A comparison of NoSQL Graph Databases

Germán Andrés Pérez E.[*], *Oswaldo Solarte Pabón*[+].

***Abstract*—NoSQL databases are a broad class of databases management systems that differ from the traditional model of relational databases (RDBMS). This article presents a comparison between different NoSQL graph databases. The comparison has into account a set of criteria that will help users to choose any of these systems. The selected database will be used to implement a prototype of an information retrieval system based on Semantic Web technologies.**

Keywords—Graph Databases; NoSQL; Benchmark.

I. INTRODUCCIÓN

El término *NoSQL*[1] se refiere a "no sólo SQL". Las bases de datos *NoSQL* son una amplia clase de sistemas de gestión de bases de datos que difieren del modelo clásico de bases de datos relacionales (*RDBMS*). La principal diferencia es que no se utiliza *SQL* como lenguaje de consulta. En estos sistemas no se requiere estructuras fijas como tablas para almacenar datos, y las operaciones de tipo *JOIN* no están soportadas. En los últimos años, las bases de datos *NoSQL* han ganado mucha popularidad y se utilizan cada vez más como alternativa a los *RDBMS*. Algunas compañías más sobresalientes de Internet como Facebook o Twitter, utilizan *Cassandra*[2], uno de los proyectos más conocidos en el mundo *NoSQL*. Existen diferentes tipos de bases de datos *NoSQL* para manejar información de acuerdo a las necesidades de los usuarios: Bases de datos documentales, llave-valor, multivalor y las orientadas a columnas y a grafos.

Las bases de datos *NoSQL* orientadas a grafos utilizan nodos, aristas y propiedades como elementos primarios, en contraste a las tablas, filas y columnas del modelo relacional . Este tipo de base de datos está diseñada para datos cuyas relaciones son bien representadas en forma de grafo, es decir, los datos que están interconectados con un número no determinado de relaciones entre ellos. Algunos ejemplos son aplicaciones de redes sociales, transporte público, mapas de carreteras o topologías de red, entre otros.

En este artículo se presenta una comparación de algunas bases de datos orientadas a grafos. Se tendrán en cuenta únicamente herramientas con licencia *open source*. Varios trabajos se han realizado [1], [2], [3], con el objetivo de comparar estas bases de datos. Sin embargo, muchas de estas propuestas se enfocan sólo en el criterio del rendimiento y se analizan características como la velocidad de carga de datos y el tiempo que toman las consultas. Por su parte, en este artículo se estudian otros criterios que también son importantes para elegir la base de datos más adecuada. Algunos de estos criterios son: el ranking de popularidad, el modelo de almacenamiento, los lenguajes de consulta soportados, entre otros. La base de datos seleccionada será utilizada para implementar un prototipo de un sistema de recuperación basado en tecnologías de la Web Semántica.

El resto del artículo está organizado así: en la sección II se describen los trabajos relacionados, en la sección III se definen los criterios a tener en cuenta para elegir una base de datos orientada a grafos. En la sección IV, se describe detalladamente cada una de las bases de datos analizadas y se hace el estudio comparativo de acuerdo a los criterios seleccionados. Finalmente, en la sección V se presentan las conclusiones y trabajo futuro.

II. ESTADO DEL ARTE

Varios trabajos se han realizado con el objetivo de comparar diferentes bases de datos *NoSQL* orientadas a grafos [1], [2], y [3]. En estas propuestas, se compara principalmente el rendimiento de diferentes bases de datos *NoSQL* orientadas a grafos. La falta de estándares en el dominio de este tipo de herramientas hace difícil poder compararlas. Muchos de estos sistemas tienen sus propias características y funcionalidades.

En 2010, *Tinkerpop*[3] empezó a trabajar en *Blueprints*[4], un *API* genérico de Java para bases de datos *NoSQL* orientadas a grafos. *Blueprints* es una colección de interfaces, implementaciones y bancos de pruebas para modelos de datos orientados a grafos. Este proyecto busca crear una interfaz estandarizada que permita unificar las características que tengan en común diferentes sistemas de bases de datos *NoSQL* orientados a grafos.

