MapReduce Examples

Main Parts of MapReduce Job

- add a mapper class that extends hadoop mapper
- add reducer class that extends hadoop reducer
- in the job configuration need to define the following:
  - mapper to use
  - reducer to use
  - Inputformat
  - OutputFormat
  - Path to input data
  - Path to store output data

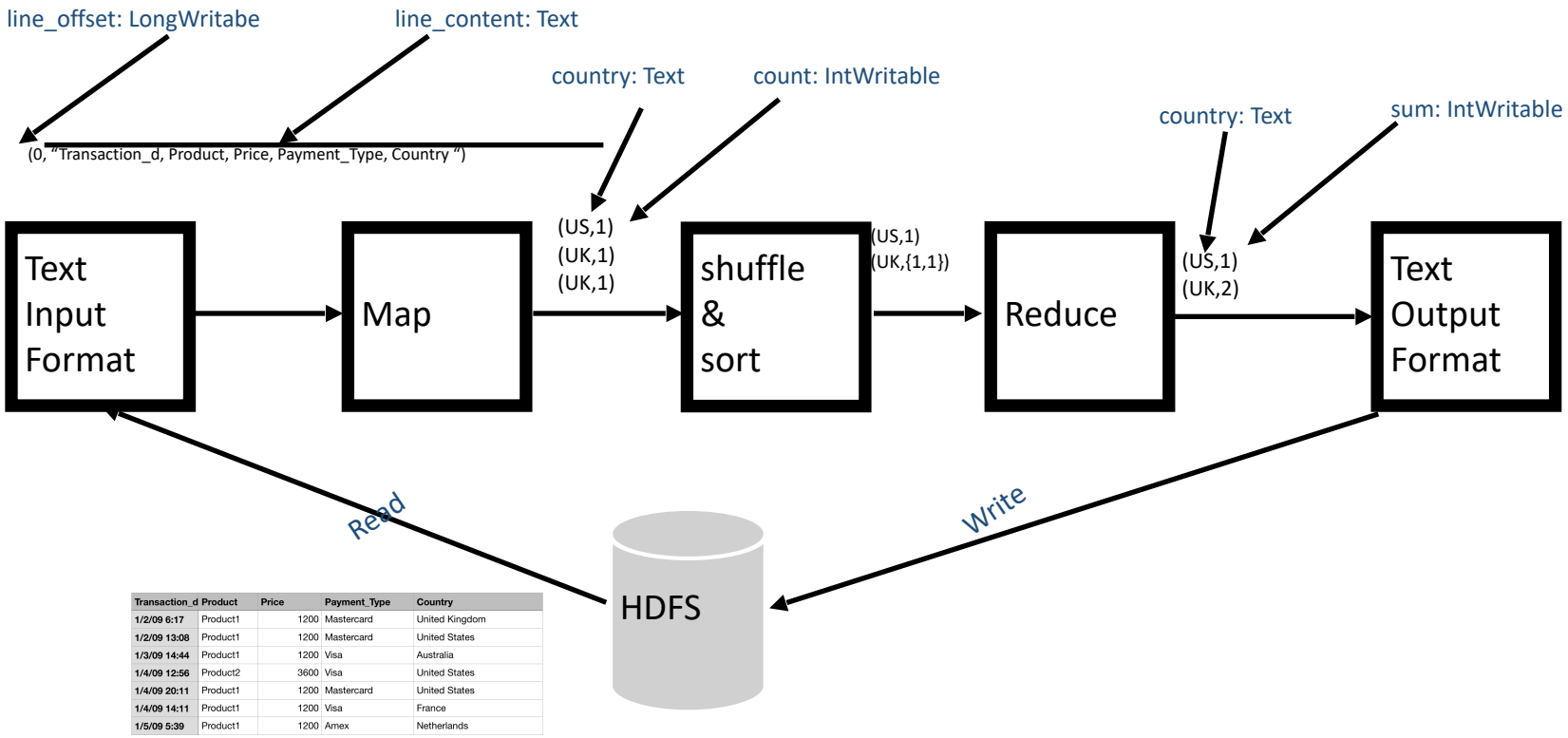
```
public class WordCount {


//Mapper
public static class MyMapper
     extends Mapper<LongWritable, Text, Text, IntWritable>{
.....
}
//Reducer
public static class MyReducer
     extends Reducer<Text, IntWritable, Text, IntWritable> {
.....
}

public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
   Job job = Job.getInstance(conf, "word count");
   job.setJarByClass(WordCount.class);
   job.setMapperClass(MyMapper.class);
   job.setCombinerClass(MyReducer.class);
   job.setReducerClass(MyReducer.class);
   job.setOutputKeyClass(Text.class);
   job.setOutputValueClass(IntWritable.class);
   job.setInputFormatClass(TextInputFormat.class)
   job.setOutputFormatClass(TextOutputFormat.class)
   FileInputFormat.addInputPath(job, new Path(args[0]));
   FileOutputFormat.setOutputPath(job, new Path(args[1]));
   System.exit(job.waitForCompletion(true) ? 0 : 1);
  }
}
```

