reduceByKey( ) Transformation

groupByKey() Transformation cannot be used on large datasets. To solve this problem, let us look us at reduceByKey() Transformation. reduceByKey is optimized with a map side combine. This means it performs the merging locally on each mapper for each key before sending results to a reducer operation. After that, the values are combined for each key using an associative and commutative reduce function. By this, less elements are sent over the network.

In [1]:
```python
from pyspark.sql import SparkSession
import pyspark

spark = SparkSession \
        .builder \
        .master("local[4]") \
        .appName("Sample Transformation") \
        .enableHiveSupport() \
        .getOrCreate()

#path of the data file on the local machine
data_file = '/Users/vaishaliyasala/Desktop/Github/Spark/Exercise_Dependencies/sales_data.csv'

#Read the csv into a dataframe
df = spark.read.csv(data_file, header = True )

df1 = df.select(df["InvoiceNo"],df["UnitPrice"],df["Quantity"]).repartition(4)

print(df1.printSchema())

#Creating view of the dataframe of with 3 required columns and sample of 2% of data
sample_df = df1.sample(0.02,134)

sample_df.show()
```

```
22/10/13 13:54:27 WARN Utils: Your hostname, Vaishalis-MacBook-Pro.local resolves to a loopback address: 127.0.
0.1; using 192.168.0.105 instead (on interface en0)
22/10/13 13:54:27 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
22/10/13 13:54:27 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builti
n-java classes where applicable
22/10/13 13:54:28 WARN Utils: Service 'SparkUI' could not bind on port 4040. Attempting port 4041.
22/10/13 13:54:28 WARN Utils: Service 'SparkUI' could not bind on port 4041. Attempting port 4042.
```

```
root
 |-- InvoiceNo: string (nullable = true)
 |-- UnitPrice: string (nullable = true)
 |-- Quantity: string (nullable = true)

None
+---------+---------+--------+
|InvoiceNo|UnitPrice|Quantity|
+---------+---------+--------+
|   536464|     2.55|       1|
|   536408|     0.65|      36|
|   536412|     1.65|       5|
|   536412|     1.65|       3|
|   536464|     1.95|       1|
|   536415|     2.95|       3|
|   536399|     1.85|       6|
|   536401|     5.95|       1|
|   536409|     0.65|      12|
|   536520|      2.1|       2|
|   536409|     2.95|       1|
|   536392|      165|       1|
|   536414|        0|      56|
|   536464|     1.25|       3|
|   536420|     2.95|       6|
|   536396|     1.06|       6|
|   536520|     1.95|       5|
|   536389|     4.95|       8|
|   536446|     0.42|      10|
|   536375|     6.95|       4|
+---------+---------+--------+
only showing top 20 rows
```

In [2]:
```python
# apply a map() transformation to rdd to create (K, V) pairs

#In this key-value pair, key is the InvoiceNo and the number is the value

#whereas the number is obtained from UnitPrice*Qunatity


import decimal


def get_price(x3):
    try:
        UnitPrice = decimal.Decimal(x3[2])
        convert = UnitPrice * decimal.Decimal(x3[1])
    except decimal.InvalidOperation:
            print("Invalid input")
    key = x3[0]
    number = convert
    return (key, number)


rdd1 = df1.rdd.map(lambda x : get_price(x))
print("Number of elements =",len(rdd1.collect()))
print("Number of Partitions =",rdd1.getNumPartitions())


#Showing the Result for the dataframe sample sample_df
sample_df_rdd = sample_df.rdd.map(lambda x : get_price(x))

print(sample_df_rdd.collect())
```

```
Number of elements = 999
Number of Partitions = 4
[('536464', Decimal('2.55')), ('536408', Decimal('23.40')), ('536412', Decimal('8.25')), ('536412', Decimal('4.
95')), ('536464', Decimal('1.95')), ('536415', Decimal('8.85')), ('536399', Decimal('11.10')), ('536401', Decim
al('5.95')), ('536409', Decimal('7.80')), ('536520', Decimal('4.2')), ('536409', Decimal('2.95')), ('536392', D
ecimal('165')), ('536414', Decimal('0')), ('536464', Decimal('3.75')), ('536420', Decimal('17.70')), ('536396',
Decimal('6.36')), ('536520', Decimal('9.75')), ('536389', Decimal('39.60')), ('536446', Decimal('4.20')), ('536
375', Decimal('27.80')), ('536373', Decimal('6.36')), ('536408', Decimal('9.90'))]
```

In [3]:
```python
# apply a reduceByKey() transformation to rdd1 to create a (key, value) pair

#  where key is the InvoiceNo and value is number

#we can create more partitions than its parent RDD.

rdd2 = rdd1.reduceByKey(lambda a, b: (a,b),10)


print(len(rdd2.collect()),rdd2.getNumPartitions())
print("Number of elements =",len(rdd2.collect()))
print("Number of Partitions =",rdd2.getNumPartitions())
```

```
66 10
Number of elements = 66
Number of Partitions = 10
```

From result of Input block 2 and 3, we can see the number of elements decreased because they are merged together when they have the same key. Additionally, it is optimized with a map side combine.

In [4]:
```python
#Below we can the result of reduceByKey() applied on sample_df_rdd

print(sample_df_rdd.reduceByKey(lambda a, b: (a,b),10).collect())
```

```
[('536464', ((Decimal('2.55'), Decimal('1.95')), Decimal('3.75'))), ('536396', Decimal('6.36')), ('536399', Dec
imal('11.10')), ('536520', (Decimal('4.2'), Decimal('9.75'))), ('536375', Decimal('27.80')), ('536414', Decimal
('0')), ('536373', Decimal('6.36')), ('536389', Decimal('39.60')), ('536412', (Decimal('8.25'), Decimal('4.9
5'))), ('536409', (Decimal('7.80'), Decimal('2.95'))), ('536408', (Decimal('23.40'), Decimal('9.90'))), ('53640
1', Decimal('5.95')), ('536420', Decimal('17.70')), ('536446', Decimal('4.20')), ('536415', Decimal('8.85')),
('536392', Decimal('165'))]
```