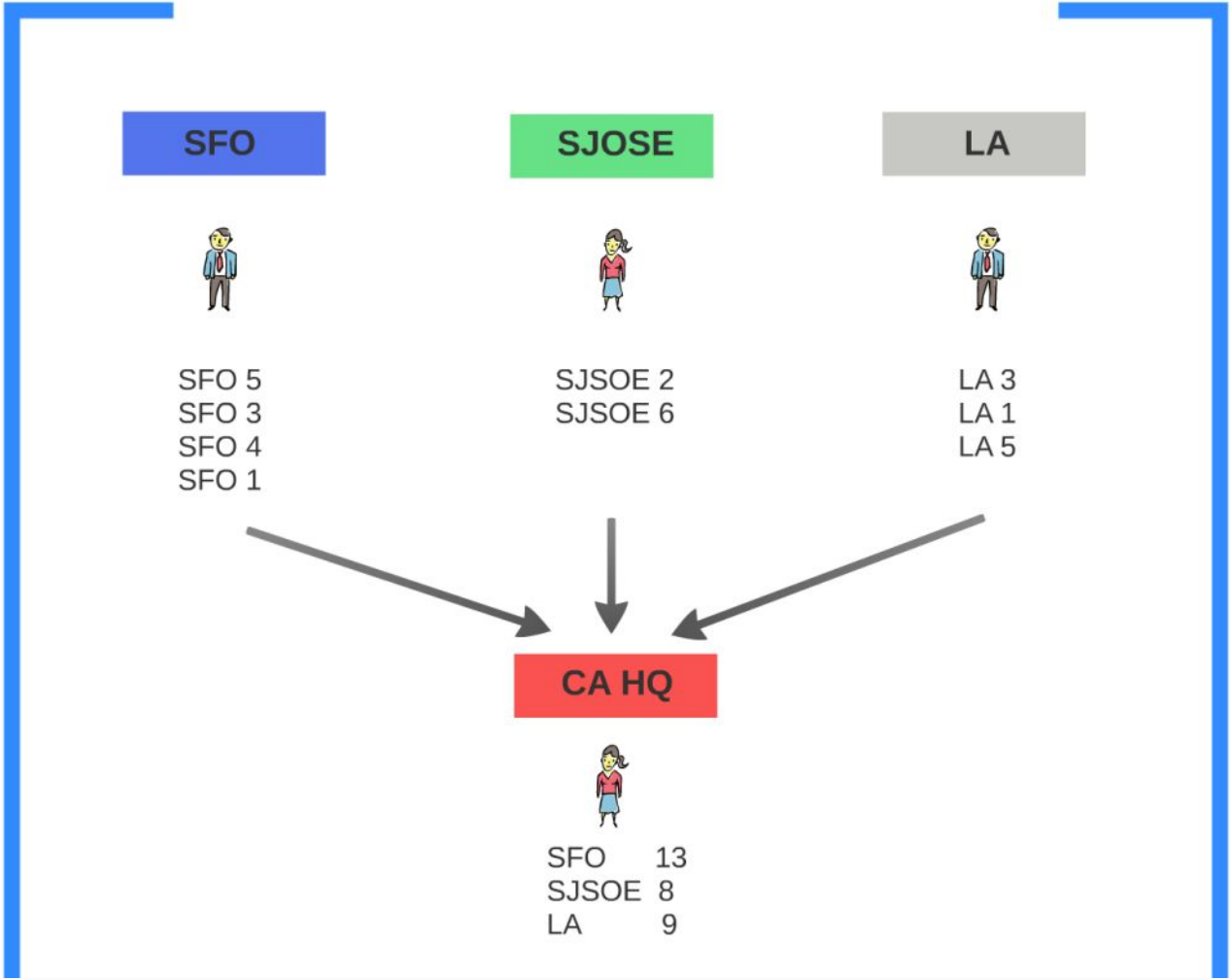
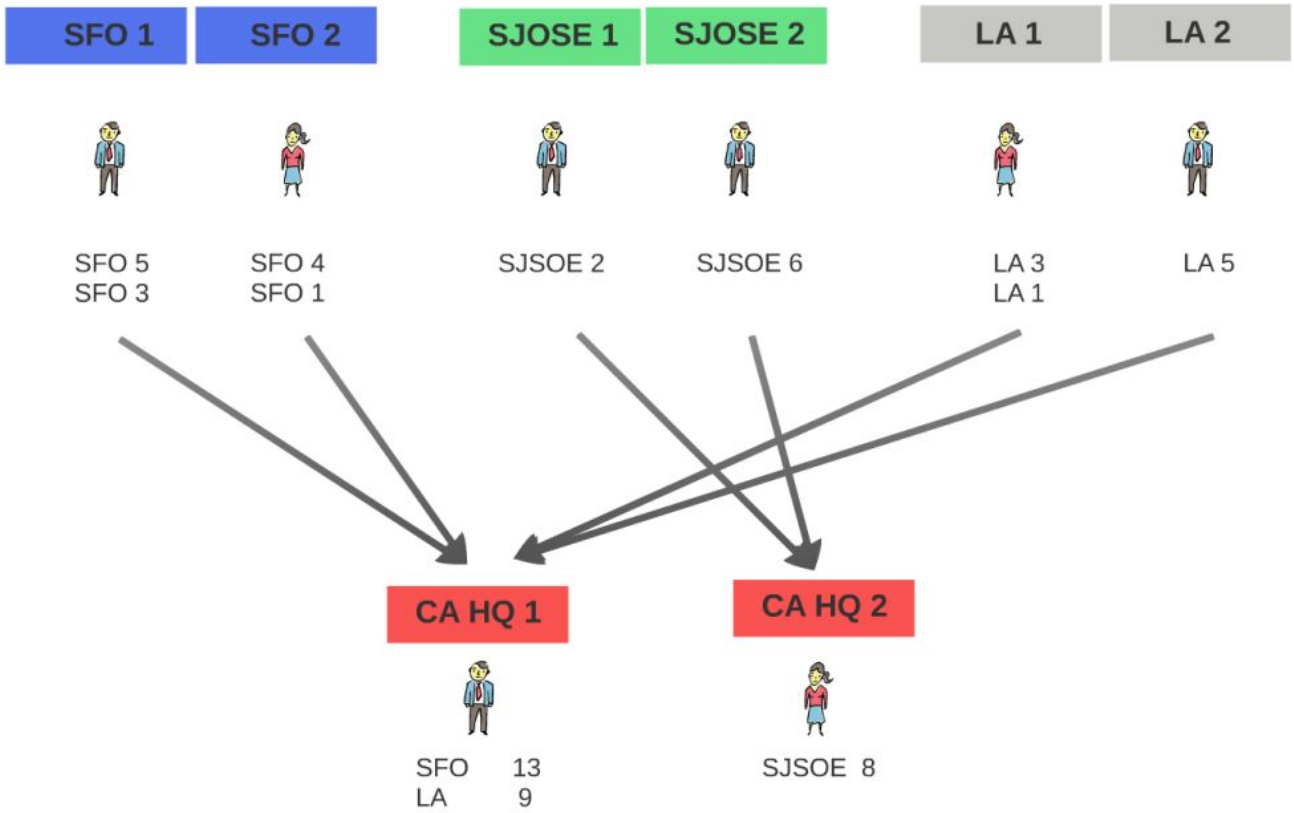
