International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE)
ISSN: 0976-1353 Volume 24 Issue 3 – FEBRUARY 2017 (SPECIAL ISSUE).

BIG DATA ANALYTICS USING HADOOP TOOLS – APACHE HIVE VS APACHE PIG

Prof R.Angelin Preethi [#1] and Prof J.Elavarasi [*2]

[#] *Department of Computer Science, Kamban College of Arts and Science for Women, TamilNadu, India*

[*] *Department of MCA,Shanmuga Industries Arts& Science College, TamilNadu, India*

***Abstract*— Big data technologies continue to gain popularity as large volumes of data are generated around us every minute and the demand to understand the value of big data grows. Big data means large volumes of complex data that are difficult to process with traditional data processing technologies. More organizations are using big data for better decision making, growth opportunities, and competitive advantages. Research is ongoing to understand the applications of big data in diverse domains such as e-Commerce, Healthcare, Education, Science and Research, Retail, Geoscience, Energy and Business. As the significance of creating value from big data grows, technologies to address big data are evolving at a rapid pace. Specific technologies are emerging to deal with challenges such as capture, storage, processing, analytics, visualization, and security of big data. Apache Hadoop is a framework to deal with big data which is based on distributed computing concepts. The Apache Hadoop framework has Hadoop Distributed File System (HDFS) and Hadoop MapReduce at its core. There are a number of big data tools built around Hadoop which together form the 'Hadoop Ecosystem.' Two popular big data analytical platforms built around Hadoop framework are Apache Pig and Apache Hive. Pig is a platform where large data sets can be analyzed using a data flow language, Pig Latin. Hive enables big data analysis using an SQL-like language called HiveQL. The purpose of this paper is to explore big data analytics using Hadoop. It focuses on Hadoop's core components and supporting analytical tools Pig and Hive.**

***Index Terms*— Big Data, Map Reduce, Hadoop, Apache Pig, Apache Hive HDFS**

I. INTRODUCTION

Apache Hadoop is an open-source software framework for storing data and running applications on clusters of commodity hardware. It provides massive storage for any kind of data, enormous processing power and the ability to handle virtually limitless concurrent tasks or jobs. The core of Apache Hadoop consists of a storage part, known as Hadoop Distributed File System (HDFS), and a processing part called MapReduce.

Hadoop splits files into large blocks and distributes them across nodes in a cluster. The term *Hadoop* has come to refer not just to the base modules above, but also to the *ecosystem*, or collection of additional software packages that can be installed on top of or alongside Hadoop, such as Apache Pig, Apache Hive, Apache HBase, Apache Phoenix, Apachespark, Apache ZooKeeper, Cloudera Impala, Apache Flume, Apache Sqoop, Apache Oozie, Apache Storm

The base Apache Hadoop framework is composed of the following modules:

- *Hadoop Common* - contains libraries and utilities needed by other Hadoop modules;
- *Hadoop Distributed File System (HDFS)*- a distributed file-system that stores data on commodity machines, providing very high aggregate bandwidth across the cluster;
- *Hadoop YARN* – a resource-management platform responsible for managing computing resources in clusters and using them for scheduling of users' applications; and
- *Hadoop MapReduce* – an implementation of the MapReduce programming model for large scale data processing.

APACHE HIVE is a datawarehouse infrastructure built on top of Hadoop for providing data summarization, query, and analysis. Hive gives an SQL-like interface to query data stored in various databases and file systems that integrate with Hadoop. The traditional SQL queries must be implemented in the MapReduce Java API to execute SQL applications and queries over a distributed data. Hive provides the necessary SQL abstraction to integrate SQL-like Queries (HiveQL) into the underlying Java API without the need to implement queries in the low-level Java API.

Apache Hive supports analysis of large datasets stored in Hadoop's HDFS and compatible file systems such as Amazon S3 filesystem. It provides an SQL-like language called HiveQL with schema on read and transparently converts queries to MapReduce, Apache Tez and Spark jobs. All three execution engines can run in Hadoop YARN. To accelerate queries, it provides indexes, including bitmap indexes.

