**Problem statement - Analyse impact of different number of mapper and reducer on same definition as practical 4. Prepare a conclusive report on analysis.**

Number of mappers set to run are completely dependent on

1) File Size and

2) Block Size

The number of **Mappers** that [**Hadoop**](https://data-flair.training/blogs/hadoop-introduction-tutorial-quick-guide/) creates is determined by the number of [**Input Splits**](https://data-flair.training/blogs/inputsplit-in-hadoop-mapreduce/) you have in your Data.

No. of Mappers = No. of Input Splits.

So, in order to control the Number of Mappers, you have to first control the Number of Input Splits Hadoop creates before running your [**MapReduce**](https://data-flair.training/blogs/hadoop-mapreduce-introduction-tutorial-comprehensive-guide/) program. One of the easiest ways to control it is setting the property **‘mapred.max.split.size’**while running your MR program.

**Example:**  
Let’s assume your Input data is 1 TB. So, number of Physical [**Data Blocks**](https://data-flair.training/blogs/data-blocks-hdfs-hadoop-distributed-file-system/) = (1 \* 1024 \* 1024 / 128) = 8192 Blocks.

By Default, if you don’t specify the Split Size, it is equal to the Blocks (i.e.) 8192. Thus, your program will create and execute 8192 Mappers !!!

Let’s say you want to create only 100 Mappers to handle your job.  
As mentioned above, 100 Mappers means 100 Input Splits. So each Split size should be set to (1 \* 1024 \* 1024 / 100) = 10486 MB

Execute it as follows:

hadoop jar <your-script.jar> <main class> -Dmapred.max.split.size=10486 <input file> <output directory>

 When reducer is set to 0 in MapReduce? Why?

If we set the number of [**Reducer**](https://data-flair.training/blogs/reducer-in-hadoop-mapreduce/) to 0 (***by setting job.setNumreduceTasks(0)***), then no reducer will execute and no aggregation will take place. In such case, we will prefer [**“Map-only job”**](https://data-flair.training/blogs/map-only-job-in-hadoop-mapreduce-example/) in [**Hadoop**](https://data-flair.training/blogs/hadoop-introduction-tutorial-quick-guide/).

**Map-Only job**

In Map-Only job, the map does all task with its [**InputSplit**](https://data-flair.training/blogs/inputsplit-in-hadoop-mapreduce/) and the reducer do no job. [**Mapper**](https://data-flair.training/blogs/mapper-in-hadoop-mapreduce/) output is the final output. Between map and reduce phases there is key, sort, and shuffle phase. Sort and shuffle phase are responsible for sorting the keys in ascending order.

Then grouping values based on same keys. This phase is very expensive. If reduce phase is not required we should avoid it. Avoiding reduce phase would eliminate sort and shuffle phase as well. This also saves network congestion. As in shuffling an output of mapper travels to the reducer, when data size is huge, large data travel to the reducer.

In MapReduce job, mapper output is written to local disk before sending to Reducer but in the map-only job, this output is directly written to [**HDFS**](https://data-flair.training/blogs/comprehensive-hdfs-guide-introduction-architecture-data-read-write-tutorial/). This further saves time and reduces cost as well.

**The number of reducer** can be set to 0 in driver class by job.setNumreduceTasks(0).This shows that there is no reducer phase and has only map phase.It is called as a [**map-only**](https://data-flair.training/blogs/map-only-job-in-hadoop-mapreduce/) job.

Map-only job:

The map-only job has only map phase.The output of [**mapper**](https://data-flair.training/blogs/hadoop-mapper-in-mapreduce/) stores directly on HDFS not on disk. The map output is final output.As it has no reducer phase, the aggregation and sorting is also not done.Generally, in map-reducer job the output after shuffling and sorting goes to the [**reducer**](https://data-flair.training/blogs/hadoop-reducer-in-mapreduce/), when the data is huge it needs good network bandwidth. As there is no shuffling and sorting in map-only job there will be less network congestion.

-D mapred.reduce.tasks=10