ANÁLISIS COMPARATIVO DE DOS BASES DE DATOS SQL Y DOS BASES DE DATOS NO SQL

JOSE EDWIN SALAZAR CARDENAS

UNIVERSIDAD TECNOLOGICA DE PEREIRA

FACULTAD DE INGENIERÍAS

PROGRAMA DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN

PEREIRA

2014

ANÁLISIS COMPARATIVO DE DOS BASES DE DATOS SQL Y DOS BASES DE DATOS NO SQL

JOSE EDWIN SALAZAR CARDENAS

PROYECTO DE GRADO

DIRECTOR DE PROYECTO

DOCENTE OMAR IVAN TRÉJOS BURITICÁ

UNIVERSIDAD TECNOLOGICA DE PEREIRA

FACULTAD DE INGENIERÍAS

PROGRAMA DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN

PEREIRA

2014

DEDICATORIA

La educación es el arma más poderosa que puedes usar para cambiar el mundo.

Nelson Mandela

En primer lugar a Dios quien me ha guiado, dado la fortaleza y perseverancia necesaria para alcanzar esta meta y llegar hasta este punto, a mis Padres, pilares fundamentales quienes me inculcaron valores, principios, perseverancia, coraje, responsabilidades y en especial mucho amor, mis Hermanos, Familiares y a una persona muy especial que estuvo en esta etapa de mi carrera por su confianza, apoyo incondicional, y por todas las enseñanzas que me han dado.

AGRADECIMIENTO

Un agradecimiento especialmente a mi mamá, nunca dejo de ayudarme, hasta en las cosas más mínimas siempre estuvo a mi lado apoyándome, preocupada y dándome mucha motivación para seguir adelante y poder culminar mi carrera con éxito.

A mi papá, con su comprensión, carácter me dio muchos ánimos para no dejarme caer, fue mi inspiración.

A mis hermanos, que siempre me apoyaron y tuvieron preocupado por mis unas de las razones por la cual estoy en este punto de mi vida, a puertas de este título profesional.

Quiero agradecer al Ingeniero Omar Iván Trejos Buriticá, director de proyecto de grado por guiarme, corregirme y su ayuda incondicional en toda la elaboración de este proyecto de grado.

A todo mis compañeros por la amistad que me brindaron durante toda esta etapa de la vida universitaria.

A todos los profesores de la carrera que a lo largo de esta me brindaron todos sus conocimientos para mi formación académica y con ello poder formarme como excelente ingeniero de sistemas.

TABLA DE CONTENIDO

LISTA DE TABLAS

LISTA DE ILUSTRACIONES

1 GENERALIDADES

1.1 PLANTEAMIENTO DEL PROBLEMA

Actualmente en el año 2014 la información almacenada en bases de datos sql se presenta desde el inicio de los años 70 que se han desarrollado como herramientas para la información utilizando un modelo entidad/relación, sin embargo al incremento de grandes cantidades de información y los requerimientos y demanda actuales como rapidez o velocidad de procesamiento en la ejecución y las grandes cantidades de volúmenes de datos basado en la idea de principal de big data, que hace referencia al proceso de analizar grandes cantidades de datos que surgen las bases de datos NOSQL en el 2009, que posee una estructura específica para su almacenamiento y manipulación de datos, que manejan columnas y tendrían mucha fiabilidad en el ámbito de rapidez y ejecución. Las bases de datos sql efectúan diversas tareas como consultas, agregar información, o con el fin de recuperar información de manera rápida y sencilla, pero cuando se tienen grandes cantidades de datos esta tiende a disminuir el tiempo de ejecución y de respuestas.

El desconocimiento de las bases de datos NOSQL podría limitar al desarrollador o al que trabaja en este campo en las herramientas tradicionales de bases de datos relacionales. En la actualidad se ha popularizado a nivel de grandes empresas como Twitter, Facebook, Google que se han visto en la necesidad de almacenar datos complejos de requerir tiempo de carga de los mismos en sus servicios, cada día de los desarrolladores prueban nuevos escenarios en los que las bases de datos NOSQL funciona muy bien principalmente en el desarrollo para almacenar muchos datos se habla de terabytes e incluso petabyte de información. Pues que la información es continua y constante y exponencialmente.

El desarrollo de este proyecto se basara en la investigación documental y bibliográfica aplicando y presentando un análisis comparativo entre rendimiento de las bases de datos NOSQL y una de tipo SQL demostrando un punto referencial la facilidad, para manejar datos NOSQL a través de análisis que permitan la obtención de datos e interpretación.

Conociendo las bondades y limitaciones las bases NOSQL los desarrolladores pueden evaluar las circunstancias en las que se pueden aprovechar las ventajas de que estas tienen sobre las bases de datos relacionales tradicionales en determinados escenario. Mediante el análisis y comparación de costo y rendimiento entre una bases de datos relacional y una NOSQL.

1.2 OBJETIVOS

1.2.1 OBJETIVO GENERAL

Realizar un análisis y estudio comparativo entre las bases de datos NOSQL en cuanto al rendimiento.

1.2.2 OBJETIVOS ESPECÍFICOS

Presentar un estudio comparativo de los sistemas de bases de datos SQL y NOSQL para determinar la mejor opción.

Determinar algunos parámetros en la evaluación comparativa de bases de datos SQL y NOSQL.

Analizar las ventajas y desventajas de los motores de bases SQL frente a los NOSQL.

Analizar los resultados obtenidos al comparar las dos bases de datos SQL y NOSQL.

1.3 JUSTIFICACIÓN

Las bases de datos NOSQL se han popularizado a nivel de grandes empresas como Facebook, Twitter, Google que se han visto en la necesidad de almacenar grandes volúmenes de datos, obteniendo logros como aplicabilidad, de alto rendimiento hoy en día, un nuevo enfoque que viene ganando reconocimiento por su versatilidad y escabilidad, son las bases de datos NOSQL ya que no imponen una estructura de datos en forma de tabla y relaciones sino que maneja columnas por lo tanto son más flexible y permiten almacenar información teniendo ventaja en la escabilidad que hace referencia al procesos de analizar grandes volúmenes de datos. Se observa que esta nueva tecnología es de grande ayudas para el manejo de información y su aplicación se expande a diferentes sectores de la sociedad donde el manejo de datos es esencial.

1.4 DISEÑO METODOLÓGICO

Este proyecto se basara en la investigación documental y bibliográfica aplicando y representando un análisis, se presentaran herramientas para la comparabilidad de bases de datos SQL y NOSQL para la solución de bases de datos en almacenamientos en modelo/relación.

1.5 HIPÓTESIS

¿Es posible Realizar un análisis y estudio comparativo entre las bases de datos NOSQL en cuanto al rendimiento?

1.6 VARIABLES

Tabla 1. Criterios de comparación entre las bases de datos.

Bases de datos	Relacional	NOSQL
Modelo de almacenamiento de datos	Almacenamiento con filas y tablas	Varía dependiendo del tipo de base de datos NOSQL. Clave-valor. Similar a la relacional, pero tiene solo dos columnas(“clave y valor”). Documento. Almacena todos los datos juntos en un documento que puede ser anidados jerárquicamente, en JSON,XML u otro formato.
Esquemas	Tipo estructura de datos, se fijan por adelantado. Si se desea agregar otro atributo, se realiza mientras la base de datos esta fuera de línea.	Dinámico. atributos se añaden sin necesidad de bajar el servicio.
Escalado	Solo vertical, o sea un servidor mas poderoso Es posible la transmisión sobre muchos servidores, pero generalmente significa ingeniería adicional	Horizontalmente, para aumentar la capacidad se deben añadir más nodos al servidor, y a la base de datos se propaga automáticamente.
Tolerancia a fallos	Bajo. Fallo en el nodo y generalmente hará fallar la consulta.	Alta. Configurados para que la perdida de algunos nodos no interrumpa funcionamiento global.
Ejemplos	MUSQL, Oracle, Database, SQLServer.	Mongodb, HBase, Casandra.

- SQL server: El motor de bases de datos de Microsoft, inicialmente fue adquirido de Sybase por 1989. Con el paso de los años SQL Server ha evolucionado actualmente posicionarse entre las bases de datos más populares.

- Oracle: tuvo su origen en 1979 en la empresa SDL, para con el tiempo convertirse en la base de datos más usada a nivel empresarial. Oracle ofrece el conjunto de herramientas más completo que se va desde la base de datos, aplicaciones comerciales, herramientas de desarrollo, herramientas de soporte de decisiones o business inteligentes.

- Casandra: desarrollada inicialmente por Facebook y luego entregado a la fundación apache, Casandra es un sistema de bases de datos distribuida que permite almacenar cantidades muy grandes de información en un entorno distribuido sin punto de fallo, es decir en sistemas de replicación en el que todo los nodos son iguales.

- MongoDB: una base de datos documental, de alto desempeño, no utiliza esquema de bases de datos. Permite almacenar la información de forma mas natural mediante documentos auto contenidos, es decir al no usar tablas con relaciones cada unidad de datos contiene en si mismo las dependencias necesarias.

1.7 INSTRUMENTOS

Se aplicara entrevista a las personas que se muevan en este espacio de desarrolladores con NOSQL y SQL. Obtener información de forma oral y personalizada sobre acontecimientos vividos y aspectos subjetivos.

1. ¿Cómo es su experiencia con las bases de datos SQL y NOSQL?
2. ¿Qué sistema de bases de datos nos recomienda?
3. ¿Por qué aparecen las bases de datos NOSQL?
4. Pero, ¿en qué se diferencian exactamente?
5. ¿En qué motor de bases de datos NOSQL ha programado?
6. ¿Por qué se llama “Relacional “al modelo de BD más utilizado?
7. ¿Las bases de datos relacionales han muerto?
8. Las bases de datos NOSQL ¿Son necesarias para cualquier desarrollo?
9. ¿Qué razones pueden conducirnos a la necesidad de integrar una solución NOSQL?
10. ¿Los grandes de la informática (Facebook, Twitter, Google, Amazon, Yahoo) Por qué las usan?
11. ¿Debo aprender a usar bases de datos NOSQL?
12. ¿Ejemplos de bases de bases de datos NOSQL?

2 MARCO REFERENCIAL

2.1 MARCO TEÓRICO

Se analizara el tema fundamental del análisis comparativo de las bases de datos SQL y NOSQL.

Una base de datos es un conjunto de información estructurada en registros y almacenada es un soporte electrónico legible desde un ordenador. Cada registro constituye una unidad autónoma de información que puede estar a su vez estructurada en diferentes campos o tipos de datos que se recogen en dicha base de datos.

Es un soporte estructurado de datos almacenados en un formato electrónico. Frente a la información impresa tiene como principal ventaja el permitirnos recuperar exactamente la información deseada y hacerlo de distintas formas. Además, podemos modificar muchos datos en muy poco tiempo. Todo ello es posible gracias a que el formato electrónico nos permite realizar determinados procesos (consultar datos, ordenarlos, imprimir informes de forma rápida y eficaz).

- ✓ Tabla

Una tabla es un objeto que almacena datos en filas y en columnas. Las filas se denominan registros y las columnas campos. Los datos almacenados en una tabla se refieren a un tema determinado dentro de la base de datos, por ejemplo, datos de los libros, autor, índice, y los diferentes grados y de las materias principales. Base de datos está determinada por lo consistentes y lógicas que sean las tablas que implementemos. Como mostramos en la siguiente imagen.

En la figura queremos dar un ejemplo que trata una serie formada por una tabla bidimensional por registros y por campos en la que recogen datos.

- ✓ Consultas

  Las consultas tienen como propósito recuperar la información almacenada en las tablas. Con esta breve descripción podríamos pensar… ¿y por qué no la miramos directamente en ellas? Pues bien, la ventaja se encuentra en la posibilidad que ofrecen las consultas de filtrar la información y mostrar sólo aquellos datos que interesen en cada caso.

Las consultas se forman a partir de diferentes expresiones que nos permitirán relacionarnos con la base de datos para extraer información de una o varias tablas.

- ✓ Formularios

Un formulario es un formato usado para adicionar, modificar o consultar información bajo criterios personalizados por el usuario.

Los formularios proporcionan listas desplegables, instrucciones, controles de desplazamiento y gráficos que ayudan a los usuarios a trabajar con los datos. De un modo u otro, los formularios hacen que esta tarea sea más agradable. Como apreciara en la siguiente figura

- ✓ Informes

Un informe es usado para imprimir los registros almacenados en una base de datos, utilizando un formato personalizado por el usuario. Los informes permiten agrupar registros, mostrar totales para los grupos o para el informe completo, etc. En la figura mostramos la apariencia típica de un informe creado.

Habitualmente, los informes se suelen construir a partir de los resultados obtenidos de la ejecución de consultas. De esta forma combinamos la posibilidad de seleccionar sólo los datos que deseemos que nos ofrecen las consultas con la ventaja de imprimirlos que aportan los informes.

2.1.1 Evolución de las bases de datos

El almacenamiento de datos se ha venido desarrollando de manera convencional por bases de datos relacionales desde 1970 Edgar Fram Codd propuso el primer modelo relacional, aunque es la compañía Oracle encargada de introducirlo de manera comercial más adelante en 1980 esa bases de datos son manejadas por un lenguaje de consulta estructurados DQL que permite, la manipulación e integridad de la información que se almacena de forma estructurada.

Las bases de datos relacionales, Es una colección de datos cuya característica principal es que los datos pueden almacenarse y administrarse en forma de tablas. Al hablarse de bases de datos relacionales, significa que se pueden crear relaciones entre las tablas de las bases de datos. Una relación entre tablas consiste en que algunos registros de una tabla tengan

datos en común con registros de otras tablas, permitiendo un manejo más eficiente y sin redundancia.

Este modelo considera la base de datos como una colección de relaciones. De manera simple, una relación representa tablas que no es más que un conjunto de filas, cada fila es un conjunto de filas, cada fila es un conjunto de campo representa un valor que interpretado descubre el mundo real. Cada fila también se puede denominar tupla o registro y a cada columna también se le puede llamar campo o atributo.

Las bases de datos relacionales pasan por un proceso al que se le conoce como normalización de una base de datos, el cual es entendido como el proceso necesario para que una base de datos sea utilizada de manera óptima.

Existen nuevos enfoques hacia almacenamiento de información de bases de datos, que implican una mejora relacional sin necesidad de remplazarlas. Bases de datos con nuevos enfoques están orientadas a almacenar información de diversas topologías, y no utilizar modelos de tablas usadas en las bases de datos relacionales.

Es de destacar la importancia de un buen diseño. Un diseño apresurado o simplemente bosquejado puede mostrarse inservible en el momento de la implementación, por tal motivo suele diseñarse en niveles, puede existir un nivel conceptual, en la cual se contempla una estructura abstracta y no implementable directamente pero que debe recoger las características básicas del sistemas, además de tener en cuenta los requerimientos proporcionados por el usuario.

2.1.2 Modelos de Datos

El modelo de bases de datos es un conjunto de herramientas conceptuales para describir datos, las relaciones entre ellos, la semántica asociada entre ellos así como las restricciones de consistencia.

Entre los modelos de bases de datos más conocidos se encuentran dos clasificaciones.

- Modelos lógico basados en objetos:
- Modelo Entidad-Relación (ER).
- Modelo Orientado a Objetos
- Modelos de datos semánticos
- Modelo funcional de datos

Modelo lógicos basados en registros:

- Modelo relacional
- Modelo de red
- Modelo jerárquico

2.1.3 Modelo relacional

Este modelo permite representar la información del mundo real de una manera intuitiva, introduciendo conceptos cotidianos y fáciles de entender por cualquier inexperto. Asimismo, mantiene información sobre las propias características de la base de datos (metadatos), que facilitan las modificaciones, disminuyendo los problemas ocasionados en las aplicaciones ya desarrolladas. Por otro lado, incorpora mecanismos de consulta muy potente, totalmente independiente del S.G.B.D., e incluso de la organización física de los datos; el propio S.G.B.D. es el encargado de optimizar estas preguntas en formato estándar, a sus características propias de almacenamiento.

2.1.4 Modelo Entida-Relación

Modelo de datos: entidad/relación. El proceso inicial en el diseño de una base de datos es la creación de un modelo de datos, siendo este la representación en escala de la realidad además refleja la estructura de negocio de la organización, por medio de datos y relaciones.

EL modelo de datos entidad-relación (E-R) es útil para hacer corresponder los significados e interacciones de las empresas del mundo real con su esquema conceptual.

El modelo entidad relación utiliza la notación que se observa en la figura 1, esta tiene en cuenta conceptos como:

- Entidad: Objeto del mundo real distinguible de otros objetos. Una entidad se describe usando un conjunto de atributos.
- Conjunto de entidades: Una colección de entidades similares.
- Relación: Asociación entre dos o más entidades.
- Atributos: Son las propiedades que caracterizan un conjunto de entidades.
- Conjunto de relaciones: colección de relaciones similares.

2.2 NOSQL

Las bases de datos NOSQL son sistemas de almacenamientos de información que no cumplen con los esquemas entidad/relación es decir que no se imponen una estructura de datos en forma de tablas y relaciones entre ellas, por lo tanto estos sistemas son más flexibles, ya que nos permiten almacenar información. El termino NOSQL fue acuñado a

principios de 1999 por un empleado de Racks Pace, Eric Evans, quien intento describir el surgimiento de un número creciente de bases de datos no relacionales y distribuidos.

La cantidad de información manejada por comunidades de redes sociales, buscadores, ámbitos de la Web es abrumador lo que ha hecho que surjan nuevas arquitecturas de almacenamiento de información, que deben de alto rendimiento, escalables y distribuidas.

Bases de datos NOSQL. Prototipos diseñados para el manejo de bases de datos no relacionales, uno de ellos es bigtable, un sistema de distribución y almacenamiento que permite gestionar datos en una escala muy grande que creo Google para gestionar sus bases de datos en el año 2004 con el fin de escalar peta byte de datos.

Estas bases de datos no relacionales no tienen garantías ACID es decir, no proporcionan garantías en cuanto a instrucciones para la realización de una transacción. Tampoco ofrecen las sentencias "Join" que permiten combinar registros de dos o más tablas en una base de datos relacional. Pero implementan características BASE como son indicadas a continuación:

- Basic Availble**:** el almacén funciona la mayoría del tiempo incluso ante fallos gracias al almacenamiento distribuido y replicado
- Soft state: Los almacenes no tienen porque ser consistentes ni sus réplicas en todo momento.
- Eventual consistencia**:** la solicitud debe estar en un estado conocido.

Para analizar las principales diferencias que se dan entre las características de ACID y BASE que representaran en la siguiente tabla.