Features of Hive:

- Indexing to provide acceleration, index type including compaction and Bitmap index as of 0.10, more index types are planned.
- Different storage types such as plain text, RCFile, HBase, ORC, and others.
- Metadata storage in an RDBMS, significantly reducing the time to perform semantic checks during query execution.
- Operating on compressed data stored into the

Hadoop ecosystem using algorithms including DEFLATE, BWT, snappy, etc.

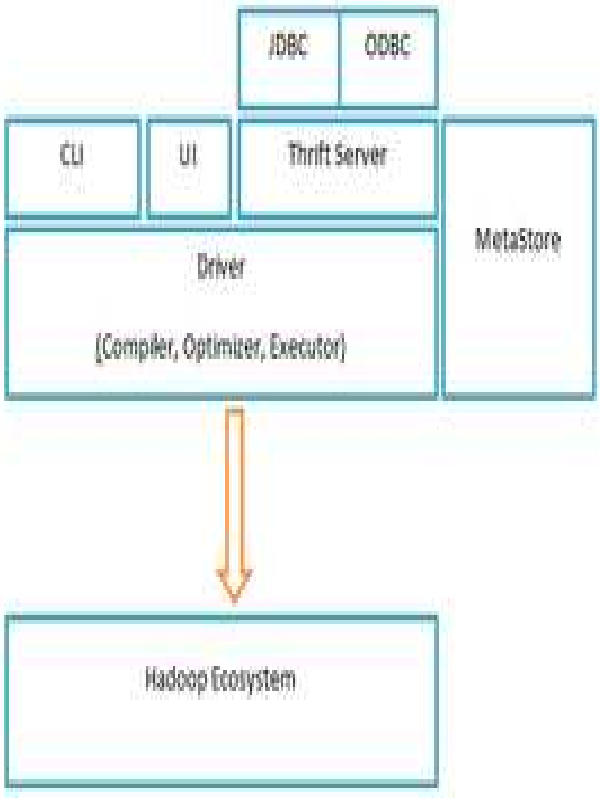
- Built-in user defined functions (UDFs) to manipulate dates, strings, and other data-mining tools. Hive supports extending the UDF set to handle use-cases not supported by built-in functions.
- SQL-like queries (HiveQL), which are implicitly converted into MapReduce or Tez, or Spark jobs.

By default, Hive stores metadata in an embedded Apache Derby database, and other client/server databases like MySQL can optionally be used.

Four file formats are supported in Hive, which are TEXTFILE, SEQUENCEFILE, ORC and .Apache Parquet can be read via plugin in versions later than 0.10 and natively starting at 0.13 Additional Hive plugins support querying of the Bitcoin Block

II. HIVE ARCHITECTURE:

A. Components of Hive Architecture



Major components of the Hive architecture are:

1. METASTORE:

Stores metadata for each of the tables such as their schema and location. It also includes the partition metadata which helps the driver to track the progress of various data sets distributed over the cluster. The data is stored in a traditional RDBMS format. The metadata helps the driver to keep a track of the data and it is highly crucial. Hence, a backup server regularly replicates the data which can be retrieved in case of data loss.**Driver:** Acts like a controller which receives the HiveQL statements. It starts the execution of statement by creating sessions and monitors the life cycle and progress of the execution. It stores the necessary metadata generated during the execution of an HiveQL statement. The driver also acts as a collection point of data or query result obtained after the Reduce operation.

2. COMPILER:

Performs compilation of the HiveQL query, which converts the query to an execution plan. This plan contains the tasks and steps needed to be performed by the Hadoop MapReduce to get the output as translated by the query. The compiler converts the query to an Abstract syntax tree (AST). After checking for compatibility and compile time errors, it converts the AST to a directed acyclic graph (DAG). DAG divides operators to MapReduce stages and tasks based on the input query and data.

