ISSN 0121–1129

Utilidad y funcionamiento de las bases de datos NoSQL

Databases NoSQL's Utility and Functioning

Fecha de recepción: 15 de noviembre de 2012
Fecha de aprobación: 12 de diciembre de 2012

Alexander Castro Romero*
Juan Sebastián González Sanabria**
Mauro Callejas Cuervo****

Resumen

Reflexiona sobre las bases de datos NoSQL, describiéndolas y analizando el porqué de su importancia y actualidad; además recopila y define algunas características de este tipo de base de datos, para revisar las taxonomías más importantes y analizar el uso conjunto de tecnologías NoSQL y relacionales, con el fin de proporcionar un punto de partida para los trabajos en esta área por parte de investigadores.

Palabras clave: NoSQL, Bases de datos, Arquitectura en bases de datos.

Abstract

These are reflections on NoSQL databases, which describe and analyze the reasons for its importance and for being an updated subject. Besides it collects and defines some characteristics to describe this type of databases, in order to review the major taxonomies and to analyze the joint use of NoSQL and relational technologies, to provide a starting point to the researchers work in this area.

Keywords: NoSQL, Databases, Database Architecture.

* Estudiante Décimo Semestre Ingeniería de Sistemas y Computación, Universidad Pedagógica y Tecnológica de Colombia. alexanderb221@gmail.com

** Especialista en Bases de Datos, Universidad Pedagógica y Tecnológica de Colombia. Ingeniero de Sistemas y Computación, Universidad Pedagógica y Tecnológica de Colombia. Docente Auxiliar Ingeniería de Sistemas y Computación, Universidad Pedagógica y Tecnológica de Colombia. Investigador Grupo de Investigación en el Manejo de la Información (GIMI). juansebastian.gonzalez@uptc.edu.co, ing.jsgonzalez@gmail.com

*** Magíster en Ciencias Computacionales. Instituto Tecnológico y de Estudios Superiores de Monterrey. Especialista en Ingeniería del Software. Universidad Antonio Nariño. Ingeniero de Sistemas. Universidad Antonio Nariño. Docente Asistente Ingeniería de Sistemas y Computación. Universidad Pedagógica y Tecnológica de Colombia. Investigador Grupo de Investigación en Software (GIS). mauro.callejas@uptc.edu.co, maurocallejas@gmail.com

I. Introducción

Día tras día el manejo de la información se hace más complejo; diferentes factores hacen que las personas involucradas en el área busquen tecnologías que les ayuden con este problema; sin embargo, el manejo comercial que se le da al tema dificulta realizar una buena elección.

Haciendo un paralelo con lo dicho por Henry Ford, "si le hubiera preguntado a la gente ¿qué querían?, me habrían dicho que un caballo más rápido"; las bases de datos no pueden quedarse estancadas en un modelo que no ofrece alternativas viables a problemas que se están presentando, por eso se hace necesario buscar posibles formas de solucionar estos problemas y que sea un avance en el área del manejo de la información, así como el automóvil fue un avance en el área del transporte.

En los últimos años ha aumentado el interés por las bases de datos NoSQL (Not only SQL)**,** un nuevo conjunto de tecnologías que pueden contribuir al manejo de la información, como lo menciona [1]: "las organizaciones que recopilan grandes cantidades de datos no estructurados son cada vez más propensas al uso de las bases de datos NoSQL"; lastimosamente, la documentación acerca de estas es escasa y generalmente se limita a pequeños artículos o ejemplos prácticos, de los cuales la mayoría no se encuentran en español. Adicionalmente, es preocupante el desconocimiento del tema en ambientes académicos.

Por lo anterior, el presente documento hace una revisión de las tecnologías NoSQL, tomando las partes más significativos para formular una reflexión que sirva como punto de partida para las personas interesadas en investigar el tema.

II. Cambio de paradigma

Desde su creación, las bases de datos han sido un soporte para la organización de la información dentro de los diferentes tipos de entidades, debido a que "las bases de datos comenzaron a aparecer a finales de 1950 y comienzos de 1960, impulsadas por dos factores tecnológicos: el incremento de la fiabilidad de los procesadores de ordenador y la expansión de la capacidad de almacenamiento secundario en cintas y unidades de disco" [2].

En 1970 se propusieron por primera vez las bases de datos relacionales y las teorías subyacentes [3], entre las que se destaca el modelo de base de datos relacional, que implicó un cambio radical en el manejo de la información apoyándose en operaciones de conjuntos que combinan tablas de datos separadas (o relaciones) para producir un conjunto de respuestas. Las consultas se especifican utilizando el lenguaje de consulta estructurado SQL (por las siglas en inglés de Structured Query Language), soportado en el álgebra relacional, y que permite a un usuario expresar su consulta en forma declarativa, sin ningún tipo de instrucciones detalladas de programación.

Sin embargo, este tipo de bases de datos está presentando inconvenientes, y la historia se repite; como se mencionó al inicio, continuos cambios en las sociedades impulsan la aparición de modelos que satisfagan nuevas necesidades, uno de estos es NoSQL, que define un conjunto de tecnologías que se apartan de lo planteado por los gestores de bases relacionales, por ejemplo, la interfaz de consulta para los usuarios en NoSQL no es soportada sobre SQL; aunque algunos autores lo describen como "un movimiento más que como una tecnología" [4].

