UTILIDAD DE LAS BASES DE DATOS NOSQL EN RELACIÓN CON LAS TÉCNICAS DE BIG DATA

JUAN DE JESÚS FERNÁNDEZ GRACIANO

CATALINA SEGURA LONDOÑO

UNIVERSIDAD TECNOLÓGICA DE PEREIRA

FACULTAD DE INGENIERÍAS

INGENIERÍA DE SISTEMAS Y COMPUTACIÓN

PEREIRA

2015

UTILIDAD DE LAS BASES DE DATOS NOSQL EN RELACIÓN CON LAS TÉCNICAS DE BIG DATA

PRESENTADO POR:

JUAN DE JESÚS FERNANDEZ GRACIANO

CATALINA SEGURA LONDOÑO

MONOGRAFÍA REALIZADA PARA OPTAR POR EL TÍTULO DE INGENIERO DE SISTEMAS Y COMPUTACIÓN

ASESOR:

PhD. OMAR IVÁN TREJOS BURITICÁ

INGENIERO DE SISTEMAS

UNIVERSIDAD TECNOLÓGICA DE PEREIRA

FACULTAD DE INGENIERÍAS

INGENIERÍA DE SISTEMAS Y COMPUTACIÓN

PEREIRA

2015

AGRADECIMIENTOS

Dedico este trabajo de grado a Dios y especialmente a mi padre, madre y hermanos quienes desde el inicio de esta etapa de mi vida estuvieron apoyándome incansablemente para lograr esta meta

Un especial agradecimiento y admiración al PhD Omar Iván Trejos por su excelente acompañamiento y orientación en la realización de esta monografía, quien estuvo siempre atento a nuestras solicitudes y aportándonos conocimiento.

Por último a mi novia quien me apoyo en la realización de esta monografía y a todos mis compañeros de estudio de quienes aprendí muchas cosas.

Juan Fernández G.

Principalmente agradezco a Dios por tantas bendiciones recibidas, por guiarme y acompañarme a lo largo de la carrera. A mi mamá, mi papá, mi abuela, mi padrastro y mis padrinos, porque sin su esfuerzo no hubiera sido posible alcanzar este logro en mi vida. Gracias por acompañarme y estar tan pendientes durante todo el proceso de mi carrera profesional.

Un agradecimiento muy especial al PhD Omar Iván Trejos por sus conocimientos, su retroalimentación rápida y su responsabilidad con el proyecto.

A mi novio por sus valiosos aportes, por estar siempre ahí y apoyarme durante el desarrollo de esta monografía. A mis amigos que de una u otra manera me ayudaron, estuvieron pendientes y me alentaban a terminar este proyecto de grado.

Catalina Segura L.

TABLA DE CONTENIDO

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

INTRODUCCIÓN

En la actualidad, la gestión de los datos ha evolucionado en las últimas cinco décadas a lo que se conoce hoy en día comoBig Data, desde la llegada de la web 2.0 la información generada por los seres humanos ha crecido de manera exponencial, alcanzando un volumen del orden de los exabytes (10^{18} bytes) y zettabytes (10^{21} bytes), ya que se recopilan datos de las diferentes redes sociales, áreas de investigación, sector financiero, gobierno, entre otros; es necesario encontrar una manera de almacenar y analizar estas grandes cantidades de información, la cual garantice una recuperación rápida y flexible de información.

Teniendo en cuenta que en la actualidad, las bases de datos más nombradas son las relacionales y no relacionales, es importante analizar cuál de estas satisface las necesidades que trae consigo la era del Big Data. Por ello, a continuación se dará a conocer las principales características de las bases de datos SQL, los diferentes tipos de las bases de datos NoSQL y las propiedades y desafíos del Big Data.

1. GENERALIDADES

1.1 DEFINICIÓN DEL PROBLEMA

Debido a la llegada del término Big Data donde se habla de grandes cantidades de información recolectadas de los análisis realizados a las redes sociales, transacciones bancarias en línea o a través de dispositivos móviles, datos de censo de población y registros médicos, además de la necesidad de los científicos y el sector financiero, de encontrar información precisa basada en el análisis de un conjunto total de datos y no en una muestra, de las grandes cantidades de información que manejan en la actualidad algunas empresas, ha surgido un problema con la forma de almacenar enormes cantidades de información, debido a esto, para una mejor administración de la información es necesario encontrar un sistema de gestión de bases de datos que brinde el rendimiento y la eficiencia necesarias para una rápida lectura y escritura.

Un estudio realizado a principios de Abril del 2015[1] en el cual se analizó la popularidad de los diferentes sistemas de gestión de bases de datos, se llegó a la conclusión que los más utilizados son los basados en el lenguaje de consulta estructurado (SQL) con el 82,8%, en el cual para almacenar la información se hace uso de relaciones y tablas lo que significa que los datos deben tener un formato estructurado. El inconveniente con esta herramienta es que en la actualidad la información que se recolecta con el Big Data es semi-estructurada o no posee una estructura específica y al tratar de salvaguardarla en las bases de datos SQL se puede perder información valiosa.

[1]Clasificación de los sistemas de gestión de bases de datos utilizados en el mundo. [En línea] Disponible en <http://db-engines.com/en/ranking>

Además, al tratar de administrar grandes cantidades de información con los modelos relaciones se presentan problemas de rendimiento y precisión, especialmente la ralentización que se presenta cuando los usuarios desean realizar consultas a tablas con demasiadas filas y lo hacen de manera concurrente, esto provoca un retardo al momento de dar respuesta a la consulta, o incluso no entregar la información solicitada.

1.2 JUSTIFICACIÓN DE LA INVESTIGACIÓN

Desde la llegada de la web 2.0 en el 2004 hasta el año 2015, se recolectan grandes cantidades de información de diferentes áreas de investigación, empresas, redes sociales y el gobierno, lo que hace necesario encontrar una manera de almacenar y administrar los enormesdatos que maneja el Big Data de forma que no se pierda información y que brinde una capacidad de escalar horizontalmente sin perder calidad, un buen rendimiento a la hora de realizar consultas con resultados satisfactorios y usando la menor cantidad de recursos logrando así la eficiencia necesaria.Para almacenar, modificar y consultar esta información, se hace uso de los diferentes sistemas de gestión de bases de datos que existen, los dos más usados hoy en día son SQL con el 82.8% el cual usa el cálculo relacional para realizar distintas operaciones dentro de una cantidad de información almacenada y el No SQL que se encuentra en el porcentaje restante, está enfocado a grandes cantidades de información que no están debidamente estructuradas donde se optimiza de mejor manera la búsqueda sin hacer usodel lenguaje SQL para ello.

Debido a que el Big Data son enormes cantidades de información que deben almacenarse de manera que puedan escalar horizontalmente, además que brinden una buena velocidad a la hora de realizar consultas y que esté disponible en cualquier momento que se deba realizar una nueva entrada, es necesario analizar los beneficios y conocer las características que los sistemas de gestión de

bases de datos brindan a las técnicas Big Data y como ayudan tanto en rendimiento, escalabilidad y rápida lectura/escritura.

1.3 OBJETIVOS DEL PROYECTO

1.3.1 Objetivo general. Desarrollar un documento analítico comparativo en el cual se den a conocer las ventajas y desventajas de las bases de datos SQL y NO SQL en relación con las técnicas Big Data.

1.3.2 Objetivos específicos

- Explicar el sistema gestor de bases de datos SQL y la forma de tratamiento de los datos.
- Describir el sistema gestor de bases de datos NoSQL y sus diferentes tipos.
- Investigar y dar a conocer las técnicas Big Data.
- Realizar una comparación entre los sistemas gestores de bases de datos SQL y NoSQL para determinar cuál cumple con las necesidades de Big Data.

1.4 MARCO REFERENCIAL

1.4.1 Marco teórico. Hasta el año 2003, los seres humanos habían producido alrededor de 5 exabytes (10^{18} bytes) de información, en el 2012, se había expandido a 2.72 zettabytes (10^{21} bytes), este crecimiento se debe al surgimiento de la web 2.0 donde las personas empezaron a compartir información a través de las redes sociales. Estos datos son generados de transacciones en línea, correos electrónicos, videos, audios, imágenes, interacciones en redes sociales, datos científicos, registros médicos, teléfonos móviles y sus aplicaciones; a estas grandes cantidades de información se le dio el nombre de Big Data, el cual cuenta con las siguientes características:

- Variedad: Los datos que se procesan actualmente no se categorizan simplemente en estructurados, ahora también se clasifican en semiestructurados y sin estructura.
- Volumen: El tamaño de los datos paso de megabytes y gigabytes a petabytes, zettabytes.
- Velocidad: La velocidad con la que se capturan los datos y se consultan ha aumentado significativamente.
- Valor: Veracidad de los resultados obtenidos a las consultas realizadas por los usuarios.
- Variabilidad: Inconsistencia en el manejo del flujo de datos.

Estas características hacen que estas enormes cantidades de información sean difíciles de capturar, almacenar, procesar, analizar e interpretar, lo que hace necesario un sistema de gestión de bases de datos que soporte las cinco características nombradas anteriormente; de allí, toman fuerza las bases de datos NoSQL, las cuales no poseen propiedades de las bases de datos relacionales tradicionales, es decir, principalmente no se realizan las consultas con SQL.

Las bases de datos NoSQL no cumplen con el modelo entidad – relación, cada tipo de base de datos NoSQL emplea diferentes modelos de datos; además no utilizan las tablas como estructuras de datos, sino que hacen uso de grafos, clave – valor, documentales, entre otras. Los modelos de datos NoSQL soportan la partición de datos horizontalmente, también están estructurados para soportar enormes cantidades de lectura y escritura concurrentemente.

1.4.2 Marco conceptual.

Bases de datos clave-valor: Los datos son almacenados en pares clave – valor o mapas, también conocidos como diccionarios. Se caracterizan por procesar en tiempo real grandes volúmenes de datos, poseer escalabilidad horizontal a través de los nodos de un clúster, fiabilidad y alta disponibilidad. Son utilizados en aplicaciones que requieren respuestas en milisegundos. [2]

Bases de datos documentales: Estas bases de datos no están compuestas de documentos de texto completo en el sentido tradicional y no actúan como los sistemas de gestión de contenidos. Las bases de datos documentales se utilizan para administrar los datos semi-estructurados principalmente en forma de pares de clave-valorutilizando estructuras sencillas como JSON.[2]

Bases de datos en grafos: La información es representada como nodos del grafo y sus relaciones como las aristas del mismo. En algunas aplicaciones es más importante las relaciones entre los objetos que los objetos mismos. Las relaciones pueden ser estáticas o dinámicas. A este tipo de datos se le conoce como datos conectados. Los datos de Twitter, Facebook, Google y LinkedIn se modelan de forma natural mediante grafos.[2]

Datos estructurados: Datos que tienen bien definidos su longitud y su formato, como las fechas, los números o las cadenas de caracteres. Se almacenan en tablas.[3]

Datos semi-estructurados: Datos que no se limitan a campos determinados, pero que contiene marcadores para separar los diferentes elementos. Es una información poco regular como para ser gestionada de una forma estándar[3].

[2]Venkat N Gudivada, DhanaRao, Vijay V. Raghavan. “NoSQL Systems for Big Data Management”. 10th World Congress on Services, pp. 190-197, 2014.
[3] Purcell, Bernice. The emergence of Big Data technology and Analytics. HolyFamilyUniversity. 2013.

Datos no estructurados: Datos en el formato tal y como fueron recolectados, carecen de un formato específico. No se pueden almacenar dentro de una tabla ya que no se puede desgranar su información a tipos básicos de datos[3].

2. ESTADO DEL ARTE

Para poder profundizar en los temas a analizar, fue necesario consultar distintos documentos para conocer el funcionamiento de los sistemas gestores de bases de datos relacionales y no relacionales y su modo de operatividad dentro de Big Data.

"Bigtable: A Distributed Storage System for Structured Data" es una investigación realizada por Google donde se muestra un nuevo horizonte para las bases de datos no relacionales, más específicamente NoSQL. Google consideró que ya era hora de tener un sistema distribuido que soportara grandes cantidades de información. Las ventajas que brinda BigTable es su alto rendimiento, disponibilidad y escalabilidad, además de esto fueron el punto base para la creación de varios tipos de bases de datos no relacionales entre ellos el de NoSQL Clave/valor, BigTable en su modelo de datos trabaja con una clave y un valor en sus filas y columnas, los cuales son asignados como un arreglo sin estructura donde su análisis depende de la forma en que se quiera trabajar con la información. Este arreglo que contiene la clave/valor de los datos fue adoptado por las bases de datos NoSQL Clave/Valor, tipo de base de datos que se explicará en un capítulo posterior teniendo en cuenta el origen y su evolución desde BigTable hasta su desarrollo por NoSQL. Además ha servido para profundizar aún más en las tecnologías de Big data como el MapReduce.

Al leer el artículo "*Utilidad y funcionamiento de las bases de datos NoSQL"* realizado por Alexander Castro Romero, Juan Sebastián González Sanabria y Mauro Callejas Cuervo se conoció cuál es el uso más relevante de las bases de datos NoSQL y su importancia en la actualidad, basándose principalmente en los cambios que se presentan tanto en la forma de almacenamiento como en su tratamiento. Explicando además los tres aspectos por los que se caracteriza que son la velocidad, el tamaño y la cantidad de la información y la falta de innovación, este articulo muestra estos tres puntos como pilares de que hoy en día hay que adaptarse al cambio. También, considerar que a partir de estos tres

aspectos se originaron las bases de datos NoSQL mostrando así su origen y todas sus características.

“NoSQLSystemsfor Big Data Management” es un artículo desarrollado por Venkat N Gudivada, Dhana Rao y Vijay V. Raghavan en el 2014 el cual brinda bastanteinformación ya que está enfocado a los sistemas NoSQL y como sus diferentes tipos trabajan con Big Data.

Este artículo abarca todo el tema relacionado con NoSQL y Big Data, explica como NoSQL no cumplen con las propiedades ACID sino con las BASE y como se adaptan mejor estas propiedades a este SGBD y a su aplicabilidad en Big Data. Además explica cómo ha sido el crecimiento de Big Data y como a través de su evolución va requiriendo muchas más características, por lo que los sistemas de bases de datos deben estar en constante evolución para suplir estas necesidades. Tanto el concepto como las características de Big Data que se exponen allí se utilizaran más adelante en el capítulo 4 puesto que son definiciones más actualizadas que brindan un mejor entendimiento del tema.

Considerando que el enfoque no es simplemente analizar cómo trabaja un modelo de bases de datos no relacional con Big Data sino analizar también el comportamiento de un modelo relacional con este, fue de gran utilidad el artículo “*A performance comparison of SQL and NoSQL databases”* de Yishan Li and Sathiamoorthy Manoharan sirve para entender y comparar como trabaja cada modelo, tanto el SQL (relacional) como el NoSQL (no relacional), la estructura, características, aplicaciones y los usos de cada uno.

El artículo “*Review of NoSQL Databases and Performance Testing on HBase”* aporta una comparación entre SQL y NoSQL con la siguiente tabla:

:

Tabla 1: Comparativo SQL - NoSQL

	Bases de Datos NoSQL	Base de Datos SQL
Simultaneidad de lectura y escritura	Rápida	Lenta
Almacenamiento de datos	Almacenamiento masivo	Almacenamiento general
Modo de expansión	Expansión horizontal	Expansión vertical
Consistencia	Pobre	Buena
Confiabilidad	Pobre	Buena
Disponibilidad	Buena	Buena
Costos de Expansión	Baja	Alta
Madurez	Baja	Alta
Programación	Compleja	Simple
Modo de Datos	Libre	Fijo
Área humana	Relativamente Alto	Medio
Soporte Técnico	Pobre	Buena
Costos de Actualización	Alta	Baja
Compatibilidad con Software y Hardware	Pobre	Buena
Flexibilidad	Buena	Pobre

En la anterior tabla es posible observar cómo actúan las bases de datos relacionales y no relacionales para algunas características de Big Data, brindando la posibilidad de determinar cuál es el más aplicable.

"Big Data: Issues, Challenges, Tools and Good Practices" es uno de los artículos máscompletos con información sobre Big Data, ya que trata desde los temas básicos como definiciones, hasta los retos y buenas prácticas usadas por Big Data. Una de las partes más importantes de este artículo es el avance de Big Data en la actualidad, puesto que da a conocer en que campos o temas está más

aplicado mencionando como punto fuerte el almacenamiento masivo en industrias TI, aplicaciones a la ciencia como en el Colisionador de hadrones, en temas de privacidad y seguridad, además de las muchas organizaciones públicas y privadas que optan por usar este servicio. Las buenas prácticas para Big Data se enfatizan principalmente en cómo lograr la integración de todos los diferentes tipos de datos(estructurados, semi estructurados o no estructurados) para que la forma en que sean tratados sea única y no exista la necesidad de aplicar distintos tipos de bases de datos para su tratamiento.

