Análisis de la información de una base de Datos Transaccional usando Hive sobre Hadoop

Mercedes Alcívar [(1),] Iván Espinoza [(2)] , Vanessa Cedeño [(3)]
Facultad de Ingeniería en Electricidad y Computación [(1)]
Escuela Superior Politécnica del Litoral
Campus Gustavo Galindo, Km 30.5 vía Perimetral
Apartado 09-01-5863. Guayaquil-Ecuador
mvalciva@espol.edu.ec [(1)] , eiespino@espol.edu.ec [(2)] ,vcedeno@espol.edu.ec[(3)]

Resumen

Debido al constante crecimiento de datos que generan las empresas hoy en día, se ha vuelto muy necesaria la búsqueda de nuevas plataformas para almacenar y analizar la información, ambientes que consuman menos recursos, que sean más escalables y que provean una alta disponibilidad. En el presente documento se explica la implementación de una solución para poder realizar el análisis de grandes volúmenes de información extraída de una Base de Datos relacional, en menor tiempo y a bajo costo. La solución consiste en el procesamiento paralelo de los datos de una base de datos de 16Gb, implementando el data warehouse HIVE sobre la plataforma HADOOP para luego realizar un rápido análisis estadístico de información transaccional y presentarlo a modo de reporte gráfico. Las pruebas realizadas con diferente número de nodos en el clúster demostraron la gran capacidad, escalabilidad, y bajo costo que tiene Hive en la ejecución de análisis del "Big Data" en comparación con Bases de Datos relacionales tradicionales.

Palabras Claves: *Hive, Hadoop, MapReduce, ESPOL.*

Abstract

Due to continued growth of data generated by businesses today, it has become necessary to find new platforms to store and analyze information, also less resource-intensive software that are more scalable and provide high availability. This document explains the implementation of a solution to perform the analysis of large volumes of information from a relational database in less time and cost. The solution is the parallel processing of data from a 16GB database, using the data warehouse Hive on the platform Hadoop and then a statistical analysis. Tests with different numbers of nodes in the cluster showed the high capacity, scalability, and low cost that has Hive in the execution of analysis of the "Big Data" compared to traditional relational databases.

Keywords*: Hive, Hadoop, MapReduce, ESPOL.*

1. Introducción

La presente investigación tiene por objetivo realizar un estudio comparativo de los tiempos de respuesta y el rendimiento entre dos herramientas que permiten el análisis de datos; un motor de base de datos relacional de paga y una infraestructura de data warehousing sobre un ambiente distribuido open source.

Generalmente las primeras versiones de los sistemas transaccionales comienzan delegando la gestión de los datos a una base de Datos relacional, esto simplifica el tiempo empleado en el desarrollo de la solución tecnológica y funciona bien durante una primera etapa, mientras la carga de requerimientos no sea tan alta. Con los años el nivel de data aumenta considerablemente y cada vez es más difícil su administración y análisis, en este punto la Base de Datos empieza a ser un problema y para poder dar servicio de buena calidad es necesario repotenciar los servidores, adquirir licencias compatibles con el nuevo hardware o en el peor de los casos cambiar toda la infraestructura por una más actual, esto funcionaria por un tiempo, pero tarde o temprano el ciclo se repetiría.

Tras reconocer la limitada capacidad de escalamiento de las Bases de Datos Relacionales, es necesario buscar soluciones alternativas que permitan aprovechar los recursos de hardware de varias máquinas que en conjunto formen un sistema distribuido.

Una primera idea que se presenta es fragmentar los datos distribuyéndolos entre diferentes Bases de Datos, esto implica complicar sustancialmente la lógica del negocio y su gestión, sin solucionar completamente el problema de la escalabilidad, ya que añadir más capacidad sigue siendo una tarea complicada. Es por ello que se suelen buscar alternativas que simplifiquen la escalabilidad del sistema.

Hadoop es una plataforma que proporciona escalabilidad horizontal, basta con agregar más máquinas al sistema para añadir más capacidad, lo cual se realiza de manera transparente y sin complicaciones.

