International Journal on Recent and Innovation Trends in Computing and Communication
Volume: 3 Issue: 11

ISSN: 2321-8169
6262 - 6265

Hadoop Distributed file system, Hive and Its Applications: A Survey

Mr. Prashant R. Mahajan
Department of Computer Engineering,
DGOI,FOE, Daund
Savitribai Phule Pune University,
Pune. India
prashant.it18@gmail.com

Prof. Amrit Priyadarshi
Department of Computer Engineering
DGOI, FOE, Daund
Savitribai Phule Pune University,
Pune, India
amritpriyadarshi@gmail.com

***Abstract*:** Business intelligence is growing area across the industry and data getting collected and analyzed in rapid way due to which legacy warehousing tools has become very costly. Hadoop is framework which is open source and stores data and runs applications on cluster of normal i.e commodity hardware. Hadoop provides large amount of processing power and storage for various kinds of data. It is able to handle concurrent tasks or jobs. HDFS (Hadoop Distributed File System) is a distributed file system which can provide high performance data access across Hadoop cluster of servers. Due to Managing pools of big data and supporting big data analytics application HDFS has become a strong tool. Developer has to write custom programs in map reduce programming model which are difficult to maintain and reuse. Hive is open source solution built on top of hadoop which is used as data ware house. Hive supports HiveQL which is SQL-like language, which are compiled into mapreduce jobs to be executed on Hadoop.

*****

1. Introduction

As HDFS is deployed on normal low cost (Commodity) hardware so server failure is common.

Hadoop file system is designed to be fault tolerant and it facilitates fast transfer of data between computer nodes. It also continues running even if node fails, which decreases the risk of sudden great damage/failure, even if nodes failing are large in numbers. For allowing parallel processing, HDFS breaks the information into small pieces and distributes them across computer nods in the cluster. This file system also provides the facility to copy each piece of data multiple time on different nodes making sure at least one copy should get placed on different rack that the others. Because of which when node fails it continues processing by finding same copy within cluster.

Fig1. Hadoop System

Apache Foundation provides the framework which includes four main modules: Hadoop Common utilities and libraries which are used by other modules. HDFS (Hadoop Distributed File System) is Java based system which stores the data on cluster of machines. MapReduce is a custom software development model which enables parallel data processing. YARN is resource management framework which is used for scheduling and handling resource requests from distributed applications that runs on top od hadoop. (YARN is an acronym for Yet another Resource Negotiator.)

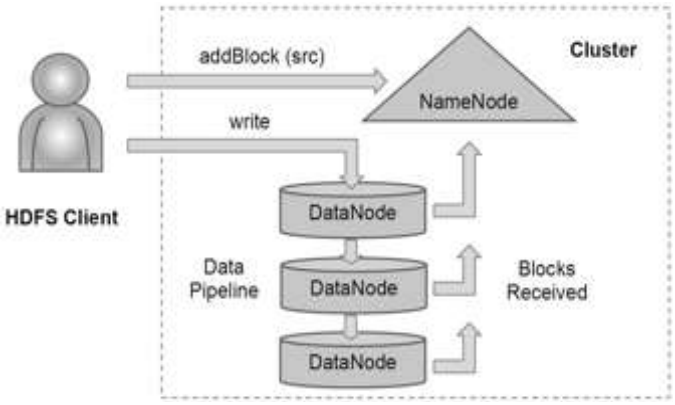

Fig2. HDFS Architecture

There are some other software components which runs on top on Hadoop and have became a top level apache project which includes Hive – a SQL like query language and a data warehousing which presents data in the form of tables. Hive programming is similar to database programming. HBase is the nonrelational database which runs on top of

hadoop. HBase serves as input and output of maproduce jobs. Pig is a platform for manipulation data stored in HDFS. Pig is a dataflow language. Pig contains MapReduce compiler and high level Pig Latin language. It provides a simple way to perform data extarctions, transformation,and loading (ETL).

2. Related Work

The HiveQL(Hive query language) consist of a subset of SQL and there are extensions that are useful in hadoop environment. Traditional SQL features like from clause subqueries, various types of joins – inner, left outer, right outer and outer joins, Cartesian products, group bys and aggregations, union all , it also supports create table as select and many useful functions on primitive and complex types make the language very SQL like. Actually many features of HQL are exactly same as SQL. So it becomes so easy for anyone familiar with SQL to start working with Hive command line interface and begin querying the system right away. Show tables and describe are the useful metadata browsing capabilities also present and so are explain plan capabilities to inspect query plans. There are some limitations e.g. only equality predicates are supported in a join predicate and the joins have to be specified using the syntax such as

```
SELECT b1.c1 as c1, a1.b1 as c2
FROM t1 JOIN t2 ON (b1.a2 = a2.b2);
instead of the more traditional
SELECT b1.c1 as c1, a1.b1 as c2
FROM t1, t2
WHERE t1.a2 = t2.b2;
```