"Big Data: A Review" De Seref Sagiroglu y Duygu Sinanc da a conocer a Big Data en su interior, explicando así las tres principales características. Estudia la Variedad, la cual hace referencia a los diferentes tipos de datos que es posible encontrar dentro de Big Data clasificándolos como datos estructurados, semi-estructurados y sin estructura, punto importante que se debe tener en cuenta a la hora de aplicar un sistema de bases de datos; el otro componente es la Velocidad, donde se tiene en cuenta la velocidad de recepción de los datos y de las consultas, siendo parte importante en el análisis de datos de Big Data y por último el componente Volumen el cual hace referencia a las medidas de almacenamiento los cuales están en constante crecimiento, medidas como Terabyte, Petabyte, Exabyte, etc.

El artículo "*Big Data Processing in Cloud Computing Environments"* ayuda a comprender como las tecnologías son aplicadas a Big Data, tecnologías como la computación en la nube que en la actualidad están en auge debido a un completo cambio en la forma de almacenamiento, el tema central es la computación en la nube y como esta cumple con las características que se deben tener en cuenta a la hora de almacenar las grandes cantidades de información que trae consigo Big Data, según el artículo" La computación en nube está asociada con el nuevo paradigma para la provisión de la infraestructura informática y métodos de procesamiento de Big Data para todo tipo de recursos. Por otra parte, algunas

nuevas tecnologías basadas en la nube deben ser adoptadas porque se trata de grandes volúmenes de datos y para el procesamiento concurrente es difícil.” Es entonces el problema que se quiere solventar en Big Data y la computación en la nube, usar las características de Big Data como escalabilidad horizontal, tecnologías como Hadoop o MapReduce entre otras que aplicadas a almacenamiento en la red mejoran la forma en que los datos son almacenados.

Un artículo más específico sobre los problemas de Big Data y la computación en la nube es “*Minimizing Big Data Problems using Cloud Computing Based on Hadoop Architecture”*, este artículo se centra en solventar los problemas de Big Data y la computación en la nube por medio de la arquitectura Hadoop. Algunos de los problemas que se presenta con Big Data y el almacenamiento en la nube son la prioridad, capacidad, concurrencia de solicitudes y numero de nodos haciendo referencia a las maquinas que existen dentro de la nube, al identificar estos problemas su propósito es la implementación de la arquitectura Hadoop dentro de la infraestructura de la nube utilizando MapReduce para que los nodos por los que está compuesta la nube sean aprovechados de manera tal que la escalabilidad horizontal sea aplicada tanto en servicios públicos como privados y en el almacenamiento en general de los datos.

3. SISTEMAS GESTORES DE BASES DE DATOS SQL Y NOSQL

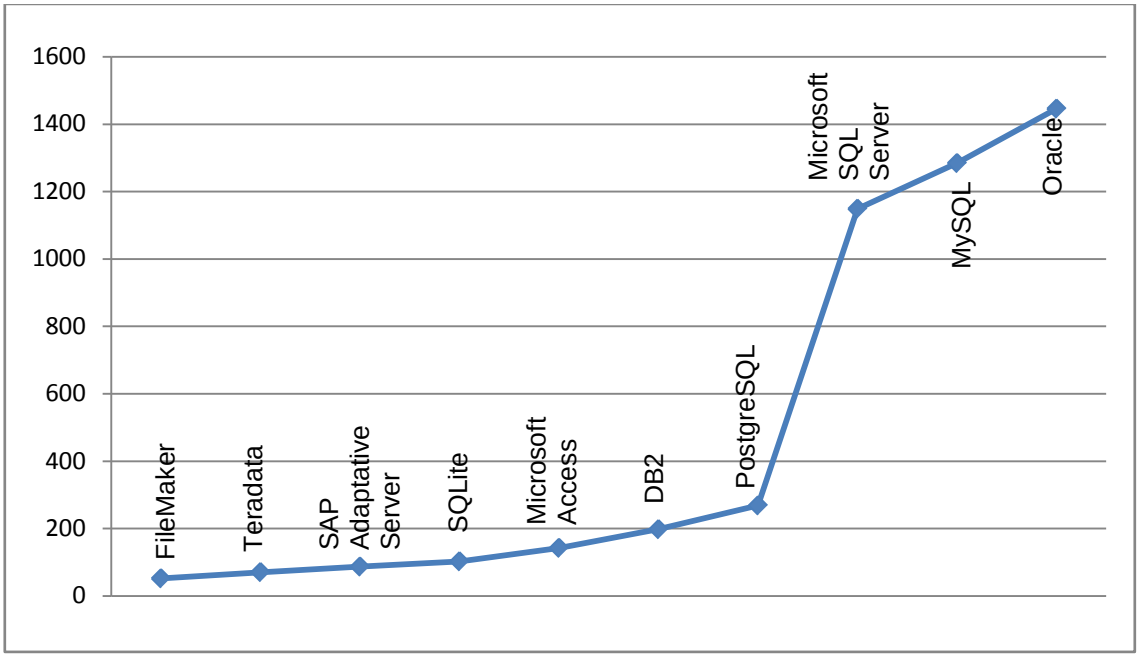
3.1 SISTEMAS GESTORES DE BASES DE DATOS SQL

3.1.1 Breve historia. A comienzos del año 1970, Edgar Frank Codd propuso el modelo relacional en un ensayo titulado “A Relational Model of Data for Large Shared Data Banks”, las investigaciones previas a la publicación se llevaron a cabo en el laboratorio de investigación de IBM. Durante esta época, los sistemas de gestión de bases de datos regían con los modelos jerárquicos y de red, los cuales usaban estructuras de datos y de almacenamiento complejos lo que resultaba ser difícil de comprender para los usuarios. Codd en el modelo relacional define una estructura de datos la cual protege la información y permite la manipulación de esta informaciónde manera que sea predecible y resistente al error.Este modelo está basado en la teoría matemática de las relaciones, con el objetivo de brindar independencia de la estructura lógica respecto a la manera en que se almacena y demás características físicas.

Durante unas ponencias realizadas en 1974 por Donald D. Chamberlin, se dio a conocer el lenguaje de datos para el modelo relacional llamado SEQUEL. Este lenguaje fue utilizado en el prototipo del sistema gestor de base de datos System R desarrollado por los investigadores de IBM a finales de la década de 1970, con el cual lograron demostrar la utilidad del modelo relacional en la implementación de sus estructuras de datos y operaciones. Años más tarde, SEQUEL evolucionó a SQL, el cual fue adoptado en 1986 por el Instituto Nacional Estadounidense de Estándares (ANSI) y en 1987 por la Organización Internacional de Normalización (ISO) como el lenguaje estándar para las bases de datos relacionales, llamado SQL 1. Se realizaron mejoras posteriores a esta versión en los años 1989, 1992, 1999, 2003, 2006, 2008 y 2011.

Según el ranking publicado en Abril del 2015[4], en la actualidad, las bases de datos relacionales son las más utilizadas con un 82,8%, de las cuales el 65,5% son software licenciado y el 34,5% son de código abierto. En el siguiente gráfico se puede apreciar las 10 más usadas según el número de menciones en sitios web, interés general en ellos, frecuencia de discusiones técnicas acerca del sistema, número de empleos ofertados, entre otros.

Ilustración 1: Bases de datos relacionales más utilizadas



Oracle es la base de datos relacional más utilizada según el ranking, la cual es de carácter comercial, es una base de datos que surgió en 1970, ha sido desarrollada para controlar grandes volúmenes de contenidos no estructurados en un único repositorio para reducir riesgos de pérdida de información. Oracle posee una estructura física compuesta por ficheros de tamaño fijo que se establecen al crear la base de datos y que se sitúan en la memoria caché para que los próximos accesos sean más rápidos.

[4]Ranking bases de datos relacionales [En línea]. Disponible en <http://db-engines.com/en/ranking/relational+dbms>

MySQL es la segunda base de datos relacional más utilizada y es de código abierto. De los 10 sistemas gestores nombrados anteriormente, 7 son software licenciado.

3.1.2 Modelo Relacional. El modelo relacional está basado principalmente en dos ramas de las matemáticas: la teoría de conjuntos y la lógica de predicados. Codd buscaba con este modelo cumplir con los siguientes objetivos[5]:

- **Independencia física de los datos**: La manera en que se almacenan los datos no debe influir en la manipulación lógica.
- **Independencia lógica de los datos**: Los cambios realizados en los objetos de las bases de datos no deben repercutir en los programas y usuarios que accedan a la misma.
- **Flexibilidad**: Brindar a los usuarios la información de la manera más adecuada según la aplicación utilizada.
- **Uniformidad**: Presentar los datos en una misma estructura lógica que facilite la concepción y manipulación de la base de datos por parte de los usuarios.
- **Sencillez:** Brindar un lenguaje sencillo para que los usuarios puedan comprender y utilizar la base de datos.

Para el cumplimiento de estos objetivos, Codd propone esquemas en los que se represente completamente la información y las relaciones entre estos datos de una manera estructurada. El elemento básico de este modelo son las relaciones; una relación es un conjunto de columnas y filas representadas estructuralmente como una tabla a través de la cual se simboliza una entidad constituida por los datos recolectados, en la que las filas corresponden a los registros individuales y las columnas a los campos o atributos de estos registros. Cada columna posee un

[5] Ramos, Alicia. Ramos, María Jesús. Montero, Fernando. Sistemas Gestores de Bases de Datos. Madrid: McGraw-Hill, 2006.Pág 53.

nombre distinto dependiendo del atributo que representa, los valores de los atributos son atómicos, es decir sólo pueden tomar un valor. Este modelo no admite tuplas duplicadas y es irrelevante el orden de las filas y columnas.

Existen tres tipos principales de relaciones[6]:

- **Una a una:** Una relación entre dos relaciones en la cual una tupla en la primera relación está relacionada con al menos una tupla en la segunda relación, y una tupla en la segunda relación esté relacionada con al menos una tupla en la primera relación.
- **Una a varias:** Una relación entre dos relaciones en la cual una tupla en la primera relación esté relacionada con ninguna, una o más tuplas en la segunda relación, pero una tupla en la segunda relación esté relacionada con al menos una tupla en la primera relación.
- **Varias a varias:** Una relación entre dos relaciones en cual una tupla en la primera relación esté relacionada con ninguna o más tuplas en la segunda relación, y una tupla en la segunda relación esté relacionada con ninguna, una o más tuplas en la primera relación.

El modelo relacional brinda ventajas significativas frente algunas de las limitaciones que poseían los modelos jerárquicos y de red, ya que las bases de datos relacionales son independientes de la aplicación, las modificaciones que se realicen a la estructura no afectan la aplicación. Además, proporciona rapidez al momento de realizar consultas y es sencillo de entender para los usuarios desde un punto de vista intuitivo, ya que las operaciones de datos se expresan de manera sencilla y no es necesario que los usuarios tengan conocimiento de las estructuras de almacenamiento.

[6]Oppel, Andy. Sheldon, Robert. Fundamentos de SQL. Tercera edición. México: McGraw-Hill, 2010. Págs 11.

3.1.2.1 Normalización. En 1972, E.F Codd desarrolló una técnica para producir un conjunto de relaciones que poseen un conjunto de ciertas propiedades que minimizan los datos redundantes y preservan la integridad de los datos almacenados tal como se mantienen (añadidos, actualizados y eliminados), este proceso es denominado normalización y es la parte central de los principios del modelo relacional.7 Siguiendo los lineamientos de la normalización, se produce un diseño muy flexible, que permite al modelo expandirse cuando sea necesario. Además, al minimizar la redundancia en la base de datos se ahorra espacio y se evitan inconsistencias en la información. Codd definió tres formas normales usando claves primarias:

- **PRIMERA FORMA NORMAL (1FN):** Cada atributo de una tupla contiene sólo un valor, es decir que los dominios de los atributos son atómicos. Además, cada tupla en una relación posee el mismo número de atributos y es única, es decir, la combinación de valores de una tupla no puede ser igual a otra tupla en la misma relación.
- **SEGUNDA FORMA NORMAL (2FN):** Para entender la segunda forma normal, debe entenderse el concepto de **dependencia funcional**, es decir una relación entre atributos de una misma relación. Una relación se encuentra en la segunda forma normal si cumple con la primera forma normal y que todos los atributos en la relación dependen del identificador único completo. La 2FN se aplica a las relaciones que poseen claves primarias compuestas por dos o más atributos.
- **TERCERA FORMA NORMAL (3FN):** Una relación cumple con la 3FN, si cumple con la 2FN y además, cada atributo debe ser independiente el uno del otro y depender del identificador único.

[7]Oppel, Andy. Sheldon, Robert. Fundamentos de SQL. Tercera edición. México: McGraw-Hill, 2010. Págs 7-9.

3.1.2.2 DOCE REGLAS DE CODD. Codd en 1985, publicó las doce reglas que todo sistema gestor de base de datos relacional debe cumplir[8, 9]:

Regla 0: Un sistema de gestión de bases de datos relacional debe gestionar sus datos almacenados sólo con el uso de sus capacidades relacionales.

Regla 1: Representación de información: Toda información de una base de datos relacional debe estar representada mediante valores en tablas

Regla 2: Acceso garantizado: Se garantiza que todos los datos de una base relacional son lógicamente accesibles si se proporciona una combinación de nombre de tabla, valor de clave primaria y nombre de columna.

Regla 3: Representación de valores nulos. Los valores nulos se utilizan para representar la falta de información de un modo sistemático e independiente de los tipos de datos.

Regla 4: Catálogo relacional. La descripción de la base de datos se representa en el ámbito lógico de la misma forma que los datos ordinarios, de modo que los usuarios autorizados pueden acceder a ellos utilizando el mismo lenguaje relacional.

Regla 5: Sublenguaje completo de datos: Un sistema relacional puede soportar varios lenguajes y varios modos de uso terminal. Sin embargo, debe existir al menos un lenguaje cuyas sentencias se puedan expresar mediante alguna sintaxis bien definida, como cadenas de caracteres y que soporte los siguientes ítems:

– Definición de datos.
– Definición de vistas.
– Manipulación de datos (interactiva y por programa).
– Restricciones de integridad.
– Autorización.
– Gestión de transacciones (comienzo, confirmación y vuelta atrás).

[8]Ramos, Alicia. Ramos, María Jesús. Montero, Fernando. Sistemas Gestores de Bases de Datos. Madrid: McGraw-Hill, 2006. Págs. 53–55.
[9] Ricardo, Catherine M. Bases de datos. México: McGraw-Hill, 2006. Págs. 156-157.

Regla 6: Actualización de vistas: Cualquier vista que sean teóricamente actualizable puede ser también actualizable por el sistema.

Regla 7: Inserción, actualización y eliminación: La capacidad de manejar una relación de base de datos o una relación derivada como un único operando se aplican no solamente a la recuperación de datos, sino también a la inserción, actualización y eliminación de los datos.

Regla 8: Independencia física de los datos: Los cambios que se realizan tanto en la representación del almacenamiento, como en los métodos de acceso no deben afectar ni a los programas de aplicación ni a las actividades con los datos.

Regla 9: Independencia lógica de los datos: Los cambios que se realicensobre las tablas de la base de datos no modifican ni a los programas ni a las actividades con los datos.

Regla 10: Independencia de la integridad: Las restricciones de integridad específicas de una base de datos relacional deben ser definidas mediante el sublenguaje de datos relacional y almacenarse en el catálogo de la base de datos.

Regla 11: Independencia de la distribución: Un SGBD es independiente de la distribución.

Regla 12: No subversión: Si el SGBDR permite un lenguaje a bajo nivel que soporte el acceso a registro a la vez, cualquier programa que use este tipo de acceso no puede pasar por alto las restricciones de integridad expresadas en el lenguaje de alto nivel.

3.1.3 Introducción a SQL. El lenguaje de consulta estructurado SQL brinda soporte para la creación y manipulación de las bases de datos relacionales, así como de la gestión y recuperación de los datos almacenados en ellas. SQL es un lenguaje no procedimental o declarativo, es decir se le indica a la computadora lo que se desea obtener o se está buscando, pero no se especifica cómo se debe recuperar dicha información. Algunos proveedores de los sistemas gestores de bases de datos relacionales complementaron el lenguaje SQL para brindar

capacidades de lenguajes procedimentales, por ejemplo Transact-SQL, Microsoft SQL Server y PL/SQL de Oracle.

En ocasiones SQL es denominado un sublenguaje, ya que aún no posee muchas de las capacidades básicas de programación de los lenguajes computacionales, es por ello que frecuentemente se utiliza junto con la aplicación de lenguajes de programación como C y Java para proveer un medio efectivo para el acceso a los datos.

Los estándares SQL publicados hasta el año 1992 se basaron únicamente en el modelo relacional, a partir de la versión del año 1999, se llegó a la conclusión que un lenguaje estrictamente relacional no satisfacía las demandas del mundo real, por esta razón en la edición de 1999 se incluyeron conceptos característicos de la programación orientada a objetos. El hecho de que SQL no soportara tipos de datos complejos y definidos por el usuario, hizo que en la versión del año 2006 se incorporarán estas y otras capacidades orientadas a objetos, como métodos y encapsulación, lo que convierte a SQL en un lenguaje de base de datos relacional a objeto.

3.1.4 Tipos de instrucciones SQL. Aunque SQL sea considerado un sublenguaje debido a que su naturaleza es declarativa, SQL posee un lenguaje completo que le permite crear y mantener objetos en una base de datos, asegurar esos objetos y manipular la información dentro de los objetos. De acuerdo a las funciones que realizan, las instrucciones SQL se dividen en tres tipos: Lenguaje de definición de datos (DDL), Lenguaje de control de datos (DCL) y Lenguaje de manipulación de datos (DML).