3. OPTIMIZER:

Performs various transformations on the execution plan to get an optimized DAG. Various transformations can be aggregated together, such as converting a pipeline of joins by a single join, for better performance. It can also split the tasks, such as applying a transformation on data before a reduce operation, to provide better performance and scalability. However, the logic of transformation used for optimization used can be modified or pipelined using another optimizer.

4. EXECUTOR:

After compilation and Optimization, the Executor executes the tasks according to the DAG. It interacts with the job tracker of Hadoop to schedule tasks to be run. It takes care of pipelining the tasks by making sure that a task with dependency gets executed only if all other prerequisites are run.

5. CLI, UI, AND THRIFT SERVER:

Command Line Interface and UI (User Interface) allow an external user to interact with Hive by submitting queries, instructions and monitoring the process status. Thrift server allows external clients to interact with Hive just like how JDBC/OD

APACHE PIG is a high-level platform for creating programs that run on Apache Hadoop. The language for this platform is called **Pig Latin**. Pig can execute its Hadoop jobs in MapReduce, Apache Tez, or Apache Spark. Pig Latin abstracts the programming from the Java MapReduce idiom into a notation which makes MapReduce programming high level, similar to that of SQL for RDBMSs. Pig Latin can be extended using User Defined Functions (UDFs) which the user can write in Java, Python, JavaScript, Ruby or Groovy and then call directly from the language Apache Pig was originally developed at Yahoo Research around 2006 for researchers to have an ad-hoc way of creating and executing MapReduce jobs on very large data sets. A scripting platform for processing and analyzing large data sets With YARN as the architectural center of Apache™ Hadoop, multiple data access engines such as Apache Pig interact with data stored in the cluster. Apache Pig allows Apache Hadoop users to write complex

MapReduce transformations using a simple scripting language called Pig Latin. Pig translates the Pig Latin script into MapReduce so that it can be executed within **YARN** for access to a single dataset stored in the Hadoop Distributed File System **(HDFS).**

Pig was designed for performing a long series of data operations, making it ideal for three categories of Big Data jobs.

III. EXTRACT-TRANSFORM-LOAD (ETL) DATA PIPELINES, RESEARCH ON RAW DATA, ANDITERATIVE DATA PROCESS

Pig runs on Apache Hadoop YARN and makes use of MapReduce and the Hadoop Distributed File System (HDFS). The language for the platform is called Pig Latin, which abstracts from the Java MapReduce idiom into a form similar to SQL. While SQL is designed to query the data, Pig Latin allows you to write a data flow that describes how your data will be transformed (such as aggregate, join and sort). Since Pig Latin scripts can be graphs (instead of requiring a single output) it is possible to build complex data flows involving multiple inputs, transforms, and outputs. Users can extend Pig Latin by writing their own functions, using Java, Python, Ruby, or other scripting languages. Pig Latin is sometimes extended using UDFs (User Defined Functions), which the user can write in any of those languages and then call directly from the Pig Latin.

The user can run Pig in two modes, using either the "pig" command or the "java" command:

- ✓ **MapReduce Mode.** This is the default mode, which requires access to a Hadoop cluster.
- ✓ **Local Mode.** With access to a single machine, all files are installed and run using a local host and file system.

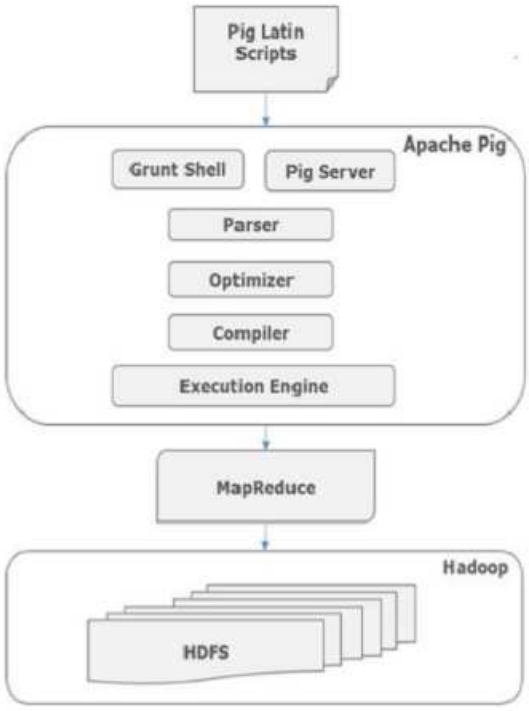
Features of Pig

Apache Pig comes with the following features :

