Nuevas arquitecturas NOSQL para el trabajo con gran cantidad de datos

Giovanni Daián Róttoli[1], Marcelo López Nocera[2] y María Florencia Pollo-Cattaneo[3]

Grupo de Estudio en Metodologías de Ingeniería de Software. Facultad Regional Buenos Aires UTN
1. Estudiante Ingeniería en Sistemas de Información, Facultad Regional Concepción del Uruguay UTN
gd.rottoli@gmail.com
2. Magister en Ingeniería en Sistemas de Información y Licenciado en Ciencias Aplicadas, Facultad Regional Buenos Aires UTN. zappapet@yahoo.com
3. Magister en Ingeniería en Software ITBA-UPM. Facultad Regional Concepción del Uruguay y Facultad Regional Concepción del Uruguay UTN.
flo.pollo@gmail.com

Resumen

Las bases de datos tradicionales, de arquitectura relacional, se han estado utilizando durante los últimos años para resolver cualquier problema asociado al almacenamiento y consulta de datos. Sin embargo, con la llegada de Big Data, se presentan nuevos desafíos que estas tecnologías no han podido resolver de manera eficiente. El surgimiento de las bases de datos NoSQL plantea una solución a estas cuestiones. El presente artículo detalla las características de estas soluciones, y un estudio para determinar en qué caso se obtiene mejores resultados, al utilizar estas nuevas arquitecturas trabajando con cantidades masivas de datos.

Summary

Traditional databases, with relational architecture, have been used in the last years to solve any kind of problem related to data storage and management. However, with the arrival of Big Data, new challenges appeared and with these technologies couldn't be efficiently solved. The rise of NoSQL data bases can set a solution to these issues. This article describes the features of these solutions, and introduces a study that determines which case gives the best result, by using these new architectures to work with massive quantities of data.

1. Introducción

Las bases de datos tradicionales parecen no ofrecer soluciones eficientes para un variado universo de nuevos problemas que surgen con la llegada de Big Data (por caso, el análisis en línea de los datos de redes sociales). Esta situación generó el surgimiento de nuevas tecnologías para el almacenamiento de datos, que se engloban dentro del concepto de NoSQL [1].

El primer problema general no resuelto por las bases de datos relacionales, es el de la discordancia de la impedancia (Impedance Mismatch) [1], es decir, la diferencia entre el modelo relacional y las estructura de datos en memoria, lo que implica una traducción necesaria entre ambas representaciones.

Otro problema se relaciona con el hecho de que las estructuras relacionales se basan en tuplas o filas que solo pueden contener tipos de datos simples o primitivos. Otras estructuras de representación de datos, tales como las listas o registros anidados, no son posibles de utilizar dentro de una tupla. Muchas tecnologías NoSQL permiten el uso de estos agregados [2].

Por otra parte, Big Data (BD) trajo consigo el problema del escalamiento del almacenamiento físico de la base de datos, el cual ha tomado dos rumbos: hacia afuera, o hacia arriba. En este último, la necesidad de escalar a servidores

cada vez más grandes y de mayor potencia es evidente, crecimiento que es caro y muy limitado [3]. La solución resulta en la utilización de muchas máquinas pequeñas en clúster, arquitectura que resulta más barata y resiliente (el clúster continua funcionando si un nodo resulta con problemas). Sin embargo, las tecnologías relacionales no fueron diseñadas para ejecutarse sobre clústeres, lo que llevó a que grandes compañías como Google (www.google.com) o Amazon (www.amazon.com), desarrollen sus propias soluciones NoSQL [3].

En el presente trabajo, se define en detalle el problema a tratar, para analizar los datos obtenidos utilizando tecnologías tanto SQL como NoSQL, aplicando esta última en una prueba de concepto para su validación.

2. Definición del Problema

El término NoSQL se hace conocido en el 2009, haciendo referencia a bases de datos que no utilizan el lenguaje ANSI SQL estándar para sus consultas [1], tratándose usualmente de proyectos de código abierto, capaces de correr sobre clústeres, con arquitecturas de procesamiento distribuido y que siguen modelos de datos distintos al modelo relacional tradicional.