Una transacción es una secuencia de una o más instrucciones SQL que conforman una unidad de trabajo. Para garantizar que la base de datos mantiene

un estado correcto a pesar de una falla de concurrencia o del sistema, todas las transacciones deben mostrar cuatro propiedades importantes, por lo general llamadas ACID[10]:

- **ATÓMICIDAD:** Se deben realizar todas las operaciones en una transacción o ninguna de ellas. Se debe ejecutar todo el conjunto de instrucciones o ninguna. El sistema gestor de bases de datos debe ser capaz de cancelar transacciones que no terminen con éxito, y anular sus efectos sobre la base de datos.
- **CONSISTENCIA (INTEGRIDAD):** La base de datos debe ser consistente al inicio y al final de la transacción. Se debe garantizar la consistencia cuando se ejecutan al mismo tiempo transacciones múltiples.
- **AISLADA:** Propiedad que asegura que una transacción no puede afectar a otras. Permite realizar dos transacciones sobre la misma información de manera independiente y sin producir ningún tipo de error.
- **DURABILIDAD (PERSISTENCIA):** Si una transacción ha sido comprometida, el DBMS debe asegurar que sus efectos se registren de manera permanente en la base de datos, aún si el sistema fallara antes de que se hubieran escrito en la base de datos los registros involucrados por la transacción.

3.1.4.1 Lenguaje de definición de datos (DDL). Las instrucciones DDL son utilizadas para la creación, modificación y eliminación de la estructura lógica que constituyen el modelo. Estos comandos pueden ser utilizados en cualquier momento para realizar cambios a la estructura de la base de datos.

- **CREAR TABLA (CREATE TABLE):** Este comando es usado para crear las tablas base que forman la base de datos relacional, puede ser utilizado en cualquier momento del ciclo de vida del sistema. Durante la definición de las tablas base es necesario describir los nombres de cada una de las columnas y

[10]Ricardo, Catherine M. Bases de datos. México: McGraw-Hill, 2009. Pág 366.

los tipos de datos que soportan. Es necesario reforzar la exactitud de la información ingresada a la base de datos, por ello se especifican las restricciones de integridad que protegen al sistema de posibles errores en la inserción de datos que crearían inconsistencias. Se puede aplicar restricciones a columnas individuales, a tablas individuales o a múltiples tablas. A continuación se explican los cinco tipos de restricciones:

- **NOT NULL:** Esta restricción puede ser utilizada únicamente como una restricción de columna, significa que necesariamente debe contener un valor. De manera predeterminada, todas las columnas aceptan valores nulos.
- **CLAVE PRIMARIA (PRIMARY KEY):** Una clave primaria es una columna o una combinación de columnas que identifican unívocamente a cada fila. Debe ser única, no nula y obligatoria. Sólo se puede definir una clave primaria por tabla. Esta restricción puede definirse para columnas o para tablas. Esta clave es importante, ya que cada fila en una tabla debe ser única, puesto que SQL no puede diferenciar entre dos filas completamente idénticas. Al crear una clave primaria, automáticamente se establece un índice que facilita el acceso a la tabla.
- **UNIQUE:** Esta restricción permite exigir que una columna o conjunto de columnas contengan valores únicos, valores diferentes de todas las demás filas en la misma tabla. Se admiten valores nulos. Al crear una restricción de este tipo, también se define un índice automáticamente.

- **CLAVE FORÁNEA (FOREIGN KEY):** Una clave foránea está constituida por una o varias columnas que están asociadas a una clave primaria de otra o de la misma tabla. Se pueden definir tantas claves foráneas como sea necesario. Esta restricción se ocupa de cómo los datos en una tabla hacen referencia a los datos en otra tabla, por esta razón se le conoce como restricción referencial, evitando así que la manipulación de los datos en una tabla afecten negativamente los datos en otra tabla. Las claves foráneas pueden ser creadas como restricción de columna o de tabla.
- **VERIFICACIÓN DE CONDICIONES (CHECK):** Esta restricción permite especificar qué valores se pueden incluir en una columna, definiendo un rango de valores o una serie de condiciones que restringen los valores exactos que permite dicha columna.

- **CREAR ÍNDICE (CREATE INDEX):** Este comando permite crear los índices que facilitan la recuperación de registros de una tabla con valores específicos en una columna de manera rápida. Un índice puede crearse para campos sencillos o combinaciones de ellos, los cuales permiten rastrear cuales valores existen para una columna indexada y que registros tienen dichos valores. Los índices no son parte del estándar SQL, pero la mayoría de sistemas gestores de bases de datos relacionales lo soportan.

- **MODIFICAR, RENOMBRAR Y ELIMINAR TABLA:** Dentro de la estructura de una base de datos relacional es posible cambiar tablas base existentes. La orden ALTER TABLE es utilizada para añadir columnas a una tabla, modificar una o más columnas existentes y eliminar una columna de una tabla, teniendo en cuenta que no se pueden suprimir todas las columnas ni eliminar claves primarias referenciadas por claves foráneas. Además, se pueden agregar, modificar o eliminar restricciones de una tabla del tipo UNIQUE, PRIMARY KEY, FOREIGN KEY, NOT NULL y CHECK. También, se puede modificar el nombre de una tabla por medio de la orden RENAME TABLE. El estándar SQL

también permite la eliminación de una tabla en cualquier momento haciendo uso del comando DROP TABLE.

3.1.4.2 Lenguaje de manipulación de datos (DML). Las instrucciones DML permiten recuperar, insertar, modificar o eliminar datos almacenados en una base de datos.

- **INSERTAR, MODIFICAR Y ELIMINAR DATOS:** El operador INSERT es utilizado para agregar datos a las diferentes tablas en una base de datos, si los nombres de columnas no se enumeran, se deberá introducir un valor para cada columna de la tabla o se puede especificar exactamente las columnas a las cuales se le insertarán datos.

  La instrucción UPDATE permite actualizar los datos en una base de datos relacional, se pueden modificar datos en una o más filas para una o más columnas.

  La orden DELETE es usada para eliminar una fila o varias filas de una tabla. Utilizando la cláusula WHERE se suprimirán sólo aquellas filas deseadas, sin esta condición se eliminarán todas las filas de la tabla.

- **CONSULTA DE DATOS:** Para el acceso a la información de la base de datos, se utiliza la instrucción SELECT. El usuario usa este comando con un nivel de complejidad bajo, ya que simplemente se define que desea obtener, no dónde ni cómo. La única cláusula obligatoria en la sentencia SELECT es la cláusula FROM, donde se especifica la tabla o lista de tablas de las que se recuperarán los datos. Las siguientes cláusulas son opcionales a la hora de usar la sentencia SELECT:

- **WHERE:** Esta cláusula evalúa un conjunto de condiciones sobre todas las filas resultantes de la cláusula FROM, devolviendo así las que cumplen con la condición de búsqueda.
- **GROUP BY:** Esta cláusula es utilizada para agrupar uno o más conjuntos de filas con el fin de resumir datos relacionados.
- **HAVING:** Esta cláusula evalúa un conjunto de condiciones sobre grupos de filas, a diferencia de WHERE que se utiliza para filas individuales. Dicha cláusula es evaluada sobre la tabla resultante del GROUP BY y controla cuál de los conjuntos de filas se debe visualizar.
- **ORDER BY:** Esta sentencia ordena los resultados obtenidos de la consulta según las columnas especificadas dentro de la cláusula ORDER BY, puede ser de manera ascendente (ASC) o descendente (DESC).

Una parte fundamental en toda base de datos relacional es la correlación existente entre dos o más tablas. Una operación *join* hace coincidir las filas en una tabla con las filas de otra tabla haciendo así que las columnas de ambas tablas puedan ser colocadas lado a lado en los resultados de la consulta como si éstos vinieran de una sola tabla[11]. Existen distintos tipos de operaciones *join*:

- **COMBINACIÓN INTERNA (INNER JOIN):** Se realiza un producto cartesiano de todos los registros de las tablas, entregando como resultado únicamente aquellos registros que cumplan con las condiciones especificadas.
  - **DE EQUIVALENCIA (EQUI JOIN):** Utiliza comparaciones de igualdad en la sentencia JOIN. Las comparaciones menor que (<) y mayor que (>) no se clasifican en este tipo de combinación.
  - **JOIN NATURAL:** Coinciden automáticamente las filas de aquellas columnas que posean el mismo nombre. No es posible

[11]Oppel, Andy. Sheldon, Robert. 2010. *Fundamentos de SQL*. Tercera edición. México: McGraw-Hill. Pág 254.

especificar las que columnas que se desean incluir en el resultado de la consulta.

- **CRUZADA (CROSS JOIN):** Realiza un producto cartesiano entre los registros de las tablas pero no se especifica una condición que filtre la consulta.

o **COMBINACIÓN EXTERNA (OUTER JOIN):** Devuelve todas las filas coincidentes con la búsqueda especificada y alguna o todas las filas no coincidentes. Existen tres tipos:

- **IZQUIERDA (LEFT):** Arroja las filas coincidentes con la condición especificada y todas las que filas no coincidentes de la tabla ubicada a la izquierda del comando JOIN.
- **DERECHA (RIGHT):**Devuelve las filas resultantes de la búsqueda junto con las filas no coincidentes de la tabla ubicada a la derecha de la sentencia JOIN.
- **COMPLETA (FULL):** Arroja las filas coincidentes y las no coincidentes de ambas tablas. Pocas implementaciones SQL soportan esta operación.

3.1.4.3 Lenguaje de control de datos (DCL). Las instrucciones DCL son las que permiten definir qué usuario o programa de aplicación tiene acceso a objetos específicos en la base de datos. Además, se puede especificar los tipos de acceso para cada usuario de la base de datos.

Un rol permite la agrupación de los usuarios que requieren los mismos privilegios sobre los objetos de la base de datos. Para la creación de roles, solo es necesario utilizar la cláusula CREATE ROLE. Cuando ya no sea necesario dicho rol, puede eliminar haciendo uso de la sentencia DROP ROLE.

Un privilegio es la capacidad de un usuario para realizar determinadas operaciones o acceder a objetos de otros usuarios de la base de datos. Para

otorgar privilegios a usuarios o roles específicos, se utiliza el comando GRANT, esta cláusula permite conceder todos los privilegios disponibles con la sentencia ALL PRIVILEGES, si se aplica este comando sobre una tabla, se le otorgará al usuario los privilegios SELECT, INSERT, UPDATE, DELETE, TRIGGER y REFERENCES. Si por el contrario, no desea conceder todos los privilegios se debe especificar uno a uno los que desea otorgar. Para revocar los privilegios concedidos, se hace uso de la instrucción REVOKE.

Cuando se agregan, modifican y eliminan datos de las tablas de la base de datos, es necesario guardar los cambios realizados por medio del comando COMMIT. Si se desea que al hacer uso de las instrucciones INSERT, UPDATE y DELETE las modificaciones se almacenen automáticamente, deberá activar el parámetro AUTOCOMMIT. Para deshacer los cambios realizados en la última transacción y regresar al estado de las tablas almacenado en el último commit, se utiliza el comando ROLLBACK.

3.2 SISTEMAS GESTORES DE BASES DE DATOS NOSQL

3.2.1 Historia. Las bases de datos NoSQL son consideradas como un sistema gestor de bases de datos relativamente nuevo. El término No SQL fue usado por primera vez en 1998 por Carlo Strozzi para referirse a su base de datos que poseía características muy particulares, era una base de datos open-source (código abierto), ligera y una de sus características más representativas es que no ofrecía una interfaz SQL, pero sin embargo seguía el modelo relacional usado en otros sistemas gestores de bases de datos. Su concepción fue específicamente como un paquete de software bien definido que a pesar de usar un modelo relacional no utilizaba el lenguaje de consulta SQL lo cual lo hacía diferente.

En el 2009 en la empresa prestadora de servicios de computación en la nube, Rackspace, un empleado llamado Eric Evans utilizó el término NoSQL para expresar una nueva generación de este concepto en el evento organizado por Johan Oskarsson de Last.fm para dar a conocer bases de datos de código abierto, en este evento se pretendía mostrar el número creciente de bases de datos no relacionales y distribuidas que no garantizaban ACID, un atributo clave en las RDBMS clásicas.

Strozzi sugirió que el término expresado en este evento debería considerarse NoREL ya que se salía completamente del modelo relacional y se daba una concepción distinta al término NoSQL planteado por él años atrás, el cuál ofrecía un nuevo sistema de bases de datos relacional que no hacía uso del lenguaje de consultas propio de este, razón por la cual llevaba este.

3.2.2 Definición. NoSQL es llamado también “no solo SQL”, y se posiciona como la siguiente generación de tecnologías relacionadas con las bases de datos. Este subconjunto de bases de datos es diferente a varios modelos de bases de datos tradicionales con características muy importantes como por ejemplo, no usa lenguaje de consultas SQL (no usa JOINs), no implementa ACID y su forma de escalar es horizontalmente, entre otras; estas características se explicarán a fondo más adelante.

Este sistema de gestión de base de datos surgió como una necesidad latente de las principales compañías de internet como Google, Facebook, Twitter y Amazon al ver que debido al crecimiento de la web de manera exponencial, se hacía necesario darle tratamiento a los datos de una forma diferente a las RDBMS tradicionales ya que no solucionaban o presentaban inconsistencias a la hora de dar un resultado. Su necesidad era proporcionar información tanto al usuario como al proveedor de servicio de manera rápida y confiable, para lograr esto era

necesario que se procesaran los grandes volúmenes de datos que se manejanen la actualidad y no solamente muestras.

Con el término NoSQL se llegó a la conclusión que el rendimiento de los datos y la velocidad de procesamiento de información era más importante que tener orden o coherencia en los datos, en lo cual las bases de datos SQL gastan demasiado tiempo ordenando datos en tablas lo que disminuye la velocidad de procesamiento de información.

3.2.2.1 Uso principal de NoSQL. Las grandes cantidades de información son la principal característica en la cual se concentran las bases de datos NoSQL es por eso que

- Indexación de grandes cantidades de documentos
- Trabajo con páginas web de mucho tráfico o demanda de solicitudes
- Envío de streaming

Son tareas que las RDBMS tradicionales optimizan para trabajar de una mejor manera pero que no logran una calidad adecuada de respuesta. NoSQL brinda servicio tanto a lectura y escritura de esta información debido a su capacidad de escalar horizontalmente.

3.2.3Características

No maneja tablas: La forma en que se guardan los datos es distinta a un RDBMS tradicional, ya que no hay que crear tablas por adelantado para estructurar la información, ni luego implementar un código de búsqueda en el servidor para extraer datos. Sus datos se almacenan de forma más flexible sin tener que usar un formato predefinido para esto.

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

Habilidad de distribución: Hace referencia a la escalabilidad horizontal que lo caracteriza, pero hace énfasis en su soporte; para ello se tiene en cuenta la habilidad de replicar y distribuir los datos sobre los servidores o nodos.

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

Libertad de esquema: Al no poseer un esquema rígido se permite mayor libertad en el modelado de los datos; además facilita la integración con los lenguajes de programación orientados a objetos, lo que evita el proceso de mapeado.

Modelo de concurrencia débil: No se implementan las características ACID (Atomicity, Consistency, Isolation and Durability), las cuales son necesarias para que una serie de instrucciones puedan ser consideradas una transacción, sin embargo NoSQL tiene en cuenta algunas consideraciones para asegurar estos aspectos, pero no de manera tan estricta, para esto implementa BASE (Basic Available, SoftState, Eventual Consistency).

Consultas simples: Las consultas requieren menos operaciones y son más naturales, lo que lo hace más simple y eficiente.

No genera cuellos de botella: Al ganar simplicidad y eficiencia con las consultas simples, no habrá tráfico de instrucciones para la administración de un SGBD NoSQL que puedan hacer que el sistema se ralentice. [12]

[12]Romero, Alexander. Gonzalez, Juan Sebastian. Callejas, Mauro. Utilidad y funcionamiento de las bases de datos NoSQL. Págs. 24 -25. 2012.

Las bases de datos NoSQL se consideran como varios sistemas gestores de bases de datos que intentan o ayudan a mejorar las limitaciones que presenta el modelo relacional al dar tratamiento a grandes cantidades de información, ya que entre más datos almacenados es necesario más escalabilidad en los servidores. Los RDBMS no hacen balanceo de carga en estos casos.

3.2.4 NoSQL y el principio BASE. En el punto 3.1.4 se define que es una transacción y para ser considerada como tal debe cumplir con el principio ACID. Partiendo de allí las instrucciones de NoSQL no garantizan ACID debido a la sencillez de sus instrucciones y por eso se adaptan mejor al principio BASE (Basic Availability, Soft State, Eventually Consistency).

DISPONIBILIDAD BÁSICA (BASIC AVAILABILITY): La prioridad ante todo es la disponibilidad de los datos.

ESTADO SUAVE (SOFT STATE): El motor de la base de datos será el encargado de la consistencia de los datos, priorizando así la propagación de los datos.

CONSISTENCIA CON EL TIEMPO (EVENTUALLY CONSISTENCY): Hace que las inconsistencias eventuales evolucionen hasta llegar a un estado final estable (Convergencia).

