MAP REDUCE CODES

EXAMPLE 1: ITEM WISE COUNT

```
package com.madhukaraphatak.hadooptraining.mrexamples.itemwisecount;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;
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;

import java.io.IOException;


/**
 * Created by madhu on 7/1/15.
 */
public class ItemCountDriver {
    public static void main(String args[]) throws IOException, ClassNotFoundException,
InterruptedException {

        Configuration configuration = new Configuration();
        Job job = Job.getInstance(configuration);
        job.setJarByClass(ItemCountDriver.class);
        job.setMapperClass(ItemCountMapper.class);
        job.setReducerClass(ItemCountReducer.class);
        job.setMapOutputKeyClass(Text.class);
        job.setMapOutputValueClass(LongWritable.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(LongWritable.class);

        //add information about input / output directories

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

        job.waitForCompletion(true);

    }
}

package com.madhukaraphatak.hadooptraining.mrexamples.itemwisecount;

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

import java.io.IOException;
```

```
/**
 * Created by madhu on 7/1/15.
 */
public class ItemCountMapper extends Mapper<LongWritable,Text,Text,LongWritable> {
   LongWritable one = new LongWritable(1);
   Text keyText = new Text();
   @Override
   protected void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {
       String [] values = value.toString().split(",");
       String transactionId = values[0];
       String customerId = values[1];
       String itemId = values[2];
       double itemValue = Double.parseDouble(values[3]);
       keyText.set(itemId);
       context.write(keyText,one);
   }
}


package com.madhukaraphatak.hadooptraining.mrexamples.itemwisecount;

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

import java.io.IOException;

/**
 * Created by madhu on 7/1/15.
 */
public class ItemCountReducer extends Reducer<Text,LongWritable,Text,LongWritable> {
   @Override
   protected void reduce(Text key, Iterable<LongWritable> values, Context context) throws
IOException, InterruptedException {
       long sum = 0;
       for (LongWritable value : values) {
         sum+= value.get();
       }
       context.write(key,new LongWritable(sum));
   }
}
```

EXAMPLE 2: ITEM FILTERING

```
package com.madhukaraphatak.hadooptraining.mrexamples.itemfiltering;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;
```

```
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;

import java.io.IOException;


/**
 * Created by madhu on 7/1/15.
 */
public class ItemFilterDriver {
    public static void main(String args[]) throws IOException, ClassNotFoundException,
InterruptedException {

        Configuration configuration = new Configuration();
        configuration.set("item_id_regex",args[1]);
        Job job = Job.getInstance(configuration);
        job.setJarByClass(ItemFilterDriver.class);
        job.setMapperClass(ItemFilterMapper.class);
        job.setNumReduceTasks(0);
        job.setMapOutputKeyClass(LongWritable.class);
        job.setMapOutputValueClass(Text.class);
        job.setOutputKeyClass(LongWritable.class);
        job.setOutputValueClass(Text.class);
        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[2]));
        job.waitForCompletion(true);

    }
}

package com.madhukaraphatak.hadooptraining.mrexamples.itemfiltering;

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

import java.io.IOException;


/**
 * Created by madhu on 7/1/15.
 */
public class ItemFilterMapper extends Mapper<LongWritable,Text,LongWritable,Text> {

    static String regex;
    @Override
    protected void setup(Context context) throws IOException, InterruptedException {
        super.setup(context);
        regex = context.getConfiguration().get("item_id_regex");
    }
```

```
    LongWritable one = new LongWritable(1);
    Text keyText = new Text();
    @Override
    protected void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {
        String [] values = value.toString().split(",");
        String transactionId = values[0];
        String customerId = values[1];
        String itemId = values[2];
        double itemValue = Double.parseDouble(values[3]);
        if(itemId.contains(regex)){
          context.write(key,value);
        }
    }
}
```