Tabla 2. Características de ACID y BASE

ACID	BASE
❖ Aislamiento ❖ Consistencia fuerte ❖ Poca disponibilidad ❖ Evolución difícil (Escalabilidad) ❖ Transacciones anidadas ❖ Requiere “commit”	❖ Disponibilidad ❖ Consistencia débil ❖ Intuitivo ❖ Evolución más fácil ❖ Respuestas más rápidas ❖ Mejor esfuerzo

Entre las características que posee NoSQL se encuentra que no presentan esquemas, tienen fácil soporte de replicación, API simple, eventualmente consistente (conocido como BASE, y contrario al concepto de ACID) y contienen enormes cantidades de datos.

Para una mayor profundidad y calidad sobre las bases de datos NoSQL que hemos mencionado al principio de este trabajo realizado destacamos las comparaciones.

2.2.1 No relacional vs. Relacional

Las bases de datos no relacionales son listas de datos almacenados en una sola tabla sin definir relaciones entre los registros. Por otro lado las relacionales reparten los datos en varias tablas más pequeñas eliminando datos duplicados y asegurando consistencia y estableciendo restricciones y relaciones con otras tablas por medio de claves primarias y foráneas; esto genera que se ocupe menos espacio ya que no tiene redundancias y hasta cierto punto es conveniente esta repartición ya que de otro modo, si se realiza un SELECT múltiple en la no relacional se tendría que recorrer la única tabla varias veces para devolver todos la información debido a duplicidad en ciertos datos.

2.2.2 Distribuida vs. Centralizada

Tener las bases de datos almacenadas en varios nodos para poder distribuir la carga en estos se denomina distribución (o descentralización), es decir que el almacenamiento se realiza en múltiples partes en una localización física, en una red interconectada o a través de internet como es el caso de cloud databases, sin compartir memoria o discos entre los nodos; en cambio en una base de datos centralizada se tiene un bus común a través de todos los nodos que comparte memoria, ésta se encuentra en una sola ubicación lo que facilita la administración, pero no la escalabilidad ya que genera cuellos de botella cuando residen muchas aplicaciones o usuarios.

2.2.3 Código abierto vs. Cerrado

Como su nombre lo indica, abierto permite la visualización del código fuente por parte de los usuarios para su modificación creando grupos colaboradores para mejorar el código y compartirlo con las demás personas y puede ser distribuido gratuitamente o pagado. No puede ser cobrado y la licencia debe ser libre; el código cerrado (o propietario) puede ser cobrado pero no permite el acceso al código fuente. Como consecuencia del código abierto en NoSQL, existe una gran oferta de propuestas en el mercado.

- Ejemplos de bases de datos propietarias: BigTable de Google, DynamoDB y simplementeDB de amazon.

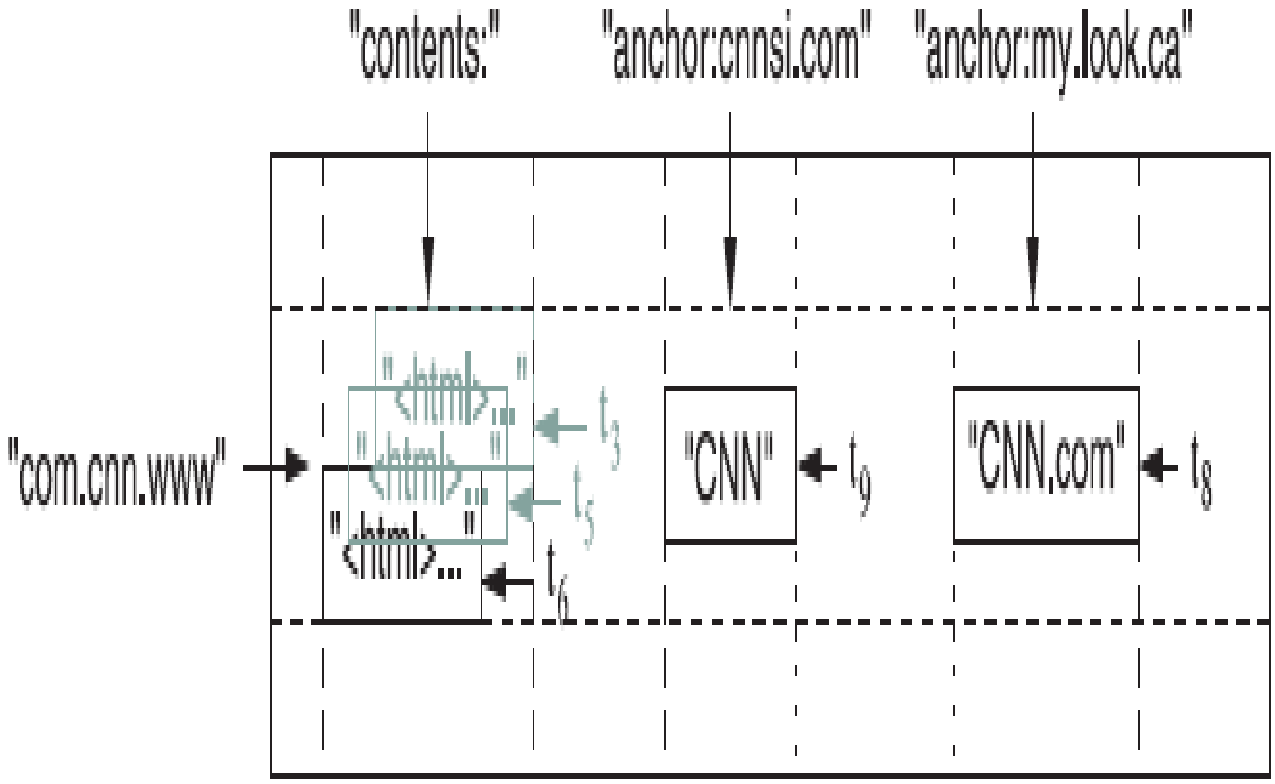
- Ejemplos de bases de datos libres: Cassandra de Facebook, CouchDB de Apache, Redis, Neo4j, MongoDB.

- Horizontalmente escalable vs. verticalmente: escalar horizontalmente significa obtener más nodos para un sistema, como adquirir un nuevo computador para una aplicación distribuida o comprar más servidores Web logrando crear cluster. En cambio, escalar verticalmente significa incrementar el número de recursos a un único nodo del sistema como adquirir más CPUs o memoria para un computador o servidor. (Escalabilidad, 2012)

En los últimos dos años y medio que hemos diseñado, implementado y desplegado un sistema de almacenamiento distribuido para gestionar datos estructurados en Google llama BigTable. Está diseñado para escalar de forma fiable a petabytes de datos y miles de máquinas.

BigTable ha logrado varias metas: amplia aplicabilidad, escalabilidad, alto rendimiento y alta disponibilidad. BigTable es utilizado por más de sesenta productos y proyectos de Google, incluyendo Google Analytics, Google Finance, Orkut, Personalizados Buscar, Writely y Google Earth. Estos productos utilizan BigTable para una variedad de cargas de trabajo más exigentes, que van desde trabajos de procesamiento por lotes rendimiento orientada a sensibles a la latencia de la porción de datos a los usuarios finales.

Ilustración 1. Imagen BigTable



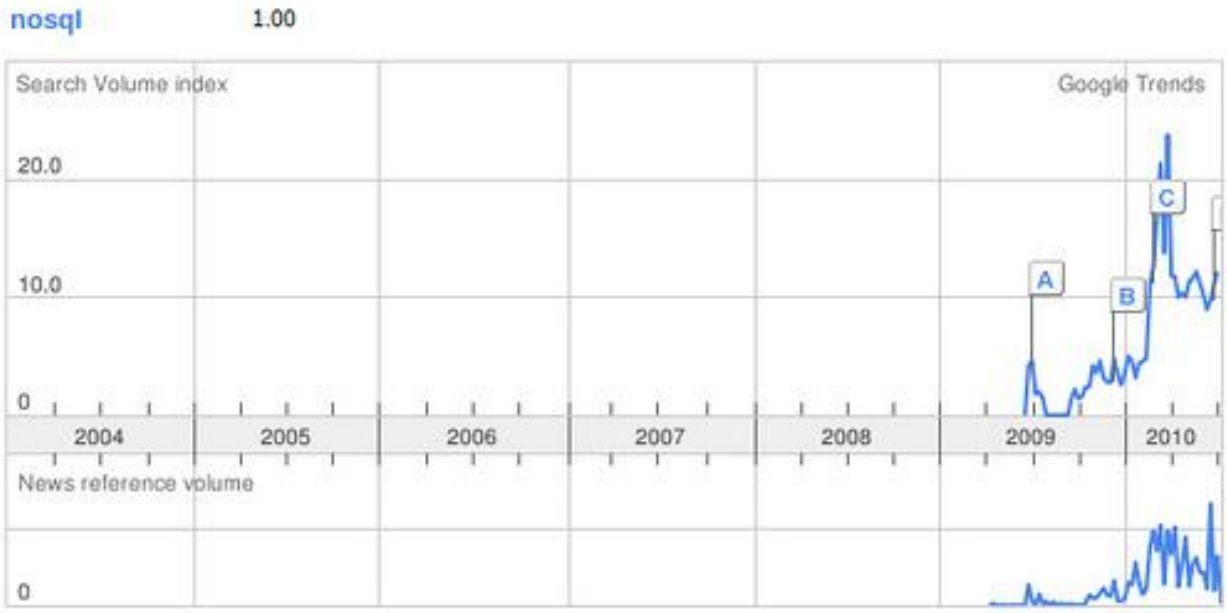
Fuente:http://www.cnet.com/news/is-google-bringing-bigtable-out-of-the-closet/

BigTable no admite un modelo de datos relacional completa en cambio, proporciona a los clientes con un modelo de datos simple que soporta control dinámico sobre diseño de datos y formato, y bajos clientes al razonar acerca de las propiedades de localidad de los datos representados en el almacenamiento subyacente. Los datos son usados en nombres de filas y columnas que pueden ser cadenas arbitrarias. BigTable también trata los datos como cadenas no interpretados, aunque los clientes a menudo serializan diversas formas de estructurarse y datos semi-estructurados en estas cadenas. Los clientes pueden controlar la localización de sus datos a través de elecciones cuidadosas en sus esquemas.

Por último, los parámetros de esquema BigTable permiten clientes controlar dinámicamente

En la actualidad se ha difundido el uso de bases de datos NoSQL, como se muestra en la siguiente figura, debido a la utilidad para la creación de redes sociales, sitios de comercio electrónico, etc., como por ejemplo: Amazon, Twitter, Facebook y Google.

Ilustración 2. Imagen computación en la nube



Fuente:http://www.como-aprender-a-usar-el-software-libre-en-5-pasos.chil.org/page/5-pen-vs-nube

Una de las últimas tendencias en el ámbito del Software libre es la computación en la nube, ya que este novedoso sistema permite guardar información en la nube, en la red, sin tener que usar pen o discos duros externos. Esta nueva posibilidad de biblioteca en la red puede abaratar costes e incrementar la productividad en las empresas o en las administraciones públicas. Ya que sabemos que las bases de datos NoSQL viene tomando gran fuerza en el ámbito de almacenamiento y que mejor que utilizarlas en las nubes, para almacenar gran cantidad de volúmenes de datos.

Las bases de datos NoSQL parten de la base en la que las “tablas” no existen como tal, sino que la información se almacena de forma distinta, generalmente como clave-valor, como una tabla en la que las columnas son dinámicas, pueden cambiar sin perder la agrupación de la información, así es que puedo tener Personas con más atributos que otras, puedo cambiar la estructura de mi información dinámicamente sin tener que re-diseñar todo de nuevo.

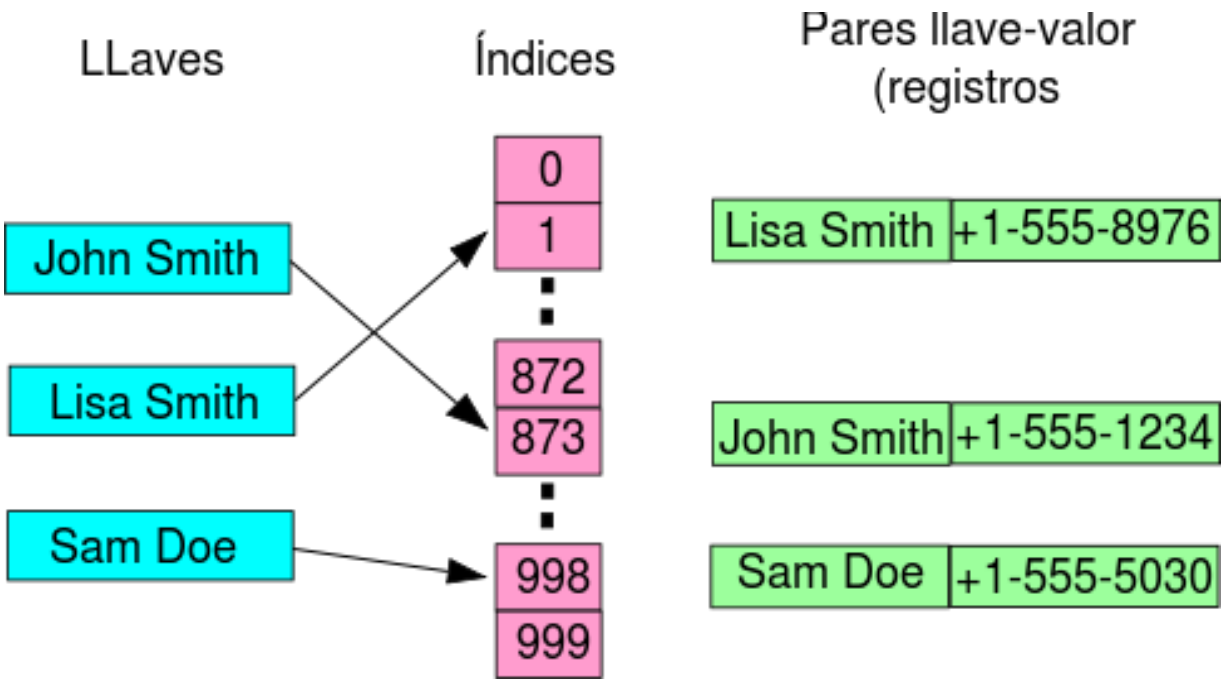
2.2.4 Modelos de bases de datos no relacionales en general hay 4 tipos de bases de datos NoSQL.

Dependiendo de cómo almacenan la información cada uno de ellos con características como mostraremos a continuación:

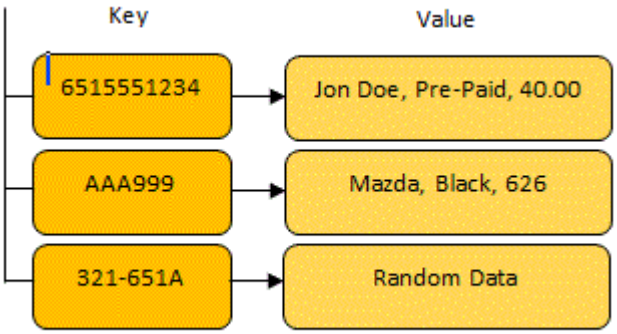
2.2.5 Key-Value:

clave-valor es la forma más típica, como un HashMap donde cada elemento está identificado por una llave única, lo que permite la recuperación de la información de manera muy rápida. Normalmente el valor se almacena como un objeto BLOB. De esta forma el tipo de contenido no es importante para la base de datos, solo la clave y el valor que tiene asociado. Son muy eficientes para lecturas y escrituras, además de que pueden escalar fácilmente particionado los valores de acuerdo a su clave, por ejemplo aquellos cuya clave está entre 1 y 1000 van a un server, los de 1001 a 2000 a otro, etc. Muchas de ellas están basadas en la publicación de Google acerca de su BigTable y de Amazon.

Ilustración 3. Imagen almacen key-Value



Fuente: http://es.wikipedia.org/wiki/Tabla_hash



Fuente:http://www.ingenioussql.com/tag/key-value-store/, http://sg.com.mx/revista/42/nosql-la-evolucion-las-bases-datos#.VHEVSPmG9Ss

Recuerda que los almacenes key/value son extremadamente rápidos pero no permiten consultas complejas más allá de buscar por su clave. Por lo que si tu aplicación necesita consultas complejas, este tipo de base de datos no se ajusta a tus necesidades .Existen implementaciones de estos almacenes solo en memoria, tales como: memcached, Oracle Coherence, JBoss Cache y WebSphere eXtreme Scale. Estas soluciones normalmente funcionan junto a un RDBMS actuando solo como un cache complementario. Además existen otras implementaciones que sí son persistentes, es decir que realmente guardan datos en filesystem y no sólo en memoria por lo que se les puede usar sin un RDBMS. Las más usadas son VMWare Redis, Amazon SimpleDB, Oracle BerkeleyDB y Tokyo Cabinet.

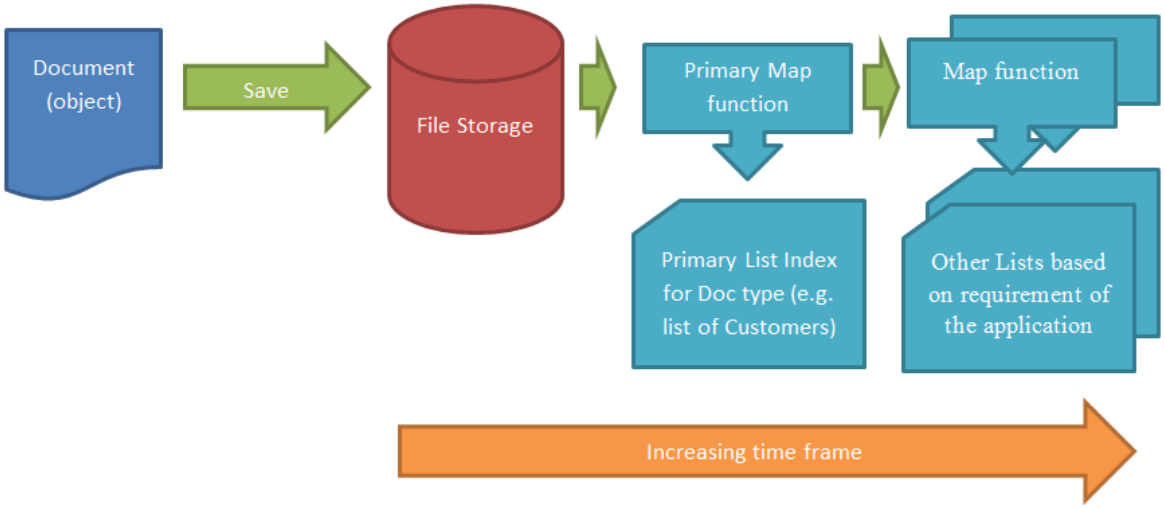
Recientemente estas bases de datos orientadas han aumentado en popularidad, en parte gracias a los sistemas basados en la implementación del paradigma Cloud Computing, que proveen a los programadores sistemas de almacenamiento sencillos.