Por otra parte, desde la aparición del término NoSQL existe un inconveniente conceptual (en un principio se pensó usar el término "NoRel", haciendo referencia a "No Relacional", pero no sonaba comercial), puesto que la denominación puede interpretarse a primera vista como oposición de SQL; por ello se ha querido dar vuelta a este concepto aclarando que NoSQL (siglas en inglés de Not only SQL) se define como "No solo SQL"; "NoSQL es usado como un término general por todas las bases de datos y almacenes de datos que no siguen los populares y bien establecidos principios RDBMS (Relational Database Management System), y a menudo está relacionado con grandes conjuntos de datos y su manipulación en una escala Web" [5].

Por tanto, se puede concluir que el término NoSQL hace referencia al conjunto de tecnologías, en bases de datos, que buscan alternativas al sistema de bases de datos relacional, en un contexto donde priman la velocidad, el manejo de grandes volúmenes de datos y la posibilidad de tener un sistema distribuido. También se hace necesario destacar que al ser tan amplio este concepto cobija a múltiples tecnologías, unas más nuevas que otras, pero para este trabajo se tendrán en cuenta las tecnologías que se están manejando actualmente y que están representando este movimiento.

III. ¿Por qué ahora?

El paradigma de bases de datos NoSQL surge a causa del cambio que se ha dado en el manejo de la información durante las últimas décadas; por ejemplo, se proyecta un crecimiento del tráfico IP global, de los centros de datos, que en la actualidad es casi de un zetabyte (1.099.511.627.776 gigabytes) por año a 4.8 zetabyte en el 2015 [6].

Hoy los volúmenes de información crecen a un ritmo sin precedentes, y cada vez se hace más compleja su administración; las empresas no solo desean almacenar esta información, sino quieren sacarle el mayor provecho; los usuarios piden cada vez más velocidad en las consultas, y la arquitectura de los sistemas ha tenido un cambio considerable. El porqué de las bases de datos NoSQL se puede resumir en los siguientes tres aspectos:

A. Tamaño y cantidad de la información

Para nadie es secreto que el tamaño de los archivos ha crecido a un ritmo exponencial; un claro ejemplo son los archivos de video, pues el aumento de la calidad vino de la mano del aumento en el tamaño de archivo; hace tan solo unos años una película usaba como medio de almacenamiento un DVD, que tenía una capacidad de 4.7 GB, hoy se distribuyen en un disco Blu-ray, con una capacidad de 25 GB. Por otro lado, la cantidad de la información también crece; por dar un ejemplo, tan solo hace unos años tomar una fotografía era algo complejo, principalmente por lo largo del proceso y los costos que implicaba, hoy se suben 4.5 millones de fotos a Flickr (sitio Web para almacenamiento de imágenes y videos) cada día [7].

Las grandes compañías que manejan altos volúmenes de información se dieron cuenta de que con la infraestructura que contaban no podrían manejar la cantidad de información que se proyecta para dentro de cinco años, por eso, son esas compañías las que están invirtiendo y creando soluciones NoSQL. Un claro ejemplo es lo sucedido en los Juegos Olímpicos de Londres 2012, donde se esperaba que "845 millones de usuarios activos mensuales de Facebook sean responsables de más de 15 terabytes de datos al día, mientras que Twitter está a la espera de más de 13.000 tweets por segundo" [8].

Estos dos aspectos: el aumento tanto en el tamaño de los archivos como en su cantidad ha incidido en el crecimiento del interés hacia tecnologías NoSQL; además, hay que tener en cuenta que las bases de datos relacionales (soluciones actuales) presentan serios problemas en cuanto a escalabilidad en el manejo de la información se refiere, lo que genera que a medida que aumentan los datos, el desempeño disminuye y se hacen menos intuitivas (las consultas son cada vez más largas y complejas).

B. Velocidad

Con la evolución en el campo de hardware, y los dramáticos aumentos de la velocidad de navegación en internet, se hace inaceptable para el usuario final una demora en segundos en una consulta a una página Web. Como lo menciona [9], es la presión de los usuarios y el surgimiento de nuevas tecnologías (como el software en la nube y el streaming) las que han impulsado el crecimiento de las tecnologías NoSQL; con millones de usuarios solicitando productos completos, pero sobre todo veloces, que al final de cuentas están basados en almacenamientos y búsqueda de la información, se hace necesario un paradigma en bases de datos orientado a la velocidad.

"Es un hecho que cualquier aplicación moderna va a tener una arquitectura distribuida. El tamaño de los conjuntos de datos modernos es sólo una de las razones

para su distribución, y no la más importante. Las aplicaciones modernas (especialmente aplicaciones Web) tienen muchos usuarios al mismo tiempo que exigen respuestas razonablemente ágiles" [10]; el usuario final no está interesado en qué tecnología de bases de datos se use, sino en que la respuesta a sus consultas se dé en un tiempo razonable, como se demuestra en un experimento realizado para medir la paciencia de los usuarios en las consultas, el cual indicó que demoras de medio segundo tienen serias consecuencias en las métricas del negocio [11].

