IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Volume-3, Issue-9 E-ISSN: 2347-2693

Query Optimization of Big Data Using Hive

A.Vinay Kumar[1*] and A. Madhuri [2]

[1*,2] Department of Computer Science and Engineering,

Prasad V Potluri Siddhartha Institute of Technology, Kanuru, India

www.ijcseonline.org

Received: Aug/19/2015 Revised: Aug/28/2015 Accepted: Sep/22/2015 Published: Sep/30/2015

***Abstract*—** Huge amounts of data are required to build internet search engines and therefore large number of machines to process this entire data. The Apache Hadoop is a framework that allows for the distributed processing of large data sets across clusters of machines. The Hadoop having two modules 1. Hadoop distributed file system and 2. Map Reduce. The Hadoop distributed file system is different from the local normal file system. The HDFS can be implemented as single node cluster and multi node cluster. The large datasets are processed more efficiently by the multi node clusters. By using the hive query language on the Hadoop and increasing number of nodes the data will be processed fastest than with the fewer nodes.

***Keywords*—** Big Data, HDFS, Map Reduce, Hive,Join

I. INTRODUCTION

Social network data analysis has become as the most essential and the plays crucial role in business intelligence and for online services providers. As the data size is increasing from Gigabytes to Petabytes, in order to handle this much amount of data by traditional databases is not an easy task. In this database they have to handle the both the transactional data and the Historical data i.e., which is stored from decades. According to present data trends the data can be of heterogeneous and homogeneous types. As the data available is in the structured and unstructured format, so in order to handle this variety of data it is not easy to handle it by relational databases. For storing this huge amount of data by relational databases is very expensive. The Data model in relational database is non-scalable. Some other issues like maintaining the consistency, two phase commit capability, basic query operations like select, update, join and other operations are not possible due to vast volumes. In order to handle this huge amount of data,big data techniques like Hadoop framework can be used. The most essential capabilities of hadoop are we can dynamically expand the number of storage servers, cost for storing this huge amount of data is negligible compared with relational databases and Hadoop can be implemented on the common commodity hardware. Basic sql operations like selection, update, and projection of data can be easily done by using the map reduce paradigm. In the map phase the selection, updating and projection of data can be done. In the reduce phase the join, deletion other sql operations can be done. In the map reduce computing process the certain job is distributed across nodes of cluster into individual tasks. But in the map reduce paradigm, cost of executing map reduce program is time taking for join operations. This can be overcome by Hive. Hive is data warehouse software that is built on the Hadoop framework architecture, which is used to summarize, analyze and allows the query processing of the dataset present in the hadoop distributed file System. Hive provides query language HiveQL which resembles as simple SQL language.

II. OVERVIEW

HADOOP

Hadoop is open source framework architecture that is introduced by the Apache software organization which is used for distributing the huge size dataset over large cluster using the commodity hardware. Hadoop comprises of two main modules i.e., Hadoop distributed file System (HDFS) and Map reduce programming paradigm.

HDFS

Hadoop distributed file System is scalable, distributed, reliable and portable file System, which is written in java platform for hadoop framework. Hadoop can be implemented in single machine or in a multiple machines. It is a block-structured file system. In this file system the data is split into small chunks, known as the blocks.

Each block size will be 64MB by default in this file system. Each block size can be increased by setting the **max.split.size** property. Here the blocks can be replicated and can be stored on multiple nodes of hadoop cluster. By the default the replication factor of hadoop distributed file system will be **three** in the multi node cluster or fully distributed mode. Using this replication factor we can achieve the reliability in the hadoop distributed file system.

HDFS will resemble the client-server architecture. It consists of a single namenode and multiple datanodes. Namenode will distribute the data processing tasks to

datanodes by using the job tracker and datanodes will accomplish the tasks assigned to them by using task tracker. Here datanodes will have to report about the replicated blocks in the form of heart beats for a certain time interval.

