In [8]:
```
words = sc.parallelize(["scala",
   "java",
   "hadoop",
   "spark",
   "akka",
   "spark vs hadoop",
   "pyspark",
   "pyspark and spark"])
```

map(f, presevesPartitioning = False)

In [9]:
```
words_map = words.map(lambda x: (x, 1))
mapping = words_map.collect()
print("Key value pair -> %s" % (mapping))
```

```
Key value pair -> [('scala', 1), ('java', 1), ('hadoop', 1), ('spark', 1),
('akka', 1), ('spark vs hadoop', 1), ('pyspark', 1), ('pyspark and spark',
1)]
```

reduce(f) - ACTION

In [10]:
```
from pyspark import SparkContext
from operator import add
sum = sc.parallelize([1, 2, 3, 4, 5])
adding = sum.reduce(add)
print("Adding all the elements in RDDs : %i" % (adding))
```

```
Adding all the elements in RDDs : 15
```

cache()

In [11]:
```
words.cache()
caching = words.persist().is_cached
print("Words got cached > %s" % (caching))
```

```
Words got cached > True
```

join(other, numPartition = None)

In [12]:
```
from pyspark import SparkContext
x = sc.parallelize([("pyspark", 1), ("hadoop", 3)])
y = sc.parallelize([("pyspark", 2), ("hadoop", 4)])
joined = x.join(y)
mapped = joined.collect()
print("Join RDD -> %s" % (mapped))
```

```
Join RDD -> [('pyspark', (1, 2)), ('hadoop', (3, 4))]
```

count() - ACTION

In [13]:
```
counts = words.count()
print ("Number of elements in RDD -> %i" % (counts))
```

```
Number of elements in RDD -> 8
```

collect() - ACTION

In [14]:
```
coll = words.collect()
print ("Elements in RDD -> %s" % (coll))
```

```
Elements in RDD -> ['scala', 'java', 'hadoop', 'spark', 'akka', 'spark vs h
adoop', 'pyspark', 'pyspark and spark']
```

first()- ACTION

In [15]:
```
coll1 = words.first()
print ("Elements in RDD -> %s" % (coll1))
```

```
Elements in RDD -> scala
```

take(n)-ACTION

In [16]:
```
coll2 = words.take(2)
print ("Elements in RDD -> %s" % (coll2))
```

```
Elements in RDD -> ['scala', 'java']
```

distinct-TRANSFORMATIONS

In [17]:
```
words.distinct().count()
```

Out[17]:
```
8
```

filter-TRANSFORMATIONS

In [18]:
```
rdd = sc.parallelize([1, 2, 3, 4, 5])
rdd.filter(lambda x: x % 2 == 0).collect()
[2, 4]
```

Out[18]:
```
[2, 4]
```

"foreach" and accumulator operation

In [19]:
```
# "foreach" and accumulator operation
accum = sc.accumulator(0)
sc.parallelize(range(1, 100+1)).foreach(lambda x: accum.add(x))
accum.value
```

Out[19]:
```
5050
```

"countByKey" Operation

In [20]:
```
# "countByKey" Operation
sc.parallelize(list("1223334444")*1000).countByKey()
```

Out[20]:
```
defaultdict(int, {'1': 1000, '2': 2000, '3': 3000, '4': 4000})
```

"takeOrdered" operation

the arugment is the number of elements to take.

In [21]:
```
sc.parallelize([10,4,5,3,2]).takeOrdered(3)
```

Out[21]:
```
[2, 3, 4]
```

"cartesian" operation

In [22]:
```
# Initialize data
dat_1=sc.parallelize(range(1, 5+1))
dat_2=sc.parallelize(range(1, 5+1))
```

In [23]:
```
dat_1.count()
dat_2.count()
dat_1.cartesian(dat_2).count()
dat_1.cartesian(dat_2).collect()
```

Out[23]:
```
[(1, 1),
 (1, 2),
 (1, 3),
 (1, 4),
 (1, 5),
 (2, 1),
 (2, 2),
 (2, 3),
 (2, 4),
 (2, 5),
 (3, 1),
 (3, 2),
 (3, 3),
 (3, 4),
 (3, 5),
 (4, 1),
 (4, 2),
 (4, 3),
 (4, 4),
 (4, 5),
 (5, 1),
 (5, 2),
 (5, 3),
 (5, 4),
 (5, 5)]
```

"union" operation

In [24]:
```
dat=dat_1.union(dat_2)
dat.collect()
```

Out[24]:
```
[1, 2, 3, 4, 5, 1, 2, 3, 4, 5]
```

"intersection" operation

In [25]:
```
dat_1.intersection(dat_1).collect()
dat_1.intersection(dat_2).collect()
```

```
/Users/elcinyutes/Apache_spark/python/lib/pyspark.zip/pyspark/shuffle.py:60
: UserWarning: Please install psutil to have better support with spilling
/Users/elcinyutes/Apache_spark/python/lib/pyspark.zip/pyspark/shuffle.py:60
: UserWarning: Please install psutil to have better support with spilling
/Users/elcinyutes/Apache_spark/python/lib/pyspark.zip/pyspark/shuffle.py:60
: UserWarning: Please install psutil to have better support with spilling
/Users/elcinyutes/Apache_spark/python/lib/pyspark.zip/pyspark/shuffle.py:60
: UserWarning: Please install psutil to have better support with spilling
/Users/elcinyutes/Apache_spark/python/lib/pyspark.zip/pyspark/shuffle.py:60
: UserWarning: Please install psutil to have better support with spilling
/Users/elcinyutes/Apache_spark/python/lib/pyspark.zip/pyspark/shuffle.py:60
: UserWarning: Please install psutil to have better support with spilling
```

Out[25]:

```
[1, 2, 3, 4, 5]
```

In [ ]:

```
```