.

2. Descripción del Problema

Se tiene una base de datos transaccional con información comercial desde el año 2003 hasta la fecha y se necesita mejorar los tiempos de respuesta de ciertos reportes que toman demasiado tiempo al ser ejecutados, sin que para esta mejora sea necesario incurrir en una inversión económica en adquisición de nuevo hardware y licenciamiento.

3. Fundamentos Teóricos de Hadoop

Hadoop es un framework que de manera transparente provee fiabilidad y manejo de grandes volúmenes de datos a las aplicaciones, ya que implementa un paradigma computacional llamado MapReduce, donde la aplicación es dividida en varios y pequeños fragmentos de tareas, los cuales son ejecutados o re-ejecutados dentro de algún nodo del clúster. Adicional a esto, cuenta con un sistema de archivos distribuido (HDFS) que almacena los datos en los nodos, proporcionando un gran ancho de banda a través del clúster. Tanto el MapReduce como el HDFS están diseñados de tal manera que si un nodo falla, el framework se hace cargo de manera automática.

En resumen, Hadoop es accesible, escalable, robusto y simple; tolerante a fallos y distribuido, capaz de almacenar y administrar cualquier tipo y volumen de datos.

4. Componentes de Hadoop

Hadoop es un framework que ejecuta aplicaciones en grandes clústeres de hardware dedicado. El framework proporciona a las aplicaciones fiabilidad y movilidad de datos de forma transparente. Implementa un paradigma computacional llamado MapReduce, donde la aplicación se divide en muchos pequeños fragmentos de trabajo, cada uno de los cuales se pueden ejecutar o volver a ejecutar en cualquier nodo del clúster.

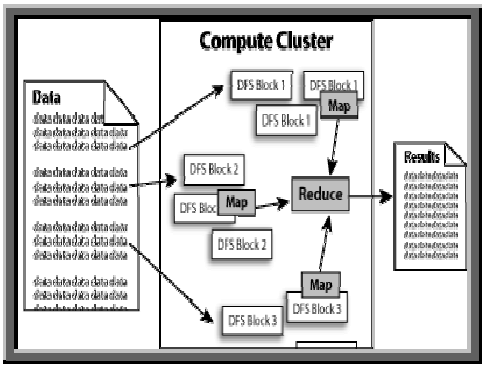

Figura 1. Generalidades Hadoop

Además, proporciona un sistema de archivos distribuido que almacena los datos en los nodos de cómputo, produciendo un alto ancho de banda agregado en todo el clúster. Ambos, MapReduce y el sistema de archivos distribuidos, están diseñados de

manera que las fallas de nodo se gestionan automáticamente mediante el framework.

5. Hive

Apache HIVE es una infraestructura de almacenamiento de datos construida sobre Hadoop para proveer resumen de datos, consultas y análisis de grandes volúmenes de datos almacenados en Hadoop o en sistemas de archivos distribuidos compatibles.

Hive define un lenguaje tipo SQL llamado HiveQL que da soporte a los usuarios familiarizados con SQL para realizar consultas de los datos. Al mismo tiempo, también permite a los programadores que trabajan con el framework de MapReduce el poder conectarse con sus mappers y reducers de tal manera que puedan realizar un análisis más sofisticado de la información. HiveQL tiene un alcance limitado a consultas tipo insert, select y create table, cuenta con soporte básico para índices, tales como bitmap index para acelerar los queries, pero carece de soporte para transacciones y vistas, se interrelaciona con Hadoop a través de un compilador que traduce la sentencia en un DAG de los jobs de MapReduce.

6. Análisis de la solución

6.1 Requerimientos Funcionales.

La Herramienta escogida debe permitir la consulta y carga de grandes cantidades de información, al momento una base de datos de 16Gb con múltiples tablas y registros que a futuro puede crecer.

Debe brindar alguna forma de importar la información desde la base de datos relacional.

El tiempo de respuesta de las consultas realizadas debe ser menor en comparación al obtenido en los servidores de base de datos relacional.