3.2.5Tipos de bases de datos NoSQL

3.2.5.1 Bases de datos Clave/Valor. Las bases de datos Key/Value (Clave/Valor) son las más usuales, trabajan con HashMap (Tablas Hash) donde cada elemento es identificado por una llave única, esta llave permite que al momento de consultar o hacer una recuperación de información lo haga de manera muy rápida debido al beneficio que le brinda su llave única asignada.

Normalmente el valor se almacena como un objeto BLOB.[13]Lo que hace importante la utilización de un objeto BLOB como almacenamiento es que no interesa que contenga la base de datos ya que se almacenará por clave y se recuperará la información de acuerdo al valor de esta.

Las bases de datos Key/Value están basadas en un estudio realizado por Google sobre un nuevo gestor de base de datos el cual denominaron como Big Table, un BigTable es un mapa ordenado persistente escaso, distribuido y multidimensional. El mapa está indexado por una clave de fila, columna de clave, y una marca de tiempo; cada valor en el mapa es una matriz de bytes no interpretados.[14]

Estas bases de datos son usadas principalmente en lecturas y escrituras, su escalabilidad es muy sencilla y una de sus principales características es que pueden particionar la información para que sea almacenada con base a su clave.

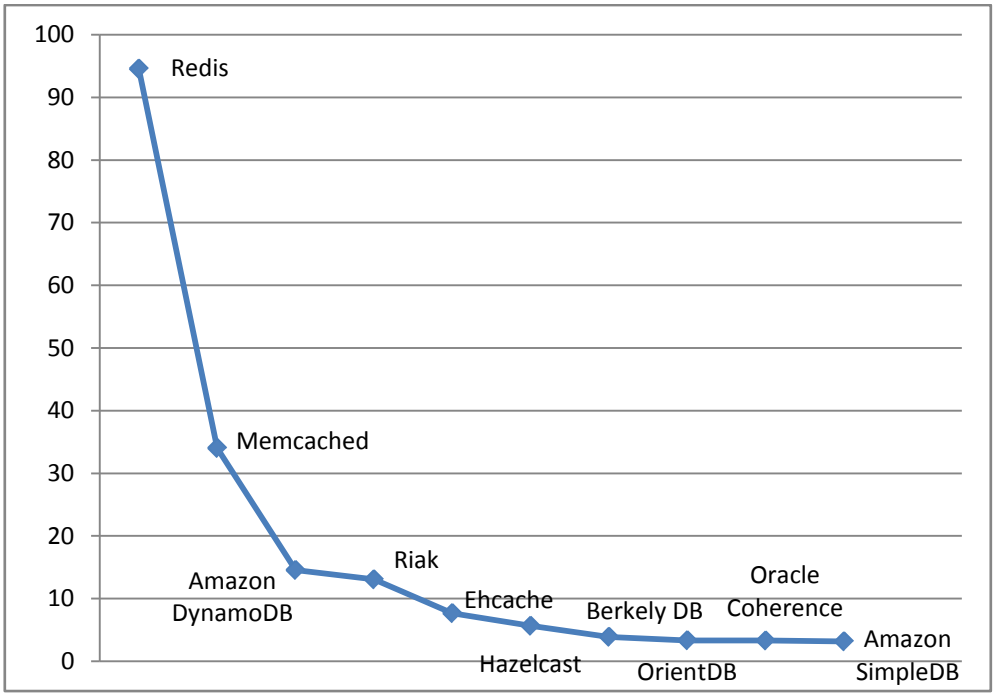
En la siguiente gráfica, se da a conocer las 10 principales bases de datos clave/valor utilizadas en la actualidad. La más usada es Redis. Redis es una base de datos de código abierto, con licencia BSD (licencia de software libre), avanzada de clave-valor cache y almacenamiento. Se refiere a menudo como un servidor de estructura de datos que puede contener claves a partir de cadenas, hashes, listas, conjuntos y conjuntos ordenados, mapas de bits y hyperloglogs.[15]

[13] BLOB: BinaryLargeObject (Objetos Binarios Grandes) Son tipos o identificadores utilizados en las bases de datos para almacenar datos de gran tamaño que pueden cambiar de forma dinámica.

[14]Chang, Fay.Dean, Jeffrey.Ghemawat, Sanjay. Hsieh, Wilson. Wallach, Deborah. Burrows, Mike. Chandra, Tushar.Fikes, Andrew. Gruber, Robert. (2006), Bigtable: A Distributed Storage System for Structured Data.Págs. 1 – 6.

[15]Página oficial Redis [En línea]. Disponible en: <http://redis.io/topics/introduction>

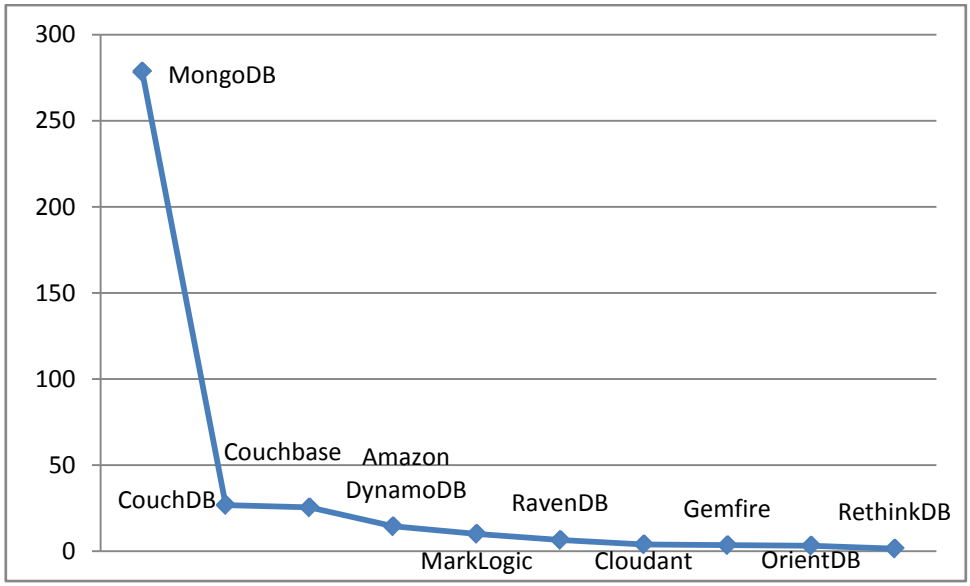
Ilustración 2: Bases de datos clave/valor más utilizadas



Nota: Amazon DynamoDB y OrientDB funcionan como documentales y clave valor y además la última también puede trabajar como una base de datos orientada a grafos.

3.2.5.2 Bases de datos documentales. La característica principal de estas bases de datos es que se almacena la información como un documento, este almacenamiento puede ser con JSON o XML. A cada documento se le asigna una clave única, la asignación de estas claves se realiza de manera similar a las bases de datos clave/valor, pero a diferencia de estas, en las bases de datos documentales es necesario que la información almacenada sea entendible. Entender esta información ofrece una ventaja significativa sobre las clave/valor, ya que el servidor puede operar con estos datos permitiendo así realizar consultas más avanzadas sobre los datos, compararlos y relacionarlos sin la necesidad de usar JOINs para esto.

Ilustración 3: Bases de datos documentales más utilizadas



MongoDB es un sistema de gestión de base de datos NoSQL lanzado en 2009. Almacena datos como documentos JSON (esquemas dinámicos este formato se llama BSON). MongoDB tiene su enfoque orientado a cuatro cosas: la flexibilidad, potencia, velocidad y facilidad de uso. Soporta servidores replicados, la indexación y ofrece un buen reconocimiento para múltiples lenguajes de programación.[16]

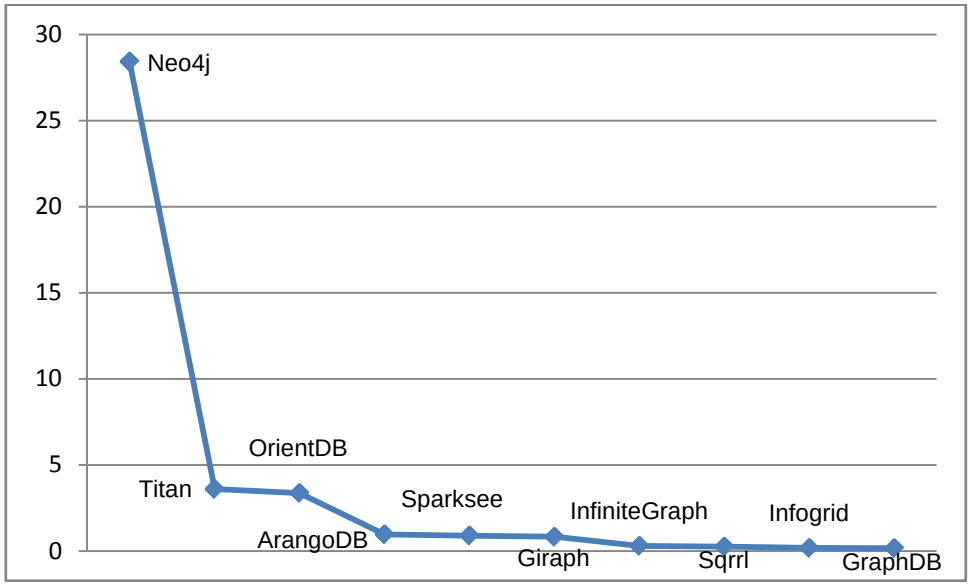
3.2.5.3 Bases de datos orientadas a grafos. Estas bases de datos están inspiradas en la teoría de grafos y en Euler. Su propósito principal es almacenar datos que tienen una topología en grafo, esto quiere decir que la información está representada por medio de nodos y por aristas para relacionar la información.

Ofrece un buen rendimiento cuando los datos están interconectados (enlaces o relaciones) y no tabulares. Además resulta más eficiente navegar entre relaciones de esta manera en comparación como lo realiza un SGBD de un modelo relacional.

[16]Boicea, Alexandru. Radulescu, Florin. Ioana, Laura (2012) MongoDB vs Oracle - database comparison. Pág 1.

Al utilizar topologías de grafos no se hace necesario la utilización de índices o identificadores en los nodos como referencia. Estas bases de datos son muy útiles para representar la información en redes sociales, ya que en las relaciones (aristas) se pueden hacer consultas directamente y también asignar atributos a las relaciones como si se tratara de un nodo, así en vez de consultar al nodo se puede ir directamente a la relación.

Ilustración 4: Bases de datos orientadas a grafos más utilizadas



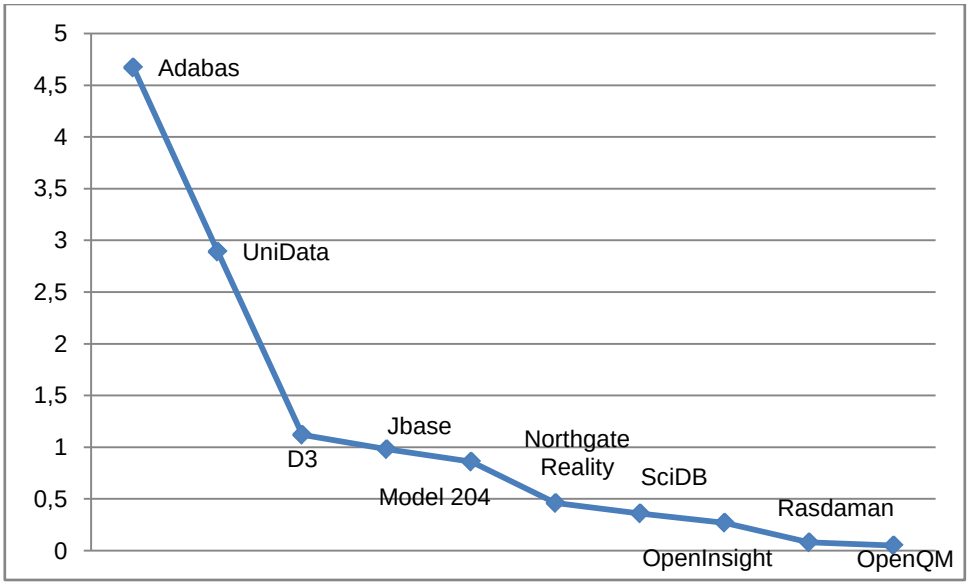
Neo4j es una base de datos de código abierto orientada a grafos implementado en Java y Scala. Desarrollada a partir de 2003, publicada desde el año 2007. Neoj4 abarca diversos casos de uso entre ellos gestión de redes, análisis de software, la investigación científica, enrutamiento, gestión organizacional y de proyectos, redes sociales y más.

Neo4j a diferencia de graficar procesamiento o bibliotecas en memoria, ofrece características de base de datos completa con ACID cumplimiento transacción, apoyo a las agrupaciones y de conmutación por error de tiempo de ejecución

haciendo estas bases de datos la más adecuada para utilizar los datos del grafo en escenarios de producción. [17]

3.2.5.4Bases de datos multivalor. Estas bases de datos tienen la propiedad de usar multivalor, la cual hace que un dato pueda estar representado por varios valores al mismo tiempo garantizando que sus características estarán contenidas en la información correspondiente a pesar de ser información independiente.

Ilustración 5: Bases de datos multivalor más utilizadas



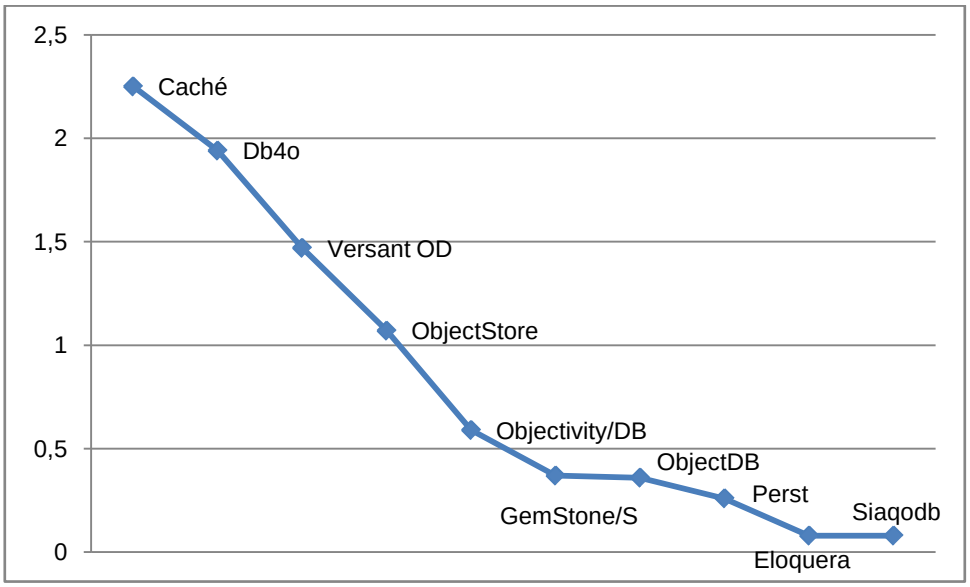
Adabas pertenece a las bases de datos NoSQL multivalor ampliamente usada en aplicaciones que requieren altos volúmenes de procesamiento de información, es reconocido por esta funcionalidad debido a que no trabaja con tablas (filas y columnas) sino con archivos anidados, el cual hace una búsqueda exhaustiva pero rápida de la información solicitada.

[17] http://neo4j.com/developer/graph-database/#_what_is_neo4j

3.2.5.5 Bases de datos orientadas a objetos. Estas bases de datos siguen el enfoque de NoSQL perfectamente, ya que al estar orientadas a objetos eliminan la limitación por los tipos de datos y los lenguajes de consulta de los sistemas de bases de datos tradicionales.

Estas bases de datos almacenan y manipulan información que pueden ser mostradas como objetos, además su acceso y manipulación de datos es ágil, incorpora en su sistema las características principales de los modelos orientados a objetos como lo son la herencia, encapsulación, polimorfismo y la forma de consulta por medio de funciones y/o métodos.

Ilustración 6: Bases de datos orientadas a objetos más utilizadas



InterSystems Caché o Caché es un sistema de gestión de base de datos avanzado el cual logra hacer avances en el procesamiento y análisis de Big Data, Ofrece un rendimiento rápido, escalabilidad masiva y fiabilidad necesitando

requisitos mínimos de mantenimiento y hardware el cual ofrece múltiples modos de acceso a datos.[18]

3.2.5.6 Bases de datos tabulares. Su administración es similar a la administración de una base de datos multidimensional. Una base de datos multidimensional es una base de datos donde la información se representa como matrices multidimensionales, cuadros de múltiples entradas o funciones de varias variables sobre conjuntos finitos. Cada una de estas matrices se denomina Cubo.[19]Un cubo da a conocer sus ejes y estructuras, los datos en cada estructura de un cubo son uniformes. Por cada dimensión tiene un campo y una métrica o hecho, que es donde se almacenan los registros, en el campo se hace referencia a las dimensiones de cada estructura del cubo y en la métrica o hecho se refiere a lo que se quiere almacenar y/o analizar.

Con base a la definición anterior una base de datos tabular es un modelo relativamente sencillo de construir, ya que aplica las mismas herramientas y enfoques de las bases de datos multidimensionales. El enfoque principal de estas bases de datos son las consultas complejas y de alto rendimiento.

