Scalable Storage System for Big Data

By
Nikita Deshmukh
Under the guidance of
Prof. Madhuri Gedam
Department of Computer Engineering,
Shree L R Tiwari College of Engineering,
Kanakia Park, Mira Road(E), Thane - 401107

Abstract—The main goal of this paper is to provide how to deal with huge amount of data that are generated from various sources. Big data, which refers to the data sets that are too big to be handled using the existing database management tools, are emerging in many important applications, such as Internet search, business informatics, social networks, social media, genomics, and meteorology. Big Data is a convergence of new hardware and algorithms that allow us to discover new patterns in large data sets patterns we can apply to making better predictions and ultimately, better decisions. Today, lot of data generated from cloud computing, mobile and social site such as whatsapp, facebook, twitter etc. The scope of this of this paper is to describe the applications associated with Big Data and also some system like Hadoop and Data Warehouse that are going to manage, store, analyse the huge amount of data.

Index terms: Big data analytics, data storage, data analytics, Hadoop, Data Warehousing.

*****

1. INTRODUCTION

Big datais a term for data sets that are so large or complex that traditional data processing application softwares are insufficient to deal with them. Difficulties include capture, storage, data curation, analysis, search, sharing, transfer, updating, visualization, querying and information privacy. The word "big data" often refers simply to the use of predictive analytics, user behavior analytics, or certain other advanced data analytics methods that extract value from data, and seldom to a particular size of data set. "There is little hesitation that the quantities of data now accessible are indeed large, but that's not the most relevant feature of this new data ecosystem." The amount of data that is regularly created in various fields such as meteorology, genomics, E-science, Complex physics simulations, environmental search and biology is so large that they face problems in storing and processing that data. Keeping in mind the uniqueness of big-data, designing a scalable big-data system faces a series of technical limitations, including:

- Because of the variety of dissimilar data sources and the sheer volume, it is difficult to gather and amalgamate data with scalability from distributed locations. For example, more than 175 million tweets containing image, text, video, social relationship are generated by millions of accounts globally.
- Big data systems need to manage and store up the enormous data that is generated on a daily basis. For instance Facebook needs to access, store and examine data over 30 petabytes in size.
- Big data analytics must efficiently mine colossal datasets at different levels in real time or near real time - including modeling, visualization, optimization, and prediction - such that intrinsic promises can be revealed to improve decision making and get further advantages.

To store and analyze data this vast, Hadoop and Data Warehousing are employed.

2. BIG DATA DEFINITION

The term has been in use since the 1990s, with some giving credit to John Mashey for coining or at least making it popular. Big data usually includes data sets with sizes beyond the ability of commonly used software tools to capture, curate, manage, and process data within a tolerable elapsed time. Big data "size" is a constantly moving target, as of 2012 ranging from a few dozen terabytes to many petabytes of data. Big data requires a set of techniques and technologies with new forms of integration to reveal insights from datasets that are diverse, complex, and of a massive scale.

Table 1: Comparison between traditional data and Big Data

	Traditional Data	Big Data
Volume	GBs	Keeps growing (in TBs or PBs)
Structure	Structured	Semi-structured or unstructured
Data Store	RDBMS	NoSQL
Data Integration	Easy	Gruesome

2.1 Characteristics of Big Data are:

- Volume:

The quantity of generated and stored data. The size of the data determines the value and potential insight- and whether it can actually be considered big data or not.

- Variety:

The type and nature of the data. This helps people who analyze it to effectively use the resulting insight.

- Velocity:

In this context, the speed at which the data is generated and processed to meet the demands and challenges that lie in the path of growth and development.

- Variability:

Inconsistency of the data set can hamper processes to handle and manage it.