En [1], los autores comparan el desempeño de cuatro bases de datos *NoSQL* orientadas a grafos: *Neo4j*, *Jena*, *HyperGraphDB* y *Sparksee*. Ellos evalúan el desempeño de cada una utilizando diferentes tamaños y operaciones típicas de grafos. Para la evaluación, utilizan *HPC Scalable Graph Analysis Benchmark*[5] (*HPC-SGAB*). Los autores ejecutan el

[*] Germán Andrés Pérez, Escuela de Ingeniería de Sistemas, Universidad del Valle, Cali, Colombia. german.perez.escobar@correounivalle.edu.co
[+] Oswaldo Solarte Pabón. Escuela de Ingeniería de Sistemas, Universidad del Valle, Cali, Colombia. oswaldo.solarte@correounivalle.edu.co

1 http://nosql-database.org/
2 http://cassandra.apache.org/
3 https://github.com/tinkerpop/
4 https://github.com/tinkerpop/blueprints/
5 http://www.graphanalysis.org/benchmark/index.html

978-1-4799-6717-9/14/$31.00 ©2014 IEEE

benchmark con diferentes pruebas para cada una de las bases de datos llegando a la conclusión de que *Sparksee* fue la base de datos que obtuvo los mejores resultados en casi todas las pruebas. Por otro lado, *Neo4j* obtuvo mejor rendimiento que *Jena* durante las operaciones de recorrido del grafo, pero *Jena* superó a *Neo4j* en las pruebas de carga de datos y escaneo de las aristas. *HyperGraphDB* fue la que obtuvo el desempeño más bajo porque en muchas pruebas no lograron cargar los datos iniciales.

En [2], se comparan diferentes sistemas de bases de datos desde el punto de vista del rendimiento para realizar operaciones de recorrido sobre la estructura del grafo. Los autores realizan su propio *benchmark* basándose en las ideas de *HPC-SGAB*. Las bases de datos que se tuvieron en cuenta son: *Neo4j*, *Sparksee*, *OrientDB*, *Sesame* y *SGDB3*[6]. Se realizan diferentes pruebas dirigidas a operaciones de recorrido local para medir el rendimiento de las bases de datos. Los autores concluyen que los casos de uso con operaciones que requieren recorridos locales en una gran red son más adecuados para los sistemas evaluados que las operaciones que requieren recorridos de toda la estructura del grafo. Al final, a pesar de que *SGDB3* es sólo un prototipo, fue la base de datos que tuvo mejor desempeño en cada prueba. Por otra parte, tanto *Neo4j* como *Sparksee* se mostraron más eficientes que *OrientDB* y que *Sesame*.

Por último, en [3], se comparan cuatro bases de datos: *Neo4j*, *Titan*, *OrientDB* y *Sparksee*. A diferencia de las propuestas anteriores, en donde se utiliza *HPC-SGAB* para realizar las comparaciones, en esta propuesta se usa *GDB* (*Graph Database Benchmark*). Se ejecutan varias pruebas con grafos de diferentes tamaños, entre ellos uno compuesto por 1.500.000 elementos (250.000 vértices y 1.250.000 aristas, *aprox.*) y otro más grande con 3.000.000 de elementos (500.000 vértices y 2.500.000 aristas, *aprox.*). Al final, se concluye que *Neo4j* obtuvo el mejor desempeño en las pruebas relacionadas con la carga de datos y recorridos del grafo y que tanto *Sparksee* como *Titan* tuvieron los mejores resultados en las pruebas de lectura-escritura.

III. DEFINICIÓN DE LOS CRITERIOS

Los trabajos descritos anteriormente sirvieron de guía para analizar los criterios que se deben tener en cuenta antes de elegir una base de datos *NoSQL* orientada a grafos. Dichos criterios son:

- **Ranking de popularidad:** Este criterio permite medir el nivel de popularidad que tiene cada una de las bases de datos que se desean comparar. Para este propósito, se usa el ranking propuesto en la página web de *DB-Engines*[7]. Para calcular el nivel de popularidad, se tienen en cuenta diferentes parámetros: el número de menciones de uso del sistema de base de datos en sitios web, su interés general, la frecuencia de discusiones técnicas acerca del sistema, el número de ofertas de trabajo en que se menciona al sistema, el número de perfiles en redes profesionales y la relevancia que tiene en las redes sociales.
- **Modelo de almacenamiento:** Este criterio se refiere al método utilizado por la base de datos para almacenar, manipular y organizar los datos. La mayor parte de los sistemas de gestión de bases de datos están construidos sobre un modelo de datos concreto, aunque es posible que soporten más de uno. En este trabajo, se tendrán en cuenta únicamente los modelos orientados a grafos y orientados a *RDF*[8].
- **Herramienta para trabajar con *RDF*:** Este criterio hace referencia a la herramienta o *plugin* de trabajo que ofrece la base de datos para almacenar *RDF*. Sin embargo, se debe tener en cuenta que este criterio se usa solo para las bases de datos nativas orientadas a grafos.
- **Lenguajes de consulta:** Las bases de datos a comparar deben soportar el lenguaje de consulta *SPARQL*[9] ya que éste es el lenguaje estándar para la búsqueda de información en grafos *RDF*.
- **Lenguajes de programación:** Este criterio se refiere a los diferentes lenguajes de programación con los que se puede acceder a las bases de datos orientadas a grafos. Para este criterio, se valora en gran medida la cantidad de lenguajes de programación que soporta cada una de las bases de datos.
- **Transacciones:** Una transacción es una unidad de la ejecución de un programa. Puede consistir en varias operaciones de acceso a la base de datos. Para que un sistema se considere transaccional, debe contar con transacciones ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad), en los datos. Para este criterio, es de gran importancia que la base de datos sea 100% transaccional.
- **Sistemas operativos:** Se refiere a la facilidad de instalación en diferentes sistemas operativos que tiene la base de datos. Hoy en día, es muy importante la flexibilidad de un software para poder ser utilizado en diferentes plataformas. Por esta razón, en este criterio, se valora la cantidad de sistemas operativos en los que se pueda instalar.
- **Documentación y respaldo:** La documentación de un sistema es muy importante porque facilita el aprendizaje y la utilización por parte del usuario. Para este criterio, se tendrán en cuenta dos aspectos: el primero, la disponibilidad de tutoriales, cursos, ejemplos, foros, etc. El segundo, la calidad de la información encontrada en los manuales. En este caso se hace una valoración numérica de 1 a 5, siendo 5 el valor más alto.

6 http://ups.savba.sk/~marek/sgdb.html
7 http://db-engines.com/en/ranking_definition
8 http://www.w3.org/RDF/
9 http://www.w3.org/TR/rdf-sparql-query/

IV. REVISIÓN DE CADA UNA DE LAS BASES DE DATOS

En esta sección, se describen dos tipos de herramientas orientadas a grafos. Por un lado, se encuentran las bases de datos nativas orientadas a grafos: *HyperGraphDB*, *Neo4j* y *Sparksee*. Estas bases de datos permiten manejar diferentes tipos de proyectos, en donde la información puede ser representada como un grafo. Las redes sociales, mapas de transporte, cadenas de proteínas, redes de negocio, son algunos ejemplos. Por otro lado, se encuentran herramientas que soportan almacenamiento de *RDF* en donde la información se puede representar también como un grafo: *AllegroGraph*, *Virtuoso*. Estas herramientas pueden ser clasificadas como un caso particular de las bases de datos orientadas a grafos. Se suelen utilizar para realizar todo tipo de proyectos basados en estándares *W3C*[10], en donde el contexto primario es la Web Semántica.

1. HyperGraphDB

HyperGraphDB[11] es una base de datos orientada a grafos escrita en Java y basada en hipergrafos dirigidos. Fue lanzada en el año 2010. Según el ranking de *DB-Engines*, ocupa el puesto 186. *HyperGraphDB* 1.2 es la versión actual de esta base de datos. Soporta transacciones ACI, pero no D, es decir son atómicas, consistentes y aisladas, pero no duraderas [4].

HyperGraphDB no cuenta con un lenguaje oficial de consulta [5]. Sin embargo, es posible realizar consultas de dos maneras: la primera, es mediante una *API* basada en expresiones condicionales que son desarrolladas gracias a las clases del paquete *org.hypergraphdb.query*. La segunda, mediante la clase *HGQuery.hg* que permite manejar una sintaxis más conveniente para trabajar con consultas. *HyperGraphDB* permite integrar un componente para trabajar con *RDF* que se llama *RDF via Sail (*implementación del estándar *RDF* utilizando el framework de *Sesame*). *HyperGraphDB* cuenta con buena documentación[12], tutoriales y ejemplos. Sin embargo, no tiene suficiente respaldo pues hay documentos que se encuentran en la página web que no son oficiales de "HyperGraphDB" sino escritos por usuarios.

2. Neo4j

Neo4j[13] es una base de datos orientada a grafos robusta, escalable y de alto desempeño. Fue desarrollada en Java y su lanzamiento fue en 2007. Según el ranking de *DB-Engines*, ocupa el puesto 22, convirtiéndola en la base de datos orientada a grafos más popular de todas. La versión actual de *Neo4j* es la 2.0.1. Esta base de datos permite manejar un grafo de miles de millones de nodos y de relaciones en un sólo servidor. *Neo4j* realiza transacciones ACID, permitiendo la fiabilidad de los datos [6].