Estas tecnologías operan sin un esquema, permitiendo agregar campos libremente a los registros de la base de datos, sin tener que definir cambios en la estructura previamente.

NoSQL es más un movimiento, una nueva tendencia, que una tecnología [4]. De hecho, coexisten varias tecnologías, varios modelos de datos y varios tipos de bases de datos distintas bajo el término NoSQL.

Las dos principales razones para utilizar la tecnología NoSQL son [1]:

- Para mejorar la productividad del programador utilizando la base de datos que mejor partido saque de una aplicación.
- Para mejorar el rendimiento de acceso a datos a través de una combinación de manejo de mayor volumen de datos, reduciendo el tiempo de latencia y mejorando el rendimiento.

Sin embargo, con datos de estructura homogénea y con una cantidad relativamente pequeña, el almacenamiento tradicional con una arquitectura relacional sigue siendo muy eficiente, con lo cual se pueden adoptar las dos arquitecturas en paralelo. Esto se conoce como la persistencia políglota [5], es decir, el uso de diferentes almacenamientos de datos en distintas circunstancias. Por ejemplo, utilizar una base de datos relacional para integración de datos, y una NoSQL para las aplicaciones con la utilización de servicios Web, lo cual permite acceso más rápido y eficiente a grandes cantidades de datos en línea.

En cuanto al modelo de datos no relacional utilizado por NoSQL, el más difundido es el denominado agregado. De hecho, tres de las cuatro tipos de bases de datos NoSQL existentes (Clave-Valor, Documentos, Columna-Familia y Grafos), lo utilizan. A diferencia de la tupla relacional, que solo permite campos con valores simples, el agregado consta de “registros” que permiten la utilización de estructuras complejas dentro de ellos, tales como registros (en el sentido tradicional), listas, arreglos, entre otros. Un ejemplo de esta situación se puede ilustrar con un cliente que posea una lista de pedidos: un solo registro con un arreglo de productos lo puede implementar. Por otro lado, esta implementación no es estática, pudiendo cambiar el enfoque a una lista de facturas, con sus importes, cabecera y detalle, si así lo requiere, en cualquier momento.

Como nueva propuesta, NoSQL se muestra apto para solucionar los problemas que surgen en el tratamiento cotidiano de datos, entre los cuales se puede definir: dificultad para escalar la base de datos [6], [7], [8], bajo rendimiento para grandes volúmenes de datos [1], [6], [7], [8], necesidad de paralelismo de procesamiento [1], [2], [4], discordancia de la Impedancia (los datos en memoria tienen una estructura distinta a la que se almacena en la base de datos física) [1], [5] , [9], dinamismo de las estructuras [1], [6], [7], [8] y finalmente, la complejidad del modelado [1], [2], [4]. Teniendo en cuenta todos estos problemas, se puede relevar y analizar comparativamente la capacidad de respuesta de las diversas tecnologías NoSQL. El resultado de dicho análisis se detalla en la Tabla 1.

Tabla 1. Problemas resueltos por cada tipo de tecnología NoSQL

Problema	Clave- Valor	Columna-Familia	Documentos	Grafos
Dificultad para escalar la base de datos	X	X	X	
Bajo rendimiento para grandes volúmenes de datos	X	X	X	
Necesidad de paralelismo de procesamiento	X	X	X	X
Discordancia de la Impedancia			X	X
Dinamismo de las estructuras			X	
Complejidad del modelado	X	X	X	

Se observa que no todas las tecnologías resuelven todos los problemas y que más de una solución podría aplicarse a un mismo problema, lo que da lugar al ya mencionado concepto de la persistencia políglota [5] [9].

Los principales puntos clave de la persistencia políglota son:

- Implica el uso de diferentes tecnologías de almacenamiento de datos para manejar las diversas necesidades de almacenamiento de los mismos.
- Puede aplicarse para la totalidad de los datos de una empresa o para un subconjunto de ellos.
- El encapsulamiento del acceso a los datos en servicios Web reduce el impacto de almacenamiento de datos en otras partes de un sistema.
- La incorporación de otras tecnologías de almacenamiento de datos aumenta la complejidad en programación y las operaciones, por lo que las ventajas de un buen almacenamiento de datos debe sopesarse con esta complejidad.

