RDD - Text data - 1 pd

NOTE : Create individual RDDs wherever required in the following process.

Consider the following text file.

India’s football team captain Sunil Chhetri recently took to Twitter pleading fans to support his team in a four-nation tournament held in Mumbai. Virat Kohli, the captain of national cricket team, doesn’t have to issue such appeals as he mostly plays in front of a full house. But football is gaining a following in India, more so after the launch of the professional Indian Super League. Over the next month, cricket will take a backseat as India joins the world in watching the FIFA World Cup.

1. Create RDD for above text file with 3 partitions.

2. display the contents of file using RDD.

3. create individual RDD for first 3 lines.

4. Give a wordcount for each token in the file and save it to another file.

5. Differentiate between functionality of map and flatmap.

6. Display data in each partition individually and load them into separate files.

Solution –

def main(args: Array[String]): Unit = {

val filePath = "C:\\Users\\H239006\\Desktop\\textFileTopgear.txt"

val conf = new SparkConf().setAppName("textFiles").setMaster("local[\*]")

val sc = new SparkContext(conf)

val spark = SparkSession.builder().appName("textFiles").master("local[\*]").getOrCreate()

//1.

val textRdd = spark.read.textFile(filePath).rdd

//2.

textRdd collect() foreach println

//3.

val sepRDD = textRdd.map(x => x.split(" "))

sepRDD.collect().foreach(newRDD => {

val newRdd = sc.parallelize(newRDD)

newRDD collect() foreach println

})

//4.

val textDF = sc.textFile(filePath)

import spark.implicits.\_

val count = textDF.flatMap(line => line.split(" ")).map(x => (x,1)).reduceByKey(\_ + \_)

println(count)

try {

count.coalesce(1).saveAsTextFile("C:\\Users\\H239006\\Desktop\\countValue.txt")

} catch {

//We can have all the other kinds of error catching methods and add this one in the default case

case e: Exception => e.printStackTree()

}

//5.

// Map will return only one value per elemnt while flatmap will return 0 or more values per element for example in our previous

//example if we would have used map instead of flatmap it will give the data of line length, so the output will be just 3 value

// for map as there are only 3 lines in this rdd while for flatmap the values retured are for each word in the result, not the lines.

//6.

try {

textDF.mapPartitionsWithIndex((index: Int, it: Iterator[String]) => {

it.toList.map(x => x).iterator

}).saveAsTextFile("C:\\Users\\H239006\\Desktop\\textDFPartition.txt")

} catch {

//We can have all the other kinds of error catching methods and add this one in the default case

case e: Exception => e.printStackTree()

}

}