Neo4j cuenta con un lenguaje declarativo de consulta simple pero muy poderoso llamado *Cypher* [7]. Tiene un *plugin* llamado *SparQL Plugin* que permite utilizar la base de datos para almacenar tripletas *RDF*. *Neo4j* contiene una documentación[14] muy amplia y de calidad en su sitio web, así como diferentes tutoriales, cursos, manuales y libros. Además, tiene una gran comunidad de usuarios que la están utilizando para diferentes proyectos.

3. Sparksee

Sparksee[15], antes conocida como *DEX*, es una base de datos orientada a grafos de alto desempeño y gran escalabilidad. Fue escrita en C++ y lanzada en el año 2007. Según el ranking de *DB-Engines*, ocupa el puesto 89. *Sparksee* 5.0 es la versión actual. Esta base de datos soporta transacciones parciales ACID denominadas "aCiD" (no siempre se garantizan aislamiento y atomicidad).

Al igual que *HyperGraphDB*, *Sparksee* tampoco cuenta con un lenguaje propio de consulta [8]. *Sparksee* tiene diferentes métodos para recuperar datos de un grafo. Muchos de ellos devuelven una instancia de la clase *Objects*. *Sparksee* cuenta con un manual de documentación[16] muy completo. Además, la página web contiene diferentes tutoriales, manuales y lecturas que sirven de apoyo.

4. Virtuoso

Virtuoso[17] es una base de datos multimodelo. Fue escrita en lenguaje C y lanzada en el año 1998. Según el ranking de *DB-Engines*, ocupa el puesto 55. *Virtuoso* 7.1 es la versión actual. Soporta diferentes modelos de almacenamiento: el modelo relacional, tripletas *RDF* y datos XML. Virtuoso realiza transacciones ACID.

Virtuoso permite utilizar diferentes lenguajes de consulta [9]. Uno de estos es el lenguaje *SQL,* usado para realizar consultas en modelos de datos relacionales. También soporta *SPARQL*, para consultar datos en *RDF*. *Virtuoso* soporta *XPath*, *XQuery* y *XSLT* para navegar a través de documentos XML. *Virtuoso* cuenta con un manual de documentación[18] bastante completo y de buena calidad. Además, en su página web hay bastantes tutoriales y artículos que sirven de respaldo.

5. AllegroGraph

AllegroGraph[19] es una base de datos de alto rendimiento orientada al almacenamiento de tripletas *RDF*. Fue lanzada en el año 2005. Según el ranking de *DB-Engines*, ocupa el puesto 111, su versión actual es la v4.13. *AllegroGraph* también realiza transacciones ACID y soporta diferentes lenguajes de consulta como *SPARQL*, *RDFS++* y *Prolog* [10]. *RDFS++*

10 http://www.w3.org/
11 http://www.hypergraphdb.org/index
12 http://www.hypergraphdb.org/learn
13 http://www.neo4j.org/
14 http://docs.neo4j.org/chunked/milestone/introduction.html
15 http://www.sparsity-technologies.com/
16 http://www.sparsity-technologies.com/downloads/UserManual.pdf
17 http://virtuoso.openlinksw.com/
18 http://docs.openlinksw.com/virtuoso/contents.html
19 http://franz.com/agraph/allegrograph/

soporta todos los predicados *RDFS* y algunos de *OWL*. *Prolog* es un lenguaje alternativo para *AllegroGraph* que permite especificar consultas declarativamente. *AllegroGraph* cuenta con amplia documentación[20] de instalación, utilización, ejemplos, etc. Además, su página web se actualiza constantemente.

Finalmente, en la tabla 1 se presenta un resumen con la información de las bases de datos orientadas a grafos mencionadas anteriormente. De acuerdo con los criterios presentados en la sección III, y las propuestas [1], [2] y [3], se tomó la decisión de utilizar *Neo4j* como base de datos nativa orientada a grafos y *Virtuoso* como herramienta para almacenar tripletas *RDF*. *Neo4j* se seleccionó porque su ranking de popularidad es el más alto, cuenta con una herramienta para trabajar con *RDF*, tiene su propio lenguaje de consulta, soporta un gran número de lenguajes de programación, es la única que permite realizar transacciones 100% ACID y es la que mejor documentación y respaldo tiene. Por otro lado *Virtuoso* se seleccionó ya que su ranking de popularidad es el más alto, permite diferentes modelos de almacenamiento, soporta más lenguajes de programación, tiene mejor documentación y respaldo. Además es la que más sistemas operativos soporta.