Las bases de datos relacionales están diseñadas para ser "organizadas"; las tablas funcionan de tal forma que las podemos entender; lastimosamente, cuando estos sistemas fueron diseñados se pensaba en sistemas pequeños, estructurados y centralizados, pero eso cambio y la información dejó de ser tan "estructurada", los sistemas crecieron a un ritmo exponencial y se hizo necesario distribuir la información, lo que ocasionó que estas bases de datos fueran cada vez más lentas, y esto se constituyó en un problema para los desarrolladores.

C. Falta de innovación

Las bases de datos tradicionales fueron creadas para la misma necesidad general de almacenamiento y manejo de la información, pero se diseñaron teniendo en cuenta las características de la época: grandes computadores y datos estructurados, sin embargo aunque la necesidad es la misma, la época ha cambiado, y por eso se hace necesario innovar en el tema para crear soluciones que respondan a las necesidades actuales.

Es claro que las soluciones SQL están de alguna manera estancadas, además, para la comunidad "Open Source" ha habido noticias inquietantes últimamente (la compra de MySQL por parte de Oracle [12]), eso sin contar con los costos de las soluciones más populares, lo cual hace que los desarrolladores miren hacia otro lugar: "Oracle te diría que con el grado justo de hardware y la configuración correcta de Oracle RAC (Real Application Clusters) y con el software mágicamente asociado, se puede lograr la misma escalabilidad. Pero ¿a qué precio?" [13].

Además, es claro que las ideas para SQL se agotaban, y la centralización de los productos aislaba las nuevas propuestas; la gente se enfrascaba en controversias sobre cuál gestor de bases de datos era mejor, y con cada versión las novedades escaseaban. "Los vendedores de Sistemas Gestores de Bases de Datos (y la comunidad de investigación) deben comenzar con una hoja de papel y sistemas de diseño para los requisitos del mañana, no continuar empujando líneas de código y arquitecturas diseñadas para las necesidades de ayer" [14].

En este punto se puede concluir que el cambio del paradigma en el manejo de la información motivó el auge del movimiento NoSQL; adicionalmente, hay que destacar el papel de grandes empresas como Google, que con la presentación de su trabajo "Bigtable: A Distributed Storage System for Structured Data", allá en el 2006, contribuyó para que la comunidad especializada hablara del tema [15].

"Las últimas tendencias tanto en la red como en la computación en la nube están haciendo de NoSQL el modelo más deseable. La necesidades de (ampliar rápidamente) escalabilidad, desvincular el hardware del modelo de datos y proporcionar bases de datos más eficientes, son todos factores que contribuyen a esta transición" [16]. Por último, debe destacarse que el tema no es nuevo, pero históricamente se puede decir que en la última década se ha hablado mucho de él [17].

IV. Características NoSQL

Aunque es difícil determinar propiedades comunes para un conjunto de tecnologías, se proponen seis características específicas para poder encasillar a las bases de datos NoSQL [18]:

- Escalabilidad horizontal: refiriéndose a la facilidad añadir, eliminar o realizar operaciones con elementos (hardware) del sistema, sin afectar el rendimiento.

- Habilidad de distribución: tiene que ver con las escalabilidad horizontal, pero haciendo énfasis

en su soporte; para ello se tiene en cuenta la habilidad de replicar y distribuir los datos sobre los servidores.

- Uso eficiente de recursos: aprovecha las nuevas tecnologías, como los discos en estado sólido, el uso eficiente de recursos como la memoria RAM y los sistemas distribuidos en general.

- Libertad de esquema: al no tener un esquema rígido se permite mayor libertad para modelar los datos; además facilita la integración con los lenguajes de programación orientados a objetos, lo que evita el proceso de mapeado.

- Modelo concurrencia débil: no implementa ACID (Atomicity, Consistency, Isolation and Durability), que reúne las características necesarias para que una serie de instrucciones puedan ser consideradas una transacción, sin embargo sí se tienen en cuenta algunas consideraciones para asegurar estos aspectos, pero no son tan estrictas.

- Consultas simples: las consultas requieren menos operaciones y son más naturales, por la tanto, se gana en simplicidad y eficiencia.

Por otro lado, se propone considerar cuatro aspectos teóricos muy importantes [19]:

- Teorema CAP (Consistency Availability Partition tolerance): en el 2000, Eric Brewer [20] propuso la idea de que en un entorno distribuido un sistema no puede mantener continuamente consistencia perfecta, disponibilidad y tolerancia partición simultáneamente.

  - Consistencia: los usuarios tienen la posibilidad de acceder simultáneamente a un mismo registro.
  - Disponibilidad: la garantía de que cada solicitud recibe una respuesta.
  - Tolerancia de reparto: el sistema sigue funcionando a pesar de la pérdida arbitraria de información.