Al surgir NoSQL, se pensó que resolvía todos los problemas, incluso los que resuelve SQL tradicionalmente también. Pero no está demostrado que esto sea así [5].

En este contexto, el presente trabajo propone un estudio comparativo del rendimiento para carga y lectura masiva de datos en situaciones diversas para cada una de las distintas tecnologías NoSQL, verificando la conveniencia de elegir una en particular u optar por la persistencia políglota.

3. Solución Propuesta

Para dar respuesta a la problemática definida en la sección 2, se han postulado anteriormente varias soluciones parciales [9]. De hecho, estos problemas se han tratado siempre con SQL [10]. Las soluciones encontradas son eficientes al considerar aspectos puntuales del problema pero, presentan inconvenientes al intentar aplicarlas en un todo [9].

Para verificar la validez, o no, de la utilización de una tecnología NoSQL, se propone un modelo de experimentación (banco de pruebas) para analizar la performance con SQL, y con cada una de las cuatro tecnologías NoSQL, para varios casos de carga masiva y recuperación de datos. El banco de pruebas propuesto se encuentra formado por un universo de 1.000.000 de datos de personas consideradas como clientes de un comercio, 100 datos de productos de una base de stock, 1.000.000 de datos de amistades extraídos de redes sociales y 1.000.000 de datos de compras. Posteriormente, se analizó la performance de carga y de lectura de esos datos para un motor SQL (PostgreSQL) y para cuatro motores NoSQL, como son Redis (Clave-Valor), MongoDB (Documentos), Cassandra (Columna-Familia) y Neo4J (Grafos), y se determinó en qué casos es mejor cada opción. Los resultados experimentales se detallan a continuación en la Sección 4, correspondiente a la prueba de concepto.

4. Prueba de Concepto

4.1. Herramientas de bases de datos utilizadas

Las herramientas de bases de datos utilizadas fueron:

· PostgreSQL 9.3 con PgAdmin III, para la tecnologías Relacional

· MongoDB 2.6 con RoboMongo 0.8.4, para la tecnología NoSQL Documental

· Redis 2.8.11, para la tecnología NoSQL Clave-Valor

· Neo4J 2.1.2, para la tecnología NoSQL de Grafos

· Cassandra 2.0.8, para tecnología NoSQL de tipo Columna-Familia.

Se utilizó programas realizados en Python 2.7.5 para la realización de las consultas, utilizando los controladores existentes para tal fin, tales como PyMongo, Redis-Py y Pycassa.

4.2. Procedimiento

En primer lugar, se generaron datos de acuerdo al diagrama de entidades de la Figura

1, en las cantidades de 1.000.000 (un millón) de Personas, 1.000.000 (un millón) de Compras, 1.000.000 (un millón) de relaciones de Amistad, y 100 (cien) Productos. [11]

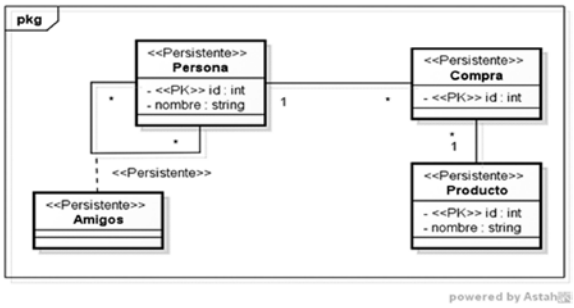

Figura 1: Diagrama de Entidades utilizado para las pruebas

Para la generación de los datos, se utilizó un programa realizado en Python para tal fin, creando archivos CSV que serían accedidos y leídos posteriormente, de manera masiva, utilizando las distintas herramientas que proveen tanto Python como los controladores utilizados, para ser cargados a los distintos motores de bases de datos [11][12].

En cada situación, y debido a las características particulares de cada tipo de tecnología de base de datos, fue necesario adaptar el modelo original para poder ajustarse a los requerimientos de cada una, manteniendo la naturaleza de las relaciones. De esta forma, por ejemplo, en el modelo documental se utilizó la referencia entre documentos mediante claves únicas, no utilizando su particularidad de documentos anidados, para mantenerse fiel a la estructura de la Figura 1. De igual manera, sucedió en cada una de las bases de datos NoSQL utilizadas.

