

DISEÑO Y DESARROLLO DE UNA GUÍA PARA LA IMPLEMENTACIÓN DE UN  
AMBIENTE BIG DATA EN LA UNIVERSIDAD CATÓLICA DE COLOMBIA

FABIÁN ANDRÉS GUERRERO LÓPEZ  
JORGE EDUARDO RODRÍGUEZ PINILLA

UNIVERSIDAD CATÓLICA DE COLOMBIA  
FACULTAD DE INGENIERÍA  
PROGRAMA DE INGENIERÍA DE SISTEMAS  
MODALIDAD TRABAJO DE INVESTIGACIÓN  
BOGOTÁ  
2013

DISEÑO Y DESARROLLO DE UNA GUÍA PARA LA IMPLEMENTACIÓN DE UN  
AMBIENTE BIG DATA EN LA UNIVERSIDAD CATÓLICA DE COLOMBIA

FABIÁN ANDRÉS GUERRERO LÓPEZ  
JORGE EDUARDO RODRÍGUEZ PINILLA

Trabajo de Grado para optar al título de  
Ingeniero de Sistemas

Director  
YASSER DE JESÚS MURIEL PEREA  
Ingeniero de Sistemas

UNIVERSIDAD CATÓLICA DE COLOMBIA  
FACULTAD DE INGENIERÍA  
PROGRAMA DE INGENIERÍA DE SISTEMAS  
MODALIDAD TRABAJO DE INVESTIGACIÓN  
BOGOTÁ  
2013



## Atribución-NoComercial-SinDerivadas 2.5 Colombia (CC BY-NC-ND 2.5)

La presente obra está bajo una licencia:  
**Atribución-NoComercial-SinDerivadas 2.5 Colombia (CC BY-NC-ND 2.5)**  
Para leer el texto completo de la licencia, visita:  
<http://creativecommons.org/licenses/by-nc-nd/2.5/co/>

### Usted es libre de:



Compartir - copiar, distribuir, ejecutar y comunicar públicamente la obra

### Bajo las condiciones siguientes:



**Atribución** — Debe reconocer los créditos de la obra de la manera especificada por el autor o el licenciante (pero no de una manera que sugiera que tiene su apoyo o que apoyan el uso que hace de su obra).



**No Comercial** — No puede utilizar esta obra para fines comerciales.



**Sin Obras Derivadas** — No se puede alterar, transformar o generar una obra derivada a partir de esta obra.

Nota de aceptación

Aprobado por el comité de grado  
en cumplimiento de los requisitos  
Exigidos por la Facultad de  
Ingeniería y la Universidad Católica  
de Colombia para optar al título de  
Ingenieros de Sistemas.

---

Director

---

Revisor Metodológico

Bogotá, 27 de Noviembre, 2013

## **AGRADECIMIENTOS**

Agradecemos en primer lugar a Dios por permitirnos llegar a esta etapa de nuestras vidas, reflejando a través de este trabajo todas las enseñanzas recibidas durante el transcurso de nuestra carrera.

En segundo lugar agradecemos la compañía de nuestras familias, por apoyarnos en cada momento, brindando su protección y ayuda para tomar las mejores decisiones.

Por último, agradecemos a nuestros amigos y compañeros de universidad y a todos los maestros que tuvimos durante la carrera; también agradecemos a la profesora Claudia Milena Rodríguez Álvarez quien nos guio en gran parte del proyecto y al profesor Yasser de Jesús Muriel Perea, quien en esta última fase acepto asesorarnos para la finalización del proyecto.

## TABLA DE CONTENIDO

	pág.
INTRODUCCIÓN	14
1. GENERALIDADES	15
1.1 ANTECEDENTES	15
1.2 PLANTEAMIENTO DEL PROBLEMA	16
1.3 OBJETIVOS	17
1.3.1 Objetivo General	17
1.3.2 Objetivos Específicos	17
1.4 JUSTIFICACIÓN	17
1.5 DELIMITACIÓN	17
1.5.1 Espacio	17
1.5.2 Tiempo	18
1.5.3 Alcance	18
1.6 MARCO TEÓRICO	18
1.6.1 Definición de Big Data	18
1.6.2 Evolución de Big Data	20
1.6.3 Arquitectura de Big Data	21
1.6.3.1 Ingreso de datos	22
1.6.3.2 Gestión de datos	22
1.6.3.3 Tiempo real de procesamiento	22
1.6.3.4 Análisis de datos	22
1.6.4 Bases de datos NoSQL	23
1.6.4.1 Almacenes Key-Value	24
1.6.4.2 Bases de datos columnares	24
1.6.4.3 Bases de datos orientadas a documentos	24
1.6.4.4 Bases de datos orientados a grafos	24
1.6.4.5 Teorema CAP	25
1.6.4.6 Base vs Acid	25
1.6.5 Hadoop	32
1.6.6 MapReduce	32
1.6.7 HDFS	33
1.6.8 Chukwa	34

1.6.9 Sqoop	35
1.6.10 Flume	35
1.6.11 Avro	36
1.6.12 Hive	36
1.6.13 Pig	37
1.6.14 Transformación de una base de datos relacional a NoSQL	38
1.6.14.1 El modelo relacional vs el modelo de datos orientado a documentos	39
1.6.14.2 El modelo relacional vs el modelo de datos orientado a grafos	42
1.7 MARCO CONCEPTUAL	45
1.7.1 Apache CouchDB	45
1.7.2 Base de datos	46
1.7.3 Bases de datos NoSQL	46
1.7.4 Big Data	46
1.7.5 Big Transaction Data	46
1.7.6 Biometrics	46
1.7.7 BSON	46
1.7.8 Cluster	47
1.7.9 Hadoop	47
1.7.10 Hbase	47
1.7.11 HDFS	47
1.7.12 Human Generated	47
1.7.13 JSON	47
1.7.14 Lenguaje SQL	48
1.7.15 Machine-to-Machine (M2M)	48
1.7.16 MapReduce	48
1.7.17 Master-Slave	48
1.7.18 Modelo de datos entidad/relación	48
1.7.19 MongoDB	49
1.7.20 Multi-master replication	49
1.7.21 Sharding	49
1.7.22 Web and Social Media	49
1.7.23 XQuery	49
1.8 METODOLOGÍA	50

1.8.1 Tipo de Estudio	50
1.8.2 Fuentes de Información	50
1.8.2.1 Fuentes primarias	50
1.8.2.2 Fuentes secundarias	50
2.CONCLUSIONES	51
BIBLIOGRAFÍA	52
ANEXOS	57



## LISTA DE TABLAS

	<b>pág.</b>
Tabla 1. Atributos de Big Data	19
Tabla 2. Comparación de propiedades Base y Acid	25
Tabla 3. Bases de datos NoSQL orientada Key-Value	27
Tabla 4. Bases de datos NoSQL orientadas a documentos	28
Tabla 5. Bases de datos NoSQL orientadas a columnas	29
Tabla 6. Bases de datos NoSQL orientadas a grafos	30
Tabla 7. Uso de SQL y NoSQL	44

## LISTA DE FIGURAS

	pág.
Figura 1. Arquitectura de un ambiente de Big Data	21
Figura 2. Comparación taxonomía	24
Figura 3. Arquitectura Hadoop	32
Figura 4. Proceso MapReduce	33
Figura 5. Arquitectura HDFS	34
Figura 6. Diagrama de Chukwa	35
Figura 7. Flujo de datos	36
Figura 8. Aspectos para utilizar NoSQL o RDBMS	38
Figura 9. Comparación entre un modelo de datos relacional y uno orientado a documentos	39
Figura 10. Proceso de normalización de una base de datos relacional	40
Figura 11. Ejemplo de formato de registro en CouchBase	41
Figura 12. Un ejemplo de grafo aciclico directo	43
Figura 13. Comparación de rendimiento entre MySQL y Neo4j	44

## LISTA DE ANEXOS

	<b>pág.</b>
Anexo A. Guía para la creación del ambiente Big Data	57
Anexo B. Guía para la implementación del caso de estudio	79

## GLOSARIO

**B-TREE:** estructuras de datos de árbol que se encuentran comúnmente en las implementaciones de bases de datos y sistemas de archivos.

**DATO:** representación de hechos referentes a una persona, cosa o transacción. Incluyen atributos, variables cuantitativas y cualitativas

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

**GRAFO:** conjunto, no vacío, de objetos llamados vértices o nodos y una selección de pares de vértices llamados aristas.

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

**JAVA:** lenguaje de programación orientado a objetos creado por Sun Microsystems que permite crear programas que funcionan en cualquier tipo de ordenador y sistema operativo.

**JAVA SCRIPT:** lenguaje de programación que permite a los desarrolladores crear acciones en sus páginas web.

**MOTOR DE BASE DE DATOS:** software dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan.

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

**SISTEMA OPERATIVO:** conjunto de programas de software que controlan el funcionamiento general del computador y que administran los recursos del mismo.

**SOFTWARE:** programas usados para dirigir las funciones de un sistema de computación o un hardware.

## ACRÓNIMOS

**ACID:** (Atomicity, Consistency, Isolation, Durability) conjunto de características o propiedades para las bases de datos.

**AGPL:** (Affero General Public License) licencia pública.

**API:** (Application Programming Interface) conjunto de funciones y procedimientos utilizado por un software para sus aplicaciones.

**BSON:** Binary JSON.

**GPL:** (GNU General Public License) Licencia Pública General de GNU.

**HDFS:** (Hadoop Distributed File System) sistemas de archivos distribuido Hadoop.

**HTTP:** (Hypertext Transfer Protocol) protocolo de transferencia de hipertexto.

**JSON:** (JavaScript Object Notation) formato ligero para el intercambio de datos.

**NOSQL:** (No Structured Query Language) bases de datos no-relacionales.

**RDBMS:** (Relational Database Management System) sistema de Gestión de Base de Datos Relacional.

**SQL:** (Structured Query Language) lenguaje declarativo de acceso a bases de datos relacionales.

**XML:** (Extensible Markup Language): lenguaje de marcas extensibles, utilizado por algunas bases de datos.

## INTRODUCCIÓN

Con el constante crecimiento de información en cada uno de los aspectos más relevantes mundialmente como sociedad, comercio y ciencia, se vuelve necesario un cambio con respecto al manejo de la información, que hasta hace poco tiempo se venía implementando a partir de bases de datos relacionales. Si bien este esquema funcionaba para ambientes que almacenaban teras de información con datos puntuales y que a través de consultas SQL se podía buscar la información requerida, se vuelve dispendioso y costoso al momento de analizar un mayor volumen de información, en ambientes donde anualmente se almacenan petas de todo tipo de datos, incluyendo aquí los no estructurados, como archivos de video y audio; un ejemplo de esto es “Google, que recibe a diario trillones de bytes, con el objetivo de ofrecer muchos de los servicios que actualmente se conocen como el mismo motor de búsqueda y Google Earth”<sup>1</sup>.

El manejo de grandes cantidades de información conlleva a pensar en la implementación de herramientas que permitan administrar y gestionar este tipo de datos no estructurados y semi-estructurados, en la búsqueda de patrones concurrentes para la toma de decisiones. “Actualmente se pueden encontrar tecnologías como Hadoop, MapReduce y bases de datos NoSQL, que se pueden implementar en la creación de un ambiente Big Data”<sup>2</sup>.

De esta manera, el proyecto busco la implementación de una arquitectura para crear un ambiente Big Data en la Universidad Católica de Colombia, teniendo en cuenta aspectos importantes como el software y el hardware que se debía utilizar para realizar dicha labor, de igual manera todos los procedimientos que implicaba empezar a utilizar bases de datos no relacionales. Todo esto se ve reflejado a través de dos guías, donde se podrá encontrar de una manera detallada los pasos para la implementación de un ambiente Big Data, sus elementos y procesos para ingresar y consultar información a través de un ejemplo específico, con el propósito de diseñar estrategias y buscar patrones que permitan una buena gestión de la información.

---

<sup>1</sup> BBVA. Big Data: ¿En qué punto estamos? [en línea]. Bogotá: BBVA [citado 7 Agosto, 2013]. Disponible en internet: <<https://www.centrodeinnovacionbbva.com/innovation-edge/21-big-data/p/153-big-data-en-que-punto-estamos>>

<sup>2</sup> CENATIC. Open Smart Cities II: Big Data de Código Abierto [en línea]. Badajoz: Cenaltic [citado 10 agosto, 2013]. Disponible en internet: <[http://observatorio.cenatic.es/index.php?option=com\\_content&view=article&id=804:open-smart-cities-ii-open-big-data-&catid=94:tecnologia&Itemid=137#ftn3](http://observatorio.cenatic.es/index.php?option=com_content&view=article&id=804:open-smart-cities-ii-open-big-data-&catid=94:tecnologia&Itemid=137#ftn3)>

# 1. GENERALIDADES

## 1.1 ANTECEDENTES

La instalación de un ambiente utilizando herramientas de Big Data puede ser una temática muy ambigua, debido a la cantidad de aplicaciones existentes y a sus diferentes usos. Esto tiene como consecuencia que existan muy pocas guías relacionadas al montaje de un ambiente de Big Data con respecto a la implementación de herramientas específicas.

En esta temática se han realizado algunos trabajos como el proyecto de las estudiantes Ana Barragán y Andrea Forero de la Universidad Católica de Colombia, titulado “Implementación de una base de datos No SQL para la generación de la matriz o/d”<sup>3</sup>, donde se expone cómo resolver una problemática relacionada al tema de transporte, utilizando como herramienta la base de datos basada en Hadoop – Hbase.

Otro ejemplo es el trabajo realizado por Carmen Palacios Díaz<sup>4</sup> estudiante de la universidad Carlos III de Madrid, quien explica la implementación de un ambiente con Hadoop y todos los requisitos necesarios para llevar a cabo dicha instalación.

Otro documento es el realizado por Albert Gutiérrez Milla, titulado “Análisis Bioinformáticos sobre la tecnología Hadoop”<sup>5</sup>, donde se explica los beneficios que conlleva utilizar elementos iniciales en Big Data como MapReduce y Hadoop y la implementación de ejemplos y aplicativos.

En cuanto a la aplicación, muchas empresas han visto la importancia de utilizar métodos más efectivos para la gestión de su información, que les permitan realizar análisis con respecto a sus productos y servicios, por tal motivo la inclusión de Big Data ofrece grandes beneficios empresariales, como el análisis de patrones para el diseño de estrategias de venta y toma de decisiones.

En el caso de empresas colombianas que han tenido la necesidad y que han optado por gestionar su información por medio de Big Data, un sistema mucho más robusto, confiable y capaz de resolver sus problemas en cuanto a toda la

---

<sup>3</sup> BARRAGAN, Ana, FORERO, Andrea. Implementación de una base de datos No SQL para la generación de la matriz o/d. Bogotá: Universidad Católica de Colombia. Facultad de Ingeniería de sistemas. Modalidad Proyecto de investigación, 2012

<sup>4</sup> UNIVERSIDAD CARLOS III DE MADRID. Evaluación de la herramienta de código libre Apache Hadoop [en línea]. Madrid: Carmen Palacios Díaz [citado 10 agosto, 2013]. Disponible en internet: <[http://e-archivo.uc3m.es/bitstream/10016/13533/1/MemoriaPFC\\_MCarmenPalacios.pdf](http://e-archivo.uc3m.es/bitstream/10016/13533/1/MemoriaPFC_MCarmenPalacios.pdf)>.

<sup>5</sup> UNIVERSIDAD AUTÓNOMA DE BARCELONA. Análisis Bioinformáticos sobre la tecnología hadoop [en línea]. Madrid: Carmen Palacios Díaz [citado 10 agosto, 2013]. Disponible en internet: <[http://ddd.uab.cat/pub/trerecpro/2012/hdl\\_2072\\_196270/GutierrezMillaAlbertR-ETISa2010-11.pdf](http://ddd.uab.cat/pub/trerecpro/2012/hdl_2072_196270/GutierrezMillaAlbertR-ETISa2010-11.pdf)>.

información que estaban almacenando, se encuentran tres casos según González<sup>6</sup>:

Activos S.A., que lleva en el sector más de 29 años, esta empresa implementó una plataforma para soportar los procesos de negocio, teniendo en cuenta el gran volumen de información que manejan diariamente, permitiendo así una mejor administración, gestión y almacenamiento de sus procesos.

Nutresa, esta multinacional tiene un sistema privado de computación en la nube, el cual permite unificar todos los procesos de negocio ya sean financieros, logísticos o de marketing, en sus diferentes sucursales, logrando una organización de sus datos para tener una alta competencia en el mercado.

Colombina S.A., esta compañía adquirió en el 2007 una plataforma tecnológica que le permitió mejorar la distribución de la información de las ventas de cada una de sus sucursales.

Teniendo en cuenta lo anterior, se puede observar que la implementación de tecnologías de Big Data adquiere mayor importancia en las organizaciones, dado la necesidad de procesar y analizar grandes cantidades información para la toma de decisiones.

## **1.2 PLANTEAMIENTO DEL PROBLEMA**

**1.2.1 Descripción del Problema.** Big data es un concepto nuevo que se está desarrollando hace pocos años y que solo las grandes empresas han iniciado a implementarlo, la documentación que se encuentra sobre este tema es poca y la mayoría se encuentra en inglés, las implementaciones que se han hecho son aisladas y no hay una descripción o una guía paso a paso que permita realizar la integración de las herramientas open source que se puedan encontrar.

Existen organizaciones y empresas que no pueden pagar o tener una consultoría adecuada sobre el manejo de la nueva tendencia Big Data, que les permita seleccionar las herramientas adecuadas y la forma de configurar un ambiente para adaptarlo a su modelo de negocio.

**1.2.2 Formulación del Problema.** De acuerdo al problema expuesto en el numeral anterior se plantean las siguientes preguntas de investigación:

---

<sup>6</sup> EMPRESAS.IT. Tres casos de éxito de Big Data en Colombia [en línea]. Bogotá: Alejandro González [citado 7 Agosto, 2013]. Disponible en internet: <<http://empresas.it/2013/05/tres-casos-de-exito-de-big-data-en-colombia/>>



¿Qué tecnologías utilizar e integrar para la construcción de un ambiente Big Data en la universidad, que permita apoyar la investigación y aprendizaje en este campo?

¿Qué aspectos se debe tener en cuenta para la configuración e instalación de las herramientas escogidas?

### **1.3 OBJETIVOS**

#### **1.3.1 Objetivo General**

- Diseñar y desarrollar una guía para la implementación de un ambiente de Big Data en la Universidad Católica de Colombia.

#### **1.3.2 Objetivos Específicos**

- Identificar las tecnologías, herramientas de software y requerimientos de hardware necesarios para la implementación de un ambiente de Big Data.
- Realizar el montaje, configuración e integración de las herramientas para la construcción del ambiente de Big Data.
- Diseñar el caso de estudio que se utilizará para realizar las pruebas del ambiente configurado.

### **1.4 JUSTIFICACIÓN**

La importancia de desarrollar el proyecto radica en apoyar el aprendizaje de las nuevas tendencias en la gestión de la información, fomentando de esta manera el campo de investigación dentro de la comunidad universitaria.

Así mismo, el proyecto contribuye y apoya las temáticas desarrolladas en el semillero de investigación GINOSKO, lo cual permitirá realizar análisis más detallados a los distintos casos que se puedan presentar y que no puedan ser resueltos mediante bases de datos relacionales, con esto se fortalecerá la temática para posteriores vinculaciones en proyectos de investigación o trabajos afines.

### **1.5 DELIMITACIÓN**

**1.5.1 Espacio.** El espacio para la realización del proyecto será en la Universidad Católica de Colombia, teniendo en cuenta que se busca implementar un ambiente de pruebas y desarrollo para Big Data.

**1.5.2 Tiempo.** El planteamiento, desarrollo e implementación del proyecto será realizado durante 4 meses.

**1.5.3 Alcance.** El proyecto culmina con el diseño y montaje del ambiente de Big Data, teniendo en cuenta que las herramientas y tecnologías serán analizadas y definidas como parte del proyecto. Las herramientas que se van a implementar serán open source, permitiendo así que se puedan descargar y trabajar sin alguna restricción de licencia.

El desarrollo del proyecto incluye:

- Un caso de estudio que permitirá realizar las pruebas necesarias del ambiente implementado.
- Guías paso a paso sobre la instalación, configuración e integración de cada una de las herramientas requeridas para la construcción del ambiente.
- Ejemplo para la conversión de un modelo E/R a modelo NoSQL.

## **1.6 MARCO TEÓRICO**

Big Data ha representado un movimiento revolucionario, en cuanto al manejo de toda la información que hasta hace poco tiempo era poco probable se pudiera analizar, con la inclusión de este concepto se puede pensar en el procesamiento de datos del tipo no estructurado, como video, audio, sistemas GPS y gran número de sensores ubicados en dispositivos móviles, automóviles y equipos industriales entre otros.

**1.6.1 Definición de Big Data.** Según Gartner<sup>7</sup>, Big data es una referencia a aquellos sistemas de información que manejan conjuntos de datos de gran volumen, de alta velocidad, de veracidad, de valor y de gran variedad de recursos, que demandan formas rentables e innovadoras de procesamiento de la información para mejorar la comprensión y la toma de decisiones.

Según Gualtieri<sup>8</sup> Big data es la solución al crecimiento exponencial de los datos, en el momento en que se hace difícil su administración con respecto al almacenamiento, procesamiento y acceso.

De esto se puede obtener beneficios como:

---

<sup>7</sup> GARTNER. Big Data [en línea]. Connecticut: Gartner [citado 22 septiembre, 2013]. Disponible en internet: <<http://www.gartner.com/it-glossary/big-data/>>

<sup>8</sup> FORRESTER. The pragmatic definition of Big data [en línea]. Cambridge: Mike Gualtieri [citado 29 septiembre, 2013]. Disponible en internet: <[http://blogs.forrester.com/mike\\_gualtieri/12-12-05-the\\_pragmatic\\_definition\\_of\\_big\\_data](http://blogs.forrester.com/mike_gualtieri/12-12-05-the_pragmatic_definition_of_big_data)>

- “Optimizar el cálculo y la precisión algorítmica para reunir, analizar, enlazar y comparar conjuntos de grandes datos”<sup>9</sup>.
- “Identificar patrones para la toma de decisiones en los ámbitos económico, social, técnico y legal”<sup>10</sup>.

La mayoría de las definiciones que se pueden encontrar de Big data están enfocadas al volumen de los datos, al almacenamiento de dicha información, de esto se puede concluir que el volumen importa pero que también existen otros atributos importantes de Big data, estos son: “la velocidad, la veracidad, la variedad y el valor”<sup>11</sup>. Estos cinco aspectos constituyen una definición comprensiva y además destruyen el mito acerca de que Big data se trata únicamente del volumen. A cada uno de estos aspectos se le atribuyen las siguientes características:

**Tabla 1. Atributos de Big Data**

<b>Volumen</b>	<b>Velocidad</b>	<b>Variedad</b>	<b>Veracidad</b>	<b>Valor</b>
Almacenamiento En terabytes	Por lotes	Estructurado	Integridad y Autenticidad	Estadísticas
Registros	Tiempo Cercano	No estructurado	Origen y Reputación	Eventos
Transacciones	Tiempo Real	Multi-factor	Disponibilidad	Correlaciones
Tablas y Archivos	Procesos	Probabilística	Responsabilidad	Hipótesis

Fuente: UNIVERSITY OF AMSTERDAM. Defining the Big Data Architecture Framework [en línea]. Ámsterdam: Yuri Demchenko [citado 22 septiembre, 2013]. Disponible en internet: <[http://bigdatawg.nist.gov/\\_uploadfiles/M0055\\_v1\\_7606723276.pdf](http://bigdatawg.nist.gov/_uploadfiles/M0055_v1_7606723276.pdf)>

Según Barranco<sup>12</sup> con el paso del tiempo, las empresas han fomentado la creación de nuevas estrategias para la toma de decisiones, dando un importante lugar al análisis predictivo, ya que con esto se han podido determinar diversos tipos de patrones entre la sociedad, generando como consecuencia gran cantidad de beneficios consistentes en la innovación, investigación y desarrollo de nuevas soluciones.

<sup>9</sup> ROUTLEDGE. Critical questions for Big data [en línea]. Cambridge: Danah Boyd & Kate Crawford [citado 29 septiembre, 2013]. Disponible en internet: <<http://www.tandfonline.com/doi/pdf/10.1080/1369118X.2012.678878>>

<sup>10</sup> Ibid.

<sup>11</sup> UNIVERSITY OF AMSTERDAM. Defining the Big Data Architecture Framework [en línea]. Ámsterdam: Yuri Demchenko [citado 22 septiembre, 2013]. Disponible en internet: <[http://bigdatawg.nist.gov/\\_uploadfiles/M0055\\_v1\\_7606723276.pdf](http://bigdatawg.nist.gov/_uploadfiles/M0055_v1_7606723276.pdf)>

<sup>12</sup> IBM. ¿Qué es big data? [en línea]. México D.F: Ricardo Barranco [citado 15 agosto, 2013]. Disponible en internet: <<http://www.ibm.com/developerworks/ssa/local/im/que-es-big-data/>>

La generación de estos datos en los últimos años ha venido creciendo de manera inmensurable y se proyecta a seguirlo haciendo, por tal motivo es que Big data se convertirá en uno de los principales aspectos a tener en cuenta dentro de los ámbitos comercial, científico y social, todo debido al gran impacto económico e innovador que este ha representado.

Lo anterior se puede ver reflejado en una recopilación de estadísticas nombrada “A comprehensive list of Big data statistics”, donde se exponen algunos puntos del porqué la importancia de Big data, algunos de ellos son:

- Actualmente en el mundo digital existen 2.7 zetabytes de datos.
- El gobierno estadounidense invierte cerca de 200 millones de dólares en investigación sobre Big data.
- La red social Facebook almacena, registra y analiza diariamente 30 petabytes de datos, 94% de los usuarios de Hadoop realiza análisis de grandes volúmenes de información que antes no se podía analizar.
- Descifrar el genoma humano tardó cerca de 10 años, actualmente ese proceso se puede realizar en una semana<sup>13</sup>.

