Conference Paper

NoSQL vs. SQL in Mass Data Management: An Empirical Study

NoSQL contra SQL en la Administración de datos masivos: Un estudio empírico

Francisco Rubio, Paul Vega, and Rolando P. Reyes Ch.

Maestría en Ingeniería de Software, Universidad de las Fuerzas Armadas - ESPE, Ecuador

Abstract

When developing a software project, it is important to choose the data base that best suits the needs of the project, whether it is relational or non-relational. This article compares the efficiency of these two types of databases in handling input and reading large amounts of data, using the SGDB MongoDB 3.2 and Microsoft SQL Server 2016. Concluding that, in projects where the handling of a large amount of data and a rapid response are primary requirements, it is better to use a non-relational database. In contrast, if the project requires the use of relationships between entities, without giving greater importance to the response time, it is better to opt for a related database.

Resumen: Al momento de desarrollar un proyecto software es importante escoger la base de datos que mejor se ajuste a las necesidades del proyecto. Las opciones de un técnico pueden estar entre una base de datos relacional o no relacional. El presente artículo compara la eficiencia de estos dos tipos de base de datos desde el punto de vista de la entrada y lectura de grandes cantidades de datos. Utilizamos a SGDB MongoDB 3.2 y Microsoft SQL Server 2016 para este estudio empírico. Concluimos que, en proyectos donde el manejo de una gran cantidad de datos y una respuesta rápida son requerimientos primordiales, y considerando estas variables, consideramos que podría ser idóneo el uso de una base de datos no relacional. En contraste, si el proyecto requiere el uso de relaciones entre entidades, sin dar mayor importancia al tiempo de respuesta, podría ser mejor optar por una base de datos relacional.

Keywords: Database, relational, nonrelational, SQL, NoSQL.
Palabras Claves: Base de datos, relacional, no relacional, SQL, NoSQL.

Corresponding Author:
Francisco Rubio
fjrubio@espe.edu.ec

Received: 24 December 2019
Accepted: 2 January 2020
Published: 8 January 2020

Publishing services provided by Knowledge E

Selection and Peer-review under the responsibility of the SIIPRIN-CITEGC Conference Committee.

1. Introducción

En los últimos años el sistema de bases de datos no relacional se ha mostrado más efectiva dentro del desarrollo de aplicaciones que requieren procesar grandes bloques de datos, de entrada y salida, en un breve periodo de tiempo.

Estas bases de datos, también denominadas NoSQL, han surgido como una innovación tecnológica en las arquitecturas empresariales de las últimas décadas [1]. De acuerdo con la encuesta realizada en el año 2012 a un conjunto de 1.300 personas por Couchbase, empresa involucrada en el desarrollo de bases de datos NoSQL, determinó que el 40% de sus encuestados piensan que la tecnología no relacional es esencial para sus operaciones diarias, y otro 37% de encuestados menciona que estas bases de datos están llegando a tener una gran importancia dentro de sus proyectos [2]. Para el caso de las bases de datos relacionales son definidas por las propiedades; atomicidad, consistencia, aislamiento y durabilidad, comúnmente identificadas con el acrónimo ACID (Atomicity, Consistency, Isolation and Durability). NoSQL fue construido bajo los conceptos de básicamente disponible, estado suave y consistente en algún punto, también denominadas características BASE (Basic Availability, Soft State and Eventual Consistency) [3].

Ahora bien, respecto a la NoSQL, esta nueva ideología soluciona diferentes problemas existentes dentro de las aplicaciones web, relacionadas al área de: robustez, eficiencia, rendimiento y disponibilidad [4]. Por esto, empresas como Amazon, Google, LinkedIn y Twitter han decido cambiar su sistema de almacenamiento relacional por uno orientado a NoSQL, en especial para el análisis de grandes volúmenes de datos en tiempo real [5].