Una vez finalizada la tarea anterior, se realizó una consulta en cada base de datos, para ubicar los productos comprados por los amigos de una determinada persona. Una vez más, se utilizó Python para realizar el programa de consulta y los controladores ya mencionados, al no poder realizar operaciones de INNER JOIN directamente en las tecnologías NoSQL, particularidad de este tipo de bases de datos [12].

La evaluación de los tiempos se realizó sin incluir el tiempo de conexión ni las operaciones del resto del programa, tales como la recuperación de datos desde los archivos CSV, contemplando solamente el código directamente relacionado con la consulta.

4.3. Resultados obtenido por la aplicación del procedimiento

A partir de la aplicación del procedimiento descripto en 4.2, se realizaron las mediciones de los tiempos correspondientes para cada una de las diferentes actividades planificadas.

Los resultados de los tiempos obtenidos se detallan en la Tabla 2.

Tabla 2. Resultados obtenidos al aplicar el procedimiento (medidos en segundos)

Modelo	Base de Datos	Personas	Productos	Compras	Amigos
RDBMS	PostgreSQL	11.617	0.011	84.019	44.256
Clave-Valor	Redis	59.5927	0.055	77.301	63.890
Documentos	MongoDB	28.195	0.014	26.269	29.659
Columna-Familia	Cassandra	495.303	0.124	450.174	496.176
Grafos	Neo4J	62.153	0.655	176.173	108.956

Se observa que, para el caso de las Personas (columna 3), la mejor solución fue SQL. El modelo documental (fila 4) sigue en segundo lugar (pero duplica el tiempo utilizado); mientras que los tres modelos restantes arrojan valores muy altos.

Para las Compras (columna 5), el modelo de mejor rendimiento resultó se Documentos (fila 4), al igual que para el caso de los Amigos (columna 6), donde se puede observar que Columna-Familia (fila 5) y Grafos (fila 6) nuevamente se encuentran alejados del rango de tiempos.

Para el caso de la Consulta (columna 7) el modelo más eficiente resultó Clave-Valor (fila3), debido a sus características de implementación sobre memoria volátil. En este caso, Grafos (fila 6) sigue estando lejos de los valores menores. Sin embargo, para el caso de los Productos (columna 4), tanto SQL como Documentos obtuvieron resultados semejantes.

5. Conclusiones

Con la llegada de las tecnologías NoSQL surge la necesidad de experimentar sobre la verdadera utilidad, en cuanto a performance, de las mismas, sometidas a diversas situaciones.

Como resultado del trabajo realizado se confirma experimentalmente que, la mejor solución a adoptar para encarar de un modo global la gestión de bases de datos con cantidades masivas de información, es la persistencia políglota,

es decir, la coexistencia de RDBMS y NoSQL, utilizando cada una en la situación que resulte con mejor rendimiento en función a la estructura de los datos que se posee, considerando además las características propias de cada tipo de solución.

Como futuras líneas de trabajo, se prevé definir otros casos similares con mayor cantidad de datos, o bien casos con otras categorías y tipos de datos para obtener validaciones empíricas de los mismos.

Otras líneas de trabajo volcadas hacia lo técnico serían repetir el procedimiento anterior con un juego de datos sin normalizar, adaptándose a las características propias de cada tipo de solución, o bien con otros motores, tanto SQL (Microsoft SQL Server, Oracle, MySql, etc.) como NoSQL, sea Clave-Valor (BerkeleyDB, LevelDB, Memcached, Voldemort, Riak, etc.), Documentos (CouchDB, OrientDB, RavenDB, Terrastore, etc.), Columna-Familia (SimpleDB de Amazon, HBase, Hypertable, etc.) o Grafos (FlockDB, Infinite Graph, HyperGraphDB, etc.). Incluso se podría repetir el procedimiento con un equipamiento de nodo único pero más potente, o directamente en clúster.

6. Referencias

[1] SADALAGE, Pramod y FOWLER, Martín. (2013). NoSQL Distilled, A Brief Guide to the Emerging World of Polyglote Persistence. Addison-Wesley. 1ra Edición. Boston, USA.