Este punto es importante, pues permite entender que para ganar velocidad se debe sacrificar por lo menos una de las características mencionadas; pero el enfoque es que el sistema puede estar haciendo cambios entre las combinaciones de estos elementos, teniendo en cuenta las siguientes posibilidades:

  - CP: el sistema ejecutará las operaciones de forma consistente, aunque se pierda la comunicación entre nodos (partición del sistema), pero no se asegura que el sistema responda (disponibilidad).
  - AP: el sistema siempre responderá a las peticiones, aunque se pierda la comunicación entre nodos (partición del sistema). Los datos procesados pueden no ser consistentes.
  - CA: el sistema siempre responderá a las peticiones y los datos procesados serán consistentes. En este caso no se permite una pérdida de comunicación entre nodos (partición del sistema).

- ACID "relajado": como ya se mencionó, ACID es muy importante en los sistemas de bases de datos relacionales; sin embargo, en los sistemas NoSQL se hizo evidente que con el fin de proporcionar una gran escalabilidad, puede ser necesario relajarse o redefinir algunas de las cualidades de este modelo, en particular consistencia y durabilidad.

  Por ejemplo, en el caso de la consistencia, en los sistemas relacionales la consistencia completa hace a estos más lentos, pues se implementan cerraduras, para (por dar dos casos) que dos usuarios no puedan modificar un archivo al mismo tiempo y que no se pueda alterar la estructura si alguien está trabajando con un elemento de esta; en los sistemas NoSQL se opta por una consistencia eventual, que permite a cada sistema hacer cambios a los datos y aprender de otras actualizaciones realizadas por otros sistemas dentro de un corto periodo de tiempo, sin ser totalmente coherente en todo momento.

  Al hacer esto se reafirma la teoría de que es necesario sacrificar algunas características que

están presentes en las bases de datos relacionales para obtener nuevas características o mejorar en algunos aspectos, pero se puede hacer de una forma que no afecte tanto al sistema.

- Datos y modelo de acceso: el modelo de datos de las bases de datos relacionales, con sus tablas, relaciones, vistas, filas y columnas, ha sido muy útil y aceptado; el orden que brinda y la estructuración son cuestiones que se destacan pero que presentan problemas, entre ellos, la relación con los lenguajes orientados a objetos y la dificultad para la adaptación a datos no estructurados.

  Por otro lado, el modelo de acceso a las bases de datos relacionales presenta dificultades en la gestión de las restricciones globales en un entorno distribuido, y los intenta solucionar con la creación de barreras para coordinar cambios; esto introduce sobrecarga de la red, y a veces puede detener el progreso en el sistema.

- Datos y procesos distribuidos: además de distribuir los datos, las bases de datos NoSQL permiten la aplicación de procesos a los datos que se encuentran distribuidos sin necesidad de centralizarlos; este almacenamiento distribuido contribuye positivamente a la fiabilidad y escalabilidad de la base de datos, pero presenta muchos retos en su implementación, como los costos extra para el almacenamiento de datos duplicados y su correcta gestión.

Se hace necesario unificar criterios para estudiar a las bases de datos NoSQL, por eso se establecen tres características generales:

- Son sistemas descentralizados: esto quiere decir que son sistemas distribuidos y que permiten una fácil escalabilidad horizontal; por eso, para futuras consideraciones se hace necesario revisar las características de sistemas distribuidos.

- Esquema flexible: no quiere decir que no exista un esquema, sino que cada tecnología maneja un esquema que no es rígido. Estos esquemas están enfocados en las características de los sistemas actuales, y que el modelado sea sencillo y natural.

- Cambio de modelo: hay quienes pretenden que NoSQL mejore sus fallas sacrificando sus principios, lo que resultaría volver al modelo relacional; sin embargo, no hay que caer en esto, hay que recordar que este es un nuevo paradigma y necesita nuevas propuestas, un ejemplo es lo que pasó con ACID, en vez a implementarlo "a las malas" se propone un sistema llamado "BASE (Basically Available, Soft State, Eventually Consistent), que es diametralmente opuesta a ACID. Donde este último es pesimista y fuerza la consistencia al final de cada operación, BASE es optimista y acepta que la consistencia de base de datos estará en un estado de flujo. La disponibilidad de BASE se logra mediante el apoyo a fallos parciales sin fallo total del sistema" [21].

Es interesante analizar cómo muchas de estas características responden a las oportunidades de investigación en bases de datos planteadas por "Reporte Claremont en investigación en bases de datos" [22], que define cinco oportunidades de investigación:

- Revisar los motores de bases de datos: que en sí es uno de los propósitos de NoSQL.

- Programación declarativa para plataformas emergentes: lo usuarios de NoSQL son capaces de escribir código robusto para entornos complejos debido a la cercanía entre el modelado en NoSQL y el modelado del negocio.

- La interacción de datos estructurados y no estructurados: como ya se ha mencionado, el manejo de datos no estructurados es uno de los puntos fuertes de NoSQL.

- Nube de servicios de datos: la computación en la nube se puede definir como una de las causas del resurgimiento de las tecnologías NoSQL, en este punto se destaca que estas tecnologías fueron diseñadas para soportar la arquitectura de este tipo de sistemas.

- Aplicaciones móviles y mundos virtuales: la necesidad de gestionar grandes cantidades de diversos datos creados por el usuario, de sintetizar de forma inteligente y de proporcionar servicios en tiempo real es satisfecha por las bases de datos NoSQL a la perfección.

