Repartition vs Coalesce

```
import pyspark
from pyspark.sql import SparkSession



spark = SparkSession.builder.master("local[1]") \
                    .appName('repartitionvscoalesce') \
                    .getOrCreate()

rdd1 = spark.sparkContext.parallelize(range(0,25), 6)
results=rdd1.glom().collect()
for i in results:
    print(i)
```

glom() function gets the values for each partition

Output

```
[0, 1, 2, 3]
[4, 5, 6, 7]
[8, 9, 10, 11]
[12, 13, 14, 15]
[16, 17, 18, 19]
[20, 21, 22, 23, 24]
```

p1
p2
p3
p4
p5
p6

Repartition

```python
import pyspark
from pyspark.sql import SparkSession



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

rdd1 = spark.sparkContext.parallelize(range(0,25), 6)
rdd2= rdd1.repartition(4)
results=rdd2.glom().collect()
for i in results:
    print(i)
```

Output

```
[6, 7, 8, 15, 16, 17, 18, 19, 20]
[0, 1, 2, 3, 4, 5]
[21, 22, 23, 24]
[9, 10, 11, 12, 13, 14]
```

p1
p2
p3
p4

In case of repartition there is full shuffle of data across all the partitions e.g p5 to p1, p1 to p2, p6 to p3, p3 to p4 etc.

All partitions Shuffled!!!

Coalesce

```python
from pyspark.sql import SparkSession



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


rdd1 = spark.sparkContext.parallelize(range(0,25), 6)
rdd2= rdd1.coalesce(4)
results=rdd2.glom().collect()
for i in results:
    print(i)
```

Output

```
[0, 1, 2, 3]
[4, 5, 6, 7, 8, 9, 10, 11]
[12, 13, 14, 15]
[16, 17, 18, 19, 20, 21, 22, 23, 24]
```

p1
p2
p3
p4

In case of coalesce there no full shuffle of data.

p1-Not Shuffled, p2-Not Shuffled, p3- Shuffled to p2,

p4- Not Shuffled, p5-Not Shuffled, p6-Shuffled to p5

Only 2 partitions shuffled out of 6!