2.2.6 Document databases

Estas almacenan la información como un documento (generalmente con una estructura simple como JSON o XML) y con una clave única. Es similar a las bases de datos Key-value, pero con la diferencia que el valor es un fichero que puede ser entendido. Si el servidor entiende los datos, puede hacer operaciones con ellos. De hecho varias de las implementaciones de este tipo de bases de datos permiten consultas muy avanzadas sobre los datos, e incluso establecer relaciones entre ellos, aunque siguen sin permitir joins. Podemos encontrar a MongoDB y CouchDB entre las más importantes de este tipo.

Para implementar una base de datos NoSQL documental en lugar de una relacional, no es suficiente con conocer las ventajas o características que ofrecen, también es necesario considerar el grado de facilidad y grado de adaptación para la implementación.

Ilustración 4. Imagen Funcionamiento de almacenamiento de datos



Fuente:**http://www.codeproject.com/Articles/375413/RaptorDB-the-Document-Store**

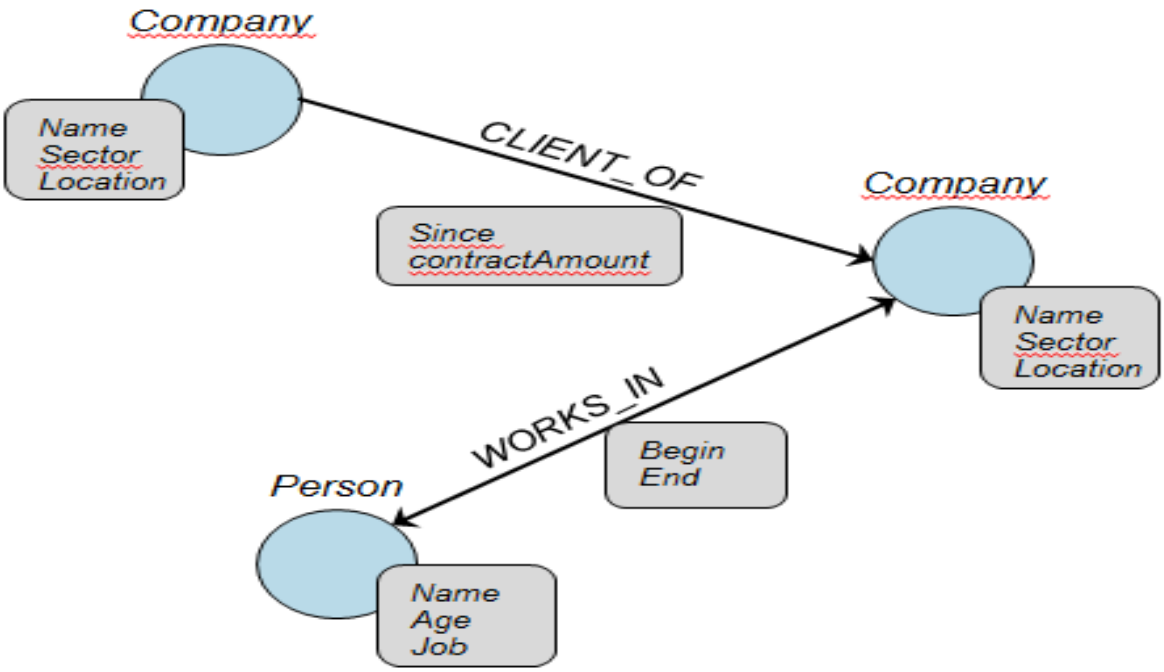
Las razones más importantes, por las cuales algunas compañías justifican el uso de las bases de datos NoSQL documentales en lugar de las relacionales, para el almacenamiento de la información son las siguientes:

- ✓ Alto rendimiento en escrituras masivas a la base de datos: las bases de datos NoSQL documentales permiten almacenar grandes cantidades de información la cual varía desde 1 TB hasta 70 TB.
- ✓ Almacenamiento en caché: la información con más accesos se almacena en caché, la de medio acceso en memoria y la que tienen una tasa baja de acceso en disco.
- ✓ No existen puntos de fallo, debido a que se distribuye el flujo de información mediante la implementación de balanceo de carga automático.

2.2.7 Graph Store:

Hay otras bases de datos que almacenan la información como grafos donde las relaciones entre los nodos son lo más importante. Son muy útiles para representar información de redes sociales. De hecho, las relaciones también pueden tener atributos y puedes hacer consultas directas a relaciones, en vez de a los nodos. Además, al estar almacenadas de esta forma, es mucho más eficiente navegar entre relaciones que en un modelo relacional. Obviamente, este tipo de bases de datos sólo son aprovechables si la información en cuestión se puede representar fácilmente como una red. Encontramos a neo4j entre otras.

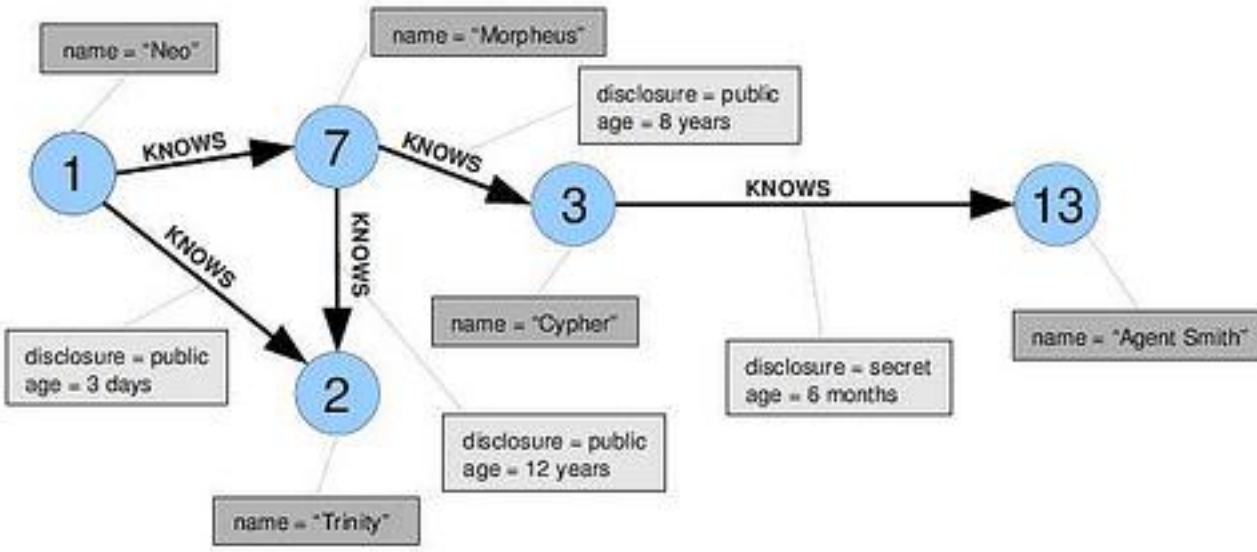
Ilustración 5. Imagen almacenan la información como grafos



Fuente: http://blog.octo.com/en/graph-databases-an-overview/

Las bases de datos orientadas a grafos son una clara alternativa a las bases de datos relacionales, sobre todo para algunas aplicaciones sociales y web que requieren elevada escalabilidad.

Ilustración 5. Imagen Grafos orientados



Fuente:http://highscalability.com/neo4j-graph-database-kicks-buttox

Con estos enfoques en las bases de datos NoSQL, los grafos en el mundo real:

- Sistemas transaccionales en línea, aplicaciones web grandes en particular, deberán responder al usuario final en milisegundos para que resulten exitosas.
- Los cambios surgen dependiendo del comportamiento del usuario, y ello exige a las organizaciones tener mucho cuidado al realizar las migraciones de datos.

- Una solución gráfica permite que los datos evolucionen a medida que evoluciona el negocio, lo que reduce los riesgos y el tiempo de lanzamiento al mercado.

Cada motor NoSQL está diseñado para cumplir una serie de retos específicos, como la redundancia y la recopilación de métricas. Pero la seguridad no es uno de los objetivos principales, aunque tampoco es un factor que se deje de lado, actualmente este requerimiento no funcional es mínimo, sin embargo, siempre se busca un entorno seguro.

2.2.8 Column Store

Como su nombre lo indica**,** guardan los datos en columnas en lugar de filas**.** Con este cambio ganamos mucha velocidad en lecturas, ya que si se requiere consultar un número reducido de columnas, es muy rápido hacerlo pero no es eficiente para realizar escrituras. Por ello este tipo de soluciones es usado en aplicaciones con un índice bajo de

escrituras pero muchas lecturas. Típicamente en data warehouses y sistemas de Business Intelligence, donde además resultan ideales para calcular datos agregados. Cabe resaltar que parte del auge actual que está provocando NoSQL se debe a la adopción de Casandra (originalmente desarrollado por y para Facebook, luego donada la fundación apache) por parte de Twitter y Digg. Apache Cassandra es la base de datos orientas a columnas más conocida y utilizada actualmente.

Conceptualmente, una tabla de base de datos es una estructura de dos dimensiones de datos con celdas organizadas en filas y columnas. La memoria de computadora sin embargo está organizada como una estructura lineal. Para almacenar una tabla en memoria lineal, existen dos opciones, como se muestra en la figura.

Una fila de almacenamiento orientado almacena una tabla como una secuencia de registros, cada uno de los cuales contiene los campos de una fila. A la inversa, en las entradas de una columna se almacenan en ubicaciones de memoria contiguas.

Ilustración 6. Imagen datos en columnas

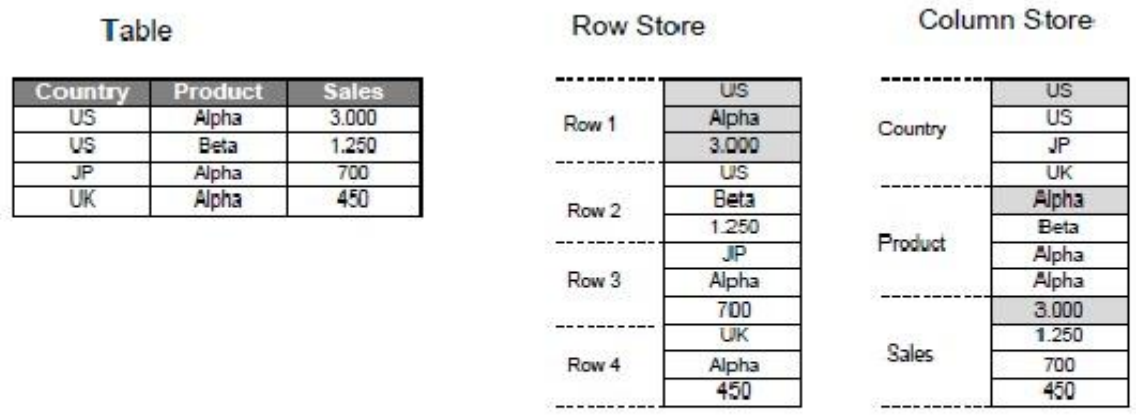


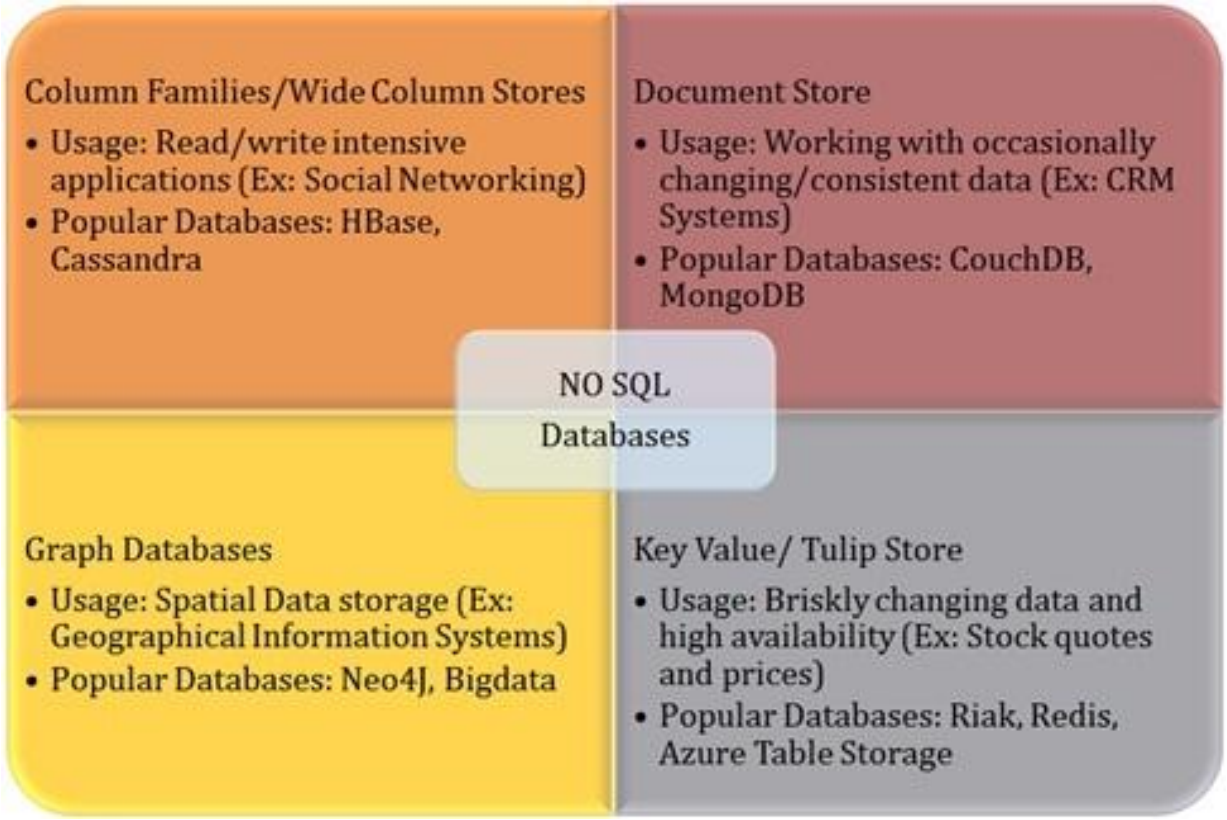
Figure 2: Row and column-based storage

Fuente:http://thecaveabap.blogspot.com/2013/03/sap-hana-database-guiade-desarrollo.html

El concepto de almacenamiento columnar de datos se ha utilizado durante bastante tiempo. Históricamente fue utilizado principalmente para el análisis y almacenamiento de datos donde las funciones de agregado jugaban un papel importante. Usando esta forma de almacenar los datos en columnas, las aplicaciones OLTP requieren un enfoque equilibrado de la inserción y la indexación de los datos de las columnas para minimizar errores de cache.

Existen diferentes bases de datos NoSQL, cuyas características y formatos de almacenamiento varían, haciendo que el modelado para cada una de ellas tenga que ser distinto. A la hora de elegir un sistema NoSQL frente a uno SQL habrá que plantearse previamente si realmente se necesita, después elegir uno en concreto en base a nuestras necesidades, olvidarnos de las formas normales y guardar los datos de distinta manera en función de cómo va a ser el acceso a éstos.

Ilustración 7. Imagen Motores NoSQL



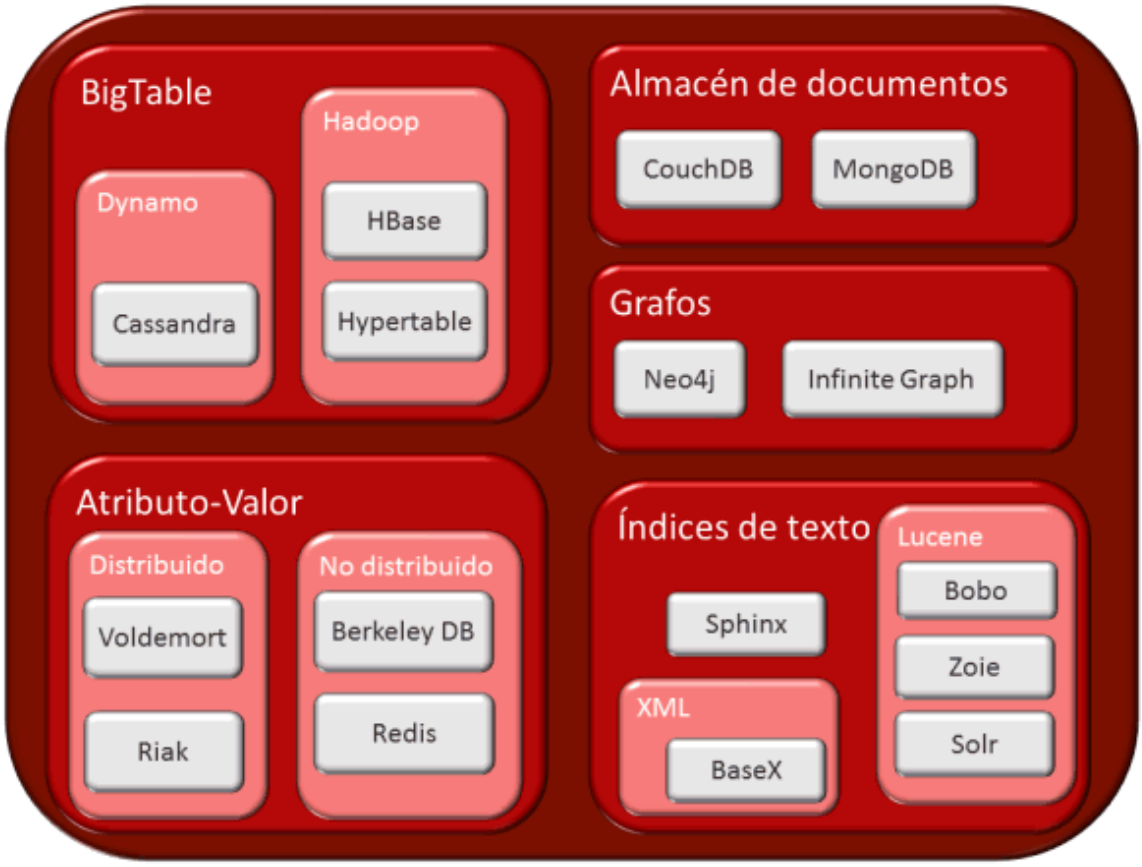
Fuente: http://www.internetria.com/blog/2013/05/08/nosql/

Con la generación de estos nuevos enfoques NoSQL, se ha visto la necesidad de contar con modelos propios para cada una de las formas de almacenamiento mencionadas, en donde es de necesario señalar que el modelo requiere de análisis y diseño riguroso. Sin embargo es necesario hablar sobre la seguridad en estos motores, por lo general, existen opciones de diseño para hacer frente a amenazas específicas y algunas pruebas previstas en la fase de garantía de calidad del ciclo de vida de desarrollo de aplicaciones.

Cada motor NoSQL está diseñado para cumplir una serie de retos específicos, como la redundancia y la recopilación de métricas. Pero la seguridad no es uno de los objetivos

principales, aunque tampoco es un factor que se deje de lado, actualmente este requerimiento no funcional es mínimo, sin embargo, siempre se busca un entorno seguro.

