# Spark函数讲解：cogroup

将多个RDD中同一个Key对应的Value组合到一起。

## 函数原型

```
def cogroup[W1, W2, W3](other1: RDD[(K, W1)],
    other2: RDD[(K, W2)], other3: RDD[(K, W3)], partitioner: Partitioner) :
    RDD[(K, (Iterable[V], Iterable[W1], Iterable[W2], Iterable[W3]))]
def cogroup[W1, W2, W3](other1: RDD[(K, W1)],
    other2: RDD[(K, W2)], other3: RDD[(K, W3)], numPartitions: Int) :
    RDD[(K, (Iterable[V], Iterable[W1], Iterable[W2], Iterable[W3]))]
def cogroup[W1, W2, W3](other1: RDD[(K, W1)],
    other2: RDD[(K, W2)], other3: RDD[(K, W3)])
    : RDD[(K, (Iterable[V], Iterable[W1], Iterable[W2], Iterable[W3]))]
def cogroup[W1, W2](other1: RDD[(K, W1)], other2: RDD[(K, W2)],
     partitioner: Partitioner)
    : RDD[(K, (Iterable[V], Iterable[W1], Iterable[W2]))]
def cogroup[W1, W2](other1: RDD[(K, W1)], other2: RDD[(K, W2)],
    numPartitions: Int)
    : RDD[(K, (Iterable[V], Iterable[W1], Iterable[W2]))]
def cogroup[W1, W2](other1: RDD[(K, W1)], other2: RDD[(K, W2)])
    : RDD[(K, (Iterable[V], Iterable[W1], Iterable[W2]))]
def cogroup[W](other: RDD[(K, W)], partitioner: Partitioner) :
    RDD[(K, (Iterable[V], Iterable[W]))]
def cogroup[W](other: RDD[(K, W)], numPartitions: Int): RDD[(K, (Iterable[V], Iterable[W]))]
def cogroup[W](other: RDD[(K, W)]): RDD[(K, (Iterable[V], Iterable[W]))]
```

cogroup函数原型一共有九个（真多）！最多可以组合四个RDD。

## 实例

```
/**
 * User: 过往记忆
 * Date: 15-03-10
 * Time: 下午06:30
 * bolg:
 * 本文地址：/archives/1280
 * 过往记忆博客，专注于hadoop、hive、spark、shark、flume的技术博客，大量的干货
 * 过往记忆博客微信公共帐号：iteblog_hadoop
```

```
 */
scala> val data1 = sc.parallelize(List((1, "www"), (2, "bbs")))
data1: org.apache.spark.rdd.RDD[(Int, String)] =
    ParallelCollectionRDD[32] at parallelize at <console>:12

scala> val data2 = sc.parallelize(List((1, "iteblog"), (2, "iteblog"), (3, "very")))
data2: org.apache.spark.rdd.RDD[(Int, String)] =
    ParallelCollectionRDD[33] at parallelize at <console>:12

scala> val data3 = sc.parallelize(List((1, "com"), (2, "com"), (3, "good")))
data3: org.apache.spark.rdd.RDD[(Int, String)] =
    ParallelCollectionRDD[34] at parallelize at <console>:12

scala> val result = data1.cogroup(data2, data3)
result: org.apache.spark.rdd.RDD[(Int, (Iterable[String],
    Iterable[String], Iterable[String]))] = MappedValuesRDD[38] at cogroup at <console>:18

scala> result.collect
res30: Array[(Int, (Iterable[String], Iterable[String], Iterable[String]))] =
Array((1,(CompactBuffer(www),CompactBuffer(iteblog),CompactBuffer(com))),
(2,(CompactBuffer(bbs),CompactBuffer(iteblog),CompactBuffer(com))),
(3,(CompactBuffer(),CompactBuffer(very),CompactBuffer(good))))
```

从上面的结果可以看到，data1中不存在Key为3的元素（自然就不存在Value了），在组合的过程中将data1对应的位置设置为CompactBuffer()了，而不是去掉了。