**1.6.2 Evolución de Big Data.** Big Data ha demostrado tener un crecimiento exponencial en los últimos años. “Su historia se remonta al nacimiento de las primeras herramientas informáticas que llegaron en 1940. En esta misma década comenzaron a aparecer programas que eran capaces de predecir posibles escenarios futuros. Por ejemplo, el equipo del Proyecto Manhattan (1944) que realizaba simulaciones por ordenador para predecir el comportamiento de una reacción nuclear en cadena”<sup>14</sup>.

Según Artaza<sup>15</sup>, no fue hasta la década de los 70 en la que se popularizó el análisis de datos. En 1978 se crea Black-Scholes, un modelo matemático que permitía predecir el precio de acciones futuras. Con la llegada de Google en 1998 y el desarrollo de algoritmos para mejorar las búsquedas en la web, se produce el estallido de Big Data.

“Con la entrada del nuevo siglo, este concepto se acuña y recoge todo el significado que se le otorga en la actualidad. Según los analistas, hoy en día se generan 2,5 trillones de bytes relaciones con el Big Data”. Además, cada vez son

---

<sup>13</sup> CENALTIC, Op.cit.

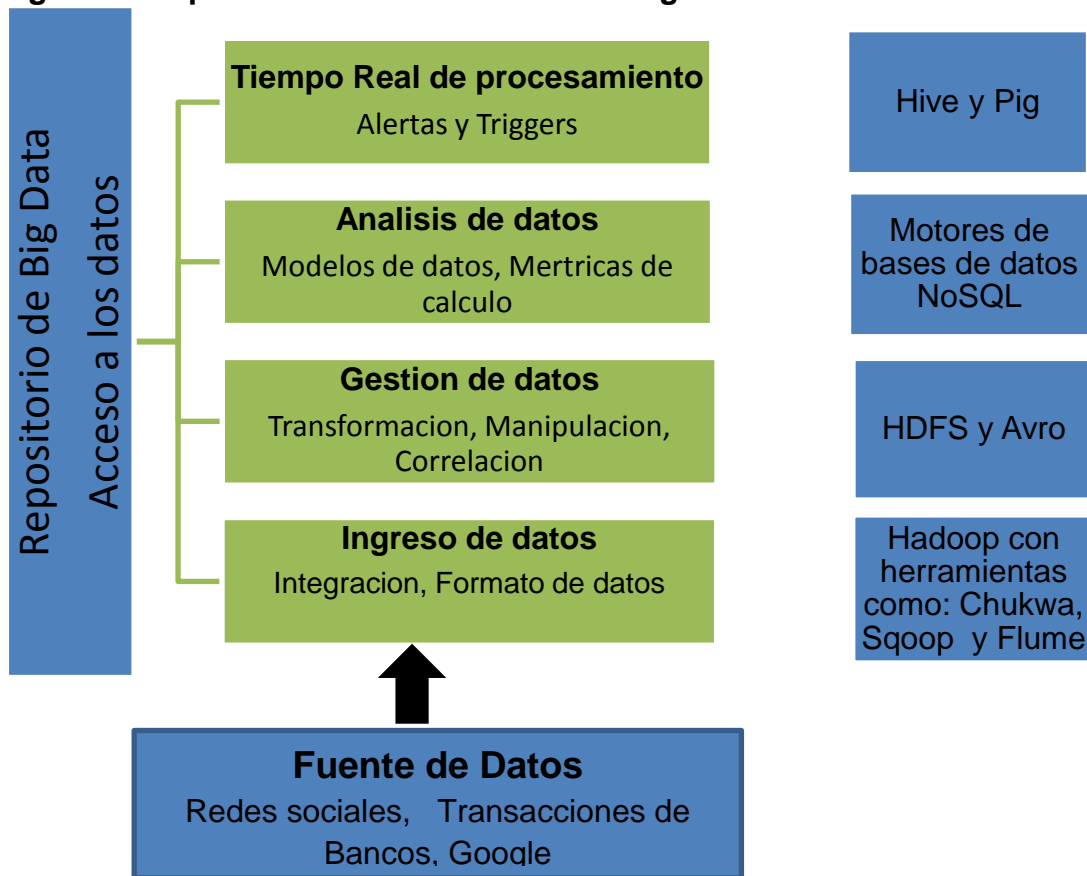
<sup>14</sup> HP. La era del Big Data [en línea]. California: Sara Artaza [citado 29 septiembre, 2013]. Disponible en internet: <<http://h30499.www3.hp.com/t5/Infraestructura-Convergente-de/La-Era-del-Big-Data/ba-p/6151357#.UkiHs4ZLMvL/>>

<sup>15</sup> Ibid.

más demandados aquellos perfiles profesionales que sean capaces de gestionar herramientas de análisis”<sup>16</sup>.

**1.6.3 Arquitectura de Big Data.** La gestión y procesamiento de Big Data es un problema abierto y vigente que puede ser manejado con el diseño de una arquitectura de 5 niveles, la cual está basada en el análisis de la información y en el proceso que realizan los datos para el desarrollo normal de las transacciones. A continuación se pueden ver los niveles que contienen un ambiente Big Data y la forma en que se relacionan e interactúan entre ellos:

**Figura 1. Arquitectura de un ambiente de big data**



Fuente: UNIVERSITY OF AMSTERDAM. Defining the Big Data Architecture Framework (BDAF) [en línea]. Ámsterdam: Yuri Demchenko [citado 22 Septiembre, 2013]. Disponible en internet: <[http://bigdatawg.nist.gov/\\_uploadfiles/M0055\\_v1\\_7606723276.pdf/](http://bigdatawg.nist.gov/_uploadfiles/M0055_v1_7606723276.pdf/)>.

<sup>16</sup> Ibid.

**1.6.3.1 Ingreso de datos.** “El Ingreso de datos es el procedimiento de obtener e importar información para su posterior uso o almacenamiento en una base de datos. Consiste en coleccionar datos de muchas fuentes con el objetivo de realizar un análisis basado en modelos de programación”<sup>17</sup>.

**1.6.3.2 Gestión de datos.** La administración de datos es el desarrollo y ejecución de arquitecturas, políticas, prácticas y procedimientos con el fin de gestionar las necesidades del ciclo de vida de información de una empresa de una manera eficaz. Es un enfoque para administrar el flujo de datos de un sistema a través de su ciclo de vida, desde su creación hasta el momento en que sean eliminados. La administración de Big data es la forma en que se organizan y gestionan grandes cantidades de datos, tanto de información estructurada como no estructurada para desarrollar estrategias con el fin de ayudar con los conjuntos de datos que crecen rápidamente, donde se ven involucrados terabytes y hasta peta bytes de información con variedad de tipos<sup>18</sup>.

**1.6.3.3 Tiempo real de procesamiento.** Según Halim<sup>19</sup>, es un proceso que automatiza e incorpora el flujo de datos en la toma de decisiones, este aprovecha el movimiento de los datos para acceder a la información estática y así lograr responder preguntas a través de análisis dinámicos. Los sistemas de procesamiento de flujo se han construido con un modelo centrado que funciona con datos estructurados tradicionales, así como en aplicaciones no estructuradas, como vídeo e imágenes.

“El procesamiento de flujos es adecuado para aplicaciones que tiene tres características: calcular la intensidad (alta proporción de operaciones de E/S), permitir paralelismo de datos y por último la capacidad de aplicar los datos que se introducen de forma continua”<sup>20</sup>.

**1.6.3.4 Análisis de datos.** “Es el proceso de examinar grandes cantidades de datos para descubrir patrones ocultos, correlaciones desconocidas y otra información útil”<sup>21</sup>. Esta información puede proporcionar ventajas competitivas y resultar en beneficios para el negocio, como el marketing para generar mayores ingresos.

---

<sup>17</sup> TECHTARGET. Data Ingestion [en línea]. Massachusetts: Margaret Rouse [citado 29 Septiembre, 2013]. Disponible en internet: <<http://whatis.techtarget.com/definition/data-ingestion>>

<sup>18</sup> TECHTARGET. Data management [en línea]. Massachusetts: Margaret Rouse [citado 29 Septiembre, 2013]. Disponible en internet: <<http://searchdatamanagement.techtarget.com/definition/data-management>>

<sup>19</sup> IBM. Stream processing [en línea]. New York: Nagui Halim [citado 29 de Septiembre de 2013]. Disponible en internet : <<http://www.ibm.com/smarter-computing/us/en/technical-breakthroughs/stream-processing.html/>>

<sup>20</sup> Ibid.

<sup>21</sup> TECHTARGET. Análisis de “big data” [en línea]. Massachusetts: Margaret Rouse [citado 29 de Septiembre de 2013]. Disponible en internet: <<http://searchdatacenter.techtarget.com/es/definicion/Analisis-de-big-data/>>

Según Rouse<sup>22</sup> el objetivo principal del análisis de datos es ayudar a las empresas a tomar mejores decisiones de negocios al permitir a los científicos y otros usuarios de la información analizar grandes volúmenes de datos transaccionales, así como otras fuentes de datos que puedan haber quedado sin explotar por la inteligencia del negocio convencional.

**1.6.4 Bases de datos NoSQL.** Con la aparición del término NoSQL en los 90's y su primer uso en el 2009 por Eric Vans<sup>23</sup>, se pretende dar una solución a las problemáticas planteadas anteriormente, dando una posibilidad de abordar la forma de gestionar la información de una manera distinta a como se venía realizando.

Para dar una definición adecuada de las bases de datos NoSQL se puede tener en cuenta las siguientes características:

- **Distribuido.** Sistemas de bases de datos NoSQL son a menudo distribuidos donde varias máquinas cooperan en grupos para ofrecer a los clientes datos. Cada pieza de los datos se replica normalmente durante varias máquinas para la redundancia y alta disponibilidad.
- **Escalabilidad horizontal.** A menudo se pueden añadir nodos de forma dinámica, sin ningún tiempo de inactividad, lo que los efectos lineales de almacenamiento logran capacidades de procesamiento general.
- **Construido para grandes volúmenes.** Muchos sistemas NoSQL fueron contruidos para ser capaz de almacenar y procesar enormes cantidades de datos de forma rápida.
- **Modelos de datos no relacionales.** Los modelos de datos varían, pero en general, no son relacional. Por lo general, permiten estructuras más complejas y no son tan rígida que el modelo relacional.
- **No hay definiciones de esquema.** La estructura de los datos generalmente no se definen a través de esquemas explícitos que la base de datos manejan. En su lugar, los clientes almacenan datos como deseen, sin tener que cumplir con algunas estructuras predefinidas<sup>24</sup>.

Dentro de las bases de datos NoSQL se pueden encontrar 4 categorías de acuerdo con la taxonomía propuesta por Scofield y Popescu<sup>25</sup>:

---

<sup>22</sup> Ibid.

<sup>23</sup> SG. NoSQL la evolución de las bases de datos [en línea]. Cataluña: Erik Camacho [citado 08 Agosto, 2013]. Disponible en internet: <<http://sg.com.mx/content/view/966>>

<sup>24</sup> NASHOLM, Petter. Extraer datos de bases de datos NoSQL. 1 ed. Gotemburgo, 2012. p.10.

<sup>25</sup> STRAUCH, Christof. NoSQL Databases. 1 ed. New York, 2011. p.26.

**1.6.4.1 Almacenes Key-Value.** “Estas son las bases de datos más simples en cuanto su uso (la implementación puede ser muy complicada), ya que simplemente almacena valores identificados por una clave. Normalmente, el valor guardado se almacena como un arreglo de bytes (BLOB) y es todo. De esta forma el tipo de contenido no es importante para la base de datos, solo la clave y el valor que tiene asociado”<sup>26</sup>.

**1.6.4.2 Bases de datos columnares.** Estas bases de datos “guardan la información en columnas en lugar de renglones, con esto se logra una mayor velocidad en realizar la consulta”<sup>27</sup>. Esta solución es conveniente en ambientes donde se presenten muchas lecturas como en data warehouses y sistemas de Business Intelligence.

**1.6.4.3 Bases de datos orientadas a documentos.** Según Camacho<sup>28</sup>, son como un almacén Key-Value, a diferencia que la información no se guarda en binario, sino como un formato que la base de datos pueda leer, como XML, permitiendo realizar consultas avanzadas sobre los datos almacenados.

**1.6.4.4 Bases de datos orientados a grafos.** “Estas bases de datos manejan la información en forma de grafo, dando una mayor importancia a la relación que tiene los datos”<sup>29</sup>. Con esto se logra que las consultas puedan ser logradas de formas más óptimas que en un modelo relacional.

A continuación se presenta un cuadro comparativo sobre el desempeño de las taxonomías descritas anteriormente.

**Figura 2. Comparación taxonomía**

	Performance	Scalability	Flexibility	Complexity	Functionality
Key-Value Stores	high	high	high	none	variable (none)
Column stores	high	high	moderate	low	minimal
Document stores	high	variable (high)	high	low	variable (low)
Graph databases	variable	variable	high	high	graph theory
Relational databases	variable	variable	low	moderate	relational algebra

Fuente: STRAUCH, Christof. NoSQL Databases. 1 ed. New York, 2011. p.26.

Se puede ver que las bases de datos relacionales dan más importancia a la consistencia y a la disponibilidad, en contraste con las NoSQL, que “dan mayor

<sup>26</sup> IBM. ¿Qué es big data?, Op.cit

<sup>27</sup> Ibid.

<sup>28</sup> Ibid.

<sup>29</sup> Ibid.



prioridad a la tolerancia y en ocasiones a la disponibilidad”<sup>30</sup>. Estos aspectos se definen en el teorema CAP.

**1.6.4.5 Teorema CAP.** El teorema explica tres requerimientos para tener en cuenta en un sistema distribuido, los cuales son:

- **Consistencia:** se refiere a la integridad de la información. Todos los nodos del sistema ven la misma información en todo momento.
- **Disponibilidad:** la aplicación debe estar siempre disponible, si falla algún nodo los demás pueden seguir operando sin inconvenientes.
- **Tolerancia al particionamiento:** el sistema continúa funcionando a pesar de que se pierdan mensajes.

El teorema CAP establece que es imposible que un sistema satisfaga los tres requerimientos al mismo tiempo, por lo cual se debe elegir dos y enfocarse en ellos<sup>31</sup>.

Para aclarar las diferencias que pueden existir dentro de las bases de datos relaciones y las NoSQL, se debe comprender dos modelos importantes: Base y Acid, los cuales se utilizan respectivamente para el manejo de las transacciones.

**1.6.4.6 Base vs Acid.** Según Cook<sup>32</sup> la característica clave de los sistemas basados en NoSQL, es que nada debe estar compartido, para cumplir el propósito de replicar y particionar los datos sobre muchos servidores, ya que esto permite soportar un gran número de operaciones de escritura y lectura por segundo, esto conlleva a que los sistemas NoSQL generalmente no proveen las propiedades transaccionales del modelo ACID las cuales son atomicidad, consistencia, aislamiento y durabilidad, pero si pueden proveer su propio modelo el cual es llamado BASE por sus propiedades donde las transacciones deben ser básicamente disponibles , eventualmente consistentes y de estado suave.

**Tabla 2. Comparación de propiedades Base y Acid**

<b>BASE</b>	<b>ACID</b>
Basically Available: esto quiere decir básicamente disponible, aquí se utiliza la replicación para reducir la probabilidad de que se presente la indisponibilidad de los datos o la fragmentación de la información a través de los distintos servidores de almacenamiento.	Atomicidad: toda la secuencia de acciones debe ser completada o abortada, es decir que la operación no puede ser parcialmente completada, debe haber una operación de commit cuando se completa o una de rollback en el momento en que la transacción no pueda ser completada.

<sup>30</sup> Ibid.

<sup>31</sup> Ibid.

<sup>32</sup> THE ENDEAVOUR. ACID versus BASE for database transactions [en línea]. Houston: John Cook [citado 10 septiembre, 2013]. Disponible en internet: <<http://www.johndcook.com/blog/2009/07/06/brewer-cap-theorem-base/>>

Tabla 2. (Continuación)

<p>Soft state: mientras los sistemas que utilizan el acrónimo ACID, en las bases de datos relacionales asumen la consistencia de la información como requerimiento de alta prioridad, los sistemas NoSQL permiten que los datos sean inconsistentes y relegan el diseño alrededor las inconsistencias de los desarrolladores de la aplicación.</p>	<p>Consistencia: esto quiere decir que la transacción toma los recursos de un estado valido para llevar a la base de datos a otro estado valido.</p>
<p>Eventualmente Consistente: Aunque las aplicaciones deben asumir la consistencia como un requerimiento de alta prioridad, “los sistemas NoSQL garantizan que en algún punto del futuro, los datos asumen un estado de consistencia, haciendo comparación con los sistemas relacionales que obligan que una transacción sea completada”<sup>33</sup>, los sistemas NoSQL brindan consistencia a lo largo del tiempo.</p>	<p>Aislamiento: el impacto de una transacción no debe afectar en el cumplimiento de otras transacciones.</p>
	<p>Durabilidad: todas las transacciones realizadas deben ser permanentes y ser tolerantes a los fallos del sistema.</p>

Fuente: Autores

A continuación se realiza un cuadro comparativo sobre algunos motores de bases de datos NoSQL, que se pueden utilizar para la construcción de un ambiente Big Data, teniendo en cuenta su taxonomía.

<sup>33</sup> Oracle. Oracle NoSQL Databases. 1 ed. California. p.4.

**Tabla 3. Bases de datos NoSQL orientada Key-Value**

<b>Motores Criterios</b>	<b>Redis</b>	<b>Riak</b>	<b>Dynamo</b>	<b>Scalaris</b>
¿Qué es?	Motor de base de datos en memoria, basado en el almacenamiento en tablas de hashes (llave, valor)	Es una, base de datos NoSQL de la aplicación de los principios de Amazon Dynamo	Bases de datos NoSQL que proporciona un rendimiento rápido y fiable con una perfecta escalabilidad	Es un almacén de claves-valor, transaccional distribuido y escalable. Fue la primera base de datos NoSQL, que apoyó las propiedades ACID
Versión actual	Versión 2.4	Versión 1.2	Beta	Versión 0.5
Plataforma operativa	Unix, Linux, Solaris, OS/X, no existe soporte oficial para Windows	Linux, BSD, Mac, OS X, Solaris	Multiplataforma	Linux, Os X
Almacenamiento	Tablas de hashes	Fragmento particiones	Atributos multi- valuados	Múltiples claves
Tipo de índices	Geoespacial	Índices Secundarios y claves compuestas	Índices Secundarios	Índices secundarios
Esquema de replicación y distribución	Maestro-esclavo	Replicación multi-master	Maestro esclavo	Replicación multi-master
Lenguaje de consulta	API Lua	JavaScript REST Erlang	API	API JSON
Herramientas con las que se integra	ActionScript, Clojure, Erlang, Go, Haskell, Javascript, PHP, Python, Ruby	Erlang, HTTP API, PBD API	SDK AWS, CloudWatch	Servidor Web Frambesia
Tipo Licencia	Licencia BSD: software de código abierto	Apache	Propietaria	Apache
Lenguaje creación	C/C++	Erlang y C, Javascript	Java	Erlang
Creado por	Salvatore Sanfilippo and Pieter Noordhuis	Apache	Amazon	Instituto Zuse de Berlín
Protocolo	Telnet-like	HTTP/REST	HTTP/REST	JSON-RPC
Características	Tiene sistemas get/set, incrementos y decrementos de números, operaciones de listas y de conjuntos	Utilizado como una base de datos gráfica de alta escalabilidad, disponibilidad y tolerancia a fallos	No presenta esquemas fijos, y cada elemento puede tener un número diferente de atributos	En un sistema basado en Erlang realizando operaciones de escritura consistentes y distribuidas

Tabla 3. (Continuación)

Utilidad	Para la gestión de sesiones de usuarios y soluciones de cache, también en mensajería instantánea.	Para desarrolladores de juegos web y móvil.	Para la publicidad digital, juegos sociales y aplicaciones de dispositivos conectados	Para la gestión de archivos en Python y Ruby
----------	---	---	---	--

Fuente: Autores

Tabla 4. Bases de datos NoSQL orientadas a documentos

<b>Motores Criterios</b>	<b>CouchDB</b>	<b>MongoDB</b>	<b>Base X</b>	<b>eXist</b>
¿Qué es?	Base de datos que abarcan completamente la web	Bases de datos NoSQL orientada a objetos más avanzada y flexible	Es un sistema de gestión de base de datos nativo y ligero sobre XML y XQuery,	Es sistema de gestión de base de datos de código abierto construida enteramente sobre tecnología XML
Versión actual	Versión 1.2	Versión 2.2	Versión 7.7	Versión 2.0
Plataforma operativa	Windows, Mac y Linux	Windows, Linux, OS X y Solaris	Multiplataforma	Multiplataforma
Almacenamiento	B-tree	BSON y JSON	documentos XML y colecciones	Documentos XML
Tipo de índices	Índices secundario y geoespacial	Índices secundario y geoespacial	Búsqueda de texto	Índices secundarios
Esquema de replicación y distribución	Replicación multi-master	Maestro-esclavo	No tiene modelo de replicación	Maestro-esclavo
Lenguaje de consulta	JavaScript REST Erlang	API JavaScript REST	XQuery	XQuery
Tipo Licencia	Apache License 2.0	AGPL(drivers: Apache)	BSD	GNU LGPL
Lenguaje creación	Lenguaje Erlang	C++	Java	Java
Creado por	Apache License 2.0	10gen	Cristian Grün	Wolfgang Meier
Protocolo	HTTP/REST	Custom, binary(BSON)	XML-RPC	XML-RPC

Tabla 4. (Continuación)

Características	Bidireccionamiento, Vistas: incrustado MapReduce, autenticación posible	Maestro/esclavo de replicación, las consultas son expresiones Javascript, tiene indexación geoespacial	Utiliza una representación tabular de estructuras de árbol XML. La base de datos actúa como un contenedor para un solo documento o una colección de documentos	Sigue estándares W3C XML, como XQuery . Soporta REST interfaces para interactuar con AJAX formularios web
Utilidad	Aplicaciones web altamente concurrentes como los juegos en línea y sistemas CMS y CRM	Es ideal para aplicaciones con estructuras complejas como blogs (post, comentarios, rollbacks, etc) o aplicaciones de analítica (Google analytics).	Para hacer seguimiento de colecciones de objetos, también para reducir informes financieros. Gestión de auditoría, control de calidad y datos de producción.	Ideal para el sector editorial y de los medios de comunicación y para el desarrollo web

Fuente: Autores

Tabla 5. Bases de datos NoSQL orientadas a columnas

<b>Motores</b> <b>Crterios</b>	<b>Cassandra</b>	<b>Hbase</b>	<b>Hypertable</b>	<b>Jackrabbit</b>
¿Qué es?	Base de datos Apache que brinda escalabilidad y alta disponibilidad sin comprometer el rendimiento	Es una bases de datos distribuida de fuente abierta, modelada después de Google BigTable	Base de datos de código abierto escalable, similar al modelo de BigTable, propiedad de Google	Es un repositorio de código abierto de contenido para la plataforma Java
Versión actual	Versión 1.1.5	Versión 0.94	Versión 0.96	Versión 2.6
Plataforma operativa	Multiplataforma	Multiplataforma	Multiplataforma	Multiplataforma
Tipo de índices	Índice secundario	Claves compuestas	Índice secundario	Búsqueda de texto
Esquema de replicación y distribución	Maestro-esclavo	Maestro-esclavo	Maestro-esclavo	No tiene modelo de replicación
Lenguaje de consulta	API CQL	API REST XML	API	API
Herramientas con las que se integra	Facebook, Twitter, dig, rockspace	Facebook	Baidu, Rediff.com o Zvents	Facebook

Tabla 5. (Continuación)

Tipo Licencia	Apache License 2.0	Apache License 2.0	GPL 2.0	Apache License 2.0
Lenguaje creación	JavaScript	Java	C++	Java
Creado por	Apache	Apache	Zvents	Apache
Protocolo	Thrift & Custom binary CQL3	HTTP/REST	Thrift, C++ library, orHQL shell	HTTP/REST
Características	Consulta por columna, ajustable para la distribución y la reproducción, compatibilidad con múltiples centros de datos de replicación	Utiliza Hadoop HDFS como almacenamiento, optimización de las consultas en tiempo real, se compone de varios tipos de nodos	Implementa diseño de BigTable de Google, la búsqueda se puede limitar a intervalos de clave/columna. Conserva los últimos valores históricos	Acceso al contenido fino y de grano grueso. Contenidos jerárquica, contenido estructurado, propiedades binarias consultas XPath, consultas SQL
Utilidad	Diseñado para aplicaciones en la industria financiera	Para realizar consultas rápidas de forma interactiva sobre un conjunto de datos definido	ideal para aplicaciones que necesitan soportar una gran demanda de datos en tiempo real	Para transacciones, flujos BPM y en sistemas de planificación de recursos empresariales

Fuente: Autores

Tabla 6. Bases de datos NoSQL orientadas a grafos

Motores Criterios	Neo4j	DEX	HyperGraphDB	AllegroGraph
¿Qué es?	Es una base de datos enfocada a grafos, debido su modelo grafico es muy ágil y rápido para las operaciones de datos	Es una base de datos orientada a grafos que permite analizar grandes volúmenes de datos	Es una base de datos de gráficos, diseñada específicamente para la inteligencia artificial y los proyectos web de semántica	Base de datos gráfica moderna, utiliza memoria en combinación con el almacenamiento basado en disco
Versión actual	Versión 1.5	Versión 4.8	Versión 1.2	Versión 4.1
Plataforma operativa	Multiplataforma	Multiplataforma	Unix, Linux Windows, Mac	Amazon EC2, Linux
Almacenamiento	Local	Memoria volátil	Nodos	Nodos local
Tipo de índices	Índice secundario	No posee índices	No posee índices	geoespacial
Esquema de replicación y distribución	No tiene modelo de replicación	No tiene modelo de replicación	No tiene modelo de replicación	No tiene modelo de replicación

Tabla 6. (Continuación)

Lenguaje de consulta	Tinkerpop Gremlin Cypher	API	API	SparQL
Herramientas con las que se integra	PHP, JavaScript, Ruby	Facebook, Twitter	JSON, XML	SOLR y MongoDB, Python
Tipo Licencia	GPL	Uso personal / Uso comercial	LGPL	Propietaria
Lenguaje creación	Java	C++	Java	Java, C#
Creado por	Neo Technology	Sparsity-Technologies		Franz Inc.
Protocolo	HTTP/REST	HTTP/REST	HTTP/REST	HTTP/REST
Características	Optimizado para lecturas, Operaciones en el Api de Java, Indexación de nodos y relaciones	Capacidad de almacenamiento de datos y rendimiento, con órdenes de magnitud de miles de millones de nodos, aristas y atributos	Gráfico orientado de almacenamiento. Recorridos por caminos y consultas de tipo relacional. Indexación personalizable	Commit, Rollback y puntos de control, basado en multiprocesamiento, reducción de paginación, un mejor rendimiento
Utilidad	Para buscar rutas en las relaciones sociales, transporte público, mapas de carreteras, o topologías de red.	Redes de seguridad y detección de fraudes, También en redes físicas como transporte y electricidad	En aplicaciones Java del lado del servidor, en bioinformática, en redes de investigación	Se utiliza en bioinformática, en agentes inteligentes y web semántica

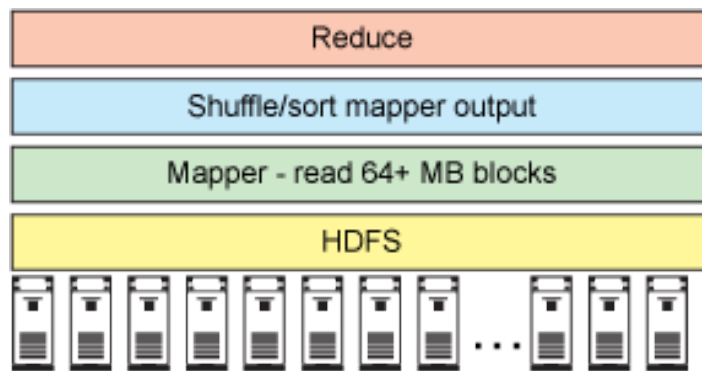
Fuente: Autores

Dentro de la arquitectura de un ambiente Big Data se pueden utilizar diferentes herramientas, cada una de estas cumple un papel importante para la implementación. A continuación se realiza una descripción de cada una de las tecnologías que están incluidas en la arquitectura.