- **Rich set of operators** -It provides many operators to perform operations like join, sort, filer, etc.
- **Ease of programming** - Pig Latin is similar to SQL and it is easy to write a Pig script if you are good at SQL.
- **Optimization opportunities** - The tasks in Apache Pig optimize their execution automatically, so the programmers need to focus only on semantics of the language.
- **Extensibility** - Using the existing operators, users can develop their own functions to read, process, and write data.
- **UDF's** - Pig provides the facility to create **User-defined Functions** in other programming languages such as Java and invoke or embed them in Pig Scripts.
- **Handles all kinds of data** - Apache Pig analyzes all kinds of data, both structured as well as unstructured. It stores the results in HDFS.

IV. APACHE PIG COMPONENTS

As shown in the figure, there are various components in the Apache Pig framework. Let us take a look at the major components.



1. PARSER

Initially the Pig Scripts are handled by the Parser. It checks the syntax of the script, does type checking, and other miscellaneous checks. The output of the parser will be a DAG (directed acyclic graph), which represents the Pig Latin statements and logical operators.

In the DAG, the logical operators of the script are represented as the nodes and the data flows are represented as edges.

2. OPTIMIZER

The logical plan (DAG) is passed to the logical optimizer, which carries out the logical optimizations such as projection and pushdown.

3. COMPILER

The compiler compiles the optimized logical plan into a series of MapReduce jobs.