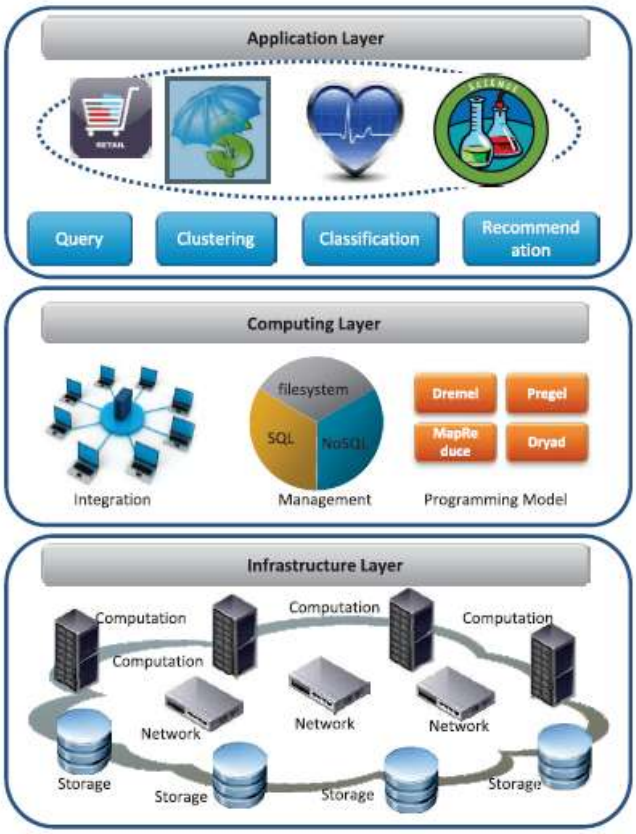
- Veracity:

The quality of captured data can vary greatly, affecting accurate analysis. [1]

2.2 Layered Architecture of Big Data

The big data architecture shown below has been decomposed into layers namely Application layer, Computing Layer and infrastructure layer.

Figure 1:Layered Architecture of Big Data



Application Layer:

This layer is upmost layer. It provides various applications i.e. functionalities provided by big data systems. The services include Querying, Clustering, Clustering etc.

Computing Layer:

This layer has various tools on top of infrastructure layer. The tools include data integration, data management and the programming model. MapReduce, Dryad, Pregel, and Dremel are parts of programming models.

Infrastructure Layrer:

This layer has various ICT resources which are organized by cloud computing resources. All of these resources are used in big data processing.

2.3A Brief History of Big Data

After going through definitions, we will get a brief on history of big data.

- **Megabyte to Gigabyte**: In the 1970s and 1980s, historical

business data introduced the earliest ``big data"
challenge in moving from megabyte to gigabyte sizes. [2]

- **Gigabyte to Terabyte**: In the late 1980s, the popularization

of digital technology caused data volumes to
expand to several gigabytes or even a terabyte. [2]

- **Terabyte to Petabyte**: During the late 1990s, when

the database community was admiring its “finished”
work on the parallel database, the rapid development
of Web 1.0 led the whole world into the Internet era,
along with massive semi-structured or unstructured webpages
holding terabytes or petabytes (PBs) of data. [2]

- **Petabyte to Exabyte:** Under current development trends,

data stored and analyzed by big companies will undoubtedly
reach the PB to exabyte magnitude soon.[2]

2.4Big Data Paradigms

Big Data analytics is the process of using powerful algorithms on huge amount of data to uncover previously unknown facts and to find hidden data patterns. According to the time requirement there are two methods used. The two Big Data paradigms namely Streaming and Batch Processing.

- **Streaming:**

Streaming is based on phenomenon that data is useful when it is most recent so it processes the data as soon as data is available. In this paradigm data arrives in stream. This paradigm is used for online applications where even micro seconds make a difference.

- **Batch Processing:**

In this method, the data is first stored and then processed. MapReduce has been the most used batch processing method.The main principal is that data is first divided into parts and these parts are processed concurrently to generate results.

2.5 Applications of Big Data

Big data has increased the demand of information management specialists so much so that Software AG, Oracle Corporation, IBM, Microsoft, SAP, EMC, HP and Dell have spent more than $15 billion on software firms specializing in data management and analytics. In 2010, this industry was worth more than $100 billion and was growing at almost 10 percent a year: about twice as fast as the software business as a whole.[1]

Big Data has applications in following various areas:

1. Banking and Securities

2. Communications, Media and Entertainment
3. Healthcare Providers
4. Education
5. Manufacturing and Natural Resources
6. Government
7. Insurance
8. Retail and Wholesale Trade
9. Transportation
10. Energy and Utilities

3. HADOOP

Apache Hadoop is an open-source software framework used for distributed storage and processing of big data sets using the MapReduce programming model Computer clusters built from commodity hardware is what Hadoop built on. The Hadoop modules are built on the fundamental that any hardware failure can occur and these failures should be automatically handled by the framework itself.

