Assignment No. 2

erformance (03)	Understanding (01)	Regularity (01)	total (05)	ated sign of subject teacher
02	01	01	04	

Date of Performance : Date of completion.

Aim: Design distributed applicationusing MapReduce which process a log file of a system. list of the users who have logged for maximum period of the system. use simple log file from the Internet and process it using a Pseudo distribution mode on hadoop platform

Objective:
To learn Map Reduce concept in Hadoop
To learn implementation of distributed in Hadoop.

Theory:

og file contain list of actions that have been occurred whenever someone access to your website or web application. These log files reside in web servers. each individual request is listed on a separate line a log file, called a log entry. these log files fits very well with the MapReduce programming model making. it a great example to understand the Hadoop Map/Reduce programming style. our implementation consists of three main parts:

1. Mapper
2. Reducer
3. Driver.

Procedure:

Step1. Write a Mapper

A Mapper overrides the -map function from the class "org.apache.hadoop.mapreduce.Mapper" which provides <key, value> pairs as the input. A Mapper implementation may output <key, value> pairs using the provided context.

nput value of the log file map task will be a ine of text from the input data file and the key would be the line number <line-number, line-of-text>, Map task outputs <Word, one> for each word in the line of text.

Psuedo-code.

```
void Map (key, value) {
  for each log entry x in value;
  output.collect (x, 1);
}
```

Step-2 Write a Reducer

A Reducer collects the intermediate <key, value> output from multiple map tasks and assemble a single result. Here, the log file program will sum up the ocurrences of each word to pairs as <logentry, occurrences>.

Pseudo Code:

```
void Reduce (key, <list of value>) {
 for each x in <list of value>:
  sum+ =x;
final-output. Collect (key, sum);
}
```

Step-3 Write Driver.

The Driver program configure and run the MapReduce job. We use the main program to perform basic configurations such as:

Job Name: name of this job

Executable (Jar) class: the main executable class. for here, Mainfest.

Mapperclass class which overrides the "map" function. for here, Map

Reducer: class which override the "reduce" function

for here, Reduce.
output key: type of output key. for here, text.
output value: type of output value. for here, IntWritable
file Input path
file output path.

Conclusion:
MapReduce is a programming framework that allows us to perform distributed and parallel processing on large data sets in a distributed environment. MapReduce consists of two distinct tasks - Map & Reduce. As the name MapReduce suggests, reducer phase takes place after mapper phase has been completed.