**1.6.5 Hadoop.** “Es un framework que permite el procesamiento distribuido de grandes conjuntos de datos a través de grupos de ordenadores que utilizan modelos de programación simple. Está diseñado para detectar y controlar los errores en la capa de aplicación”<sup>34</sup>.

Apache Hadoop tiene dos componentes centrales, el almacenamiento de archivos llamado Hadoop Distributed File System (HDFS), y la infraestructura de programación llamada MapReduce, como se muestra en la Figura N° 3.

**Figura 3. Arquitectura hadoop**



Fuente: DEVELOPERWORKS. Big data de código abierto [en línea]. México D.F.: Marty Lurie [citado 25 septiembre, 2013]. Disponible en internet: <<http://www.ibm.com/developerworks/ssa/data/library/techarticle/dm-209hadoopbigdata/>>

**1.6.6 MapReduce.** “El modelo de programación MapReduce se basa en dos funciones llamadas Map y Reduce. La entrada a dicho modelo es un conjunto de pares clave/valor y la salida es otro conjunto de pares clave/valor”<sup>35</sup>.

- **Función Map.** A partir del conjunto de pares clave/valor de entrada se genera un conjunto de datos intermedios. La función Map asocia claves idénticas al mismo grupo de datos intermedios. Cada grupo de datos intermedios estará formado por una clave y un conjunto de valores por lo tanto, estos datos intermedios van a ser a su vez la entrada de la función de Reduce.

<sup>34</sup> APACHE. Apache Hadoop [en línea]. Los Ángeles: Apache Software Foundation [citado 25 septiembre, 2013]. Disponible en internet: <<http://hadoop.apache.org/>>

<sup>35</sup> JAVA4DEVELOPERS. Introducción a Big Data y Hadoop [en línea]. Madrid: Java4Developers [citado 25 septiembre, 2013]. Disponible en internet: <<http://java4developers.com/2013/introduccion-a-big-data-y-hadoop/>>

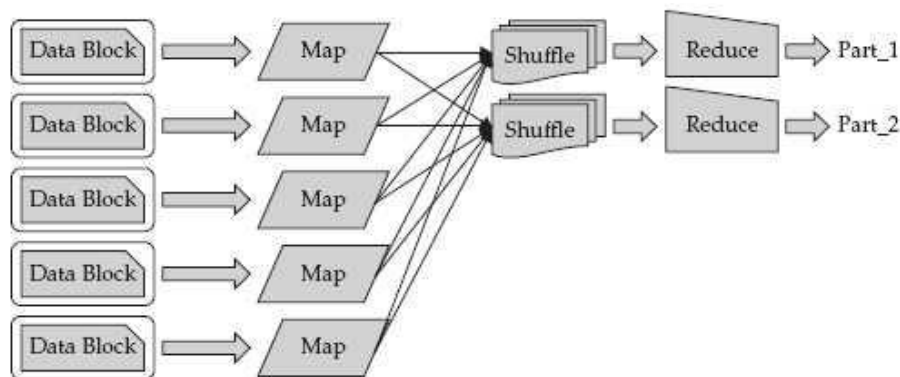


- **Función Reduce.** La fase de Reduce se encargará de manipular y combinar los datos provenientes de la fase anterior para producir a su vez un resultado formado por otro conjunto de claves/valores<sup>36</sup>.

“Una fase intermedia es la denominada Shuffle la cual obtiene las tuplas del proceso Map y determina que nodo procesará estos datos dirigiendo la salida a una tarea Reduce en específico”<sup>37</sup>.

La siguiente figura da un ejemplo del flujo de datos en un proceso sencillo de MapReduce.

**Figura 4. Proceso MapReduce**



Fuente: IBM. ¿Qué es Big Data? [en línea]. México D.F.: Ricardo Barranco Fragoso [citado 2 Octubre, 2013]. Disponible en internet: <<http://www.ibm.com/developerworks/ssa/local/im/que-es-big-data/index.html>>

**1.6.7 HDFS.** “Es el sistema de ficheros distribuido utilizado por Hadoop”<sup>38</sup>. Por lo tanto está especialmente diseñado para cumplir con las necesidades propias de Hadoop. Las dos ideas principales de HDFS es por un lado que sea un sistema de ficheros que facilite una alta escalabilidad tolerante a fallos. Por otro lado Hadoop necesita que los problemas que se estén intentando solucionar involucren un gran número de datos. HDFS debe garantizar un alto rendimiento de datos para que Hadoop sea capaz de procesar.

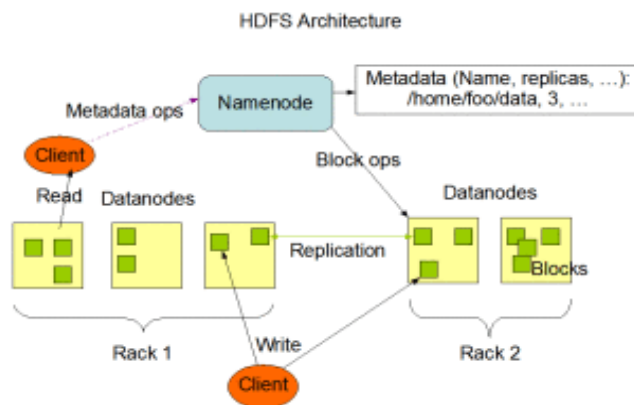
A continuación se puede observar la arquitectura que presenta HDFS con sus respectivos componentes:

<sup>36</sup> JAVA4DEVELOPERS, Op.cit.

<sup>37</sup> IBM. ¿Qué es big data?, Op.cit.

<sup>38</sup> JAVA4DEVELOPERS, Op.cit.

**Figura 5. Arquitectura HDFS**



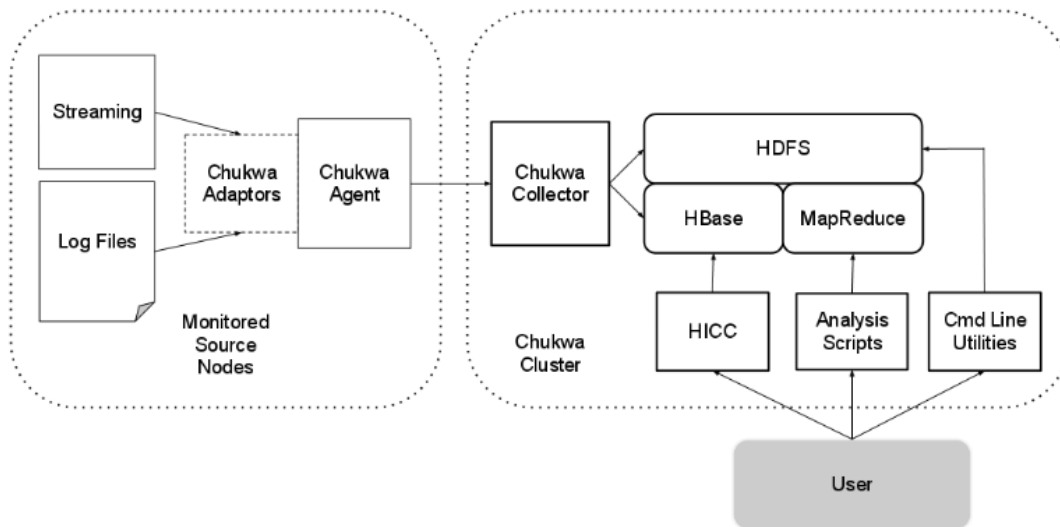
Fuente: JAVA4DEVELOPERS. Introducción a Big Data y Hadoop [en línea]. Madrid: Java4Developers [citado 25 de septiembre, 2013]. Disponible en internet: <<http://java4developers.com/2013/introduccion-a-big-data-y-hadoop/>>

**1.6.8 Chukwa.** “Es un sistema de recopilación de datos de código abierto para el seguimiento de grandes sistemas distribuidos. Se construye en la parte superior del sistema de archivos distribuido Hadoop (HDFS) y Map/Reduce. Chukwa también incluye un conjunto de herramientas flexibles para la visualización, seguimiento y análisis de resultados de los datos recogidos”<sup>39</sup>.

La siguiente figura muestra la interacción de Chukwa con las otras tecnologías y la función que tiene dentro de los procesos de datos.

<sup>39</sup> APACHE. Apache Chukwa [en línea]. Los Ángeles: Apache Software Foundation [citado 09 octubre, 2013]. Disponible en internet: <<http://incubator.apache.org/chukwa/index.html>>

**Figura 6. Diagrama de Chukwa**



Fuente: APACHE. Chukwa Administration Guide [en línea]. Los Ángeles: Apache Software Foundation [citado 09 octubre, 2013]. Disponible en internet: <<http://incubator.apache.org/chukwa/docs/r0.5.0/admin.html/>>

**1.6.9 Sqoop.** “Es una herramienta diseñada para transferir datos entre Hadoop y bases de datos relacionales. Sqoop importa los datos de un sistema de gestión de bases de datos relacionales (RDBMS) como MySQL u Oracle al sistema de archivos distribuido Hadoop (HDFS), donde transforma los datos y luego los exporta de nuevo a un RDBMS”<sup>40</sup>.

“Sqoop automatiza la mayor parte de este proceso, basándose en la base de datos para describir el esquema de importación de los datos. Sqoop utiliza MapReduce para importar y exportar los datos, lo que proporciona el funcionamiento en paralelo, así como tolerancia a fallos”<sup>41</sup>.

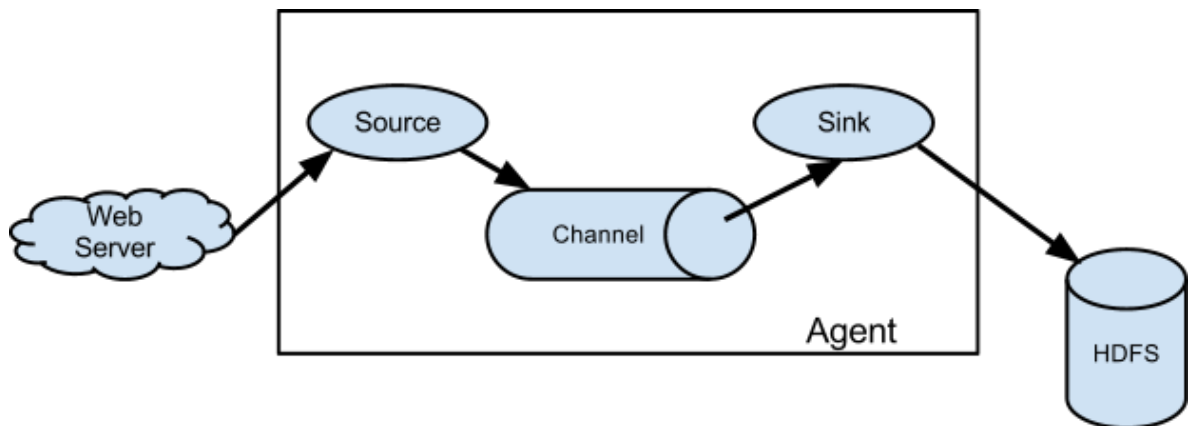
**1.6.10 Flume.** “Es un servicio distribuido, confiable y disponible para recolectar, agregar y mover grandes cantidades de datos de registro eficientemente. Cuenta con una arquitectura simple y flexible basada en transmisión de flujos de datos. Es robusto y tolerante a fallos con los mecanismos de fiabilidad, conmutación por error y los mecanismos de recuperación. Se utiliza un modelo de datos extensible simple que permite una aplicación analítica en línea”<sup>42</sup>, el cual se puede ver en la siguiente figura.

<sup>40</sup> APACHE. Apache Sqoop [en línea]. Los Ángeles: Apache Software Foundation [citado 09 octubre, 2013]. Disponible en internet: <[http://sqoop.apache.org/docs/1.4.2/SqoopUserGuide.html#\\_introduction](http://sqoop.apache.org/docs/1.4.2/SqoopUserGuide.html#_introduction)>

<sup>41</sup> Ibid.

<sup>42</sup> APACHE. Apache Flume [en línea]. Los Ángeles: Apache Software Foundation [citado 09 octubre, 2013]. Disponible en internet: <<http://flume.apache.org/>>

**Figura 7. Flujo de datos**



Fuente: APACHE. Apache Flume [en línea]. Los Ángeles: Apache Software Foundation [citado 09 octubre, 2013]. Disponible en internet: <<http://flume.apache.org/>>

**1.6.11 Avro.** Es un sistema de serialización de datos; contiene lo siguiente:

- Estructuras de datos.
- Formato de datos binario.
- Un archivo contenedor para almacenar datos persistentes.
- Llamada a procedimiento remoto (RPC).
- Fácil integración con lenguajes dinámicos. La generación de código no está obligado a leer y escribir archivos de datos ni a utilizar o implementar protocolos RPC.

Cuando los datos de Avro se almacenan en un archivo, su esquema se almacena con él, para que los archivos se puedan procesar posteriormente con cualquier programa. Si el programa de lectura de los datos espera un esquema diferente esto puede ser fácilmente resuelto, ya que ambos esquemas están presentes<sup>43</sup>.

**1.6.12 Hive.** “Es la infraestructura de almacenamiento de datos construida sobre Apache Hadoop para proporcionar el resumen de datos, consultas ad-hoc y análisis de grandes conjuntos de datos. Proporciona un mecanismo para proyectar en la estructura de los datos en Hadoop y consultar los datos utilizando un

<sup>43</sup> APACHE. Apache Avro [en línea]. Los Ángeles: Apache Software Foundation [citado 09 octubre, 2013]. Disponible en internet: <<http://avro.apache.org/docs/current/>>

lenguaje similar a SQL llamado HiveQL (HQL)”<sup>44</sup>. Hive facilita la integración entre Hadoop y herramientas para la inteligencia de negocios y la visualización. Hive permite al usuario explorar y estructurar los datos, analizarlos y luego convertirla en conocimiento del negocio.

Estas son algunas de las características de Hive:

- Cientos de usuarios pueden consultar simultáneamente los datos utilizando un lenguaje familiar para los usuarios de SQL.
- Los tiempos de respuesta son típicamente mucho más rápido que otros tipos de consultas sobre el mismo tipo de conjuntos de datos.
- Controladores JDBC y ODBC, aplicaciones para extraer datos. Hive permite a los usuarios leer los datos en formatos arbitrarios, usando SerDes y entrada / Formatos de salida<sup>45</sup>.

**1.6.13 Pig.** “Es una plataforma para el análisis de grandes conjuntos de información que consiste en un lenguaje de alto nivel para la expresión de los programas de análisis de datos, junto con la infraestructura necesaria para la evaluación de estos programas. La propiedad más importante es que su estructura es susceptible de paralelismo, que a su vez les permite manejar grandes conjuntos de datos”<sup>46</sup>.

“En la actualidad, la capa de infraestructura de Pig se compone de un compilador que produce secuencias de programas de Mapa-Reduce, para el que ya existen implementaciones paralelas a gran escala (por ejemplo, el Hadoop subproyectos)”<sup>47</sup>.

Pig posee las siguientes características:

- **Facilidad de programación.** Es trivial para lograr la ejecución paralela de tareas de análisis simples. Las tareas complejas compuestas de múltiples transformaciones de datos relacionados entre sí, están codificados explícitamente como secuencias de flujo de datos, lo que hace que sean fáciles de escribir, entender y mantener.
- **Oportunidades de optimización.** La forma en que se codifican las tareas permite que el sistema pueda optimizar su ejecución de forma automática, lo que permite al usuario centrarse en la semántica en lugar de la eficiencia.

---

<sup>44</sup> HORTONWORKS. Apache Hive [en línea]. California: HortonWorks [citado 09 octubre, 2013]. Disponible en internet: <<http://hortonworks.com/hadoop/hive/>>

<sup>45</sup> Ibid.

<sup>46</sup> APACHE. Apache Pig [en línea]. Los Ángeles: Apache Software Foundation [citado 09 octubre, 2013]. Disponible en internet: <<http://pig.apache.org/>>

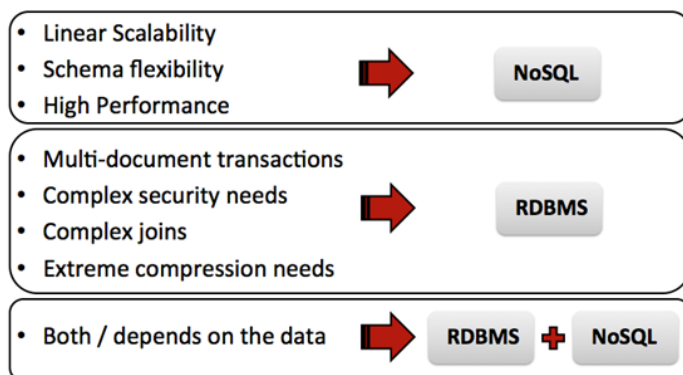
<sup>47</sup> Ibid.

- **Extensibilidad:** los usuarios pueden crear sus propias funciones para realizar el procesamiento de propósito especial<sup>48</sup>.

**1.6.14 Transformación de una base de datos relacional a NoSQL.** Según Couchbase<sup>49</sup> con la publicidad que ha tenido el concepto de una base de datos no relacional (NoSQL) que ha sido abrumadora, se tiene un concepto que se encuentra por debajo de las demandas que a menudo son exageradas, debido a que como la mayoría de cosas los beneficios tienen un costo. Un ejemplo de esto son los desarrolladores, que acostumbrados al modelamiento y desarrollo de datos con tecnología para bases de datos relacionales, tendrán que enfocarse de una manera diferente para realizar sus tareas. De esta manera se debe recalcar en por que es importante realizar la transición de un modelo relacional a un modelo NoSQL, teniendo en cuenta sus diferencias y las implicaciones que posee cada modelo para el desarrollo de una aplicación.

“Los cambios suelen ser difíciles y raramente son adoptados, más cuando se tiene un sistema establecido, pero cuando estos cambios sirven para dar solución y resolver nuevas necesidades presentadas, es necesario aceptarlos. En este caso la utilización de una tecnología NOSQL se debe a la necesidad de tener flexibilidad en una Base de datos tanto en el modelo de datos como en el modelo a escala”<sup>50</sup>.

**Figura 8. Aspectos para utilizar NoSQL o RDBMS**



Fuente: INFOQUEUE. Transitioning from rdbms to NoSQL [en línea]. California: Avram Abel [citado 26 septiembre, 2013]. Disponible en internet: <<http://www.infoq.com/articles/Transition-RDBMS-NoSQL>>.

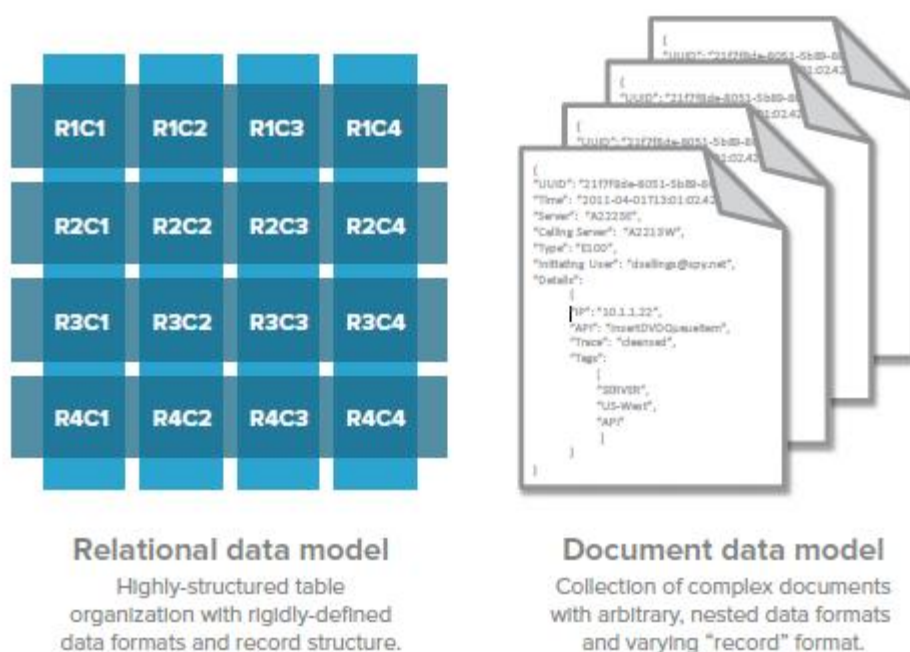
<sup>48</sup> Ibid.

<sup>49</sup> COUCHBASE. Making the shift from Relational to Nosql [en línea]. California: Couchbase.Inc [citado 26 septiembre, 2013]. Disponible en internet: <[http://www.couchbase.com/sites/default/files/uploads/all/whitepapers/Couchbase\\_Whitepaper\\_Transitioning\\_Relational\\_to\\_NoSQL.pdf](http://www.couchbase.com/sites/default/files/uploads/all/whitepapers/Couchbase_Whitepaper_Transitioning_Relational_to_NoSQL.pdf)>

<sup>50</sup> Ibid.

**1.6.14.1 El modelo relacional vs el modelo de datos orientado a documentos.** Los beneficios de una administración de datos sin ningún esquema es importante, ya que con una base de datos relacional se debe definir un esquema antes de agregar los registros a la base de datos y cada registro agregado debe cumplir con las especificaciones de este, cambiar este esquema particularmente cuando se tiene una base de datos relacional distribuida alrededor de muchos servidores es difícil, ya que si la captura de datos necesita evolucionar constantemente, un esquema rígido se vuelve un obstáculo para alcanzar este objetivo. Las bases de datos NoSQL son a escala ya siendo orientadas a key–value, documentos o columnas, por tanto no necesitan de un esquema previamente definido, haciendo que la prioridad sea el ingreso de datos en el momento en que la base de datos requiera evolucionar<sup>51</sup>.

**Figura 9. Comparación entre un modelo de datos relacional y uno orientado a documentos.**



Fuente: COUCHBASE. Making the shift from Relational to Nosql [en línea]. California: Couchbase.Inc [citado 26 septiembre, 2013]. Disponible en internet: <[http://www.couchbase.com/sites/default/files/uploads/all/whitepapers/Couchbase\\_Whitepaper\\_Transitioning\\_Relational\\_to\\_NoSQL.pdf](http://www.couchbase.com/sites/default/files/uploads/all/whitepapers/Couchbase_Whitepaper_Transitioning_Relational_to_NoSQL.pdf)>

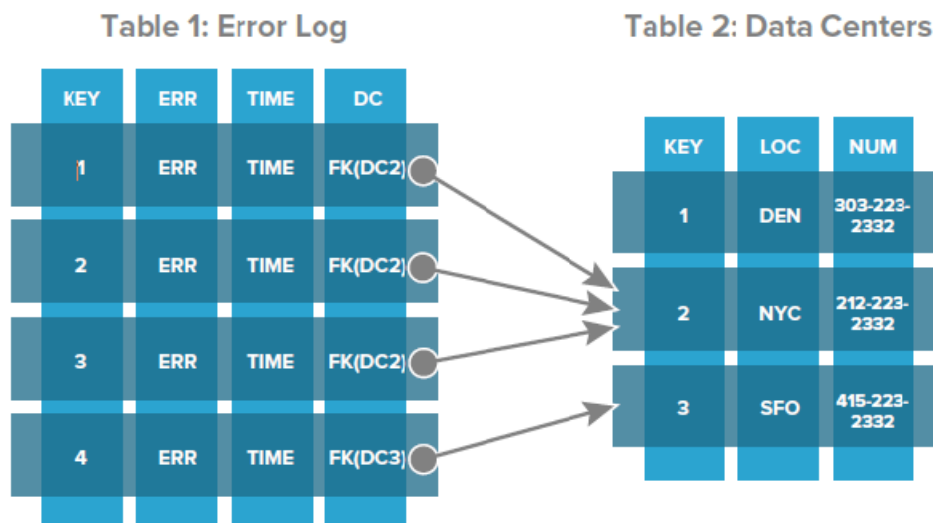
Como se muestra en la figura anterior, según Couchbase<sup>52</sup> cada registro en una base de datos relacional conforma un esquema según un número establecido de

<sup>51</sup> Ibid.

