MapReduce Operation Examples

CELEBRATING 30 YEARS
Marlan and Rosemary Bourns
College of Engineering

Preliminaries

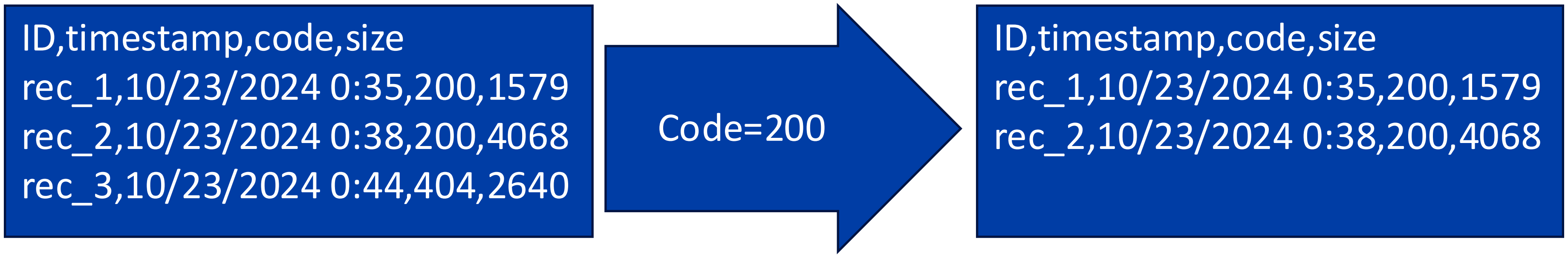
- All inputs files are in CSV format

```
ID,Name,Dept
1,Alex,CS
2,Ben,EE
3,Chu,ME
…
```

- Each line is read as a text line in the format (offset, Text)
- String#split is used to split the string around the comma
- Integer#parseInt parses a string into an integer

CELEBRATING 30 YEARS
UCR Marlan and Rosemary Bourns College of Engineering

Filter

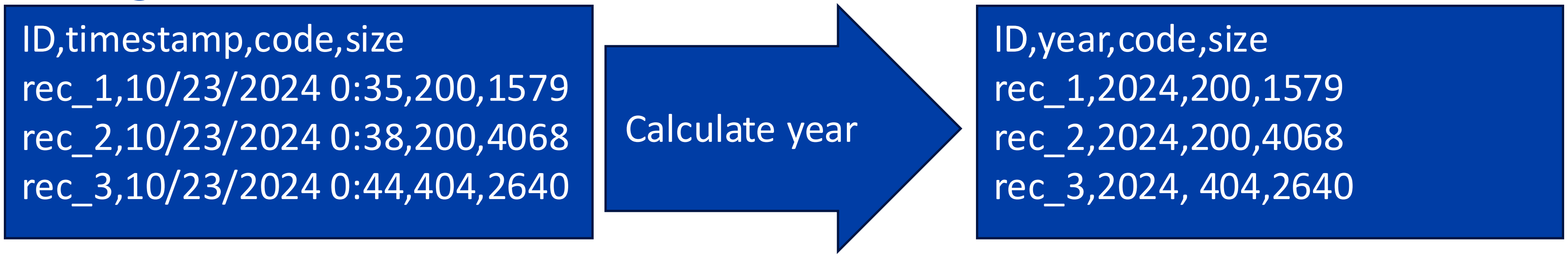


```
ID,timestamp,code,size
rec_1,10/23/2024 0:35,200,1579
rec_2,10/23/2024 0:38,200,4068
rec_3,10/23/2024 0:44,404,2640
```

```
ID,timestamp,code,size
rec_1,10/23/2024 0:35,200,1579
rec_2,10/23/2024 0:38,200,4068
```

```java
public static class FilterMapper extends Mapper<LongWritable, Text, NullWritable, Text> {
  public void map(LongWritable offset, Text line, Context context)
      throws IOException, InterruptedException {
    String[] parts = line.toString().split(",");
    int code = Integer.parseInt(parts[2]);
    if (code == 200)
      context.write(NullWritable.get(), line);
  }
}
```