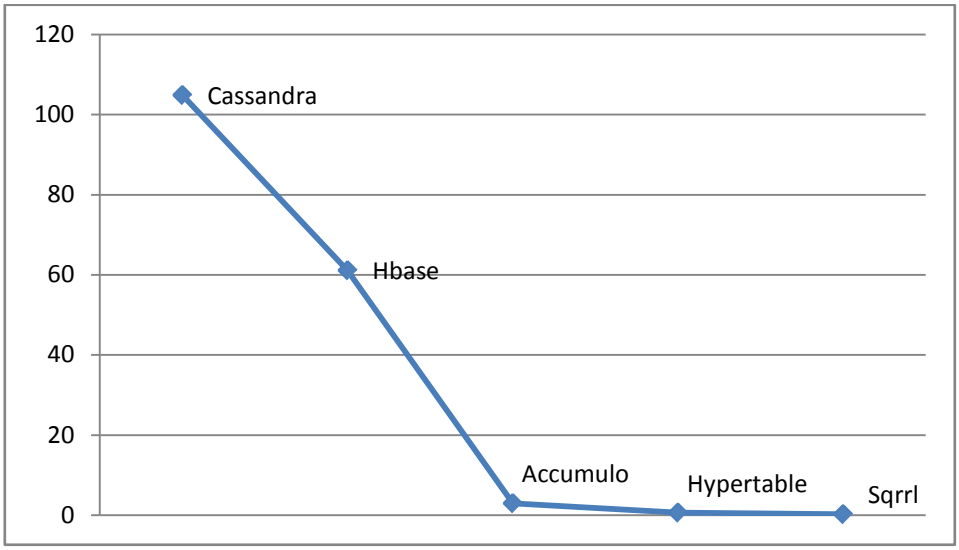
En la siguiente gráfica, se dan a conocer las 10 bases de datos tabulares más utilizadas según la página db-engines. La más usada es Cassandra. Cassandra es un sistema de bases de datos distribuido, que funciona como base de datos tipo clave/valor y tabular haciéndola altamente escalable.

[18] http://www.intersystems.com/our-products/cache/cache-overview/
[19]Carpani, Fernando. (2000) CMDM: Un Modelo Conceptual para la Especificación de Bases de Datos Multidimensionales. Págs. 13 – 16.

Cassandra reúne las tecnologías de sistemas distribuidos de Dynamo y el modelo de datos de BigTable de Google. Fue abierta en 2088 proveniente de Facebook.[20]

El modelo de datos de Cassandra ofrece comodidad con los índices de columnas y el rendimiento en los cambios de registros de datos estructurados, soporte a la desnormalización, vistas materializadas y potente capacidad de almacenamiento en caché convirtiéndose en una plataforma perfecta para la infraestructura en la nube. [21]

Ilustración 7:Bases de datos tabular más utilizadas



[20] Wang, Gouxi. Tang, JianfengTheNoSQL Principles and Basic Application of Cassandra Model. 2012. Pág 2.
[21]Página principal Cassandra [En línea]. Disponible en<http://cassandra.apache.org/>

4. BIG DATA

4.1 INTRODUCCIÓN

La cantidad de información generada por los seres humanos hasta el 2003 era de aproximadamente 5 exabytes (10^{18} bytes), a partir de la llegada del concepto de la Web 2.0 en el 2004, esta cifra creció de tal manera que en el 2012 se producían 2.72 zettabytes (10^{21} bytes). La web 2.0 permite una interacción entre los usuarios a través de las redes sociales, además de colaboración entre sí de los contenidos de las diferentes comunidades virtuales.

Estas enormes cantidades de información, llamadas Big Data son recolectadas de las diferentes redes sociales, hasta el año 2013 Twitter cada 72 horas recolectaba un billón de tweets de los aproximadamente 140 millones de usuarios activos; Facebook poseía 955 millones de cuentas activas mensuales, las cuales hacían uso de 70 lenguajes, 140 billones de fotos subidas, 125 billones de conexiones entre amigos y cada día 2.7 billones de “likes” y comentarios eran publicados; Cada minuto, 48 horas de video eran cargados a Youtube y cada día, 4 billones de vistas se producían a los videos. Google monitorizaba alrededor de 7.2 billones de páginas por día y procesaba 20 petabytes (10^{15} bytes) de datos diariamente.[22] También se reúnen grandes cantidades de información de diferentes áreas de investigación, por ejemplo, el CERN puede llegar a producir 40 terabytes de datos por segundo durante los experimentos.[23]Las organizaciones dedicadas a la investigación y el desarrollo (I+D) han tenido problemas al momento de obtener suficiente potencia de cálculo para ejecutar modelos complejos o para el procesamiento de datos científicos. La dificultad no radica únicamente en el almacenamiento, sino en procesar y transformar estos datos en información de valor.

[22] The Human Face Of Big Data [EnLínea]. Disponible en: <http://www.humanfaceofbigdata.com/>
[23] Torres, Jordi. Del cloud computing al big data. 2012. Pág. 22.

Algunos ejemplos de Big Data en la literatura son astronomía, ciencia de la atmósfera, genómica, ciencias biológicas, ciencias de la vida, registros médicos, investigación científica, gobierno, desastres naturales, sector privado, vigilancia militar, servicios financieros, teléfonos móviles, sensores y redes de telecomunicación.

La gestión y el análisis de estos datos ofrecen grandes beneficios a las diferentes organizaciones de todas las industrias, ya que se obtiene información precisa al realizar un análisis completo al conjunto total de datos y no simplemente a una muestra, facilitando así la toma de decisiones y la solución de problemas en tiempo real.

4.2 EVOLUCIÓN DE LA GESTIÓN DE LOS DATOS

Cada era en la gestión de los datos nace de la necesidad de tratar de resolver un problema específico en el manejo de la información. Cuando una nueva solución tecnológica llega al mercado, se requiere descubrir nuevos enfoques. Cuando surgieron las bases de datos relacionales, se hizo necesario un conjunto de herramientas que permitiera a los administradores estudiar las relaciones entre los datos. Cuando las empresas empezaron a almacenar datos no estructurados, los analistas necesitaron nuevas herramientas con mejores capacidades de análisis con el fin de obtener información útil para las empresas en un lenguaje natural.

Para entender el Big Data, es necesario conocer el fundamento de las eras anteriores y las razones por las que se evoluciona, ya que en ellas se basa la nueva era**.**

4.2.1 Creación de estructuras de datos. A finales de 1960, cuando la computación empezó a trasladarse al mercado comercial, los datos se almacenaban en archivos planos sin ninguna estructura. Cuando las empresas requerían información detallada de clientes, debían aplicar métodos de fuerza bruta, como modelos de programación muy detallados para la creación de algún valor. Se realiza un cambio significativo en la década de 1970 con el nacimiento del modelo relacional y del sistema gestor de base de datos relacional (RDBMS), ya que se implantaba el concepto de estructura y un método para mejorar el rendimiento. Además, el modelo relacional añadía un nivel de abstracción con su lenguaje estructurado de consultas (SQL), a través del cual se podían generar informes y gestionar los datos, de tal manera que los programadores pudieran satisfacer las crecientes demandas del negocio para extraer valor de la información.

El modelo relacional ayudó a las empresas a almacenar su información de una manera ordenada permitiendo así que los gerentes de empresas pudieran examinar estos datos y en base a ello, tomar las decisiones necesarias. Pero este modelo presentaba varios inconvenientes, almacenar ese volumen de datos era caro y el acceso a la información era lento, además, muchos de los datos almacenados estaban duplicados.

Cuando el volumen de los datos que manejaban las empresas creció de manera exponencial, surgieron en la década de 1990 los almacenes de datos (Data Warehouse, los cuales permitían a la organización seleccionar el subconjunto de datos que deseaba almacenar, permitiendo así obtener valor de dicha información de manera más fácil. Estos almacenes de datos ayudan a las empresas reduciendo los volúmenes de datos centrándolos únicamente en áreas particulares de las mismas. Además, almacenan información de años anteriores para entender el desempeño organizacional, identificando tendencias y patrones de comportamiento.

Estos almacenes de datos en sí eran demasiado complejos y grandes y no ofrecían la velocidad y la agilidad que se requería. Para resolver este problema surgieron los mercados de datos (Data Marts), los cuales realizaban un refinamiento adicional a los datos. Estos mercados de datos eran mucho más ágiles ya que se centraban en cuestiones específicas de negocio y brindaban consultas rápidas.

Aunque los almacenes de datos y los mercados de datos resolvían muchos problemas de las empresas, al momento de gestionar grandes volúmenes de datos no estructurados o semiestructurados, estos no eran capaces de evolucionar lo suficiente como para satisfacer la demanda. Los almacenes de datos se alimentaban en ciertos intervalos de tiempo, por lo general semanal o diariamente, lo cual lo hacía demasiado lento para los actuales entornos empresariales.

Para almacenar datos no estructurados, se añadieron capacidades como BLOB (BinaryLargeObjects, Grandes Objetos Binarios) de manera tal que se almacenaba esta información en las bases de datos relacionales como una porción contigua de datos. Este objeto podría ser etiquetado, pero no se podía visualizar lo que había dentro de este.

Las bases de datos de objetos almacenan el BLOB como un conjunto accesible de piezas, permitiendo visualizar lo que se encuentra allí. Estas bases de datos incluyen un lenguaje de programación y una estructura para los elementos de los datos, haciendo más fácil la manipulación de diversos objetos de datos sin necesidad de programación ni de complejos *joins.* Se introduce un nuevo nivel de innovación que ayudó al desarrollo de la segunda era de la gestión de datos.

4.2.2 Gestión del contenido web. En la década de 1990 con el auge de la web, las organizaciones buscaban ir más allá de los documentos, esperaban almacenar y administrar contenido web, imágenes, audio y video.

El mercado evolucionó a partir de un conjunto de soluciones desconectadas creando así un modelo unificado el cual incorporaba la gestión de procesos del negocio, control de versiones, reconocimiento de la información, gestión de texto y colaboración. Esta nueva generación añadió metadatos, es decir, información de la organización y características de los datos almacenados. Nuevos requisitos han surgido, como la convergencia de factores, incluyendo la web, la virtualización y el Cloud Computing, los cuales nos conducen a la nueva etapa.

4.2.3 Gestión del Big Data. Big Data se encuentra en la parte superior de la evolución de la gestión de los datos de las últimas cinco décadas. El costo de los ciclos computacionales y el almacenamiento de la información han alcanzado un punto de inflexión. Esto es realmente importante, ya que hace unos pocos años, las organizaciones solo almacenaban subconjuntos de información significativos debido al alto costo de almacenamiento y procesamiento que limitaba salvaguardar todo lo que se deseaba analizar.

Gracias al Big Data, ahora es posible la virtualización de los datos, ya que se pueden almacenar de manera eficiente y al hacer uso de la computación en la nube, los costos se reducen, haciéndolo más rentable. Las mejoras en la velocidad y la fiabilidad de la red han eliminado otras limitaciones físicas, permitiendo así manejar grandes cantidades de datos a un ritmo aceptable. Añadiendo además el impacto de los cambios en el precio y la sofisticación de la memoria del computador. Con estos avances en la tecnología, las empresas hoy en día pueden aprovechar toda la información que hace cinco años atrás no podían almacenar. Las organizaciones quieren poseer la capacidad de analizar los

diferentes tipos de datos a una velocidad aceptable sin importar la cantidad, obteniendo así conocimientos a partir de ellos que permitan una adecuada toma de decisiones.

Si las empresas pueden analizar petabytes de datos con un rendimiento aceptable para determinar patrones y anomalías, las organizaciones pueden empezar a dar sentido a los datos de nuevas maneras. El Big Data no se trata únicamente de la información de los negocios, también se incluyen las actividades científicas, de investigación, y del gobierno. Basta con pensar en el análisis del genoma humano o de todos los datos astronómicos recogidos en los observatorios para avanzar en la comprensión del mundo que nos rodea. También se puede considerar la cantidad de datos que el gobierno recauda en sus actividades.

Existen diferentes enfoques en el manejo de los datos, dependiendo si se tratan los datos en movimiento o en reposo. Un ejemplo de los datos en movimiento es en el momento en que una empresa fuese capaz de analizar la calidad de sus productos en el proceso de fabricación, evitando así errores costosos. Un analista de negocios puede hacer uso de los datos en reposo para comprender mejor los patrones de compra de los clientes actuales con base en todos los aspectos de la relación con el cliente, incluyendo ventas, datos de las redes sociales y las interacciones con el servicio al cliente.

4.3 CARACTERÍSTICAS DEL BIG DATA

Big Data no es una sola tecnología, sino una combinación de nuevas y viejas tecnologías. Big Data es la capacidad de manejar grandes cantidades de datos, a la velocidad adecuada, y dentro de un margen de tiempo que permita el análisis en tiempo real. En la literatura, Big Data se divide tres características: Volumen, Velocidad y Variedad.

4.3.1 Volumen. Esta es la principal característica del Big Data, que como su nombre lo indica, son enormes cantidades de información que pasaron de ser del orden de los megabytes y gigabytes a petabytes y zettabytes. Las redes sociales producen diariamente grandes cantidades de datos que son difíciles de almacenar y manejar con los sistemas tradicionales.

El beneficio obtenido de la capacidad de procesar grandes cantidades de información es el análisis de estos enormes datos. Analizar más datos ayuda a generar mejores modelos. Esta característica presenta el desafío más inmediato a las estructuras tradicionales, es necesario un almacenamiento escalable y un enfoque distribuido para las consultas. En la actualidad, muchas empresas ya cuentan con grandes cantidades de datos archivados, pero no con la capacidad para procesarla.

Teniendo en cuenta que los volúmenes de datos son más grandes que la capacidad de la infraestructura de las bases de datos convencionales, es necesario migrar a arquitecturas de procesamiento masivo en paralelo como almacenes de datos, soluciones basadas en el Apache Hadoop u otros tipos de bases de datos. Un factor importante en esta elección es la variedad de los datos, ya que los almacenes de datos imponen esquemas predeterminados, satisfaciendo así un conjunto de datos regular y de lenta evolución, en cambio,

Hadoop no posee unas condiciones específicas a las estructuras de datos a procesar.

4.3.2 Velocidad. La tasa a la cual los datos fluyen dentro de una organización sigue un patrón similar al del volumen. Esto es debido al internet y la era móvil, es decir a la manera en que se prestan y se consumen productos y servicios, la cual se incrementa cada vez más, generando así un flujo de datos hacia el proveedor. Los vendedores en línea son capaces de recopilar la interacción de los clientes, cada enlace visitado, no únicamente la venta final. Aquellos que son capaces de hacer uso rápidamente de la información para así recomendar compras adicionales, obtienen una ventaja competitiva. El aumento de los teléfonos inteligentes incrementa la tasa de entrada de datos, ya que los consumidores llevan con ellos una fuente de transmisión de imágenes geolocalizadas y datos de audio.

Esta característica no se limita únicamente a la velocidad de recepción de los datos, sino a la velocidad de salida, es decir, con la que se procesa y analiza esta información que está en constante movimiento, lo cual producirá mayor ventaja competitiva, por ejemplo las recomendaciones de Facebook. En la terminología de la industria suelen llamar a estos datos en rápido movimiento, Streaming Data (Flujo de datos). Existen dos razones importantes para considerar el procesamiento del flujo de datos. La primera razón es cuando los datos producidos son demasiado rápidos para almacenarlos en su totalidad, por ejemplo, el Gran Colisionador de Hadrones del CERN genera tantos datos que los científicos descartan la gran mayoría de estos, esperando que ninguno de los datos perdidos haya sido información útil. La segunda razón para considerar son las aplicaciones que demandan respuestas inmediatas de datos, por ejemplo, aplicaciones móviles y juegos en línea.

Los programas existentes para el manejo de flujo de datos se dividen en propietarios como InfoSphereStreams de IBM y los frameworks de código abierto como Storm de Twitter y S4 de Yahoo.

La necesidad de la velocidad, sobre todo en la web, ha impulsado el crecimiento de las bases de datos clave/valor y en columnas, las cuales proveen una recuperación rápida de la información.

4.3.3 Variedad. Raramente los datos se presentan de una manera ordenada y lista para su procesamiento. Los datos producidos son diversos y no encajan en estructuras relacionales. Esta información podría ser texto de las redes sociales, imágenes, datos recibidos directamente de un sensor.

Incluso en la web, donde se creería que la comunicación entre máquinas debería brindar ciertas garantías, los datos generados por estos son desordenados. Los distintos navegadores envían diferentes datos, ya que pueden estar utilizando diferentes versiones de software o proveedores para comunicarse. Aún cuando el proceso es generado por un ser humano, podrá existir error e incoherencia.

Un uso común del procesamiento de Big Data es tomar los datos no estructurados y extraerlos de manera que encajen en un esquema ordenado, ya sea para la utilización por parte de los seres humanos o como una entrada estructurada a una aplicación. Este proceso implica pérdida de información, ya que al tratar de encajar en una estructura, se desechan muchos datos. Cabe subrayar un principio del Big Data, el cual indica que cuando se tenga la posibilidad, se almacene toda la información, ya que puede haber información importante en los bits desechados, y después de perder estos bits, los datos no se pueden recuperar.

