**NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY**

**(An autonomous institution with A Grade by NAAC /UGC, Affiliated to Visvesvaraya Technological University, Belgaum, Approved by UGC/AICTE/Govt. of Karnataka)**

**Yelahanka, Bengaluru-560064**

**ACADEMIC YEAR 2020-2021**

BIGDATA LABORATORY

REPORT ON

**HADOOP**

Submitted By

**SHRESHT V S (1NT17IS158)**

**SUMUKHA VASISHT (1NT17IS168)**

Submitted To

**Mr. PRASHANTH B S**

**DEPARTMENT OF INFROMATION SCIENCE AND ENGINEERING**

**CONTENTS**

1. Hadoop

|  |
| --- |
| * 1. Introduction |
| * 1. Hadoop architecture |
| * 1. Components of Hadoop |
| * 1. Hadoop |
| * 1. Features of Hadoop |
| * 1. Advantages and Disadvantages |

1. Map Reduce

|  |
| --- |
| * 1. Introduction |
| * 1. Architecture of map reduce |
| * 1. Application of map reduce. |

1. Problem statement
2. Solution
3. Code
4. Output

**HADOOP**

**INTRODUCTION:**

Hadoop is a framework of the open source set of tools distributed under Apache License. It is used to manage data, store data, and process data for various big data applications running under clustered systems. In the previous years, Big Data was defined by the “3Vs” but now there are “5Vs” of Big Data which are also termed as the characteristics of Big Data.

1. **Volume:** With increasing dependence on technology, data is producing at a large volume. Common examples are data being produced by various social networking sites, sensors, scanners, airlines and other organizations.
2. **Velocity:** Huge amount of data is generated per second. It is estimated that by the end of 2020, every individual will produce 3mb data per second. This large volume of data is being generated with a great velocity.
3. **Variety:** The data being produced by different means is of three types:
   * StructuredData**:** It is the relational data which is stored in the form of rows and colums.
   * UnstructuredData**:** Texts, pictures, videos etc. are the examples of unstructured data which can’t be stored in the form of rows and columns.
   * Semi Structured Data**:** Log files are the examples of this type of data.
4. **Veracity:**The term Veracity is coined for the inconsistent or incomplete data which results in the generation of doubtful or uncertain Information. Often data inconsistency arises because of the volume or amount of data e.g. data in bulk could create confusion whereas less amount of data could convey half or incomplete Information.
5. **Value:**After having the 4 V’s into account there comes one more V which stands for Value!. Bulk of Data having no Value is of no good to the company, unless you turn it into something useful. Data in itself is of no use or importance but it needs to be converted into something valuable to extract Information. Hence, you can state that Value! is the most important V of all the 5V’s

**Hadoop Architecture:**

Hadoop follows a master slave architecture design for data storage and distributed data processing using HDFS and MapReduce respectively. The master node for data storage is hadoop HDFS is the NameNode and the master node for parallel processing of data using Hadoop MapReduce is the Job Tracker. The slave nodes in the hadoop architecture are the other machines in the Hadoop cluster which store data and perform complex computations. Every slave node has a Task Tracker daemon and a DataNode that synchronizes the processes with the Job Tracker and NameNode respectively. In Hadoop architectural implementation the master or slave systems can be setup in the cloud or on-A screenshot of a cell phone

Description automatically generatedpremise.

**Components of Hadoop:**

Hadoop has three components:

1. **HDFS:** Hadoop Distributed File System is a dedicated file system to store big data with a cluster of commodity hardware or cheaper hardware with streaming access pattern. It enables data to be stored at multiple nodes in the cluster which ensures data security and fault tolerance.
2. **Map Reduce :** Data once stored in the HDFS also needs to be processed upon. Now suppose a query is sent to process a data set in the HDFS. Now, Hadoop identifies where this data is stored, this is called Mapping. Now the query is broken into multiple parts and the results of all these multiple parts are combined and the overall result is sent back to the user. This is called reduce process. Thus while HDFS is used to store the data, Map Reduce is used to process the data.
3. **YARN :** YARN stands for**Yet Another Resource Negotiator**. It is a dedicated operating system for Hadoop which manages the resources of the cluster and also functions as a framework for job scheduling in Hadoop. The various types of scheduling are First Come First Serve, Fair Share Scheduler and Capacity Scheduler etc. The First Come First Serve scheduling is set by default in YARN.