El resultado de las consultas realizadas debe generar un archivo plano, con los datos en un formato legible, que pueda ser leído por la aplicación web que genera los gráficos estadísticos.

6.2 Requisitos No Funcionales.

Escalabilidad: La posibilidad de una infraestructura que pudiera escalar junto con el crecimiento de los datos de forma sencilla.

Velocidad: Es decir poder aumentar la velocidad de la extracción de la información y lograr hacer consultar que se ejecuten en el menor tiempo posible.

Economía: Con las bases de datos tradicionales a medida que el tiempo transcurre se hace cada vez más costoso su mantenimiento, por lo tanto la herramienta que se debe usar debe ser lo más económico a pesar de los años.

7. Diseño de la solución

En esta sección se explicará todo lo referente al diseño de nuestra solución para realizar el análisis de la información de la Base de Datos Transaccional usando Hive, además Se describe el diseño General utilizando el paradigma MAPREDUCE sobre Hadoop.

7.1 Diseño General

A continuación presentamos los principales puntos de diseño:

Dataset: Son los archivos de entrada TXT (input) y los resultados del proceso MapReduce (output).
Clúster: Son los recursos computacionales necesarios para correr nuestra solución. Se implementa la computación paralela dada por Hadoop donde los Map y los Reduce se ejecutan.
Request: Cuando se ejecutan las consultas solicitadas.
Response: Es la respuesta que el clúster provee a la aplicación. En el response vienen los datos necesarios para generar un gráfico.

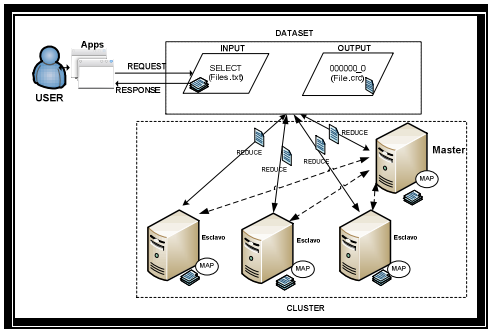

Figura 2. Diseño de la Solución.

7.2 Diseño MapReduce Implementado por Hadoop.

El proceso MapReduce se inicia con la consulta escrita por el usuario en Hive, entonces busca las coincidencias y retorna un archivo formado por el resultado del queries. El diseño general se muestra en el gráfico:

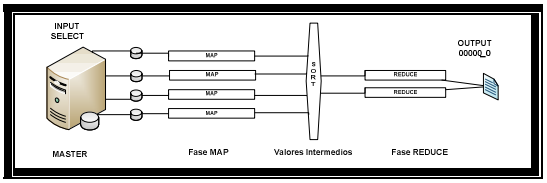

Figura 3. Proceso Map-Reduce.

Fase Map: El map de la solución trabaja sobre tres parámetros: las tablas que forman parte de la consulta, los campos elegidos y los campos de unión entre las tablas (debido al uso de un "join"). Internamente la función Map toma cada tabla seleccionada y realiza una búsqueda de los campos solicitados por el usuario en el contenido de la consulta, generando de esta forma pares <clave, valor>, en donde la clave corresponde al campo que se repite en las tablas del join y el valor contiene los campos del resultado del select ingresado.

Fase Reduce: Cada función reduce se encarga de concatenar todas las ocurrencias encontradas en la misma tabla. Generando así los pares <Tabla, ocurrencia | ocurrencia2 |……>.

7.3. Modelamiento de Datos para su Procesamiento

Para poder realizar nuestra solución debemos realizar un proceso de extracción de la información de las tablas de la Base de Datos hacia archivos textos.

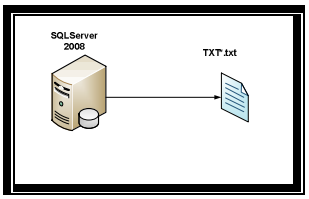

Figura 0. Conversión de Archivos de Entrada.