Para los administradores de base de datos, el desafío se encuentra en la selección del tipo de base de datos a utilizar en el desarrollo de su proyecto; después de revisar la necesidad de su proyecto deben decidir si escoger, la simplicidad y documentación de una base de datos relacional, o la estabilidad y velocidad de una base de datos no relacional. Por lo tanto, la intención de este documento es contrastar el tiempo de respuesta de ambos tipos de bases de datos, en las peticiones de consulta y de creación para una gran cantidad de datos. Para realizar esta comparación se utilizó el Sistema de Gestión de Base de Datos (SGBD) Micorsoft SQL Server 2016 y el Sistema de Base de Datos MongoDB 3.2, debido a que ambos se encuentran entre los productos más usados para el desarrollo de proyectos software.

Este artículo está estructurado de la siguiente manera: Sección 2 Antecedentes que hace referencia a las características de bases de datos relacionales como no relacionales. Sección 3 la metodología que se describe la hipótesis con la que se va a trabajar para realizar el estudio empírico. Los resultados del estudio de eficiencia entre bases de datos relacionales y no relaciones son explicados en la Sección 4. Finalmente, las conclusiones en la Sección 5.

2. Antecedentes

Las bases de datos relacionales se describen como un conjunto de datos organizados dentro de tablas y relaciones, variando el máximo de columnas que puede tener dependiendo del SGBD utilizado. Por ejemplo, Microsoft SQL Server permite tener 1024 columnas por tabla [6]. Por otro lado, el número máximo de filas depende del espacio libre en disco y más importante de la facilidad de mantenimiento [6]. Estas bases de datos se apoyan en el lenguaje de dominio específico Structured Query Language, más conocido por sus siglas SQL, encajando perfectamente con el modelo relacional, brindando facilidad al momento de ejecución de una operación identificando el conjunto de resultados de una consulta [6].

Las características de estas bases de datos están establecidas por las propiedades ACID, acrónimo que viene de: Atomicidad (Atomicity), Consistencia (Consistency), Aislamiento (Isolation), Durabilidad (Durability) [5],[7], las cuales aseguran que el SGBD cumpla sin ningún error o interrupción.

Las bases de datos NoSQL, son consideradas como las primeras en seguir en el modelo no relacional [8], cuyo éxito, combinado con el proyecto de base de datos Dynamo de Amazon [9], dio inicio al desarrollo abierto y cerrado de varias bases de datos no relacionales. Gran cantidad de bases de datos NoSQL hacen referencia a su almacenamiento de tipo key-value-stores, esto significa, que en lugar de almacenar datos se almacenan objetos en forma de tiene documentos de formato XML, JSON y BSON [10], siendo denominadas base de datos orientadas a documentos.

Mongo DB toma el lugar como uno de los Sistemas de Base de Datos más populares, siendo ágil, escalable y basado en el modelo de almacenamiento por documentos [11]. Su principal objetivo es unir el paradigma key-value-stores junto con las funcionalidades del tradicional Sistema de Gestión de Base de Datos Relacional (RDBMS) [5].

MongoDB permite adecuar el esquema de la base de datos a las necesidades de la aplicación, permitiendo disminuir el tiempo y coste de la puesta en producción. Esto se lo puede realizar mediante la modificación del esquema desde el código mismo de la aplicación, sin ejecutar labores de administración de la base de datos.

La principal diferencia entre los dos tipos de bases de datos es que, el motor de base de datos NoSQL, no realiza las operaciones frecuentes tales como: JOIN, LIMIT, ni filtrar con la condición WHERE.

Referente a la sintaxis las bases de datos relacionales utilizan las instrucciones como SELECT, UPDATE, DELETE, del mismo modo PL/SQL [3].

Por otro lado, las bases de datos NoSQL constan de sintaxis diferentes para las tareas de crear, eliminar y modificar; las consultas se las realiza con el lenguaje de programación JavaScript [3],[10].

3. Metodología

Para comparar el tiempo de respuesta de las bases de datos se planteó las siguientes hipótesis nula y alternativa:

- *H0: Las bases de datos no relacionales tardan menos tiempo en ejecutar operaciones de inserción y consulta, en comparación con las bases de datos relacionales.*
- *H1: Las bases de datos no relacionales tardan igual o mayor tiempo en ejecutar operaciones de inserción y consulta, en comparación con las bases de datos relacionales.*

Con la finalidad de aceptar o rechazar la hipótesis nula, se realizó una serie de pruebas con diferentes conjuntos de datos, iniciado con 1000 registros y continuando con 5000, 10000, 50000 hasta 100000 registros. Por cada conjunto de datos se ejecutó 3 veces la sentencia de la operación y se tomó el tiempo de respuesta del log de salida; el resultado final es un promedio de los 3 tiempos obtenidos.