Ilustración 8. Imagen tipos de motores



Fuente: http://lapastillaroja.net/2012/02/nosql-for-non-programmers/

Las aplicaciones NoSQL se pueden dividir en los grupos como se muestra en la figura anterior.

Presentaremos las definiciones de las aplicaciones NoSQL de sus respetivos motores por categorías.

Sistemas BigTable: Es un mapa multidimensional ordenado, distribuido y persistente, que está pensado para ser la base de datos que almacena la información perteneciente a todos los productos Google.

Bigtable es un sistema de almacenamiento distribuido para la gestión de datos estructurados que está diseñado para escalar a un tamaño muy grande en petabytes de datos a través de

miles de servidores básicos. Muchos proyectos de datos de la tienda de Google en Bigtable, incluyendo la indexación web, Google Earth y Google Finance.

Estas aplicaciones colocan exigencias muy diferentes en Bigtable, tanto en términos de tamaño de los datos (de URLs a páginas web a las imágenes de satélite) y requisitos de latencia (de procesamiento a granel backend a datos en tiempo real que sirven). A pesar de estas variadas demandas, Bigtable ha proporcionado con éxito una solución flexible y de alto rendimiento para todos estos productos de Google. En este trabajo se describe el modelo de datos sencillo proporcionado por Bigtable, que ofrece a sus clientes un control dinámico sobre diseño de datos y formato, y se describe el diseño e implementación de Bigtable.

Hbase: HBase es una base de datos de código abierto con la misma arquitectura y funcionalidad que la base de datos de Google, Bigtable. Se apoya en el sistema de ficheros distribuido HDFS para el almacenamiento y tiene una arquitectura columnar.

Está orientada a columnas de datos donde todos los datos se identifican mediante Un ID único de la fila (clave). "Bloom filtros" se utilizan para filtrar las filas y las columnas que están vacías.

Hbase tiene muchas características como son las siguientes:

- ✓ Lineal y escalabilidad modular.
- ✓ Estrictamente lecturas consistentes y escrituras.
- ✓ Sharding automática y configurable de mesas
- ✓ Soporte de failover automático entre RegionServers.
- ✓ Clases de partida ideal para realizar copias de trabajos Hadoop MapReduce con mesas Apache HBase.
- ✓ Fácil de usar la API de Java para el acceso de clientes.
- ✓ Caché de bloques y Bloom Filtros para consultas en tiempo real.
- ✓ Predicado de consulta empuje hacia abajo a través de servidores secundarios Filtros.
- ✓ Thrift gateway y un servicio Web REST ful que soporta XML, protobuf, y opciones de codificación de datos binarios.
- ✓ (JIRB) shell basado en jruby Extensible.
- ✓ El apoyo a la exportación de las métricas a través del subsistema de métricas Hadoop a archivos o ganglios; oa través de JMX

Hypertable: Hypertable fue diseñado con el propósito expreso de resolver el problema de escalabilidad, un problema que no se maneja bien por un RDBMS tradicional. Mientras que es posible diseñar un sistema RDBMS distribuido por romper el conjunto de datos en fragmentos, esta solución requiere una enorme cantidad de esfuerzo de ingeniería y el

sistema resultante tendrá debilidades inherentes porque el motor de base de datos central no fue diseñado para la escalabilidad. Hypertable se basa en un diseño desarrollado por Google para satisfacer sus requisitos de escalabilidad y resuelve el problema de la escala mejor que cualquiera de las otras soluciones NoSQL por ahí.

Hypertable se ha diseñado e implementado para lograr la máxima eficiencia y un rendimiento óptimo. Al optar por hacer la implementación en un lenguaje compilado que no incurre en los costos de rendimiento y estabilidad de la recolección de basura y la interpretación en tiempo de ejecución, Hypertable puede ofrecer capacidad de base de datos equivalente a una fracción del hardware. Esto se traduce en menos equipo, menos consumo de energía, y menos espacio del centro de datos.

La otra ventaja de diseño e implementación muy eficiente de Hypertable es que ofrece todas las ventajas que obtiene de un mejor rendimiento. Para aplicaciones en vivo, Hypertable pueden ayudar a ofrecer una experiencia de usuario mucho más sensible al reducir solicitud latencia global. Para aplicaciones fuera de línea, se logra un mayor rendimiento que significa más trabajo se puede lograr en una cantidad dada de tiempo.

Hypertable es una base de datos consistente. Muchas de las ofertas de bases de datos NoSQL escalables están diseñados en torno al concepto de consistencia eventual que hace que las bases de datos más difícil razonar acerca. Con el tiempo las bases de datos consistentes requieren complejas reconciliadores sintácticas o semánticas y en algunas circunstancias incluso pueden perder datos. Cuando una aplicación escribe datos en Hypertable y consigue un éxito respuesta, la modificación es durable y siempre se refleja en las operaciones posteriores.

Casandra: Base de datos escrita en Java, de tipo Column Family, de código abierto por Facebook en 2008, diseñada por Avinash Lakshman (uno de los autores de Dynamo, de Amazon) y Prashant Malik (Ingeniero de Facebook). De varias maneras se puede pensar en Cassandra como Dynamo 2.0 o una unión de Dynamo y BigTable. Cassandra se encuentra en producción en Facebook, pero aún se encuentra bajo fuerte desarrollo.

Altamente escalable, eventualmente consistente, distribuida y almacenamiento estructurado key-value. Agrupa las tecnologías de sistemas distribuidos de Dynamo y el modelo de datos BigTable de Google. Como Dynamo, es eventualmente consistente. Como BigTable, provee modelo de datos basado en ColumnFamily más enriquecido que los sistemas comunes key-value. (Apache Cassandra, 2012).

Existen varias herramientas para la visualización y administración de los datos, la más destacada es OpsCenter que ofrece gestión y administración para los cluster, esta contiene

una edición comunitaria y una empresarial que incluye características adicionales: alertas, balanceo automático de cargas, respaldos en vivo, entre otras.

Adicional a esta se pueden encontrar otras: Cassandra Cluster Admin, Cassandra Explorer y Helenos

Lenguaje de consulta: CQL (Cassandra Query Language)

Entre algunos usuarios se encuentran:

- Despegar: Usa un cluster de Cassandra para almacenar la sesión de los usuarios de su sitio de consulta de hotel, y también como cache persistente de itinerarios de viaje.
- Facebook: Tiene el mayor número conocido de clusters en operación con cerca de 150 nodos.

- Twitter: Almacenar y consultar la base de datos de lugares de interés; almacenar resultados de minería de datos sobre la base de usuarios; desarrollo interno y externo para análisis en tiempo real a gran escala. (King, 2010).

- EBay: Usado para soportar múltiples aplicaciones con anillos extendidos sobre varios data centers.

Las características de cassadra:

Mantenibilidad: Al utilizar el lenguaje de consultas CQL, similar al estándar SQL facilita un poco el entendimiento e identificación para los desarrolladores que ya han utilizado otros motores. Características de la máquina: Requiere la última versión estable de Java 1.6 JRE o una más actualizada. En cuanto a sistema operativo y versión no se encuentra explícitamente, por lo que depende de la instalación del ambiente de java.

Instalación/Implementación: En el sitio web oficial cassandra.apache.org se encuentra una sección dedicada a la descarga y otra a instalación y requerimientos necesarios.

Usabilidad: Comandos ingresados por medio de Cassandra CLI (interfaz de línea de comandos) o por medio de las herramientas gráficas y de gestión.

Versión: La última versión estable es la 2.0.0, lanzada el 3 de septiembre de 2013. Soporte y solución de inquietudes: Contiene una sección llamada Wiki con información general y configuraciones; y otra nombrada FAQ (Frequently Asked Questions) con un listado de

posibles preguntas. Adicionalmente contiene un vínculo IRC a irc.freenode.net - canales de chat gratuito- donde los desarrolladores y los miembros de la comunidad están disponibles para responder preguntas e inquietudes.

Madurez: Gran recorrido ya que fue inicialmente desarrollo interno de Facebook, y en 2008 salió a la luz como código abierto, se encuentra en versión estable. La primera versión para el público fue lanzada durante el segundo semestre de 2009.

Documentación: Como se mencionó en los ítems anteriores, contiene varios apartados que pueden guiar al usuario en diferentes actividades. Para el caso del lenguaje CQL, contiene un repositorio con las consultas soportadas, sus notaciones y respectivos ejemplos.

Ventajas: Orientada a ColumFamilies, tolerante a fallos, ya que replica los datos de forma automática a múltiples nodos; cuando un nodo falla pueden ser reemplazado sin ningún periodo de inactividad. Permite replicación a múltiples data centers; almacenamiento de los datos tipo ColumnFamily.

Limitaciones: El valor de una columna no debe ser mayor a 2GB, el máximo número de columnas por fila es de 2 billones, la llave y los nombres de las columnas deben ser menores a 64 KB.

Almacén de Documentos: Motores de bases de datos documentales. Estos motores están orientados a la indexación a texto completo y realizan búsquedas más potentes. Los documentos que se manejan son un conjunto de datos identificados por etiquetas, los intereses generales son la sencillez, velocidad y escalabilidad. A continuación se presenta la revisión bibliográfica, para los motores más representativos:

MongoDB: MongoDB Es la base de datos NoSQL líder y permite a las empresas ser más ágiles y escalables. Organizaciones de todos los tamaños están usando MongoDB para crear nuevos tipos de aplicaciones, mejorar la experiencia del cliente, acelerar el tiempo de comercialización y reducir costes.

Es una base de datos ágil que permite a los esquemas cambiar rápidamente cuando las aplicaciones evolucionan, proporcionando siempre la funcionalidad que los desarrolladores esperan de las bases de datos tradicionales, tales como índices secundarios, un lenguaje completo de búsquedas y consistencia estricta.

MongoDB ha sido creado para brindar escalabilidad, rendimiento y gran disponibilidad, escalando de una implantación de servidor único a grandes arquitecturas complejas de centros multidatos. MongoDB brinda un elevado rendimiento, tanto para lectura como para escritura, potenciando la computación en

memoria (in-memory). La replicación nativa de MongoDB y la tolerancia a fallos automática ofrece fiabilidad a nivel empresarial y flexibilidad operativa

Principales características de MongoDB:

- ✓ Alta disponibilidad.
- ✓ Escalabilidad: desde un servidor aislado a arquitecturas distribuidas de grandes clusters (Permite particionar, de manera transparente para el usuario, nuestra base de datos en tantas como shards tengamos. Esto aumenta el rendimiento del procesado de los datos al ser cada una de ellas más pequeña que la original).
- ✓ Agregation Framework- Procesamiento batch de datos para cálculos agrupados usando operaciones nativas de MongoDB.
- ✓ Auto balanceado de carga a través de los distintos shards. El balanceador decide cuándo migrar los datos, y a qué Shard, para que estén uniformemente distribuidos entre todos los servidores del cluster. Cada shard aloja los datos correspondientes a un rango de la clave escogida para particionar nuestra colección.
- ✓ Replicación nativa: sincronización de datos entre servidores.
- ✓ Seguridad: autenticación, autorización, etc.
- ✓ Gestión avanzada de usuarios.
- ✓ Automatic failover: elección automática de un nuevo primario cuando este se ha caído.
- ✓ Es capaz de actualizarse sin dejar de dar servicio.
- ✓ No tiene los cuellos de botella que se producen en las bases de datos relacionales (RDBMS) al procesar grandes cantidades de información.
  .
- ✓ Utiliza objetos JSON para guardar y transmitir la información. JSON es el estándar web hoy en día, por lo que supone una gran ventaja que web y base de datos hablen el mismo lenguaje.
- ✓ Permite realizar consultas y cálculos espaciales, tanto en 2D como en 3D.

- ✓ Permite utilizar Map-Reduce para el procesado de la información a través de funciones JavaScript que se ejecutan en los servidores.

- ✓ Puede almacenar y ejecutar funciones JavaScript en el servidor.

- ✓ Tiene una potente herramienta web que nos permite monitorizar todo lo que pasa en nuestras bases de datos y en nuestras máquinas, MongoDB Management Service(MMS): Monitorización, Backup, Automatización de tareas (próximamente).

En resumen, MongoDB ha sido diseñada para que sea rápida (por ejemplo, sustituye los joins por documentos embebidos), flexible (sin rígidos esquemas de datos), escalables (utiliza escalabilidad horizontal dejando atrás la vertical), para reducir al mínimo las labores de administración (replication, disaster recovery, automatic failover, sharding, load balancing, etc), para que a los programadores les resulte fácil de aprender y dotada de potentes herramientas de análisis de datos (aggregation framework).

CouchDB: CouchDB es catalogado muchas veces como una base de datos NoSQL, un término que se hizo cada vez más popular a finales de 2009 y principios de 2010. Si bien este término es una caracterización más bien genérica de una base de datos, o almacén de datos, sí define claramente un descanso de SQL tradicional bases de datos. Una base de datos CouchDB carece de un esquema o estructuras de datos pre-definidos rígidos tales como tablas.

Los datos almacenados en CouchDB es un documento (s) JSON. La estructura de los datos, o documento(s), puede cambiar dinámicamente para adaptarse a las necesidades cambiantes.

CouchDB es una base de datos que abarca por completo la web. Almacene sus datos con documentos JSON. Tenga acceso a sus documentos y consultar sus índices con su navegador web, a través de HTTP.Índice, combinar y transformar sus documentos con JavaScript. CouchDB funciona bien con la web moderna y aplicaciones móviles. Usted puede incluso servir aplicaciones web directamente de CouchDB. Y usted puede distribuir sus datos o sus aplicaciones, de manera eficiente mediante la replicación incremental de los CouchDB. CouchDB soporta configuraciones maestro-maestro con detección automática de conflictos.

CouchDB viene con una serie de características, como la transformación de documentos sobre la marcha y notificaciones de cambio en tiempo real, que hace que el desarrollo de aplicaciones web una brisa. Incluso viene con un fácil utilizar la consola de administración web. Las principales características son las siguientes:

- ✓ Almacenamiento de documentos: Almacena los datos como documentos esto es, uno o más pares campo/valor expresados en JSON. Los valores de los campos pueden ser datos simples como cadenas de caracteres, números o fechas. Pero también se pueden usar listas ordenadas y vectores asociativos. Todos los documentos en una base de datos CouchDB tienen un identificador único y no requieren un esquema determinado.
- ✓ Vistas e índices Map/Reduce: Los datos almacenados se estructuran por medio de vistas. En CouchDB, cada vista se construye por medio de una función JavaScript que actúa como la mitad Map de una operación map/reduce. La función recibe un documento y lo transforma en un único valor, retornándolo. CouchDB puede indexar vistas y mantener actualizados esos índices a medida de que se agregan, eliminan o actualizan documentos.
- ✓ Arquitectura distribuida con replicación: Se diseñó con teniendo en mente la replicación bidireccional (o sincronización) y la operación off-line. Eso significa que múltiples réplicas pueden tener cada una sus propias copias de los mismos datos, modificarlas y luego sincronizar esos cambios en un momento posterior.

- ✓ Interfaz REST: Todos los ítems tienen una URI única que queda expuesta vía HTTP. REST usa los métodos HTTP POST, GET, PUT y DELETE para las cuatro operaciones básicas CRUD (Create, Read, Update, Delete) con todos los recursos.

- ✓ Consistencia Eventual: Garantiza consistencia eventual para poder ofrecer tanto disponibilidad como tolerancia a las particiones.
- ✓ Hecha para operar offline: Puede replicar datos a dispositivos (como smartphones) que pueden quedar offline y manejar automáticamente la sincronización de los datos cuando el dispositivo vuelve a estar en línea.

Grafos: Es una base de datos que tiene como propósito almacenar estructuras de datos que tienen topología de grafo, es decir, que la información que se almacena se puede representar por medio de nodos y aristas entre ellos. Por definición, una BDG agruparía a cualquier solución de almacenamiento en la que los elementos que están conectados se enlazan sin hacer uso de una referencia por medio de índices (que sería el método habitual de simular una relación en una Base de Datos relacional), de esta forma, los vecinos de una entidad son accesibles directamente por medio de una referencia directa, sin pasar por estructuras intermedias que hagan el proceso de referenciado.

En esta definición no tenemos en cuenta el tipo de grafo (en su sentido más amplio) que nuestros datos seguirán, ni en el tipo de aristas (dirigidas o no), ni en la multiplicidad de las mismas entre dos nodos (unirelacional o multirelacional), ni en la árida que reflejen las aristas (grafo o hipergrafo).

Neo4j: Es una base de datos orientada a grafos escrita en Java, es decir la información se almacena de forma relacionada formando un grafo dirigido entre los nodos y las relaciones entre ellos. Se integra perfectamente con múltiples lenguajes como Java, PHP, Ruby, .Net, Python, Node, Scala, etc. La base de datos está embebida en un servidor Jetty. Está especialmente indicada para modelar redes sociales y sistemas de recomendación.

Se distribuye en dos versiones: la community edition (open source) y la enterprise edition. Para hacer pruebas de concepto nos basta con la community edition pero si quieres sacarle todo el partido a Neo4j la opción enterprise es la más recomendable ya que permite ponerla en cluster, monitorización, backups en caliente y un sistema de cache de alto rendimiento, además de soporte de sus creadores

Otra de las ventajas que tiene Neo4j es que se pueden efectuar las consultas directamente a través de un API Rest lo que hace especialmente interesante su integración con aplicaciones web.

Principales características de neo4j:

- ✓ Alto desempeño y alta disponibilidad (Escalamiento de lectura)
  Soporte sólido y real para transacciones ACID
- ✓ Escalable: 32 miles de millones de Nodos, 32 miles de millones de Relaciones, 64 miles de millones de Propiedades
- ✓ Servidor con una API REST o usable como una biblioteca Java

Infinite granph: Es una base de datos gráfica implementada en Java, y es de una clase de NoSQL (o no sólo) SQL tecnologías de bases de datos que se centran en las estructuras de datos del gráfico. Los desarrolladores utilizan InfiniteGraph encontrar relaciones útiles y, a menudo ocultos en conjuntos de datos grandes altamente conectadas.

Infinite Graph es multiplataforma y escalable, nube habilitado, y está diseñado para manejar un rendimiento muy alto, permite a las organizaciones a lograr una mayor rentabilidad de sus datos de inversión relacionados ayudándoles a "conectar los puntos" a escala global, hacer preguntas más profundas y complejas, a través de las tiendas nuevas o existentes de datos.

Principales características de Infinite Granph:

- ✓ Distributed Database Gráfico: consistes en sacarle todo el provecho de los datos difundidos a través de los lugares de almacenamiento flexibles y configurables, y

distribuir la carga de procesamiento de la manera más eficiente para su aplicación. Puede añadir ubicaciones de almacenamiento y zonas a su base de datos gráfica en cualquier momento, por lo que de inmediato a disposición de todas las aplicaciones que acceden el gráfico.