8. Implementación de la solución

Para realizar el análisis de la información de la Base de Datos transaccional que usamos en este proyecto hemos dividido el proceso en 6 fases:

1.-**Instalación del Sistema Operativo**Dado que la herramienta elegida para el proyecto necesita funcionar sobre un ambiente Linux, es necesario instalar el sistema operativo CentOS 6.0 y crear el usuario Hadoop (en cada una de los nodos/máquinas), el cual nos servirá para trabajar con el clúster.

2.-**Configuración de Hadoop y Hive:** Una vez iniciada el CentOS ingresamos con el usuario Hadoop, y luego de haber instalado Java, configuramos Hadoop y Hive, para esto extraemos en el home del usuario los archivos hadoop-0.20.203.tar.gz y hive-0.7.0.tar.gz, los cuales pueden ser descargados de http://hadoop.apache.org/common/releases.html y http://hive.apache.org/releases.html, respectivamente.

En todas las máquinas del clúster realizamos las siguientes configuraciones necesarias:

En el archivo Hadoop-env.sh colocar el PATH del JDK que es JAVA_HOME=/usr/lib/jvm/java-1.6.0-openjdk-1.6.0.0/jre.

En el archivo Hdfs-site.xml las rutas para la información del Namenode y el Datanode:

```
<property>
   <name>dfs.replication</name>
   <value>1</value>
 </property>
 <property>
   <name>dfs.name.dir</name>
   <value>/home/hadoop/dfs/name</value>
 </property>
 <property>
   <name>dfs.data.dir</name>
   <value>/home/hadoop/dfs/data</value>
</property>
```

En el archivo Core-site.xml el nombre del nodo Master:

```
<property>
   <name>fs.default.name</name>
   <value>hdfs://FIEC-
SOFTWARE00:8020</value>
</property>
```

En el archivo Mapred-site.xml el nombre de la máquina Jobtracker que en este caso será la máquina master, y el directorio.

```
<property>
   <name>mapred.job.tracker</name>
   <value>FIEC-SOFTWARE00:8021</value>
</property>
<property>
   <name>mapred.system.dir</name>
<value>/home/hadoop/mapred/system</value>
</property>
```

Además en la máquina Master en el archivo masters colocamos el nombre de la máquina master, y en el archivo slaves colocamos todos los nombres de las máquinas esclavos, en las máquinas esclavos estos archivos deben estar vacíos.

Todas las máquinas del cluster deben tener acceso automático entre sí y consigo misma a través del ssh con el usuario hadoop sin necesidad de ingresar la clave, para esto primero en cada máquina ingresamos las IP y los nombres de todas las máquinas del cluster en el archivo /etc/hosts, luego realizamos los siguientes pasos en la consola:

```
ssh-keygen -t dsa -P '' -f ~/.ssh/id_dsa
cd
cd .ssh
scp id_dsa.pub hadoop@fiec-software02:~/
ssh hadoop@fiec-software02
cat id_dsa.pub >>.ssh/authorized_keys
chmod 700 ~/.ssh
chmod 600 ~/.ssh/authorized_keys
```

Luego de terminar la configuración, ya estamos listos para empezar a trabajar con Hadoop, nos colocamos en la carpeta donde instalamos Hadoop en la máquina master y realizamos el format al Namenode con el comando bin/hadoop namenode -format y se crean los directorios /home/hadoop/dfs/name en el master y /home/hadoop/dfs/data en los esclavos.

Después levantamos los servicios del HDFS en la máquina master con el comando bin/start-dfs.sh

Y levantamos el servicio del MapRed para iniciar el jobtracker en el master y el tasktracker en los nodos con el comando bin/start-mapred.sh.

Si se necesitan detener los servicios sustituimos el START por STOP en los comandos anteriores.

Es importante recordar que Hadoop proporciona varias interfaces web en las que se pueden consultar el estado y los distintos resultados de nuestros programas. Para ver si se levantaron correctamente los nodos http://<localhost>:50030/jobtracker.jsp.
Para verificar el estado del sistema de archivos http://<localhost>:50070/dfshealth.jsp.