<sup>52</sup> Ibid.

campos o columnas, cada uno de estos especificando un tipo de dato y un propósito diferente, teniendo con esto que si en el futuro se requiere ingresar diferentes tipos de datos, se deberá replantear el esquema de la base de datos. Adicionalmente a eso un modelo de base de datos relacional se caracteriza por utilizar la “normalización” proceso donde las tablas más grandes son descompuestas en unas más pequeñas, como se muestra en la siguiente figura.

**Figura 10. Proceso de normalización de una base de datos relacional.**



Fuente: COUCHBASE. Making the shift from Relational to Nosql [en línea]. California: Couchbase.Inc [citado 26 septiembre, 2013]. Disponible en internet: <[http://www.couchbase.com/sites/default/files/uploads/all/whitepapers/Couchbase\\_Whitepaper\\_Transitioning\\_Relational\\_to\\_NoSQL.pdf/](http://www.couchbase.com/sites/default/files/uploads/all/whitepapers/Couchbase_Whitepaper_Transitioning_Relational_to_NoSQL.pdf/)>

Según Couchbase<sup>53</sup> la figura anterior, la base de datos es usada para guardar información de los errores presentados, cada registro de error consiste en un numero de error (ERR), el tiempo en el que ocurrió el error (TIME) y el centro de datos donde ocurrió el error, en vez de repetir toda la información de los centros de datos en cada registro como: ubicación y número telefónico, cada registro de error apuntara a una fila de la tabla Data centers (ver figura N° 10, tabla 2) la cual ya incluye todo el número y ubicación del centro de datos.

En una base de datos relacional, los registros son listados a través de múltiples tablas con algunos datos compartidos por medio de otros registros, donde en este caso varios registros de error comparten la misma información del centro de datos, dando como ventaja que serían menos datos duplicados dentro de la base de datos, pero teniendo como

<sup>53</sup> Ibid



desventaja que si se realiza un solo cambio en uno de los registros de la base de datos, varias tablas pueden quedar bloqueadas en nuestra base de datos solo para garantizar que esta no quede en un estado de inconsistencia<sup>54</sup>.

“El uso del término documento puede ser confuso, ya que una base de datos orientada a documentos realmente no tiene nada que ver con un documento literalmente, ya que un documento en este caso se refiere a un registro que se auto describe con los datos y elementos que contiene, ejemplos de esto pueden ser los documentos XML, documentos HTML, y los documentos JSON”<sup>55</sup>.

Tomando como referencia el motor de base de datos CouchBase, los registros de error en este caso se verían de la siguiente manera:

**Figura 11. Ejemplo de formato de registro en CouchBase**

```
{
  "ID": 1,
  "ERR": "Out of Memory",
  "TIME": "2004-09-16T23:59:58.75",
  "DC": "NYC",
  "NUM": "212-223-2332"
}
{
  "ID": 2,
  "ERR": "ECC Error",
  "TIME": "2004-09-16T23:59:59.00",
  "DC": "NYC",
  "NUM": "212-223-2332"
}
```

Fuente: COUCHBASE. Making the shift from Relational to Nosql [en línea]. California: Couchbase.Inc [citado 26 septiembre, 2013]. Disponible en internet: <[http://www.couchbase.com/sites/default/files/uploads/all/whitepapers/Couchbase\\_Whitepaper\\_Transitioning\\_Relational\\_to\\_NoSQL.pdf](http://www.couchbase.com/sites/default/files/uploads/all/whitepapers/Couchbase_Whitepaper_Transitioning_Relational_to_NoSQL.pdf)>

Como se puede observar, “los datos son desnormalizados, cada registro contiene un conjunto completo de información asociada al error sin una referencia externa, obteniendo beneficios como hacer que sea fácil mover un registro entero a otro servidor, ya que toda la información simplemente viene con este y mejorando el rendimiento en lectura de dicha información”<sup>56</sup>.

Según Couchbase<sup>57</sup>, puede tomar un tiempo olvidar hábitos, pero entendiendo las alternativas que serán capaces de hacer más eficiente el uso de la información, se podrá elegir sabiamente que herramienta utilizar. Por ejemplo una aplicación

---

<sup>54</sup> Ibid.

<sup>55</sup> Ibid.

<sup>56</sup> Ibid.

<sup>57</sup> Ibid.

existente, donde en vez de dividir el modelo en tablas y columnas, se puede cambiar a JSON y hacer de él una cantidad de documentos, donde cada uno obtenga un id mediante el cual se pueda realizar su búsqueda fácilmente.

“Las llaves primarias en NoSQL se convierten en el id único otorgado a cada documento, esto hace que la búsqueda a través de este ID se vuelva demasiado rápida y seleccionando los necesarios, se pueden hacer mucho más fácil la búsqueda de la información que se requiere”<sup>58</sup>.

#### **1.6.14.2 El modelo relacional vs el modelo de datos orientado a grafos.**

Según Couchbase<sup>59</sup> un grafo es uno de las abstracciones fundamentales en las ciencias de la computación, como tal, hay muchas aplicaciones para los grafos esto conlleva a que virtualmente cada aplicación para grafos tenga una necesidad de guardar y realizar consultas sobre estos, recientemente se ha incrementado el interés para utilizar los grafos con el objetivo de representar herramientas como las redes sociales o estructuras de los sitios web entre otros.

En los últimos años, se tiene que los usuarios han comenzado a llegar más allá del modelo relacional, antes no era muy factible cambiar a otro modelo de datos debido a la relación costo/beneficio que representaba realizar dicho cambio, por tal motivo es que las bases de datos relacionales han sido la estructura de trabajo de la industria de las bases de datos por décadas, pero ahora debido a las necesidades que se han venido desarrollando en todos los ámbitos, se ha presentado la oportunidad de adoptar nuevas alternativas para el modelo de datos a utilizar<sup>60</sup>.

Hay que tener en cuenta que la procedencia significa linaje, de esto se infiere que la procedencia de un objeto de datos es el linaje que tiene este, permitiendo describir lo que este es y cómo ha llegado a serlo. La procedencia de los datos, incluye los detalles acerca de los procesos utilizados como también la procedencia de los datos de entrada que fueron necesarios para crearlos.

Frecuentemente, la procedencia de un dato es almacenada como un grafo acíclico directo (DAG “Directed Acyclic Graph”), guardar este grafo en una base de datos relacional, parece sencillo pero para realizar consultas y recorrerlo se vuelve ineficiente con respecto al tiempo, debido al número de joins, por esta razón es importante considerar un modelo de datos no tradicional para realizar el almacenamiento de estos grafos, las bases de datos orientadas a grafos son la mejor alternativa<sup>61</sup>.

---

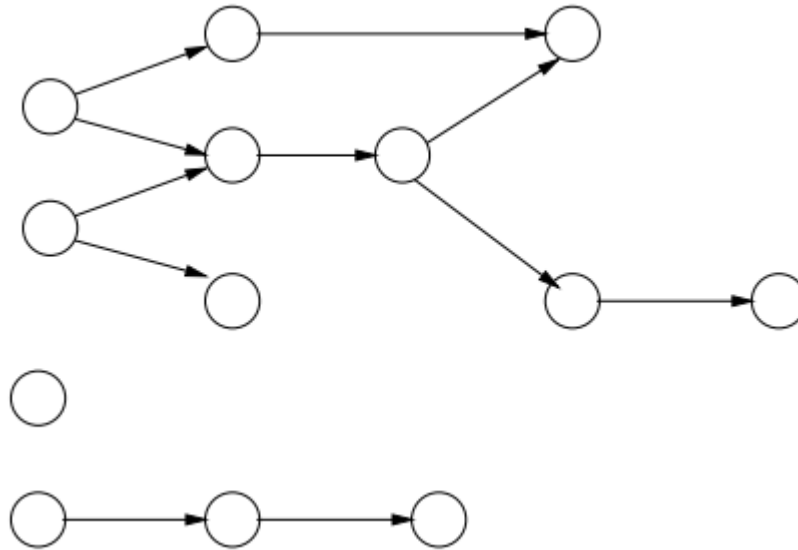
<sup>58</sup> Ibid.

<sup>59</sup> UNIVERSITY OF MISSISSIPPI. A comparison of a Graph Database and a Relational Database [en línea]. Misisipi: Chad Vicknair, Michael Macias, Zhendong Zhao, Xiaofei Nan, Yixin Chen, Dawn Wilkins [citado 13 octubre, 2013]. Disponible en internet: <[http://www.cs.olemiss.edu/~ychen/publications/conference/vicknair\\_acmse10.pdf](http://www.cs.olemiss.edu/~ychen/publications/conference/vicknair_acmse10.pdf)>

<sup>60</sup> Ibid.

<sup>61</sup> Ibid.

**Figura 12. Un ejemplo de grafo aciclico directo**



Fuente: UNIVERSITY OF MISSISSIPPI. A comparison of a Graph Database and a Relational Database [en línea]. Misisipi: Chad Vicknair, Michael Macias, Zhendong Zhao, Xiaofei Nan, Yixin Chen, Dawn Wilkins [citado 13 octubre, 2013]. Disponible en internet: [http://www.cs.olemiss.edu/~ychen/publications/conference/vicknair\\_acmse10.pdf](http://www.cs.olemiss.edu/~ychen/publications/conference/vicknair_acmse10.pdf) >

Las investigaciones de bases de datos orientadas a grafos iniciaron en 1990, pero con la llegada de XML, se hizo más importante la investigación sobre este ámbito dejando atrás todo el tema del modelo orientado a grafos. Fue hasta hace poco que con la aparición de internet como herramienta para todo el público en general, los datos iniciaron a crecer exponencialmente en cuanto a volumen e interconexión. Haciendo esto posible la necesidad de utilizar el modelo de grafos para representar inmensas cantidades de datos, usualmente todos aquellos almacenados en el pasado.

Según experimentos realizados para comparar un modelo de base de datos relacional con un motor MySQL contra un modelo de base de datos orientado a grafos como Neo4J, se implementaron 12 bases de datos por cada motor, donde estas almacenaron cada una, un DAG consistente de algunos nodos, las bases de datos almacenarían información estructurada para representar el grafo e información de carga asociada a cada nodo. La información de carga seria datos semiestructurados en XML o en JSON<sup>62</sup>.

Los resultados del experimento se muestran en la siguiente figura.

---

<sup>62</sup> Ibid.

**Figura 13. Comparación de rendimiento entre MySQL y Neo4j.**

Database	#Nodes	Data Type	MySQL Size	Neo4j Size
1000int	1000	Int	0.232M	0.428M
5000int	5000	Int	0.828M	1.7M
10000int	10000	Int	1.6M	3.2M
100000int	100000	Int	15M	31M
1000char8k	1000	8K Char	18M	33M
5000char8k	5000	8K Char	87M	146M
10000char8k	10000	8K Char	173M	292M
100000char8k	100000	8K Char	1700M	2900M
1000char32k	1000	32K Char	70M	85M
5000char32k	5000	32K Char	504M	406M
10000char32k	10000	32K Char	778M	810M
100000char32k	100000	32K Char	6200M	7900M

Fuente: UNIVERSITY OF MISSISSIPPI. A comparison of a Graph Database and a Relational Database [en línea]. Misissippi: Chad Vicknair, Michael Macias, Zhendong Zhao, Xiaofei Nan, Yixin Chen, Dawn Wilkins [citado 13 octubre, 2013]. Disponible en internet: <[http://www.cs.olemiss.edu/~ychen/publications/conference/vicknair\\_acmse10.pdf](http://www.cs.olemiss.edu/~ychen/publications/conference/vicknair_acmse10.pdf)>

De acuerdo con la anterior figura de, se tiene que los datos de carga utilizados consistían en enteros aleatorios, cadenas aleatorias de 8KB y 32KB. Esta comparación brinda los detalles acerca de las bases de datos y el espacio en disco requerido para cada una de ellas, teniendo en cuenta que ambos motores usan indexación de texto completo se llegó a la conclusión de que Neo4j fue alrededor de 1.25 a 2 veces más grande en cuanto al tamaño de los datos almacenados con respecto a MySQL<sup>63</sup>.

Es necesario conocer las situaciones en las que se debe utilizar SQL o NOSQL según el contexto, Teniendo en cuenta los ámbitos que maneja cada estructura en cuanto a base de datos.

**Tabla 7. Uso de SQL y NoSQL**

Ámbitos de uso para SQL	Ámbitos de uso para NOSQL
Educativo: para aportar a los estudiantes conocimientos acerca de la estructuración de información.	Redes Sociales: Es Obligatorio. Gracias a las redes sociales, esta tecnología comenzó a despegar y mostrar utilidad en el campo de la informática y la estadística.

<sup>63</sup> Ibid.

Tabla 7. (Continuación)

Desarrollo Web: para mantener una misma jerarquía de los datos que llegan de la gran autopista, pero siempre y cuando la capacidad de concurrencia, almacenamiento y mantenimiento no sean de considerable dificultad y la información siempre sea consistente.	Desarrollo Móvil: En estos momentos, las empresas están lidiando con un problema grande conocido como Bring Your Own Device en realidad no es un problema, es un fenómeno social, por lo que la información que se recolecte siempre será diferente por más que uno desee estructurarla y mantenerla estática.
Rama de negocios: para inteligencia de negocios, análisis de negocios, bodegas de datos, minería de datos, minería de texto son temas que requieren el uso de SQL para facilitar el consumo de la información y la identificación de patrones en los datos.	Big Data: Como se observa en search Business Analytics, la administración de grandísimas cantidades de información y su evidente heterogeneidad hace de NOSQL un excelente candidato en esta área.
Empresarial: para el software a la medida y el software empresarial, ambos de escritorio, poseen la característica de mantener información con una estructura consistente y SQL es ideal para esta tarea.	Cloud (XaaS): Termino XaaS (Everything as a service) que indica “cualquier cosa como servicio (sic)” y todos los temas relacionados en la nube, con NoSQL pueden adaptarse casi a cualquier necesidad del cliente, que evidentemente son heterogéneos.

Fuente: WORDPRESS. SQL vs NoSQL [en línea]. Medellín: Esteban Zapata [citado 15 agosto, 2013]. Disponible en internet: <<http://estebanz01.wordpress.com/2013/04/04/sql-vs-nosql-cual-es-el-mejor/>>

## 1.7 MARCO CONCEPTUAL

Los conceptos mencionados son claves para la comprensión del contenido de las secciones tratadas durante el desarrollo de este documento.

**1.7.1 Apache CouchDB.** Es una base de datos NoSQL, que carece de un esquema o de estructuras de datos predefinidas como las tablas en las bases de datos relacionales, la información almacenada son documentos JSON. Y la estructura de los datos o documentos puede acomodarse a las necesidades de cambio o a la evolución del software.

CouchDB es una base de datos que abarca completamente la red, utiliza documentos en JSON para guardar los datos, permite acceder a los datos

desde un navegador web a través del protocolo http, permite realizar operaciones utilizando JavaScript<sup>64</sup>.

**1.7.2 Base de datos.** “Una base de datos o banco de datos (en inglés: database) es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso”<sup>65</sup>.

**1.7.3 Bases de datos NoSQL.** Las bases de datos NoSQL “representan una evolución en la arquitectura de aplicación del negocio, están diseñadas para proveer el almacenamiento de datos confiables, escalables y disponibles a través de un conjunto de sistemas configurables que funcionan como nodos de almacenamiento”<sup>66</sup>.

**1.7.4 Big Data.** “Es la tendencia en el avance de la tecnología que ha abierto las puertas hacia un nuevo enfoque de entendimiento y toma de decisiones, la cual es utilizada para describir enormes cantidades de datos (estructurados, no estructurados y semi-estructurados) que tomaría demasiado tiempo y sería muy costoso cargarlos a un base de datos relacional para su análisis”<sup>67</sup>.

**1.7.5 Big Transaction Data.** Es un tipo de dato “que se utiliza en registros de facturación, en telecomunicaciones registros detallados de las llamadas (CDR), etc. Estos datos transaccionales están disponibles en formatos tanto semi-estructurados como no estructurados”<sup>68</sup>.

**1.7.6 Biometrics.** Es un tipo de dato “en la que se incluye huellas digitales, escaneo de la retina, reconocimiento facial, genética, etc. En el área de seguridad e inteligencia, los datos biométricos han sido información importante para las agencias de investigación”<sup>69</sup>.

**1.7.7 BSON.** “Es una serialización binaria con codificación similar de documentos JSON. BSON apoya la incorporación de documentos y matrices, también contiene extensiones que permiten la representación de los tipos de datos que no son parte de la especificación JSON”<sup>70</sup>.

---

<sup>64</sup> APACHE. Apache CouchDB [en línea]. Los Ángeles: Apache Software Foundation [citado 18 septiembre, 2013]. Disponible en internet: <<http://couchdb.apache.org/>>

<sup>65</sup> ALDANA, Luis. Introducción a las bases de datos. 1 ed. Puebla. p.7.

<sup>66</sup> ORACLE. Oracle NoSQL Database [en línea]. California: Oracle [citado 18 septiembre, 2013]. Disponible en internet: <<http://www.oracle.com/technetwork/products/nosqldb/overview/index.html>>

<sup>67</sup> IBM. ¿Qué es Big data?, Op.cit.

<sup>68</sup> Ibid.

<sup>69</sup> Ibid.

<sup>70</sup> BSON. Definition BSON [en línea]. BSON Foundation [citado 23 octubre, 2013]. Disponible en internet: <<http://bsonspec.org/>>

**1.7.8 Cluster.** Según Strauch<sup>71</sup>, es otro enfoque para la partición de datos que se esfuerza por la transparencia hacia los clientes que deberían manejar un grupo de servidores de bases de datos en lugar de un único servidor. Mientras este enfoque puede ayudar a la escala de la capa de persistencia de un sistema a un cierto grado muchos critican que las características de agrupación sólo se han añadido en la parte superior de los sistemas de gestión de bases de datos que no fueron diseñados originalmente para distribución.

**1.7.9 Hadoop.** “Es un marco de desarrollo de código abierto que permite el procesamiento de grandes conjuntos de datos, de manera distribuida a través de un grupo o clúster de computadoras, usando un modelo de programación sencillo”<sup>72</sup>.

**1.7.10 Hbase.** “Es el sistema de almacenamiento no relacional para Hadoop. Es una base de datos de código abierto, distribuido y escalable para el almacenamiento de Big Data. Está escrita en Java e implementa el concepto de Bigtable desarrollado por Google”<sup>73</sup>.

**1.7.11 HDFS.** “Es el sistema de almacenamiento, es un sistema de ficheros distribuido. Fue creado a partir del Google File System (GFS). HDFS se encuentra optimizado para grandes flujos y trabajar con ficheros grandes en sus lecturas y escrituras. Su diseño reduce la E/S en la red. La escalabilidad y disponibilidad son otras de sus claves, gracias a la replicación de los datos y tolerancia a los fallos”<sup>74</sup>.

**1.7.12 Human Generated.** “Es un tipo de dato que las personas generan, como la información que guarda un call center al establecer una llamada telefónica, notas de voz, correos electrónicos, documentos electrónicos, estudios médicos, etc”<sup>75</sup>.

**1.7.13 JSON.** Es un formato ligero de intercambio de datos, está basado en un subconjunto del lenguaje de programación JavaScript.

Está basado en dos estructuras:

- Una colección de valores pares. En varios lenguajes esto se realiza como un registro, estructura, o arreglo asociado a objetos.
- Una lista ordenada de valores, en la mayoría de lenguajes esto es realizado a través de un arreglo, un vector o una lista<sup>76</sup>.

---

<sup>71</sup> STRAUCH, Op.cit. p.42

<sup>72</sup> IBM. ¿Qué es Big data?, Op.cit.

<sup>73</sup> CENALTIC, Op.cit.

<sup>74</sup> TICOUT. Introducción a Hadoop y su ecosistema [en línea]. Madrid: Ticout [citado 10 de agosto, 2013]. Disponible en internet: <<http://www.ticout.com/blog/2013/04/02/introduccion-a-hadoop-y-su-ecosistema/>>

<sup>75</sup> IBM. ¿Qué es Big data?, Op.cit.

<sup>76</sup> JSON. Introducing JSON [en línea]. JSON Foundation [citado 18 septiembre, 2013]. Disponible en internet: <<http://www.json.org/>>

**1.7.14 Lenguaje SQL.** Es el idioma para bases de datos relacionales. “Se trata de una consulta declarativa, SQL es estandarizada por el Instituto Americano de Estándares Nacionales (ANSI) y Organización Internacional de Normalización (ISO) a partir de 1986 y tiene varias revisiones desde entonces. Bases de datos relacionales de código abierto, como MySQL y PostgreSQL han aumentado la comprensión de SQL entre los desarrolladores de software”<sup>77</sup>.

**1.7.15 Machine-to-Machine (M2M).** “Se refiere a las tecnologías que permiten conectarse a otros dispositivos. M2M utiliza dispositivos como sensores o medidores que capturan algún evento en particular (velocidad, temperatura, presión, variables meteorológicas, variables químicas como la salinidad, etc.) los cuales transmiten a través de redes alámbricas, inalámbricas o híbridas a otras aplicaciones que traducen estos eventos en información significativa”<sup>78</sup>.

**1.7.16 MapReduce.** Es el corazón de hadoop, es el paradigma de programación que permite escalabilidad a través de cientos y miles de servidores en un clúster hadoop. El término MapReduce se refiere actualmente a dos tareas distintas que los programas en hadoop ejecutan, la primera de ellas es el “map job”, el cual toma un conjunto de datos y lo convierte en otro; donde los elementos individuales son desglosados en tuplas, la segunda tarea referente al “reduce job”, toma los datos de salida del “map job” como entrada y combina las tuplas volviéndolas un pequeño conjunto de tuplas. Como la secuencia del nombre MapReduce indica el “reduce Job” siempre se ejecutará después de que el “map Job” haya sido ejecutado<sup>79</sup>.

**1.7.17 Master-Slave.** “Es un modelo para una comunicación de protocolo en el que un dispositivo o proceso (conocido como el maestro) controla uno o más de otros dispositivos o procesos (conocida como esclavos). Una vez establecida la relación maestro /esclavo, la dirección de control es siempre desde el maestro al esclavo”<sup>80</sup>.

**1.7.18 Modelo de datos entidad/relación.** “Es la representación en escala de la realidad de la base de datos, además refleja la estructura de negocio de la organización, por medio de datos y relaciones. El modelo de datos entidad-relación (E-R) es útil para hacer corresponder los significados e interacciones de las empresas del mundo real con un esquema conceptual”<sup>81</sup>.

---

<sup>77</sup> SUTINEN, Oilly. NoSQL—Factors Supporting the Adoption of Non-Relational Databases. Tampere: University of Tampere. Department of Computer Sciences. M.Sc. thesis, 2010, p. 21.

<sup>78</sup> IBM. ¿Qué es Big data?, Op.cit.

<sup>79</sup> IBM. What is MapReduce? [en línea]. New York: IBM [citado 10 agosto, 2013]. Disponible en internet: <<http://www-01.ibm.com/software/data/infosphere/hadoop/mapreduce/>>

<sup>80</sup> SEARCHNETWORKING. Master/slave [en línea]. Massachusetts: Margaret Rouse [citado 23 de octubre, 2013]. Disponible en internet: <<http://searchnetworking.techtarget.com/definition/master-slave>>

<sup>81</sup> SILBERSCHATZ, Abraham. Fundamentos de bases de datos. 5 ed. Madrid: Mac Graw Hill, 2006. p.241.



**1.7.19 MongoDB.** Es un sistema de base de datos multiplataforma orientado a documentos, de esquema libre, esto significa que cada entrada o registro puede tener un esquema de datos diferentes, con atributos o “columnas” que no tienen por qué repetirse de un registro a otro. Está escrito en C++, lo que le confiere cierta cercanía al bare metal, o recursos de hardware de la máquina, de modo que es bastante rápido a la hora de ejecutar sus tareas. Además, está licenciado como GNUAGPL 3.0 de modo que se trata de un software de licencia libre. Funciona en sistemas operativos Windows, Linux, OS X y Solaris<sup>82</sup>.

**1.7.20 Multi-master replication.** En una configuración multi-master, los datos se actualizan en varios patrones. Cada maestro mantiene un registro de cambios, y los cambios realizados en cada master se replican en los demás servidores. Cada maestro tiene el papel de proveedor y el consumidor. La replicación multi-master utiliza un modelo de replicación de consistencia suelta, esto significa que las mismas entradas pueden ser modificadas de forma simultánea en diferentes servidores. Cuando se envían los cambios entre los dos servidores, cualquier cambio en conflicto debe ser resuelto<sup>83</sup>.

**1.7.21 Sharding.** “Medios para dividir los datos de tal manera que los datos solicitados y actualizados estén en el mismo nodo y que el volumen de carga y almacenamiento se distribuyan entre los servidores. Los fragmentos de datos también pueden ser replicados por razones de fiabilidad y de equilibrio de carga y pueden esperarse a escribir en sólo una réplica dedicado o a todas las réplicas que mantienen una partición de los datos”<sup>84</sup>.

**1.7.22 Web and Social Media.** Es un tipo de dato que “incluye contenido web e información que es obtenida de las redes sociales como Facebook, Twitter, LinkedIn, etc, blogs”<sup>85</sup>.

**1.7.23 XQuery.** “Es un lenguaje de consulta para XML. Los casos de uso más comunes para XQuery implican publicación XML para crear XML para los mensajes web, sitios web dinámicos, o aplicaciones de publicación. Los datos originales se pueden encontrar en archivos XML, o que pueden estar en un almacén de datos tal como una base de datos relacional”<sup>86</sup>.

