Session 12: Advanced MapReduce-I

**Assignment 1**

# Problem Statement

**Give a brief answers to the questions below;**

1. **What is the purpose of RecordReader in Hadoop?**

**Ans:**

RecordReader class reads the data in the format of (key, value) pair from InputSplit.

It converts the byte-oriented view of the input, provided by the InputSplit, and presents a record-oriented view for the [Mapper](https://hadoop.apache.org/docs/r2.6.4/api/org/apache/hadoop/mapred/Mapper.html) & [Reducer](https://hadoop.apache.org/docs/r2.6.4/api/org/apache/hadoop/mapred/Reducer.html) tasks for processing.

For example: our input data is splitted like

Row1: My name is

Row2: Jigyasa

The above data is read as ‘My name is Jigyasa’ by using RecordReader.

1. **What happens if the number of reducers is 0?**

**Ans:**

If the number of reducers is zero then the outputs of the map-tasks go directly to the FileSystem, into the output path set by setOutputPath (Path).

The framework does not sort the map-outputs before writing them out to the FileSystem.

1. **What is meant by Map-side and Reduce-side join in Hadoop?**

**Ans:**

Joins are the interesting features in Mapreduce.

The join performed by Mapper are known as **Map-side** Join whereas join performed by Reducer are known as **Reduce-side** join.

**Map-Side Join:**

The input data must be partitioned and stored by the same key (the join key) in each source for each map.All the records for a particular key must reside in the same partition and which is mandatory. A map-side join can be used to  
join the outputs of several jobs that had the same number of reducers, the same keys and output files that are no bigger than the HDFS block size.

**Reduce-Side Join:**

Reduce-Side joins are simpler than Map-Side joins since the input datasets need not to be structured. But it is less efficient as both datasets have to go through the MapReduce shuffle phase. The records with the same key are brought together in the reducer.

1. **What is the significance of conf.setMapper class?**

**Ans:**

Conf.setMapper class sets the mapper class and all the stuff related to map jobs such as reading the data , generating the key value pair from the mapper.

1. **Give an example scenario on the usage of counters.**

**Ans:**

Hadoop MapReduce Counter provides a way to measure the progress or the number of operations that occur within MapReduce programs. Basically, MapReduce framework provides a number of built-in counters to measure basic I/O operations, such as FILE\_BYTES\_READ/WRITTEN and Map/Combine/Reduce input/output records. These counters are very useful especially when you evaluate some MapReduce programs.

1. **Elaborate some problems which can only be solved by MapReduce and cannot be solved by PIG?**

**Ans:**

Let say, we have a problem where we want to count the population in two cities.

We have a data set and sensor list of different cities. We want to count the population by using one MapReduce for two cities.

Let us assume that one is Mumbai and the other is Noida. So I need to consider key of Mumbai city similar to Noida through which I can bring the population data of these two cities to one reducer. The idea behind this is somehow I have to instruct map reducer program – whenever you find city with the name ‘Mumbai ‘and city with the name ‘Noida’, you create the alias name which will be the common name for these two cities so that  you create a common key for both the cities and it get passed to the same reducer. For this, we have to write custom partitioner.

In MapReduce when you create a ‘key’ for city,  you have to consider ’city’ as the key. So, whenever the framework comes across a different city, it considers it as a different key. Hence, we need to use customized partitioner. There is a provision in Mapreduce only, where you can write your custom partitioner and mention if city = Mumbai or Noida then pass similar hashcode.  However, we cannot create custom partitioner in Pig. As Pig is not a framework, we cannot direct execution engine to customize the partitioner. In such scenarios, MapReduce works better than Pig.

1. **In what kind of scenarios, MR jobs will be more useful than PIG?**

**Ans**: See Ans 7

1. **What are combiners and when are these used in a MapReduce job?**

**Ans:**

A Combiner, also known as a semi-reducer, is an optional class that operates by accepting the inputs from the Map class and thereafter passing the output key-value pairs to the Reducer class.

The main function of a Combiner is to summarize the map output records with the same key. The output (key-value collection) of the combiner will be sent over the network to the actual Reducer task as input.

The combiner class is used between Mapper class and reducer class to reduce the volume of data transfer between Map and reduce.

The output data from mapper class is very huge and the data sent to the reducer class is very less in compare to the output from the mapper class.

How it works:

* A combiner does not have a predefined interface and it must implement the Reducer interface’s reduce() method.
* A combiner operates on each map output key. It must have the same output key-value types as the Reducer class.
* A combiner can produce summary information from a large dataset because it replaces the original Map output.

Although, Combiner is optional yet it helps segregating data into multiple groups for Reduce phase, which makes it easier to process.