Luego de verificar que está corriendo correctamente nuestro cluster, procedemos a levantar Hive, primero realizamos las configuraciones iniciales desde el directorio de Hive con los siguientes comandos en la consola:

```
export HIVE_HOME=/home/hadoop/hive-0.7.0
(path hive)
export PATH=$HIVE_HOME/bin:$PATH
export HADOOP_HOME=/home/hadoop/hadoop-
0.20.203.0    (PATH hadoop)
Crear las carpetas en el HDFS para HIVE
 - $HADOOP_HOME/bin/hadoop fs -mkdir /tmp
 - $HADOOP_HOME/bin/hadoop fs -mkdir
/user/hive/warehouse
```

Procedemos a levantar Hive con el comando $HIVE_HOME/bin/hive y ya estamos listos para empezar a trabajar con Hive.

3.-**Extracción de la información de la Base de Datos SQLServer 2008 en formato TXT**: Ingresamos al SQL Server 2008, elegimos la base datos de la que vamos a extraer la información y damos clic derecho en Tasks → Export Data.

Se debe ingresar el origen de los datos, servidor y base de datos, acompañado con la autenticación respectiva.

Paso seguido se define como destino el archivo de texto y la ubicación donde se lo va a guardar.

Se procede a elegir lo que se va a exportar, tablas/queries

Para finalizar elegimos la tabla que se desea exportar, verificando que el delimitador de columnas sea la coma, necesaria para la importación en Hive.

Verificamos el resultado de la operación donde se muestra el número de registros exportados y el estado de cada etapa de exportación.

Se ejecuta el mismo proceso de exportación con las 4 tablas restantes, requeridas para poder realizar los queries

4.-**Replicar la información para los análisis en HIVE:** Para poder importar las tablas exportadas desde la base de datos origen, es necesario crearlas en Hive, debemos considerar un detalle muy importante, el separador que va a tener cada campo, en este caso, la coma:

```
CREATE TABLE sci_kardex (
cod_empresa int,
cod_local string,
nom_inventario string,
val_precio_distribuidor int,
por_desc_politica int,
cod_politica_descto string)
  ROW FORMAT DELIMITED
    FIELDS TERMINATED BY ','
  STORED AS TEXTFILE;
```

Luego de copiar las tablas en formato texto al path /bases/, procedemos a cargar la información a Hive con el siguiente comando:

```
hive> LOAD DATA LOCAL INPATH
'./bases/sci_kardex.txt'
    OVERWRITE INTO TABLE sci_kardex;
```

Hive enviará un mensaje indicando si la carga se ha realizado con éxito.

5.-**Realizar los Queries requeridos:** Se ingresan los queries en Hive indicando el directorio en el que queremos el resultado del query. La ejecución de cada query dará como resultado un archivo llamado 00000_0 en el directorio indicado.

6.-**Presentar en forma gráfica el resultado de los queries**: A fin de presentar una funcionalidad adicional a la herramienta analizada en este proyecto, se realizó la implementación de una pequeña aplicación web, desarrollada con PHP, que presenta un gráfico estadístico de los archivos generados en las pruebas.

FIGURA 5. Visualización de los datos del archivo subido en el servidor.

La aplicación tiene como función principal el leer los archivos resultantes de los queries ejecutados en el ambiente Hive y generar un archivo XML que es utilizado por AmCharts para generar un gráfico estadístico.

FIGURA 6. Gráfico de líneas.

9. Pruebas

Para crear el clúster se usaron 4 equipos y como servidor principal se consideraron 3 configuraciones diferentes, con las siguientes características:

Tabla 1. Características de hardware del ambiente de pruebas.

