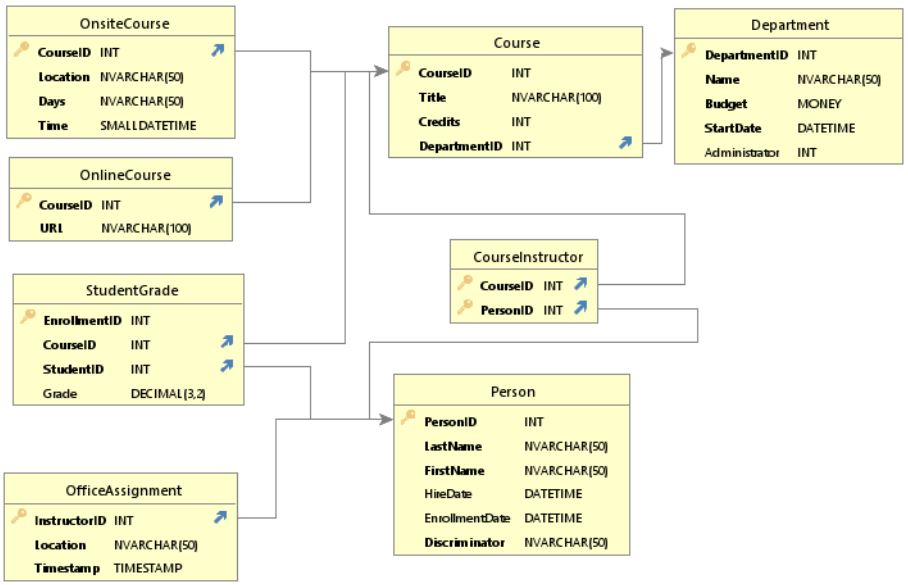

Figura 1. Modelo entidad relación de la base de datos propuesta por Microsoft denominada "School Sample Database".

3.2. Sistemas gestores de bases de datos relaciones y no relaciones

Las bases de datos relaciones son también conocidas como bases de datos SQL, por otro lado, las no relaciones son NoSQL. Se trabajará haciendo referencia a estos paradigmas con su segunda sintaxis (SQL y NoSQL) en adelante. Los

sistemas gestores de bases de datos (SGBD) utilizados para realizar la comparativa de rendimiento en la operación de inserción fueron Postgres y SQL server para el paradigma SQL y MongoDB para NoSQL. Además, como ya se dió a conocer el esquema de la base de datos, se recalca en la selección de únicamente cuatro tablas las cuáles son para el curso, departamento, intructor del curso y finalmente la persona o instructor a cargo. Adicionalmente, todos los recursos, tanto los archivos de definición y manipulación de datos, que fueron generados para la presente investigación, los recursos se encuentra evidenciados en el siguiente repositorio: Repositorio de evidencias

En la Figura 2 se puede observar el escenario propuesto de forma clara y precisa, mostrando; se visualiza los sistemas gestores de base de datos tanto para SQL como NoSQL. Además se logra transmitir que la relación y el aporte de la presente investigación recae en el análisis de tiempos de ejecución para ingresar registros en la base de datos propuestas. Finalmente, se enlistan las cuatro tablas seleccionadas para realizar la operación de inserción de datos masivos.

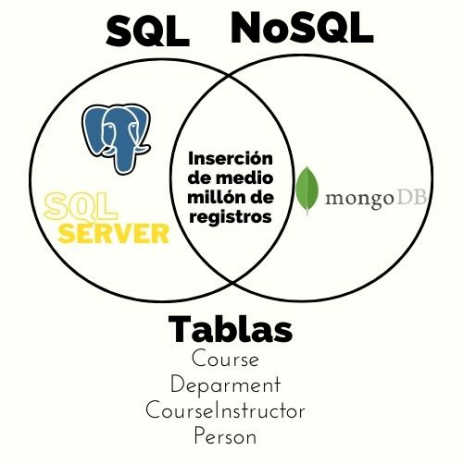

Figura 2. Contextualización de los sistemas gestores de bases de datos y tablas seleccionadas para el escenario propuesto

4. Prueba y Análisis de Resultados

Para la etapa de pruebas, en primera instancia se prepara el entorno de trabajo tomando en cuenta la temperatura ambiente y del computador (22 y 60 grados respectivamente), el estrés o la capacidad de la memoria de acceso aleatorio (RAM) disponible, la capacidad del disco de memoria y procesador. Los equipos a utilizar para la fase de pruebas son ordenadores portátiles y cuentan con diversas características y especificaciones que podemos visualizar en el

Cuadro 1, se debe tomar en cuenta que de todas las especificaciones la más importante es el procesador, específicamente, la cantidad de núcleos físicos e hilos ya que permitirán realizar mas o menos acciones afectando significativamente el resultado de la operación.