Los datos que se procesan en la actualidad se clasifican en tres tipos:

- **Estructurados:** Este término se refiere generalmente a los datos que tienen definido un tamaño y un formato. Algunos ejemplos de datos estructurados incluyen números, cadenas de caracteres, fechas, entre otros. Muchos expertos afirman que este tipo de datos representan el 20% de la información total.

  La evolución de la tecnología provee nuevas fuentes que producen datos estructurados:

  - **Generados por computadores/máquinas:** Datos que son creados sin intervención humana.

    - **Sensores:** Algunos ejemplos son los medidores inteligentes, identificación por radiofrecuencia (RFID) y el sistema de posicionamiento global GPS.
    - **Datos de los registros web:** Captura todos los tipos de datos de las actividades realizadas en servidores y aplicaciones.
    - **Puntos de venta:** Por ejemplo, al pasar el código de barras de algún producto por el lector, se generan todos los datos relacionados con ese producto.
    - **Datos financieros:** Muchos de los servicios financieros son programados, estos operan sobre reglas predefinidas que automatizan los procesos.

- **Generados por interacción humana:** Datos que los seres humanos crean a partir de la interacción con el computador. Algunos de estos datos estructurados pueden ser:

    - **Datos de entrada:** Cualquier información que el usuario ingrese como nombre, apellidos, edad, entre otros. Todos estos datos pueden ser útiles para entender el comportamiento básico de los clientes.
    - **Datos de los sitios web visitados:** Al analizar todos los datos generados al visitar sitios web es posible determinar el comportamiento de los usuarios y construir patrones.
    - **Datos relacionados con juegos:** Cada movimiento realizado en un juego es posible guardarlo. Esto resulta ser muy útil para entender como los usuarios se mueven dentro de un portafolio de juegos.

o **Semiestructurados:** Este tipo de datos no se ajusta necesariamente a una estructura fija. Por ejemplo, el intercambio electrónico de datos (EDI), XML y SWIFT.

o **No estructurados:** Los datos no estructurados no siguen un formato específico. Este tipo de datos es el más encontrado en la actualidad. Hasta hace muy poco, la tecnología no soportaba este tipo de datos, solo permitía su almacenamiento y el análisis se realizaba de manera manual.

- **Generados por computadores/máquinas:** Algunos ejemplos son:

    - **Imágenes satelitales:** Incluye datos del clima e imágenes capturadas por el gobierno a través de la vigilancia por satélite.
    - **Datos científicos:** Por ejemplo, imágenes sísmicas y datos atmosféricos.

        - **Fotografías y vídeos:** Resultantes de cámaras de vigilancia y vídeos de tráfico.
        - **Radares:** Datos provenientes de radares vehiculares, meteorológicos y oceanográficos.

- **Generados por la interacción humano-máquina:** Estos son algunos de ellos:
    - **Información interna de la empresa:** Es decir, documentos, registros, encuestas y correos electrónicos.
    - **Datos de las redes sociales:** Estos datos son generados a partir de las plataformas como Facebook, Youtube, Twitter, LinkedIn, Flickr, entre otros.
    - **Datos móviles:** Incluye mensajes de texto e información de ubicación.
    - **Contenido de sitios web:** Estos datos son provenientes de sitios de contenido no estructurado, como Youtube, Flickr o Instagram.

A pesar de la popularidad de las bases de datos relacionales, en muchos casos no son utilizadas debido a que necesariamente se debe seguir un esquema, ni siquiera aún cuando se intenta que los datos encajen en la estructura, ya que de esta manera se pierde mucha información. Las bases de datos NoSQL semiestructuradas satisfacen la necesidad de flexibilidad, pues no hace uso de un esquema exacto, sino que proporciona una estructura para organizar los datos. Existen bases de datos que encajan mejor para ciertos tipos de datos, por ejemplo, las relaciones de las redes sociales son grafos por naturaleza y se manejan de una manera más simple y eficiente a través de las bases de datos en grafo.

Las principales características del Big Data son las mencionadas anteriormente, aunque en algunos libros agregan el valor como otra característica importante, ya que al realizar un análisis a los datos almacenados se podrá deducir resultados importantes que ayudarán a la toma de decisiones. Este análisis de la información también ayuda a encontrar tendencias de negocio a partir de las cuales se podrán modificar las estrategias de negocio, además, de mejorar el rendimiento, encontrar nuevas fuentes de segmentación de clientes, entre otras.

4.4 BENEFICIOS DEL BIG DATA

El instituto global McKinsey clasificó los beneficios producidos por el Big Data en cinco categorías principales[24]:

- **Salud:**Sistemas de apoyo a las decisiones clínicas, análisis individuales aplicados a los perfiles de los pacientes, medicina personalizada, precios basados en el desempeño del personal, análisis de patrones de enfermedades y mejorar la salud pública.
- **Sector público:** Creación de transparencia a los datos relacionados accesibles, descubrir necesidades, mejorar el rendimiento, personalizar las acciones y los productos de manera adecuada, toma de decisiones con sistemas automatizados para disminuir los riesgos e innovar nuevos productos y servicios.
- **Comercio al por menor:** Análisis de comportamiento, variedad y optimización de precios en las tiendas, diseño de la publicidad por emplazamiento, mejorar el rendimiento, optimización de los insumos de trabajo, optimización de la distribución y la logística y mercados basados en la web.

[24]J. Manyika, M. Chui, B. Brown, J. Bughin, R. Dobbs, C. Roxburgh and A.H. Byers, "Big data: The next frontier for innovation, competition, and productivity", McKinsey Global Institute, 2011,

- **Fabricación:** Mejorar la predicción de la demanda, planificar la cadena de suministro, soporte de ventas y aplicaciones web basadas en búsquedas.
- **Datos personales de ubicación:** Enrutamiento inteligente, geopublicidad dirigida, planificación urbana y nuevos modelos de negocios.

4.5 DESAFÍOS DEL BIG DATA

El rápido crecimiento de los datos en la era del Big Data trae consigo enormes desafíos tanto en la adquisición de los datos como en el almacenamiento, gestión y análisis. Los sistemas tradicionales de gestión y análisis de información se basan en los sistemas gestores de bases de datos relacionales (RDBMS), los cuales aplican únicamente a los datos estructurados. Además, estos están utilizando hardware cada vez más caro debido a que no pueden manejar grandes volúmenes de datos y su heterogeneidad. Se han propuesto diversas soluciones desde diferentes perspectivas, por ejemplo, la computación en la nube se utiliza para cumplir con los requisitos de infraestructura paragrandes volúmenes de información, por ejemplo, elasticidad y eficiencia de los costos. Para las soluciones de almacenamiento y gestión de datos desordenados a gran escala, surgieron los sistemas de archivos distribuidos y las bases de datos NoSQL.

Algunos autores hablan acerca de los obstáculos que hay que superar en el desarrollo de las aplicaciones de Big Data. Algunos de los retos claves son[25]:

- **Representación de los datos:** Muchos conjuntos de datos tienen ciertos niveles de heterogeneidad en el tipo, la estructura, la semántica, la organización, la granularidad y la accesibilidad. La representación de los datos tiene como objetivo hacer los datos más significativos para el análisis de la computadora y la interpretación del usuario. Una representación inadecuada de los datos reducirá el valor de los datos originales e incluso puede llegar a

[25]LABRINIDIS, Alexandros y JAGADISH, HV. Challenges and opportunities with big data.*Proceedings of the VLDB Endowment*, Págs. 2032–2033, 2012.

obstruir el análisis eficaz de los datos. En cambio, una representación de datos eficiente reflejará la estructura de datos, clase, tipo, así como tecnologías integradas, con el objetivo de permitir operaciones eficaces en diferentes bases de datos.

- **Reducción de la redundancia y la compresión de los datos:** En general, hay un alto nivel de redundancia en los conjuntos de datos. La reducción de la redundancia y la compresión de los datos es eficaz para reducir el costo indirecto de todo el sistema en la premisa de que los valores potenciales de los datos no se ven afectados. Por ejemplo, la mayoría de los datos generados por las redes de sensores son altamente redundantes, que pueden ser filtrados y comprimidos.
- **Gestión del ciclo de vida de los datos:** En comparación con el avance relativamente lento de los sistemas de almacenamiento, sensores y de la computación, los datos están creciendo a velocidades y escalas sin precedentes.Uno de los desafíos a los que se enfrenta el Big Data, es que los sistemas de almacenamiento actuales no soportan estos datos masivos. Lo que hacía necesario determinar un valor analítico para seleccionar que datos serían almacenados y cuáles serían desechados.
- **Sistema analítico:** El sistema analítico del Big Data deberá procesar grandes cantidades de datos heterogéneos en un tiempo limitado. Los sistemas de gestión de bases de datos relacionales RDBMS no cumplen con los requisitos de rendimiento, ya que poseen un diseño estricto al cual le falta escalabilidad y capacidad de ampliación. Las bases de datos no relacionales han demostrado ventajas en el procesamiento de los datos no estructurados, convirtiéndose así en las más usadas en el análisis del Big Data. Algunas empresas han utilizado una arquitectura de base de datos mixta que integra las ventajas de ambos tipos, un ejemplo de dichas organizaciones es Facebook.
- **Confidencialidad de los datos:** En la actualidad, los proveedores de servicios o propietarios de la información no pueden mantener de forma efectiva estos enormes datos ni analizarlos debido a su capacidad limitada. Estos deben

confiar en profesionales o herramientas para analizar los datos, aumentando así los riesgos potenciales de seguridad, pues contienen detalles de la granularidad e información sensible como números de tarjetas de crédito. Por lo tanto, el análisis de grandes volúmenes de datos solo debe ser entregado a terceros cuando se han tenido en cuenta las medidas preventivas adecuadas para proteger los datos sensibles y así garantizar la seguridad.

- **Control de energía:** Con el aumento del volumen de los datos y la demanda de análisis de los mismos, la capacidad de procesamiento, almacenamiento y la transmisión de los datos masivos se empieza a consumir mucha más energía eléctrica. Por lo tanto, se deben establecer controles al consumo de energía permitiendo que la capacidad de expansión y la accesibilidad sea garantizada.
- **Capacidad de expansión y escalabilidad:** El sistema de análisis de datos masivos debe ser compatible con los conjuntos de datos actuales y futuros. El algoritmo de análisis debe ser capaz de procesar conjuntos de datos más complejos y los cuales están en constante crecimiento.
- **Cooperación:** El análisis del Big Data es una investigación interdisciplinaria, que requiere expertos en los diferentes campos con el fin de aprovechar todo el potencial de los datos masivos. Una arquitectura de la red del Big Data integral debe ser establecida para ayudar a los científicos e ingenieros en diversas áreas, permitiendo acceder a diferentes tipos de datos y utilizar completamente su experiencia, con el fin de cooperar para completar los objetivos analíticos.

4.6 TECNOLOGÍAS BIG DATA

Algunas de las tecnologías más nombradas que suplen con las características del Big Data son las mencionadas a continuación.

4.6.1 Computación en la nube. El poder de la nube es que los usuarios pueden acceder a recursos informáticos y de almacenamiento con poco o ningún soporte de TI y sin la necesidad de comprar más hardware o software. Una de las principales características de la nube es la escalabilidad elástica, es decir, los usuarios pueden agregar o retirar recursos según los cambios en los requerimientos casi en tiempo real. La nube juega un papel importante dentro del mundo del Big Data. Los cambios drásticos ocurren cuando los componentes de la infraestructura se combinan con los avances en la gestión de los datos. La escalabilidad horizontal soporta la implementación del Big Data.

La computación en la nube proporciona un conjunto de recursos informáticos compartidos que incluyen aplicaciones, almacenamiento, capacidad de procesamiento y redes, así como los procesos de negocio. Ejemplos muy conocidos de los beneficios de la nube en el soporte de Big Data son Google y Amazon. Ambas compañías dependen de la capacidad de manejar grandes cantidades de datos para la continuidad del negocio. Estos proveedores necesitan infraestructuras y tecnologías que puedan soportar aplicaciones a gran escala. Por ejemplo, los millones y millones de mensajes que Google procesa por día como parte de su servicio. Google ha sido capaz de optimizar el sistema operativo Linux y su entorno de software para dar soporte al correo electrónico de la manera más eficiente. Por lo tanto, puede admitir fácilmente cientos de millones de usuarios. Aún más importante, Google es capaz de captar y aprovechar las enormes cantidades de datos de los usuarios de correo y del motor de búsqueda.

Amazon.com, con sus centros de datos IaaS, esta optimizado para soportar estas cargas de trabajo de tal manera que pueda seguir ofreciendo nuevos servicios y dar soporte a un número creciente de clientes. Para hacer crecer su negocio, Amazon debe ser capaz de gestionar datos sobre su mercancía, sus compradores y su canal de comerciantes asociados. La publicidad dirigida basada en patrones de compra de los clientes es fundamental para el éxito de la compañía.

4.6.1.1 Modelos de implementación. Dos modelos clave en la nube son importantes en el almacenamiento de los grandes datos, nubes públicas y nubes privadas. Algunos factores que tienen en cuenta las empresas para equilibrar los proveedores públicos y privados dependen de una serie de cuestiones como la privacidad, latencia y propósito.

- **NUBE PÚBLICA:** La nube pública es un conjunto de hardware, redes, almacenamiento, servicios, aplicaciones operado por un tercero para su uso por otras empresas y particulares. Estos proveedores comerciales crean un centro de datos altamente escalable que oculta los detalles de la infraestructura del consumidor. Las nubes públicas son viables, ya que suelen gestionar cargas de trabajo relativamente repetitivas o sencillas. Por ejemplo, el correo electrónico es una aplicación muy sencilla. Por lo tanto, un proveedor de la nube puede optimizar el entorno de modo que es el más adecuado para soportar un gran número de clientes, incluso si guarda muchos mensajes.

  Un típico centro de datos soporta tantas aplicaciones diferentes y cargas de trabajo que no se pueden optimizar fácilmente. Una nube pública puede ser muy eficaz cuando una organización está ejecutando un proyecto de análisis de datos muy complejo y además necesita de ciclos de computación adicionales para manejar la tarea. Las empresas pueden optar por almacenar datos en una nube pública, donde el costo por gigabyte es relativamente barato

en comparación con el almacenamiento adquirido. Las cuestiones primordiales con las nubes públicas para grandes volúmenes de datos son los requisitos de seguridad y la cantidad de latencia aceptable. La elección dependerá de la naturaleza de sus proyectos Big Data y la cantidad de riesgo que puede asumir.

- **NUBE PRIVADA:** Una nube privada proporciona los mismos servicios pero difieren en que es operado por una organización para el uso de sus empleados, socios y clientes. Una nube privada puede ser creada y gestionada por terceros para el uso de exclusivo de una empresa. La nube privada es un entorno altamente controlado no abierto para el consumo público. Por lo tanto, se encuentra detrás de un firewall. La nube privada podría ser la mejor elección ya que está enfocada en la seguridad y el cumplimiento.

Una nube híbrida es una combinación de una nube privada combinada con el uso de servicios de la nube pública. El objetivo es crear un entorno unificado, automatizado y bien administrado que pueda combinar servicios y datos de una variedad de modelos de nube.

4.6.1.2 Modelos de prestación.

- **INFRAESTRUCTURA COMO SERVICIO (IAAS):** IaaS es la prestación de servicios informáticos, incluyendo hardware, redes, almacenamiento y espacio del centro de datos basado en un modelo de arrendamiento. El consumidor del servicio adquiere un recurso y se cobra según la cantidad utilizada y la duración de dicho uso. Existen versiones IaaS públicas y privadas. En el IaaS público, el usuario utiliza una tarjeta de crédito para la adquisición de estos recursos, cuando el usuario deja de pagar el recurso desaparece. Por lo general en un servicio privado de IaaS, la organización crea la infraestructura

para proporcionar recursos según la demanda de los usuarios internos y a veces socios de negocios.

- **PLATAFORMA COMO SERVICIO (PAAS):** PaaS es un mecanismo para combinar IaaS con un conjunto abstracto de servicios middleware, desarrollo de software y herramientas de implementación que permitan a la organización crear e implementar aplicaciones en la nube. Un PaaS ofrece servicios para las diferentes fases del ciclo de desarrollo y pruebas de software. Se le ofrece la plataforma de desarrollo y las herramientas de programación por lo que puede desarrollar aplicaciones propias y controlar la aplicación, pero no controla la infraestructura[26].

- **SOFTWARE COMO SERVICIO (SAAS):** SaaS es una aplicación de negocio creada y organizada por un proveedor en un modelo multiusuario, es decir, una sola instancia de la aplicación se ejecuta en un entorno de nube, pero sirve para múltiples organizaciones, manteniendo todos sus datos separados. En algunas ocasiones, los usuarios poseen control sobre estas aplicaciones, modificándolas según sus necesidades. Se elimina la necesidad de instalarlas en los computadores de los clientes, evitando así costos de mantenimiento. Los clientes pagan por el servicio, de manera mensual o anual.