V. Taxonomía

Aunque existen muchas tecnologías en bases de datos NoSQL, cuando se habla de categorías se pueden destacar cuatro [23]:

A. Almacenamiento clave-valor

De las bases de datos NoSQL, las de clave/valor son las más simples; en ellas se asigna una clave única (llamada llave) a un valor que es típicamente una cadena arbitraria. La operación de encontrar el valor asociado a una clave se denomina "lookup" (indexación), y la relación entre una clave y su valor se denomina correlación (vinculante).

La naturaleza poco estructurada de las bases de datos de clave/valor asegura una buena escalabilidad, sin embargo, es complejo crear un llave representativa para cada registro; además, hay que tener en cuenta que la novedad con este tipo de bases de datos NoSQL es la posibilidad de tener los datos en un ambiente distribuido. Se recomienda su uso en casos donde se necesita velocidad en las consultas o se tienen muchos datos con estructura simple que requieren ser procesados una y otra vez y tienen valores cambiantes, por ejemplo, listas de los más vendidos, carritos de la compra, las preferencias-cliente, gestión de sesiones, rango de venta y catálogo de productos [24].

Ejemplo: Redis [25] es una base de datos NoSQL de clave-valor, donde los valores pueden contener tipos de datos más complejos, con operaciones atómicas definidas para ese tipo de datos. Los tipos de datos Redis están estrechamente relacionados con las estructuras de datos fundamentales y se exponen al programador como tal, sin capas de abstracción adicionales.

Ejemplo práctico:
Set name Batman
Get name

lpush heroes "Batman"
lpush heroes "Superman"
lpush heroes "Green Lanter"
lrange heroes 0 -1 //Muestra los dos primeros
sort heroes alpha //los ordena alfabéticamente

hset user name "Batman"
hset user tierra "Tierra 1"

En el ejemplo se aprecia cómo se asigna como valor los nombres de los héroes; Redis se encarga se generar una clave única para cada uno, lo que permite facilitar su recuperación; para ello existen diferentes funciones, como "irange" y "sort".

B. Almacenamiento de documentos

Un almacén de documentos gestiona los registros de datos estructurados jerárquicamente y proporciona medios para recuperar registros en función de su contenido real. Esta categoria de bases de datos NoSQL proporciona la capacidad de manejar millones de lecturas simultáneas, puesto que ya tienen una lectura simple (como típicamente un documento que contiene toda la información requerida).

En el almacenamiento de documentos se mantiene toda la información relacionada a una entidad en solo un documento [26].

Ejemplo: MongoDB [27] (de "humongous" -descomunal-) es una base de datos NoSQL escalable, de alto rendimiento y de código abierto; guarda estructuras de datos en documentos tipo JSON (JavaScript Object Notation) con un esquema dinámico, haciendo que la integración de los datos en ciertas aplicaciones sea más fácil y rápida.

En este tipo de base de datos NoSQL se manejan cuatro elementos:

- Base de datos: contiene un conjunto de colecciones.

- Colección: contiene un conjunto de documentos, puede relacionarse con las tablas del modelo relacional, pero hay que tener en cuenta que acá se pueden almacenar documentos con diferentes atributos.

- Documento: es un conjunto de campos.

- Campo: es una pareja compuesta por una Llave y un Valor, donde la llave es el nombre del campo y el valor su contenido.

Ejemplo práctico:

```
Use dc
dc.historia.save({“nombre”: “El caballero de la
noche regresa”, “Paginas”: 25, “personajes”: [
{“nombre”: “Batman”, “Tierra”: “Tierra 1”},
{“nombre”: “Superman”, “Tierra”: “Tierra 1”}
]
})
Consulta:
dc.historia.find({“personajes.nombre”:”Batman”})
```

En el ejemplo se aprecia cómo en un único documento se puede guardar tanto la información de una historieta (su nombre y número de páginas) como los nombres de los personajes que aparecen en ella (además de la tierra en la que viven); en el modelo relacional serían necesarias al menos dos tablas para modelar este problema. Adicionalmente, las consultas se expresan de una manera natural, es decir, manejan una estructura más sencilla y semántica.

C. Almacenamiento de familias de columnas

Una base de datos orientada a columnas almacena su contenido por columnas, en lugar de por filas. Las bases de datos orientadas a columnas tienden a ser un híbrido de las clásicas bases de datos relacionales y la tecnología orientada a columna. La columna es la base, es un elemento compuesto de un nombre, un valor y una marca de tiempo. La base de datos almacena sus datos (físicamente por familias de columna) de manera que pueden ser rápidamente agregados, con menos actividad de entrada y salida.

Como los datos de la columna son de tipo uniforme, la optimización del tamaño de almacenamiento se logra mediante esquemas tales como algoritmos de compresión. Una desventaja de almacenar los datos en columnas es que los datos de una entidad se esparcen entre varias columnas; por lo tanto, la inserción y la actualización o lectura del contenido completo de una entidad puede ser más lenta y compleja que en una base de datos relacional.

Las bases de datos orientadas a columnas se recomiendan para grandes almacenes de datos cuya lectura es más frecuente que su escritura, y es necesario realizar muchas operaciones con los atributos de las entidades, como en el manejo de datos estadísticos [28].