Hive currently does not support inserting into an existing table or data partition and all inserts overwrite the existing data. Another limitation is in how inserts are done. Accordingly, we make this explicit in our syntax as follows:

```
INSERT OVERWRITE TABLE s1
SELECT * FROM s2;
```

Most of the time these problems are never been the restrictions. There is rarely seen a case where the query cannot be written as an equi-join and since most of the data is loaded into warehouse daily or hourly, Simply loading the data into a new partition of the table for that day or hour. However, there is a case that with more frequent loads the number of partitions can become very large and may require implementing INSERT INTO semantics. The absence of UPDATE , INSERT INTO and DELETE in Hive allows to use very simple way to deal with writer ,reader concurrency without implementing any type of locking protocols. Although these restrictions are there, HiveQL also has extensions which supports analyzing map-reduce program in any programming language of user's choice. Because of this it becomes easy for users to write a complex logic in MapReduce programs which can be used in Hive queries easily.

3. Storage features and File Formats

Conside the example that test_table gets mapped to <root_directory>/test_table in hdfs. Here root_directory is specified by the warehouse root directory parameter in hive-site.xml. By default this parameter's value is set to /user/hive/warehouse. Table metadata associates the data in a table to hdfs directories, while the tables are logical data units in Hive. The primary data units and their mappings in the hdfs name space are as follows:

• Partitions – A partition of the table is stored in a subdirectory within a table's directory.

• Tables – A table is stored in a directory in hdfs.

• Buckets – A bucket is stored in a file within the partition's or table's directory depending on whether the table is a partitioned table or not.

A table can be expressed as partitioned or non-partitioned. A partitioned table is created by specifying the PARTITIONED BY clause in the CREATE TABLE statement.

CREATE TABLE Test(d1 string, d2 int) PARTITIONED BY (ds string, hrint);

In above example the partitioned table shall be stored in /user/hive/warehouse/Test directory in HDFS. A partition exists for every distinct value of ds and hr specified by the user .We should note that the columns on which partitioning is done are not part of the table data and the partitioned column values are encoded in the directory path of that partition and they are also stored in the table metadata. A new partition can be created through an ALTER statement or INSERT through an statement that adds a partition to the table. Both the following statements

```
ALTER TABLE Test ADD PARTITION
(ds='2009-02-02', hr=11);
INSERT OVERWRITE TABLE
Test PARTITION (ds='2009-01-01', hr=12)
SELECT * FROM Test;
```

Above statements adds a new partition to the table Test. The INSERT statement also populates the partition with data from table Test, whereas the alter table creates an empty partition. Both these statements end up creating the corresponding directories in the table's HDFS directory. While evaluating query the Hive compiler is able to use this information to find the directories that need to be scanned for data.

Hadoop files can be stored in different formats. From a fileformat in Hadoop we can determine how records are stored in a file. Text files are stored in the TextInputFormat and binary files can be stored asSequenceFileInputFormat. The format can be specified when the table is created. Users are able to implement their own file formats in Hadoop. Hive does not have any restrictions on the type of file input formats that the data is stored in. Hive also provides an RCFileInputFormat which is able to store the data in a column oriented manner. column orientation can give high performance improvements while queries that do not access all the columns of the table. Users can add their own file formats and associate them to a table.as

4. System Architecture and Components

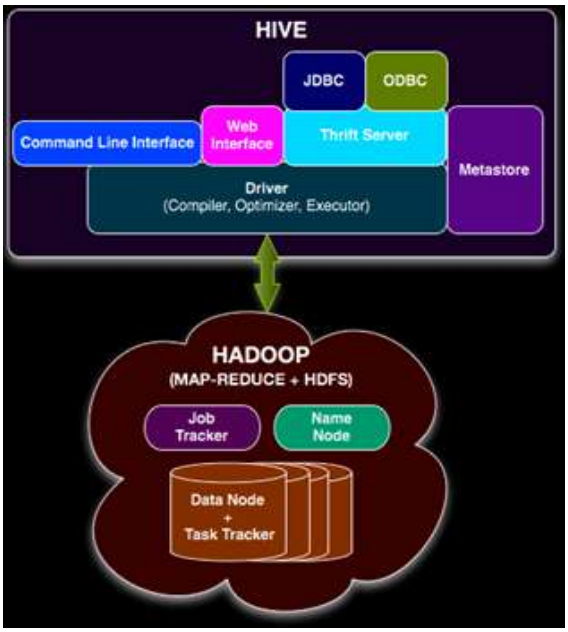

Fig.3. Hive System Architecture.