3.1 Components of Hadoop

The Hadoop project is made up of three pieces: HadoopDistributed File System(HDFS), the Hadoop MapReducemodel and Hadoop Common. To totally understandHadoop, you must get a grasp of the fundamental infrastructure of the file systemand the MapReduce programming model. Firstly we will see HDFS which allows the applications to be run.

3.1.1 The Hadoop Distributed File System

The Hadoop Distributed File System (HDFS) is a distributed file system designed to run on commodity hardware. It has various resemblances with existing distributed file systems. Although, the differences from other distributed file systems are noteworthy. HDFS is designed to be deployed on low-cost hardware and is highly fault-tolerant. HDFS is suitable for applications that have huge data sets and offers high levels of throughput access to application data. A few POSIX requirements are relaxed by HDFS to enable streaming access to file system data. For a project called Apache Nutch web search engine is what HDFS was initially built for. It is now an Apache Hadoop subproject.

3.1.2 Basics of MapReduce

MapReduce is the heart of Hadoop. It is this programming paradigm that allows for massive scalability across hundreds or thousands of servers in a Hadoop cluster**.**

The term MapReduceactually refers to two separate and distinct tasks thatHadoop programs perform. The first is the map job, which takes a set of data and converts it into another set of data, where individual elements are broken down into *tuples* (key/value pairs). The reduce job takes the output from a map as input and combines those data tuples into a smaller set of tuples. As the sequence of the name MapReduce implies, the reduce job is always performed after the map job.[4]

An application submits a job to a specific node in a Hadoop cluster, which is running a daemon called the JobTracker. The JobTracker communicates with the NameNode to find out where all of the data required for this job exists across the cluster, and then breaks the job down into map and reduce tasks for each node to work on in the cluster. These tasks are scheduled on the nodes in the cluster where the data exists. Note that a node might be given a

task for which the data needed by that task is not local to that node. In such a case, the node would have to ask for the data to be sent across the network interconnect to perform its task. Of course, this isn't very efficient, so the JobTracker tries to avoid this and attempts to schedule tasks where the data is stored. This is the concept of data locality we introduced earlier, and it is critical when working with large volumes of data. In a Hadoop cluster, a set of continually running daemons, referred to as TaskTracker agents, monitor the status of each task. If a task fails to complete, the status of that failure is reported back to the JobTracker, which will then reschedule that task on another node in the cluster.[4]

Figure 2: The flow of data in a simple MapReduce job

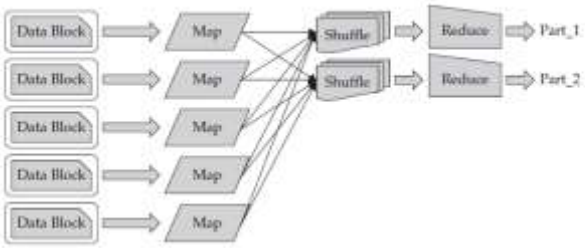


Figure 2 shows an example of a MapReduce flow. You can see that multiple reduce tasks can serve to increase the parallelism and improve the overall performance of the job. In the case of Figure 4-2, the output of the map tasks must be directed (by key value) to the appropriate reduce task. If we apply our maximum temperature example to this figure, all of the records that have a key value of Toronto must be sent to the same reduce task to produce an accurate result. This directing of records to reduce tasks is known as a Shuffle, which takes input from the map tasks and directs the output to a specific reduce task. Hadoop gives you the option to perform local aggregation on the output of each map task before sending the results off to a reduce task through a local aggregation called a Combiner. [4]

All MapReduce programs that run natively under Hadoop are written in Java, and it is the Java Archive file (jar) that's distributed by the JobTracker to the various Hadoop cluster nodes to execute the map and reduce tasks.[4]

3.1.3 Other Hadoop Components

Many other open source projects fall under the Hadoop umbrella, either as Hadoop subprojects or as top-level Apache projects, with more coming up as time goes on. Some of them are namely ZooKeeper, HBase, Oozie, Lucene, and more.