Como objeto de prueba se decidió utilizar uno de los objetos más común entre diversas aplicaciones web, el registro con la información de inicio de sesión del usuario. Este registro contiene los campos: Identificador único, nombre de usuario, nombre, apellido, género, contraseña y estado.

Debido a que el resultado de estas pruebas depende de la computadora sobre la cual son efectuadas, es importante señalar que esta comparación se realizó en un mismo computador con las siguientes características: Windows 10 Pro 64-bits, procesador Intel Core i7 (2.20 GHz), 8, 00 GB RAM de memoria.

3.1. Inserción de datos

Para las pruebas de la función insertar en la base de datos de Microsoft Server SQL, se tiene como referencia que cada tabla cuenta con una clave primaria siendo un campo auto-incremental.

La sentencia SQL para la inserción se describe a continuación:

```
INSERT INTO 'user_details'
('user_id', 'user_name', 'first_name', 'last_name',
'gender', 'password', 'status')
VALUES
```

— (null, 'rogers63', 'David', 'John', 'Male', 'e6a33eee180b07e563d74fee8c2c66b8', 1)

La Tabla 1 refleja los resultados en segundos, posteriormente de haber realizado las operaciones en la base de datos Microsoft SQL Server 2016.

TABLE 1: Tiempos de inserción en Microsoft SQL Server 2016.

Núm. Registros	Tiempo Prueba 1 (s)	Tiempo Prueba 2 (s)	Tiempo Prueba 3 (s)	Tiempo Promedio (s)
1 000	1.000	2.000	1.000	1.333
5 000	9.000	8.000	7.000	8.000
10 000	15.000	16.000	13.000	14.667
50 000	54.000	48.000	51.000	51.000
100 000	104.000	106.000	104.000	104.667

La sintaxis para realizar la tarea de inserción de datos en la base de datos NoSQL MogoDB es la siguiente:

```
db.PI3_Test.insert (
  {
    "user_id": "1",
    "username": "rogers63",
    "first_name": "David",
    "last_name": "John",
    "gender": "Male",
    "password": "e6a33eee180b07e563d74feeBc2c66b8",
    "status": 1
  }

  )
```

La Tabla 2 refleja los resultados en segundos, posteriormente de haber realizado las operaciones en la base de datos MongoDB 3.2.

TABLE 2: Tiempos de inserción en MongoDB 3.2.

Núm. Registros	Tiempo Prueba 1 (s)	Tiempo Prueba 2 (s)	Tiempo Prueba 3 (s)	Tiempo Promedio (s)
1 000	0.117	0.155	0.146	0.159
5 000	0.180	0.192	0.187	0.186
10 000	0.360	0.384	0.374	0.373
50 000	1.802	1.920	1.870	1.864
100 000	3.600	3.840	3.740	3.727

3.2. Consulta de datos

Para la prueba de consulta de datos en Microsoft SQL Server 2016 se utilizó el query de consulta:

```
SELECT * FROM user_details;
```

La Tabla 3 muestra los tiempos resultantes en segundos de las consultas realizadas en Microsoft SQL Server 2015 y el promedio final por cantidad de datos almacenada.

TABLE 3: Tiempos de consulta en Microsoft SQL Server 2016.

Núm. Registros	Tiempo Prueba 1 (s)	Tiempo Prueba 2 (s)	Tiempo Prueba 3 (s)	Tiempo Promedio (s)
1 000	0.000	0.000	0.000	0.000
5 000	0.000	0.000	0.000	0.000
10 000	0.000	0.000	0.000	0.000
50 000	1.000	0.000	1.000	0.667
100 000	1.000	2.000	2.000	1.667

La sintaxis para realizar la tarea de consultas en MongoDB 3.2 se la estructura de la siguiente forma:

```
db.PI3_Test.find();
```

La Tabla 4 muestra los tiempos resultantes en segundos de las consultas realizadas en MongoDB 3.2 y el promedio final por cantidad de datos almacenada.