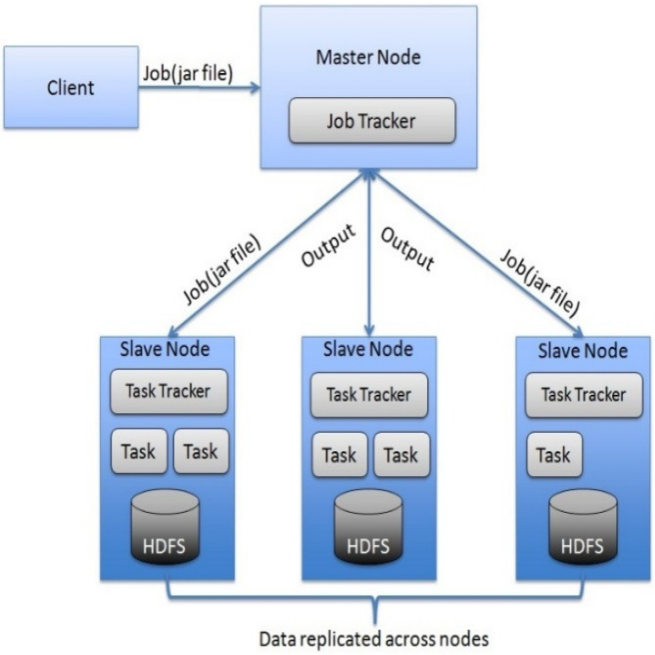

Fig 1: block replication

Datanodes will also communicate with each other for copying the blocks if there any blocks missing, so we can achieve the high availability. Namenode will save about all data processing operations in the form fsimage and editlogs. HDFS will also provide the backup for this file system i.e., in the form of secondary namenode by save the fsimage and edit logs parallel with namenode.

MAP REDUCE

Map reduce is the programming paradigm which is used to process large amount of data present in the hadoop cluster. This computing process is splits into two phases map phase and reduce phase. Map reduce will be implemented using the java API. In this programming, all the data manipulation operations like selection, projection, distinct and other operations of SQL will be done in the map phase. In the reduce phase all the other joins, union, grouping and other operations will be done in this phase. Map phase results of the program will be taken as an input to reduce phase. In the map phase it will take key/value pair as input line by line and generate the output key /value pair into context area. After that these will be taken as input key/value pair for reduce phase.

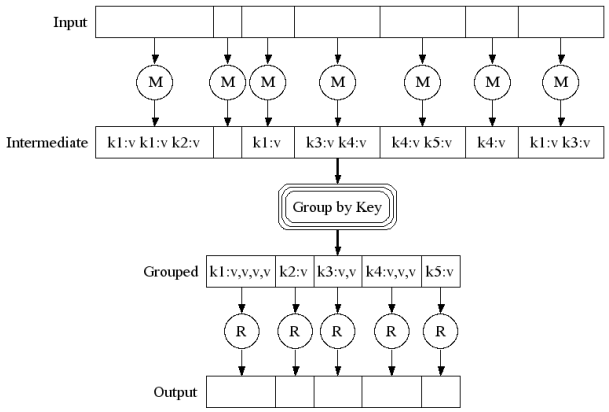

Fig 2: Mapreduce process

Hive

Hive is data warehouse software which is resides on the top of the hadoop distributed file system. This is used for summarizing, analyze and query processing of large datasets that stored in the hadoop file system. It also provides a Query language which is like SQL type language called HiveQL. It will transform the queries to map/reduce program and executes the job as usually like map reduce paradigm.

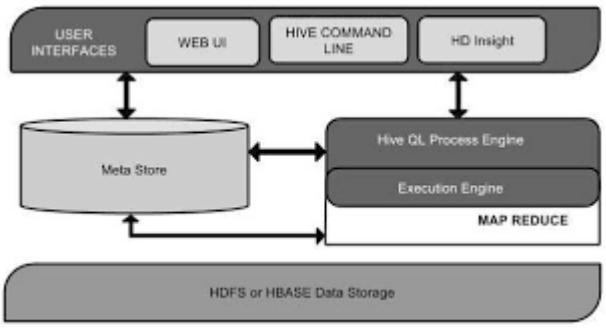

Fig 3: Hive Architecture

It is a faster processing language in which the query processing results will be obtained in less time compared to relational databases and map reduce. Here the input file given will be stored in hdfs by creating a directory with the related schema name. By default hive will store the data in the text format file. It also supports the other file formats like sequence file, orc file, csv files and rc file formats.