---

<sup>82</sup> GENBETADEV. Una introducción a MongoDB [en línea]. Madrid: Carlos Paramio [citado 18 septiembre, 2013]. Disponible en internet: <<http://www.genbetadev.com/bases-de-datos/una-introduccion-a-mongodb>>

<sup>83</sup> ORACLE. Concepts of Multi-Master Replication [en línea]. California: ORACLE [citado 23 de octubre, 2013]. Disponible en internet: <<http://docs.oracle.com/cd/E19656-01/821-1502/aalfq/index.html>>

<sup>84</sup> STRAUCH, Op.cit, p.42.

<sup>85</sup> IBM. ¿Qué es Big data?, Op.cit.

<sup>86</sup> PROGRESS DATADERICT. Using XQuery [en línea]. Massachusetts: DataDirect Technologies [citado 23 de octubre, 2013]. Disponible en internet: <[http://www.xquery.com/tutorials/xquery\\_tutorial/](http://www.xquery.com/tutorials/xquery_tutorial/)>

## 1.8 METODOLOGÍA

**1.8.1 Tipo de Estudio.** La metodología de investigación elegida para el desarrollo del proyecto es la sintética, “puesto que a partir de varios elementos por separado se pretende llegar a un resultado concreto”<sup>87</sup>.

Para cumplir a cabalidad esta metodología, inicialmente se plantearán todas las tecnologías a utilizar durante el desarrollo del proyecto, realizando previamente un análisis frente a sus características, con el objetivo de determinar cuáles son las más viables para la implementación de la guía.

En la segunda fase, una vez definidas las herramientas a utilizar y teniendo claro los objetivos a cumplir, se iniciará con la construcción del ambiente Big data ejecutando las pruebas en los equipos de la Universidad Católica de Colombia, siguiendo así la finalidad de esta metodología, donde se parte de lo abstracto para llegar a lo concreto.

**1.8.2 Fuentes de Información.** Dentro de las fuentes de información que se van a utilizar para el desarrollo del proyecto, se encuentran:

**1.8.2.1 Fuentes primarias.** Libros especializados en el manejo del tema de Big Data, que logran identificar conceptos claves para conseguir una implementación óptima de la guía; también artículos de internet, los cuales se tomarán como guía para escoger e instalar las herramientas más adecuadas para la construcción del ambiente.

**1.8.2.2 Fuentes secundarias.** La participación en conferencias sobre Big Data, tanto de manera presencial como de manera virtual, adicionalmente la asesoría con personas que tengan altos conocimientos acerca del tema, logrando así un óptimo desarrollo de la guía.

---

<sup>87</sup> MIALARET, Gastón. Psicología de la Educación. 1ed. Paris: Siglo XXI Editores, 2001. p.57

## **2. CONCLUSIONES**

Big data es una nueva tendencia para el manejo de grandes volúmenes de información, utilizado principalmente por grandes empresas, pero gracias a las nuevas tecnologías y su fácil acceso podrá ser utilizado por cualquier empresa o institución que desee vincularse al nuevo proceso que se puede lograr en la gestión de la información.

La estructura de un ambiente Big Data ayuda a mejorar la manipulación de los datos, optimizando la gestión de la información respecto a tiempo y costo, logrando obtener mejores resultados en las estadísticas para una buena toma de decisiones.

La creación de un ambiente Big Data se debe realizar dentro de un cluster, el cual permita integrar todas las aplicaciones que se van a utilizar, como en este caso Hadoop, en el cual se almacena la información y las aplicaciones corren dentro del mismo nodo, evitando conflictos durante la ejecución.

Es importante resaltar que existen muchas maneras para transformar el mismo modelo relacional al modelo basado en columnas, ya que se pueden tomar distintos caminos para la unión de los datos, esto depende de la información que se desee encontrar o saber. Para obtener una adecuada transformación se deben tener en cuenta las llaves primarias, las cuales se convertirán en las row key, que permitirá integrar toda la información dentro de una misma columna, mejorando la manipulación que se darán a los datos.

## BIBLIOGRAFÍA

ALDANA, Luis. Introducción a las bases de datos. 1 ed. Puebla. 80 p.

APACHE. Apache Avro [en línea]. Los Ángeles: Apache Software Foundation [citado 09 octubre, 2013]. Disponible en internet: <<http://avro.apache.org/docs/current/>>

APACHE. Apache CouchDB [en línea]. Los Ángeles: Apache Software Foundation [citado 18 septiembre, 2013]. Disponible en internet: <<http://couchdb.apache.org/>>

APACHE. Apache Chukwa [en línea]. Los Ángeles: Apache Software Foundation [citado 09 octubre, 2013]. Disponible en internet: <<http://incubator.apache.org/chukwa/index.html>>

APACHE. Apache Flume [en línea]. Los Ángeles: Apache Software Foundation [citado 09 octubre, 2013]. Disponible en internet: <<http://flume.apache.org/>>.

APACHE. Apache Hadoop [en línea]. Los Ángeles: Apache Software Foundation [citado 25 septiembre, 2013]. Disponible en internet: <<http://hadoop.apache.org/>>

APACHE. Apache Pig [en línea] Los Ángeles: Apache Software Foundation [citado 09 octubre, 2013]. Disponible en internet: <<http://pig.apache.org/>>

APACHE. Apache Sqoop [en línea]. Los Ángeles: Apache Software Foundation [citado 09 octubre, 2013]. Disponible en internet: <[http://sqoop.apache.org/docs/1.4.2/SqoopUserGuide.html#\\_introduction](http://sqoop.apache.org/docs/1.4.2/SqoopUserGuide.html#_introduction)>

AYENDE@RAHIEN. Column Family [en línea]. California: Ayende [citado 10 Noviembre, 2013]. Disponible en internet: <<http://ayende.com/blog/4500/that-no-sql-thing-column-family-databases>>

BARRAGAN, Ana, FORERO, Andrea. Implementación de una base de datos No SQL para la generación de la matriz o/d. Bogotá: Universidad Católica de Colombia. Facultad de Ingeniería de sistemas. Modalidad Proyecto de investigación, 2012. 205 p.

BBVA. Big Data: ¿En qué punto estamos? [en línea]. Bogotá: BBVA [citado 7 Agosto, 2013]. Disponible en internet: <<https://www.centrodeinnovacionbbva.com/innovation-edge/21-big-data/p/153-big-data-en-que-punto-estamos>>

BSON. Definition BSON [en línea]. BSON Foundation [citado 23 octubre, 2013] Disponible en internet: <<http://bsonspec.org/>>

CENATIC. Open Smart Cities II: Big Data de Código Abierto [en línea]. Badajoz: Cenaltic [citado 10 agosto, 2013]. Disponible en internet: <[http://observatorio.cenatic.es/index.php?option=com\\_content&view=article&id=804:open-smart-cities-ii-open-big-data-&catid=94:tecnologia&Itemid=137#ftn3](http://observatorio.cenatic.es/index.php?option=com_content&view=article&id=804:open-smart-cities-ii-open-big-data-&catid=94:tecnologia&Itemid=137#ftn3)>

COUCHBASE. Making the shift from Relational to Nosql [en línea]. California: Couchbase.Inc [citado 26 septiembre, 2013]. Disponible en internet: <[http://www.couchbase.com/sites/default/files/uploads/all/whitepapers/Couchbase\\_Whitepaper\\_Transitioning\\_Relational\\_to\\_NoSQL.pdf](http://www.couchbase.com/sites/default/files/uploads/all/whitepapers/Couchbase_Whitepaper_Transitioning_Relational_to_NoSQL.pdf)>

DEVELOPERWORKS. Big data de código abierto [en línea]. México D.F.: Marty Lurie [citado 25 septiembre, 2013]. Disponible en internet: <<http://www.ibm.com/developerworks/ssa/data/library/techarticle/dm-209hadoopbigdata/>>

EMPRESAS.IT. Tres casos de éxito de Big Data en Colombia [en línea]. Bogotá: Alejandro González [citado 7 Agosto, 2013]. Disponible en internet: <<http://empresas.it/2013/05/tres-casos-de-exito-de-big-data-en-colombia/>>.

FORRESTER. The pragmatic definition of Big data [en línea]. Cambridge: Mike Gualtieri [citado 29 septiembre, 2013]. Disponible en internet: <[http://blogs.forrester.com/mike\\_gualtieri/12-12-05-the\\_pragmatic\\_definition\\_of\\_big\\_data](http://blogs.forrester.com/mike_gualtieri/12-12-05-the_pragmatic_definition_of_big_data)>

GARTNER. Big Data [en línea]. Connecticut: Gartner [citado 22 septiembre, 2013]. Disponible en internet: <<http://www.gartner.com/it-glossary/big-data/>>

GENBETADEV. Una introducción a MongoDB [en línea]. Madrid: Carlos Paramio [citado 18 septiembre, 2013]. Disponible en internet: <<http://www.genbetadev.com/bases-de-datos/una-introducción-a-mongodb>>

HORTONWORKS. Apache Hive [en línea]. California: HortonWorks [citado 09 octubre, 2013]. Disponible en internet: <<http://hortonworks.com/hadoop/hive/>>

HP. La era del Big Data [en línea]. California: Sara Artaza [citado 29 septiembre, 2013]. Disponible en internet: <<http://h30499.www3.hp.com/t5/Infraestructura-Convergente-de/La-Era-del-Big-Data/ba-p/6151357#.UkiHs4ZLMvL>>

IBM. ¿Qué es big data? [en línea]. México D.F: Ricardo Barranco [citado 15 agosto, 2013]. Disponible en internet: <<http://www.ibm.com/developerworks/ssa/local/im/que-es-big-data/>>

IBM. Stream Processing [en línea]. New York: Nagui Halim [citado 29 de Septiembre de 2013]. Disponible en internet: <<http://www.ibm.com/smarter-computing/us/en/technical-breakthroughs/stream-processing.html>>

IBM. What is MapReduce? [en línea]. New York: IBM [citado 10 agosto, 2013]. Disponible en internet: <<http://www-01.ibm.com/software/data/infosphere/hadoop/mapreduce/>>

INFOQUEUE. Transitioning from rdbms to NoSQL [en línea]. California: Avram Abel [citado 26 septiembre, 2013]. Disponible en internet: <<http://www.infoq.com/articles/Transition-RDBMS-NoSQL>>

JAVA4DEVELOPERS. Introducción a Big Data y Hadoop [en línea]. Madrid: Java4Developers [citado 25 septiembre, 2013]. Disponible en internet: <<http://java4developers.com/2013/introduccion-a-big-data-y-hadoop/>>

JSON. Introducing JSON [en línea]. JSON Foundation [citado 18 septiembre, 2013]. Disponible en internet: <<http://www.json.org/>>

MIALARET, Gastón. Psicología de la Educación. 1ed. Paris: Siglo XXI Editores, 2001. 147 p.

NASHOLM, Petter. Extracting Data from NoSQL Databases. Gotemburgo, 2012. 67 p.

ORACLE. Concepts of Multi-Master Replication [en línea]. California: ORACLE [citado 23 de octubre, 2013]. Disponible en internet: <<http://docs.oracle.com/cd/E19656-01/821-1502/aalfq/index.html>>

ORACLE. Oracle NoSQL Databases. 1 ed. California, 2011. 12 p.

ORACLE. Oracle NoSQL Database [en línea]. California: Oracle [citado 18 septiembre, 2013]. Disponible en internet: <<http://www.oracle.com/technetwork/products/nosqldb/overview/index.html>>

PROGRESS DATADERICT. Using XQuery [en línea]. Massachusetts: DataDirect Technologies [citado 23 de octubre, 2013]. Disponible en internet: <[http://www.xquery.com/tutorials/xquery\\_tutorial/](http://www.xquery.com/tutorials/xquery_tutorial/)>

ROUTLEDGE. Critical questions for Big data [en línea]. Cambridge: Danah Boyd & Kate Crawford [citado 29 septiembre, 2013]. Disponible en internet: <<http://www.tandfonline.com/doi/pdf/10.1080/1369118X.2012.678878> >

SEARCHNETWORKING. Master/slave [en línea]. Massachusetts: Margaret Rouse [citado 23 de octubre, 2013]. Disponible en internet: <<http://searchnetworking.techtarget.com/definition/master-slave>>

SG. NoSQL la evolución de las bases de datos [en línea]. Cataluña: Erik Camacho [citado 08 Agosto, 2013]. Disponible en internet: <<http://sg.com.mx/content/view/966>>

SILBERSCHATZ, Abraham. Fundamentos de bases de datos. 5 ed. Madrid: Mac Graw Hill, 2006. 797 p.

STRAUCH, Christof. NoSQL Databases. 1 ed. New York, 2011. 149 p.

SUTINEN, Olly. NoSQL–Factors Supporting the Adoption of Non-Relational Databases. Tampere: University of Tampere. Department of Computer Sciences. M.Sc. thesis, 2010. 47 p.

TECHTARGET. Análisis de “big data” [en línea]. Massachusetts: Margaret Rouse [citado 29 de Septiembre de 2013]. Disponible en internet: <<http://searchdatacenter.techtarget.com/es/definicion/Analisis-de-big-data/>>

TECHTARGET. Data Ingestion [en línea]. Massachusetts: Margaret Rouse [citado 29 Septiembre, 2013]. Disponible en internet: <<http://whatis.techtarget.com/definition/data-ingestion>>

TECHTARGET. Data management [en línea]. Massachusetts: Margaret Rouse [citado 29 Septiembre, 2013]. Disponible en internet: <<http://searchdatamanagement.techtarget.com/definition/data-management>>

THE ENDEAVOUR. ACID versus BASE for database transactions [en línea]. Houston: John Cook [citado 10 septiembre, 2013]. Disponible en internet: <<http://www.johndcook.com/blog/2009/07/06/brewer-cap-theorem-base/>>

TICOUT. Introducción a Hadoop y su ecosistema [en línea]. Madrid: Ticout [citado 10 de agosto, 2013]. Disponible en internet: <<http://www.ticout.com/blog/2013/04/02/introduccion-a-hadoop-y-su-ecosistema/>>

UNIVERSIDAD AUTÓNOMA DE BARCELONA. Análisis Bioinformáticos sobre la tecnología hadoop [en línea]. Madrid: Carmen Palacios Díaz [citado 10 agosto, 2013]. Disponible en internet: <[http://ddd.uab.cat/pub/trerecpro/2012/hdl\\_2072\\_196270/GutierrezMillaAlbertR-ETISa2010-11.pdf](http://ddd.uab.cat/pub/trerecpro/2012/hdl_2072_196270/GutierrezMillaAlbertR-ETISa2010-11.pdf)>.

UNIVERSIDAD CARLOS III DE MADRID. Evaluación de la herramienta de código libre Apache Hadoop [en línea]. Madrid: Carmen Palacios Díaz [citado 10 agosto, 2013]. Disponible en internet: <[http://e-archivo.uc3m.es/bitstream/10016/13533/1/MemoriaPFC\\_MCarmenPalacios.pdf](http://e-archivo.uc3m.es/bitstream/10016/13533/1/MemoriaPFC_MCarmenPalacios.pdf)>.

UNIVERSITY OF AMSTERDAM. Defining the Big Data Architecture Framework [en línea]. Ámsterdam: Yuri Demchenko [citado 22 septiembre, 2013]. Disponible en internet: < [http://bigdatawg.nist.gov/\\_uploadfiles/M0055\\_v1\\_7606723276.pdf](http://bigdatawg.nist.gov/_uploadfiles/M0055_v1_7606723276.pdf)>

UNIVERSITY OF MISSISSIPPI. A comparison of a Graph Database and a Relational Database [en línea]. Mississippi: Chad Vicknair, Michael Macias, Zhendong Zhao, Xiaofei Nan, Yixin Chen, Dawn Wilkins [citado 13 octubre, 2013]. Disponible en internet: <[http://www.cs.olemiss.edu/~ychen/publications/conference/vicknair\\_acmse10.pdf](http://www.cs.olemiss.edu/~ychen/publications/conference/vicknair_acmse10.pdf)>

WORDPRESS. SQL VS NOSQL [en línea] Medellín: Esteban Zapata [citado 15 agosto, 2013]. Disponible en internet: <<http://estebanz01.wordpress.com/2013/04/04/sql-vs-nosql-cual-es-el-mejor/>>



## **ANEXOS**

### **Anexo A. Guía para la creación del ambiente Big Data**

En este apartado se describen los pasos necesarios para la creación de un ambiente Big Data, utilizando tres herramientas: Hadoop, Hive y Hbase, las cuales se escogieron por su fácil integración y el manejo que le dan a los datos que se van a utilizar como caso de estudio.

Para lograr el montaje de un ambiente Big Data se debe tener en cuenta los siguientes requerimientos y pasos a seguir, para no tener ningún inconveniente durante el proceso.

#### **Requerimientos de Hardware y Software**

- Sistema operativo: Ubuntu 13.04.
- Tipo de sistema operativo: 32 bits.
- JDK: Java 6.
- Memoria RAM: 2 GB.
- Espacio libre en disco: 1 GB (Esto incluye únicamente la instalación, se puede requerir más dependiendo del tamaño de la data a procesar).

#### **Pasos a seguir**

- Instalar Hadoop con el fin de configurar un cluster y lograr incluir todas las herramientas en el mismo nodo.
- Instalar Hive (Framework para consultas SQL).
- Integrar Hive con Hadoop.
- Instalar Hbase dentro del cluster de Hadoop (motor NoSQL).
- Integrar Hbase con Hive.

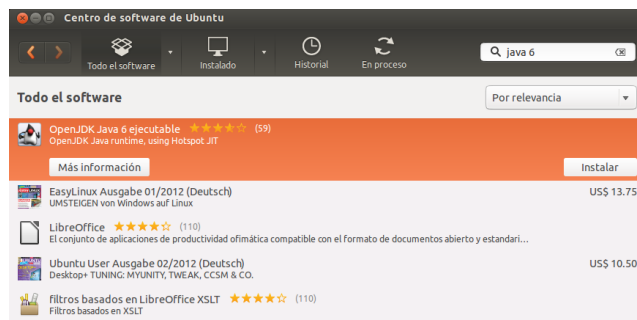
## INSTALACIÓN HADOOP

Hadoop es la herramienta que se utiliza como nodo, donde va a quedar integrado todo el ambiente. Esta herramienta permite correr todas las aplicaciones dentro del mismo cluster, por lo tanto toda la información quedara alojada allí y podrá ser utilizada sin ningún inconveniente.

A continuación se describen los pasos la instalación de Hadoop:

1. Antes de empezar con la instalación de Hadoop se debe tener en la máquina virtual Java, por lo tanto se instala Java 6 desde el centro de Software de Ubuntu, como se muestra en la siguiente figura:

**Figura 14. Instalación Java**



Fuente: Autores

2. Verificar que java quedo instalado probando desde la terminal con el comando java. Debe arrojar la información que se muestra en la siguiente figura:

**Figura 15. Validación Java**

```
warrior@warrior-VirtualBox: ~
warrior@warrior-VirtualBox:~$ java
Usage: java [-options] class [args...]
        (to execute a class)
    or java [-options] -jar jarfile [args...]
        (to execute a jar file)
where options include:
    -d32          use a 32-bit data model if available
    -d64          use a 64-bit data model if available
    -client       to select the "client" VM
    -server       to select the "server" VM
    -j9vm         to select the "j9vm" VM
    -cacao        to select the "cacao" VM
    -zero         to select the "zero" VM
    -hotspot      is a synonym for the "client" VM [deprecated]
                  The default VM is client.

    -cp <class search path of directories and zip/jar files>
    -classpath <class search path of directories and zip/jar files>
                  A : separated list of directories, JAR archives,
                  and ZIP archives to search for class files.
    -D<name>=<value>
                  set a system property
    -verbose[:<class>|gc|jni]
                  enable verbose output
```

Fuente: Autores

3. Instalar OpenJdk6 desde la terminal, con el comando `sudo apt-get install openjdk-6-jdk`. Cuando aparezca el mensaje ¿Desea continuar [s/n]?, escribir s como se muestra en la figura N°16, dar enter y seguir con la instalación.

**Figura 16. Instalar OpenJdk**

```
warrior@warrior-VirtualBox: ~  
warrior@warrior-VirtualBox:~$ sudo apt-get install openjdk-6-jdk  
[sudo] password for warrior:  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias  
Leyendo la información de estado... Hecho  
Se instalarán los siguientes paquetes extras:  
  libice-dev libpthread-stubs0 libpthread-stubs0-dev libsm-dev libx11-6  
  libx11-dev libx11-doc libxau-dev libxcb1 libxcb1-dev libxdmcp-dev  
  libxt-dev libxt6 x11proto-core-dev x11proto-input-dev x11proto-kb-dev  
  xorg-sgml-doctools xtrans-dev  
Paquetes sugeridos:  
  libice-doc libsm-doc libxcb-doc libxt-doc openjdk-6-demo openjdk-6-source  
  visualvm  
Se instalarán los siguientes paquetes NUEVOS:  
  libice-dev libpthread-stubs0 libpthread-stubs0-dev libsm-dev libx11-dev  
  libx11-doc libxau-dev libxcb1-dev libxdmcp-dev libxt-dev openjdk-6-jdk  
  x11proto-core-dev x11proto-input-dev x11proto-kb-dev xorg-sgml-doctools  
  xtrans-dev  
Se actualizarán los siguientes paquetes:  
  libx11-6 libxcb1 libxt6  
3 actualizados, 16 se instalarán, 0 para eliminar y 251 no actualizados.  
Se necesita descargar 16,4 MB/17,4 MB de archivos.  
Se utilizarán 53,2 MB de espacio de disco adicional después de esta operación.  
¿Desea continuar [S/n]? s
```

Fuente: Autores

4. Verificar la instalación del OpenJdk a través del comando `javac`. Debe arrojar la información como muestra la siguiente figura:

**Figura 17. Verificación OpenJdk**

```
warrior@warrior-VirtualBox: ~  
warrior@warrior-VirtualBox:~$ javac  
Usage: javac <options> <source files>  
where possible options include:  
  -g               Generate all debugging info  
  -g:none          Generate no debugging info  
  -g:{lines,vars,source} Generate only some debugging info  
  -nowarn          Generate no warnings  
  -verbose         Output messages about what the compiler is doing  
  -deprecation     Output source locations where deprecated APIs are  
used  
  -classpath <path> Specify where to find user class files and annota  
tion processors  
  -cp <path>       Specify where to find user class files and annota  
tion processors  
  -sourcepath <path> Specify where to find input source files  
  -bootclasspath <path> Override location of bootstrap class files  
  -extdirs <dirs>   Override location of installed extensions  
  -endorseddirs <dirs> Override location of endorsed standards path  
  -proc:{none,only} Control whether annotation processing and/or comp  
ilation is done.  
  -processor <class1>[,<class2>,<class3>...] Names of the annotation processors  
to run; bypasses default discovery process  
  -processorpath <path> Specify where to find annotation processors  
  -d <directory>    Specify where to place generated class files
```

Fuente: Autores

5. Descargar la última versión de Hadoop en la página de Apache <http://apache.mirrors.tds.net/hadoop/common/>, escoger la carpeta stable, como se muestra en la figura N°18 y allí escoger el archivo con extensión .tar.gz.

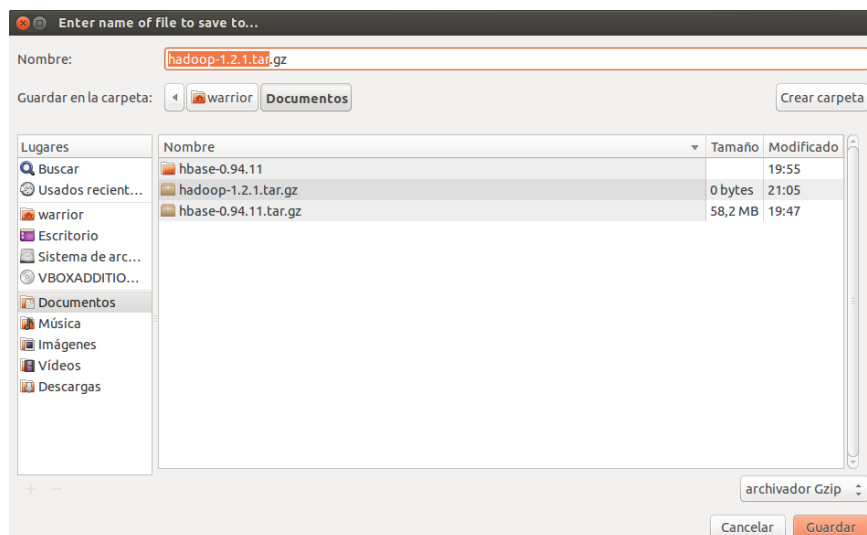
**Figura 18. Descargar Hadoop**



Fuente: Autores

6. Guardar el archivo en la carpeta Documentos, como se ilustra en la siguiente imagen:

**Figura 19. Guardar archivo Hadoop**



Fuente: Autores

7. Descomprimir el archivo desde la terminal a través del comando: `$ tar xzf hadoop-1.2.1.tar.gz`. Luego mover el archivo descomprimido a una carpeta de

nombre hadoop a través del comando mv, como se muestra en la siguiente figura:

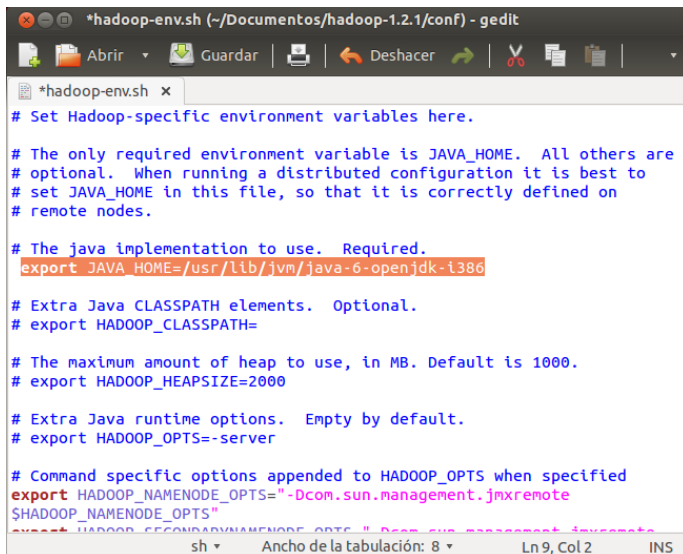
**Figura 20. Descomprimir archivo Hadoop**

```
warrior@warrior-VirtualBox:~/Documentos$  
warrior@warrior-VirtualBox:~/Documentos$ mv hadoop-1.2.1 hadoop  
warrior@warrior-VirtualBox:~/Documentos$
```

Fuente: Autores

8. Abrir el archivo hadoop-env.sh que se encuentra en la carpeta conf con el Editor de texto. Quitar el comentario en la dirección del JAVA HOME, como se muestra en la figura N° 21 y cambiar la dirección que se encuentra allí por la que se encuentra instalado el java, que por defecto es usr/lib/jvm/java-6-openjdk-i386. Guardar la configuración.