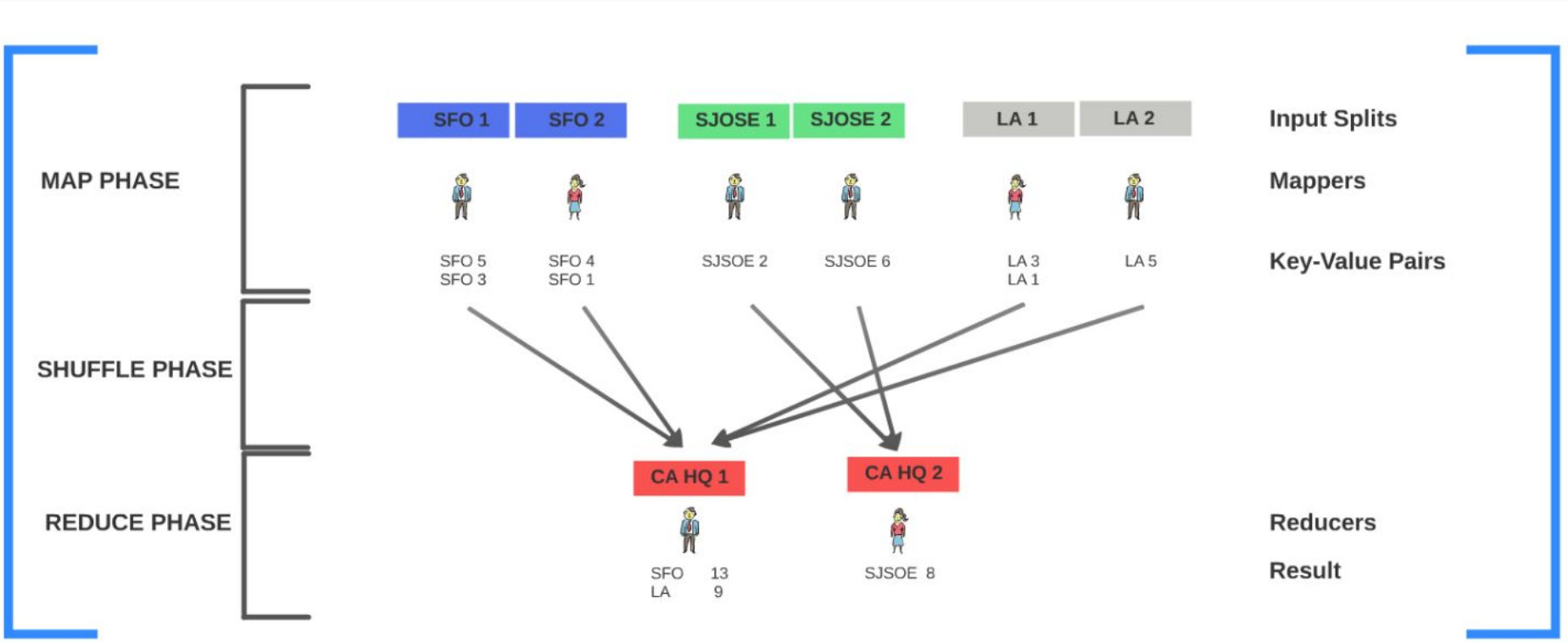
MAP REDUCE

