*Word count program in pyspark*

\*import retail\_db data into hdfs

$ sqoop import-all-tables --connect 'jdbc:mysql://localhost:3306/retail\_db' --username root --password cloudera -m 1 --warehouse-dir /user/cloudera/retail\_db

$ hadoop fs -ls /user/cloudera/retail\_db/orders

\*

>>> from pyspark import SparkContext

>>> sc.stop()

>>> from pyspark import SparkConf

>>> conf = SparkConf().setAppName("rdd1").set('spark.executor.memory','100GB')

>>>sc=SparkContext(conf=conf)

!--

>>> order\_rdd=sc.textFile("hdfs:///user/cloudera/retail\_db/orders/part-m-00000")

>>> order\_rdd.take(10)

>>> order\_items\_rdd=sc.textFile("hdfs:///user/cloudera/retail\_db/order\_items/part-m-00000")

>>> order\_items\_rdd.take(10)

>>> categories\_rdd=sc.textFile("hdfs:///user/cloudera/retail\_db/categories/part-m-00000")

>>> categories\_rdd.take(10)

>>> customers\_rdd=sc.textFile("hdfs:///user/cloudera/retail\_db/customers/part-m-00000")

>>> customers\_rdd.take(10)

>>> departments\_rdd=sc.textFile("hdfs:///user/cloudera/retail\_db/departments/part-m-00000")

>>> departments\_rdd.take(10)

>>> products\_rdd=sc.textFile("hdfs:///user/cloudera/retail\_db/departments/part-m-00000")

>>> products\_rdd.take(10)

--!

!--

\*\*\* word count program

txtfile:

$ vi inputdata.txt

1. to read input file

>>> input\_data=sc.textFile("file:///home/cloudera/inputdata.txt")

>>> print(input\_data)

>>> input\_data.take(20)

2. split input data into seperate word

>>> flatmap\_input\_data=input\_data.flatMap(lambda z : z.split(" "))

>>> flatmap\_input\_data.take(20)

-for count all the data

>>> flatmap\_input\_data.count()

3. assign 1 to each word

>>> map\_flatmap\_input\_data = flatmap\_input\_data.map(lambda z : (z,1))

>>> map\_flatmap\_input\_data.take(20)

4. count no. of word per key word

>>> reduce\_map\_flatmap\_input\_data = map\_flatmap\_input\_data.reduceByKey(lambda x,z : x+z)

-for collect all data from rdd

>>> reduce\_map\_flatmap\_input\_data.collect()

5. saveAsTextFile

>>> reduce\_map\_flatmap\_input\_data.saveAsTextFile("hdfs:///user/inputdata2.txt")

\*\*\*\*at hdfs

$ hadoop fs -ls /user/inputdata2.txt

$ hadoop fs -cat /user/inputdata2.txt/part-00000 # parts are depends on data length.

--!

!--

\*\*\*join

>>> rdd01=sc.parallelize([("one",1),("two",1),("three",4)])

>>> rdd02=sc.parallelize([("one",1),("two",1),("four",1)])

>>> rdd03=rdd01.join(rdd02)

>>> rdd03.collect()

output : [('two', (1, 1)), ('one', (1, 1))]

\*\*\*union

>>> rdd04=rdd01.union(rdd02)

>>> rdd04.collect()

output : [('one', 1), ('two', 1), ('three', 4), ('one', 1), ('two', 1), ('four', 1)]

--!

*Word count*

from pyspark import SparkContext

sc.stop()

from pyspark import SparkConf

conf = SparkConf().setAppName("rdd1").set('spark.executor.memory','100GB')

sc=SparkContext(conf=conf)

input\_data=sc.textFile("file:///home/cloudera/inputdata.txt")

flatmap\_input\_data=input\_data.flatMap(lambda z : z.split(" "))

map\_flatmap\_input\_data = flatmap\_input\_data.map(lambda z : (z,1))

reduce\_map\_flatmap\_input\_data = map\_flatmap\_input\_data.reduceByKey(lambda x,z : x+z)

reduce\_map\_flatmap\_input\_data.collect()

reduce\_map\_flatmap\_input\_data.saveAsTextFile("hdfs:///user/inputdata3.txt")

*Word count*

from pyspark import SparkContext

sc.stop()

sc=SparkContext.getOrCreate()

>>> def getword(lines):

... word=[]

... for line in lines:

... for i in line.split(","):

... word.append((i,1))

... return word

...

>>> for i in sc.textFile("file:///home/cloudera/abc.txt").mapPartitions(lambda lines:getword(lines)).reduceByKey(lambda x,y:x+y).take(10):

... print(i)

...

(u'one', 4)

(u'three', 2)

(u'two', 3)