**Figura 21. Configurar Java Home**



```
*hadoop-env.sh (-~/Documentos/hadoop-1.2.1/conf) - gedit  
# Set Hadoop-specific environment variables here.  
  
# The only required environment variable is JAVA_HOME. All others are  
# optional. When running a distributed configuration it is best to  
# set JAVA_HOME in this file, so that it is correctly defined on  
# remote nodes.  
  
# The java implementation to use. Required.  
export JAVA_HOME=/usr/lib/jvm/java-6-openjdk-i386  
  
# Extra Java CLASSPATH elements. Optional.  
# export HADOOP_CLASSPATH=  
  
# The maximum amount of heap to use, in MB. Default is 1000.  
# export HADOOP_HEAPSIZE=2000  
  
# Extra Java runtime options. Empty by default.  
# export HADOOP_OPTS=-server  
  
# Command specific options appended to HADOOP_OPTS when specified  
export HADOOP_NAMENODE_OPTS="-Dcom.sun.management.jmxremote  
$HADOOP_NAMENODE_OPTS"  
export HADOOP_SECONDARYNAMENODE_OPTS="-Dcom.sun.management.jmxremote  
$HADOOP_SECONDARYNAMENODE_OPTS"  
sh Ancho de la tabulación: 8 Ln 9, Col 2 INS
```

Fuente: Autores

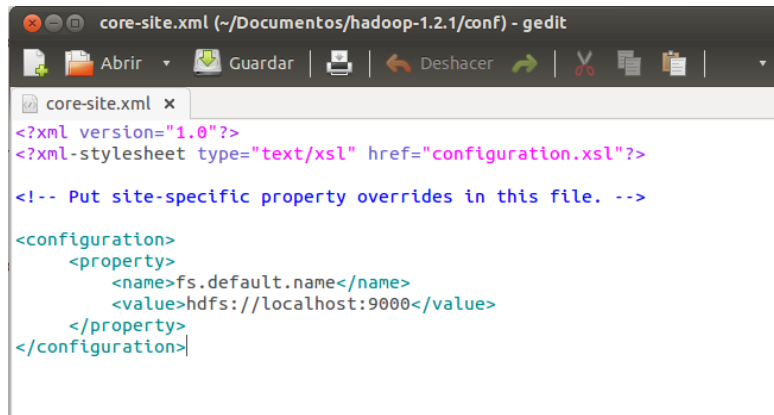
A continuación se realiza la modificación de los archivos para ejecutar hadoop en un modo pseudo-distribuido, esto permitirá que las demás herramientas se puedan integrar correctamente.

9. Cambiar la configuración del archivo core-site.xml que se encuentra en la carpeta conf, como se muestra en la figura N°22, por la siguiente información:

```
<configuration>  
  <property>  
    <name>fs.default.name</name>  
    <value>hdfs://localhost:9000</value>
```

```
</property>
</configuration>
```

**Figura 22. Configurar el archivo core**



Fuente: Autores

10. Cambiar la configuración del archivo hdfs-site.xml que se encuentra en la carpeta conf, como se muestra en la figura N°23, por la siguiente información:

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
</configuration>
```

**Figura 23. Configuración archivo hdfs**

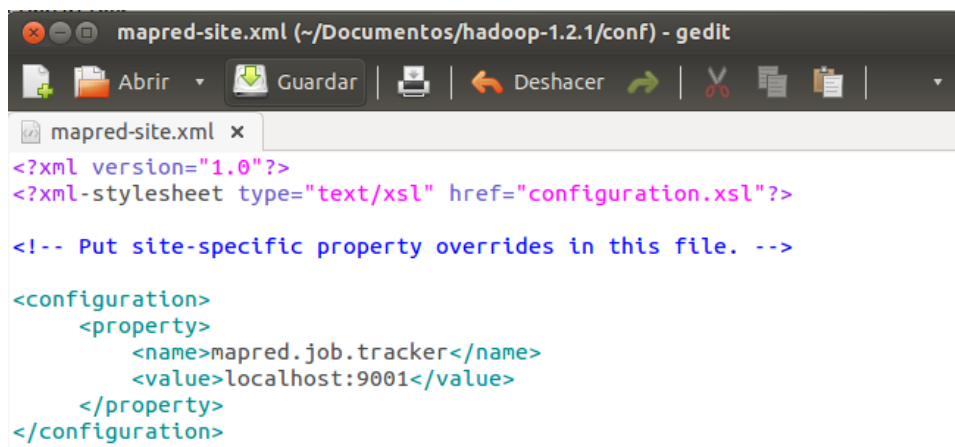


Fuente: Autores

11. Cambiar la configuración del archivo mapred-site.xml que se encuentra en la carpeta conf, como se muestra en la figura N°24, por la siguiente información:

```
<configuration>
  <property>
    <name>mapred.job.tracker</name>
    <value>localhost:9001</value>
  </property>
</configuration>
```

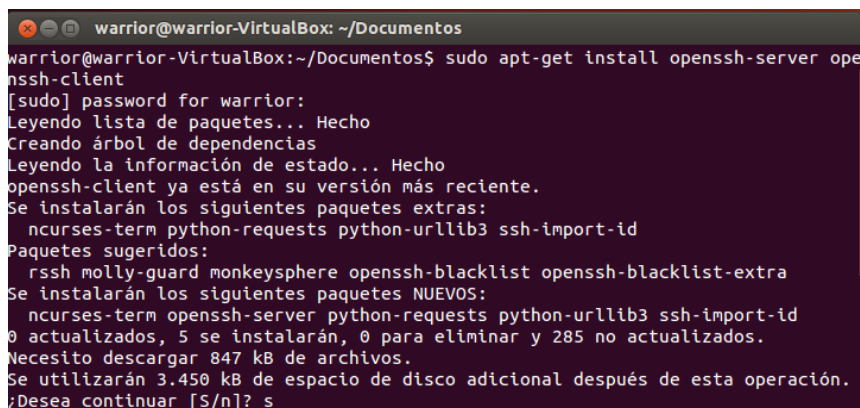
**Figura 24. Configuración archivo mapred**



Fuente: Autores

12. Instalar en la terminal el complemento ssh para poder utilizar los scripts de Hadoop a través del siguiente comando: \$ sudo apt-get install openssh-server openssh-client. Cuando aparezca el mensaje ¿Desea continuar [s/n]?, como aparece en la figura N°25, se escoge la opción “s” y se sigue la instalación.

**Figura 25. Instalación ssh**



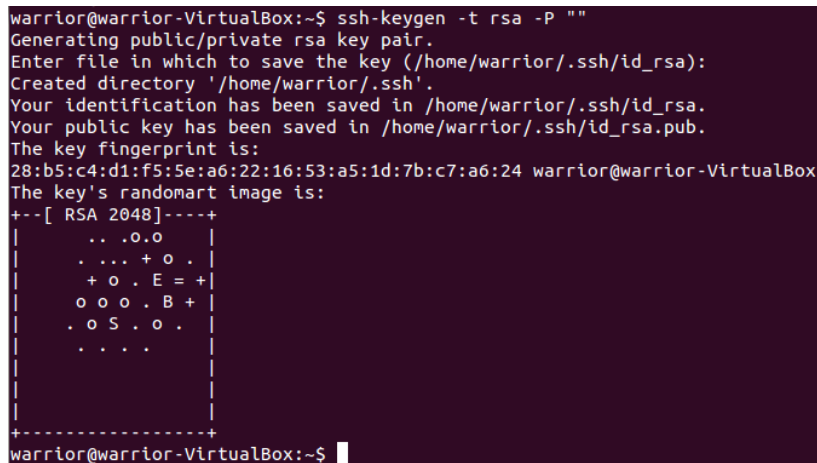
Fuente: Autores

13. Configurar el complemento ssh, como se muestra en la figura N°26, a través de los siguientes comandos:

```
$ ssh-keygen -t rsa -P "".
```

```
$ cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys.
```

**Figura 26. Configuración ssh**

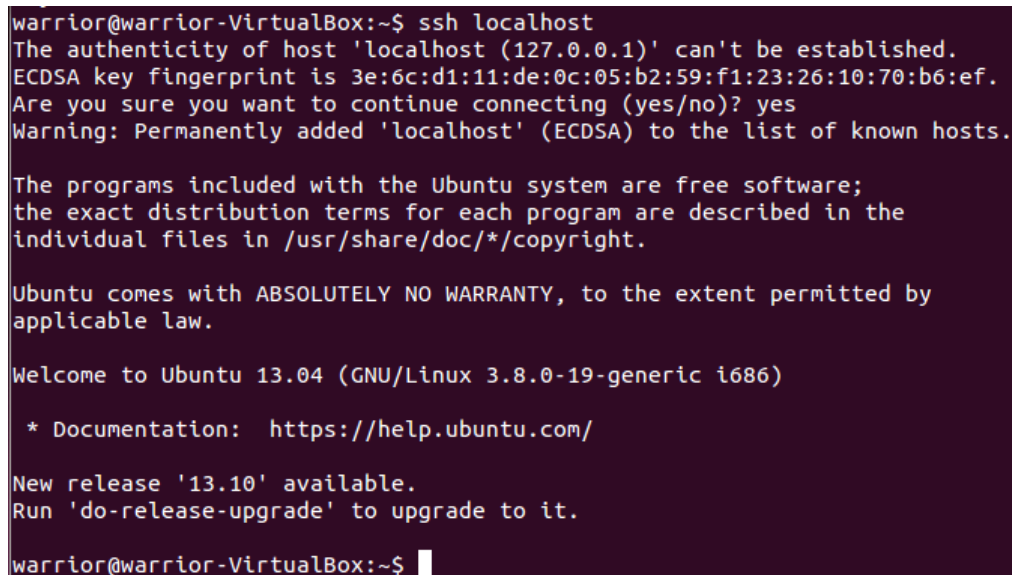


```
warrior@warrior-VirtualBox:~$ ssh-keygen -t rsa -P ""
Generating public/private rsa key pair.
Enter file in which to save the key (/home/warrior/.ssh/id_rsa):
Created directory '/home/warrior/.ssh'.
Your identification has been saved in /home/warrior/.ssh/id_rsa.
Your public key has been saved in /home/warrior/.ssh/id_rsa.pub.
The key fingerprint is:
28:b5:c4:d1:f5:5e:a6:22:16:53:a5:1d:7b:c7:a6:24 warrior@warrior-VirtualBox
The key's randomart image is:
+--[ RSA 2048 ]-----+
|      .. .o.o      |
|      . ... + o .   |
|      + o . E = +   |
|      o o o . B +   |
|      . o S . o .   |
|      . . . .       |
+-----+
warrior@warrior-VirtualBox:~$
```

Fuente: Autores

14. Iniciar el script ssh como se muestra en la siguiente figura:

**Figura 27. Inicio script ssh**



```
warrior@warrior-VirtualBox:~$ ssh localhost
The authenticity of host 'localhost (127.0.0.1)' can't be established.
ECDSA key fingerprint is 3e:6c:d1:11:de:0c:05:b2:59:f1:23:26:10:70:b6:ef.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

Welcome to Ubuntu 13.04 (GNU/Linux 3.8.0-19-generic i686)

 * Documentation:  https://help.ubuntu.com/

New release '13.10' available.
Run 'do-release-upgrade' to upgrade to it.

warrior@warrior-VirtualBox:~$
```

Fuente: Autores



15. Crear la variable HADOOP\_HOME en el path como se muestra en la figura N°28, a través de los siguientes comandos:

```
$ export HADOOP_HOME=/Documentos/hadoop/bin (dirección de descarga).  
$ export HADOOP_IN_PATH=/Documentos/hadoop/bin.
```

**Figura 28. Configuración PATH hadoop**

```
warrior@warrior-VirtualBox:~$ export HADOOP_HOME=/Documentos/hadoop/bin  
warrior@warrior-VirtualBox:~$ export HADOOP_IN_PATH=/Documentos/hadoop/bin  
warrior@warrior-VirtualBox:~$
```

Fuente: Autores

16. Dar formato al namenode, como se muestra en la figura N°29, desde la ruta de la carpeta bin a través del siguiente comando:

```
$/hadoop namenode -format
```

**Figura 29. Formato del namenode**

```
warrior@warrior-VirtualBox: ~/Documentos/hadoop/bin  
warrior@warrior-VirtualBox:~/Documentos/hadoop/bin$ ./hadoop namenode -format  
13/10/22 23:13:24 INFO namenode.NameNode: STARTUP_MSG:  
/*****  
STARTUP_MSG: Starting NameNode  
STARTUP_MSG:  host = warrior-VirtualBox/127.0.1.1  
STARTUP_MSG:  args = [-format]  
STARTUP_MSG:  version = 1.2.1  
STARTUP_MSG:  build = https://svn.apache.org/repos/asf/hadoop/common/branches/b  
ranch-1.2 -r 1503152; compiled by 'mattf' on Mon Jul 22 15:23:09 PDT 2013  
STARTUP_MSG:  java = 1.6.0_27  
*****/  
13/10/22 23:13:25 INFO util.GSet: Computing capacity for map BlocksMap  
13/10/22 23:13:25 INFO util.GSet: VM type      = 32-bit  
13/10/22 23:13:25 INFO util.GSet: 2.0% max memory = 1013645312  
13/10/22 23:13:25 INFO util.GSet: capacity     = 2^22 = 4194304 entries  
13/10/22 23:13:25 INFO util.GSet: recommended=4194304, actual=4194304  
13/10/22 23:13:26 INFO namenode.FSNamesystem: fsOwner=warrior  
13/10/22 23:13:27 INFO namenode.FSNamesystem: supergroup=supergroup  
13/10/22 23:13:27 INFO namenode.FSNamesystem: isPermissionEnabled=true  
13/10/22 23:13:27 INFO namenode.FSNamesystem: dfs.block.invalidate.limit=100  
13/10/22 23:13:27 INFO namenode.FSNamesystem: isAccessTokenEnabled=false accessK  
eyUpdateInterval=0 min(s), accessTokenLifetime=0 min(s)  
13/10/22 23:13:27 INFO namenode.FSEditLog: dfs.namenode.edits.toleration.length  
= 0
```

Fuente: Autores

17. Iniciar los daemons de hadoop, ejecutando el archivo start-all.sh que se encuentra en la carpeta bin, a través del siguiente comando: \$ ./start-all.sh.

Se debe ver la información que aparece en la siguiente figura:

**Figura 30. Iniciar hadoop**

```
warrior@warrior-VirtualBox:~/Documentos/hadoop/bin$ ./start-all.sh
starting namenode, logging to /home/warrior/Documentos/hadoop/libexec/./logs/hadoop-warrior-namenode-warrior-VirtualBox.out
localhost: starting datanode, logging to /home/warrior/Documentos/hadoop/libexec/./logs/hadoop-warrior-datanode-warrior-VirtualBox.out
localhost: starting secondarynamenode, logging to /home/warrior/Documentos/hadoop/libexec/./logs/hadoop-warrior-secondarynamenode-warrior-VirtualBox.out
starting jobtracker, logging to /home/warrior/Documentos/hadoop/libexec/./logs/hadoop-warrior-jobtracker-warrior-VirtualBox.out
localhost: starting tasktracker, logging to /home/warrior/Documentos/hadoop/libexec/./logs/hadoop-warrior-tasktracker-warrior-VirtualBox.out
warrior@warrior-VirtualBox:~/Documentos/hadoop/bin$
```

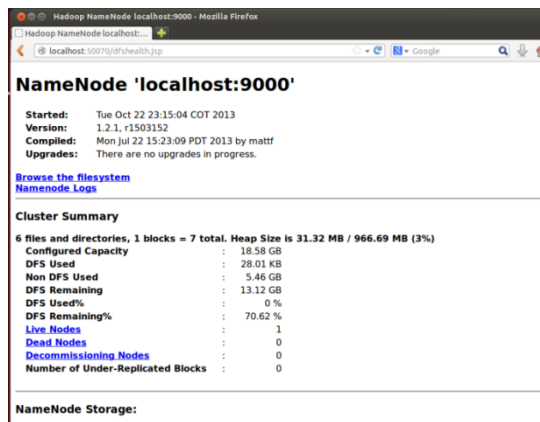
Fuente: Autores

18. Comprobar los servicios que están arriba a través del comando jps, debe mostrar algo como lo siguiente:

```
14799 NameNode
14977 SecondaryNameNode
15183 DataNode
15596 JobTracker
15897 TaskTracker
```

19. Comprobar la conexión a través de <http://localhost:50070>, arrojará la siguiente imagen.

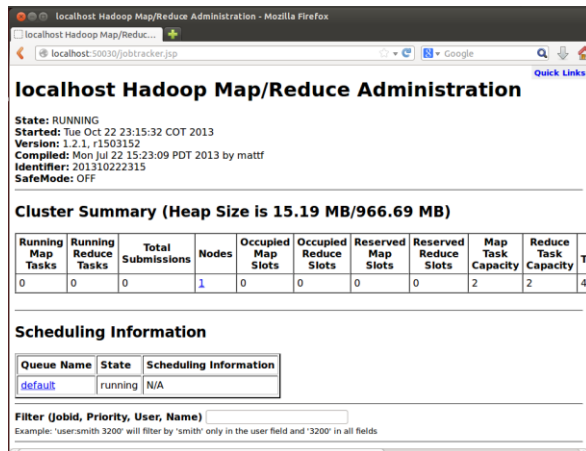
**Figura 31. Interfaz web namenode**



Fuente: Autores

20. Comprobar la conexión de job tracker a través de <http://localhost:50030>, debe mostrar la siguiente imagen.

**Figura 32. Interfaz web Job Tracker**



Fuente: Autores

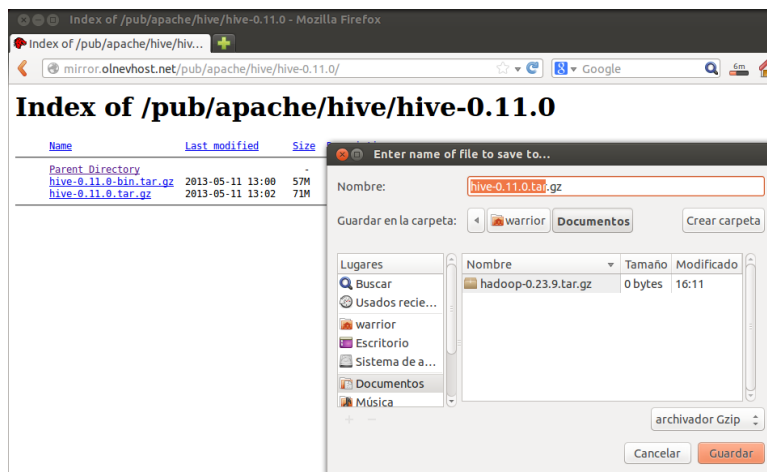
## INSTALACIÓN HIVE

Hive es un framework que permite ingresar datos y realizar consultas dentro de la base de datos NoSQL Hbase, utilizando lenguaje SQL. En la sección de integración se explicara la manera de realizar este procedimiento.

A continuación se describen los pasos para la configuración de la herramienta.

1. Descargar la última versión de Hive en la página de Apache <http://mirror.olinevhost.net/pub/apache/hive/>, escoger la carpeta stable, como se muestra en la figura N°33 y allí escoger el archivo con extensión .tar.gz. Guardar el archivo en Documentos.

**Figura 33. Descargar Hive**



Fuente: Autores

2. Descomprimir el archivo que se encuentra en la ruta Documentos, a través del comando `tar xzf`, como se ilustra en la figura N° 34. Luego mover el archivo descomprimido a una carpeta de nombre `hive` con el siguiente comando: `$ mv hive-0.11.0 hive`.

**Figura 34. Descomprimir hive**

```
warrior@warrior-VirtualBox:~$ cd Documentos
warrior@warrior-VirtualBox:~/Documentos$ tar xzf hive-0.11.0.tar.gz
warrior@warrior-VirtualBox:~/Documentos$ mv hive-0.11.0 hive
warrior@warrior-VirtualBox:~/Documentos$ cd hive
```

Fuente: Autores

3. Crear la variable `HIVE_HOME` en el path como se ilustra en la figura N° 35, a través de los siguientes comandos, esto con el fin de que se realice correctamente la instalación, ya que Hive requiere de Hadoop para su ejecución.

```
$ export HIVE_HOME=/Documentos/hive (dirección de descarga).
$ export PATH=$HIVE_HOME/bin:$PATH.
```

**Figura 35. Configuración PATH de hive**

```
warrior@warrior-VirtualBox:~/Documentos/hive$ export HIVE_HOME=/Documentos/hive
warrior@warrior-VirtualBox:~/Documentos/hive$ export PATH=$HIVE_HOME/bin:$PATH
warrior@warrior-VirtualBox:~/Documentos/hive$
```

Fuente: Autores

4. Instalar el complemento de subversión a través del siguiente comando: `$ sudo apt-get install subversión`. Cuando aparezca el mensaje `¿Desea continuar?`, escribir `s` como se muestra en la figura N° 36 y continuar con la instalación.

**Figura 36. Instalación complemento subversión**

```
warrior@warrior-VirtualBox: ~/Documentos/hive
warrior@warrior-VirtualBox:~/Documentos/hive$ sudo apt-get install subversion
[sudo] password for warrior:
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes extras:
  libapr1 libaprutil1 libserf1 libsvn1
Paquetes sugeridos:
  subversion-tools db5.1-util
Se instalarán los siguientes paquetes NUEVOS:
  libapr1 libaprutil1 libserf1 libsvn1 subversion
0 actualizados, 5 se instalarán, 0 para eliminar y 285 no actualizados.
Necesito descargar 1.615 kB de archivos.
Se utilizarán 4.578 kB de espacio de disco adicional después de esta operación.
¿Desea continuar [S/n]? s
Des:1 http://co.archive.ubuntu.com/ubuntu/ raring/main libapr1 i386 1.4.6-3ubun
u1 [90,5 kB]
Des:2 http://co.archive.ubuntu.com/ubuntu/ raring/main libaprutil1 i386 1.4.1-3
[85,0 kB]
Des:3 http://co.archive.ubuntu.com/ubuntu/ raring/main libserf1 i386 1.1.0-2 [4
,4 kB]
Des:4 http://co.archive.ubuntu.com/ubuntu/ raring-updates/main libsvn1 i386 1.7
5-1ubuntu3.1 [1.113 kB]
Des:5 http://co.archive.ubuntu.com/ubuntu/ raring-updates/main subversion i386
```

Fuente autores

5. Realizar build a través del comando: `$ svn co http://svn.apache.org/repos/asf/hive/trunk hive`, como se muestra en la siguiente figura:

**Figura 37. Construcción Hive**

```
warrior@warrior-VirtualBox:~/Documentos/hive$ svn co http://svn.apache.org/repos/asf/hive/trunk hive

Fetching external item into 'hive/hcatalog/src/test/e2e/harness':
Se obtuvo recurso externo en la revisión 1535131.

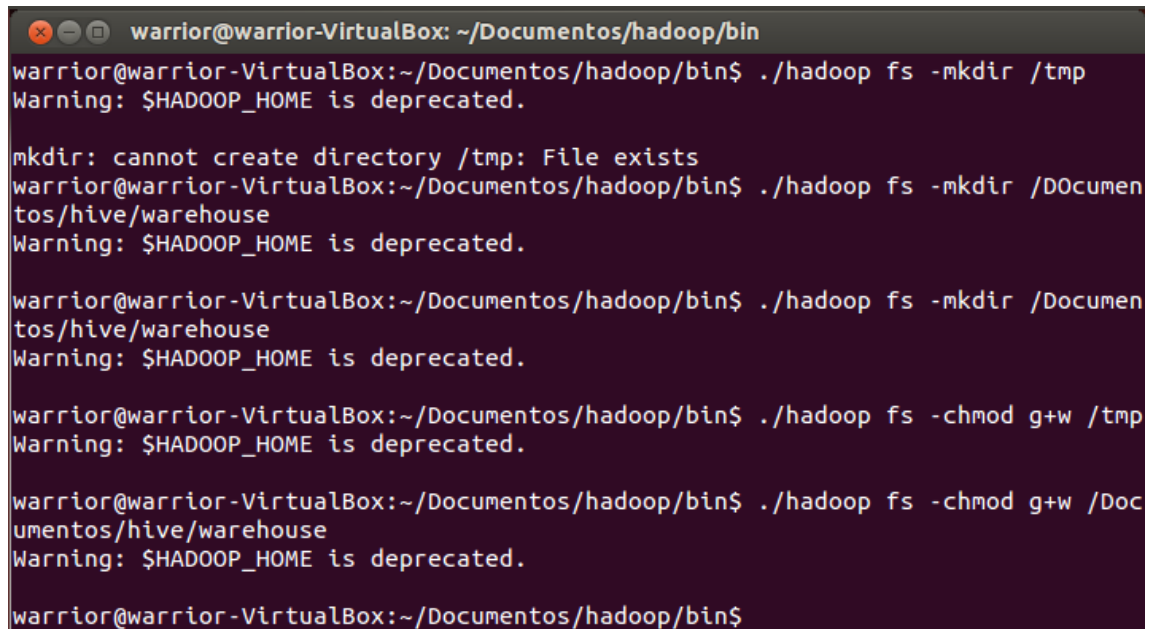
Revisión obtenida: 1535131
warrior@warrior-VirtualBox:~/Documentos/hive$
```

Fuente: Autores

6. Configurar Hive para integrar con Hadoop con la creación de carpetas desde la ruta de `/Documentos/hadoop/bin`, como se muestra en la figura N° 38, a través de los siguientes comandos:

```
$ ./hadoop fs -mkdir /tmp
$ ./hadoop fs -mkdir /Documentos/hive/warehouse
$ ./hadoop fs -chmod g+w /tmp
$ ./hadoop fs -chmod g+w /user/hive/warehouse
```

**Figura 38. Configuración Hive**



```
warrior@warrior-VirtualBox: ~/Documentos/hadoop/bin
warrior@warrior-VirtualBox:~/Documentos/hadoop/bin$ ./hadoop fs -mkdir /tmp
Warning: $HADOOP_HOME is deprecated.

mkdir: cannot create directory /tmp: File exists
warrior@warrior-VirtualBox:~/Documentos/hadoop/bin$ ./hadoop fs -mkdir /Documents/hive/warehouse
Warning: $HADOOP_HOME is deprecated.

warrior@warrior-VirtualBox:~/Documentos/hadoop/bin$ ./hadoop fs -mkdir /Documents/hive/warehouse
Warning: $HADOOP_HOME is deprecated.

warrior@warrior-VirtualBox:~/Documentos/hadoop/bin$ ./hadoop fs -chmod g+w /tmp
Warning: $HADOOP_HOME is deprecated.