MAPREDUCE


SFO
SJOSE
LA
SFO 5
SFO 3
SFO 4
SFO 1
SJSOE 2
SJSOE 6
LA 3
LA 1
LA 5
CA HQ
SFO 13
SJSOE 8
LA 9


SFO 1
SFO 2
SJOSE 1
SJOSE 2
LA 1
LA 2
SFO 5
SFO 3
SFO 4
SFO 1
SJSOE 2
SJSOE 6
LA 3
LA 1
LA 5
CA HQ 1
CA HQ 2
SFO 13
LA 9
SJSOE 8


MAP PHASE
SHUFFLE PHASE
REDUCE PHASE
SFO 1
SFO 2
SJOSE 1
SJOSE 2
LA 1
LA 2
Input Splits
Mappers
Key-Value Pairs
SFO 5
SFO 3
SFO 4
SFO 1
SJSOE 2
SJSOE 6
LA 3
LA 1
LA 5
CA HQ 1
CA HQ 2
Reducers
SFO 13
LA 9
SJSOE 8
Result

WHAT IS MAPREDUCE?

- Distributed Programming model for processing large data sets
- Conceived at Google
- Can be implemented in any programming language
- MapReduce is NOT a programming language
- Hadoop implements MapReduce
- MapReduce System (Hadoop) - Manage communications, data transfers, parallel execution across distributed servers

DISSECTING MAPREDUCE COMPONENTS

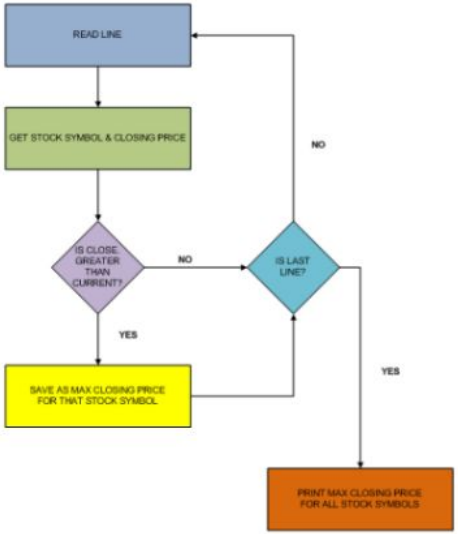
SAMPLE BIG DATA PROBLEM

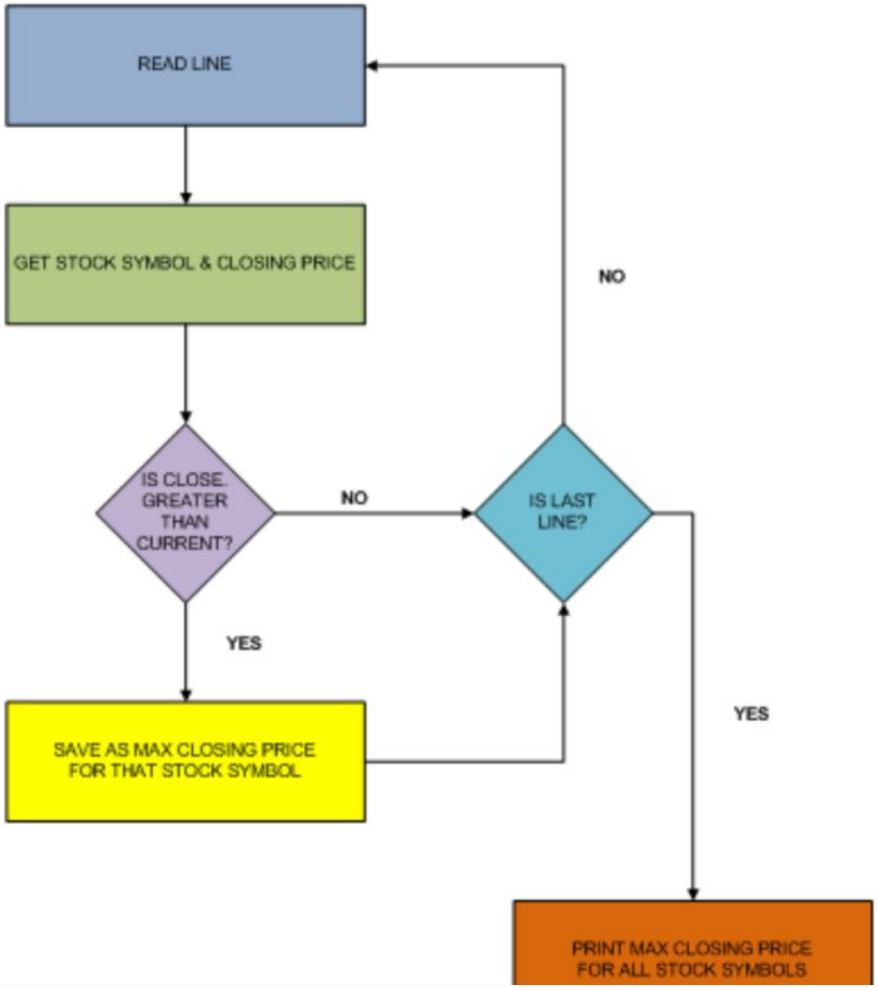
- Sample Stocks Dataset
- Each record has symbol, date, open, close...
- Find Maximum Closing Price for each symbol

```
ABCSE,B7J,2008-10-28,6.48,6.74,6.22,6.72,44300,5.79
ABCSE,B7J,2008-10-27,6.21,6.78,6.21,6.40,55200,5.51
ABCSE,B7J,2008-10-24,6.39,6.66,6.21,6.40,67400,5.51
ABCSE,B7J,2008-10-23,6.95,6.95,6.50,6.59,59400,5.68
ABCSE,B7J,2008-10-22,6.92,7.17,6.80,6.80,55300,5.86
ABCSE,B7J,2008-10-21,7.20,7.30,7.10,7.10,54400,6.11
ABCSE,B7J,2008-10-20,6.94,7.31,6.94,7.12,45700,6.13
ABCSE,B7J,2008-10-17,6.43,6.93,6.42,6.90,57700,5.94
ABCSE,B7J,2008-10-16,6.61,6.69,6.21,6.53,83200,5.62
ABCSE,B7J,2008-10-15,6.84,6.90,6.36,6.36,78900,5.48
ABCSE,B7J,2008-10-14,7.15,7.32,6.93,6.96,74700,5.99
ABCSE,B7J,2008-10-13,6.00,6.57,6.00,6.57,75700,5.66
ABCSE,B7J,2008-10-10,5.05,5.72,4.79,5.72,158400,4.93
ABCSE,B7J,2008-10-09,6.30,6.41,6.00,6.02,140500,5.18
ABCSE,B7J,2008-10-08,5.60,6.47,5.60,6.28,292000,5.41
ABCSE,B7J,2008-10-07,7.59,7.59,6.66,6.69,89900,5.76
ABCSE,B7J,2008-10-06,7.83,7.90,7.00,7.40,159600,6.37
```