Hive will store the metadata of schemas in the Apache Derby database default which embedded in the hive. It will also provide the UDF (user Defined Functions) to manipulate the different types of data like string, date etc.

Apache Derby

Apache Derby is open source relational database that can be used for storing the metadata about schemas in the hive query language. Derby database provides the

embedded database Engine and Derby Network Server. Embedded database engine is a relational database engine which supports the SQL as an API. Derby Network Server is used to connect remote relational databases. It supports the JDBC, ODBC and Command line interface.

III. QUERY PROCESSING USING HIVE

HiveQL is the dialect as SQL language. In this query language the query have to pass many phases. The query is passed to respective driver. It will pass this to compiler, where the query is broken into different query blocks and expressions. This will also generate the query plan which is optimized to execution plan. Compiler will fetch metadata about related schemas and generates the query plan. This plan is optimized to generate execution plan. This plan is the map reduce job which it is passed to job tracker, whereas the job tracker will divide job to different tasks and send to task trackers of datanodes. The task trackers will process the tasks and sends the outputs to job tracker .There it will merge all outputs of all tasks and generate the s the final output to query.

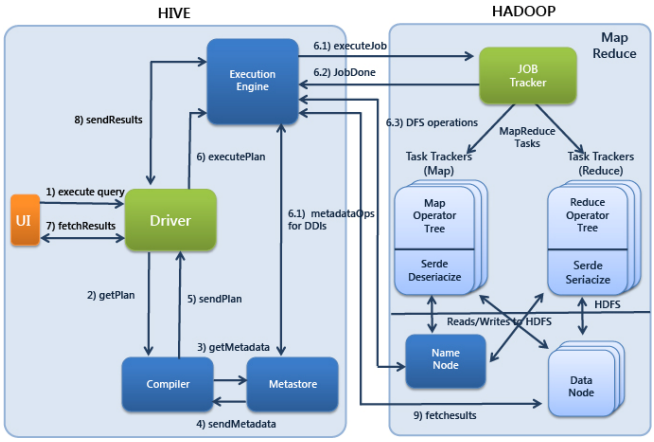

Fig 4: Working of Hive on Hadoop

Prerequisites of Hive

Installing the Java

Java is programming language which is based on class-oriented and objects oriented programming approach. It is meant for "Write Once Run Anywhere" once the java code is complied, byte code can be run on any platform that supports JVM architecture irrelevant of operating systems where it runs. Hadoop has extended the java API by adding few classes to normal java API.

Command: sudo apt-get install openjdk-1.7-jdk

Hdfs on multinode cluster

Firstly to get download the hadoop of respected version

huser@ubuntu$:~wget http://apache.claz.org/hadoop/common/hadoop-1.2.0/hadoop-1.2.0.tar.gz

In this cluster we have to configure the different properties as mentioned: **fs.default.name**to locate the host with the port number the process is running in core-site.xml.Change the default properties values **dfs.replication which has** default value **is 3**which is hdfs-site.xml. Also there is a need to set the path for namenode and datanode for storing FSImage and editlogs of HDFS by setting the properties **dfs.name.dir** and **dfs.data.dir.** Also need to set the job tracker running physical address in Mapred-site.xml

Install open ssh server

Open SSH Server is freely available secure shell protocol. This is used for controlling and transferring of files between computers. As the traditional tools such as telnet or rcp are unsecure and they reveal the password in clear text format.

huser@ubuntu$:~ sudo apt-get install openssh-server

Hive installation

Firstly download the Apache hive using the following

command:wgethttp://www.eu.apache.org/dist/hive/hive-0.14.0/apache-hive-0.14.0-bin.tar.gz

Set class path in the .bashrc file

export CLASSPATH=
$CLASSPATH:/usr/lib/vinay/hive/lib/*
We need to export the JAVA_HOME in **hive-env.sh**
export HADOOP_HOME=/usr/lib/vinay/hadoop