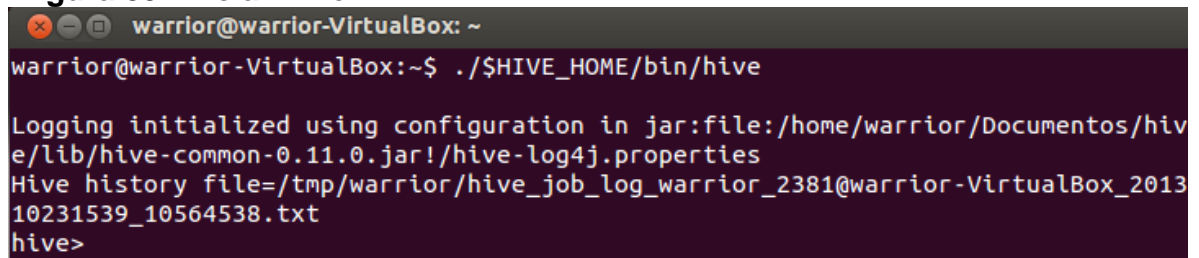
warrior@warrior-VirtualBox:~/Documentos/hadoop/bin$ ./hadoop fs -chmod g+w /Documents/hive/warehouse
Warning: $HADOOP_HOME is deprecated.

warrior@warrior-VirtualBox:~/Documentos/hadoop/bin$
```

Fuente: Autores

7. Iniciar hive como se muestra en la figura N°39, a través del siguiente comando:  
\$ ./HIVE\_HOME/bin/hive.

**Figura 39. Iniciar Hive**



```
warrior@warrior-VirtualBox: ~
warrior@warrior-VirtualBox:~$ ./HIVE_HOME/bin/hive

Logging initialized using configuration in jar:file:/home/warrior/Documentos/hive/lib/hive-common-0.11.0.jar!/hive-log4j.properties
Hive history file=/tmp/warrior/hive_job_log_warrior_2381@warrior-VirtualBox_201310231539_10564538.txt
hive>
```

Fuente: Autores

## INSTALACIÓN HBASE

Hbase es el motor de Bases de datos NoSQL utilizado para el ambiente Big Data, dentro de este se implementara el caso de estudio con la creación de las tablas, la inserción y la consulta de los datos.

A continuación se describen los pasos para la configuración de Hbase:

1. Descargar la última versión de HBASE desde el enlace <http://apache.mirrors.tds.net/hbase/>. Escoger la carpeta stable versión. Allí

escoger el archivo con extensión .tar.gz., como se muestra en la siguiente figura:

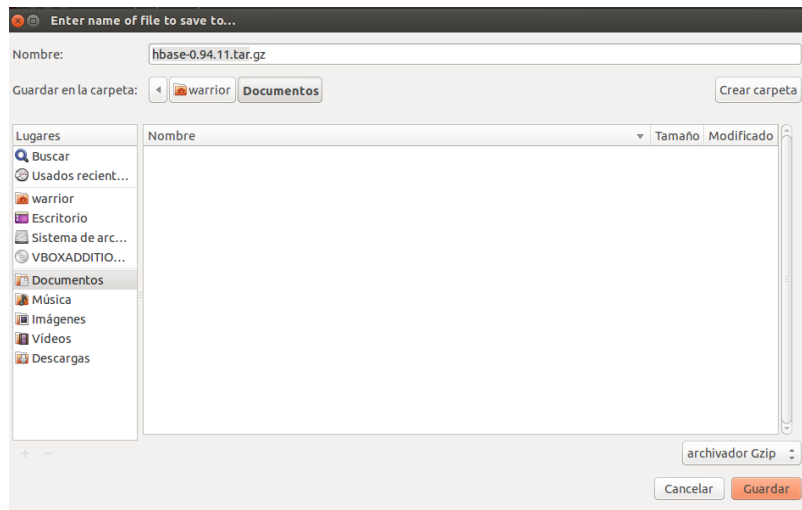
**Figura 40. Descargar de HBASE**



Fuente: Autores

2. Guardar el archivo en la carpeta Documentos como se muestra en la figura N°41.

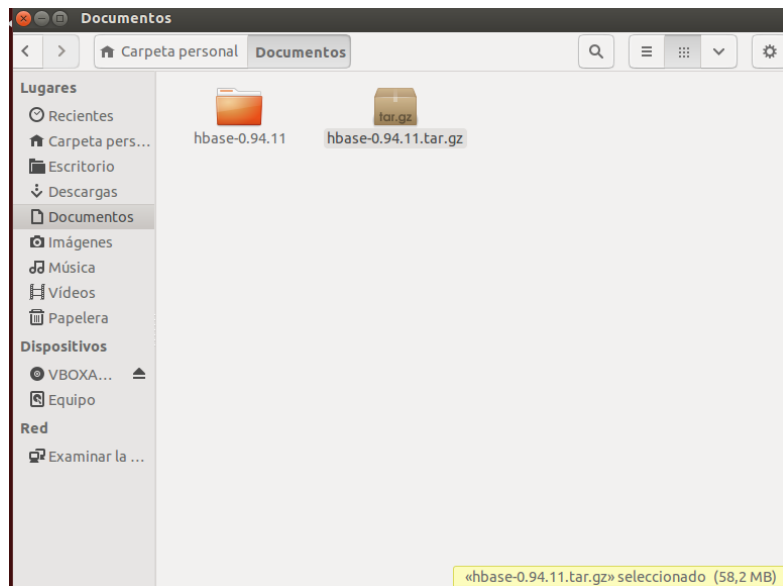
**Figura 41. Guardar archivo Hbase**



Fuente: Autores

3. Descomprimir el archivo en la dirección escogida anteriormente. Debe aparecer una carpeta con el mismo nombre como muestra la figura.

**Figura 42. Descomprimir archivo Hbase**



Fuente: Autores

4. Abrir el archivo hbase-site.xml que está en la carpeta conf (dentro de la carpeta descomprimida en el punto anterior) con el Editor de texto, cambiar la información que está allí por:

```
<configuration>
  <property>
    <name>hbase.rootdir</name>
    <value>hdfs://localhost:9000/hbase</value>
  </property>
  <property>
    <name>hbase.cluster.distributed</name>
    <value>true</value>
  </property>
  <property>
    <name>hbase.zookeeper.quorum</name>
    <value>localhost</value>
  </property>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
  <property>
    <name>hbase.zookeeper.property.clientPort</name>
    <Value>2181</value>
  </property>
```

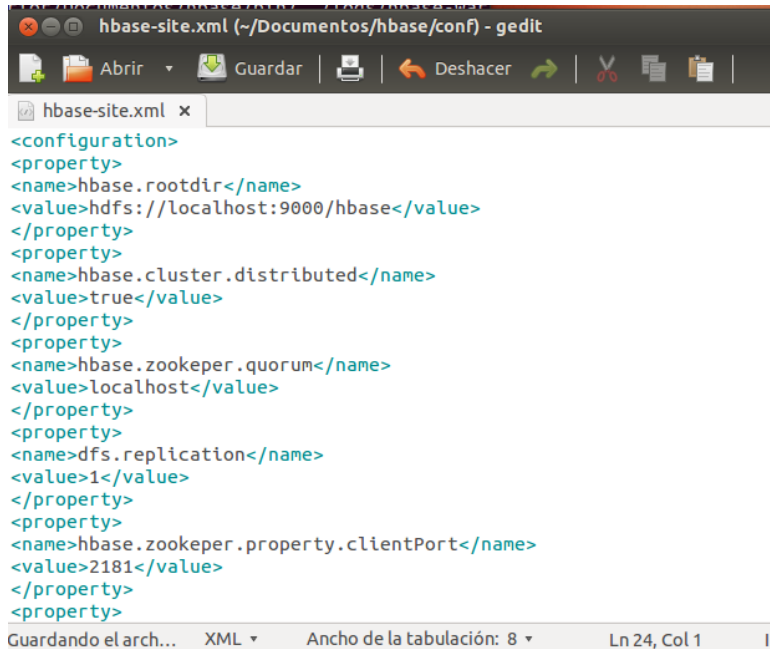


</configuration>

Esto permitirá ejecutar Hbase en un modo pseudo-distribuido integrando la aplicación en el cluster de hadoop.

Por último se guarda la configuración realizada, como ilustra la siguiente figura:

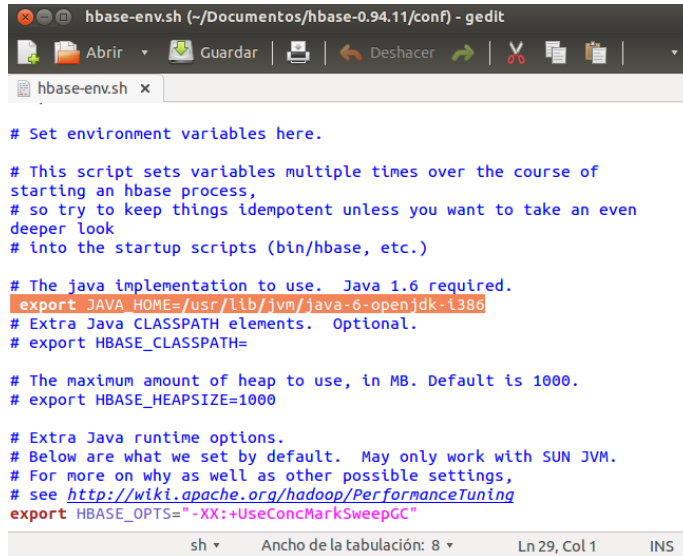
**Figura 43. Configuración Hbase**



Fuente: Autores

5. Abrir el archivo hbase-env.sh que se encuentra en la carpeta conf con el Editor de texto. Quitar el comentario en la dirección del PATH, como se muestra en la figura N°44 y cambiar la dirección que se encuentra allí por la que se encuentra instalado el java, que por defecto es usr/lib/jvm/java-6-openjdk-i386. Guardar la configuración

**Figura 44. Configuración Path Java**



```
hbase-env.sh (~/Documentos/hbase-0.94.11/conf) - gedit
# Set environment variables here.

# This script sets variables multiple times over the course of
# starting an hbase process,
# so try to keep things idempotent unless you want to take an even
# deeper look
# into the startup scripts (bin/hbase, etc.)

# The java implementation to use. Java 1.6 required.
export JAVA_HOME=/usr/lib/jvm/java-6-openjdk-1386
# Extra Java CLASSPATH elements. Optional.
# export HBASE_CLASSPATH=

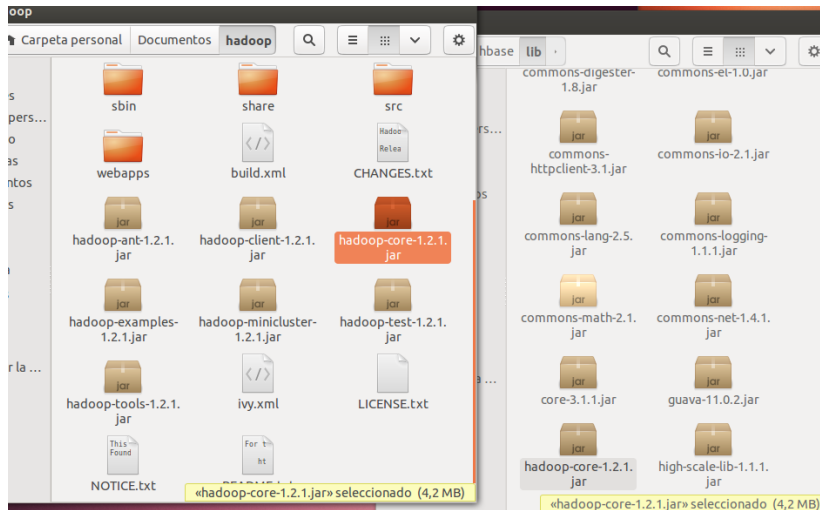
# The maximum amount of heap to use, in MB. Default is 1000.
# export HBASE_HEAPSIZE=1000

# Extra Java runtime options.
# Below are what we set by default. May only work with SUN JVM.
# For more on why as well as other possible settings,
# see http://wiki.apache.org/hadoop/PerformanceTuning
export HBASE_OPTS="-XX:+UseConcMarkSweepGC"
```

Fuente: Autores

6. Antes de ejecutar el motor NoSQL Hbase se debe copiar la librería de hadoop con el nombre `hadoopcore-1.2.1.jar`, como se muestra en la figura N° 45, en la carpeta `lib` de Hbase. Esto permitirá ejecutar Hbase sobre hadoop sin ningún problema.

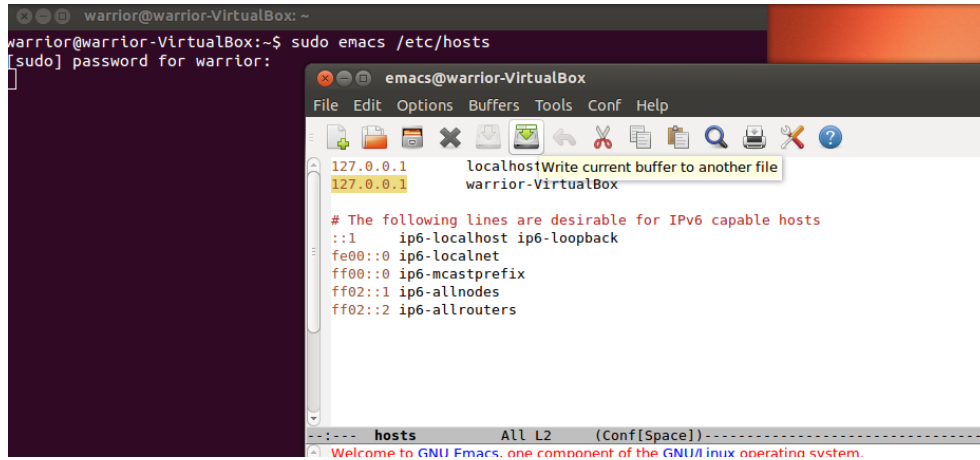
**Figura 45. Copiar librerías Hadoop**



Fuente: Autores

7. Cambiar la dirección de ejecución de la máquina que se encuentra por defecto en `127.0.1.1` por la dirección local `127.0.0.1`, a través del comando: `sudo emacs /etc/hosts`, como se muestra en la siguiente figura.

**Figura 46. Cambiar dirección de la maquina**



Fuente: Autores

8. Iniciar Hbase desde la terminal ejecutando el archivo start-hbase.sh, que se encuentra en la carpeta bin, a través del comando `$ ./start-hbase.sh`.

La configuración se muestra en la siguiente figura:

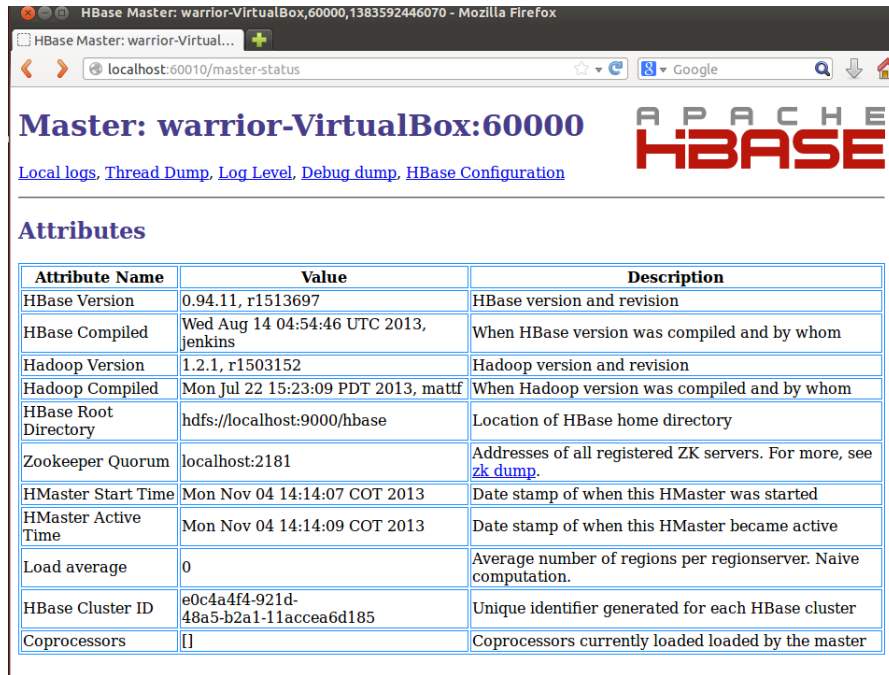
**Figura 47. Iniciar Hbase**

```
warrior@warrior-VirtualBox:~$ ./Documentos/hbase/bin/start-hbase.sh  
localhost: starting zookeeper, logging to /home/warrior/Documentos/hbase/bin/../  
logs/hbase-warrior-zookeeper-warrior-VirtualBox.out  
starting master, logging to /home/warrior/Documentos/hbase/bin/../logs/hbase-warrior-master-warrior-VirtualBox.out  
localhost: regionserver running as process 3974. Stop it first.  
warrior@warrior-VirtualBox:~$
```

Fuente: Autores

9. Comprobar que se encuentra correctamente instalado Hbase a través de `http://localhost:60010`, debe arrojar la siguiente imagen:

**Figura 48. Interfaz Hbase**

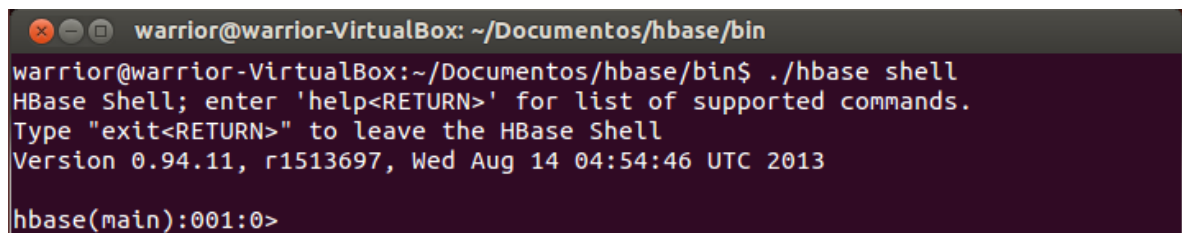


Attribute Name	Value	Description
HBase Version	0.94.11, r1513697	HBase version and revision
HBase Compiled	Wed Aug 14 04:54:46 UTC 2013, jenkins	When HBase version was compiled and by whom
Hadoop Version	1.2.1, r1503152	Hadoop version and revision
Hadoop Compiled	Mon Jul 22 15:23:09 PDT 2013, mattf	When Hadoop version was compiled and by whom
HBase Root Directory	hdfs://localhost:9000/hbase	Location of HBase home directory
Zookeeper Quorum	localhost:2181	Addresses of all registered ZK servers. For more, see <a href="#">zk_dump</a> .
HMaster Start Time	Mon Nov 04 14:14:07 COT 2013	Date stamp of when this HMaster was started
HMaster Active Time	Mon Nov 04 14:14:09 COT 2013	Date stamp of when this HMaster became active
Load average	0	Average number of regions per regionserver. Naive computation.
HBase Cluster ID	e0c4a4f4-921d-48a5-b2a1-11acce6d185	Unique identifier generated for each HBase cluster
Coprocessors	[]	Coprocessors currently loaded loaded by the master

Fuente: Autores

10. Ahora puede empezar a utilizar HBase, ejecutando hbase Shell desde la ruta bin, como se muestra en la siguiente figura:

**Figura 49. Hbase shell**



```
warrior@warrior-VirtualBox: ~/Documentos/hbase/bin
warrior@warrior-VirtualBox:~/Documentos/hbase/bin$ ./hbase shell
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 0.94.11, r1513697, Wed Aug 14 04:54:46 UTC 2013

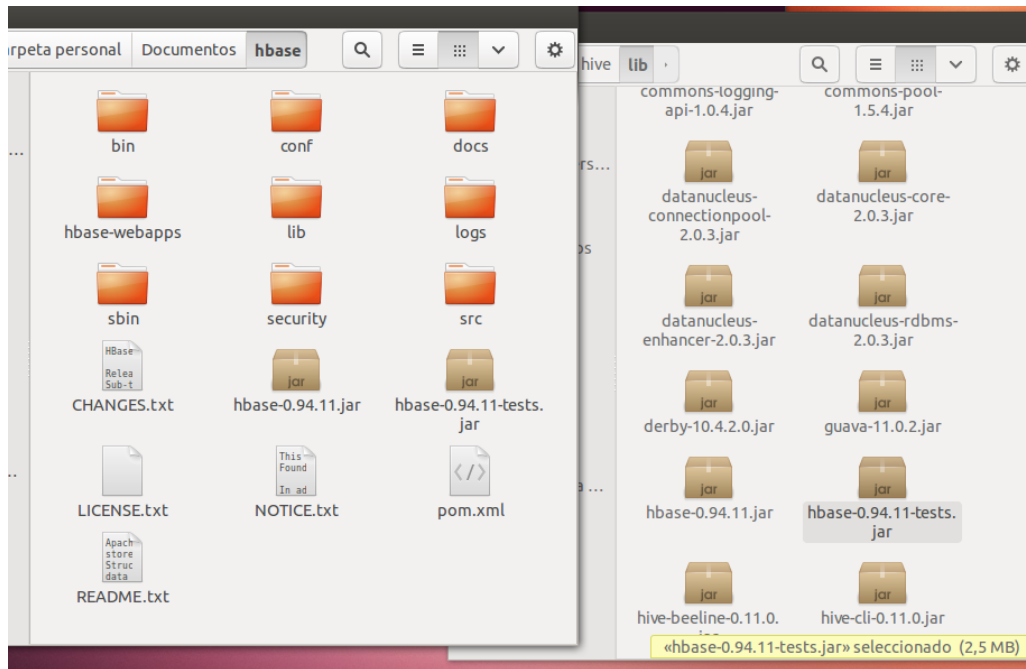
hbase(main):001:0>
```

Fuente: Autores

## INTEGRAR HBASE Y HIVE

Para comenzar la integración se deben copiar las librerías de Hbase como se muestra en la figura N° 50, que se encuentra en la carpeta Hbase con el nombre hbase-0.94.11.jar y hbase-0.94.11-tests.jar; luego se reemplaza por las librerías que se encuentra en la ruta hive/lib. Esto permitirá que hive se pueda compilar con la versión que se tiene de Hbase.

**Figura 50. Copiar librerías de Hbase**



Fuente: Autores

Dentro de sus librerías Hive trae los siguientes jar:

guava-11.0.2.jar

hive-hbase-handler-0.11.0.jar

Zookeeper-3.4.3.jar

Esto permite la integración de las dos herramientas, logrando que Hive utilice las librerías de Hbase para conectar sus componentes.

Hive puede ingresar datos y realizar consultas sobre tablas que se encuentran en Hbase a través de la creación de tablas externas, que permiten conectar los datos de ambas aplicaciones. A continuación se explica el procedimiento a realizar con un ejemplo práctico:

1. Crear una tabla en Hbase a través del comando `create`, el cual tiene la siguiente estructura :

`create 'nombre tabla', 'Familia1', 'Familia2', ..., 'FamiliaN'`

Se crea la tabla de nombre 'hbase' con los campos info, con el siguiente comando:  
`create 'hbase', 'info'`

2. Ingresar datos dentro de la tabla creada con las columnas user y pass, a través del comando `put` que tiene la siguiente estructura:

Put 'nombre de la tabla', 'row key', 'familia: columna' , 'valor de la columna'

Se crean los campos con los siguientes datos:

```
put 'hbase','row1', 'info: user','hola'  
put 'hbase','row 2', 'info: pass','123'
```

3. Crear la tabla externa en Hive, se debe tener en cuenta la relación que tienen las columnas en Hbase para adaptarla a tabla que se crea en Hive. La tabla se crea con la siguiente sentencia en lenguaje SQL:

```
CREATE EXTERNAL TABLE hive (key string, usuario string, password string)  
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'  
WITH SERDEPROPERTIES  
("hbase.columns.mapping" = ":key, info: user, info: pass") campos de Hbase  
TBLPROPERTIES ("hbase.table.name" = "hbase");
```

Con la creación de la tabla externa de nombre 'hive' ya se encuentran conectadas las dos aplicaciones, ahora se podrá conocer o ingresar datos a través del lenguaje SQL desde la terminal de Hive.

## Anexo B. Guía para la implementación del caso de estudio

El caso de estudio que se va a implementar dentro del ambiente Big Data, es el modelo de una base de datos relacional sobre el manejo de las transacciones bancarias, al cual se realizara la transformación a una estructura de columnas para adaptarla al motor NoSQL HBase.

Los pasos para la ejecución del caso de estudio son los siguientes:

1. Estructura de Bases de datos NoSQL orientadas a columnas.
2. Transformación del modelo relacional al modelo orientado a columnas.
3. Creación de tablas en el motor NoSQL Hbase.
4. Insertar datos en el ambiente configurado.

### 1. Estructura Bases de datos NoSQL orientada a columnas

“Las bases de datos orientada a columnas son muy similares a las bases de datos relacionales, pero tienen algunas diferencias en la estructura y en el manejo de los datos; por ejemplo en el almacenamiento, las bases de datos relacional almacenan por filas, mientras las orientadas a columnas almacenan los datos a través de columnas, lo cual permite unir información del mismo tipo dentro de una misma tabla para optimizar su consulta”<sup>88</sup>.

Los siguientes conceptos son fundamentales para entender cómo funcionan las bases de datos orientadas a columna:

- “Familias columna. Es la forma de almacenar los datos en el disco. Todos los datos en una sola columna de familia se guardan en el mismo archivo. Una familia de columna puede contener súper-columnas o columnas.
- Súper-columna. Es una columna que contiene un conjunto de columnas.
- Columna. Es una tupla de nombre y valor”<sup>89</sup>.

A continuación se puede observar un ejemplo para comprender la forma de almacenar los datos en las bases NoSQL orientada a columnas:

---

<sup>88</sup> AYENDE@RAHIEN. Column Family [en línea]. California: Ayende [citado 10 Noviembre, 2013]. Disponible en internet: <<http://ayende.com/blog/4500/that-no-sql-thing-column-family-databases>>

<sup>89</sup> Ibid.

**Tabla 8. Estructura Bases de datos orientada a columnas**

Tabla	Row (clave)	Familia	Super-columna	Columna
Blog	aa/mm/dd	información	Autor, titulo, subtítulo	Valor de cada columna
		Comentario título:		Descripción del comentario
usuario	login-nombre	información	Nombre, password	Valor para cada columna

Fuente: Autores

Teniendo en cuenta el cuadro anterior, se observa que dentro de una misma tabla se pueden incluir diferentes tipos de familias (se puede tomar como tablas del modelo relacional), esto permite interpretar la estructura que tiene las bases de datos orientadas a columnas y la manera que se puede adaptar el modelo relacional a través de la unión de tablas.

## 2. Transformación del modelo relacional al modelo orientado a columnas

Para lograr pasar un modelo relacional a un modelo orientado a columnas se debe tener en cuenta dos aspectos importantes: cuales son los datos más primordiales dentro de la base de datos y cuáles son las consultas más relevantes. Esto permitirá relacionar los datos dentro de una misma familia de columnas, para poder encontrar fácilmente la información.

Es importante resaltar que existen muchas maneras para transformar el mismo modelo relacional, ya que se pueden tomar distintos caminos para la unión de los datos, esto depende de la información que se desee encontrar o saber.

En la siguiente tabla se puede observar los aspectos equivalentes entre el modelo relacional y el modelo orientado a columnas:

**Tabla 9. Comparación modelo relacional vs orientado a columnas**

Modelo Relacional	Modelo orientado a columnas
Entidades	Tablas
Atributos	Columnas
Llaves primarias	Row key

Fuente: Autores

Para realizar la transformación se tomara como ejemplo una base de datos relacional del manejo de las transacciones bancarias, la cual contiene 8 tablas,

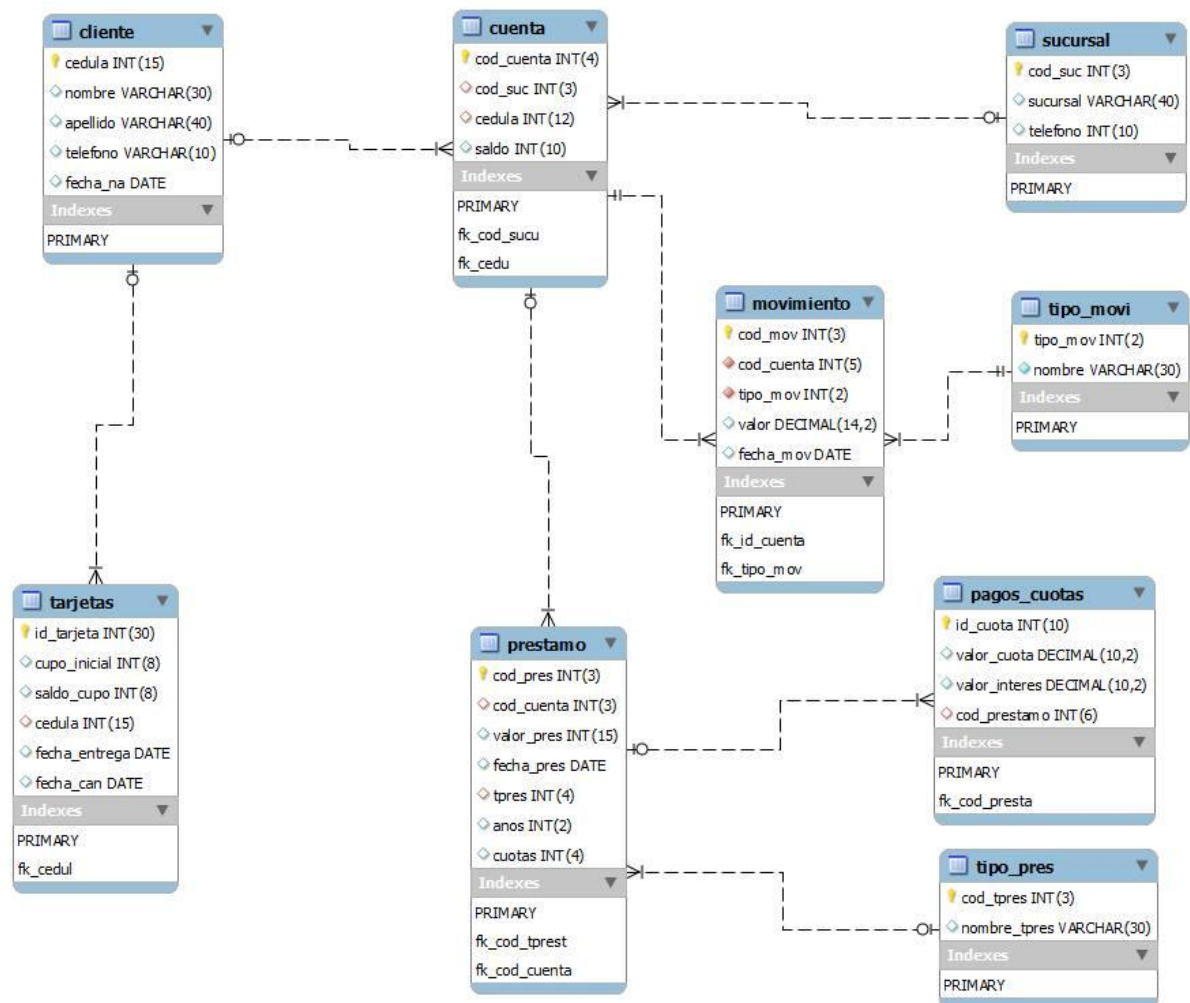


cada una de ellas con sus respectivos atributos y las relaciones que existen entre las mismas.

En este caso para la transformación del modelo se tomara como aspecto primordial las transacciones realizadas por los clientes, ignorando la sucursal en la que se encuentra asociado.

A continuación se muestra el modelo entidad relación de la base de datos:

**Figura 51. Diagrama Entidad relación**



Fuente: Autores

## Pasos para transformación

Para lograr una transformación adecuada del modelo entidad relación se deben tener en cuenta los siguientes pasos:

1. Identificar las tablas dependientes, para lograr su unión y determinar en qué orden se realizara. En el caso de estudio sería de la siguiente manera:

**Tabla 10. Tablas dependientes**

Tabla	Tablas dependientes
Préstamo	Pago_cuota
Tipo_pres	prestamo
Tipo_movi	movimiento
Cuenta	Tipo_pres y tipo_movi
Cliente	Tarjeta y Cuenta

Fuente: Autores

El cuadro anterior da a conocer que todas las tablas serán integradas dentro del cliente que sería el valor principal, del cual se desea conocer las transacciones que realiza. Pero para manejar una mejor consulta de los datos se van a crear dos tablas: cliente y cuenta.

2. Integrar las tablas dependientes dentro de las principales. Para lo cual se debe conocer las llaves primarias, que se convertirán en las “row key”. Además la tabla que se une se convierte en una familia columna de la principal. Cuando se integra una tabla con dos o más familia-columna se convierte en una super-columna de la tabla principal.

A continuación se realizara el proceso para cada tabla del caso de estudio:

En la siguiente tabla se muestra la unión entre la tabla préstamo y la tabla pago\_cuota, donde se crean dos familias: información y pago\_cuota con sus respectivas columnas.

**Tabla 11. Transformación tabla préstamo**

Tabla	Row (key)	Familia	Columnas
prestamo	Cod_pres	información	Valor, fecha, año, cuotas
		Pago_cuota	Id_cuota                      valor_cuota, valor_interes

Fuente: Autores

En la tabla N°12 se observa la unión entre la tabla tipo\_pres y la tabla préstamo, donde se crean dos familias: información y préstamo. Como la tabla préstamo tenía dos familias se crea una super-columna con nombre presta.

**Tabla 12. Transformación tabla tipo-préstamo**

Tabla	Row (key)	Familia	Super-columnna	Columnas
Tipo_pres	Cod_tpres	información		Nombre
		Prestamo	Presta	Valor, fecha, año, cuotas, Id_cuota valor_cuota, valor_interes

Fuente: Autores

En la siguiente tabla se muestra la unión entre las tablas tipo\_movi y la tabla movimiento. En esta se crean dos familias: información y movimiento con sus respectivas columnas.

**Tabla 13. Transformación tabla tipo-movimiento**

Tabla	Row (key)	Familia	Columnas
Tipo_movi	Tipo_mov	información	Nombre
		movimiento	Cod_mov, valor y fecha

Fuente: Autores

En la tabla N°14 se realiza la unión entre las tablas cuenta, tipo\_pres y tipo\_mov, allí se crean tres familias: información, préstamo y movimiento. Como las tablas tipo\_pres y tipo\_mo tenían cada una más de una familia, se crean dos super-columnas con nombre tipo\_pres y tipo\_mov. Esta tabla se crea dentro del ambiente Big Data.

**Tabla 14. Transformación tabla cuenta**

Tabla	Row (key)	Familia	Super-columnna	Columnas
Cuenta	Cod_cuenta	información		Saldo
		prestamo	Tipo_pres	Nombre, valor, fecha, año, cuotas id_cuota valor_cuota, valor_interes.
		movimiento	Tipo_mov	nombre, código, valor y fecha

Fuente: Autores

En la tabla N° 15 se realiza la unión entre las tablas cliente, cuenta y tarjeta, donde se crean tres familias: información, cuenta y tarjeta. Esta es la segunda tabla que se crea dentro del motor NoSQL Hbase.

**Tabla 15. Transformación tabla cliente**

Tabla	Row (key)	Familia	Columnas
Cliente	Cedula	Información	Nombre, apellido, teléfono, fecha_naci
		Cuenta	Cod_cuenta
		Tarjeta	Id_tarjeta, Cupo_inicial, saldo, fecha_entrega, fecha_cancelacion

Fuente: Autores

### 3. Creación de tablas en el motor NoSQL Hbase

Se crean dos tablas dentro de la consola Hbase, la tabla cliente y la tabla cuenta, lo cual permitirá conocer las transacciones que realiza el cliente teniendo como referencia las cuentas que maneja.

Para crear las tablas dentro de Hbase se realiza través del comando “create”, el cual reconoce la siguiente estructura:

create ‘nombre de la tabla’, ‘columna-familia1’, ‘columna-familia2’,..., ‘columna-familiaN’

- **Creación tabla cuenta.** Para la creación de la tabla “cuenta” se debe conocer el número de columnas-familia, las cuales son: información, préstamo y movimiento. Por lo tanto su creación se realiza a través del siguiente comando:

create ‘cuenta’, ‘información’, ‘prestamo’, ‘movimiento’

- **Creación tabla cliente.** La tabla cliente cuenta con tres columnas-familia: información, cuenta y tarjeta. Su creación se realiza a través del siguiente comando:

create ‘cliente’, ‘información’, ‘cuenta’, ‘tarjeta’

Las tablas deben quedar creadas correctamente confirmando a través del comando list como se muestra en la siguiente imagen:

**Figura 52. Creación de tablas**

```
hbase(main):002:0> create 'cuenta', 'informacion','prestamo','movimiento'
0 row(s) in 1.7970 seconds

hbase(main):003:0> create 'cliente', 'informacion','cuenta','tarjeta'
0 row(s) in 1.3120 seconds

hbase(main):004:0> list
TABLE
cliente
cuenta
hbase
3 row(s) in 0.2570 seconds
```

Fuente: Autores

#### **4. Insertar datos en el ambiente configurado**

Los datos se insertan dentro de la consola Hbase, con el comando put, el cual reconoce la siguiente estructura:

put 'nombre de la tabla', 'row key', 'familia: columna', 'valor de la columna'

Esto permite ver que para insertar los datos dentro de las tablas creadas se debe realizar el mismo proceso, por cada columna que pertenezca a la misma familia.

##### **Tabla cuenta**

Se insertan los datos a la tabla cuenta, con la referencia N° 345678, la cual será el row key de los datos creados.

##### **Familia: información**

Se crea la columna saldo con valor de \$ 2'000.000 como se muestra en la figura N°53, a través de la siguiente sentencia:

put 'cuenta', '345678','informacion:saldo','2000000'

**Figura 53. Creación columna saldo**

```
hbase(main):005:0> put 'cuenta', '345678','informacion:saldo','2000000'
0 row(s) in 0.5140 seconds
```

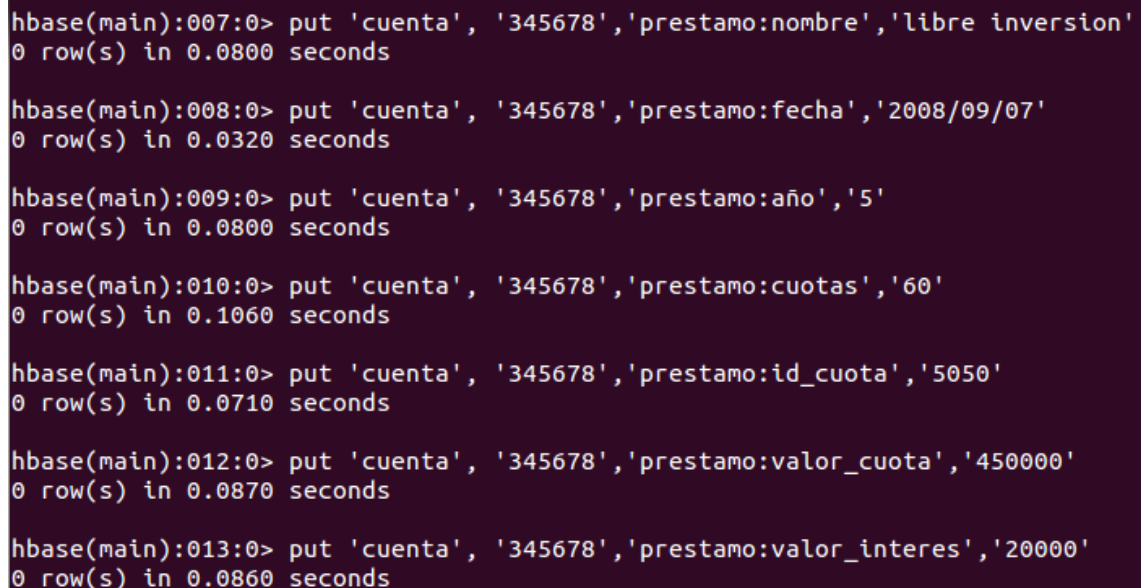
Fuente: Autores

## Familia préstamo

Se crea la Familia préstamo con sus respectivas columnas como se muestra en la figura N° 54, con los siguientes valores:

```
put 'cuenta', '345678', 'prestamo:nombre', 'Libre inversion'
put 'cuenta', '345678', 'prestamo:valor', '40000000'
put 'cuenta', '345678', 'prestamo:fecha', '2008/09/07'
put 'cuenta', '345678', 'prestamo:año', '5'
put 'cuenta', '345678', 'prestamo:cuotas', '60'
put 'cuenta', '345678', 'prestamo:id_cuota', '5050'
put 'cuenta', '345678', 'prestamo:valor_cuota', '450000'
put 'cuenta', '345678', 'prestamo:valor_interes', '20000'
```

**Figura 54. Creación familia préstamo**



```
hbase(main):007:0> put 'cuenta', '345678', 'prestamo:nombre', 'libre inversion'
0 row(s) in 0.0800 seconds

hbase(main):008:0> put 'cuenta', '345678', 'prestamo:fecha', '2008/09/07'
0 row(s) in 0.0320 seconds

hbase(main):009:0> put 'cuenta', '345678', 'prestamo:año', '5'
0 row(s) in 0.0800 seconds

hbase(main):010:0> put 'cuenta', '345678', 'prestamo:cuotas', '60'
0 row(s) in 0.1060 seconds

hbase(main):011:0> put 'cuenta', '345678', 'prestamo:id_cuota', '5050'
0 row(s) in 0.0710 seconds

hbase(main):012:0> put 'cuenta', '345678', 'prestamo:valor_cuota', '450000'
0 row(s) in 0.0870 seconds

hbase(main):013:0> put 'cuenta', '345678', 'prestamo:valor_interes', '20000'
0 row(s) in 0.0860 seconds
```

Fuente: Autores.

## Familia movimiento

Se crea la familia movimiento con sus columnas como se ilustra en la figura N° 55, con los siguientes valores:

```
put 'cuenta', '345678', 'movimiento:nombre', 'Consignacion'
put 'cuenta', '345678', 'movimiento:codigo', '0001'
put 'cuenta', '345678', 'movimiento:fecha', '2012/09/08'
put 'cuenta', '345678', 'movimiento:valor', '200000'
```

**Figura 55. Creación familia movimiento**

```
hbase(main):014:0> put 'cuenta', '345678','movimiento:nombre','consignacion'
0 row(s) in 0.0720 seconds

hbase(main):015:0> put 'cuenta', '345678','movimiento:codigo','0001'
0 row(s) in 0.1040 seconds

hbase(main):016:0> put 'cuenta', '345678','movimiento:fecha','2012/09/08'
0 row(s) in 0.0790 seconds

hbase(main):017:0> put 'cuenta', '345678','movimiento:valor','200000'
0 row(s) in 0.0670 seconds
```

Fuente: Autores

### **Tabla cliente**

Se insertan los datos a la tabla cliente con el número de cedula 45680580, que será la row key relacionada a la cuenta N° 345678

### **Familia: información**

Se crea la familia información de la tabla cliente con sus respectivas columnas como se muestra en la figura N°56, a través de las siguientes sentencias:

```
put 'cuenta', '45680580','informacion: nombre', 'Andrés'
put 'cuenta', '45680580','informacion: apellido', 'López'
put 'cuenta', '45680580','informacion: telefono','7789065'
put 'cuenta', '45680580','informacion: fecha_naci','1991/08/06'
```

**Figura 56. Creación familia información en el cliente**

```
hbase(main):024:0* put 'cliente','45680580','informacion:nombre','Andres'
0 row(s) in 0.2100 seconds

hbase(main):025:0> put 'cliente','45680580','informacion:apellido','Lopez'
0 row(s) in 0.0620 seconds

hbase(main):026:0> put 'cliente','45680580','informacion:telefono','7789065'
0 row(s) in 0.1780 seconds

hbase(main):027:0> put 'cliente','45680580','informacion:fecha_naci','1991/08/06'
0 row(s) in 0.0340 seconds

hbase(main):028:0> 
```

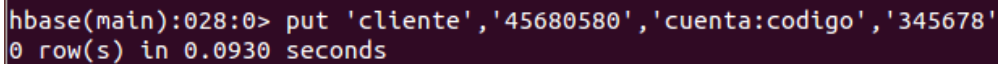
Fuente: Autores

## Familia cuenta

Se crea la columna código como se ilustra en la figura N°57, con el valor de la cuenta N° 345678, la cual será la referencia que se tendrá en la tabla cuenta para consultar las transacciones.

```
put 'cuenta', '45680580', 'cuenta:codigo', '345678'
```

**Figura 57. Creación columna cuenta**



```
hbase(main):028:0> put 'cliente','45680580','cuenta:codigo','345678'
0 row(s) in 0.0930 seconds
```

Fuente: Autores

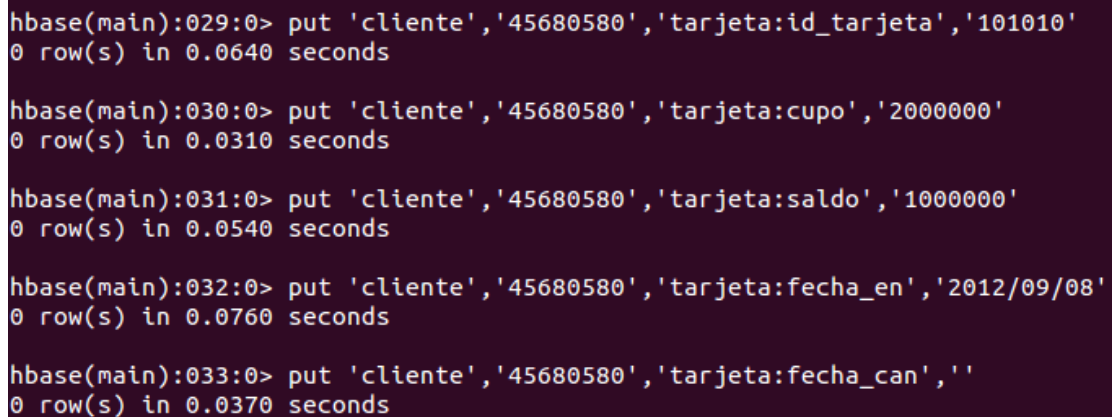
## Familia tarjeta

Por último se crea la familia tarjeta con los siguientes valores:

```
put 'cuenta', '45680580', 'tarjeta:id_tarjeta', '101010'
put 'cuenta', '45680580', 'tarjeta:cupo', '2000000'
put 'cuenta', '45680580', 'tarjeta:saldo', '1000000'
put 'cuenta', '45680580', 'tarjeta:fecha_entrega', '2012/09/08'
put 'cuenta', '45680580', 'tarjeta:fecha_cancelacion', 'dejar en blanco'
```

Su configuración se ilustra en la siguiente figura:

**Figura 58. Creación familia tarjeta**



```
hbase(main):029:0> put 'cliente','45680580','tarjeta:id_tarjeta','101010'
0 row(s) in 0.0640 seconds

hbase(main):030:0> put 'cliente','45680580','tarjeta:cupo','2000000'
0 row(s) in 0.0310 seconds

hbase(main):031:0> put 'cliente','45680580','tarjeta:saldo','1000000'
0 row(s) in 0.0540 seconds

hbase(main):032:0> put 'cliente','45680580','tarjeta:fecha_en','2012/09/08'
0 row(s) in 0.0760 seconds

hbase(main):033:0> put 'cliente','45680580','tarjeta:fecha_can',''
0 row(s) in 0.0370 seconds
```

Fuente: Autores



Se puede notar que no existe ningún inconveniente al insertar los datos en las tablas creadas, ya que las bases de datos NoSQL como Hbase no toman en cuenta los tipos de datos sino el tamaño en bytes de los mismos.

## Consultas

Para realizar una consulta sobre la consola Hbase se hace a través del comando get, el cual maneja la siguiente estructura:

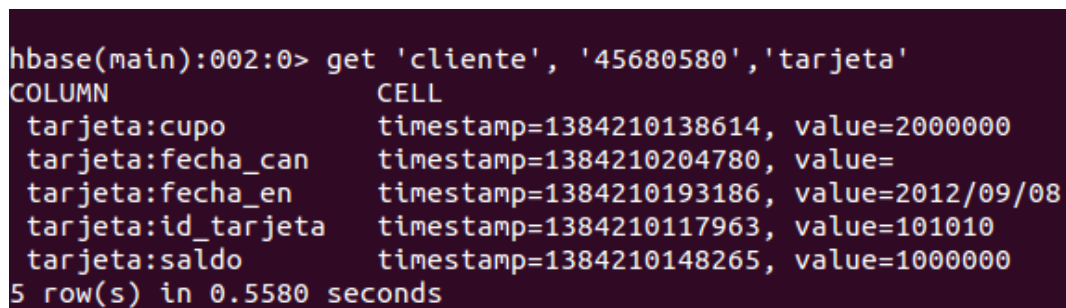
get 'nombre\_tabla', 'row key', 'nombre columna o familia'

Por ejemplo se desea saber los movimientos de la tarjeta del cliente con cedula N° 45680580, esto se lograría a través del siguiente comando:

get 'cliente','45680580','tarjeta'

Los resultados se pueden ver en la siguiente figura:

**Figura 59. Consulta de Tarjeta**



```
hbase(main):002:0> get 'cliente', '45680580','tarjeta'
COLUMN                                CELL
tarjeta:cupo                          timestamp=1384210138614, value=2000000
tarjeta:fecha_can                      timestamp=1384210204780, value=
tarjeta:fecha_en                      timestamp=1384210193186, value=2012/09/08
tarjeta:id_tarjeta                    timestamp=1384210117963, value=101010
tarjeta:saldo                         timestamp=1384210148265, value=1000000
5 row(s) in 0.5580 seconds
```

Fuente: Autores

En otro caso se desea saber todos los movimientos que tiene la cuenta N° 345678, esto se logra a través de la siguiente sentencia:

get 'cuenta', '345678', 'movimiento'

Los resultados se pueden ver en la siguiente figura:

**Figura 60. Consulta de movimientos**

```
hbase(main):004:0> get 'cuenta', '345678','movimiento'
COLUMN                                CELL
movimiento:codigo                    timestamp=1384209715162, value=0001
movimiento:fecha                     timestamp=1384209732967, value=2012/09/08
movimiento:nombre                    timestamp=1384209698332, value=consignacion
movimiento:valor                     timestamp=1384209750959, value=200000
4 row(s) in 0.4280 seconds
```

Fuente: Autores

Los ejemplos ilustran las consultas dentro de un ambiente Big Data mostrando la eficiencia en tiempos y en la cantidad de valores que se pueden traer durante un proceso normal, en comparación de lo brindado por un ambiente de base de datos relacionales.