Amazon Distinct Reviews Counter

(Numerical Summarization Pattern)

The counting with counters is a numerical summarization pattern where each attribute's number of occurrences can be counted. In this analysis, the counters are used to count the number of ratings appeared in the code from 1 – 5.

OUTPUT:

./hadoop jar /home/sayali/Desktop/AmazonDistinctReviewsCounter.jar sayali.AmazonReviews3.AmazonDistinctReviewsCounter /AmazonReviews/AmazonReviews.tsv /AmazonDistinctReviews

```
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        State
                1=8439
                2=5046
                3=7311
                4=13186
                5=56018
        File Input Format Counters
                Bytes Read=46360248
        File Output Format Counters
                Bytes Written=0
Rating 1 is given to 8439 products
Rating 2 is given to 5046 products
Rating 3 is given to 7311 products
Rating 4 is given to 13186 products
Rating 5 is given to 56018 products
```

Mapper and Driver Code

Mapper :

```
public class AmazonDistinctReviewsCounterMapper extends Mapper<Object, Text, NullWritable,
NullWritable> {
	public static final String State_COUNTER_GROUP = "State";
	public static final String UNKNOWN_COUNTER = "Unknown";
	public static final String NULL_OR_EMPTY_COUNTER = "Null or empty";

	private String[] StatesArray = new String[] { "1", "2", "3", "4", "5" };

	private HashSet<String> States = new HashSet<String>
	(Arrays.asList(StatesArray));

	public void map(Object key, Text value, Context context) throws IOException,
InterruptedException {
		//Map<String, String> parsed = MRDPUtills.transformXmlToMap(value.toString());
		String[] token = value.toString().split("\\t");
		String StateName = token[7].trim();

		if (StateName != null && !StateName.isEmpty()) {
			String[] StateTokens = StateName.toUpperCase().split("\\s");

			boolean unknown = true;
			for (String State : StateTokens) {
				if (State.contains(State)) {
				context.getCounter(State_COUNTER_GROUP, State).increment(1);
				unknown = false;
				break;
				}
			}
if (unknown) { context.getCounter(State_COUNTER_GROUP, UNKNOWN_COUNTER).increment(1);
			}
		} else {
	context.getCounter(State_COUNTER_GROUP, NULL_OR_EMPTY_COUNTER).increment(1);
		}
	}
}
```

Driver :

```
public class AmazonDistinctReviewsCounter {
	public static void main(String[] args) throws IOException, InterruptedException,
ClassNotFoundException {

		Configuration conf = new Configuration();
		Job job = Job.getInstance(conf, "Counting With Counters");
```

```
                job.setJarByClass(AmazonDistinctReviewsCounterMapper.class);
                job.setMapperClass(AmazonDistinctReviewsCounterMapper.class);
                job.setMapOutputKeyClass(Text.class);
                job.setMapOutputValueClass(Text.class);

                FileInputFormat.addInputPath(job, new Path(args[0]));
                FileOutputFormat.setOutputPath(job, new Path(args[1]));

                Path out = new Path(args[1]);
                FileSystem.get(conf).delete(out, true);

                int code = job.waitForCompletion(true) ? 0 : 1;

                if (code == 0) {
                        for (Counter counter :
job.getCounters().getGroup(AmazonDistinctReviewsCounterMapper.State_COUNTER_GROUP)) {
                                {
System.out.println( "Rating " + counter.getDisplayName() + " is given to " + counter.getValue() + "
products.");  }
                        }
                }
                System.exit(code);
        }
}
```