1. What is the purpose of RecordReader in Hadoop?

* In short we can say Mapper always understand key value pair, so RecordReader reads line by line from input split and convert it into key value pair for mapper.
* A record reader uses the data within the boundaries created by the input split to generate K,V pairs. In the context of file based input, the start is the byte position in the file where the record reader should start generating K,V pairs.the end is where stop reading the records.

1. What happens if the number of reducers is 0?
   * + In this case 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.
     + 0 reducer means reduce step will be skipped and mapper output will be the final out.
2. What is meant by Map-side and Reduce-side join in Hadoop?
   * Map-side join helps in minimizing the cost that is incurred for sorting and merging in the *shuffle* and *reduce* stages.
   * Map-side join also helps in improving the performance of the task by decreasing the time to finish the task.
   * Reduce side joins are more simple than mapside joins asince the input datasets need not be structured, but its less efficient as both datasets have to go through the mapreduce shuffle phase.the record with same key are brought together in the reducer.
3. What is the significance of conf.setMapper class?

* Conf.setMapperclass sets the mapper class and all the stuff related to map job such as reading a data and generating a key-value pair out of the mapper.

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

* 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. Besides, the MapReduce Counter allows users to employ your own counters. Since MapReduce Counters are automatically aggregated over Map and Reduce phases, it is one of the easiest way to investigate internal behaviors of MapReduce programs. In this post, I’m going to introduce how to use your own MapReduce Counter. The example sources described in this post are based on Hadoop 0.21 API.

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

* Let us take a scenario where we want to count the population in two cities. I have a data set and sensor list of different cities. I want  to count the population by using one mapreduce for two cities. Let us assume that one is Bangalore and the other is Noida. So I need to consider key of  Bangalore city  similar to Noida through which I can bring the population data of these two cities to one reducer. The idea behind this is some how I have to instruct map reducer program – whenever you find city with the name ‘Bangalore‘ 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 = bangalore 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?

* Pig is a high-level platform that makes many Hadoop data analysis issues easier to execute. The language we use for this platform is:Pig Latin. A program written in Pig Latin is like a query written in SQL, where we need an execution engine to execute the query. So, when a program is written in Pig Latin, Pig compiler will convert the program into MapReduce jobs. Here, MapReduce acts as the execution engine.

1. What are combiners and when are these used in a MapReduce job?
   1. 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.
   2. 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.
   3. Usally the output from the map task is large and data transfer between map and reduce tasks will be high, since data transfer across the network is expensive and to limit the volume of data transfer between map and reduce tasks.
   4. It summarizes the map output with same key and putput of combiner will be sent over network to actual reduce task as input.