TABLE 4: Tiempos de consulta en MongoDB 3.2.

Núm. Registros	Tiempo Prueba 1 (s)	Tiempo Prueba 2 (s)	Tiempo Prueba 3 (s)	Tiempo Promedio (s)
1 000	0.006	0.004	0.004	0.005
5 000	0.007	0.004	0.006	0.006
10 000	0.014	0.008	0.012	0.011
50 000	0.070	0.040	0.060	0.057
100 000	0.140	0.080	0.120	0.113

4. Resultados

Concluidas las pruebas se agruparon los tiempos promedio, para base de datos, para su análisis estadístico.

4.1. Inserción de datos

Los resultados promedios de inserción de ambas bases de datos se muestran para su comparación en la Tabla 5.

TABLE 5: Tiempos de inserción promedio.

Núm. Registros	Tiempo Microsoft SQL Server (s)	Tiempo MongoDB (s)
1 000	1.333	0.159
5 000	8.000	0.186
10 000	14.667	0.373
50 000	51.000	1.864
100 000	104.667	3.727

Como se puede observar en la Tabla 5, Microsoft SQL Server 2016 maneja de forma rápida la inserción de una pequeña cantidad de datos. Sin embargo, pasada la cantidad de 1 000 registros se empieza a visualizar un fuerte incremento en el tiempo de ejecución.

Las operaciones realizadas en MongoDB 3.2 se las ejecutó en tiempo cortos, donde se apreció que al llegar a los 100 000 registros se nota un incremento significativo a su tiempo promedio.

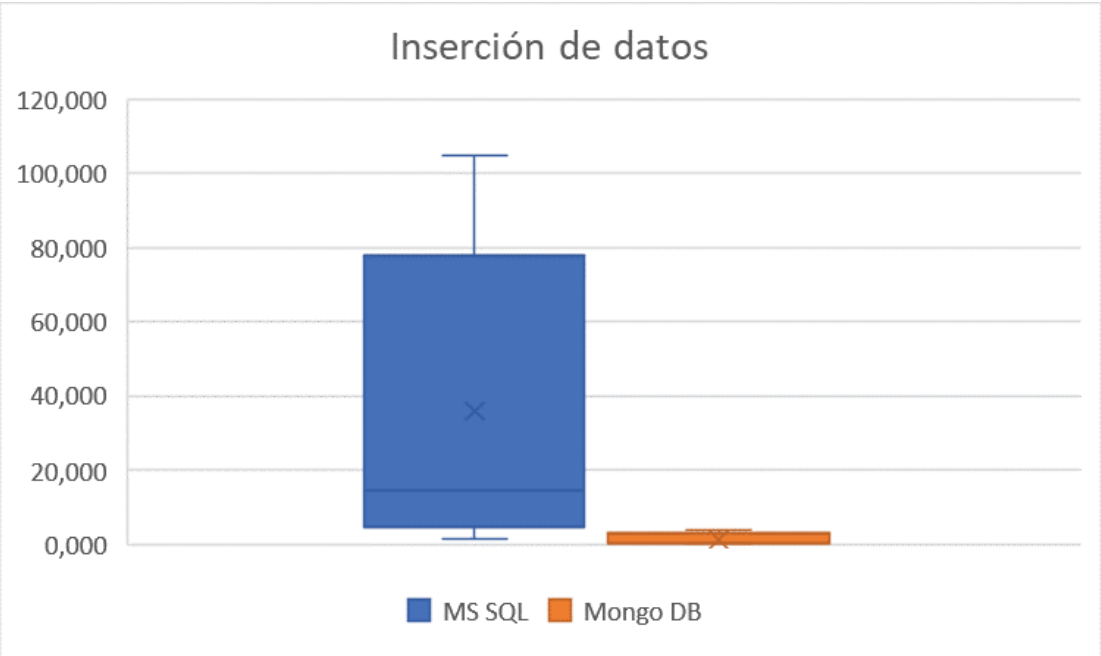

Figure 1: BoxPlot de tiempos promedio en la inserción de datos.