**Features of Hadoop:**

* + **Easy to Use:** The projects or set of tools provided by Apache Hadoop are easy to work upon in order to analyse complex data sets.
  + **Economically Feasible:** It is cheaper to store data and process it than it was in the traditional approach. Since the actual machines used to store data are only commodity hardware.
  + **Open Source:** Since Hadoop is distributed as an open source software under Apache License, so one does not need to pay for it, just download it and use it.
  + **Fault Tolerance:** Since Hadoop stores three copies of data, so even if one copy is lost because of any commodity hardware failure, the data is safe. Moreover, as Hadoop version 3 has multiple name nodes, so even the single point of failure of Hadoop has also been removed.
  + **Scalability:** Hadoop is highly scalable in nature. If one needs to scale up or scale down the cluster, one only needs to change the number of commodity hardware in the cluster.
  + **Distributed Processing:** HDFS and Map Reduce ensures distributed storage and processing of the data.
  + **Locality of Data:** This is one of the most alluring and promising features of Hadoop. In Hadoop, to process a query over a data set, instead of bringing the data to the local computer we send the query to the server and fetch the final result from there. This is called data locality.

**Advantages:**

* Ability to store a large amount of data.
* High flexibility.
* Cost effective.
* High computational power.
* Tasks are independent.
* Linear scaling.

**Disadvantages:**

* Not very effective for small data.
* Hard cluster management.
* Has stability issues.
* Security concerns.

**MAP REDUCE**

**INTRODUCTION:**

**MapReduce** is a parallel and distributed solution approach. MapReduce has two key components. Map and Reduce. A map is a function which is used on a set of input values and calculates a set of key/value pairs. Reduce is a function which takes these results and applies another function to the result of the map function. Or with other words: Map transforms a set of data into key value pairs and Reduce aggregates this data into a scalar. A reducer receives all the data for a individual "key" from all the mappers.

MapReduce incorporates usually also a framework which supports MapReduce operations. A master controls the whole MapReduce process. The MapReduce framework is responsible for load balancing, re-issuing task if a worker as failed or is to slow, etc. The master divides the input data into separate units, send individual chunks of data to the mapper machines and collects the information once a mapper is finished. If the mapper are finished then the reducer machines will be assigned work. All key/value pairs with the same key will be send to the same reducer.

**Architecture of Map Reduce:**

A picture containing clock

Description automatically generated

The entire Map reduce process is a massively parallel processing setup where the computation is moved to the place of the data instead of moving the data to the place of the computation. This kind of approach helps to speed the process, reduce network congestion and improves the efficiency of the overall process.

The entire computation process is broken down into the mapping, shuffling and reducing stages.

**Mapping Stage:** This is the first step of the MapReduce and it includes the process of reading the information from the Hadoop Distributed File System (HDFS). The data could be in the form of a directory or a file. The input data file is fed into the mapper function one line at a time. The mapper then processes the data and reduces it into smaller blocks of data.