	Nodos	Servidor 1	Servidor 2	Servidor 3 (pc)
CPU	Core i3 2.27Ghz	Xeon 2.4Ghz	Xeon 3.06Ghz	Core i3 2.2Ghz
Memoria	4Gb	12Gb	2Gb	4Gb
HDD	400Gb	1.2Tb RAID 5	80Gb RAID 5	500Gb

Tabla 2. Costos aproximados de implementación de cada ambiente

	Hardware	Software	Inversión
Nodos	$450 c/u	$0 (CentOS, Hadoop, Hive)	$1800
Servidor1	$6366	$923 (Windows. Server)	$8597.99
Servidor2	$3500	$1095.68 (SQL Server)	$5731.99
Servidor3 (pc)	$460	$32.31 (CAL Windows) $181 (CAL SQL Server) ***Total:*** $2231.99	$2691.99

Tabla 3. Tiempos de respuesta de los queries ejecutados en Hive

Nodo/ Query	Ventas x Sucursal	Ventas x Negocio	Ventas Full	Rentabilidad
1	60	326	561	223
2	52	237	477	156
3	37	202	384	134
4	35	193	318	124

Tabla 4. Tiempos de respuesta de los queries ejecutados en SQL Server.

Servidor/ Query	Ventas x Sucursal	Ventas x Negocio	Ventas Full	Rentabilidad
Servidor 1	5	11	33	25
Servidor 2	67	101	371	182
Servidor 3 (pc)	70	271	529	168

Tabla 5. Tiempos de respuesta de los queries Hive vs SQL Server 2008.

	Ventas x Sucursal	Ventas x Negocio	Ventas Full	Rentabilidad
HIVE 4 nodos	35	193	318	124
Servidor 1	5	11	33	25
Servidor 2	67	101	371	182
Servidor 3 (pc)	70	271	529	168

10. Análisis de los Resultados

Con la información obtenida a partir de las pruebas, podemos observar que a mayor número de nodos menor es el tiempo de respuesta al ejecutar las consultas, lo cual es más notorio a partir del segundo nodo.

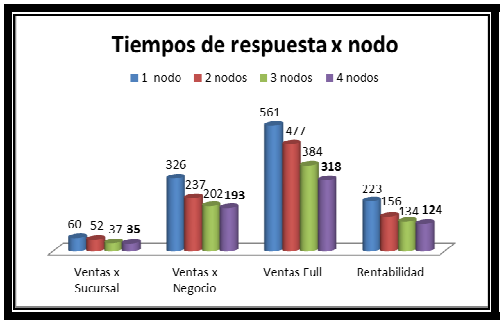

Figura 7. Ejecución de queries en Hive con distintos nodos

Luego al comparar los tiempos de respuesta de SQL Server 2008 con Hive (4 nodos) logramos mejorar a dos de las tres configuraciones de servidores.

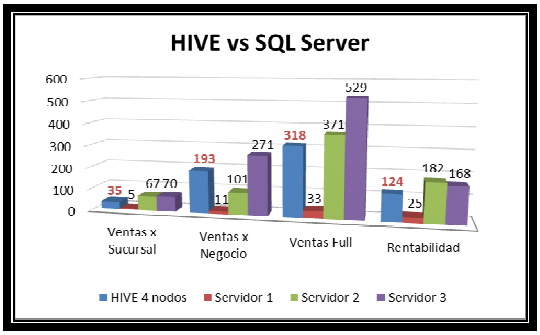

Figura 8. Ejecución de Queries Hive (4 nodos) vs SQL Server

Dado que el ambiente de pruebas solo se pudo armar con 4 nodos y con dicha configuración no fue posible equiparar al Servidor 1, consideramos que es necesario determinar el número de nodos requeridos para igualar o mejorar los tiempos de respuesta del Servidor 1, por lo tanto, haciendo uso de la extrapolación exponencial de datos pudimos obtener los siguientes resultados:

Tabla 6. Tiempos de procesamiento de Datos Hive vs SQL Server 2008

	Ventas x Sucursal	Ventas x Negocio	Ventas Full	Rentabilidad
Servidor 1	5	11	33	25
# Nodos	13	19	16	12

Mostramos a continuación los gráficos de extrapolación de uno de los queries

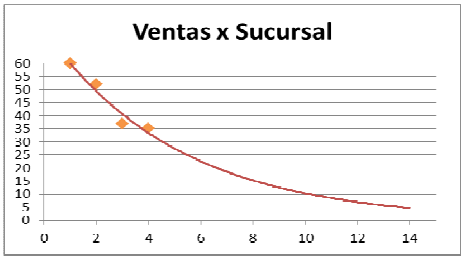

Figura 9. Extrapolación de datos del query "Ventas x Sucursal".