- ✓ Intuitivo, API Java Centrado-Graph Optimizado para Data Relaciones: Construir aplicaciones basadas en el gráfico de datos intensivos rápidamente utilizando una API que soporta de forma nativa los conceptos de vértices y aristas. Sacar el máximo provecho de la anotación rica y flexible disponible para bordes, que son objetos de primera clase.

- ✓ Acelerado Ingest: Datos ingerir en una escala masiva en paralelo usando múltiples agentes de tuberías en servidores locales o remotos.

- ✓ Gestionado Colocación: Técnica Leverage InfiniteGraph configurable, basado en modelos para la colocación de nuevos elementos en una base de datos gráfica. ubicación gestionada proporciona rápido, fuera de la caja de uso a través del modelo de la colocación inicial incluye con cada base de datos gráfica nueva, o puede crear una costumbre modelo para collocate físicamente elementos de gráfico que a menudo se accede juntos para mejorar el rendimiento de consulta de navegación. Usted puede también físicamente separada o aislar los objetos de datos de acceso frecuente para evitar la contención de bloqueo.

- ✓ Potente Consultas Navegación: Consulta el gráfico a través de una potente API de navegación, la ejecución en proceso o en servidores remotos que distribuyen la carga de procesamiento y maximizar la eficiencia de las consultas que se ejecutan en la que se almacenan los datos.

- ✓ Potente filtrado: Definir un subconjunto de su gráfico en el que se ejecute una consulta de navegación, la racionalización de la consulta y la mejora general del rendimiento. Esto se logra con un objeto de vista gráfico, que ofrece una variedad de técnicas de filtrado para la omisión de objetos o tipos de objetos no pertinentes para su navegación.

- ✓ Política-Driven modelos de consistencia: Acelerar ingesta mediante la relajación de consistencia o elegir el modo de ACID completa cuando sea necesario.

- ✓ Indexación y consultas: Marco indexación Leverage InfiniteGraph para crear índices automáticos, a nivel gráfico en varios campos clave. Crear, objetos de consulta seguras para subprocesos reutilizables para ejecutar de alto rendimiento, las exploraciones a nivel gráfico de los objetos calificados por cadenas de predicado. Objetos de consulta aprovechan automáticamente los índices a nivel gráfico relevantes para un rendimiento óptimo.

- ✓ Marco Plugin: Paquete y desplegar navegantes reutilizables personalizados y formateadores de resultados en aplicaciones Infinite Granph, incluyendo el visualizador.

- ✓ Visualización de Datos: Herramienta de visualización gráfica flexible que te permite navegar por sus datos o realizar consultas de navegación mediante la carga de sus plugins de navegación personalizados.
- ✓ Blueprints: Proporcionar acceso a InfiniteGraph través de una API de código abierto establecido, que apoya el uso de las herramientas populares como Tinkerpop Gremlin.

Atributos/Valor: Son las más sencillas en cuanto a funcionalidad. Este tipo de bases de datos se suelen usar para almacenar y recuperar objetos dónde la estructura interna no es visible a la aplicación cliente. A continuación se presenta la revisión bibliográfica, para los motores representativos.

Los sistemas atributo-valor no distribuidos como Berkeley DB o Redis suelen usarse como software base de otras aplicaciones, bien para la capa de persistencia bien para la implementación de caches. Entre los sistemas atributo valor distribuidos podemos destacar Voldemort y Riak este último incorporando también algunas ideas inspiradas en Dynamo.

Los sistemas atributo-valor no son muy prácticos para almacenar datos estructurados, se suelen utilizar para almacenar y recuperar objetos cuya estructura interna es opaca a la aplicación cliente, por ejemplo, el almacenamiento de imágenes que usa Tumblr está basado en el servicio atributo-valor de Amazon S3. Además de Amazon S3, en modalidad de software como servicio de tuplas atributo-valor también está disponible Azure Table Storage de Microsoft.

Voldemort: Voldemort es una base de datos NoSQL creada por LinkedIn para solucionar los problemas de escalabilidad que sufrían con las bases de datos relacionales.
Voldemort fue diseñada para almacenar datos de forma clave-valor. Permite configurar

diferentes nodos los Cuale contienen los datos y a la vez los datos se van replicando de forma que si se cae un nodo la base siga trabajando.

Es un sistema distribuido de bases de datos clave-valor. Aplica el concepto clásico de clúster, cada uno de sus nodos son independientes (no maestro - esclavo).

Principales características:

Los datos se replican automáticamente a través de servidores múltiples.

Los datos son automáticamente particionados por lo tanto cada servidor contiene sólo un subconjunto de los datos totales.

Las Fallas en el servidor son manejado de forma transparente.

Permite realizar con diferentes frameworks Protocol Buffers, Thrift, Avro y Java Serialization; además permite serializar objetos complejos como listas, arregles, etc.

Los elementos de datos están versionados para maximizar la integridad de los datos sin comprometer la disponibilidad del sistema.

Cada nodo es independiente de otros nodos.

Un buen rendimiento solo nodo: se puede esperar 10-20k de operaciones por segundo en función de las máquinas, la red, el sistema de disco, y el factor de replicación de datos.

Utiliza una estrategia que permite tener nodos en distintos lugares geográficos.

Riak: es una base de datos distribuida diseñada para ofrecer la máxima disponibilidad de los datos mediante la distribución de datos a través de múltiples servidores. Mientras su cliente Riak puede alcanzar un servidor de Riak, debe ser capaz de escribir datos.

Es una base de datos NoSQL key-value código abierto escalable y simplifica el desarrollo, dando a los usuarios la capacidad de formar rápidamente prototipos, probar y desplegar sus aplicaciones.

Principales características:

- ✓ Baja latencia: Riak está diseñado para almacenar datos y atender las solicitudes de esperar y rápidamente, incluso durante las horas punta.

- ✓ Disponibilidad**:** Riak replica y recupera datos de forma inteligente, haciéndolo disponible para las operaciones de lectura y escritura, incluso en condiciones de fallo.
- ✓ Fault-Tolerancia: Riak es tolerante a fallos para que pueda perder el acceso a los nodos debido a la partición de la red o de hardware y nunca perder datos.
- ✓ La simplicidad operacional: Riak permite agregar máquinas al clúster fácilmente, sin una gran carga operativa.
- ✓ Escalabilidad: Riak distribuye automáticamente los datos en todo el clúster y se obtiene un aumento casi lineal rendimiento que se añade la capacidad.

Berkeley db: Permite el desarrollo de soluciones de gestión de datos personalizado, sin el añadido tradicionalmente asociado con este tipo de proyectos personalizados. Berkeley DB ofrece una colección de tecnologías de construcción de bloques de eficacia probada que se pueden configurar para hacer frente a sus necesidades de aplicación del dispositivo de mano para el centro de datos, a partir de una solución de almacenamiento local aun mundial distribuida uno, de kilobytes a petabytes.

Principales características:

- ✓ Caché configurable para modificar el rendimiento.
- ✓ Posibilidad de realizar copias de seguridad y replicación en caliente.
- ✓ Transacciones y recuperación ante errores ACID. Esto es configurable de forma que se puede ir relajando en función de la aplicación.
- ✓ Es compatible con algunas interfaces históricas para bases de datos en UNIX como dbm, ndbm y hsearch.
- ✓ Permite crear bloqueos de forma detallada. Esto es especialmente útil para trabajos concurrentes sobre la base de datos de forma que se bloquea una página de registros durante una transacción para evitar que se modifiquen hasta que termine pero permitiendo actuar sobre el resto de páginas.
- ✓ Los datos se almacenan en el formato nativo del lenguaje de programación.
- ✓ Permite utilizar la característica de snapshots para poder efectuar varias transacciones sobre los mismos registros de manera simultánea.

Redis: Es un motor de base de datos libre de tipo clave-valor (key-value) persistentes que residen en memoria ram y posteriormente vuelca el conjunto de datos almacenados al disco duro. Redis es cliente/servidor por lo que levanta su servicio y responde peticiones, cuenta con interfaz de red lo cual hace posible conectar clientes o nodos desde otros host.

Se puede ejecutar operaciones atómicas sobre estos tipos, como agregar elementos a una cadena; incrementar el valor de un hash, empujando a una lista, intersección de conjuntos de computación, unión y diferencia, o conseguir el miembro de más alto rango en un conjunto ordenado.

Redis resuelve de manera sencilla y eficiente problemas que no necesitan la complejidad de las bases de datos relacionales, como lo es en la mayoría de los casos de usos el almacén y gestión de estructuras de datos temporales, por lo cual redis es mayoritariamente usado para incorporar soluciones sofisticadas de cache de datos o como backend de operaciones en línea en escenarios de alta demanda. Redis no pierde tiempo pensando en relaciones, restricciones ni tipos de datos, se enfoca en hacer eficientemente su trabajo que es establecer y recuperar (set y get) datos sobre las estructuras con las que cuenta.

Redis está escrito en ANSI C y funciona en la mayoría de los sistemas POSIX como Linux, * BSD, OS X sin dependencias externas. Linux y OSX son los dos sistemas operativos en los Redis se desarrolla y más probados, y que recomiendan el uso de Linux para el despliegue .Redis puede funcionar en los sistemas derivados de Solaris como SmartOS, pero el apoyo *es* el mejor esfuerzo. No hay soporte oficial para Windows construye, pero Microsoft desarrolla y mantiene un puerto de Redis Win-64.

Principales características:

- Escalabilidad
- Transacciones: Permiten la ejecución de un grupo de comandos en una sola etapa, con dos importantes garantías.
- Pub: están programados para enviar sus mensajes a receptores específicos (usuarios registrados).
- Lua scripting: se utilizan para evaluar las secuencias de comandos utilizando el intérprete de Lua incorporado en Redis partir de la versión 2.6.0.
- Expiran claves segundos: Establezca un tiempo de espera en clave. Una vez transcurrido el tiempo de espera, se eliminará automáticamente la clave. Una clave con un tiempo de espera asociado a menudo se dice que es **volátil** la terminología Redis.

Índice de texto: Puede crear un índice de texto en tablas de apodos para una base de datos federada que apunta a tablas de una base de datos remota.

Deben tener una clave primaria para ser elegibles para el soporte de índice de texto. Al crear un índice de texto en un apodo, se crean entradas en el esquema SYSIBMTS del catálogo de búsqueda de texto y se añade un objeto de índice de búsqueda de texto informativo a la tabla de catálogo de sistema SYSIBM.

Sphinx: Es un motor para la búsqueda de textos open source (licencia GPLv2). Está desarrollado en C++ lo que lo diferencia claramente de su competidor Lucene que está desarrollado en java. El nombre Sphinx es un acrónimo inglés que significa SQL Phrase Index (índice de frase de SQL).

Sphinx es un paquete de software independiente que proporciona búsquedas en textos rápidas y relevantes. Como su nombre índica ha sido especialmente diseñado para integrar información almacenada en bases de datos SQL, y para ser fácilmente accesible por lenguajes de script (guión). Sin embargo, Sphinx no depende o requiere de ninguna base de datos específica para su funcionamiento y puede emplear otras fuentes de datos, como por ejemplo XMLS.

Principales características:

- ✓ indexación por lotes e incremental
- ✓ soporte para atributos no textuales (escalares, cadenas, conjuntos)
- ✓ soporte para búsqueda distribuida
- ✓ sintaxis para búsqueda de texto completo

relevancia de resultados utilizando factores adicionales al estándar

Otras alternativas

- Keyspace
- 4store
- Simplebd
- Hsearch
- Hive
- Dynamo

2.3 Bases de datos SQL vs. NoSQL.

Las bases de datos relacionales juegan un papel integral en el diseño e implementación de aplicaciones de negocio, puesto que retienen información de usuarios, productos, entre otros. Estas bases de datos funcionan bien para un número limitado de registros, pero con el aumento desproporcional de los datos, algunos detalles de la arquitectura de estas bases muestran signo de debilidad.

Las bases de datos son contenedores de información. Existen distintas formas de ordenar esa información; uno de los lenguajes que se usa para organizar esa información es SQL, modelo relacional y la no relacional (NoSQL). Cuando se habla de Bases de datos NoSQL lo que realmente intenta decir es que son bases de datos que no utilizan el modelo relacional.

SQL es considerado un estándar en la manera en que se deben tratar y administrar todos los datos de una aplicación. La aparición de este modelo relacional llevó consigo la aparición de conceptos nuevos como tablas, relaciones, claves, filas, etc; lo cual facilitó la tarea del programado desde 1970 hasta la actualidad.

NoSQL al salirse del modelo estándar puede complicar la tarea del desarrollo y la gestión de datos. Pero centrándonos en términos de uso de este modelo, NoSQL facilita la gestión de la información en ciertos aspectos como la captura o el soporte de escalabilidad y acceso. En cuanto a la escala horizontal responde mucho mejor y además, es capaz de almacenar cantidades de datos muy grandes.

Representaremos las comparaciones entre las bases de datos relacionales y no relacionales en la siguiente tabla.

Tabla 3. Comparaciones de bases de datos relacionales y no relacionales

Descripción	No relacional	Relacionales
Estructura de datos	Es una estructura flexible, no es necesario definir una estructura de datos	Asume una estructura bien definida de datos tienen que ser uniformes y las propiedades de estos datos pueden definirse por adelantado
Relaciones entre tablas	No existe relación entre	Tiene que estar un muy bien

	colección, no obstante puede depender del modelo de datos.	establecidas y ser referenciadas de forma sistemática.
Transaccionalidad	Se pierde integridad en las transacciones	Utiliza ACID
Consultas e índices	Disminuye es uso de indexación y el poder de las consultas	

Fuente: investigación propia

Mostraremos las diferencias entre las bases de datos relacionales y no relacionales en la siguiente tabla.

Tabla 4. Diferencias entre las bases de datos relacionales y no relacionales

Base de datos Relacional	**Base de datos No relacional**
Escalabilidad Baja	Escalabilidad Alta
Rendimiento bajo	Rendimiento alto
La fiabilidad es alta	La fiabilidad es baja
Utiliza propiedades ACID	Utiliza propiedades BASE
Implementación costosa	Costo de implementación moderados
Alta seguridad	Muy baja seguridad
Procesamiento de datos lento	Procesamiento de datos veloz

Fuente: http://www.kasperu.com/courses/BG011/II/default.htm

Si bien NoSql, no pretende erradicar ni suplantar al SQL y a las bases de datos relacionales, sino pretende ser una alternativa dentro del desarrollo del software donde las condiciones sean requeridas principalmente cuando se manejan millones y millones de datos y el procesamiento y escalabilidad sean puntos críticos, Sin embargo Las Comunidades NoSQL anuncian que no tienes que ser Google Para usarlo.

Aunque sabemos que las bases de datos son sistemas de almacenamiento de información, estas también cuentan con un límite de registros o datos máximo, no porque ya no se puedan almacenar registros sino porque empiezan a crecer los tiempos de espera

considerablemente. Las NoSQL son DB que sacrifican orden por manejar un número mayor de datos las cuales llegan hasta en Terabytes.

Ilustración 9. Imagen bases nosql más utilizadas en el mercado

NOSQL MAS USADAS	CouchDB	MongoDB	REDIS	RIAK
QUE ES ?	Base de datos que abarca completamente la web	Base de datos NoSQL orientada a objetos más avanzada y flexibles.	Motor de base de datos en memoria, basado en el almacenamiento en tablas de hashes (llave, valor)	Es un NoSQL base de datos de la aplicación de los principios de Amazon Dynamo
VERSION ACTUAL	Version 1.2	Version 2.2	Version 2.4	Version 1.2
PLATAFORMA OPERATIVA	Windows, mac y linux	Windows, Linux, OS X y Solaris.	Unix, Linux y sus derivados, Solaris, OS/X , no existe soporte oficial para Windows.	Linux , BSD , Mac OS X , Solaris
ALMACENAMIENTO	B-tree	Estructuras de datos enn documentis tipo BSON (binary JSON)	Tablas de hashes	Fragmento particiones
HERRAMIENTRAS CON LAS QUE SE INTEGRA	Android y facebook	PHP,symfony	ActionScript, C, C++, C#, Clojure, Common Lisp, Erlang, Go, Haskell, haXe, Io, Java, server-side JavaScript (Node.js), Lua, Objective-C, Perl,PHP, Pure Data, Python, Ruby, Scala, Smalltalk y Tcl.	Erlang, HTTP API, PBD API
TIPO LICENCIA	Apache License 2.0	AGPL (Drivers: Apache)	Licencia BSD: permite el uso del código fuente en software no libre.	Apache
LENGUAJE CREACION	Lenguaje Erlang	c++	C/C++	Erlang & C, JavaScript
TIPO DE ALMACENAMIENTO	Colecciones de documentos	Orientada a documentos	Atributo/valor	Atributo/valor
CREADOR POR	Apache	10gen	Salvatore Sanfilippo and Pieter Noordhuis	Apache
PROTOCOLO	HTTP/REST	Custom, binary (BSON)	Telnet-like	HTTP/REST
CARACTERISTICAS	Bidireccionamiento, Vistas: incrustado Map/ reduce, Autenticación posible	Maestro / esclavo de replicación, Las consultas son expresiones javascript, Tiene indexación geoespacial	Maestro-esclavo de replicación, sin disco-swap, Los valores simples o tablas hash de llaves	Ajustable para la distribución y replicación,utilizado como una base de datos gráfica, índices secundarios: pero sólo uno a la vez
UTILIDAD	Requiere acumulación, con consultas predefinidas. Lugares donde el control de versiones es importante	Para consultas dinámicas. No define índices, map / reduce las funciones y buen rendimiento en datos grandes.	Para los rápidos cambios de datos con un tamaño de base de datos previsible	Necesita escalabilidad, disponibilidad y tolerancia a fallos, pero debe pagar por múltiples sitios de replicación.