Projection

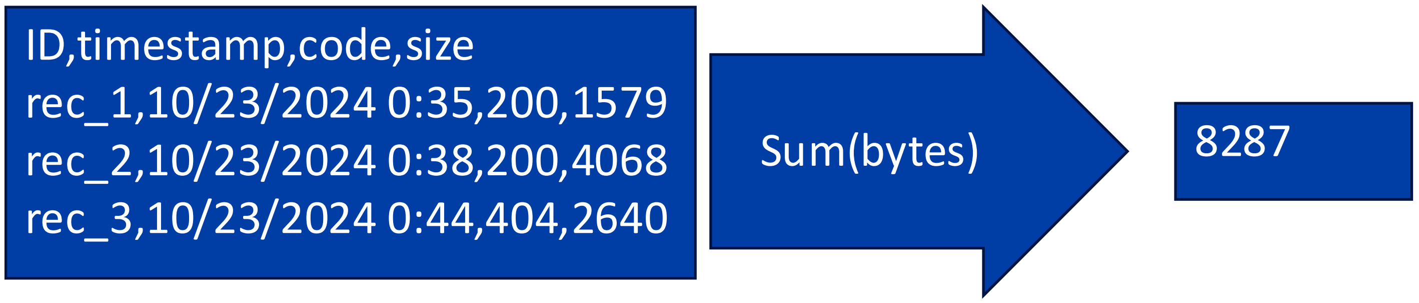


```
ID,timestamp,code,size
rec_1,10/23/2024 0:35,200,1579
rec_2,10/23/2024 0:38,200,4068
rec_3,10/23/2024 0:44,404,2640
```

```
ID,year,code,size
rec_1,2024,200,1579
rec_2,2024,200,4068
rec_3,2024, 404,2640
```

```java
public static class ProjectionMapper
    extends Mapper<LongWritable, Text, NullWritable, Text> {
  Calendar calendar = Calendar.getInstance();
  SimpleDateFormat formatter = new SimpleDateFormat("MM/dd/yyyy HH:mm");
  Text output = new Text();

  public void map(LongWritable offset, Text line, Context context)
      throws IOException, InterruptedException {
    String[] parts = line.toString().split(",");
    String dateString = parts[1];
  try {
      calendar.setTime(formatter.parse(dateString));
      parts[1] = String.valueOf(calendar.get(Calendar.YEAR));
      output.set(String.join(",", parts));
      context.write(NullWritable.get(), output);
    } catch (ParseException e) {
      // Cannot parse date string -> ignore the line
    }
  }
}
```

Aggregation



```
ID,timestamp,code,size
rec_1,10/23/2024 0:35,200,1579
rec_2,10/23/2024 0:38,200,4068
rec_3,10/23/2024 0:44,404,2640
```

```java
public static class AggregationMapper
    extends Mapper<LongWritable, Text, NullWritable, LongWritable> {
  LongWritable bytes = new LongWritable();

  public void map(LongWritable offset, Text line, Context context)
      throws IOException, InterruptedException {
    String[] parts = line.toString().split(",");
    bytes.set(Long.parseLong(parts[3]));
    context.write(NullWritable.get(), bytes);
  }
}
```

```java
public static class AggregationCombinerReducer
    extends Reducer<Object,LongWritable,Object,LongWritable> {
  private LongWritable result = new LongWritable();

  public void reduce(Object key, Iterable<LongWritable> values, Context context)
      throws IOException, InterruptedException {
    long sum = 0;
    for (LongWritable val : values)
      sum += val.get();
    result.set(sum);
    context.write(key, result);
  }
}
```

Grouped Aggregation

```
ID,timestamp,code,size
rec_1,10/23/2024 0:35,200,1579
rec_2,10/23/2024 0:38,200,4068
rec_3,10/23/2024 0:44,404,2640
```

Sum(bytes)
Group by code

```
Code,size
200,5647
404,2640
```

```java
public static class AggregationMapper
    extends Mapper<LongWritable, Text, IntWritable, LongWritable> {
  IntWritable code = new IntWritable();
  LongWritable bytes = new LongWritable();

  public void map(LongWritable offset, Text line, Context context)
      throws IOException, InterruptedException {
    String[] parts = line.toString().split(",");
    code.set(Integer.parseInt(parts[2]));
    bytes.set(Long.parseLong(parts[3]));
    context.write(code, bytes);
  }
}
```

```java
public static class AggregationCombinerReducer
    extends Reducer<Object,LongWritable,Object,LongWritable> {
  private LongWritable result = new LongWritable();

  public void reduce(Object key, Iterable<LongWritable> values, Context context)
      throws IOException, InterruptedException {
    long sum = 0;
    for (LongWritable val : values)
      sum += val.get();
    result.set(sum);
    context.write(key, result);
  }
}
```