ZooKeeper

ZooKeeperis an open source Apache project that provides services that enable synchronization across a cluster and a centralized infrastructure. Common objects needed in large cluster environments are maintained by ZooKeeper. Configuration information, hierarchical naming space etc are some of the examples of these objects. These services are

leveraged by applications to coordinate distributed processing across large clusters.

An infrastructure for cross-node synchronization is provided by ZooKeeper. Applications use this infrastructure to make sure that tasks throughout the cluster are synchronized or serialized. ZooKeeper servers maintain this status type information in memory. A ZooKeeper server is a machine that perseveres this information in local log files and maintains a copy of the state of the whole system. Multiple ZooKeeper servers can support a very large Hadoop cluster. To update and retrieve the synchronization information, each client machine communicates with one of the ZooKeeper servers among several ones.

HBase

HBase is a column-oriented database management system that runs on top of HDFS. Hbase unlike relational database systems,does not support query language like Strucuted Quesry Language (SQL). In fact Hbase is not a relational data store at all. Java is the language in which Hbase applications are written in. Typical MapReduce applications are also written in Java. HBase does support writing applications in Avro, REST, and Thrift languages. This paper does however does not cover these aspects of Hbase.

An HBase system consists of set of tables. Each table contains rows and columns, much like a traditional database.Hbase allows column familieswhere many attributes can be grouped together so that elements of a column family are stored together. In traditional row-oriented relational databases all the columns of a given row are stored together. This is in contrast to Hbase. In Hbase it is mandatory that you have predefined table schema and specified the columns family/families. Hbase is built on similar concept of HDFS having NameNode and slave nodes, MapReduce having Job-Tracker and Tasktracker slaves. There is a *master node* that manages the cluster and *region servers* perform the work on data. This master node and region servers are primary components of Hbase.

Oozie

Oozieis an open source project that simplifies workflow and coordination between jobs. Actions can be defined and then the dependency between those actions is provided by Ooozie. After this, Oozie will start to execute the actions once the required dependencies have been met. Oozie is based on Directed Acyclical Graph (DAG), also what a workflow in Oozie is called. The meaning of Acyclical is that there is a starting and an ending point to the graph. In other words there are no loops. All the tasks and dependencies are executed from start to end with no going back meaning a don't-look-back approach. A DAG is made up of action nodes and dependency nodes. MapReduce jobs, Pig applications, Java applications, or even a file system task can be an action node. The flow control in graph is provided by node elements based on the input of the previous or preceding task. Examples of flow control nodes are decisions, forks, and join nodes.

4. DATA WAREHOUSING

Data warehousing is a phenomenon that grew from the huge amount of electronic data stored in recent years and from the urgent need to use that data to accomplish goals that go beyond the routine tasks linked to daily processing. In a typical scenario, a large corporation has many branches, and senior managers need to quantify and evaluate how each branch contributes to the global business performance. The corporate database stores detailed data on the tasks performed by branches. Tailor-made queries can be fired to get the required data, to meet the managers' requirements. For this process to work, a proper query must be formulated after studying the database like an aggregate SQL query. Then the query is processed. The processing of the query might take many hours depending on the complexity and the workload that takes to process the request. After that, a final report is generated and handed over to the managers in the required form, mostly a spreadsheet.

Many years ago, database designers realized that such an approach is hardly feasible, because it is very demanding in terms of time and resources, and it does not always achieve the desired results. Moreover, a mix of analytical queries with transactional routine queries inevitably slows down the system, and this does not meet the needs of users of either type of query. Today's advanced data warehousing processes separate online analytical processing (OLAP) from online transactional processing (OLTP) by creating a new information repository that integrates basic data from various sources, properly arranges data formats, and then makes data available for analysis and evaluation aimed at planning and decision-making processes (Lechtenbörger, 2001).

4.1 What is Data Warehousing

Data warehousing is a collection of methods, techniques, and tools used to support knowledge workers—senior managers, directors, managers, and analysts—to conduct data analyses that help with performing decision-making processes and improving information resources.