An external client such as thrift, odbc or jdbc interfaces can be used to submit HiveQL statement via the CLI, the web UI or The driver first passes the query to the compiler where it gets typically parse, type checked and semantic analysis phases, using the metadata stored in the Metastore. The compiler used to generates a logical plan which is then optimized through a simple rule based optimizer. Finally an optimized plan in the form of a map-reduce tasks and HDFS tasks is generated. The execution engine will be executing these tasks in the order of their dependencies, using Hadoop.

Metastore is very critical for Hive. Without the system catalog it is not possible to impose a structure on hadoop files. The Metastore is acting as the system catalog for Hive. It stores all the information about the tables, their partitions, the schemas, the columns and their types, the table locations etc. The execution plan can be generated by using metadata stored in the Metastore. Similar to compilers in traditional databases, the Hive compiler processes HiveQL statements in the following steps:

Parse, Type checking and semantic Analysis, Optimization. Finally execution of the tasks is done in the order of their dependencies. if all of its prerequisites have been executed then each dependent task is only executed. The final results are stored in a temporary location. At the end of the entire query, the final data is moved to the desired location in case of DMLs. In the case of queries the data is served as such from the temporary location. First serialization takes place of a map/reduce task ,its part of the plan into a plan.xml file. This file is then added to the job cache for the task and instances of ExecMapper and ExecReducers are using Hadoop. Each of these classes deserializes the plan.xml and executes the relevant parts.

4. Applications

a. As Reporting tool

Hive can be used for creating a reports using HiveQL on top of the hadoop distributed file system. HQL scripts runs on top HDFS to extract data and provides reports. Scripts can be scheduled using oozie to run on any particular time as per the business requirement.

b. Data Warehouse

Hive is a data warehouse structure which is build on top of Hadoop which is used for ad-hoc querying, data summarization, analysis. Hive supports indexing to provide acceleration. Hive support plain text, RCFiles, HBase as different storage types, and others. As Hive stores metadata in an RDBMS so it reduces significant time to perform the semantic check during the query execution.

5. Conclusion

HDFS and Hive are very important and advance tools that can be effectively used for reporting and as data warehouse. Using HDFS and Hive , we can save much time and money as Hadoop infrastructure is open source.

Acknowledgement

I express great many thanks to Prof. Amrit Priyadarshi and Prof. Sachin S. Bere for their great effort of supervising and leading me, to accomplish this fine work. To college and department staff, they were a great source of support and encouragement. To my friends and family, for their warm, kind encourages and loves. To every person who gave me something too light along my pathway. I thanks for believing in me.

References

[1] Apache Hadoop. Available at http://wiki.apache.org/hadoop.
[2] Facebook Lexicon at http://www.facebook.com/lexicon.
[3] Hive wiki at http://www.apache.org/hadoop/hive.
[4] Hadoop Map-Reduce Tutorial at http://hadoop.apache.org/common/docs/current/mapred_tutorial.html.
[5] Hadoop HDFS User Guide at http://hadoop.apache.org/common/docs/current/hdfs_user_guide.html.
[6] 6] Mysql list partitioning at http://dev.mysql.com/doc/refman/5.1/en/partitioning-list.html.
[7] Apache Thrift. Available at http://incubator.apache.org/thrift.
[8] DataNucleus .Available at http://www.datanucleus.org.
[9] A. Pavlo et. al. A Comparison of Approaches to Large-Scale Data Analysis. In Proc. of ACM SIGMOD, 2009.
[10] Hive Performance Benchmark. Available at http://issues.apache.org/jira/browse/HIVE-396
[11] TPC-H Benchmark. Available at http://www.tpc.org/tpch
[12] Running TPC-H queries on Hive. Available at http://issues.apache.org/jira/browse/HIVE-600
[13] Vinod, P. Suri P, "Maintaining a Binary Search Tree Dynamically. Proceedings of the 10th International Conference on Information Visualization", London, UK, 2006.
[14] J. Derryberry, D. D. Sleator, and C. C. Wang, "A lowerbound framework for binary search trees with rotations", Carnegie Mellon University, 2005.

Authors

Mr. Prashant R. Mahajan received his B.E. degree in I.T from University of Pune in 2011. He has 3.5 years of experience of working with MNCs in Pune and Mumbai. He is currently working toward the M.E. Degree in Computer Engineering from University of Pune. His research interests lies in Data Mining, Software Engineering and Business Process Management.

Mr. Amrit Priyadarshi received his B.E. degree in Electronics Engineering and MTech in Computer Science and Engineering. He has 10 years of experience as Assistant professor and he is currently perusing his Phd Degree.