EXAMPLE 3: CUSTOM WRITABLE

```
package com.madhukaraphatak.hadooptraining.mrexamples.customwritable;


import org.apache.hadoop.io.WritableComparable;

import java.io.DataInput;
import java.io.DataOutput;
import java.io.IOException;

/**
 * Created by madhu on 7/1/15.
 */
public class SalesRecordWritable implements WritableComparable<SalesRecordWritable> {
    private String transactionId;
    private String customerId;
    private String itemId;
    private double itemValue;

    public SalesRecordWritable(String transactionId, String customerId, String itemId, double
itemValue) {
        this.transactionId = transactionId;
        this.customerId = customerId;
        this.itemId = itemId;
        this.itemValue = itemValue;
    }

    public SalesRecordWritable() {
    }

    public String getTransactionId() {
        return transactionId;
    }
```

```
    public void setTransactionId(String transactionId) {
        this.transactionId = transactionId;
    }

    public String getCustomerId() {
        return customerId;
    }

    public void setCustomerId(String customerId) {
        this.customerId = customerId;
    }

    public String getItemId() {
        return itemId;
    }

    public void setItemId(String itemId) {
        this.itemId = itemId;
    }

    public double getItemValue() {
        return itemValue;
    }

    public void setItemValue(double itemValue) {
        this.itemValue = itemValue;
    }

    @Override
    public int compareTo(SalesRecordWritable o) {
        return this.transactionId.compareTo(o.transactionId);
    }

    @Override
    public void write(DataOutput out) throws IOException {
        out.writeUTF(transactionId);
        out.writeUTF(customerId);
        out.writeUTF(itemId);
        out.writeDouble(itemValue);
    }

    @Override
    public void readFields(DataInput in) throws IOException {
        transactionId = in.readUTF();
        customerId = in.readUTF();
        itemId = in.readUTF();
        itemValue = in.readDouble();
    }
}
```

```
package com.madhukaraphatak.hadooptraining.mrexamples.customwritable;

import org.apache.hadoop.io.Text;

/**
 * Created by madhu on 7/1/15.
 */
public class SalesRecordParser {
    public static SalesRecordWritable parse(Text record) throws MalformedRecordException{
        String [] values = record.toString().split(",");
        if(values.length <4 ) throw  new MalformedRecordException();
        else {
            String transactionId = values[0];
            String customerId = values[1];
            String itemId = values[2];
            double itemValue = Double.parseDouble(values[3]);
            return new SalesRecordWritable(transactionId,customerId,itemId,itemValue);
        }
    }
}
```

```
package com.madhukaraphatak.hadooptraining.mrexamples.customwritable;

/**
 * Created by madhu on 7/1/15.
 */
public class MalformedRecordException extends Exception {
}
```

EXAMPLE 4: ITEM WISE DISCOUNT – CHAIN MAPPER EXAMPLE

```
package com.madhukaraphatak.hadooptraining.mrexamples.discount;

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

import java.io.IOException;


/**
```

```
 * Created by madhu on 7/1/15.
 */
public class SalesMMRDriver {
   public static void main(String args[]) throws IOException, ClassNotFoundException,
InterruptedException {
       Configuration configuration = new Configuration();
       Job job = Job.getInstance(configuration);
       job.setJarByClass(SalesMMRDriver.class);
       ChainMapper.addMapper(job, SalesRecordMapper.class, LongWritable.class, Text.class,
Text.class, DoubleWritable.class, configuration);
       ChainMapper.addMapper(job, ItemDiscountMapper.class, Text.class, DoubleWritable.class,
Text.class, DoubleWritable.class, configuration);
       job.setCombinerClass(DoubleReducer.class);
       job.setReducerClass(DoubleReducer.class);
       job.setOutputKeyClass(Text.class);
       job.setOutputValueClass(DoubleWritable.class);
       FileInputFormat.addInputPath(job, new Path(args[0]));
       FileOutputFormat.setOutputPath(job, new Path(args[1]));
       job.waitForCompletion(true);
   }
}


package com.madhukaraphatak.hadooptraining.mrexamples.discount;

import com.madhukaraphatak.hadooptraining.mrexamples.customwritable.SalesRecordParser;
import com.madhukaraphatak.hadooptraining.mrexamples.customwritable.SalesRecordWritable;
import
com.madhukaraphatak.hadooptraining.mrexamples.customwritable.MalformedRecordException;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

import java.io.IOException;

/**
 * Created by madhu on 7/1/15.
 */
public class SalesRecordMapper extends Mapper<LongWritable,Text,Text,DoubleWritable>{
   @Override
   protected void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {
       try {
          SalesRecordWritable salesRecordWritable = SalesRecordParser.parse(value);
          context.write(new Text(salesRecordWritable.getCustomerId()),
                new DoubleWritable(salesRecordWritable.getItemValue()));
       } catch (MalformedRecordException e) {
          e.printStackTrace();
       }

   }
```

```
}


package com.madhukaraphatak.hadooptraining.mrexamples.discount;

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

import java.io.IOException;


/**
 * Created by madhu on 7/1/15.
 */
public class ItemDiscountMapper<K> extends Mapper<K,DoubleWritable,K,DoubleWritable> {
   @Override
   protected void map(K key, DoubleWritable value, Context context) throws IOException,
InterruptedException {
       // 5 percent discount
       double afterDiscount = value.get() - (value.get() * 5)/100.0;
       context.write(key,new DoubleWritable(afterDiscount));
   }

}

package com.madhukaraphatak.hadooptraining.mrexamples.discount;

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

import java.io.IOException;

/**
 * Created by madhu on 7/1/15.
 */
public class DoubleReducer<K> extends Reducer<K,DoubleWritable,K,DoubleWritable> {
   @Override
   protected void reduce(K key, Iterable<DoubleWritable> values, Context context) throws
IOException, InterruptedException {
       double sum = 0;
       for (DoubleWritable value : values) {
         sum+= value.get();
       }
       context.write(key,new DoubleWritable(sum));
   }
}
```