MAX CLOSING PRICE ALGORITHM

- One Node
- Not Distributed




READ LINE
GET STOCK SYMBOL & CLOSING PRICE
IS CLOSE. GREATER THAN CURRENT?
NO
IS LAST LINE?
NO
YES
SAVE AS MAX CLOSING PRICE FOR THAT STOCK SYMBOL
YES
PRINT MAX CLOSING PRICE FOR ALL STOCK SYMBOLS

DISTRIBUTED

INPUT SPLIT 1

NODE A

MAPPER 1

INPUT SPLIT 2

NODE B

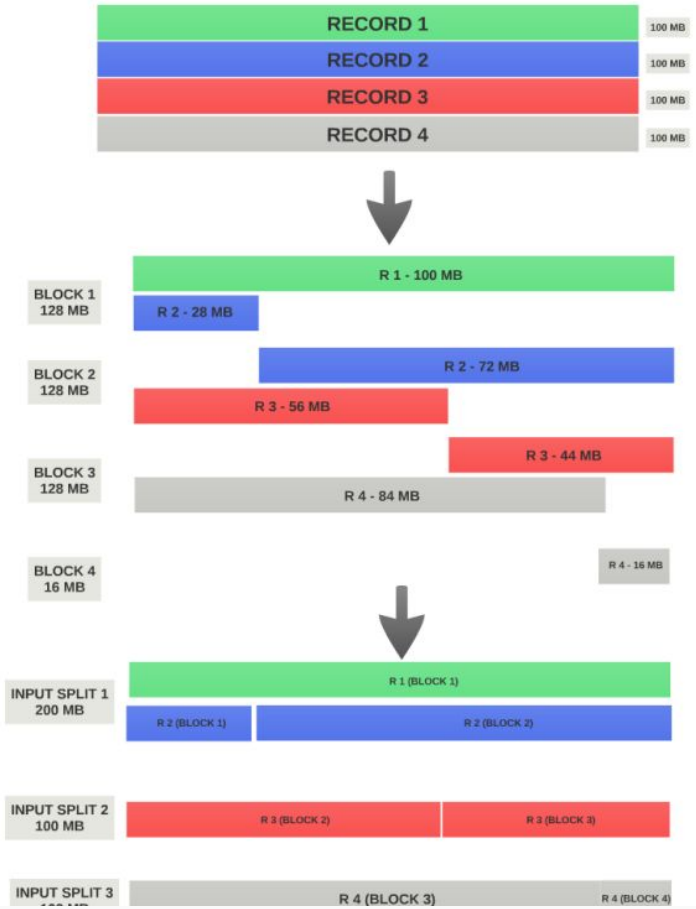
MAPPER 2

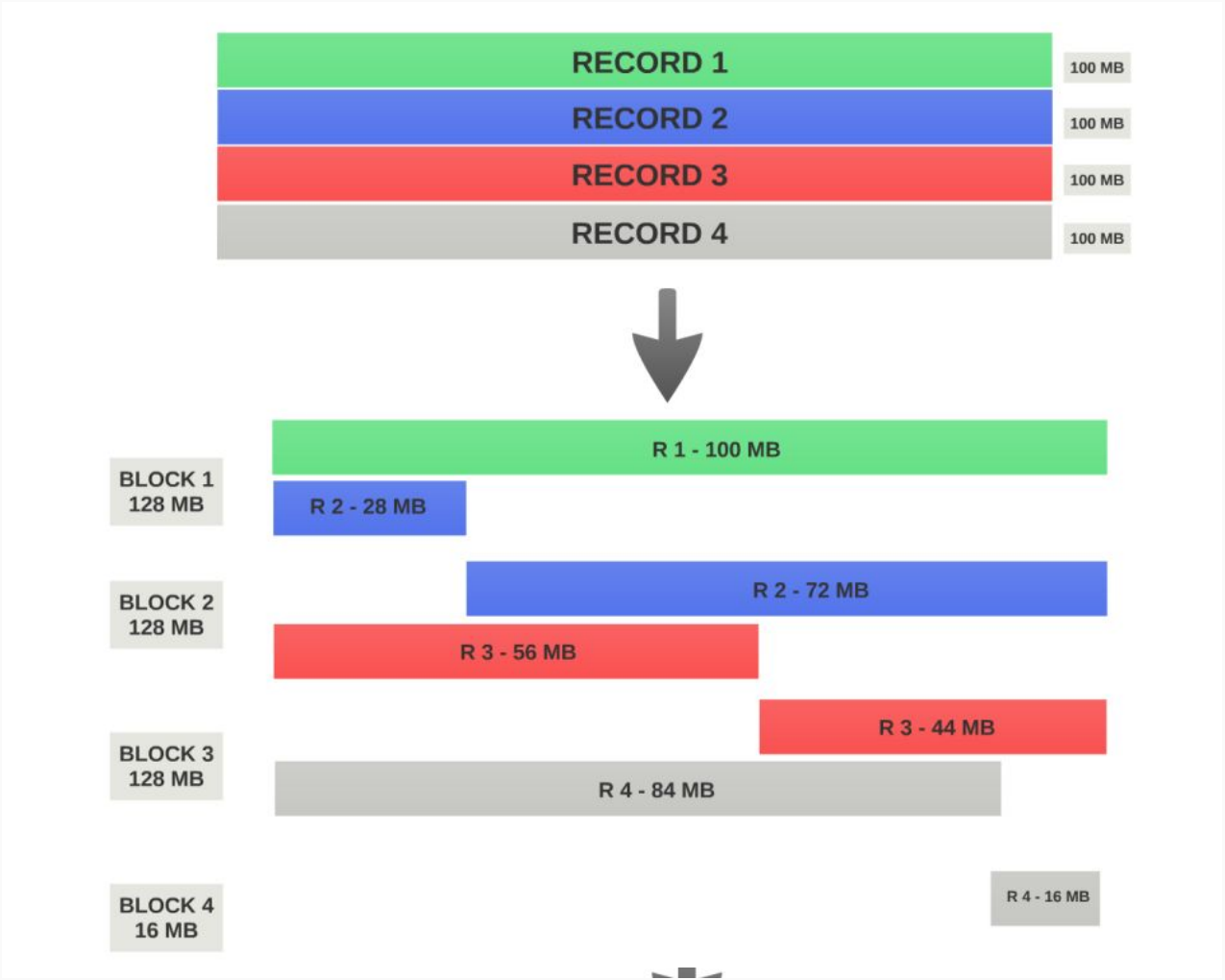
INPUT SPLIT 3

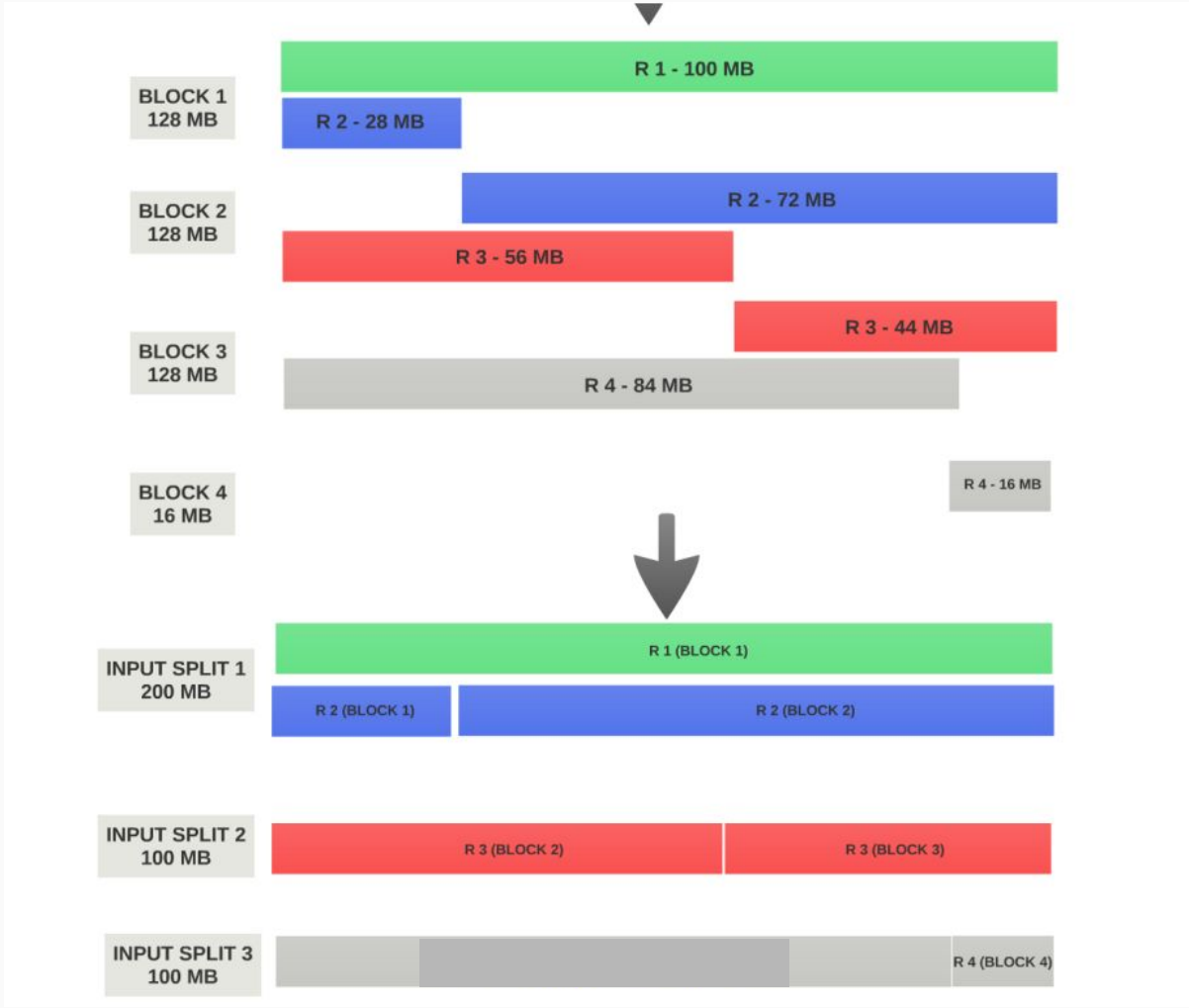
NODE C

MAPPER 3

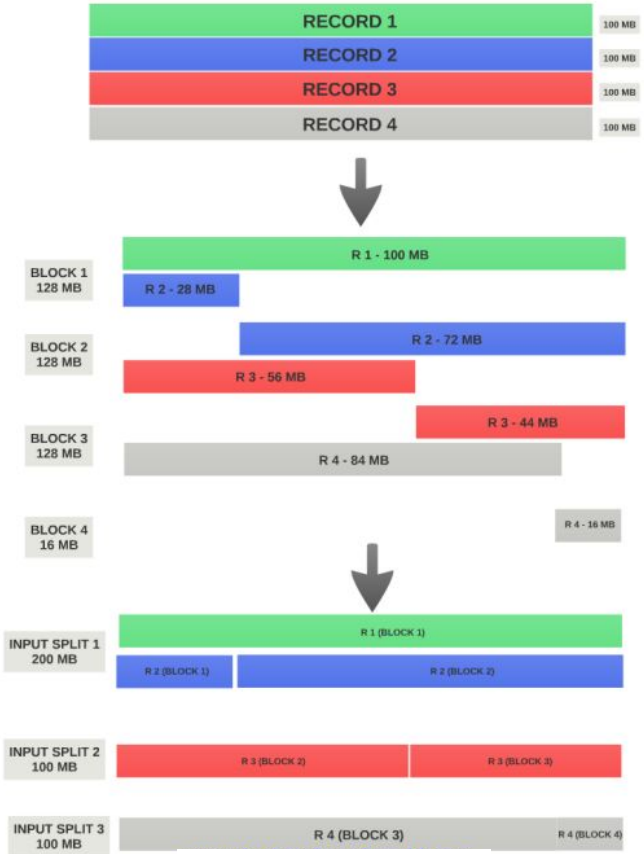