V. CONCLUSIONES Y TRABAJO FUTURO

Las bases de datos *NoSQL* orientadas a grafos son herramientas muy útiles que permiten representar información en forma de grafo, es decir, los datos están interconectados con un número no determinado de relaciones entre ellos. A diferencia de los *RDBMS*, estas bases de datos permiten mayor escalabilidad y no requieren estructuras fijas como tablas para almacenar datos.

En este trabajo se presentó una comparación de diferentes bases de datos *NoSQL* orientadas a grafos. Se definieron algunos criterios que facilitan al usuario la elección de alguna de estas herramientas. Como trabajo futuro se plantea utilizar esta base de datos para implementar un prototipo de un sistema de recuperación basado en tecnologías de la Web Semántica. La base de datos seleccionada servirá de soporte para almacenar anotaciones semánticas sobre documentos de texto.

REFERENCIAS

[1] D. Dominguez-Sal, P. Urbón-Bayes, A. Giménez-Vanó, S. Gómez-Villamor, N. Martínez-Bazán, J. L. Larriba-Pey, "Survey of Graph Database Performance on the HPC Scalable Graph Analysis Benchmark", DAMA-UPC, Universitat Politècnica de Catalunya, Barcelona, España.

[2] M. Ciglan, A. Averbuch, L. Hluchy, "Benchmarking traversal operations over graph databases", Institute of Informatics, Slovak Academy of Sciences, Bratislava, Slovakia.

[3] S. Jouili, V. Vansteenberghe, "An empirical comparison of graph databases", Euro Nova R&D, 1435 Mont-Saint-Guibert, Belgium.

[4] "HyperGraphDB Documentation: Transaction Handling". Disponible: http://www.hypergraphdb.org/learn?page=IntroTransactions&project=hypergraphdb

[5] "HyperGraphDB Documentation: Querying in HyperGraphDB". Disponible: http://www.hypergraphdb.org/learn?page=IntroQuerying&project=hypergraphdb

[6] "The Neo4j manual v2.1.1: Chapter 1. Neo4j Highlights". Disponible: http://docs.neo4j.org/chunked/milestone/introduction-highlights.html

[7] "The Neo4j manual v2.1.1: 7.1. What is Cypher?". Disponible: http://docs.neo4j.org/chunked/stable/cypher-introduction.html

[8] "User Manual, Sparksee by Sparsity Technologies: Query Operations". Disponible: http://www.sparsity-technologies.com/downloads/StartingGuide.pdf

[9] "User Manual, Sparksee by Sparsity Technologies: Query Operations". Disponible: http://www.sparsity-technologies.com/downloads/StartingGuide.pdf

[10] "AllegroGraph 4.13.2 Introduction: Query the Database". Disponible: http://franz.com/agraph/support/documentation/current/agraph-introduction.html

20 http://franz.com/agraph/support/documentation/current/agraph-introduction.html

	HyperGraphDB	Neo4j	Sparksee	Virtuoso	AllegroGraph
Ranking de popularidad	186	22	89	55	111
Modelo de almacenamiento	Graph DBMS	Graph DBMS	Graph DBMS	- RDF store - Relational DBMS - XML DBMS	RDF store
Herramienta para trabajar con RDF	RDF via Sail	SparQL Plugin	No tiene		
Lenguajes de consulta	No tiene	Cypher	No tiene	- SQL - SPARQL - XPath - XQuery - XSLT	- SPARQL - RDFS++ - Prolog
Lenguajes de programación que soporta	- Java	- Java - JavaScript - Python - .Net - Ruby - PHP - Perl - Scala - Go - Clojure - Groovy	- Java - C++ - Python - .Net	- Java - JavaScript - C - C++ - C# - Perl - PHP - Python - Ruby - Visual Basic	- Java - Python - C# - Ruby - Clojure - Scala - Lisp - Perl
Transacciones	ACI	ACID	aCiD	ACID	ACID
Sistemas operativos	- Linux - Windows - Mac OS	- Linux - Windows - Mac OS	- Linux - Windows - Mac OS - iOS, Android, BB10	- Linux - Windows - Mac OS - Unix: FreeBSD, AIX, HP-UX, Solaris	- Linux - Windows - Mac OS
Documentación y respaldo (1 a 5)	3	5	4	5	4

Tabla 1. Resumen de cada una de las bases de datos