Practical 4

Name: Saurin Anilkumar Prajapati

Roll No.: 19BCE239

Course Code and Name: 2CS702 Big Data Analytics

Batch: D1

AIM:

Setup single node Hadoop cluster and apply HDFS commands on single node Hadoop Cluster.

Design MapReduce algorithms to take a very large file of integers and produce as output:
a) The largest integer
b) The sum of all the integers.

---

a) The Largest Integer

Algorithm:

```
import java . io . IOException ;
import java . util . StringTokenizer ;
import org . apache . hadoop . conf . Configuration ;
import org . apache . hadoop . fs . Path ;
import org . apache . hadoop . io . IntWritable ;
import org . apache . hadoop . io . Text ;
import org . apache . hadoop . mapreduce . Job ;
import org . apache . hadoop . mapreduce . Mapper ;
import org . apache . hadoop . mapreduce . Reducer ;
import org . apache . hadoop . mapreduce . lib . input . FileInputFormat ;
import org . apache . hadoop . mapreduce . lib . output . FileOutputFormat ;
public class max {
public static class TokenizerMapper
extends Mapper < Object , Text , Text , IntWritable >{
// private final static IntWritable one = new IntWritable (1) ;
private Text word = new Text (" max ") ;
public void map ( Object key , Text value , Context context
) throws IOException , InterruptedException {
StringTokenizer itr = new StringTokenizer ( value . toString () ) ;
```

```
int max = Integer . MIN_VALUE ;
int num ;
while ( itr . hasMoreTokens () ) {
num = Integer . parseInt ( itr . nextToken () ) ;
if( num > max ) {
max = num ;
}
}
context . write ( word , new IntWritable ( max ) ) ;
}
}
public static class IntSumReducer
extends Reducer < Text , IntWritable , Text , IntWritable > {
private IntWritable result = new IntWritable () ;

3

public void reduce ( Text key , Iterable < IntWritable > values ,

Context context
) throws IOException , InterruptedException {

int max = Integer . MIN_VALUE ;
int num ;
for ( IntWritable val : values ) {
num = val . get () ;
if( num > max ) {
max = num ;
}
}
result . set ( max ) ;
context . write ( key , result ) ;
}
}
public static void main ( String [] args ) throws Exception {
Configuration conf = new Configuration () ;
Job job = Job . getInstance ( conf , " find max ") ;
job . setJarByClass ( max . class ) ;
job . setMapperClass ( TokenizerMapper . class ) ;
job . setCombinerClass ( IntSumReducer . class ) ;
job . setReducerClass ( IntSumReducer . class ) ;
job . setOutputKeyClass ( Text . class ) ;
job . setOutputValueClass ( IntWritable . class ) ;
FileInputFormat . addInputPath ( job , new Path ( args [0]) ) ;
FileOutputFormat . setOutputPath ( job , new Path ( args [1]) ) ;
System . exit ( job . waitForCompletion ( true ) ? 0 : 1) ;
}
}
```

b) The Sum of all the Integers

Algorithm:

```
import java . io . IOException ;
import java . util . StringTokenizer ;
import org . apache . hadoop . conf . Configuration ;
import org . apache . hadoop . fs . Path ;
import org . apache . hadoop . io . IntWritable ;
import org . apache . hadoop . io . Text ;
import org . apache . hadoop . mapreduce . Job ;
import org . apache . hadoop . mapreduce . Mapper ;
import org . apache . hadoop . mapreduce . Reducer ;
import org . apache . hadoop . mapreduce . lib . input . FileInputFormat ;
import org . apache . hadoop . mapreduce . lib . output . FileOutputFormat ;
public class sum {
public static class TokenizerMapper
extends Mapper < Object , Text , Text , IntWritable >{

4

// private final static IntWritable one = new IntWritable (1) ;
private Text word = new Text (" Sum : ") ;
public void map ( Object key , Text value , Context context
) throws IOException , InterruptedException {
StringTokenizer itr = new StringTokenizer ( value . toString () ) ;
int sum = 0;
int num ;
while ( itr . hasMoreTokens () ) {
num = Integer . parseInt ( itr . nextToken () ) ;
sum += num ;
}
context . write ( word , new IntWritable ( sum ) ) ;
}
}
public static class IntSumReducer
extends Reducer < Text , IntWritable , Text , IntWritable > {
private IntWritable result = new IntWritable () ;
public void reduce ( Text key , Iterable < IntWritable > values ,

Context context
) throws IOException , InterruptedException {

int sum = 0;
int num ;
for ( IntWritable val : values ) {
num = val . get () ;
sum += num ;
}
result . set ( sum ) ;
context . write ( key , result ) ;
}
}
public static void main ( String [] args ) throws Exception {
```

```
Configuration conf = new Configuration () ;
Job job = Job . getInstance ( conf , " find sum ") ;
job . setJarByClass ( sum . class ) ;
job . setMapperClass ( TokenizerMapper . class ) ;
job . setCombinerClass ( IntSumReducer . class ) ;
job . setReducerClass ( IntSumReducer . class ) ;
job . setOutputKeyClass ( Text . class ) ;
job . setOutputValueClass ( IntWritable . class ) ;
FileInputFormat . addInputPath ( job , new Path ( args [0]) ) ;
FileOutputFormat . setOutputPath ( job , new Path ( args [1]) ) ;
System . exit ( job . waitForCompletion ( true ) ? 0 : 1) ;
}
}
```

Thank You.