**Shuffle operations in Spark**

In Spark, large dataset can be partitioned across the spark cluster. The shuffle operation occurs when particular data has to be shuffled across the cluster. For example, if there are 3 partitions across the cluster, each with its own executor, each partition might contain different group of data. So, if partition 1 has data grouped by (a, b, c), partition 2 data (c, b, a), and partition 3 data (b, a, c), the shuffle operation will make sure that partition 1 will have only single group of data, either a, b or c, and the same operation will be applied to partition 2 and 3 across the cluster. What shuffle operation is doing is basically redistributing the data across different executor within the cluster

**What happens in terms of memory when you are using shuffling?**

In Spark, there is a term called RDD (Resilient Distributed Datasets). RDD is a collection of Read-Only or Immutable object which are partitioned across the cluster. Spark can run the map task on all partitions and group the data in memory. If the data is bigger than the memory, spark would use the available disk space to store the rest of the data or map task. The shuffle operation on memory is usually faster. Another important point to note is the caching operation during shuffling. This is the process in which spark stores the shuffled data to reuse for future calculation/computation