Ejemplo: HBase[29] es una base de datos NoSQL de código abierto inspirada en Big Table de Google, la cual comprende un conjunto de tablas que contienen filas y columnas; además, debe tener un elemento definido, como una clave principal, y todos los intentos de acceso a ella han de usar esta clave principal. Una columna representa un atributo de un objeto; cada fila podría es una entrada de registro. En HBase se permite que muchos atributos puedan ser agrupados juntos, en lo que se conoce como familias de columna, de tal manera que los elementos de una familia columna se almacenan todos juntos.

D. Almacenamiento de grafos

Una base de datos de grafos utiliza estructuras grafos con nodos (que vienen a ser objetos o entidades), bordes (relación entre los objetos o entidades) y propiedades para representar y almacenar información. Un grafo (o una red) es una estructura de datos flexible y que se integra más fácilmente con la estructura de aplicaciones orientadas a objetos. Las bases de datos de grafos se pueden escalar de forma más natural a conjuntos de datos de gran tamaño, y son más adecuadas para la gestión ad-hoc y el cambio de datos con esquemas cambiantes. Este

tipo de bases de datos se recomienda para estructuras dinámicas complejas, como las redes sociales, como se aprecia en [30].

Ejemplo: Neo4j [31] es una base de datos NoSQL basada en grafos de alto rendimiento, con todas las características de una base de datos madura y robusta.

Ejemplo práctico:

```
node_auto_indexing=true
node_keys_indexable=name,encuentros
relationship_auto_indexing=true
relationship_keys_indexable=ROOT,FRIEND,ENEMY
mkrel -t ROOT -c –v //Crea el primer nodo
cd 1 //Va al primer nodo
set name “Batman” // Asigna información
mkrel -t FRIEND –cv // Crea nuevo nodo con la relación FRIEND DESDE 1
cd 2 //Va al nodo creado
set name “Superman”
cd .. //Se devuelve a Batman
mkrel -t ENEMY –cv
cd 3 //Va al nodo creado
set name “Joker”
mkrel -t ENEMY 2 // Hace que el Joker se vuelva enemigo de Superman
ls –rv // Lista de relaciones del Joker
cd -r 2 // Toma la relación con Batman
set -t int age 3 //Le asigna un número de encuentros igual a 3
Consulta:
start joker = node:node_auto_index(name=Joker)
match joker-[: ENEMY]-jfb return jfb.name;
```

Como se aprecia en el ejemplo, existen relaciones complejas que son modeladas sin necesidad de elementos adicionales y conservando la lógica del negocio.

VI. ¿Distintos destinos?

Como se ha podido apreciar, NoSQL y SQL presentan ventajas y desventajas, pero hay algo interesante: las ventajas de una son las desventajas de la otra, lo cual puede enfocar la investigación al uso conjunto de ellas “Las tecnologías NoSQL son útiles para resumir un conjunto de datos enorme, mientras que SQL se puede utilizar para un análisis más detallado” [33].

EMC (compañía norteamericana en el área del almacenamiento de datos) está utilizando una mezcla de bases de datos relacionales y almacenes de datos NoSQL para analizar la percepción pública de la empresa y sus productos. La idea es analizar espacios de opinión en internet, buscando menciones de la marca y sus productos, y evaluar si las referencias son positivas o negativas; para ello, EMC reúne los textos completos de todos los blogs y páginas Web de la compañía, y los compila en una versión de MapReduce, luego utiliza Hadoop para eliminar el código de marcado Web y las palabras no esenciales, y por último pasa la lista de palabras basadas en SQL en bases de datos, donde se realiza un análisis cuantitativo más profundo [32].

Pero ¿por qué no ir más allá? Si se trata de usar la herramienta correcta para la tarea adecuada, lo más interesante sería que las dos tecnologías trabajaran de la mano; a esto se le conoce como persistencia políglota, de la cual se presentan dos casos: “la persistencia políglota se producirá en la empresa cuando diferentes aplicaciones utilizan diferentes tecnologías de almacenamiento de datos. También se producirá dentro de una sola aplicación, donde diferentes partes del almacenamiento de datos de una aplicación tienen diferentes características de acceso” [33].

Se tiene la perspectiva de que “con el tiempo las organizaciones van a empezar a pensar más y más sobre el tipo de persistencia que necesitan para diferentes problemas, y que en última instancia generara que los “Sistemas Gestores de Bases de Datos Relacionales no serán más que una de las opciones disponibles para el manejo de la persistencia de los datos” [34].

VII. Retos para NOSQL

Los retos para el movimiento NoSQL son muchos y muy variados; entre ellos se destacan:

- Realizar una estandarización: uno de los puntos fuertes de movimiento NoSQL es su cantidad de tecnologías; lastimosamente, esto puede convertirse en una debilidad, debido a que "al parecer, hay cerca de 50 motores NoSQL, cada uno con una interfaz de usuario diferente. La mayoría tienen un modelo de datos que es único para ese sistema… Mi gurú empresarial estaba muy preocupado con de la proliferación de motores con tantas singularidades. En contraste, SQL ofrece un entorno estándar" [35]. Para todos es evidente la necesidad de establecer un estándar para las tecnologías NoSQL.