Fuente: http://repository.ucatolica.edu.co/xmlui/handle/10983/690?show=full

Ilustración 10. Imagen bases nosql columm family

NOSQL Clones of Google's Bigtable	CASSANDRA	HBASE	HYPERTABLE	DYNAMO
QUE ES ?	La base de datos Apache que brinda escalabilidad y alta disponibilidad sin comprometer el rendimiento	Es una fuente abierta , bases de datos distribuidas modelada después de Google BigTable	Codigo abierto, base de datos escalable , similar al modelo de Bigtable, propiedad de Google.	Base de datos NoSQL que proporciona un rendimiento rápido y fiable con una perfecta escalabilidad
VERSION ACTUAL	Versión 1.1.5	Version 0.94	Version 0.96	beta
PLATAFORMA OPERATIVA	Multiplataforma	Multiplataforma	Cruz-plataforma	Multiplataforma
ALMACENAMIENTO	ColumnFamily	Conjunto de tablas	Matriz	Atributos multi valuados.
HERRAMIENTRAS CON LAS QUE SE INTEGRA	Facebook, twitter, dig, rockspace	Facebook	Baidu, Rediff.com o Zvents	SDK AWS, CloudWatch
TIPO LICENCIA	Apache License 2.0	Apache License 2.0	GPL 2.0	propietaria
LENGUAJE CREACION	JavaScript	Java	C++	Java
TIPO DE ALMACENAMIENTO	Orientada a columnas	Columna de base de datos orientada	Matriz asociativa	orientadas a clave-valor
CREADOR POR	Apache	Apache	Zvents	Amazon
PROTOCOLO	Thrift & custom binary CQL3	HTTP/REST	Thrift, C++ library, or HQL shell	HTTP/REST
CARACTERISTICAS	Consulta por columna, Ajustable para la distribución y la reproducción, compatibilidad con múltiples centros de datos de replicación	Utiliza Hadoop HDFS como almacenamiento, optimización de las consultas en tiempo real,se compone de varios tipos de nodos	Implementa diseño BigTable de Google, la búsqueda se puede limitar a intervalos de clave / columna, Conserva los últimos valores N histórico	Ajustable para la distribución y replicación,utilizado como una base de datos gráfica, Índices secundarios: pero sólo uno a la vez
UTILIDAD	Cassandra proporciona una estructura de clave y valor tienda con consistencia sintonizable	Proporciona tolerancia a fallos permite almacenar grandes cantidades de datos y si se necesita al azar, en tiempo real de lectura / escritura es adecuada	Igual que HBase. Cualquier lugar donde se realice escaneos enormeso.	Necesita escalabilidad, disponibilidad y tolerancia a fallos, pero debe pagar por múltiples sitios de replicación.

Fuente: http://repository.ucatolica.edu.co/xmlui/handle/10983/690?show=full

En las figuras anteriores demostramos las bases de datos NoSQL más utilizadas en el mercado, y las columm family con sus respetivas características, con esto queremos enfatizar y dar a conocer los principales motores de las bases de datos NoSQL como son cassandra, hbase, hypertable, dynamo como habíamos mencionado en las páginas anteriores cuando definimos cada una de ellas con esta figura se quiere demostrar con más detalles esos motores.

Podemos mostrar también en la figura el rendimiento que puede servir para diferentes propósitos, demostrar que cumplen con los criterios de rendimiento, comparar sistemas, medir la carga de trabajo, para lograr unos propósitos a la hora de comparar lo necesario que utilizan las herramientas para determinar la eficacia de un sistema al momento de realizar respetivas tareas, validando y verificando los atributos de calidad de los sistemas en sus recursos, para ver el novel de rendimiento de cada uno de ellos con la visualización de la figura en base se establecen quien tiene un mejor rendimiento.

Para poder llegar a escoger el tipo de base de datos cual puede llegar hacer la adecuada para una determinada aplicación podemos tener como guía las figuras mencionadas.

Lo que sigue es un gran recurso para cualquiera considerando diferentes opciones para su trabajo el marco del estilo NoSQL. Como siempre, el enfoque de talla única para todo.

Así como una nota al margen que MongoDB, Cassandra y CouchDB son las tres principales habilidades buscadas en indeed.com (agregados más populares de bolsas de trabajos) de todas las bases de datos en la tabla de abajo. Tener un gran talento es siempre una buena cosa para su elección tecnológica.

Una base de datos NoSQL documental almacena la información en forma de documento, utilizando punteros para el acceso de la información almacenada en la base de datos, esto se realiza sin la ayuda de una estructura entidad relación predeterminada, como en el caso de las bases de datos relacionales. La búsqueda de la información se realiza con base al contenido del documento.

Para realizar una comparativa se deben analizar los aspectos de funcionamiento de las bases de datos NoSQL según la naturaleza de la bases de datos y los esquemas de búsqueda de información de los mismos, en los siguientes apartados se detallan los aspectos mencionados en la siguiente figura.

Ilustración 11. Imagen evaluación de las bases de datos NoSQL

Attributes		NoSQL Databases								
Database model		Document-Stored		Wide-Column Stored				Key-Value Stored		Graph-oriented
	Features	MongoDB	CouchDB	DynamoBD	HBase	Cassandra	Accumulo	Redis	Riak	Neo4j
Design & Features	Data storage	Volatile memory File System	Volatile memory File System	SSD	HDFS		Hadoop	Volatile memory File System	Bitcask LevelDB Volatile memory	File System Volatile memory
	Query language	Volatile memory File System	JavaScript Memcached-protocol	API calls	API calls REST XML Thrift	API calls CQL Thrift		API calls	HTTP JavaScript REST Erlang	API calls REST SparQL Cypher Tinkerpop Gremlin
	Protocol	Custom, binary (BSON)	HTTP, REST	-	HTTP/REST Thrift	Thrift & custom binary CQL3	Thrift	Telnet-like	HTTP, REST	HTTP/REST Tembedding in Java
	Conditional entry updates	Yes	Yes	Yes	Yes	No	Yes	No	No	
	MapReduce	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	No
	Unicode	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	TTL for Entries	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	
	Compression	Yes	Yes	-	Yes	Yes	Yes	Yes	Yes	
Integrity	Integrity model	BASE	MVCC	ASID	Log Replication	BASE	MVCC	-	BASE	ASID
	Atomicity	Conditional	Yes	Yes	Yes	Yes	Conditional	Yes	No	Yes
	Consistency	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
	Isolation	No	Yes	Yes	No	No	-	Yes	Yes	Yes
	Durability (data storage)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	-	Yes
	Transactions	No	No	No	Yes	No	Yes	Yes	No	Yes
	Referential integrity	No	No	No	No	No	No	Yes	No	Yes
	Revision control	No	Yes	Yes	Yes	No	Yes	No	Yes	No
Indexing	Secondary Indexes	Yes	Yes	No	Yes	Yes	Yes	-	Yes	-
	Composite keys	Yes	Yes	Yes	Yes	Yes	Yes	-	Yes	-
	Full text search	No	No	No	No	No	Yes	No	Yes	Yes
	Geospatial Indexes	Yes	No	No	No	No	Yes	-	-	Yes
	Graph support	No	No	No	No	No	Yes	No	Yes	Yes
Distribution	Horizontal scalable	Yes	Yes	Yes	Yes	Yes	Yes		Yes	No
	Replication	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes
	Replication mode	Master-Slave-Replica Replication	Master-Slave Replication	-	Master-Slave Replication	Master-Slave Replication	-	Master-Slave Replication	Multi-master replication	-
	Sharding	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes
	Shared nothing architecture	Yes	Yes	Yes	Yes	Yes	-	-	Yes	-
System	Value size max.	16MB	20MB	64KB	2TB	2GB	1EB	-	64MB	
	Operating system	Cross-platform	Ubuntu Red Hat Windows Mac OS X	Cross-platform	Cross-platform	Cross-platform	NIX 32 entries Operating system	Linux *NIX Mac OS X Windows	Cross-platform	Cross-platform
	Programming language	C++	Erlang C++ C Python	Java	Java	Java	Java	C C++	Erlang	Java

Fuente: http://arxiv.org/ftp/arxiv/papers/1307/1307.0191.pdf

En esta sección, proporcionamos evaluación de algunas de las bases de datos NoSQL 4 (cuatro categorías) con una matriz sobre la base de unos pocos atributos, el diseño, la integridad, la indexación, la distribución, el sistema.

Tabla 5. Conceptos de las bases de datos

		Bases de datos NoSQL			
	Características	Cassandra	MongoDB	Neo4j	Dynamo
1	¿Tiene marco de trabajo gráfico?	Si	Si	Si	Si
2	¿Se tiene soporte por parte del proveedor?	No	No	Si	Si
3	¿Se puede integrar con otras aplicaciones?	Si	Si	Si	Si
4	¿Optimiza el tiempo de repuesta en las transacciones?	Si	Si	Si	Si
5	¿Tiene una versión estable de por lo menos este año	Si	Si	Si	Si
6	¿Almacena los datos de forma óptima?	Si	Si	Si	Si
7	¿Contiene procedimientos almacenados, vistas, triggers, etc.?	Si	No	Si	Si
8	¿Se pueden particionar las tablas que contienen los datos?	Si	Si	Si	Si
9	¿Permite la replicación de los datos?	Si	Si	Si	Si
10	¿Puede funcionar bajo sistema Operativo Windows?	Si	Si	Si	Si
11	¿Puede funcionar bajo sistema operativo AIX?	No	No	Si	No

12	¿Se pueden realizar procesos de back up varias veces a la semana?	Si	Si	Si	Si
13	¿Es posible sincronizar los datos con otro servidor?	Si	Si	Si	Si
14	¿Se asegura la integridad, confidencialidad y disponibilidad de los datos?	Si	Si	Si	Si
15	¿Tiene medidas de aseguramiento de recuperación de desastres?	Si	Si	Si	Si
16	¿Tiene herramientas que apoyen la administración de los datos?	Si	Si	No	Si
17	¿Los datos son almacenados en servidores propios del usuario?	Si	Si	Si	No
18	¿Es usuario es el encargado de la gestión y mantenimiento de los datos?	Si	Si	Si	No

Fuente: http://repository.eia.edu.co/bitstream/11190/411/1/INFO0050.pdf

Pregunta 1: ¿Tiene marco de trabajo?

De acuerdo al análisis realizado de las respuestas de la segunda pregunta de la encuesta, la usabilidad que incluye no fue una consideración tan importante para las empresas como lo fue la integración, pero si ayuda a agilizar el proceso de creación y modificaciones en la base de datos y hace más amigable la interacción del usuario con el sistema. Todas las bases de datos NoSQL de alguna forma cumplen este objetivo, ya sea porque tienen herramientas realizadas por la misma empresa que desarrolló el motor o implementadas por terceros.

Pregunta 2: ¿Se tiene soporte por parte del proveedor?

Requisito que es necesario por todas las empresas para tener asesoría y respaldo de la funcionalidad operativa de la base de datos.

Para Cassandra y MongoDB no existe soporte alguno, lo único que se ofrecen son las ayudas por medio de chat, foros, manuales, etc.

Para Neo4j, la versión que no tiene soporte es la open source (versión comunitaria), las otras dos alternativas implican una compra y acuerdo de servicio, lo que conlleva a un soporte.

DynamoDB cuenta con AWS Support, un canal de soporte personalizado cualquier día de la semana y tendrá un costo dependiendo del tipo de servicio que se adquiera (Basic, Developer, Business, Enterprise) cada uno incluyendo más medios de comunicación y respuestas más rápidas

Pregunta 3: ¿Se puede integrar con otras aplicaciones?

A simple vista, de los sistemas NoSQL escogidos, la opción que parece permitir integración con otras aplicaciones o servicios es DynamoDB, ya que pertenece al grupo de servicios AWS y permite interactuar con otras bases de datos y herramientas de desarrollo como Eclipse, pero Neo4j tiene algunas integraciones con otras como Gelphi, una plataforma de visualización y exploración interactiva para todas las clases de redes y sistemas completos, dinámicos y gráficos jerárquicos. (Gelphi Consortium) Cassandra y MongoDB también cuentan con integraciones un poco más técnicas con otros sistemas y se pueden considerar como mejoras o modificaciones entre el motor NoSQL y el otro software que hacen uso de los APIs para almacenar información e indexar datos y convertir objetos de JavaScript por ejemplo.

Pregunta 4: ¿Optimiza el tiempo de repuesta en las transacciones?

Un 80% de las empresas encuestadas respondieron que era preferible respuestas rápidas a preocuparse por el espacio de almacenamiento usado. En este contexto, las bases de datos NoSQL se diseñaron exactamente con este propósito, que fueran escalables y mantenibles, por esto muchas están basadas en algoritmos de funcionalidades que se ha visto que han tenido éxito como las implementaciones internas de Google y Amazon. De acuerdo con lo anterior, todas opciones son consideradas optimizadas, claro que algunas están dirigidas conceptos y públicos diferentes.

Pregunta 5: ¿Tiene una versión estable de por lo menos este año?

En lo que lleva de este año, se debe esperar que se haya lanzada al público una versión estable, de lo contrario indica que el proyecto está atrasado o parado y se puede considerar un riesgo implementar el motor debido a errores futuros que no serán corregidos o tomarán una larga espera, además muestra el interés y esfuerzo por darse a conocer en el mercado.

En este caso, los 4 motores seleccionados cumplen este requisito:

Cassandra: versión estable1.1.5

MongoDB: versión estable 2.0.6

Neo4j: versión estable 1.8

DynamoDB: se hizo público a comienzos de este año, y aún se encuentra en la versión beta.

Pregunta 6: ¿Almacena los datos de forma óptima?

Como la pregunta 4, NoSQL cambia los paradigmas de las bases de datos SQL y permite mayor flexibilidad y agilidad. Cada motor NoSQL tiene su propia distribución de los datos dependiendo del público objetivo al que apuntan como el caso de los tipos mencionados: alternativas basadas en key-value y orientadas a grafos logrando mejores resultados en el modelo de almacenamiento.

Pregunta 7: ¿Contiene procedimientos almacenados, vistas, triggers, etc.?

Asociada a la pregunta 4 de la encuesta realizada a las empresas, las funcionalidades que se evalúan son: procedimientos almacenados, vistas, triggers, funciones y eventos programados.

MapReduce permite utilizar procesos batch y operaciones agregadas

Cassandra: análogo a los trigger SQL, contiene procedimientos llamados Asynchronous triggers para responder a ciertos eventos en la base de datos. Para los demás elementos Cassandra se ha integrado con funcionalidades como Apache Hadoop y Apache Hive –entre otras más- para usar funciones y facilitar la extracción, transformación y carga de datos (proceso ETL), además las sentencias son ejecutadas por medio de MapReduce para la realización de tareas.

MongoDB: no tiene las funcionalidades comúnmente encontradas en los sistemas SQL, pero tiene diferentes formas para ejecutar actividades y funciones como MapReduce por medio de JavaScript en el servidor. El uso de triggers, no está implementado aún, pero parece que se buscará un acercamiento.

Neo4j: como sus anteriores contiene componentes que realizan tareas similares: para el caso de triggers, cuenta con funciones para registrar las tareas por medio de controladores de eventos para cambios en los nodos; de igual forma estas funciones son similares a las funciones SQL.

En los sistemas Cassandra y Neo4j, la posibilidad de programar las propias funciones con características de trigger, permiten agendar cuándo ejecutar estas por lo que cumplirían con la creación de eventos programados.

DynamoDB: usando Amazon Elastic MapReduce (EMR) facilita el desarrollo de scripts en diferentes lenguajes de programación para la creación de funcionalidades y consultas a la base de datos.

Con las descripciones anteriores se tomará que Cassandra, Neo4j y DynamoDB soportan las características o funcionalidades mencionadas en la pregunta.

Pregunta 8: ¿Se pueden particionar las tablas que contienen los datos?

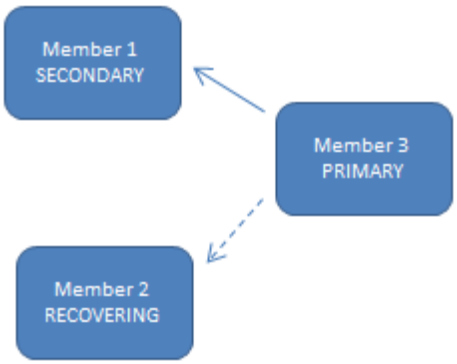
La partición de las tablas o los conceptos análogos en los sistemas NoSQL se permiten a través del sharding o fragmentación, esto permite reducir el número de filas por tabla, lo que incrementa el rendimiento de las transacciones cuando se realizan operaciones sobre ellas. Esto hace que el concepto NoSQL sea escalable horizontalmente adquiriendo más nodos en vez de mayores recursos para un solo servidor o nodo

Pregunta 9: ¿Permite la replicación de los datos?

Cassandra almacena copias, llamadas replicas, de cada fila basada en la clave de la fila, este proceso se realiza cuando se está creando el keyspace –elemento que contiene las "tablas" de la base de datos.

MongoDB soporta la replicación asíncrona entre varios servidores en caso de fallas y para redundancia de datos, de modo que es posible proveer un servicio continuo. De este modo un servidor escribe solamente mientras los otros están atentos para la lectura y se asegura que los datos estén actualizados como se observa brevemente en la figura siguiente.

Ilustración 12. Imagen mongoDB



Fuente: http://www.mongodb.org/display/DOCS/Replication

Neo4j funciona como el caso anterior, con un concepto de tipo maestro esclavo en los *clusters* en el que uno escribe y el otro escucha coordinando y replicando los datos.

DynamoDB replica los datos de manera automática y sincronizada entre tres zonas de disponibilidad de una región a fin de ofrecer un alto nivel de disponibilidad y durabilidad de los datos frente a fallos de la máquina. (Amazon.com Inc.)

Pregunta 10: ¿Puede funcionar bajo sistema operativo Windows?

Según las repuestas dadas a la pregunta cinco de la encuesta, el sistema Windows fue la opción más elegida y la que tiene compatibilidad las base de datos NoSQL, de igual forma, no en todas estas se especifica exactamente cuál versión de Windows es la que soportan, por ejemplo Neo4j funciona mínimo bajo Windows XP.

Con esto se debería probar primero si efectivamente funciona para la versión del sistema que se tenga de Windows para verificar el correcto funcionamiento y evitar futuras complicaciones.

Pregunta 11: ¿Puede funcionar bajo sistema operativo AIX?

Se consideró poner este requisito, ya que fue el segundo sistema operativo más usado por las empresas. Unix también quedó en el segundo lugar, pero Unix en sí tiene varios sistemas como FreeBSD, Solaris e incluso AIX es un sistema Unix, por lo que no se consideraron todas estas variantes.

Con el motor Cassandra, no se encontró información que demostrara que se podía instalar, por lo que no cumpliría; MongoDB, no lo soporta; Neo4j no lo indica explícitamente, pero parece que un archivo de configuración tiene una condición para evaluar donde se encuentran los archivos correspondientes a JRE en el sistema de IBM; por último la administración de DynamoDB funciona como un servicio suministrado por Amazon, no se tiene que instalar o implementar nada directamente en la máquina, sólo acceso a internet a través de un navegador web para gestionar los diferentes servicios de AWS por lo que no podría acceder desde este sistema operativo. En nuestra opinión si es solamente para la gestión y comportamiento del motor, se puede ingresar a través de otros navegadores que sí son permitidos.

Pregunta 12: ¿Se pueden realizar procesos de back up varias veces a la semana?

Los procesos de respaldo son implementados o controlados cuando se comienza a crear la base de datos y son constantemente usados en la replicación. En las versiones empresariales de Neo4j y de la aplicación OpsCenter para Cassandra es posible realizarlos en vivo.

Pregunta 13: ¿Es posible sincronizar los datos con otro servidor?

Para todos los motores seleccionados es posible este atributo, ya que hace parte de la funcionalidad de la replicación.

Pregunta 14: ¿Se asegura la integridad, confidencialidad y disponibilidad de los datos?