Random Sample

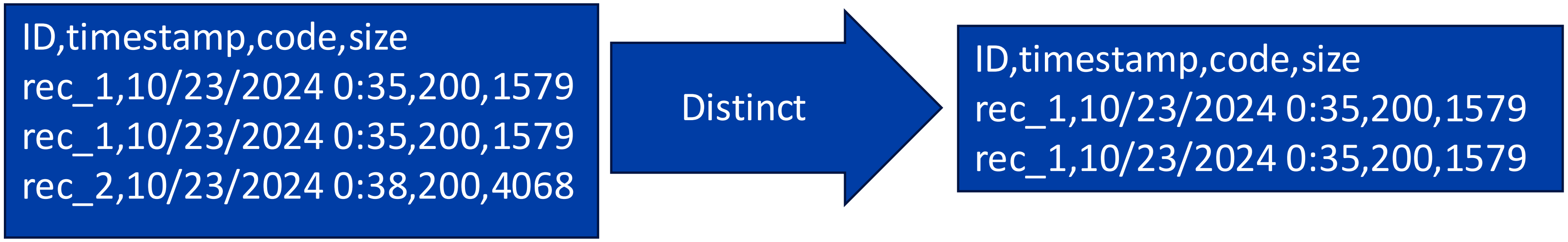
```
ID,timestamp,code,size
rec_1,10/23/2024 0:35,200,1579
rec_2,10/23/2024 0:38,200,4068
rec_3,10/23/2024 0:44,404,2640
```

sample(0.3)

```
ID,timestamp,code,size
rec_2,10/23/2024 0:38,200,4068
```

```java
public static class RandomSampleMapper
    extends Mapper<LongWritable, Text, NullWritable, Text> {

  private float sampleFraction;
  private Random random;

  @Override
  protected void setup(Context context) throws IOException, InterruptedException {
    sampleFraction = context.getConfiguration().getFloat("sample.fraction", 0.1f);
    long randomSeed = context.getConfiguration().getLong("sample.seed", 0);
    int taskId = context.getTaskAttemptID().getTaskID().getId();
    random = new Random(randomSeed + taskId);
  }

  @Override
  public void map(LongWritable key, Text value, Context context)
      throws IOException, InterruptedException {
    if (random.nextFloat() < sampleFraction)
      context.write(NullWritable.get(), value);
  }
}
```

Distinct



```
ID,timestamp,code,size
rec_1,10/23/2024 0:35,200,1579
rec_1,10/23/2024 0:35,200,1579
rec_2,10/23/2024 0:38,200,4068
```

```
ID,timestamp,code,size
rec_1,10/23/2024 0:35,200,1579
rec_1,10/23/2024 0:35,200,1579
```

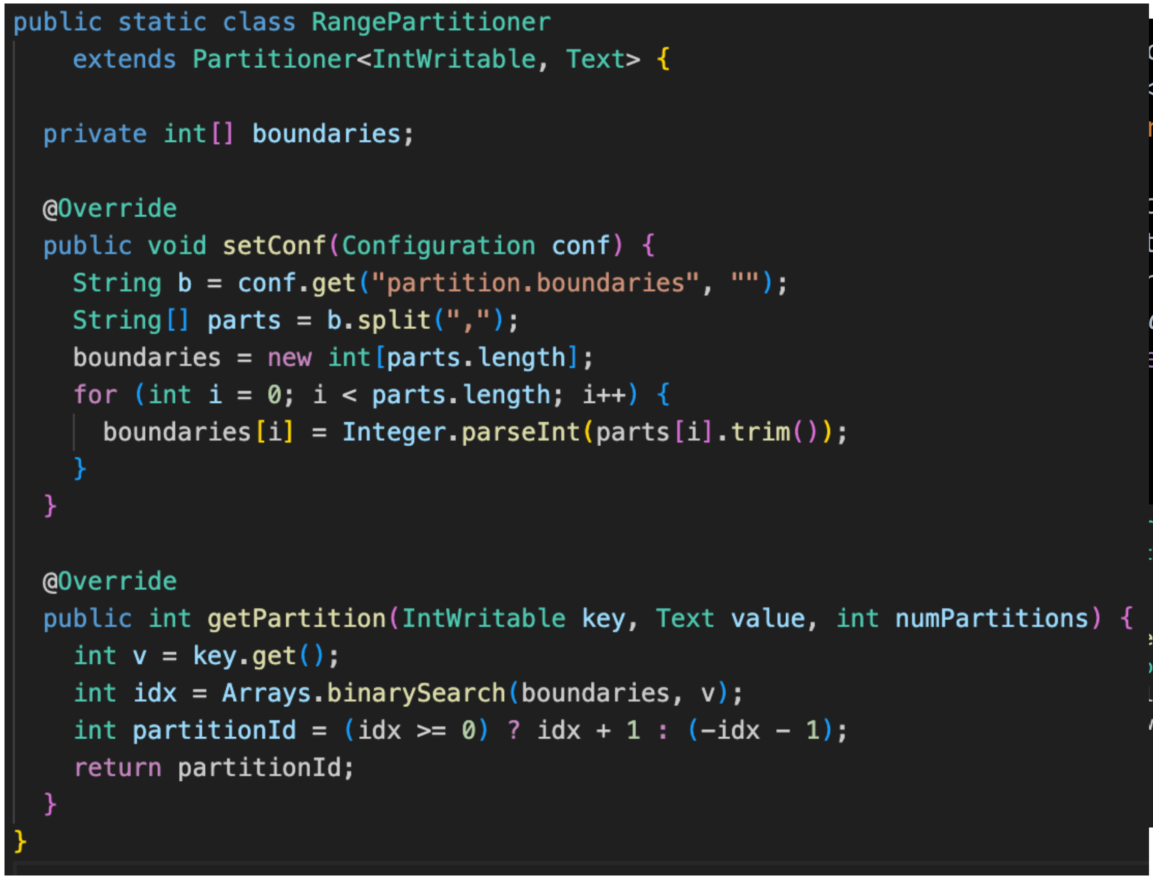
```java
public static class DistinctMapper
    extends Mapper<LongWritable, Text, Text, NullWritable> {

  @Override
  protected void map(LongWritable offset, Text line, Context context)
      throws IOException, InterruptedException {
    context.write(line, NullWritable.get());
  }
}

public static class DistinctReducer
    extends Reducer<Text, NullWritable, Text, NullWritable> {

  @Override
  protected void reduce(Text key, Iterable<NullWritable> values, Context context)
      throws IOException, InterruptedException {
    context.write(key, NullWritable.get());
  }
}
```