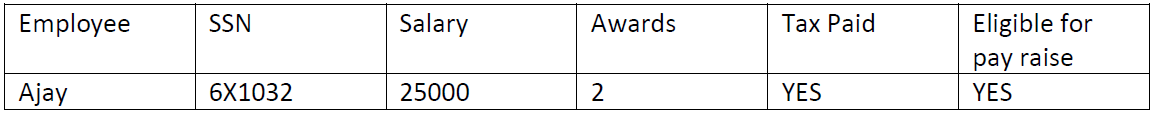
**Reducing Stage:** The reducer phase can consist of multiple processes. In the shuffling process, the data is transferred from the mapper to thereducer**.** Without the successful shuffling of the data, there would be no input to the reducer phase. But the shuffling process can start even before the mapping process has completed. Next, the data is sorting in order to lower the time taken to reduce the data. The sorting actually helps the reducing process by providing a cue when the next key in the sorted input data is distinct from the previous key. The reduce task needs a specific key-value pair in order to call the reduce function that takes the key-value as its input.  The output from the reducer can be directly deployed to be stored in the HDFS.

**Applications of map reduce:**

* Logfile analysis: Big logfiles are split and a mapper search for different webpages which are accessed. Every time a webpage is found in the log a key / value pair is emitted to the reducer where the key is the webpage and the value is "1". The reducers aggregate the number of for certain webpages. As a end result you have the total number of hits for each webpage.
* Full text indexing: The mapper would map every phrase / word in one document to the document and the reducer would write these mappings to an index.
* Distributed Grep
* Reverse Web-Link Graph: Map function outputs (URL target, source) from an input webpage (source). The reduce function concatenates the list of all source URLs associated with a give target URL and returns (target, list(sources))
* Word count in a number of documents.

**Problem statement 2:**

Create a dataset in excel as .csv file and it should contain the following fields with at least 20 sample datasets in it.



Use the Hadoop MapReduce programming framework to come up with a Program which will take the data from this .csv file and computes the following

1. Total number of employees who are eligible for the pay raise

2. Total number of cumulative awards the company had this year

CODE:

**package** my.mapred.pack;

**import** java.io.IOException;

**import** java.util.\*;

**import** org.apache.hadoop.fs.Path;

**import** org.apache.hadoop.io.\*;

**import** org.apache.hadoop.mapred.\*;

**public** **class** Count1118 {

//MAPPER CODE

**public** **static** **class** Map **extends** MapReduceBase **implements** Mapper<LongWritable, Text, Text, IntWritable> {

**private** **final** **static** IntWritable ***one*** = **new** IntWritable(1);

**public** **void** map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output, Reporter reporter) **throws** IOException {

String myString = value.toString();

String[] count = myString.split(",");

IntWritable awards = **new** IntWritable(Integer.*parseInt*(count[3]));

**if** (Integer.*parseInt*(count[3]) > 0) {

output.collect(**new** Text("Total Awards"), ***one***);

output.collect(**new** Text(count[0]), awards);

}

**if** (count[5].equals("YES")) {

output.collect(**new** Text(" Total Eligible for pay rise"), ***one***);

}

}

}

// REDUCER

**public** **static** **class** Reduce **extends** MapReduceBase **implements** Reducer<Text, IntWritable, Text, IntWritable> {

**public** **void** reduce(Text key, Iterator<IntWritable> values, OutputCollector<Text, IntWritable> output, Reporter reporter) **throws** IOException {

**int** finalCount = 0 ;

Text mykey = key ;

**while**(values.hasNext()) {

IntWritable value = values.next();

finalCount += value.get();

}

output.collect(mykey, **new** IntWritable(finalCount));

}

}

// DRIVER CONFIG

**public** **static** **void** main(String[] args) **throws** Exception {

JobConf conf = **new** JobConf(Count1118.**class**);

conf.setJobName("employee");

conf.setOutputKeyClass(Text.**class**);

conf.setOutputValueClass(IntWritable.**class**);

conf.setMapperClass(Map.**class**);

conf.setCombinerClass(Reduce.**class**);

conf.setReducerClass(Reduce.**class**);

conf.setInputFormat(TextInputFormat.**class**);

conf.setOutputFormat(TextOutputFormat.**class**);

FileInputFormat.*setInputPaths*(conf, **new** Path(args[0]));

FileOutputFormat.*setOutputPath*(conf, **new** Path(args[1]));

JobClient.*runJob*(conf);

} }

**OUTPUT:**