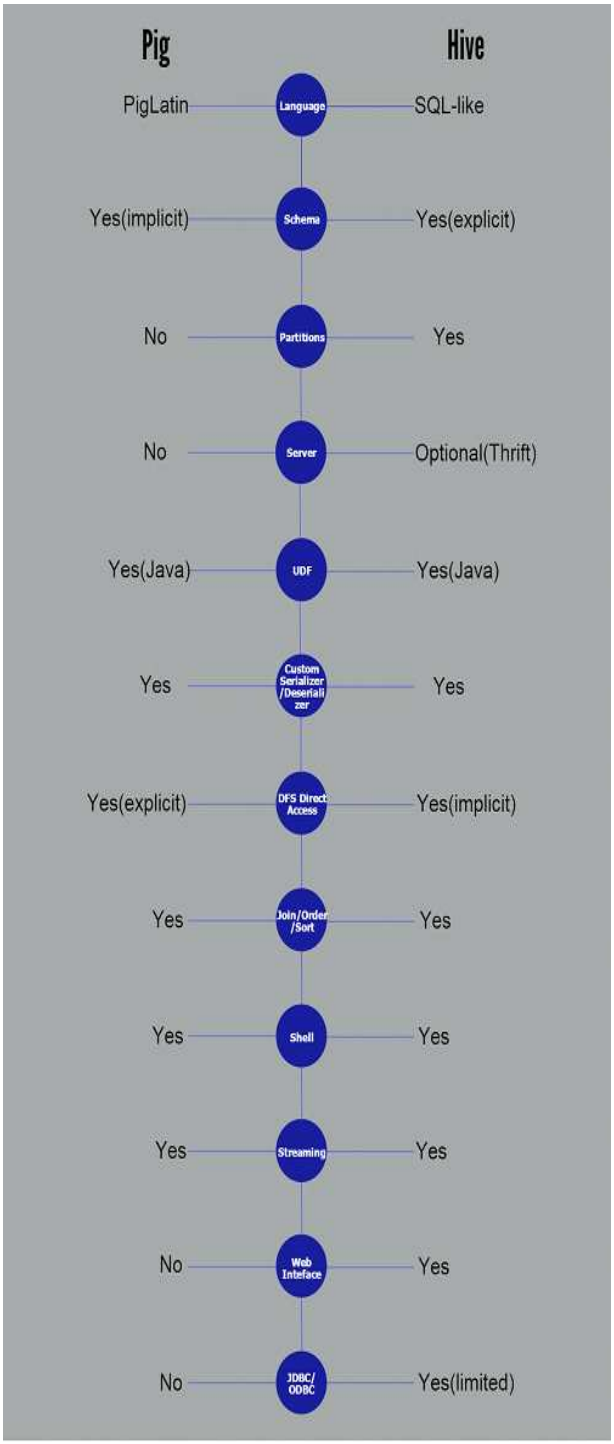
4. EXECUTION ENGINE

Finally the MapReduce jobs are submitted to Hadoop in a sorted order. Finally, these MapReduce jobs are executed on Hadoop producing the desired results

APACHE PIG VS APACHE HIVE

Both Apache Pig and Hive are used to create Map Reduce jobs. And in some cases, Hive operates on HDFS in a similar way Apache Pig does. In the following table, we have listed a few significant points that set Apache Pig apart from Hive.

APACHE PIG	APACHE HIVE
Apache Pig uses a language called Pig Latin. It was originally created at Yahoo	Hive uses a language called HiveQL. It was originally created at Facebook.
Pig Latin is a data flow language	HiveQL is a query processing language
Pig Latin is a procedural language and it fits in pipeline paradigm	HiveQL is a declarative language.
Apache Pig can handle structured, unstructured, and semi-structured data	Hive is mostly for structured data.



V. CONCLUSION:

To conclude with after having understood the difference between pig and Hive , to me both Hive Hadoop and Pig Hadoop Component will help you achieve the same goals, we can say that Pig is a script kiddy and hive comes in, innate for all the natural database developers. when it comes to access choices, Hive is said to have more features over Pig. Both the Hive and Pig components are reportedly having near about the same number of committers in every project and likely in the near future we are going to see great advancements in both on the development front.

Pig and Hive afford advanced level of abstraction whereas

Hadoop MapReduce provides low level of abstraction. Hadoop MapReduce requires additional lines of code when compared to the Pig and Hive. Hive requires very few lines of code when compared to the Pig and Hadoop MapReduce .The concluding result illustrate that the analysis performed by both of the map reduce machines is successful. The performance of both was nearly same.

REFERENCES

[1] 1. Ramesh R, Divya G, Divya D, Merin K Kurian , “Big Data Sentiment Analysis using Hadoop “, (IJIRST)International Journal for Innovative Research in Science & Technology,Volume 1 , Issue 11 , April 2015 ISSN : 2349-6010.

[2] 2. An Oracle White Paper, “Hadoop and NoSQL Technologies and the Oracle Database”, February 2011.

[3] 3. Russom, “Big Data Analytics”, TDWI Research, 2011.

[4] 4. S. Dhawan and S. Rathee, “Big data analytics using Hadoop components like Pig and Hive”, American International Journal of Research in Science, Technology, Engineering & Mathematics, vol. 2, (2013), pp. 88-93. [4] A. Gates, “Pig and Hive at yahoo”, YAHOO developer network”, (2010), https://developer.yahoo.com/blogs/hadoop/pig-hive-yahoo-464.html.

[5] 5. B. Jakobus and P. McBrien, “Pig versus Hive: Benchmarking high level query languages”, IBM developer Works, (2014), http://www.ibm.com/developerworks/library/ba-igvhive/ pighivebenchmarking.pdf

[6] 6. P. J. Jamack, “Hive as a tool for ETL or ELT”, IBM developerWorks, (2014), http://www.ibm.com/developerworks/library/bd-hivetool/.

[7] 7. M.T. Jones, “Process your data with Apache Pig”, IBM developerWorks, (2012), http://www.ibm.com/developerworks/library/l-apachepigdataquery.

[8] 8. Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao,Prasad Chakka, Suresh Anthony, Hao Liu, Pete Wyckoff and Raghotham Murthy.Hive - A Warehousing Solution Over a MapReduce Framework