Los tres conceptos para la seguridad se cumplen en las bases de datos, aunque se debe tener en cuenta que en algunos casos como la versión comercial Enterprise de Neo4j se ofrece mayor disponibilidad que sus otras dos.

Confidencialidad: autorizaciones y autenticaciones como Cassandra; con algoritmos para mantener los datos seguros como lo hace DynamoDB; autenticación y modos seguros en las conexiones como MongoDB; y aseguramiento de puertos y conexiones remotas, soporte HTTPS y control de accesos de rangos de IP y URLs como Neo4j.

Integridad y disponibilidad: a través de la fragmentación (denominado sharding) y la replicación de datos.

Pregunta 15: ¿Tiene medidas de aseguramiento de recuperación de desastres?

Esta es una funcionalidad que les permite a las empresas seguir operando y ofreciendo sus servicios y a su vez evitando la pérdida de los datos como se ha mencionado anteriormente.

En este caso la función de replicación lo cubre, ya que permite, en caso de error, seleccionar otro nodo que tiene los mismos datos para que continúe con las actividades normales como se vio en la figura anterior.

Pregunta 16: ¿Tiene herramientas que apoyen la administración de los datos?

Esta característica les permite a las empresas analizar los datos y el comportamiento de la base de datos para tomar mejores decisiones en el futuro.

Cassandra: cuenta con aplicaciones de terceros para la administración que permiten lagestión y análisis de los datos.

MongoDB: al igual que Cassandra, cuenta con herramientas de terceros para el monitoreo de la información. La mayoría es de tipo open source.

Neo4j: No se encontraron aplicaciones que apoyen la gestión de los datos.

DynamoDB: se logra a través de la misma herramienta que proporciona Amazon denominada AWS Management Console para visualizar los datos, monitorear los recursos de escritura, lectura, red y almacenamiento, y controlar el estado de las bases de datos.

Pregunta 17: ¿Los datos son almacenados en servidores propios del usuario?

Para los motores Cassandra, MongoDB y Neo4j, el usuario se debe encargar de descargar los archivos fuentes para la instalación e ingreso de datos, por lo que son contenidos en equipos de cliente.

Contrario ocurre con DynamoDB, que son guardados en los servidores de Amazon.

Pregunta 18: ¿Es usuario es el encargado de la gestión y mantenimiento de los datos?

Esta pregunta complementa la anterior, ya que generalmente si un proveedor ofrece los servicios de almacenamiento, provee también servicios adicionales como ocurre con DynamoDB.

Amazon controla la cantidad de los recursos según los solicite la aplicación, así como mantener operando continuamente la base de datos, quitándole esta carga al cliente. Para los otros tres motores, el usuario se debe encargar de tener las características necesarias de software y hardware para la replicación y configuración de los servidores.

Almacenamiento de bases de datos de documentos y su lenguaje de consulta

Tabla 6. Almacenamiento de bases de datos de documentos y su lenguaje de consulta

Name	Language	Notes
TokuMX	C++, C#, Go	MongoDB with Fractal Tree indexing
Solr	Java	Search engine
SimpleDB	Erlang	online service
Couchbase Server	C, C++, Erlang	Support for JSONand binary documents
Sedna	C++, XQuery	XML database

Name	Language	Notes
OrientDB	Java	JSON, SQL support
OpenLink Virtuoso	C++, C#, Java, SPARQL	middleware and database engine hybrid
ObjectDatabase++	C++, C#, TScript	Binary Native C++ class structures
IBM Notes and IBM Domino	LotusScript, Java, IBM X Pages, others	MultiValue
MongoDB	C++, C#, Go	BSON store (binary format JSON)
MarkLogic Server	Java, REST, XQuery	XML database with support for JSON, text, and binaries
Jackrabbit	Java	Java Content Repository implementation
eXist	Java, XQuery	XML database
ElasticSearch	Java	JSON, Search engine
CoreFoundation Property list	C, C++, Objective-C	JSON, XML, binary

Name	Language	Notes
Clusterpoint	C, C++, REST, XML, full text search	XML database with support for JSON, text, binaries
Cloudant	C, Erlang, Java, Scala	JSON store (online service)
BaseX	Java, XQuery	XML database
Oracle NoSQL Database	C, Java	
Apache CouchDB	Erlang	JSON database

El concepto central de un almacén de documentos es la noción de un documento. Mientras que cada aplicación de base de datos orientada a documentos difiere en los detalles de esta definición, en general, todos asumen que los documentos encapsulan y codificar datos (o información) en algunos formatos o codificaciones estándar. Codificaciones en uso incluyen XML, YAML, y JSON, así como las formas binarias como BSON.

Diferentes implementaciones ofrecen diferentes formas de organizar y / o agrupar documentos:

- Colecciones
- Etiquetas
- Metadatos para no visible
- Jerarquías de directorios

En comparación con bases de datos relacionales, por ejemplo, colecciones podrían considerarse análoga a las tablas y documentos análogos a los registros. Pero son diferentes: cada registro de una tabla tiene la misma secuencia de campos, mientras que los documentos en una colección pueden tener campos que son completamente diferentes.

Los documentos se abordan en la base de datos a través de una única tecla que representa ese documento. Una de las otras características que definen a una base de datos orientada a documentos es que, más allá de utilizar el documento clave sencilla (valor-clave) de búsqueda para recuperar un documento, la base de datos ofrece un lenguaje API o consulta que recupera documentos en función de su contenido.

Tabla 7. Modelos de las bases de datos NoSQL

Modelo de Datos	**Rendimiento**	**Escalabilidad**	**Flexibilidad**	**Complejidad**	**Funcionalidad**
Key-Value Store	alto	alto	Alto	ninguno	variable (ninguno)
Columna-Oriented tienda	alto	alto	moderado	bajo	mínimo
Documento Orientado tienda	alto	variable (alto)	Alto	bajo	variable (bajo)
Gráfico de base de datos	variable	variable	Alto	alto	la teoría de grafos
Base de Datos Relacional	variable	variable	Bajo	moderado	álgebra relacional

Las tecnologías NoSQL permiten el procesamiento rápido y eficiente de conjuntos de datos dando la mayor importancia al rendimiento, la fiabilidad y la agilidad. Los diferentes tipos de bases de datos NoSQL existentes se pueden agrupar en cuatro categorías: documentales, grafos, clave-valor y basadas en columnas.

En ese sentido a menudo, las bases de datos NoSQL están altamente optimizadas para las operaciones recuperar y agregar, y normalmente no ofrecen mucho más que la funcionalidad de almacenar los registros (almacenamiento clave-valor). La pérdida de flexibilidad en tiempo de ejecución, comparado con los sistemas SQL clásicos, se ve compensada por ganancias significativas en escalabilidad y rendimiento cuando se trata con ciertos modelos de datos.

Su ventaja es que no requieren de una constante de estructura de datos. Son útiles cuando sus requerimientos y por lo tanto su diseño de base de datos cambia continuamente, o cuando se trabaja con conjuntos de datos que pertenecen juntos, pero todavía se ven de manera muy diferente. Cuando usted tiene un montón de mesas con dos columnas llamadas clave y valor, entonces estas podrían ser que vale la pena considerar.

Mientras que la mayoría de las bases de datos NoSQL abandonar el concepto de la gestión de datos, relaciones, estas bases de datos abrazan aún más que los también llamados bases de datos relacionales.

Pensadas para realizar consultas y agregaciones sobre grandes cantidades de datos, funcionan de forma parecida a las bases de datos relacionales, pero almacenando columnas de datos en lugar de registros. Las más simples de entender, guardan tuplas clave-valor.

Tabla 8. Posibilidades de consulta

Database		POSIBILIDADES DE CONSULTA			
		REST API	JAVA APT	*Query Language*	*Map Reduce Support*
Key Value *Stores*	Voldemort	-	+	-	-
	Redis	-	+	-	-
	Membase	+	+	-	-
Document stores	Riak	+	+	-	+
	MongoDB	+	+	-	+
	CouchDB	+	+	-	+
Column Family Stores	Cassandra	-	+	+	+
	HBase	+	+	+	+
	Hypertable	-	+	+	+

Graph Databases	Sesame	+	+	+	-
	BigData	+	+	+	-
	Neo4J	+	+	+	-
	GraphDB	+	+	+	-
	FlockDB	-	+	-	-

Dado que los modelos de datos están estrechamente vinculadas con la consulta posibilidades, el análisis de las consultas que deben ser apoyados por la base de datos es un proceso muy importante con el fin de encontrar un modelo de datos adecuado.

Bases de datos NoSQL no sólo difieren en su modelo de datos proporcionada, también difieren en la riqueza de sus funcionalidades de consulta ofrecidos. El estado actual de la NoSQL puede ser comparado con el tiempo. Similar a la vez que una gran cantidad de técnicas bases de datos heterogéneas planteadas en los últimos años que difieren en Su modelo de datos que ofrece, lenguajes de consulta y API. Por lo tanto, una cierta investigación se realiza con el fin de lograr una más amplia de los sistemas NoSQL adoptando mediante el desarrollo de la consulta estandarizada lenguas por alguna de estas tiendas. Hasta saber que los desarrolladores todavía tienen que hacer frente a las características específicas de cada tienda NoSQL.

Además de considerar el modelo de datos soportada y cómo influye en las consultas sobre atributos específicos, es necesario tener una mirada más cercana en las interfaces ofrecidas en a fin de encontrar una base de datos adecuada para un caso de uso específico. Si una interfaces simples, se requiere lenguaje API REST inespecífica puede ser una solución adecuada especialmente para aplicaciones web, mientras que las consultas de críticos de desempeño deben ser intercambiados sobre aplicaciones específicas de idiomas que están disponibles para casi cada lenguaje de programación comunes como Java. Manejando idiomas que ofrece un mayor nivel de abstracción con el fin de reducir complejidad. Por lo tanto su uso es muy útil cuando más consultas complicadas deben ser manejadas. Si el cálculo intensivo Se requiere consultas más grandes conjuntos de datos, MapReduce deben ser utilizados.

Tabla 9. Control de concurrencia

Database	CONTROL DE CONCURRENCIA		
	Locks	Optimistic	MVCC

			Locking	
Key Value *Stores*	Voldemort	-	+	-
	Redis	-	+	-
	Membase	-	+	-
Document stores	Riak	-	-	+
	MongoDB	-	-	-
	CouchDB	-	-	+
Column Family Stores	Cassandra	-	-	-
	HBase	+	-	-
	Hypertable	-	-	+
Graph Databases	Sesame	+	-	-
	BigData	-	-	-
	Neo4J	+	-	-
	GraphDB	-	-	+
	FlockDB	-	-	-

Si muchos usuarios tienen acceso a una fuente de datos en paralelo, estrategias para evitar incompatibilidad sobre la base conflictiva las operaciones son necesarias. Bases de datos tradicionales utilizan pesimista estrategias de consistencia con acceso exclusivo a un conjunto de datos.

Estas estrategias son adecuadas, si los costos de cierre son bajos y conjuntos de datos no se bloquean durante un largo tiempo. Desde cerraduras son muy caro en grupos de base de datos que se distribuye en gran distancia y muchas aplicaciones web tienen que soportar muy altas tasas de solicitud de lectura, estrategias de consistencia pesimistas pueden causar pérdida masiva rendimiento.

Bloqueo y transacciones están apoyados por Sesame y Ne04j. HBase es el único sistema basado NoSQL clave, que apoya una variante limitada de bloqueos y transacciones clásicos, ya que estas técnicas sólo se pueden aplicar en las filas individuales.

Las solicitudes que no tienen que lidiar con muchos conflictos de operaciones pueden elegir bloqueo optimista para proporcionar actualización de fila múltiple consistente y operaciones de actualización borrar, leer. Bloqueo optimista con el apoyo de Voldemort, y Redis Membase. Las aplicaciones que necesitan para responder demasiados paralelos, leer y escribir peticiones y que sólo tienen que proporcionar un cierto nivel de consistencia debe utilizar Riak, CouchDB, Hypertable o GraphDB. Si la actualización son suficientes, MongoDB, Cassandra y FlockDB se puede utilizar con el fin de manejar enormes tasas de petición, evitando MVCC encima de la cabeza al mismo tiempo.

Tabla 10. Particiones

Database		PARTICIONES	
		Range based	Consistent Hashing
Key Value *Stores*	Voldemort	-	+
	Redis	-	+
	Membase	+	+
Document stores	Riak	+	+
	MongoDB	+	+
	CouchDB	+	+
Column Family Stores	Cassandra	-	+
	HBase	+	+
	Hypertable	-	+
Graph Databases	Sesame	+	+
	BigData	+	+
	Neo4J	+	+
	GraphDB	+	+
	FlockDB	-	+

Todas las tiendas de valores clave antes mencionados y el documento tiendas Riak y CouchDB se basan en hashing consistente, mientras que los documentos MongoDB están divididos por la gama de su identificación. A diferencia de valor y documentos almacenes de claves, la columna tiendas de la familia pueden ser divididas verticalmente, también. Columnas de la misma familia columna se almacenan en el mismo servidor con el fin de aumentar el rendimiento de consulta gama de atributos. Conjuntos de datos Cassandra se repartió horizontalmente por hashing consistente, mientras que el Basado clones BigTable gama HBase y Hypertable uso particionamiento. Desde datamodels familiares columna

pueden ser particionada de manera más eficiente, estas bases de datos son más adecuadas para grandes conjuntos de datos que almacenan los documentos.

Desde complejidad de los sistemas distribuidos aumento sistema de masivamente, la partición debe ser evitado si no es absolutamente necesario. Sistemas que luchan en su mayoría con alta peticiones que pueden escalar a grandes carga de trabajo más fácilmente a través de replicaciones.

Tabla 11. Replicaciones

Database		REPLICACIÓN		
		Read from Replica	Write to Replica	Consistency
Key Value *Stores*	Voldemort	+	-	+/-
	Redis	+	-	-
	Membase	-	-	+/-
Document stores	Riak	+	-	+/-
	MongoDB	+	-	-
	CouchDB	+	+	+/-
Column Family Stores	Cassandra	+	-	+
	HBase	-	-	+
	Hypertable	-	-	+
Graph Databases	Sesame	-	-	+
	BigData	+	-	+
	Neo4J	+	+	-
	GraphDB	-	-	+
	FlockDB	+	+	+/-

Además de leer mejor rendimiento a través de la carga equilibrio, la replicación trae también una mejor disponibilidad y durabilidad, debido a falta de nodos puede ser sustituida por otros servidores.

Desde bases de datos distribuidas deben ser capaces de hacer frente a fallos en los nodos y temporales de la red, que sólo la disponibilidad total o plena coherencia se puede garantizar en una vez en sistemas distribuidos. Si todas las réplicas de un maestro servidor se actualizan de forma sincrónica, el sistema no sería disponible hasta que todos los esclavos habían cometido una operación de escritura.

Si los mensajes se perdieron debido a problemas en la red, el sistema no estará disponible para un período de tiempo más largo, para las plataformas que se basan en la alta disponibilidad, esta solución no es adecuada? porque incluso unos pocos milisegundos de latencia pueden tener grandes Influencias en el comportamiento de los usuarios.

De acuerdo con Amazon, sus ventas se hundieron en el 1% de un retraso de 100 milisegundos durante una prueba. Google se dio cuenta de un 25% menos de tráfico como 30 en lugar de búsqueda.

Sistemas que son eventualmente consistentes con el fin de aumentar disponibilidad son Redis, CouchDB y Ne04j. Teniendo en cuenta que CouchDB y Ne04j también ofrecen la replicación maestro-maestro, estos sistemas son adecuados para offline apoyo necesario por ejemplo, en móvil las aplicaciones. Voldemort, Riak, MongoDB, Cassandra y FlockDB ofrecer replicación optimista, por lo cual pueden ser utilizado en cualquier contexto. Desde Membase, HBase, Hypertable y GraphDB no utilizan la replicación de carga equilibrio, estas tiendas ofrecen una total coherencia. Bigdata es la única tienda, que apoya la plena coherencia y la replicación de forma nativa.

Usar la herramienta adecuada para el trabajo es la ideología propagada de la comunidad NoSQL, porque cada base de datos NoSQL es especializado en determinados casos de uso. Puesto que no hay evaluación disponible, que responden.

En general, se deben utilizar los almacenes de claves-valor para operaciones muy rápidas y sencillas, almacenamiento de documentos ofrecen una modelo de datos flexible con grandes posibilidades de consulta, la columna family stores son adecuadas para grandes bases de datos que tienen que puede escalar a gran tamaño, y las bases de datos del gráfico se deben utilizar en dominios, donde las entidades son tan importantes como las relaciones entre ellos.

Tabla 12. Sistemas de gestión de base de datos: un análisis nosql

	Flat File Database	RDBMS	NoSQL
Data Model	Flat File	Tables	Columns, Graph, Document, Key/Value
Schema	Schema-less	Fixed Schema	Schema-less
Query Languages	CQL	SQL	API calls, JavaScript and REST
Integrity Model	None	ACID	CAP, BASE
Applicability	Any	Relational and transactional data	Non-relational data
Security	No security	Limited security mechanisms, vulnerable to SQL injection	Authorisation and authentication weaknesses, no encryption, Multiple interfaces increase attack surface.
Advantages	Simpler to use, Less expensive, suited for small scale use	Ensures data integrity between transactions, better security, supports medium to larger sized organisations, provides backup and recovery controls	Can cater for Big Data, unstructured data and distributed systems
Disadvantages	No support for multi-user access, redundancy and integrity problems	Expensive and difficult to manage in distributed systems, Complex and difficult to learn, not suitable for unstructured data	Security is a concern (no encryption), lack of standard query language, Too many varied databases thus no single solution for different purposes
Examples	MsDOS	Oracle, Postgres, MySQL, Microsoft SQL Server	MongoDB, Cassandra, Neo4J

Los fundamentales que han guiado la gestión de datos durante los últimos 40 años se basaron en el modelo de transacción que abrazó el principio ACID ahora visto como inflexible y demasiado estricto a la luz de los datos no estructurados, actualizaciones frecuentes y acceso a enormes cantidades de información almacenada a través de varios datos stores. ACID modelo funcional bien estructurado con bases de datos relacionales.

Sin embargo no les va bien en grandes sistemas distribuidos en el que la disponibilidad y el rendimiento son de suma importancia. Un modelo más flexible conocido como BASE se introdujo en línea con la evolución hacia las bases de datos NoSQL como una forma de contrarrestar los desafíos planteados por el modelo de ACID. El modelo de ACID se queda corto en situaciones caracterizadas por grandes datos de caracteres estructurados y Grandes Usuarios. En este sentido se convierte en un modelo más eficaz que ACID puede ser un obstáculo para la operación de base de datos.
En este análisis se quiere demostrar la postura que adopta más global mediante la incorporación en cierta medida en un momento dado.