BLOCKS vs. INPUT SPLIT



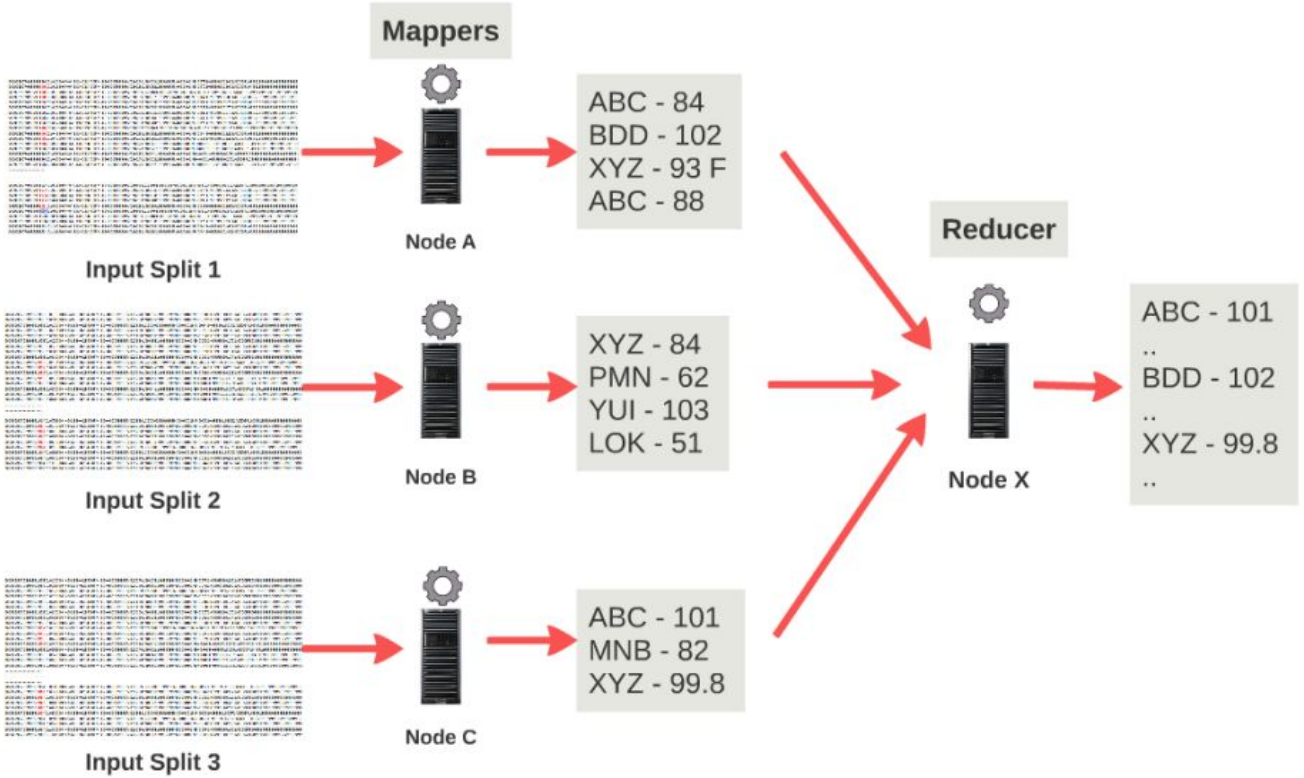

RECORD 1
100 MB
RECORD 2
100 MB
RECORD 3
100 MB
RECORD 4
100 MB
BLOCK 1
128 MB
R 1 - 100 MB
R 2 - 28 MB
BLOCK 2
128 MB
R 2 - 72 MB
R 3 - 56 MB
BLOCK 3
128 MB
R 3 - 44 MB
R 4 - 84 MB
BLOCK 4
16 MB
R 4 - 16 MB