Sort

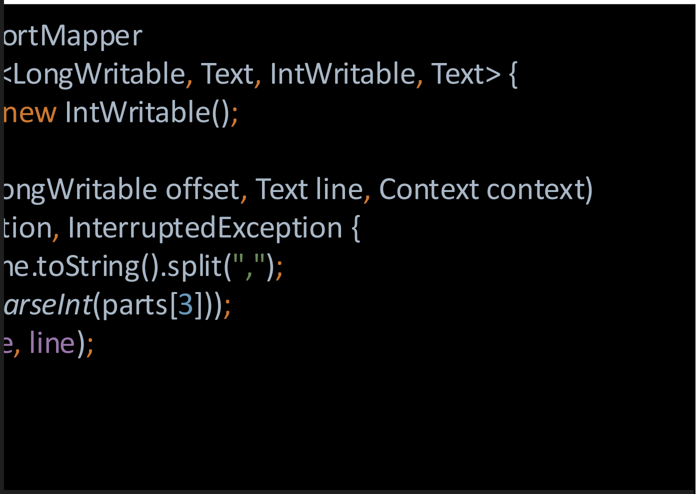
```
ID,timestamp,code,size
rec_1,10/23/2024 0:35,200,1579
rec_2,10/23/2024 0:38,200,4068
rec_3,10/23/2024 0:44,404,2640
```

Sort by Size

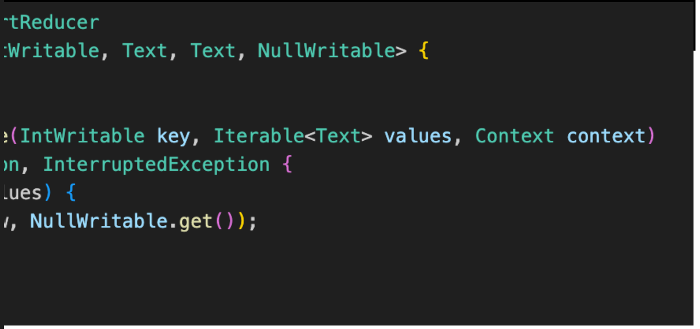
```
ID,timestamp,code,size
rec_1,10/23/2024 0:35,200,1579
rec_3,10/23/2024 0:44,404,2640
rec_2,10/23/2024 0:38,200,4068
```



```java
public static class RangePartitioner
    extends Partitioner<IntWritable, Text> {

  private int[] boundaries;

  @Override
  public void setConf(Configuration conf) {
    String b = conf.get("partition.boundaries", "");
    String[] parts = b.split(",");
    boundaries = new int[parts.length];
    for (int i = 0; i < parts.length; i++) {
      boundaries[i] = Integer.parseInt(parts[i].trim());
    }
  }

  @Override
  public int getPartition(IntWritable key, Text value, int numPartitions) {
    int v = key.get();
    int idx = Arrays.binarySearch(boundaries, v);
    int partitionId = (idx >= 0) ? idx + 1 : (-idx - 1);
    return partitionId;
  }
}
```



```
ortMapper
<LongWritable, Text, IntWritable, Text> {
new IntWritable();

ongWritable offset, Text line, Context context)
tion, InterruptedException {
ne.toString().split(",");
arseInt(parts[3]));
e, line);
```



```
tReducer
Writable, Text, Text, NullWritable> {

(IntWritable key, Iterable<Text> values, Context context)
on, InterruptedException {
lues) {
, NullWritable.get());
```