The definition of data warehousing presented here is intentionally generic; it gives an idea of the process but does not include specific features of the process. To understand the role and the useful properties of data warehousing completely, you must first understand the needs that brought it into being. In 1996, R. Kimball efficiently summed up a few claims frequently submitted by end users of classic information systems:[6]

• "We have heaps of data, but we cannot access it!" This shows the frustration of those who are responsible for the future of their enterprises but have no technical tools to help them extract the required information in a proper format. [6]

• "How can people playing the same role achieve substantially different results?" In midsize to large enterprises, many databases are usually available, each devoted to a specific business area. They are often stored on different logical and physical media that are not conceptually integrated. For this reason, the results achieved in every business area are likely to be inconsistent. [6]

• "We want to select, group, and manipulate data in every possible way!" Decision-making processes cannot always be planned before the decisions are made. End users need a tool that is user-friendly and flexible enough to conduct ad hoc

analyses. They want to choose which new correlations they need to search for in real time as they analyze the information retrieved. [6]

• "Show me just what matters!" Examining data at the maximum level of detail is not only useless for decision-making processes, but is also self-defeating, because it does not allow users to focus their attention on meaningful information. [6]

• "Everyone knows that some data is wrong!" This is another sore point. An appreciable percentage of transactional data is not correct—or it is unavailable. It is clear that you cannot achieve good results if you base your analyses on incorrect or incomplete data. [6]

4.2Data Warehouse

A data warehouse is a collection of data that supports decision-making processes. It provides the following features :

• It is subject-oriented.
• It is integrated and consistent.
• It shows its evolution over time and it is not volatile.

Data warehouses are subject-oriented because they operate on enterprise-specific concepts, such as customers, products, sales, and orders. On the contrary, operational databases hinge on many different enterprise-specific applications.

A data warehouse should proffer a unified view of all the data. We can say that data warehousing systems don't need the new data but rearrange the data that is coming continuously . This indirectly says that an information system should be previously available.

We know that operation data covers a short period of time the reason being that most of the data is latest. A data warehouse in contrast covers the data that spans a few years. This is the reason that data warehouses are regularly updated and operational data always keeps growing.

4.3 Data Marts

A data mart is a subset or an aggregation of the data stored to a primary data warehouse. It includes a set of information pieces relevant to a specific business area, corporate department, or category of users.

The data marts populated from a primary data warehouse are often called dependent. Although data marts are not strictly necessary, they are very useful for data warehouse systems in midsize to large enterprises because

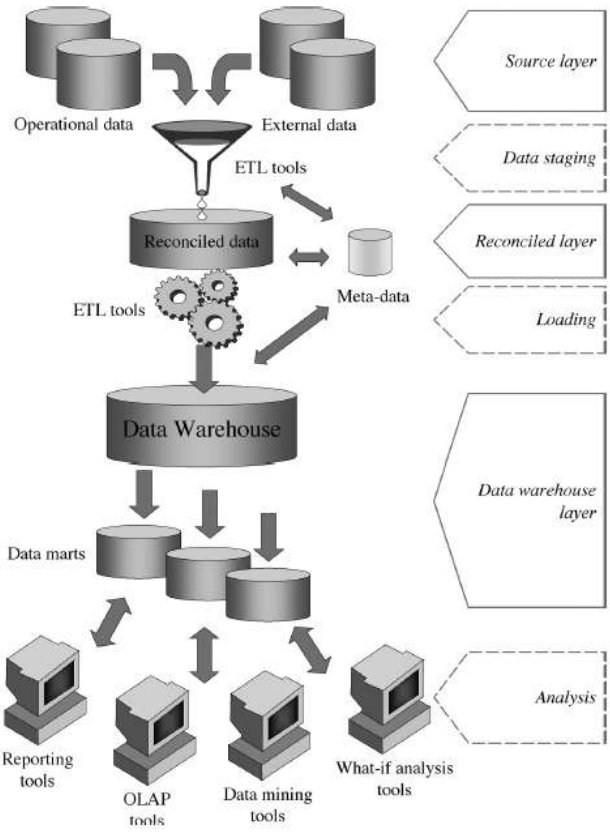
• they are used as building blocks while incrementally developing data warehouses;
• they mark out the information required by a specific group of users to solve queries;
• they can deliver better performance because they are smaller than primary data warehouses.