- **DATOS COMO SERVICIO (DAAS):** Teniendo en cuenta que se está hablando de la computación en la nube con Big Data, se debe conocer además los datos como servicio. DaaS está estrechamente relacionado con SaaS. DaaS es un servicio independiente de la plataforma que le permitirá conectarse a la nube para almacenar y recuperar sus datos. Además, encontrará una serie de servicios especializados de datos que son de gran beneficio en el entorno Big Data. Por ejemplo, Google ofrece un servicio que puede procesar una consulta con 5 terabytes de datos en tan sólo 15 segundos. Cientos de servicios

[26] Plataforma como servicio [En Línea]. Disponible en: <http://es.wikipedia.org/wiki/Computaci%C3%B3n_en_la_nube#Plataforma_como_servicio>

analíticos especializados han sido desarrollados por empresas como IBM y otros.

4.6.1.3 Características de la nube. Las características de la nube que suplen con las necesidades del ecosistema del Big Data son:

- **Escalabilidad:** En cuanto a hardware se refiere a la capacidad de ir desde pequeñas a grandes cantidades de potencia de procesamiento con la misma arquitectura. La nube puede escalar a grandes volúmenes de datos. La computación distribuida trabaja bajo la premisa de “Divide y vencerás”, de esta manera si se tiene un gran volumen de datos, este puede ser repartido a través de servidores de la nube. Una característica importante de IaaS es que se puede escalar de forma dinámica, es decir, si se requieren más recursos de lo esperado, se pueden obtener.
- **Elasticidad:** Capacidad para ampliar o reducir la demanda de recursos en tiempo real, basado en la necesidad. Uno de los beneficios de la nube es que os clientes tienen la posibilidad de acceder a una mayor cantidad de un servicio cuando se requiera para hacer frente al volumen y la velocidad de los datos.
- **Agrupación de recursos:** La arquitectura de la computación en la nube permite la creación eficiente de grupos de recursos compartidos que forman una nube económicamente viable.
- **Autoservicio:** El usuario a través de un navegador o una interfaz adquiere los recursos necesarios para su proyecto.
- **Costos reducidos:** A menudo los costos de la nube pueden ser reducidos ya que no se están comprando enormes cantidades de hardware o arrendando un nuevo espacio para hacer frente al Big Data.

- **Pago por consumo:** Cuando no se tiene clara la cantidad de recursos que se necesita para llevar a cabo el proyecto, se factura según los recursos utilizados.
- **Tolerancia a fallos:** Los proveedores de servicios de nube deben tener la tolerancia a fallos integrada en su arquitectura, se debe prestar un servicio ininterrumpido a pesar de que fallen uno o más componentes del sistema.

4.6.1.4 Usos de la nube para Big Data. La naturaleza de la nube lo convierte en un entorno de computación ideal para grandes volúmenes de datos. Algunos ejemplos del uso de la nube para Big Data:

- **IaaS en una nube pública:** Utilizar la infraestructura de un proveedor de nube pública para los servicios del Big Data, ya que no se desea utilizar la infraestructura física propia. IaaS puede proporcionar la creación de máquinas virtuales con almacenamiento y poder de computo casi ilimitado. Se puede elegir el sistema operativo que se desea, además posee la capacidad de crecer dinámicamente según las necesidades cambiantes. Por ejemplo, hacer uso del servicio Amazon Elastic Compute Cloud (Amazon EC2) para ejecutar un modelo predictivo en tiempo real que requiere procesar los datos utilizando procesamiento masivo en paralelo. Puede ser un servicio que procesa la información de las ventas de grandes superficies. Es posible que desee procesar miles de millones de secuencias de clics para orientar a los clientes con el anuncio correcto en tiempo real.
- **PaaS en una nube privada:** PaaS es una infraestructura que se puede utilizar para diseñar, implementar y desarrollar aplicaciones y servicios. Los proveedores de PaaS están empezando a incorporar dentro de su portafolio, tecnologías para Big Data, como Hadoop y MapReduce. Por ejemplo, es posible que desee construir una aplicación especializada para analizar grandes cantidades de datos médicos. La aplicación debería hacer uso tanto de los

datos en tiempo real como de los que no. Para ellos, se va a requerir Hadoop MapReduce para el almacenamiento y el procesamiento.

La ventaja de utilizar PaaS en este escenario es la rapidez con la que se puede implementar la aplicación.

- **SaaS en una nube híbrida:** Considerando que desea analizar la opinión del clientes desde múltiples canales. Muchas empresas se han dado cuenta de que una de las fuentes de datos más importantes es lo que el cliente piensa y dice acerca de su empresa, productos y servicios. Obtener acceso a este tipo de información proporciona información muy valiosa sobre los comportamientos y las acciones. Cada vez más, los clientes están dando su punto de vista en sitios públicos a través de internet. El proveedor de SaaS proporciona la plataforma para el análisis, así como los datos de los medios sociales. Además, es posible hacer uso de los datos del CRM de la empresa en su entorno de nube privada para su inclusión en el análisis.

4.6.2 Hadoop. Los motores de búsqueda como Yahoo y Google necesitaban encontrar una manera de generar valor con las enormes cantidades de datos que sus motores estaban recogiendo. Estas empresas necesitaban analizar la información recolectada y determinar cómo se podrían obtener beneficios económicos a partir de estos datos y así soportar su modelo de negocio. Hadoop fue desarrollado debido a la necesidad de las empresas de gestionar estos grandes volúmenes de datos de manera sencilla. Hadoop permite dividir los grandes problemas en elementos más pequeños que pueden ser procesados en paralelo y al final reagrupar las piezas pequeñas para presentar los resultados, lo cual permite que el análisis se pueda realizar de forma rápida y rentable.

Hadoop fue originalmente construido por los ingenieros Doug Cutting y Mike Cafarella empleados de Yahoo. En el 2006, se convirtió en un proyecto de código abierto manejado por Apache Software Foundation, el cual es ampliamente

implementado por Yahoo, Facebook y otras empresas de internet. Hadoop está diseñado para paralelizar el procesamiento de los datos a través de los nodos para acelerar los cálculos y ocultar la latencia. Hadoop es ampliamente utilizado en aplicaciones Big Data, por ejemplo, el filtrado del correo no deseado, búsqueda de redes, análisis del flujo de clics y la recomendación social. En la actualidad, el más grande clúster Hadoop es operado por Yahoo que cuenta con 4000 nodos utilizados para el procesamiento y análisis de datos, incluyendo anuncios, datos financieros y registros de usuarios.

Hadoop está diseñado para procesar grandes cantidades de datos estructurados y no estructurados, es implementado en racks de servidores genéricos como un clúster Hadoop. Los servidores se pueden agregar o quitar dinámicamente del clúster ya que Hadoop es auto-recuperación, es decir, es capaz de detectar los cambios, incluido los daños y ajustarse a ellos y continuar funcionando sin interrupción.

Hadoop posee muchas ventajas, pero los siguientes aspectos son relevantes en el manejo y análisis del Big Data:

- **Capacidad de expansión:** Hadoop permite el aumento o disminución de la infraestructura del hardware sin necesidad de cambiar los formatos de datos. El sistema automáticamente volverá a distribuir los datos y las tareas de computación se adaptarán a los cambios del hardware.
- **Eficiencia a bajo costo:** Hadoop aplica computación paralela a gran escala en servidores comerciales, lo que reduce en gran medida el costo por Terabyte requerido en la capacidad de almacenamiento requerida. La computación a gran escala también permite adaptar el volumen de datos en continuo crecimiento.
- **Flexibilidad:** Hadoop puede manejar muchos tipos de datos de diversas fuentes. Además, estos datos pueden ser sintetizados en Hadoop para su

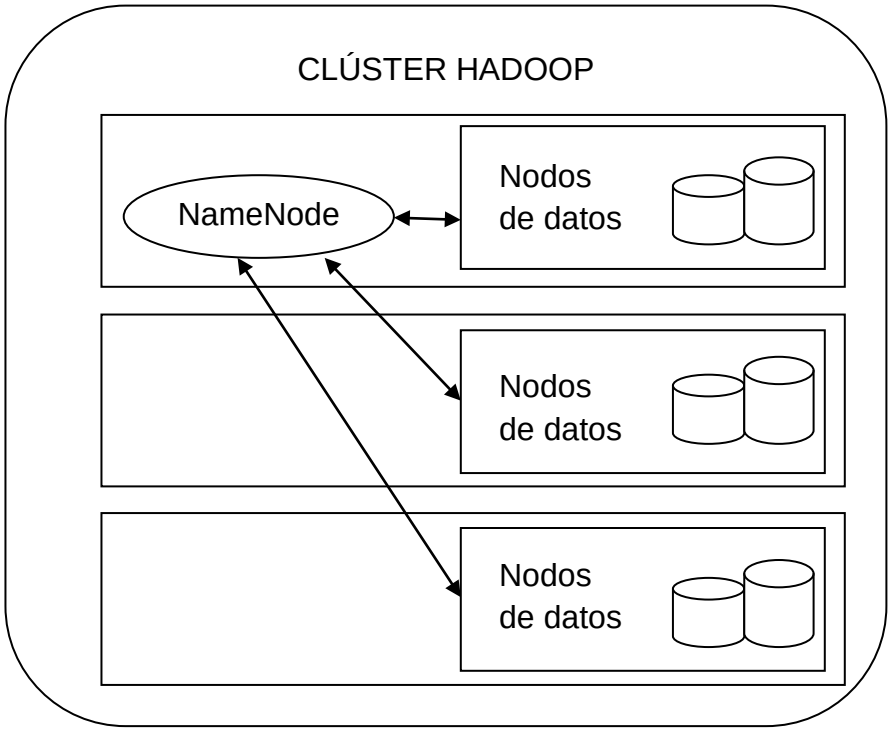
posterior análisis. Por lo tanto, puede afrontar muchos de los retos que trae el Big Data.

- **Alta tolerancia a fallos:** Es común que al momento de analizar grandes volúmenes de datos se presenten pérdidas de datos y errores de cálculo, pero Hadoop puede recuperar datos causados por fallos en los nodos o congestión de la red.

4.6.2.1 Componentes Hadoop. Hadoop contiene dos componentes principales.

- **Sistema de archivos distribuidos Hadoop (HDFS).**El sistema de archivos distribuidos Hadoop es un clúster de almacenamiento de datos resistente, versátil, confiable, de alto ancho de banda y bajo costo que permite la gestión de grandes cantidades de archivos. HDFS no es el destino final de los archivos. Es decir, es un servicio de datos que ofrece un conjunto único de capacidades necesarias cuando se presentan grandes volúmenes de datos y altas velocidades. Distribuye los archivos en bloques de 128 MB o 256 MB y almacena dichos bloques de datos en diferentes nodos del clúster, a fin de permitir la computación paralela. Se proporciona unos niveles más altos de rendimientos cuando todo el clúster se encuentra ubicado en el mismo rack. HDFS incluye un “NameNode” para la gestión de los metadatos del sistema de archivos y nodos de datos para almacenar los datos. En esencia, el NameNode realiza un seguimiento de dónde se almacenan físicamente los datos. Las copias de los bloques se distribuyen a diferentes nodos para evitar la pérdida de datos. La siguiente figura muestra la arquitectura básica de HDFS.

Ilustración 8: Clúster Hadoop



A continuación se explicarán los elementos de esta arquitectura:

- **NameNode:** HDFS trabaja dividiendo archivos de gran tamaño en partes más pequeñas llamadas bloques. Los bloques se almacenan en los nodos de datos y es responsabilidad del NameNode saber la ubicación de los bloques que conforman el archivo completo. Además, maneja todos los accesos a los archivos, incluyendo lectura, escritura, creación, eliminación y replicación de los datos en los nodos. A la colección completa de todos los archivos en el clúster, en ocasiones, se le denomina *namespace*. Es el trabajo del NameNode gestionar este *namespace*.

  A pesar de que existe una fuerte relación entre el NameNode y los nodos de datos, estos se encuentran débilmente acoplados, lo que permite que los elementos del clúster puedan comportarse de manera dinámica, añadiendo o eliminando servidores a medida que la demanda aumente o disminuya. En una

configuración típica, es posible que encuentre el NameNode y un nodo de datos ejecutándose en un servidor del rack. Otros servidores sólo funcionan como nodos de datos.

Los nodos de datos constantemente le preguntan al NameNode si hay algo que ellos deban hacer. Este comportamiento permite al NameNode determinar que nodos están fuera de servicio y cuales están muy ocupados. Los nodos de datos también se comunican entre sí, de modo que puedan cooperar durante las operaciones normales del sistema de archivos. Esta comunicación es necesaria ya que los bloques de un archivo son almacenados en diferentes nodos de datos, para proteger los archivos de un solo punto de fallo, evitar las perdidas y permitir el correcto funcionamiento del clúster.

- **Nodos de datos:** Los nodos de datos no son inteligentes, pero son resistentes. Dentro del clúster HDFS, los bloques se replican a través de múltiples nodos de datos y el acceso es administrado por el NameNode. El mecanismo de replicación está diseñado para un rendimiento óptimo cuando todos los nodos del clúster se encuentran en un rack. De hecho, el NameNode utiliza un identificador de rack para realizar el seguimiento a los nodos de datos del clúster. Los nodos de datos también proporcionan mensajes “HeartBeat”, los cuales detectan y aseguran la conectividad entre el NameNode y los nodos de datos. Cuando estos mensajes dejan de producirse, el NameNode elimina el nodo de datos de su mapa y sigue funcionando como si nada hubiera pasado. Cuando el NameNode recibe un nuevo mensaje “HeartBeat”, se añade al clúster transparentemente.

  Como con todos los archivos, la integridad de los datos es un factor clave. HDFS apoya una serie de funciones diseñadas para proporcionar la integridad de los datos. Al momento de dividir los archivos en partes pequeñas es posible que se afecten la integridad, es por eso que HDFS utiliza registros de

transacciones y validación de suma de verificación para garantizar la integridad de todo el clúster.

Los registros de transacciones son una práctica muy común en el sistema de archivos y el diseño de las bases de datos, ya que realizan un seguimiento de todas las operaciones y son eficaces en las auditorías o la reconstrucción de los sistemas de archivos cuando se presenta algún fallo.

La validación a través de la suma de verificación se utiliza para garantizar el contenido de los archivos en HDFS. Cuando un cliente solicita un archivo, se puede verificar el contenido mediante la suma de verificación. Si la suma no coincide, se informa el error, de lo contrario la operación se realiza normalmente. Los archivos de suma de verificación están ocultos para evitar la manipulación indebida de los mismos.

Los nodos de datos utilizan los discos locales en los servidores para la persistencia. Todos los bloques de datos se almacenan localmente, principalmente por razones de rendimiento. Los bloques de datos son replicados en varios nodos de datos, lo que garantiza que si un servidor fallael archivo no se verá afectado. El grado de replicación, el número de nodos de datos y el *namespace* HDFS son establecidos cuando el clúster es implementado. Debido a que el HDFS es dinámico, todos los parámetros pueden ser modificados durante el funcionamiento del clúster.

- **MapReduce:** Hadoop MapReduce es una implementación del algoritmo desarrollado por el proyecto Hadoop de Apache. Está compuesto por dos etapas de procesamiento:

  - **Map:** En contraste con las tradicionales bases de datos relacionales, en las cuales los datos son organizados de una manera rígida en filas y columnas y almacenados en tablas. MapReduce usa pares clave/valor. En primer

lugar, la clave identifica el tipo de información que se está viendo. Cuando se compara con una base de datos relacional, una clave es equivalente a una columna.El valor es una instancia de los datos asociados con la clave.

En la fase Map del MapReduce los registros de la fuente de datos se introducen en la función Map() como pares clave/valor. La función map() produce uno o más valores intermedios junto con una clave de salida desde los datos ingresados.

- **Reduce:** Después de que la fase Map ha finalizado, todos los valores intermedios para una clave de salida se reúnen en una lista. La función Reduce() combina los valores intermedios entre uno o más valores finales para la misma clave.

  Este es un enfoque mucho más simple de la computación a gran escala, además abstrae gran parte de la complejidad del procesamiento en paralelo. Sin embargo, a pesar de su simplicidad, MapReduce permite procesar cantidades masivas de información mucho más rápido.

4.6.3 Bases de datos no relacionales. En la actualidad, las bases de datos relacionales son las más usadas ya que las empresas pequeñas y grandes almacenan su información en estas, pero a la hora de hablar del Big Data, las bases de datos no relacionales ofrecen algunas soluciones para los desafíos que trae consigo estas enormes cantidades de información. Como se explicó en el capítulo anterior, las bases de datos NoSQL no se basan en el modelo relacional, cada uno de los tipos de estas bases de datos tienen su propio conjunto de características centradas en los problemas específicos que no resuelven las bases de datos relacionales. Los datos manejados en el Big Data requieren una persistencia especial y unas técnicas diferentes de manipulación de datos.

A continuación, se dará a conocer las bases de datos no relacionales más usadas actualmente.

4.6.3.1 Base de datos clave/valor Riak. Una de las bases de datos clave/valor de código abierto más utilizada se llama Riak. Fue desarrollada por una compañía llamada Basho Technologies y está disponible bajo la licencia de software Apache versión 2.0.