Example

- Suppose we have a huge amount of data from an online shopping website
- this data is stored in txt files
  - it contains data as shown in the table (comma separated)
  - we want to write a code for generating statistics about total number of products sold by country

Transaction_d	Product	Price	Payment_Type	Country
1/2/09 6:17	Product1	1200	Mastercard	United Kingdom
1/2/09 13:08	Product1	1200	Mastercard	United States
1/3/09 14:44	Product1	1200	Visa	Australia
1/4/09 12:56	Product2	3600	Visa	United States
1/4/09 20:11	Product1	1200	Mastercard	United States
1/4/09 14:11	Product1	1200	Visa	France
1/5/09 5:39	Product1	1200	Amex	Netherlands

Think of the flow



Transaction_d	Product	Price	Payment_Type	Country
1/2/09 6:17	Product1	1200	Mastercard	United Kingdom
1/2/09 13:08	Product1	1200	Mastercard	United States
1/3/09 14:44	Product1	1200	Visa	Australia
1/4/09 12:56	Product2	3600	Visa	United States
1/4/09 20:11	Product1	1200	Mastercard	United States
1/4/09 14:11	Product1	1200	Visa	France
1/5/09 5:39	Product1	1200	Amex	Netherlands

How can we write this code using MapReduce

- Input is text files, so we can use TextInputFormat class
  - to transform input files into key-value pairs
- A mapper class (call it SalesMapper)
  - Implement a map function that just reads key-values
    - parse the line content
    - get the country name
    - output (country,1) as key-value pair

How can we write this code using MapReduce

- A reducer class (call it SalesCountryReducer)
  - implement a reduce function that get all values of the same key
    - iterate over values
    - sum
    - generate final output (country, total number of products)
- Use TextOutputFormat class to write output to HDFS

Design the Mapper

```
public static class SalesMapper
    extends Mapper<LongWritable, Text, Text, IntWritable>{

private final static IntWritable one = new IntWritable(1);

@Override
public void map(LongWritable key, Text value, Context context
    ) {
 //value: “Transaction_d, Product, Price, Payment_Type, Country”
 String valueString = value.toString();
 String[] tokens = valueString.split(“,”);
 String country = tokens[4]
 context.write(new Text(country), one);

}
}
```

Design the Reducer

```
public static class SalesCountryReducer
    extends Reducer<Text, IntWritable, Text, IntWritable> {

  @Override
   public void reduce(Text key, Iterable<IntWritable> values,
      Context context) {
    int totalForCountry = 0;
    for (IntWritable val : values) {
     totalForCountry += val.get();
    }

    context.write(key, new IntWritable (totalForCountry));
  }
}
```

Job Config

```
public class SalesCountry {

public static void main(String[] args) throws Exception {
  Configuration conf = new Configuration();
  Job job = Job.getInstance(conf, "Sales Count");
  job.setJarByClass(SalesCountry.class);
  job.setMapperClass(SalesMapper.class);
  job.setCombinerClass(SalesCountryReducer.class);
  job.setReducerClass(SalesCountryReducer.class);
  job.setOutputKeyClass(Text.class);
  job.setOutputValueClass(IntWritable.class);
  job.setInputFormatClass(TextInputFormat.class)
  job.setOutputFormatClass(TextOutputFormat.class)


  FileInputFormat.addInputPath(job, new Path(args[0]));
  FileOutputFormat.setOutputPath(job, new Path(args[1]));
  System.exit(job.waitForCompletion(true) ? 0 : 1);
 }
//Mapper
public static class SalesMapper
    extends Mapper<LongWritable, Text, Text, IntWritable>{
.....
}
//Reducer
public static class SalesCountryReducer
    extends Reducer<Text, IntWritable, Text, IntWritable> {
.....
}
}
```

Create a job instance, give it a name, choose the main class

Configure the mapper, reduce, and combiner classes

Configure type for the output key, and output values

Configure InputFormat class (how to read the data), and OutputFormat class for how to store final output

Input/output paths

Second Example

- Based on the same data set, return for each country the sold product with quantity

Example 3

- Suppose we have a collection of Web documents
  - create a link graph (how these pages are connected)
    - for example pageA, pageB, “click her to see page B”

    - Write a MapReduce code that gives the following
    - For each page, total number of pages that has link to it
    - For example pageB,1000