Cuadro 1. Especificaciones de los equipos de prueba

Especificaciones Equipos de Prueba					
# Equipo	**Procesador**	**Frecuencia Procesador**	**RAM**	**Almacenamiento**	**Núcleos físicos:virtuales**
Equipo 1 (Anthony Goyes)	Intel (R) Core (TM) i5-5200U	2.20 GHz	6 GB	698 GB HDD	2 :4
Equipo 2 (Genesis Heredia)	Intel(R) Core(TM) i5-1035G1	1.19 GHz	8 GB	236 GB SSD	4:8

4.1. SQL Server

Una vez seleccionadas las tablas a las cuáles se les realizará la operación de inserción, se debe tener presente la estructura que maneja cada una de ellas. Mediante una secuencia de comandos (script) se realiza la inserción haciendo el llamado a cada columna de la tabla en cuestión; mediante funciones propias del motor de base de datos SQL Server (por ejemplo la función random) se genera información única, esto fue realizado en un bucle que se repetirá n veces, donde n es la cantidad de registros que se agregarán a la base de datos (medio millón de registros). Además, se usó un iterador para generar variaciones en los datos a insertar. El Cuadro 2 muestra el resultado obtenido de los tiempos de ejecución en la inserción de medio millón de registros en el motor de base de datos SQL Server.

Cuadro 2. Tiempo de ejecución en el sistema gestor SQL Server con la herramienta Microsoft SQL Server Managment Studio SSMS

Tiempo de ejecución por tablas en el sistema gestor SQL Server con SSMS				
# Equipo	**Curso**	**Departamento**	**Instructor**	**Persona**
Equipo 1 (Anthony Goyes)	61.33 minutos	81.12 minutos	52.55 minutos	168.47 minutos
Equipo 2 (Génesis Heredia)	40.29 minutos	68.10 minutos	41.47 minutos	138.37 minutos

La información anterior fue recolectada y procesada para posteriormente ser graficada con la finalidad de obtener conclusiones en base a información estadística (la información gráfica logra ser más representativa en comparación a la

presentada en el Cuadro 2). En base a la Figura 3 se logra observar que existe una constante diferencia entre los resultados obtenidos entre el Equipo1 y Equipo2 en relación a los resultados del tiempo de procesamiento, siendo el Equipo2 quién tiene menores tiempos en cada una de las pruebas realizadas debido a que el motor SQL Server trabaja fuertemente con la capacidad del procesador y los núcleos del mismo; en el Cuadro 1 se logra observar que el Equpo2 duplica la cantidad de núcelos y esta es la razón de la diferencia en el tiempo de procesamiento obtenido. Además, se logra observar que la cantidad de columnas que tenga la tabla tiene un gran impacto, donde la tabla que relaciona la persona de instructor (CourseInstructor), la cuál tiene un total de dos campos, tiene el menor tiempo de procesamiento. Por otro lado, la tabla de persona (Person), cuya tabla es la que tiene más campos en su estructura definida, tiene el mayor tiempo de ejecución entre las tablas presentadas.

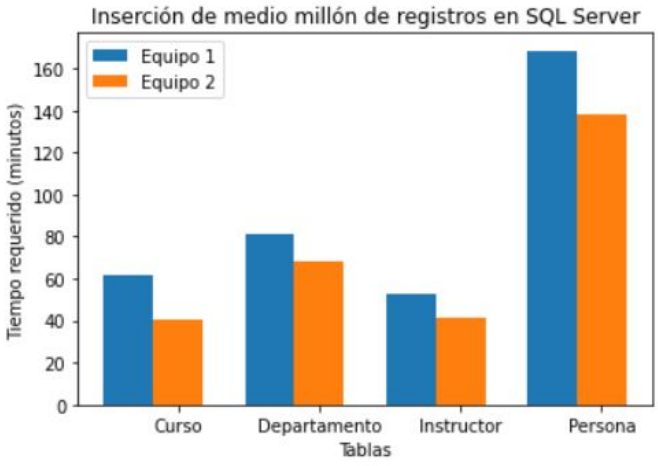

Figura 3. Resultado comparativo de los tiempos de ejecución para cada equipo al insertar medio millón de registros en SQL Server