Riak es una implementación muy rápida y escalable de las bases de datos clave/valor. Es compatible con entornos de grandes volúmenes de datos que cambian rápidamente. Riak es particularmente eficaz para el análisis en tiempo real de los servicios financieros. Utiliza “buckets” como un mecanismo para organizar las colecciones de claves y valores. Las implementaciones Risk son clústeres de nodos físicos o virtuales en formato peer-to-peer. No poseen un nodo principal, por lo que el clúster es resistente y altamente escalable. Todos los datos y las operaciones se distribuyen en todo el clúster. Los clústeres Riak tienen un rendimiento interesante, ya que entre más grande es el clúster, es decir posee más nodos, son mejores y rápidos que los clústeres con menos nodos. La comunicación en el clúster se implementa a través de un protocolo denominado “Gossip”, el cual almacena información del estado del clúster y comparte información de los “buckets”.

Las principales características de Risk son:

- **Procesamiento en paralelo:** Haciendo uso de MapReduce, Riak posee la capacidad de descomponer y recomponer consultas a través del clúster para el análisis en tiempo real.
- **Enlaces:** Riak puede ser construido para imitar una base de datos basada en grafos usando enlaces. Un enlace puede ser pensado como una conexión

unidireccional entre pares clave/valor. Los enlaces proporcionarán un mapa de relaciones entre pares clave/valor.

- **Búsqueda:** La búsqueda Riak tiene una capacidad de búsqueda de texto completo, la cual es distribuida y soporta la tolerancia a fallos. Los “buckets” pueden ser indexados para una hallar una solución rápida de los valores a las claves.
- **Índices secundarios:** Los desarrolladores pueden etiquetar los valores con una o más claves. La aplicación puede consultar el índice y devolver una lista de claves coincidentes. Esto puede ser muy útil en implementaciones Big Data ya que la operación es atómica y soportará comportamientos en tiempo real.

Las implementaciones Riak son las más adecuadas para las siguientes situaciones:

- Los datos de usuario de las redes sociales, comunidades o juegos.
- Grandes volúmenes, recopilación y almacenamiento de datos.
- Aplicaciones móviles que requieren flexibilidad y fiabilidad.

4.6.3.2 Base de datos documental MongoDB. MongoDB es el nombre del proyecto “Hu(mongo)us database”, bases de datos enormes. Es mantenido por una compañía llamada 10gen como código abierto. MongoDB está creciendo en popularidad y puede ser una buena opción a la hora de almacenar los datos de las aplicaciones Big Data. MongoDB se compone de bases de datos que contienen colecciones. Una colección está compuesta de documentos y cada documento está compuesto de campos. Al igual que las bases de datos relacionales, es posible indexar una colección, ya que de esta manera se aumentaría el rendimiento de la búsqueda de datos. A diferencia de otras bases de datos, MongoDB devuelve un cursor, que sirve como puntero a los datos. Esta es una capacidad muy útil, ya que ofrece la opción de contar o clasificar los datos sin

extraerlos. Nativamente, MongoDB soporta BSON, implementación binaria de documentos JSON.

MongoDB consta de los siguientes elementos:

- Servicios de alta disponibilidad y replicación para la ampliación a través de redes locales y de área amplia.
- Un sistema de archivos basado en grid (GridFS) lo que permite el almacenamiento de grandes objetos dividiéndolos entre varios documentos.
- Utiliza MapReduce para apoyar el análisis y la agregación de diferentes colecciones/documentos.
- Un servicio de fragmentación que distribuye una sola base de datos a través de un clúster de servidores en uno o más centros de datos.
- Un servicio de consultas que admite consultas ad hoc, consultas distribuidas y búsqueda de texto completo.

Las implementaciones en las que MongoDB actúa de manera de eficaz son:

- Gran volumen de gestión de contenidos
- Redes sociales
- Archivos
- Análisis en tiempo real

4.6.3.3 Base de datos documental COUCHDB. Otra base de datos no relacional muy popular es CouchDB. Así como MongoDB, CouchDB es de código abierto. Es mantenido por la Fundación de Software Apache y está disponible bajo la licencia Apache v2.0. A diferencia de MongoDB, CouchDB fue diseñado para asemejarse a la web en todos los aspectos. Por ejemplo, CouchDB es resistente a los abandonos de red y seguirá funcionando muy bien en zonas donde la conectividad

de red es irregular. CouchDB está disponible también en casa, en un teléfono inteligente o en un centro de datos. Todo esto viene con algunas ventajas y desventajas. Debido a la semejanza con la web, CouchDB es de alta latencia haciéndolo preferente para el almacenamiento de datos local. Aunque es capaz de trabajar de una manera no-distribuida, CouchDB no se adapta bien a las implementaciones más pequeñas.

Las bases de datos CouchDB están compuestas de documentos que constan de campos y archivos adjuntos, así como una descripción del documento en forma de metadatos que se mantiene de forma automática por el sistema. Cuenta con todas las capacidades ACID que se conocen del mundo de las bases de datos relacionales, pero la ventaja de CouchDB es que los datos se empaquetan y están listos para su almacenamiento y manipulación en lugar de dispersarlos en filas y tablas.

CouchDB cuenta con las siguientes capacidades:

- **Compactación:** Las bases de datos se comprimen para eliminar espacio desperdiciado. Esto ayuda a alcanzar el rendimiento y la eficiencia necesaria para lograr la persistencia.
- **Modelo vista:** Un mecanismo para filtrar, organizar e informar sobre los datos que utilizan un conjunto de definiciones que son almacenados como documentos en la base de datos.
- **Servicios de distribución y replicación:** El almacenamiento de documentos está diseñado para proporcionar la replicación bidireccional. Las réplicas parciales se pueden mantener para apoyar la distribución basada en criterios o migración a dispositivos con conectividad limitada.

Las situaciones en que las implementaciones de CouchDB resultan efectivas son:

- Gestión de contenidos de gran volumen
- Aplicaciones con conectividad de red limitada o lenta
- Escalabilidad desde teléfonos inteligentes hasta centros de datos

4.6.3.4 Base de datos tabular HBase. Una de las bases de datos tabulares más populares es HBase. Es también un proyecto de la Apache Software Foundation. HBase utiliza el sistema de archivos Hadoop y MapReduce para el almacenamiento de los datos.

El diseño de HBase está inspirado en BigTable de Google, una forma eficiente de almacenar datos no relacionales. Por lo tanto, las implementaciones de HBase son altamente escalables, dispersas, distribuidas y persistentes. El mapa esta indexado por una clave de fila, una clave de columna y una marca de tiempo, cada valor en el mapa es una matriz de bytes no interpretados. Cuando una implementación de Big Data requiera acceso en tiempo real para lectura y escritura de datos, HBase es una buena solución. A menudo son utilizadas para almacenar los resultados para su posterior procesamiento analítico.

Las características más importantes de HBase son las siguientes:

- **Consistencia:** Aunque no implementa las características ACID, HBase ofrece una lectura/ escritura consistente. Esto significa que se puede utilizar cuando se requiere alta velocidad, siempre y cuando no se necesiten el resto de características adicionales ofrecidas por los sistemas gestores de bases de datos relacionales, como soporte completo de transacciones.

- **Fragmentación:** Debido a que los datos se distribuyen a través del sistema de archivos, HBase ofrece transparencia, división automática y redistribución del contenido.
- **Alta disponibilidad:** HBase soporta conmutación por error y recuperación en redes LAN y WAN, utilizando un servidor principal.
- **Cliente API:** HBase ofrece acceso mediante programación a través de una API de Java.
- **Apoyo a las operaciones de TI**: Los implementadores pueden exponer el rendimiento y otros parámetros a través de un conjunto de páginas web incorporadas.

Las implementaciones en las cuales HBase es una buena opción son las siguientes:

- Alto volumen, recolección y procesamiento de datos incrementales.
- Intercambio de información en tiempo real.
- Contenido que cambia frecuentemente.

4.6.3.5 Base de datos orientada a grafos Neo4J. Una de las bases de datos orientadas a grafos más utilizada es Neo4J. Es un proyecto de código abierto bajo la licencia GNU versión 3.0. Neo4J es una base de datos que cumple con las características ACID, proporcionando una alta disponibilidad. Es una base de datos confiable y escalable, fácil de modelar debido a que siguen las propiedades de los grafos. No requiere de un esquema, ni requiere de tipos de datos, lo que lo hace muy flexible.

Con esta flexibilidad vienen algunas limitaciones, como que no se pueden auto-referenciar los nodos, así que si esto es necesario, Neo4J no es la mejor solución. La capacidad de replicación de Neo4J es muy buena, sólo se pueden replicar

grafos enteros, colocando un límite en el tamaño total del grafo de aproximadamente 34 mil millones de nodos y de 34 mil millones de relaciones.

Las características más importantes de esta base de datos es que permite la integración con otras bases de datos, permitiendo la interoperabilidad con almacenes de datos que no están basados en grafos. Además, ofrece servicios de sincronización basados en eventos, lo que brinda una sincronización periódica. Neo4J realiza copias de seguridad en frío (Es decir, cuando la base de datos no se está ejecutando) y en caliente (Cuando está funcionando). También soporta un lenguaje de consulta declarativo llamado Cypher, diseñado especialmente para consultar grafos y sus componentes. Los comandos Cypher están basados en la sintaxis SQL y están dirigidos a consultas ad hoc de los datos del gráfico.

Las implementaciones de la base de datos Neo4J son las más adecuadas cuando se trata de redes sociales o clasificación de dominios biológicos o médicos.

5. CONCLUSIONES

Teniendo en cuenta que en las bases de datos relacionales los datos son organizados en filas y columnas para ser almacenados en una tabla, donde es necesario que los datos posean un estructura, lo que hace que sea difícil dar soporte a los tipos de datos que trae consigo Big Data, ya que muchos de ellos no tienen una estructura definida y al tratar de encajar en el modelo relacional se pierde mucha información. En cambio, las bases de datos no relacionales satisfacen la necesidad de flexibilidad, puesto que cada uno de sus tipos encajan mejor para ciertos tipos de datos

Debido al gran volumen de datos que se manejan en el Big Data, es necesario que los sistemas puedan escalar de una manera rápida y económica. Los sistemas de bases de datos relacionales son centralizados, escalan verticalmente, es decir la única manera en que pueden crecer es adicionando hardware muy costoso. Por el contrario, las bases de datos no relacionales escalan horizontalmente, es decir distribuyen la información en un conjunto de nodos, denominados clúster. Estos nodos son servidores genéricos.

Las bases de datos relacionales se caracterizan por administrar la información en tablas, de igual manera las relaciones que existan entre estas, esto significa que los datos deben llevar un orden, para poder ser administrados y que no se presente duplicidad en estos, esta técnica es muy útil para las bases de datos que no son muy grandes. Sin embargo, para Big Data este método no es el más eficiente debido a que este administra sus datos sin ninguna estructura definida y como el modelo relacional solo puede soportar una lectura exclusiva de lenguajes estructurados no brinda la mejor solución en cuanto a velocidad y rendimiento se refiere, cuestión que el modelo no relacional brinda ya que al utilizar clave – valor en sus implementaciones logra dividir una solicitud en varias tareas haciendo que se procese de manera simultánea en distintos nodos y optimizando la velocidad de procesamiento de la información sin importar la falta de estructura de Big Data.

Sabiendo que las características principales de Big Data son velocidad, variedad y volumen se puede concluir que las bases de datos que más se ajustan son las no relacionales ya que cumplen a cabalidad estos requisitos, al ser estos SGBD escalables horizontalmente se adaptan según el crecimiento constante de Big Data. Además puede administrar datos estructurados, semiestructurados y no estructurados, por lo cual brindan mayor flexibilidad puesto que pueden soportar los diferentes tipos de datos que trae consigo el Big Data. Por último la velocidad es uno de los factores más importantes pues en la actualidad la disposición de la información tiene que ser precisa y rápida para los altos flujos de solicitudes que se manejan para lo cual las implementaciones del modelo no relacional trabajan en nodos diferentes dividiendo una solicitud en varias tareas haciendo que la velocidad de procesamiento sea mucho más rápida y económica.

BIBLIOGRAFÍA

ADNAN, Muhammad. AFZAL, Muhammad. JAN, Roohi y MARTINEZ, A M. Minimizing Big Data Problems using Cloud Computing Based on Hadoop Architecture. En: IEEE High-capacity Optical Networks and Emerging/Enabling Technologies. Diciembre de 2014. Págs. 99 – 103. [Citado en Abril de 2014].

BOICEA, Alexandru, RADULESCU, Florin y AGAPIN, Laura. MongoDB vs Oracle– database comparison. En: IEEE Third International Conference on Emerging Intelligent Data and Web Technologies. Septiembre, 2012; págs 330 - 335. [Citado el 28 de Marzo de 2015].

CARPANI, Francisco. CMDM: Un Modelo Conceptual para la Especificación de Bases Multidimensionales.Uruguay, 2000, 158 p. Tesis de Maestría. Universidad de la República. Facultad de Ingeniería.

CASTRO, Alexander, GONZALEZ, Juan Sebastián y CALLEJAS, Mauro. Utilidad y funcionamiento de las bases de datos.Tunja: Facultad de Ingeniería UPTC, 2012.

CHANG, Fay. DEAN, Jeffrey. GHEMAWAT, Sanjay. HSIEH, Wilson. WALLACH, Deborah. BURROWS, Mike. CHANDRA, Tushar. FIKES, Andrew y GRUBER, Robert.Bigtable: A Distributed Storage System for Structured Data. En: Usenix [En línea]. 2006. Disponible en: <http://static.usenix.org/events/osdi06/tech/chang/chang_html/?em_x=22>.

GUDIVADA, Venkat. RAO, Dhan y RAGHAVAN, Vijah. NoSQL Systems for Big Data Management. En: IEEE 10th World Congress on Services. 27 de Junio – 2 de Julio del 2014. Págs 190 -197. [Citado en Marzo de 2015].

HURWITZ, Judith. NUGENT, Alan. HALPER, Fern y KAUFMAN, Marcia. Big Data For Dummies, New Jersey: John Wiley y Sons, INC, 2013.

HURWITZ, Judith. KAUFMAN, Marcia y HALPER, Fern. Cloud Services ForDummies. New Jersey: John Wiley y Sons, INC, 2012.

JI, Changquing. LI, Yu. QIU, Wenming. AWADA, Uchechukwu y LI, Kequi. Big Data Processing in Cloud Computing Environments. En: IEEE International Symposium on Pervasive Systems, Algorithms and Networks. Diciembre de 2012. Págs 17 – 23. [Citado en Abril de 2015].

KATAL, Avita, WAZID, Mohammad y GOUDAR, R H. Big Data: Issues, Challenges, Tools and Good Practices. En: IEEE. Agosto del 2013. Págs. 404 – 409. [Citado en Abril de 2015].

LI, Yishan y MANOHARAN, Sathiamoorthy. A performance comparison of SQL and NoSQL databases. En: IEEE. Agosto de 2013. Págs. 15 -19. [Citado en Abril de 2015].

LOPEZ, David. Análisis de las posibilidades de uso de Big Data en las organizaciones. Cantabria, 2012, 67 p. Tesis de Maestría. Universidad de Cantabria.

NAHEMAN, Wumuti y WEI, Jianxin. Review of NoSQL Databases and Performance Testing on HBase. En: IEEE International Conference on Mechatronic Sciences, Electric Engineering and Computer (MEC). Diciembre de 2013. Págs. 2304 – 2309. [Citado en Abril de 2015].

OPPEL, Andy y SHELDON, Robert. Fundamentos de SQL. Tercera Edición. Traducido por Carlos Fabián Jiménez Castillo. México: McGraw Hill InterAmerica Editores SA, 2010.

O´Really Media. Big Data Now: 2012 Edition. Primera edición. Estados Unidos: O´Really Media Inc, 2012.

O´Really Media. Planning for Big Data. Primera edición. Estados Unidos: O´Really Radar Team, 2012.

PURCELL, Bernice. Big Data using Cloud Computing. En: Journal of Technology Research [En línea]. 2013. Disponible en: <http://www.aabri.com/manuscripts/131457.pdf>.

PURCELL, Bernice. The emergence of “big data” technology and analytics. En: Journal of Technology Research [En línea]. 2013. Disponible en: <http://www.aabri.com/manuscripts/121219.pdf>.

RAMOS, Maria, RAMOS, Alicia y MONTERO, Fernando. Sistemas gestores de bases de datos. España: McGraw-Hill, 2006.

RICARDO, Catherine. Bases de Datos. Segunda edición. Ciudad de Mexico: McGraw-Hill InterAmerica Editores SA, 2004.

SAGIROGLU, Seref y SINANC Duygu.Big Data: A Review. En: IEEE International Conference on Collaboration Technologies and Systems. Mayo del 2013. Págs. 42 – 47. [Citado en Abril de 2015].

SOLANO, Loanny. Análisis de los Sistemas de Gestión de Bases de Datos actuales como soporte para las tecnologías de Internet de las Cosas. Valencia, 2013, 89 p. Tesis de Maestría (Ingeniería del Software). Universitat Politècnica de València.

STRAUCH, Christof. NoSQL Databases. Stuttgart: Stuttgart Media University, 2011.

WANG, Gouxi y TANG, Jianfeng. TheNoSQL Principles and Basic Application of Cassandra Model. En: IEEE International Conference on Computer Science and Service System. Agosto 2012. Págs. 1332 – 1335. [Citado en Abril de 2015].