EXAMPLE 5: TOTAL ITEM COST DISCOUNT – CHAIN REDUCER EXAMPLE

```
package com.madhukaraphatak.hadooptraining.mrexamples.discount;

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


import java.io.IOException;


/**
 * Created by madhu on 7/1/15.
 */
public class SalesMRMDriver {
   public static void main(String args[]) throws IOException, ClassNotFoundException,
InterruptedException {
       Configuration configuration = new Configuration();
       Job job = Job.getInstance(configuration);
       job.setJarByClass(SalesMRMDriver.class);
       job.setMapperClass(SalesRecordMapper.class);

ChainReducer.setReducer(job,DoubleReducer.class,Text.class,DoubleWritable.class,Text.class,
               DoubleWritable.class,configuration);

ChainReducer.addMapper(job,SumDiscountMapper.class,Text.class,DoubleWritable.class,Text.cla
ss,
               DoubleWritable.class,configuration);
       job.setOutputKeyClass(Text.class);
       job.setOutputValueClass(DoubleWritable.class);
       FileInputFormat.addInputPath(job, new Path(args[0]));
       FileOutputFormat.setOutputPath(job, new Path(args[1]));
       job.waitForCompletion(true);
   }
}


package com.madhukaraphatak.hadooptraining.mrexamples.discount;

import com.madhukaraphatak.hadooptraining.mrexamples.customwritable.SalesRecordParser;
import com.madhukaraphatak.hadooptraining.mrexamples.customwritable.SalesRecordWritable;
```

```
import
com.madhukaraphatak.hadooptraining.mrexamples.customwritable.MalformedRecordException;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

import java.io.IOException;

/**
 * Created by madhu on 7/1/15.
 */
public class SalesRecordMapper extends Mapper<LongWritable,Text,Text,DoubleWritable>{
   @Override
   protected void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {
       try {
          SalesRecordWritable salesRecordWritable = SalesRecordParser.parse(value);
          context.write(new Text(salesRecordWritable.getCustomerId()),
                new DoubleWritable(salesRecordWritable.getItemValue()));
       } catch (MalformedRecordException e) {
          e.printStackTrace();
       }

   }
}


package com.madhukaraphatak.hadooptraining.mrexamples.discount;

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

import java.io.IOException;

/**
 * Created by madhu on 7/1/15.
 */
public class DoubleReducer<K> extends Reducer<K,DoubleWritable,K,DoubleWritable> {
   @Override
   protected void reduce(K key, Iterable<DoubleWritable> values, Context context) throws
IOException, InterruptedException {
       double sum = 0;
       for (DoubleWritable value : values) {
         sum+= value.get();
       }
       context.write(key,new DoubleWritable(sum));
   }
}


/**
```

```
 * Created by madhu on 7/1/15.
 */


package com.madhukaraphatak.hadooptraining.mrexamples.discount;

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

import java.io.IOException;

/**
 * Created by madhu on 7/1/15.
 */
public class SumDiscountMapper<K> extends Mapper<K,DoubleWritable,K,DoubleWritable> {
    @Override
    protected void map(K key, DoubleWritable value, Context context) throws IOException,
InterruptedException {
        // 10 percent discount if more than 1000
        if(value.get() >500) {
            double afterDiscount = value.get() - (value.get() * 10) / 100.0;
            context.write(key, new DoubleWritable(afterDiscount));
        } else context.write(key,value);
    }
}
```