Hadoop counters for reporting custom metrics

Hadoop counters are a way to report basic statistics of a job in Hadoop.

While developing a Hadoop application, we might be interested in gathering information about the data we are analysing, like how many types of records were processed, how many invalid records were found while running the job, etc. This functionality is provided by ***Counters***in Hadoop, which are lightweight objects helping us in tracking the progress of Map and reduce tasks.

Let’s assume we have data set with timestamp as follows. If we need to check how many month-wise records are processed by our mappers we need to just use counters.

one,1386023259550

two,1389523259550

three,1389523259550

four,1389523259550

**Built In Counters In Hadoop**

Hadoop provides some inbuilt counters for every job, for purposes like counting the number of records or bytes processed. These built in counters are divided into various groups. The basic types of inbuilt counters in Hadoop are :-

**1) Map Reduce Task Counters -**These counters are used for collecting information about the various tasks in a job. An example of this type of counter is '*MAP\_INPUT\_RECORDS*'  which is used to count the number of input records read by each map task. Th output is aggregated over all the tasks in a particular job.

**2) Job Counters -**Job Counters are maintained by the JobTracker to collect statistics about the entire job. Example of this counter is '*TOTAL\_LAUNCHED\_MAPS*' which is used to count the number of map tasks that were launched over the course of a job.

**Custom Java Counters**

Map Reduce allows users to specify their own counters for performing their own counting operation. A custom counter is defined by a Java enum, that groups related counters. The syntax for defining a Custom Counter is:-

*enum MyCounter {*

*MISSING,*

*TOTAL*

*}*

where the counter MISSING is used to maintain the count of the missing records and the counter TOTAL is used to maintain the count of the total number of records . MyCounter represents the group of these two counters. A user can define any number of enums to group related counters.

***Note***:- Counters are global. The counter values are shared by all the map and reduce tasks across the MapReduce framework and aggregated at the end of the job across all the tasks.

**Using Counters**

Usage of a counter is simple. Once a counter is defined, it can be incremented.

*public void reduce(Text key, Iterable values, Context context)****throws****IOException,*  
 *InterruptedException {*

*if(missing)       //if missing condition*

*context.getCounter(MyCounter.MISSING).increment(1);*

*}*