Tomando como muestra el número de nodos necesarios para igualar los 5 segundos de tiempo de respuesta del Servidor 1 podemos notar que sigue siendo más económica la configuración del clúster con 14 nodos

Tabla 7. Costos comparativos entre Hive (14 nodos) y Servidor 1

	Hardware	Software	Inversión
Nodos	$450 $6300 (14)	$0 (CentOS, Hadoop,Hive)	**$6300**
Servidor 1	$6366	$923 (Windows Server) $1095.68 (SQL Server) $32.3 (CAL Windows) $181 (CAL SQL Server) ***Total:*** $2231.99	**$8597.99**

Si mantenemos las mismas configuraciones durante un periodo 5 años (tiempo de vida útil de los equipos), la inversión necesaria en el caso de Windows y SQL Server 2008 (licenciamiento renovable cada 3 años mediante contrato) sería muy elevado, por lo tanto ratificamos el bajo costo de implementar un ambiente Open Source como lo es Hive sobre Hadoop

11. Conclusiones

La configuración inicial de un ambiente Hive sobre la plataforma Hadoop es, hasta cierto punto, manejable. Pero al momento de realizar las pruebas modificando el número de nodos entre cada ejecución de las consultas, se tuvo que

ejecutar en la consola administrativa algunos pasos repetitivos que no siempre daban el resultado deseado, lo cual afectó la fluidez operativa en las pruebas.

Basados en el ambiente de pruebas configurado, podemos concluir que los tiempos de respuesta en Hive (como promedio 3 nodos) son menores en comparación a los obtenidos con dos de las tres configuraciones del motor de base de datos SQL Server 2008. Para poder igualar el desempeño del Servidor 1 es necesario implementar un clúster con un promedio de 14 nodos

Los costos para implementar un ambiente basado en Hive son menores en comparación con la implementación de una Base de Datos Relacional de paga, tanto en hardware como software (licenciamiento).

El uso de Hive proporciona escalabilidad, a medida que aumentamos mas nodos al clúster, los tiempos de respuesta mejoraron en cada una de las consultas.

La creación de consultas para el análisis de la información en HIVE es sencilla, solo se necesita tener conocimientos básicos de SQL.

12. Recomendaciones

Investigar la posibilidad del uso de conexiones automáticas de HIVE con la base de datos origen. Implementar una interfaz más amigable que permita a usuarios que no sean expertos el poder realizar el levantamiento de la plataforma de manera más sencilla e intuitiva. Sería recomendable que la solución se pueda implementar en un clúster con un mayor número de nodos físicos, es decir un Datacenter especializado pre configurado con varios ambientes de la tecnología Apache Hadoop

13. Referencias

[1] Jason Venner, Pro HADOOP Build scalable, distributed applications in the cloud, Estados Unidos, 2009, Páginas.

[2] Chuck Lam, Hadoop in Action, Estados Unidos, 2011, Páginas 3-160, 246-265.

[3] The Apache Software Foundation: “Apache Hadoop” <http://hadoop.apache.org/> [Consultado: martes 4 de octubre de 2011].

[4] The Apache Software Foundation: “Apache Hive” < http://hive.apache.org/> [Consultado: martes 4 de octubre de 2011].

[5] Wikipedia <http://en.wikipedia.org/wiki/Apache_ Hadoop> [Consultado: martes 4 de octubre del 2011].

[6] VMware <http://www.vmware.com/> [Consultado: martes 4 de octubre del 2011].

[7] SlideShare <http://www.slideshare.net/glud/hadoop-en-accion-9249763> [Consultado: martes 4 de octubre del 2011].

[8] Humbug <http://www.humbug.in/serverfault/es/que-es-hadoop-y-que-es-utilizado-para--27829.html> [Consultado: martes 4 de octubre del 2011].