4.2. Postgres

Para realizar el proceso anterior en Postgres se debe volver a estructurar la sentencia para la creación de la base de datos, debido a que la sintaxis que maneja el motor de SQL Server difiere mínimamente con Postgres no requirió mayor dificultad realizarlo. Una vez generado, se procedió a realizar la inserción del medio millón de registros mediante la sentencia propia de postgres **generate_series()**, a diferencia de la transacción en bucle para los **insert into** en SQL server, considerándose también como un script. El resultado de la acción anterior se puede ver reflejado en el Cuadro 3, donde por cada tabla del esquema de la

base de datos, se obtienen los tiempos de ejecución en cada uno de los equipos. Se puede observar que los tiempos de ejecución obtenidos son considerablemente inferiores en comparación con los resultados obtenidos en SQL Server.

Cuadro 3. Tiempo de ejecución en el sistema gestor PostgresSQL con su propia herramienta Postgres

Tiempo de ejecución en el sistema gestor PostgresSQL con Postgres				
# Equipo	**Curso**	**Departamento**	**Instructor**	**Persona**
Equipo 1 (Anthony Goyes)	20.19 minutos	25.41 minutos	18.36 minutos	38.27 minutos
Equipo 2 (Génesis Heredia)	15.25 minutos	22.39 minutos	11.57 minutos	27.42 minutos

En la Figura 4 se muestran los resultados obtenidos al insertar medio millón de registros por clase en la base de datos propuesta, pero de manera gráfica. Se observa nuevamente el mismo comportamiento donde el tiempo de ejecución depende estrictamente de la cantidad de columnas o campos que tenga la tabla afectada. Existe una diferencia de 130.20 minutos en el escenario con mayor coste, lo que implica una gran diferencia en el rendimiento. Además, existe un menor tiempo de ejecución por parte del Equipo2 dando a conocer que los núcleos físicos del procesador juegan un papel importante en la operación de inserción para Postgres.

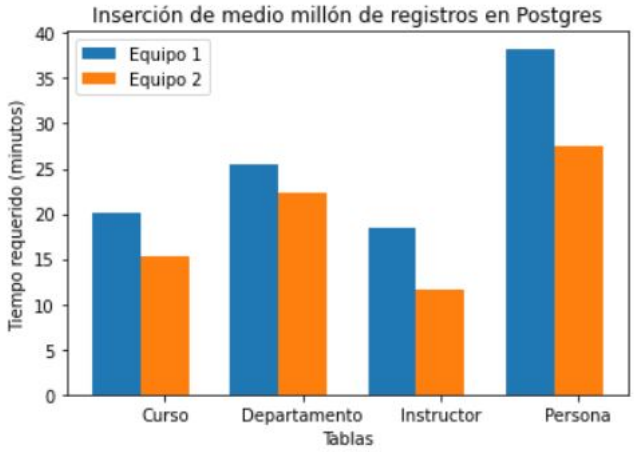

Figura 4. Resultado comparativo de los tiempos de ejecución para cada equipo al insertar medio millón de registros en Postgres

4.3. MongoDB

Para esta sección de pruebas no se adaptar sintaxis que genera la estructura de la base de datos, por otro lado, se necesita generar estas sentencias basadas en una notación de objeto JavaScript (JSON) basados en claves y valores que es característico de los motores de base de datos no estructurado, en este caso MongoDB. Cuando se hace referencia a JSON en MongoDB se la denomina como BJSON. A pesar que no se tiene un esquema de la base de datos explícito mediante llaves primarias y foráneas, propias de base de datos estructuradas, se generar identificadores que generarán las relaciones entre los cuatro documentos (de manera análogo el documento, en NoSQL, es equivalente a una tabla en SQL) con la finalidad de cumplir con las mismas condiciones de los escenarios anteriores.

El proceso de inserción fue realizado mediante una secuencia de comandos mediante Python, por lo que fue necesario realizar una petición de conexión hacia MongoDB para realizar la operación de insertar el medio millón de registros y mediante ciclos, respetando la estructura definida por documentos JSON. El Cuadro 4 muestra los resultados obtenidos de la inserción de medio millón de claves y valores en los distintos cuatro documentos donde se logra visualizar el mismo comportamiento de los dos escenarios anteriores, donde se ve una clara similitud con los resultados obtenidos en Postgres (véase Cuadro 3).

