db.runCommand(

{

mapReduce: <collection>,

map: <function>,

reduce: <function>,

out: <output>,

}

)

Db.mycollection.mapReduce(

mapFunction,

reduceFunction,

out: “mr-out1”

)

Map is called 1st, then reducer and then finalizer

When we don’t return one value ie we return array, we use finalizer

For /r %%i in (\*) do echo %%i

Pause

Replication

Data1

Data2

Data3

Mongodb1

Mongodb2

Mongodb3

Mapreduce in Hadoop:

1. Mapper Class
2. Reducer Class
3. Create a Job in Main Method (Driver Class)
4. Create a Jar file
5. Run the jar file as follows:

Hadoop jar my.jar mainclass inputDir outDir

Note:

Class User{

}

Main(){

User user – Driver class;

}

1. Define InputFormat
2. Create a Mapper
3. Define OutputFormat
4. Define a Reducer

Steps:

**Step 1:** Create MAPPER class

Class WordMapper extends Mapper<KeyIn, ValueIn, Text, IntWritable> {

//Override map method

}

**Step 2:** Create REDUCER class

Class WordReducer extends Reducer< Text, IntWritable, KeyOut, ValueOut>{

//override reduce method

}

**Note:**

In Step1 , TextInputFormat<LongWritable,Text>

Mapper< LongWritable, Text, Text, IntWritable >

In Step 2, TextOutputFormat<Text,IntWritable>

Reducer< Text, IntWritable, Text,IntWritable >

**Step 3:**

Main(){

Job job = ..;

Job.setMapperClass();

Job.setReducerClass();

Job.setNumReduceTask();

Job.setIntputFormat(TextInputFormat);

Job.setoutputFormat(TextOutputFormat);

}

**Step 4:**

My.jar

**Step 5:**

Hadoop jar my.jar mainClass /inputDir/OutDir