- Superar el "hype": al ser un movimiento que propone un conjunto de tecnologías novedosas, se presenta el fenómeno de "juguete nuevo", es decir, las personas solo se interesan por el tema por ser nuevo, y después de un rato no le prestan atención; sumado a esto la teoría de "si funciona no lo cambie", presenta un complejo reto para las bases de datos NoSQL. Las nuevas tecnologías comienzan con la ingenua euforia de su inventor y su círculo, quienes auto-promueven sus ideas para generar fondos para continuar con el trabajo; el "hype" ayuda a estas tecnologías cumplir esta meta, sin embargo, después de esto casi siempre hay una reacción negativa exagerada a las ideas que no están completamente desarrolladas, y esto conduce inevitablemente a un choque de trenes. Muchas nuevas tecnologías que pasan por este punto se desvanecen, las que sobreviven lo hacen porque alguien encuentra un buen uso para sus ideas básicas [36].

- Demostrar que es algo nuevo y bueno: para ello la comunidad de NoSQL tiene que generar contenidos y apartarse un poco de la parte empresarial y acercarse a la parte académica: "las innovaciones recientes más significativas vinieron de los laboratorios de investigación de varias empresas que hicieron frente a los desafíos urgentes y sin precedentes de tamaño y complejidad en los datos, y los millones de los usuarios que ejecutan varios miles de transacciones por segundo. Estos desarrollos no han sido plenamente anticipados por la comunidad de investigación de bases de datos, y curiosamente ni siquiera por los proveedores de bases de datos tradicionales, cuyas ofertas fueron empequeñecidos productos por la escala y la complejidad de los nuevos dominios de aplicación" [37].

En síntesis, los retos de NoSQL en el momento vienen dados por la novedad del movimiento, y si quiere superarlos se tiene que fortalecer, apoyado en la comunidad académica, establecer estándares y seguir con sus principios.

VII. Conclusiones

NoSQL hace referencia al conjunto de tecnologías en bases de datos que buscan alternativas al sistema de bases de datos relacional, en un contexto donde priman la velocidad, el manejo de grandes volúmenes de datos y la posibilidad de tener un sistema distribuido.

Las tres grandes causas del "bullicio" que está causando NoSQL son: los cambios en el tamaño y la cantidad (e incluso estructura) de los datos, la necesidad de velocidad por parte de los usuarios y la falta de innovación en el área.

Las características que plantean las bases de datos NoSQL responden a las necesidades actuales de las diferentes organizaciones, por lo que se convierten en una alternativa a las tradicionales bases de datos, debido a su capacidad y a la velocidad, que son dos principios que priman hoy en el mundo de los sistemas.

Al movimiento NoSQL le queda mucho camino por delante; las tecnologías tienen que madurar, y para ello se hace necesaria la participación de la comunidad académica; por esto el presente trabajo es un punto de partida para conocer del tema y comenzar a vincularse y tener en cuenta este movimiento.

VIII. Trabajos futuros

Esta investigación se ha enfocado en aspectos generales de las bases de datos NoSQL, teniendo en cuenta como trabajos futuros la investigación en cada una de las taxonomías mencionadas y la comparación entre ellas, para determinar una guía de selección; así mismo, se propone el análisis de una herramienta específica en una tecnología para la posterior creación de aplicaciones que hagan uso de esta herramienta.

Referencias

[1] N. Leavitt, "Will NoSQL Databases Live Up to Their Promise?," Computer, vol. 43, no. 2, pp. 12–14, Feb. 2010.

[2] D. Berndt *et al*. (2012, Apr.), "SiteWit Corporation: SQL or NoSQL that is the Question", Grandon Gill's Website [Online], Available: http://grandon.com/publications/SiteWit_NoSQL.pdf.

[3] E. Codd, "A relational model of data for large shared data banks," Commun. ACM, vol. 13, no. 6, pp. 377–387, Jun. 1970.

[4] S. Drobi (2012, Aug. 3), "Rich Hickey and Justin Sheehy about Datastores, NoSql and CAP", InfoQ [Online], Available: http://www.infoq.com/interviews/rich-Hickey-and-justin-sheehy-about-datastores,-nosql-and-cap.

[5] Professional NoSQL, Shashank Tiwari, John Wiley & Sons, 2011.

[6] Cisco and/or its affiliates (2011, Nov.), "Cisco Global Cloud Index: Forecast and Methodology 2010–2015" [Online], Available: http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns1175/Cloud_Index_White_Paper.pdf.

[7] Pingdom (2012, Jan.), "Internet 2011 in numbers", Pingdom [Online], Available: http://royal.pingdom.com/2012/01/17/internet-2011-in-numbers/.

[8] M. Butter (2012, Aug.), "London Olympics Get Gold Medal for Big Data (Infographic)", NetApp [Online], Available: http://www.forbes.com/sites/netapp/2012/08/08/london-olympics-get-gold-medal-for-big-data-infographic/.

[9] J. Lecat (2010, Sep.), "Jérôme Lecat on scality vol. 1", The NoSQL Tapes [Online], Available: http://nosqltapes.com/video/jerome-lecat-on-scality.