[2] HECHT, Robin; JABLONSKI, Stefan. (2011). NoSQL Evaluation: A Use Case Oriented Survey. 2011 IEEE International Conference on Cloud and Service Computing. Pages 336-341. Hong Kong, China. 2011.12.12-2011.12.14. ISBN: 978-1-4577-1637-9/11. http://rogerking.me/wp-content/uploads/2012/03/DatabaseSystemsPaper.pdf (verificado el 11-03-2015)

[3] LÓPEZ, David. (2012). Análisis de las posibilidades de uso de Big Data en las organizaciones, Manuscript. Universidad de Cantabria. Santander, España. 75p. On-Line: http://repositorio.unican.es/xmlui/bitstream/handle/10902/4528/TFM%20-%20David%20L%C3%B3pez%20Garc%C3%ADaS.pdf?sequence=1 (verificado el 11-03-2015)

[4] RAMÍREZ ARÉVALO, Helio Henry; HERRERA, Cubides; FRANCINED, John. (2013). Un viaje a través de bases de datos espaciales. Redes de Ingeniería 4(2): 57-69 . Universidad Distrital Francisco José de Caldas. Bogotá, Colombia. ISSN: 2248-762X. On-Line: http://revistas.udistrital.edu.co/ojs/index.php/REDES/article/view/5923/7426 (verificado el 11-03-2015)

[5] NAYAK, Ameya; PORIYA, Anil; POOJARY, Dikshay. (2013). Types of NOSQL Databases and its Comparison with Relational Databases. International Journal of Applied Information Systems (IJAIS) () 5(4): 16-19. Foundation of Computer Science FCS. New York, USA. ISSN: 2249-0868 On-Line: http://research.ijais.org/volume5/number4/ijais12-450888.pdf (verificado el 11-03-2015)

[6] STRAUCH, Ch.(2011). NoSQL Databases. Lecture Selected Topics on Software-Technology Ultra-Large Scale Sites, Manuscript. Stuttgart Media University, 2011, 149 p. On-Line: http://www.christof-strauch.de/nosqldbs.pdf (verificado el 11-03-2015)

[7]DEL BUSTO, Hansel Gracia; ENRIQUEZ, Osmel Yanes. (2012). Bases de datos NoSQL. Revista Telem@tica. 11(3): 21-33. ISSN 1729-3804. On-Line: http://revistatelematica.cujae.edu.cu/index.php/tele/article/view/74/74 (verificado el 11-03-2015)

[8] BUGIOTTI, Francesca; CABIBBO, Luca. (2013). A Comparison of Data Models and APIs of NoSQL Data Stores. Dipartamento d'Ingegnieria dell'Università di Roma. Roma. On-Line: http://www.bugiotti.it/downloads/publications/noamSEBD13.pdf (verificado el 11-03-2015)

[9] NANCE, Cory; LOSSER, Travis, IYPE, Reenu; HARMON, Gary. (2013). NoSQL vs. RDBMS - why there is room for both. Proceedings of the Southern Association for Information Systems Conference. Pages 111-116. Savannah, GA, USA. On-Line: http://sais.aisnet.org/2013/Nance.pdf (verificado el 11-03-2015)

[10] MANNINO, M. (2007). Administración de Base de Datos. MCGRAW-HILL Interamericana. 3ra Edición. ISBN: ISBN 978-970-10-6109-1

[11] RÓTTOLI, Giovanni Daián. (2014).

Datos para Comparativa de Carga y Consulta en Bases de Datos NoSQL. Argentina. On-Line: https://drive.google.com/file/d/0B7rpYh0QA1xgYmt5UXhITE0wbzQ/view?usp=sharing (verificado el 11-03-2015)

[12] RÓTTOLI, Giovanni Daián. (2014). Utilización de No SQL para resolución de problemas al trabajar con cantidades masivas de datos (Anexos A y B). Argentina. On-Line: https://drive.google.com/file/d/0B7rpYh0QA1xgNjl4RmF4VWlZUUk/view?usp=sharing (verificado el 11-03-2015)