La Figura 1 confirma que la base de datos no relacional MongoDB 3.2 responde en un menor tiempo las sentencias de inserción de datos, mostrando un mejor manejo de cantidades de datos masivos en esta operación.

4.2. Consulta de datos

En la Tabla 6, se aprecia los resultados promedios al ejecutar las consultas en las diferentes bases de datos.

TABLE 6: Tiempos de consulta promedio.

Núm. Registros	Tiempo Microsoft SQL Server (s)	Tiempo MongoDB (s)
1 000	0.000	0.005
5 000	0.000	0.006
10 000	0.000	0.011
50 000	0.667	0.057
100 000	1.667	0.113

Como se puede observar en la Tabla 6 ambas bases de datos realizan las consultas en tiempos rápidos. Es solamente al llegar a 100 000 números de registros donde se puede visualizar la ventaja en eficiencia que tiene MongoDB 3.2 al momento de realizar consultas de datos masivos.

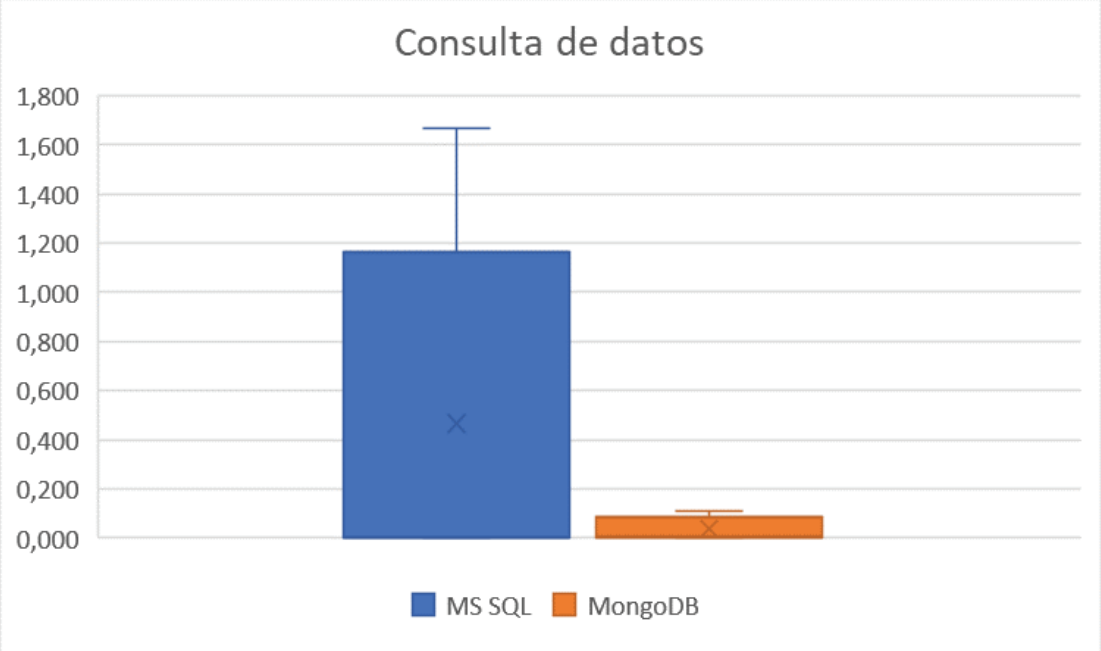

Figure 2: BoxPlot de tiempos promedio en la consulta de datos.

Analizando la Figura 2 se concluye que, la base de datos MongoDB 3.2 realiza la consulta de datos en un menor tiempo comparado con la base de datos relacional. Recalcando la ventaja que tiene MongoDB 3.2 en la administración de cantidades de datos masivos.