BLOCK 1
128 MB
R 1 - 100 MB
R 2 - 28 MB
BLOCK 2
128 MB
R 2 - 72 MB
R 3 - 56 MB
BLOCK 3
128 MB
R 3 - 44 MB
R 4 - 84 MB
BLOCK 4
16 MB
R 4 - 16 MB
INPUT SPLIT 1
200 MB
R 1 (BLOCK 1)
R 2 (BLOCK 1)
R 2 (BLOCK 2)
INPUT SPLIT 2
100 MB
R 3 (BLOCK 2)
R 3 (BLOCK 3)
INPUT SPLIT 3
100 MB
R 4 (BLOCK 4)

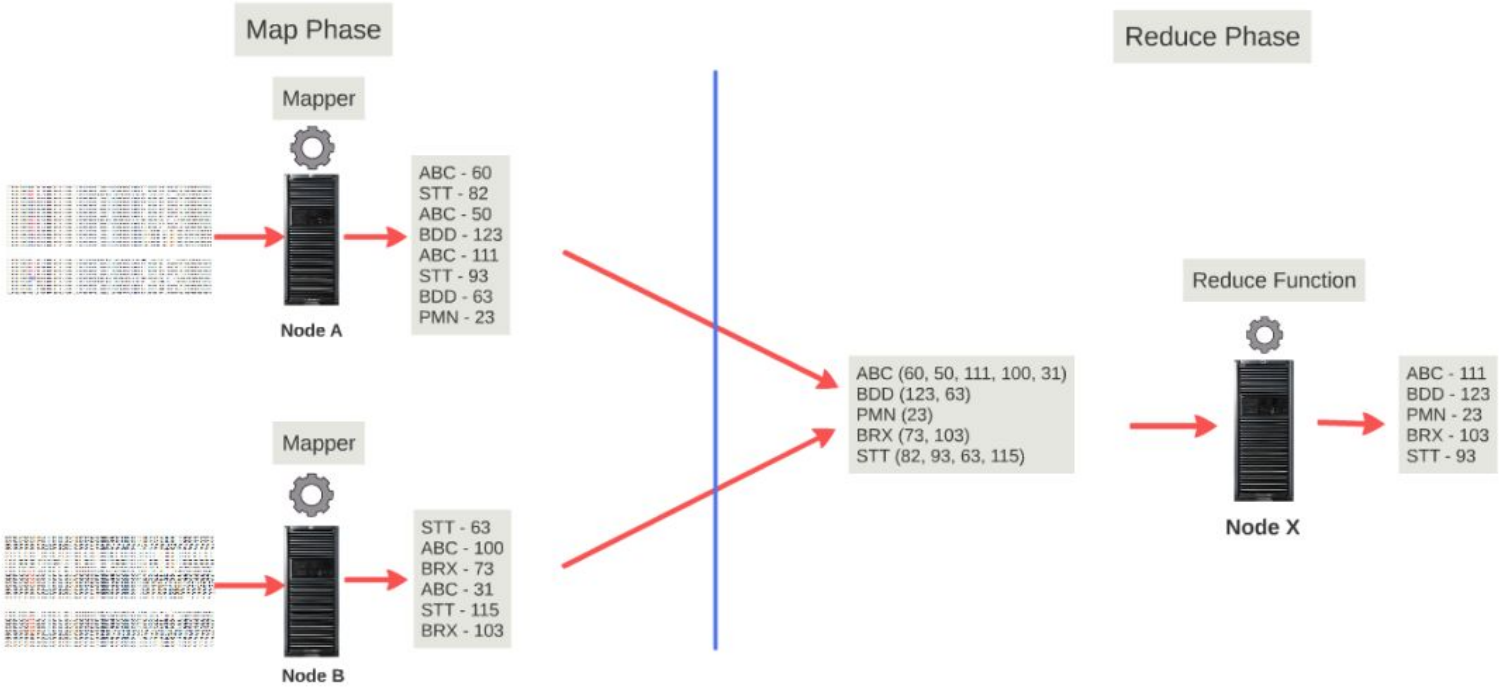
BLOCKS vs. INPUT SPLIT



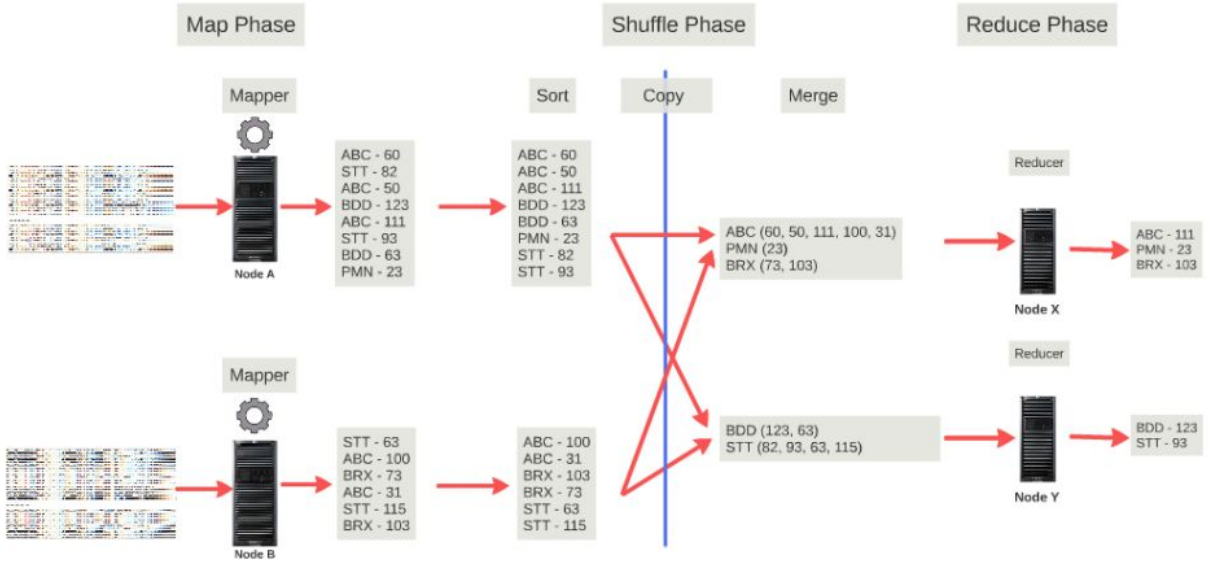
MAP PHASE