[10] M. Loukides (2012, Feb.), "The NoSQL movement: How to think about choosing a database", O'Really Strata [Online], Available http://strata.oreilly.com/2012/02/nosql-non-relational-database.html.

[11] E. Schurman and J. Brutlag, "The User and Business Impact of Server Delays, Additional Bytes, and HTTP Chunking in Web Search in", Velocity 2009 Conference [Online], 2009, Available: http://www.youtube.com/watch?v=bQSE51-gr2s.

[12] E. Arcos (2009, Apr.), "Oracle compra Sun", ALT140 La guía de geek [Online], Available: http://alt1040.com/2009/04/oracle-compra-sun.

[13] E. Lai (2009, Jul.), "No to SQL? Anti-database movement gains steam", ComputerWorld [Online], Available: http://www.computerworld.com/s/article/9135086No_to_SQL_Anti_database_movement_gains_steam_?taxonomyId=173&pageNumber=2;

[14] M. Stonebraker et al., "The end of an architectural era: (it's time for a complete rewrite)," in Proceedings of the 33rd international conference on Very large data bases, 2007, pp. 1150–1160.

[15] A. Marcus, "The NoSQL Ecosystem", The Architecture of Open Source Applications, 2011.

[16] R. Prasad et al., " RDBMS to NoSQL: Reviewing Some Next-Generation Non-Relational Database's", International Journal Of Advanced Engineering Sciences And Technologies, vol. 11, no. 1, pp. 15-30-

[17] Google Trends (2012, Oct.), "Informe de búsquedas del término: NoSQL", Google [Online], Available: http://www.google.de/trends/?q=nosql&ctab=0&geo=all&date=all.

[18] R. Cattell, "Scalable SQL and NoSQL data stores", SIGMOD Rec., vol. 39, no. 4, pp. 12–27, May. 2011.

[19] G. Burd, "NoSQL", Usenix, vol. 36, pp. 5-12, Oct. 2012.

[20] E. A. Brewer, "Towards robust distributed systems (abstract)," in Proceedings of the nineteenth annual ACM symposium on Principles of distributed computing, New York, USA, 2000, pp. 7–11.

[21] D. Pritchett, "BASE: An Acid Alternative," Queue, vol. 6, no. 3, pp. 48–55, Mayo 2008.

[22] R. Agrawal *et al.*, "The Claremont report on database research," SIGMOD Rec., vol. 37, no. 3, pp. 9–19, Sep. 2008.

[23] R. Sasirekha, "NoSQL, the database for the Cloud", Tata Consulting service [Online], Available: http://www.tcs.com/SiteCollectionDocuments/White%20Papers/Consulting_Whitepaper_No-SQL-Database-For-The-Cloud_04_2011.pdf.

[24] G. DeCandia et al., "Dynamo: amazon's highly available key-value store," SIGOPS Oper. Syst. Rev., vol. 41, no. 6, pp. 205–220, Oct. 2007.

[25] Editores Redis, Redis [Online], Available: http://redis.io/.

[26] P. Membrey etal., The Definitive Guide to MongoDB: The NoSQL Database for Cloud and Desktop Computing. Apress, 2010.

[27] Editores 10gen, MongoDB [Online], Available: http://www.mongodb.org/.

[28] T. Wellhausen (2012, May),"Highly Scalable, Ultra-Fast and Lots of Choices: A Pattern Approach to NoSQL", Tim-Wellhausen [Online], Available: http://tim-wellhausen.de/papers/NoSQL-Patterns.pdf.

[29] Editores The Apache Software Foundation, HBase [Online], Available: http://hbase.apache.org/.

[30] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: a collaboratively created graph database for structuring human knowledge," in Proceedings of the 2008 ACM SIGMOD international conference on Management of data, New York, NY, USA, 2008, pp. 1247–1250.

[31] Neo4j Community, Neo4j [Online], Available: http://neo4j.org/.

[32] J. Jackson (2011, Aug.), "NoSQL Offers Users Scalability, Flexibility, Speed", PCWorld [online], Available: http://www.pcworld.com/article/238868/nosql_offers_users_scalability_flexibility_speed.html.

[33] M. Fowler and P. Sadalage (2012, Feb.),"The future is:NoSQL Databases Polyglot Persistence", Thoughtworks [Online], Available: http://martinfowler.com/articles/nosql-intro.pdf.

[34] S. Leberknight, "Polyglot Persistence", Near infinity [Online], Available: http://www.nearinfinity.com/blogs/scott_leberknight/polyglot_persistence.html.

[35] M. Stonebraker, "Stonebraker on NoSQL and enterprises," Commun. ACM, vol. 54, no. 8, pp. 10–11, Aug. 2011.

[36] J. C. Bezdek, "Fuzzy models - What are they, and why? [Editorial]," IEEE Transactions on Fuzzy Systems, vol. 1, no. 1, pp. 1 –6, Feb. 1993.

[37] G. Feuerlicht, "Database Trends and Directions: Current Challenges and Opportunities". In: DATESO 2010, pp. 163-174, Apr. 2010.