Cuadro 4. Tiempo de ejecución en el sistema gestor MongoDB con instrucciones en python

Tiempo de ejecución en el sistema gestor MongoDB con instrucciones en python				
# Equipo	**Curso**	**Departamento**	**Instructor**	**Persona**
Equipo 1 (Anthony Goyes)	25.59 minutos	34.11 minutos	21.32 minutos	52.17 minutos
Equipo 2 (Génesis Heredia)	18.31 minutos	26.19 minutos	15.48 minutos	46.47 minutos

La Figura 5 muestra el gráfico generado con la información del Cuadro 4 donde el tiempo de ejecución de la inserción de medio millón de registros en el motor de MongoDB se encuentra con menor equilibrio en sus resultados por documento, a diferencia de de los resultados con Postgres (véase Figura 4). Se observa que el rendimiento es ligeramente superior para el Equipo2 y también existe una tendencia a aumentar el tiempo de procesamiento en relación a la cantidad de claves por documento.

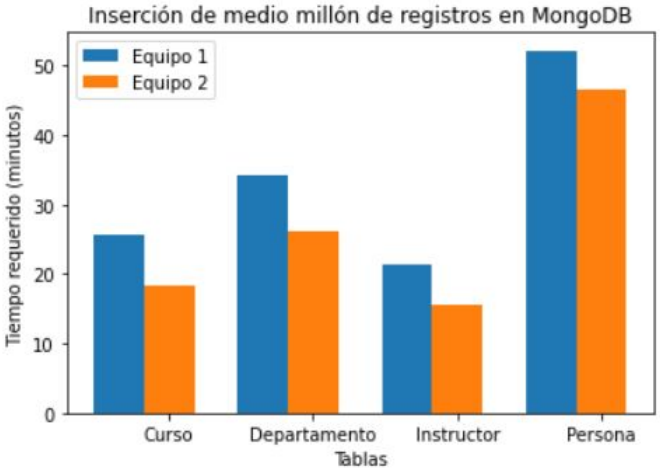

Figura 5. Resultado comparativo de los tiempos de ejecución para cada equipo al insertar medio millón de registros en MongoDB

4.4. Comparativa final

Los resultados anteriores permitieron establecer los principales resultados como la relación existente entre la cantidad de núcleos físicos en el tiempo de procesamiento y el impacto en el rendimiento de la operación de inserción en dependencia de la cantidad de columnas presentes en una tabla. Sin embargo, se plantea el realizar una comparación con los resultados promedio obtenidos anteriormente con la finalidad de comparar directamente los distintos motores de base de datos seleccionados.

La Figura 6 muestra el resultado comparativo por cada motor de base de datos, tanto SQL como NoSQL. Esta estadística fue realizada con los escenarios anteriores en los cuáles sus resultados fueron promediados debido a que el análisis individual ya fue previamente realizado. Se logra visualizar que existe una clara diferencia del tiempo de ejecución con el motor SQL Server en comparación con MongoDB y Postgres, considerando que SQL Server tiene el peor rendimiento para las condiciones establecidas. Por otro lado, tanto MongoDB como Postgres tienen un rendimiento similar, sin embargo, Postgres es ligeramente superior con un menor tiempo de ejecución en la inserción de medio millón de registros, esto puede deberse que la inserción de datos masivos realizada en MongoDB se realizó mediante una conexión hacia el motor por lo que requiere una operación extra.