5. Conclusiones

La finalidad de este estudio fue comparar la eficiente que puede existir al ejecutar operaciones entre base de datos relacional y base de datos no relacional, con el manejo de datos de masivos. Entre las cuales, las bases de datos no relacional (NoSQL) resultan ser más eficientes al trabajar con grandes cantidades de datos que las bases de datos relacionales. Esto parece no ser algo extraño. Sin embargo, al finalizar las pruebas, basados en los resultados obtenidos, logramos aceptar nuestra hipótesis nula (H0), las bases de datos no relacionales tardan menos tiempo en ejecutar operaciones de inserción y consulta, en comparación con las bases de datos relacionales.

MongoDB, siendo una base de datos no relacional (NoSQL), brinda una mayor eficiencia al momento de trabajar con una gran cantidad de datos y flexibilidad en su estructura debido a que no se impone a ningún documento al momento de realizar un proyecto, lo que permite realizar operaciones con un corto tiempo de ejecución y almacenar todo tipo de datos. Además, el formato utilizado para las operaciones de administración es JSON, un formato de texto ligero y comúnmente usado para el desarrollo de aplicaciones web.

Estas características permiten a la base de datos MongoDB 3.2 tomar un menor tiempo de ejecución en las operaciones de consulta e inserción en comparación a Microsoft SQL Server 2016. Aunque en cantidades mínimas de datos ambas bases de datos demostraron una buena eficiencia, a medida que el conjunto de datos llega a superar los 50 000 registros es en donde se empieza a ver la ventaja del uso de una base de datos no relacional para proyectos que requieran manejar grandes cantidades de datos, esperando un tiempo de ejecución corto y realizando peticiones múltiples.

Es importante mencionar que, las bases de datos relacionales pueden tener un tiempo de respuesta menor, pero sus características ACID acrónimo que viene de: Atomicidad (Atomicity), Consistencia (Consistency), Aislamiento (Isolation), Durabilidad (Durability), y funcionalidades, como consultas combinadas, facilitan el trabajo con estructuras complejas. Estas bases de datos funcionan mejor en proyectos que requirieran relación entre entidades y una estructura de arreglo.

En conclusión, si el proyecto a desarrollar tiene la necesidad de una base de datos rápida y flexible lo mejor es escoger una base de datos no relacional; por otra parte si el proyecto da mayor prioridad a las relaciones entre entidades que a la velocidad de respuesta, es mejor escoger una base de datos relacional.

References

[1] S. Tiwari, Professional NoSQL. John Wiley & Sons, 2011.

[2] A. B. M. Moniruzzaman y S. A. Hossain, «Nosql database: New era of databases for big data analytics-classification, characteristics and comparison», ArXiv Prepr. ArXiv13070191, 2013.

[3] A. Boicea, F. Radulescu, y L. I. Agapin, «MongoDB vs Oracle--database comparison», en Emerging Intelligent Data and Web Technologies (EIDWT), 2012 Third International Conference on, 2012, pp. 330--335.

[4] R. Hecht y S. Jablonski, «NoSQL evaluation: A use case oriented survey», en Cloud and Service Computing (CSC), 2011 International Conference on, 2011, pp. 336--341.

[5] C. Strauch, U.-L. S. Sites, y W. Kriha, «NoSQL databases», Lect. Notes Stuttg. Media Univ., vol. 20, 2011.

[6] A. Beaulieu, Learning SQL: Master SQL Fundamentals. O'Reilly Media, Inc., 2009.

[7] V. Sharma y M. Dave, «Sql and nosql databases», Int. J. Adv. Res. Comput. Sci. Softw. Eng., vol. 2, n.o 8, 2012.

[8] F. Chang et al., «Bigtable: A distributed storage system for structured data», ACM Trans. Comput. Syst. TOCS, vol. 26, n.o 2, p. 4, 2008.

[9] G. DeCandia et al., «Dynamo: amazon's highly available key-value store», en ACM SIGOPS operating systems review, 2007, vol. 41, pp. 205--220.

[10] Y. Li y S. Manoharan, «A performance comparison of SQL and NoSQL databases», en Communications, computers and signal processing (PACRIM), 2013 IEEE pacific rim conference on, 2013, pp. 15--19.

[11] B. Dayley, NoSQL with MongoDB in 24 Hours, Sams Teach Yourself. Sams Publishing, 2014.