POPU

Date 08/02/2021

Page No. 17

Topic

Experiment-5

Q. Actions

1 create a RDD from mydata.txt file using textFile()
→ data = sc.textFile("FileStore/tables/mydata.txt")
data.collect()

2 count the number of elements in text file.
→ data.count()

3 Display all the elements of a RDD using collect()
→ data.collect()

4 Display first element of RDD.
→ data.first()

5 Display first 3 element of RDD.
→ data.take(3)

6 CountByKey
→ a = sc.parallelize(("W", "B", "R", "W", "O", "WH", "J", "W", "J"))
a = a.map(lambda K: (K, 1))
a.countByKey().items()

Teacher's Signature

7. CountByValues

→ r = sc.parallelize(("W", "B", "R", "W", "O", "WH", "J", "W", "J"))

```
r.map(lambda K:(K,1)).countByvalue().items()
```

8. Save the elements of the RDD as a textfile using saveAsTextFile.

→ r = sc.parallelize([1, 2, 3, 4])

```
r.saveAsTextFile("res1.txt")
```

Outputs: Date - 08/02/2021

```
Out[1]: ['1 2 3 4 5 6 7 8 9 10',
'a b c d',
'x c v b f w'
'1 2 3 4 5 6'
'a b',
.
.
'b']

Out[2]: 10

Out[3]: ['1 2 3 4 5 6 7 8 9 10',
'a b c d'
'x c v b f w',
'sd tf',
'1 2 3 4 5 6',
'a, b',
'a',
'a',
'b',
'b']

Out[4]: '1 2 3 4 5 6 7 8 9 10'

Out[5]: ['1 2 3 4 5 6 7 8 9 10', 'a b c d', 'x c v b f w']
```

Teacher's Signature ....

Out[6]:
```
dict_items([('W', 3), ('B', 1), ('R', 1), ('O', 1)
            ('WH', 1), ('J', 2)])
```

Out[7]:
```
dict_items([(('W', 1), 3), (('B', 1), 1), (('R', 1), 1),
         (('O', 1), 1), (('WH', 1), 1), (('J', 1), 2)])
```

Out[8]:
```
org.apache.hadoop.mapred.FileAlreadyExistsException:
    Output directory dfs:/res1.text already exiss
```

14 Union

```
⇒ a=sc.textFile("dbfs:/FileStore/tables/city1.txt").
b =sc.textFile("dbfs:/FileStore/tables/city2.txt")
rdd =a.union(b)
rdd.collect()
```

15 Zip

```
→ a= sc.textFile ("dbfs:/Filestore/tables/city.txt")
b=sc.textFile ("dbfs:/FileStore/tables/ city2.txt")
rdd = a.zip(b)
rdd.collect()
```

16 Subtract

```
→ a=sc.textFile("dbfs:/FileStore/tables/ city1.txt")
b= sc.textFile ("dbfs:/ FileStore/ tables/city2.txt")
rdd =a.subtract(b)
rdd.collect()
```

17 Intersection

```
→ a = sc.textFile ("dbfs: /FileStore/tables/city1.txt")
b= sc.textFile("dbfs:/FileStore / tables/ city2.txt")
rdd = a.Intersect(b)
rdd.collect().
```

⑬ cobroup:

```
⇒ rdd1 = sc.parallelize ([("A",1), ("B", 2), ("C", 3)])
rdd2= sc.parallelize([("B", 5), ("E", 5)])
cg = rdd1.cogroup(rdd2)
cg.map(lambda x:(x[0], list(x[1][0]), list(x[1][1]))).
        collect()
```

```
Out[15]: [('Los Angeles', 'seattle'),
          ('chicago', 'Las vegas'),
          ('Dallas', 'chicago'),
          ('san franciso', 'Pittsburgh'),
          ('seattle, 'Dallas')]

Out[16]: ['San francisco', 'Los Angeles']

Out[17]: ['Dallas', 'chicago', 'seattle']

Out[18]: [('B', [2], [5]), ('E', [], [5]), ('c', [3], [])
          ('A', [1], [])]
```