Week 9

SPARK :-

12015/2025

PAGE NO :

DATE :

(1) using RDD and flatmap count new many times each word appears in file.

```
from pyspark import SparkContent
SC = SparkContent (appname = "word count")

filepath = " path . text "
text-file = SC.textt-file (File_path)
words = text-File. flatmap (lamda : line
                              split())

word_pairs = words. map(lambda)
result = filter-words. collect ()
for word, count in result :
    print ( word_count )

SC. stop()
```

output:

```
text_rdd = Sc. parallelize ([
    "hello world",
    "hello spark",
    " hello helloworld spark spark spark"
])
```

```
[('hello', 4), ('spark', 4)]
```

PAGE NO :
DATE :

(2) Java Hadoop Mapreduce Program: Top 10 frequent words in Alphabetical order.

Mapper class

```
import java.io.IOException;
import org.apache.hadoop.io.Intwritable;
import org.apache.hadoop.io.Longwritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class WordCountMapper extends Mapper<Longwrit
    -able, Text, text, IntWritable> {
  Private final static IntWritable one = new IntWritable(1);
  private Text word = new Text();

  Public void map(LongWritable key, Text value, Context
      context) throws IOException, InterruptedException {
    String[] words = value.toString().toLowercase().
        split("\\W+");
    for (String w : words) {
      if (!w.isEmpty()) {
        word.set(w);
        context.write(word, one);
      }
    }
  }
}
```

Reducer class

```
import java.io.IOException;
import java.util.*;
import org.apache.hadoop.io.Intwritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
```

PAGE NO :
DATE :

```
Public class wordCountReducer extends Reducer
 <Text, IntWritable, Text, IntWritable> {
    private map<string, Integer> countMap = new
                                      HashMap<> ();
 Public void reduce(Text key, Iterable<IntWritable>
      values, context context) {
    int sum = 0;
    for (IntWritable val : values) {
        sum += val.get();
    }
    countMap.put(key.toString(), sum);
 }}
  Protected void cleanup(context context) throws
   IOException, InterruptedException {
   countMap.entrySet().Stream()
      .sorted(map.Entry.<String, Integer>comparin
        ByValue(Comparator.reversorder())
        .thenComparing(map.Entry.comparingByKey

      .limit(10)
      .forEach(entry -> {
         try {
            context.write(new Text(entry.getKey()),
         new IntWritable(entry.getValue()));
      } catch (IOException | InterruptedExcept e)
         e.printStackTrace();
      }
   });
 }
}
```

```
Public class Top10words {
    Public Static void main(String[] args) ,
            throws Exception {

     configuration conf = new configuration();
     Job job = Job.getInstance(conf, "Top 10 Tn");

     job.setMapOutputKeyclass(Text.class);
     job.setMapOutputValueclass(IntWritable.class);
     job.set output key class (Text.class);

     system.exit(job.waitForCompletion(true) ? 0 : 1);
  }
}
```

Output :-

apple banana apple mango banana orange
apple apple banana grape mango apple
banana grape.

```
apple 5
banana 4
grape 2
mango 2
orange 1
```

③ Cleaning the text :

```
from pyspark import SparkContext
import re.
```

PAGE NO :
DATE :

```
def clean-text (rdd) :
    clean_rdd = rdd.map (lambda : re.sub)
    return clean-rdd

def main() :
    sc = SparkContent (appName = "Text clean")
    ssc = StreamingContent (sc, 1)
    Port = 999
    lines = ssc.socket Text Stream (port)
    cleaned_lines = clean-text (lines)
    cleaned_lines.pprint ( )
    ssc.start ( )
    ssc.awaitTermination ( )

    if  name == "main" :
        main ( )
```

Output :-

```
This is an example of streaming text
spark is very useful for Big Data processing

Example streaming text
spark useful big data processing.
```

[illegible] 20/5/25