Hive Configuration in hive-site.xml

It is used for scratch space of map reduce job

```
<property>
        <name>hive.exec.local.scratchdir</name>
        <value>/usr/lib/vinay/hive/iotmp</value>
        <description>Local scratch space for Hive jobs</description>
</property>
<property>
        <name>hive.querylog.location</name>
        <value>/usr/lib/vinay/hive/iotmp</value>
        <description>Location of Hive run time structured log file</description>
</property>
<property>
        <name>hive.downloaded.resources.dir</name>
        <value>/usr/lib/vinay/hive/iotmp</value>
        <description>Temporary local directory for added resources in the remote file system.</description>
</property>
<property>
        <name>javax.jdo.option.ConnectionURL</name>
```

```
        <value>jdbc:derby://localhost:1527/metastore_db;create=true</value>
        <description>JDBC connect string for a JDBC metastore</description>
</property>
<property>

<name>javax.jdo.option.ConnectionDriverName</name>
        <value>org.apache.derby.jdbc.ClientDriver</value>
        <description>Driver class name for a JDBC metastore</description>
</property>
<property>
        <name>javax.jdo.option.ConnectionUserName</name>
        <value>APP</value>
        <description>Username to use against metastore database</description>
</property>
<property>
        <name>javax.jdo.option.ConnectionPassword</name>
        <value>mine</value>
        <description>password to use against metastore database</description>
</property>
<property>
        <name>hive.metastore.warehouse.dir</name>
        <value>/user/hive/warehouse</value>
        <description>location of default database for the warehouse</description>
</property>
```

Hive queries

Creating the schema or table in the hive query language

hive>create table ifnotexists emp (eid int, name String, salary float) row format delimitedfields terminated by '\t' stored as textfile;

In query, compiler will check whether table exists or not in metastore. If exists it will delete and create new table in it place. In this query it will create table with the field's eid as integer data type, name with String data type and salary with float data type. Here the row format delimited is meant to delimit each line of data file with '\n' by default, fields are terminated by '\t'.In this query not only '\t' also other types like comma (','), space ('\s') based on format of the input file. Loading the data into table or schema

Normally in the sql language data can be inserted by row by row. But here it is not possible to huge datasets row wise. So loading the data in two types first is from local file system i.e, load data local inpath '/path/to/file' overwrite into table 'table'.

hive**>load data local inpath 'home/documents/emp.txt' overwrite into table 'emp'**.

Other way to load the data into from hdfs by removing local in the load statement as below

hive>load data inpath 'hdfs:/user/hive/warehouse/emp/emp.txt' >overwrite into table 'emp';

This tells about loading the data from hdfs file system.

IV. RESULTS

In the query processing the given query will be divided into maps and the data will selected from table by the execution plan. Here the query plan will be broken into tokens nad the data will be fetched.

For instance we run a hive query on the hive query language and run the same query in traditional databases like MySQL.

select a.id,b.name,a.desig,b.station,a.pscale,a.grade,c.model, b.vehins **from** table1 a **join** table2 b on a.id=b.id **join** table3 cona.id=c.id**where**a.grade='2400';

The above query will done in the stage-1 of the query plan.

In the hive query execution plan stage-1 is used as selection of data is only done here.In the traditional databases the query processing time is 33.916 sec. But where as in the hive query language it had taken upto 30.025 sec. Here the configuration of multimode cluster plays vital role data processing. The above results are done for few datanodes i.e, no.of datanodes is 3. If we increase the no.of datanodes in the cluster it will less time compared to previous output.

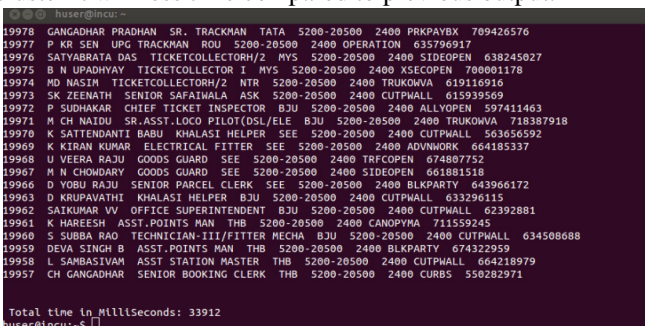

Fig. 5 MySQL query execution time

The fig.5 shows the query processing time for the given query. The performance variation can be observed if you see the hive query process also.

