Podcasts Analytics via MapReduce[*]

Montasser AKERMI

Last Update: March, 2025

[*]The course material is hosted at `https://akermi.org/`.

Learning objectives

A podcast company hosts a platform where users can listen to various podcasts. The data were collected as shown below. Write a MapReduce application that measures:

- The number of unique listeners of each podcast.
- The number of times each podcast was skipped.
- The total number of times each podcast was listened to.

Input data

UserId;PodcastName;Skip

```
1;Foo;False
2;Bar;True
3;Baz;False
2;Foo;True
1;Foo;True
1;Baz;False
2;Bar;False
1;Baz;False
1;Foo;True
```

Output data

PodcastName;UniqueListeners;TimesSkipped;TotalListened

```
Foo;2;3;4
Bar;1;1;2
Baz;2;0;3
```

Solution

Listing 1: Mapper class

```java
import java.io.IOException;

import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class PodcastMapper extends Mapper<Object, Text, Text, Text> {
  private static final int USER_ID = 0;
  private static final int PODCAST_NAME = 1;
  private static final int SKIP = 2;

  private Text userId = new Text();
  private Text podcastName = new Text();
  private Text skip = new Text();

  public void map(Object key, Text value, Context context) throws IOException, Int
    String[] data = value.toString().split(";");
    userId.set(data[USER_ID]);
    podcastName.set(data[PODCAST_NAME]);
    skip.set(data[SKIP]);

    if (data.length == 3) {
      context.write(podcastName, new Text(userId.toString() + ";" + skip.toString(
    } else {
      context.getCounter(COUNTERS.INVALID_RECORD_COUNT).increment(1L);
    }
  }
}
```

Listing 2: Reducer class

```java
import java.io.IOException;
import java.util.HashSet;
import java.util.Set;

import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

public class PodcastReducer extends Reducer<Text, Text, Text, Text> {
  private static final int USER_ID = 0;
  private static final int SKIP = 1;

  public void reduce(Text key, Iterable<Text> value, Context context)
      throws IOException, InterruptedException {
    int total = 0;
    int skipped = 0;
    Set<Integer> userSet = new HashSet<Integer>();

    for (Text record : value) {
      String[] data = record.toString().split(";");
      total++;
      userSet.add(Integer.parseInt(data[USER_ID]));
      if (Boolean.parseBoolean(data[SKIP])) {
        skipped++;
      }
    }
    context.write(key, new Text(userSet.size() + ";" + skipped + ";" + total));
  }
}
```

Listing 3: PodcastAnalytics.java

```java
import java.io.IOException;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class PodcastAnalytics {

  public static void main(String[] args)
```

```java
      throws IOException, ClassNotFoundException, InterruptedException {
    Configuration conf = new Configuration();
    conf.set("mapred.textoutputformat.separator", ";");
    Job job = Job.getInstance(conf, "Podcasts Analytics");
    job.setJarByClass(PodcastAnalytics.class);
    job.setMapperClass(PodcastMapper.class);
    job.setReducerClass(PodcastReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(Text.class);
    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
  }
}
```

Listing 4: COUNTERS.java

```java
public enum COUNTERS {
  INVALID_RECORD_COUNT
}
```

Note: It is highly recommended to manually write the code instead of copy/pasting.