4.4Data Warehouse Architecture (Three-Layer Architecture)

In this architecture, the third layer is the reconciled data layer or operational data store. This layer materializes operational data obtained after integrating and cleansing source data. As a result, those data are integrated, consistent, correct, current, and detailed. Figure 3 shows a data warehouse that is not populated from its sources directly, but from reconciled data.[7]

The main advantage of the reconciled data layer is that it creates a common reference data model for a whole enterprise. At the same time, it sharply separates the problems of source data extraction and integration from those of data warehouse population. Remarkably, in some cases, the reconciled layer is also directly used to better accomplish some operational tasks, such as producing daily reports that cannot be satisfactorily prepared using the corporate applications, or generating data flows to feed external processes periodically so as to benefit from cleaning and integration. However, reconciled data leads to more redundancy of operational source data. Note that we may assume that even two-layer architectures can have a reconciled layer that is not specifically materialized, but only virtual, because it is defined as a consistent integrated view of operational source data. [7]

Figure 3:Three-layer architecture for a data warehouse system



4.4.1 Data Staging and ETL

Data Stage layer consists of ETL processes that Extract Transform and load the data from various heterogeneous data sources. The data sources can be operational or external data sources. This is layer is of paramount importance for the reason that it provides the data that will be processed further in the entire data warehousing process. The reconciled data is detailed, comprehensive and top notch quality data that goes on ahead.

4.4.2 Extraction

This part consists of extraction of data from various data sources. This represents the most important aspect of ETL since extracting correct data sets the stage for further successive processes. Data warehousing projects combine data from an array of different data sources. So basically, the extraction stage aims to convert the data into a single format appropriate for transformation processing.

4.4.3Cleansing

The cleansing phase is crucial in a data warehouse system because it is supposed to improve data quality—normally quite poor in sources. The following list includes the most frequent mistakes and inconsistencies that make data "dirty":

- Duplicate data
- Inconsistent values that are logically associated
- Missing data
- Unexpected use of fields
- Impossible or wrong values

4.4.4Transformation

Transformation is the core of the reconciliation phase. It converts data from its operational source format into a specific data warehouse format. If you implement a three-layer architecture, this phase outputs your reconciled data layer.

4.4.5 Loading

Loading into a data warehouse is the last step to take. Loading can be carried out in two ways:

• **Refresh:**

Data warehouse data is completely rewritten. This means that older data is replaced. Refresh is normally used in combination with static extraction to initially populate a data warehouse.

• **Update:**

Only those changes applied to source data are added to the data warehouse. Update is typically carried out without deleting or modifying preexisting data. This technique is used in combination with incremental extraction to update data warehouses regularly.

5. CONCLUSION

The data that is being generated on a daily basis is simply put huge. On top of that organizations that depend on this data need advanced tools to operate on that data and derive insights from it. Social networks such as Facebook, twitter etc create data in terabytes within a very short period frequently. Also countless other fields such as space research, digital media, medical industry heavily rely on the big data analytics for further improving the quality of their operations. With data being generated at speeds never seen before, the limits have been pushed to herculean heights for the data storage systems to meet the requirements. Until now, all the existing methods and technologies have been able to suffice if not satisfy the needs of data storages of various organizations. With more and more online businesses thriving, researches generating data, the storage and manipulation of data will require the existing applications and tools to be scaled to larger sizes day by day.

REFERENCES

[1] https://en.m.wikipedia.org/wiki/Big_data
[2] Toward Scalable Systems for Big Data Analytics: A Technology Tutorial, Han Hu, Yonggang Wen, Tat-Seng Chua, And Xuelong Li
[3] "Big Data Definition". MIKE2.0. Retrieved 9 March 2013.
[4] Understanding Big Data, Chris Eaton, Dirk Deroos, Tom Deautch, George Lapis 2011
[5] Jens Lechtenbörger (Universitat Monster): Data Warehouse Schema Design. 2001. ISBN 3-89838-479-9
[6] The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling, The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling Margy Ross Ralph Kimball, 1996
[7] Data Warehouse Design: Modern Principles and Methodologies Matteo Golfarelli, Stefano Rizzi, 2009
[8] "Welcome to Apache Hadoop!". hadoop.apache.org. Retrieved 2016-08-25.
[9] Hilbert, Martin. "Big Data for Development: A Review of Promises and Challenges. Development Policy Review.". martinhilbert.net. Retrieved 2015-10-07.
[10] Data Mart Concepts". Oracle. 2007.