There is a time variance hive environment on the same query compared with the relational database. Is show in the figures 6(a) & 6(b)

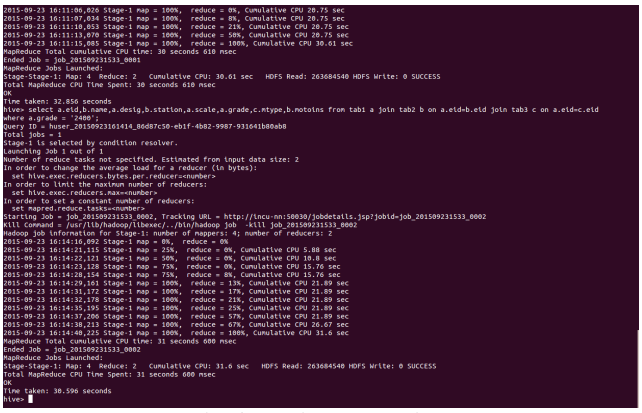

Fig.6(a) Hive query time

Hive query performance is better by seeing the variance of 4sec of time.

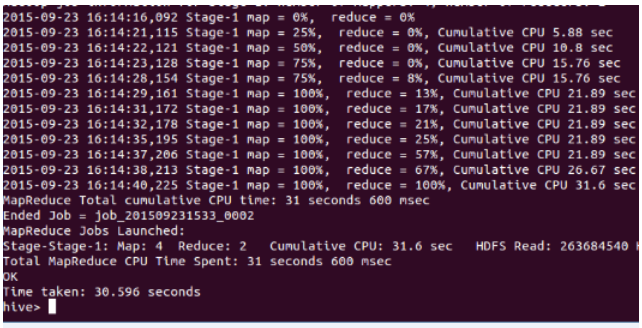

Fig.6(b) Hive query time

V. CONCLUSION

Process of huge amount of data can done very easily in the hadoop using hive. It also says that as the no.of datanodes increases we can easily process the much faster than present response times. Moreover there are other faster, scalable environments in the hadoop framework which may give better results like Tez, Spark, Pig, Hue, Oozie etc.

REFERENCES

[1] Map-Reduce-Merge: Simplified Relational Data Processing on Large Clusters by YangDasdan and Hasio ,Parker Vol-8,Issue-7,1029-1040,**2007**
[2] J.Dean and S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. In OSDI, pages 137–150, **2004**,
[3] Liu Liu, Jiangtao Yin, Lixin Gao, "Efficient Social Network Data Query Processing on MapReduce" ACM August 16, **2013**.
[4] Stephen Kaisler, Frank Armour, J. Alberto Espinosa, William Money, "Big Data: Issues and Challenges Moving Forward" 1530-1605/12, Jan **2013**.
[5] "Hadoop Mapreduce Outline in Big Figures Analytics" IJCSE,Vol-2,Issue-9 100-104,Sep **2014**.
[6] ApacheHadoop.http://hadoop.apache.org/.friday 2 Dec,**14**
[7] https://en.wikipedia.org/wiki/Apache_Hadoop, 25 Jan,**15**
[8] http://hashprompt.blogspot.in/2014/06/multi-node-hadoop-cluster-on-ubuntu-1404.html, 7 April,**2015**

AUTHORS PROFILE

A. Vinay Kumar is presently M.Tech Student, Dept. of Computer Science & Engineering, Prasad V Potluri Siddhartha Institute of Technology (Autonomous), kanuru, India.

A. Madhuri is presently Assistant Professor, Dept. of Computer Science & Engineering, Prasad V Potluri Siddhartha Institute of technology(Autonomous), kanuru, India.