Tiene en conocimiento el hecho de que si se produce una partición de red puede haber fallos parciales pero los datos no fallan completamente del sistema por lo tanto todavía estará disponible en medio de pequeños retrasos. Se destaca como una técnica más robusta en una base de datos NoSQL de CAP. Detenerse en la PAC como una herramienta para el diseño de sistema de bases de datos escalable moderna podría plantear una serie de problemas. (Mapanga & Kadebu, 2013)

2.4 MARCO CONCEPTUAL

DATOS: conjunto básico de hechos referentes a transacciones. Incluyen cosas como: tamaño, cantidad, volumen, tasa, nombre o lugar.

MOTOR DE BASES DE DATOS: software dedicado a servir de interfaz entre la bases de datos, usuario y aplicaciones que utilizan.

JAVASCRIPT: lenguaje de programación interpretado, implementado en una navegador web, permitiendo así mejoras en la interfaz usuario y páginas web dinámicas.

JSON: objeto de notación javascript, es un formato ligero utilizado para el intercambio de datos basados en javascript el cual no requiere de uso de XML.

JOIN: unión interna de dos tablas, la cual retorna las filas que el usuario desea cuando hay al menos una coincidencia entre ambas tablas.

ERLANG: lenguaje de programación utilizado para construir sistemas escalables en tiempo real, con alta disponibilidad.

HARDWARE: unidades de almacenamiento secundario, principalmente discos duros, discos compactos, cintas magnéticas.

PHP: lenguaje de programación gratuito y multiplataforma, se ejecuta en el servidor web justo antes de enviar la página web a través de internet al cliente.

SQL: lenguaje declarativo de acceso a bases de datos relacionales

NOSQL: bases de datos no-relacionales, distribuidos escalables horizontalmente.

COUCHDB: base de datos documental desarrollada con el objetivo de escalar horizontalmente con facilidad.

HTTP: protocolo de transferencia de hipertexto.

ACID: conjunto de características necesariamente para que una serie de instrucciones en una base de datos sean consideras como una transacción. ACID es el acrónimo de atomicidad, consistencia, aislamiento y durabilidad.

2.5 MARCO LEGAL

La expedición de la ley 29 de 1990 y los decretos 393, 585 y 591 de 1991, mediante los cuales se conforma el SNCYT, constituyen un avance importante en materia política científica y tecnológica en el país en las últimas décadas. De igual forma, la adscripción de Colciencias al departamento Nacional de Planeación (ley 29 1990), permite articular de manera más eficiente de las actividades científicas y tecnológicas con los requerimientos y problemáticas de los diferentes sectores de la vida nacional.

Ley 29 de 1990. Por la cual se dictan disposiciones para el fomento de la investigación científica y el desarrollo tecnológico y se otorgan facultades extraordinarias.

Decreto 393 de 1991. Por cual se dictan normas sobre asociación para actividades científicas y tecnológicas, proyectos de investigación y creación de tecnología

Decreto 591 de 1991. Por el cual se regulan las modalidades específicas de contratos de fomentos de actividades científicas y tecnológicas.

2.6 ESTADO DEL ARTE

Este trabajo pretende analizar y comparar los sistemas de bases de datos en SQL y las bases de datos NOSQL discutir sus representaciones de datos como establecer las comparaciones.

Las bases de datos NOSQL aparecieron como una necesidad para el manejo de información voluminosa y distribuida en los grandes sitios web pero serán soluciones para las grandes empresas y las medianas empresas? solucionan los problemas que las bases de datos Relacionales solucionan? Qué tipo de problemas solucionan? Este artículo presenta este modelo y sus actuales logros y resultados. (Díaz Sepúlveda, 2013)

Palabras Claves: NOSQL, bases de datos, relacional, Not Only SQL, CAP, ACID, BASE

Este artículo consiste en dar algunas respuestas a estas inquietudes que genera la tecnología de bases de datos NOSQL que vienen en vertiginoso desarrollo, especialmente desde el 2009, presenta las principales ventajas de las BD NoSQL en la distribución de datos y la buena escalabilidad de las bases de datos y la importancia del futuro que deben conocer la computación en la nube, bases de datos NOSQL y de las posibilidades de productos, servicios, modelos, soluciones, arquitecturas.

El aporte que le da este artículo a mi trabajo es dar a conocer como surgen las bases de datos NoSQL, en qué consisten, el origen. Este trabajo pretende dar algunas respuestas a estas inquietudes que genera la tecnología de bases de datos NOSQL que vienen en vertiginoso desarrollo, especialmente desde el 2009.

Las bases de datos NoSQL han experimentado un importante incremento en su aplicación en los últimos tiempos. La gran flexibilidad que ofrecen y las posibilidades que brindan desde el punto de vista de la optimización en sus diseños de acuerdo al problema a resolver las convierten en una atractiva variante a tener en cuenta para los desarrolladores de aplicaciones de gestión de información. En el presente artículo se hace un recorrido por la evolución de los tipos de bases de datos hasta llegar a las relacionales, las cuales se analizan con el objetivo de mostrar los aspectos asociados a estas que propiciaron el surgimiento de las NoSQL. (Enríquez & Gracia del Busto, 2012)

Este artículo consiste de las bases de datos NoSQL que nos ofrecen una alternativa para las bases de datos relacionales, no un reemplazo como muchas personas piensan a la hora que se mencionan, simplemente entregan más alternativas de las cuales podemos elegir. Este articulo presenta como las bases de datos relacionales están ampliamente extendidas y cuentan con potentes herramientas que facilitad la interacción. A diferencia, las bases de datos NoSQL son relativamente incipientes y disponen de pocas herramientas de gestión

que no ofrecen muchas de las prestaciones que son encontradas en las herramientas para la gestión de las bases de datos relacionales.

Este artículo aporta el entendimiento que me permite optimizar los resultados de las consultas desde la etapa de diseño en las comparaciones en las bases de datos SQL y NoSQl ya que es muy interesante saber cómo es el funcionamiento de ellas para este trabajo que estoy realizando.

Las bases de datos NoSQL son sistemas de almacenamiento de información que no cumplen con el esquema entidad-relación. Mientras que las tradicionales bases de datos relacionales basan su funcionamiento en tablas, joins y transacciones. Las bases de datos NoSQL no imponen una estructura de datos en forma de tablas y relaciones entre ellas sino que proveen un esquema mucho más flexible.

Las bases NoSQL son adecuadas para una escalabilidad realmente enorme, y tienden a utilizar modelos de consistencia relajados, no garantizando la consistencia de los datos, con el fin de lograr una mayor performance y disponibilidad. A esto se agrega el inconveniente de que no tienen un lenguaje de consulta declarativo, por lo que requiere de mayor programación para la manipulación de los datos. (Martín, Chávez, Rodríguez, Valenzuela, & Murazzo, 2013)

El articulo consiste sobre BigTable que hace parte de las bases de datos NoSQL consiste en gran parte en el área de las bases de datos en cuanto a minería de datos, que hace énfasis en Framework, Almacenamiento Clave-Valor, Almacenamientos de Documentos, Sistemas de Bases de Datos Graficas.

Este articulo seria mucha utilidad para mi proyecto de las comparaciones de bases de datos SQL y NySQL por los datos BigTable que manejan.

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. (Castro Romero, González Sanabria, & Callejas Cuervo, 2012)

NoSQL, Bases de datos, Arquitectura en bases de datos.

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.

Es interesante el aporte de este artículo ya que destaca, el análisis cómo muchas de estas características responden a las oportunidades de investigación en bases de datos NoSQL.

Actualmente, aunque las bases de datos relacionales son utilizadas con éxito para diferentes tipos de escenarios y datos, el concepto NoSQL se escucha con fuerza, a pesar de todas las ventajas que puede ofrecer el modelo relacional, los cuales, a través de su almacén de tuplas y con habilidades de actualización, recuperación de información y estabilidad, pueden complementarse sobre cualquier sistema con gran eficiencia. En este artículo se presenta una revisión global de lo hoy conocido como tendencia NoSQL, analizando sus ventajas en implementaciones de tipo espacial y geográfico. Como objetivo final de este artículo se plantea la identificación de elementos de análisis al momento de desarrollar sistemas espaciales, que le permita al lector identificar si NoSQL es una buena opción como modelo a implementar. (Ramírez Arévalo & Herrera Cubides, 2013)

Consiste en Hacer una exploración general sobre los antecedentes de las bases de datos NoSQL, para luego entrar a examinar el concepto que define las bases de datos NoSQL, identificando algunas de las desventajas que presentan dicho tipo de bases de datos; para terminar con una revisión acerca de la incursión de dichas bases de datos en campos profesionales como los ambientes geoespaciales y catástrateles.

El aporte a la fundamentación que debe enfocar como un conjunto de características para diferentes almacenes de datos, sin un dominio homogéneo. Actualmente existen diferentes categorías de frameworks NoSQL.

En los último años, han surgido nuevas tecnologías basadas en otro tipo de sistemas que no son modelos relacionales los cuales manipulan grandes volúmenes de datos donde su principal característica es la rapidez que se puede almacenar y recuperar grandes cantidades de información, lo que cuenta primordialmente en estas bases de datos es el rendimiento y sus propiedades de tiempo real que la coherencia que las bases de datos relacionales las manejan dedicando mucho tiempo al proceso. distribuidas, mismos que se desarrolla a continuación. (Candia Condori , 2012)

Este artículo consiste sobre manejar bases de datos NoSQL en una nube que está realizando Amazon por medio DynamoDB, que es una base de datos desarrollada internamente por medio de NoSQL.

El articulo presenta que en los últimos años las empresas están proporcionando servicios mucho más agiles y que no sea necesario contar con servidores sino utilizar lo que se encuentra en la nube.

Este artículo le da un valor muy importante a mi trabajo de grado ya que podemos apreciar que las bases de datos NoSQL vienen tomando mucha fuerza, y ver que se pueden utilizar en otros aspectos como es la nube.

Por más de veinte años el escenario de almacenamiento de la información se ha dominado por las bases de datos relacionales. Un novedoso concepto de almacenamiento de datos gana espacio, las llamadas bases de dato NoSQL, especialmente debido a la escalabilidad y velocidad de sus tiempos de respuestas, superiores a las de los sistemas relacionales. En el presente artículo se realiza una introducción a este tipo de tecnologías, haciendo énfasis en MongoDB, CouchDB y Terrastore, bases de datos orientadas a documentos y de código abierto, con gran aceptación en el mercado pues facilitan el trabajo en la Web, sobre todo en aplicaciones de comercio electrónico. Se detallan las características más relevantes de cada una, se brindan elementos para la selección de alguna acorde a las necesidades de entornos reales de producción. (Vazquez Ortíz & Sotolongo León, 2013)

En este artículo podemos apreciar los entornos que manejan las bases de datos NoSQL y como se dividen: llave-valor, llave valor por columnas, grafos y orientadas a documentos.

Este articulo consiste en demostrar las características generales de cada herramienta que manejan las bases de datos NoSQL en los entornos llave-valor, llave valor por columnas, grafos y orientadas a documentos con resultados y discusión con la capacidad de modelación, capacidad de consulta, replicación, fragmentación y almacenamientos de archivos.

El aporte que le da este articulo a mi trabajo es las herramientas que manejan las bases de datos NoSQL y de código abierto orientado a documentos MongoDB, CouchDB y Terrastore, que destaca la capacidad de replicación y fragmentación de los datos, que nos permiten escalar horizontalmente con facilidad.

El desarrollo y crecimiento masivo de las redes de computadoras y medios de almacenamiento, a lo largo de los últimos años, ha motivado la aparición de un creciente interés por los sistemas de clasificación automática de documentos. Las bases de datos tradicionales tienen limitaciones a la hora de indexar y realizar búsquedas en tiempo real sobre grandes volúmenes de datos. De hecho, de un poco a esta parte se ha creado una tendencia en lo que es la búsqueda en tiempo real a migrar de bases de datos relacionales a sistemas de índices invertidos. Por tal motivo en este trabajo se pretende usar una arquitectura de búsqueda que se basa en un índice invertido altamente eficiente en una base de datos NoSQL. (Vazquez Ortíz & Sotolongo León, 2013)

Este artículo consiste en la búsqueda de una solución que proponen los requisitos de escala de almacenamiento de datos, busca añadir un sistema de búsqueda invertida y la solución de las necesidades de búsqueda a escala.

El aporte que le da este artículo a mi trabajo muestra otra parte de cómo se pueden implementar las bases de datos NoSQL, ya que me parece un tema interesante como se pueden utilizar en búsquedas de textos y no solo en el punto de vistas para las grandes compañías como Facebook, Twiter, Google o Amazon.

Motivated by requirements of Web 2.0 applications, a plethora of non-relational databases raised in recent years. Since it is very difficult to choose a suitable database for a specific use case, this paper evaluates the underlying techniques of NoSQL databases considering their applicability for certain requirements. These systems are compared by their data models, query possibilities, concurrency controls, partitioning and replication opportunities. (Hecht & Jablonski, 2011)

Este artículo consiste en cómo tratar con peticiones teras y petabytes de datos, lectura masiva y escribir como tienen que ser respondido sin ninguna latencia notable. En un orden para hacer frente a estos requisitos, y nos presenta evaluar las técnicas subyacentes de estos sistemas, desde enormes cantidades de datos de alto rendimiento con algunos lenguajes que ofrece unos modelos de datos flexibles con grandes posibilidades de consultas.

El aporte que le da este articulo a mi trabajo dar a entender como las bases de datos NoSQL muestran una evolución, y las ventajas y desventajas que nos ofrecen para la comparación de este trabajo.

3 CONCLUSIÓN

El trabajo de grado presentó un nuevo enfoque de las bases de datos no relacionales, útiles para el almacenamiento y gestión de grandes volúmenes de datos, siendo estas bases utilizadas en diferentes sectores de la sociedad, específicamente para manejo de grandes volúmenes de datos a la hora de almacenar demasiados datos como es el caso de Facebook, Twiter, Google, Amazon y especialmente las nubes.

El principal beneficio que se obtiene al momento de utilizar una base de datos NoSQL es la flexibilidad de almacenar grandes cantidades de datos, si se llega a requerir campos extras o informaciones extras no es necesario cambiar la estructura de la base de datos como ocurre en las bases de datos relacionales.

Las bases de datos NoSQL son esenciales para la actualidad ya que todo gira al entorno de redes sociales y es ahí donde son utilizadas para almacenar grandes cantidades de datos, las bases de datos NoSQL no quieren reemplazar las bases de datos relacionales sino, dar otras alternativas para la fluidez, escalabilidad, rapidez y agilidad a la hora de que se necesite guardar tanta información. Presentan una gran ventaja a la hora de almacenar especialmente en la nube que ha tomado tantas fuerzas en estos últimos años para almacenar información y eso no los presta las bases de datos NoSQL.

4 BIBLIOGRAFÍA

Internetría Consultoría Digital. (8 de Mayo de 2013). Recuperado el 15 de Noviembre de 2014, de http://www.internetria.com/blog/2013/05/08/nosql/

SQL Thoughts from IngeniousSQL. (21 de Febrero de 2013). Recuperado el 10 de Noviembre de 2014, de http://www.ingenioussql.com/tag/key-value-store/

BARRAGAN CHARRY, A. M. (2012). *DSpace Repository*. Recuperado el 10 de Diciembre de 2014, de http://repository.ucatolica.edu.co/xmlui/handle/10983/690?show=full

Cairó, O., & Hill, M. (16 de Octubre de 2014). *wikipedia*. Recuperado el 8 de Noviembre de 2014, de http://es.wikipedia.org/wiki/Tabla_hash

Candia Condori , M. L. (2012). NoSQL en la NUBE. *revistasbolivianas*.

Castro Romero, A., González Sanabria, J. S., & Callejas Cuervo, M. (2012). Utilidad y funcionamiento de las bases de datos NoSQL. 21.

Díaz Sepúlveda, W. (2013). BASES DE DATOS NOSQL: LLEGARON PARA QUEDARSE.

Domenjoud, M. (12 de Julio de 2012). *logo Octo Technology*. Recuperado el 10 de Noviembre de 2014, de http://blog.octo.com/en/graph-databases-an-overview/

Enríquez, O. Y., & Gracia del Busto, H. (2012). Bases de datos NoSQL.

Farber, D. (4 de Abril de 2010). *CNET*. Recuperado el 6 de Noviembre de 2014, de http://www.cnet.com/news/is-google-bringing-bigtable-out-of-the-closet/

Gholam, M. (20 de Junio de 2014). *CODe PROYECT*. Recuperado el 10 de Noviembre de 2014, de http://www.codeproject.com/Articles/375413/RaptorDB-the-Document-Store

go, H. (8 de Marzo de 2013). *tHE cAvE ABAP/4*. Recuperado el 18 de Noviembre de 2014, de http://thecaveabap.blogspot.com/2013/03/sap-hana-database-guiade-desarrollo.html

Hecht, R., & Jablonski, S. (2011). NoSQL Evaluation. *Conference on Cloud and Service Computing*, 336.

López, D. (18 de Julio de 2013). *slideshare*. Recuperado el 6 de Noviembre de 2014, de http://es.slideshare.net/dipina/nosql-introduccin-a-las-bases-de-datos-no-estructuradas

Mapanga, I., & Kadebu, P. (2013). Database Management Systems: A NoSQL Analysis. *Communication Technologies & Research*, 17-18.

Martín, A., Chávez, S., Rodríguez, N., Valenzuela, A., & Murazzo, M. (2013). Bases de Datos NoSql en Cloud Computing. 166.

Montoro, S. (7 de Enero de 2011). *La Pastilla Roja*. Recuperado el 8 de Noviembre de 2014, de http://lapastillaroja.net/2011/01/adios-open-source-hola-cloud/

Ramírez Arévalo, H. H., & Herrera Cubides, J. F. (2013). UN VIAJE A TRAVÉS DE BASES DE DATOS ESPACIALES NoSQL.

Ramírez Valenzo, M., Cuevas Valencia, R., & Martínez Castro, J. M. (2011). INTEGRACIÓN DE BÚSQUEDAS DE TEXTO COMPLETO EN BASES DE DATOS NOSQL. *REVISTA VINCULOS*, 81.

Seguin, K. (2011). *The Little MongoDB Book.* Autoedición.

Slater, N., Lehnardt, J., & Anderson, C. (2010). *CouchDB.* United States of America: O'REILLY.

Vazquez Ortíz, Y., & Sotolongo León, A. R. (2013). Mirada a bases de datos NoSQL de código abierto orientadas a. *Serie Científica de la Universidad de las Ciencias Informáticas*, 61.

Velásquez, W. (16 de Enero de 2014). *SlideShares*. Recuperado el 12 de Noviembre de 2014, de http://academica-e.unavarra.es/bitstream/handle/2454/10203/629103.pdf?sequence=1

Veliz , M. (12 de Febrero de 2012). *KILL YOUR IDOLS*. Recuperado el 12 de Diciembre de 2014, de http://academica-e.unavarra.es/bitstream/handle/2454/10203/629103.pdf?sequence=1