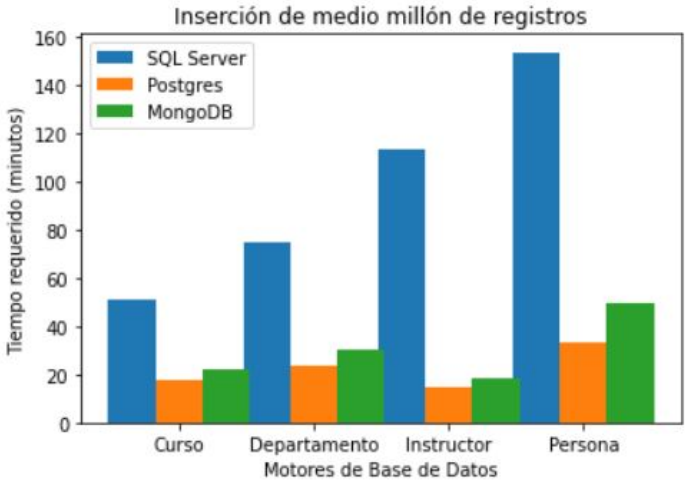

Figura 6. Comparativo de los tiempos de ejecución por cada motor de base datos

5. Conclusiones

Mediante la implementación y pruebas realizadas en cada uno de los motores de bases de datos, podemos deducir que, Postgres obtuvo una acogida favorable frente a las pruebas realizadas, demostrando que su proceso de inserción de datos de forma masiva responde de forma óptima y favorable en base a los requerimientos definidos. Por otro lado, MongoDB también presenta un rendimiento óptimo, demostrando que los dos motores tienen un buen desempeño para poder realizar trabajos eficientes permitiendo diseñar, interpretar y desarrollar sistemas robustos y eficientes. Los datos recopilados referentes a SQL Server mostraron una gran deficiencia en el tiempo de procesamiento para la inserción masiva de datos y por este motivo existen guías que detallan técnicas de optimización para operaciones para insertar, actualizar, visualizar y eliminar (operaciones CRUD) en SQL; ocasionado por una deficiencia en como trabajan los índices en las tablas.

Otro punto por tomar en cuenta al realizar la inserción de datos masiva es considerar las características y especificaciones que poseen cada uno de los equipos utilizados, debido a que los tiempos de ejecución serán en torno a las características que tienen. Por lo tanto, se tiene que el Equipo 2 obtuvo mejores resultados debido a las características de procesamiento que posee como la RAM y los núcleos físicos y virtuales, aunque la diferencia no es abrumadora por que la frencuencia a la que trabaja el procesador es es menor que el Equipo 1 . Los resultados obtenidos pueden variar mínimamente, siempre y cuando las condiciones planteadas se cumplan. Sin embargo, al realizar el mismo procedimiento se pueden obtener resultados diferentes si se genera una variación en la cantidad de datos que se insertan, tambien, si la RAM ya se encuentra estresada antes de comenzar el proceso o si la temperatura ambiente como la del procesador es elevada.

Referencias

1. Seyyed Hamid Aboutorabi[a], Mehdi Rezapour, Milad Moradi, and Nasser Ghadiri. Performance evaluation of sql and mongodb databases for big e-commerce data. In *2015 International Symposium on Computer Science and Software Engineering (CSSE)*, pages 1–7. IEEE, 2015.
2. Christodoulos Asiminidis, George Kokkonis, and Sotirios Kontogiannis. Database systems performance evaluation for iot applications. *International Journal of Database Management Systems (IJDMS) Vol*, 10, 2018.
3. Antonio Barbuzzi, Pietro Michiardi, Ernst Biersack, and Gennaro Boggia. Parallel bulk insertion for large-scale analytics applications. In *Proceedings of the 4th International Workshop on Large Scale Distributed Systems and Middleware*, pages 27–31, 2010.
4. Alexandru Boicea, Florin Radulescu, and Laura Ioana Agapin. Mongodb vs oracle–database comparison. In *2012 third international conference on emerging intelligent data and web technologies*, pages 330–335. IEEE, 2012.
5. David DeWitt and Jim Gray. Parallel database systems: The future of high performance database systems. *Communications of the ACM*, 35(6):85–98, 1992.
6. Adam Dziedzic, Manos Karpathiotakis, Ioannis Alagiannis, Raja Appuswamy, and Anastasia Ailamaki. Dbms data loading: An analysis on modern hardware. In *Data Management on New Hardware*, pages 95–117. Springer, 2016.
7. Ryan Johnson, Ippokratis Pandis, Nikos Hardavellas, Anastasia Ailamaki, and Babak Falsafi. Shore-mt: a scalable storage manager for the multicore era. In *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*, pages 24–35, 2009.
8. Boon Wee Low, Boon Yaik Ooi, and Chee Siang Wong. Scalability of database bulk insertion with multi-threading. In *International Conference on Software Engineering and Computer Systems*, pages 151–162. Springer, 2011.
9. Zachary Parker, Scott Poe, and Susan V Vrbsky. Comparing nosql mongodb to an sql db. In *Proceedings of the 51st ACM Southeast Conference*, pages 1–6, 2013.
10. Zhu Wei-Ping, LI Ming-Xin, and Chen Huan. Using mongodb to implement textbook management system instead of mysql. In *2011 IEEE 3rd International Conference on Communication Software and Networks*, pages 303–305. IEEE, 2011.