Mappers
Input Split 1
Node A
ABC - 84
BDD - 102
XYZ - 93 F
ABC - 88
Input Split 2
Node B
XYZ - 84
PMN - 62
YUI - 103
LOK - 51
Input Split 3
Node C
ABC - 101
MNB - 82
XYZ - 99.8
Reducer
Node X
ABC - 101
..
BDD - 102
..
XYZ - 99.8
..

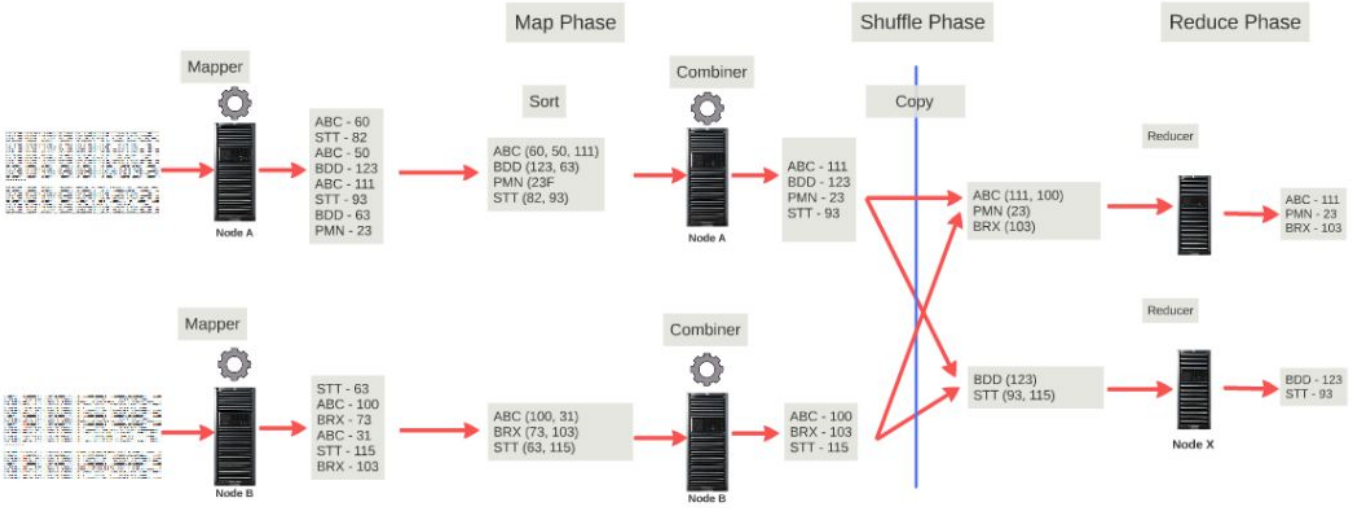
REDUCE PHASE



MULTIPLE REDUCERS



COMBINER (OPTIONAL)



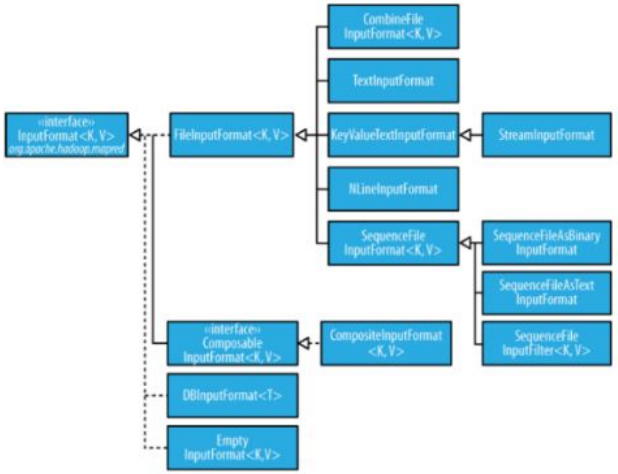
DISSECTING MAPREDUCE PROGRAM

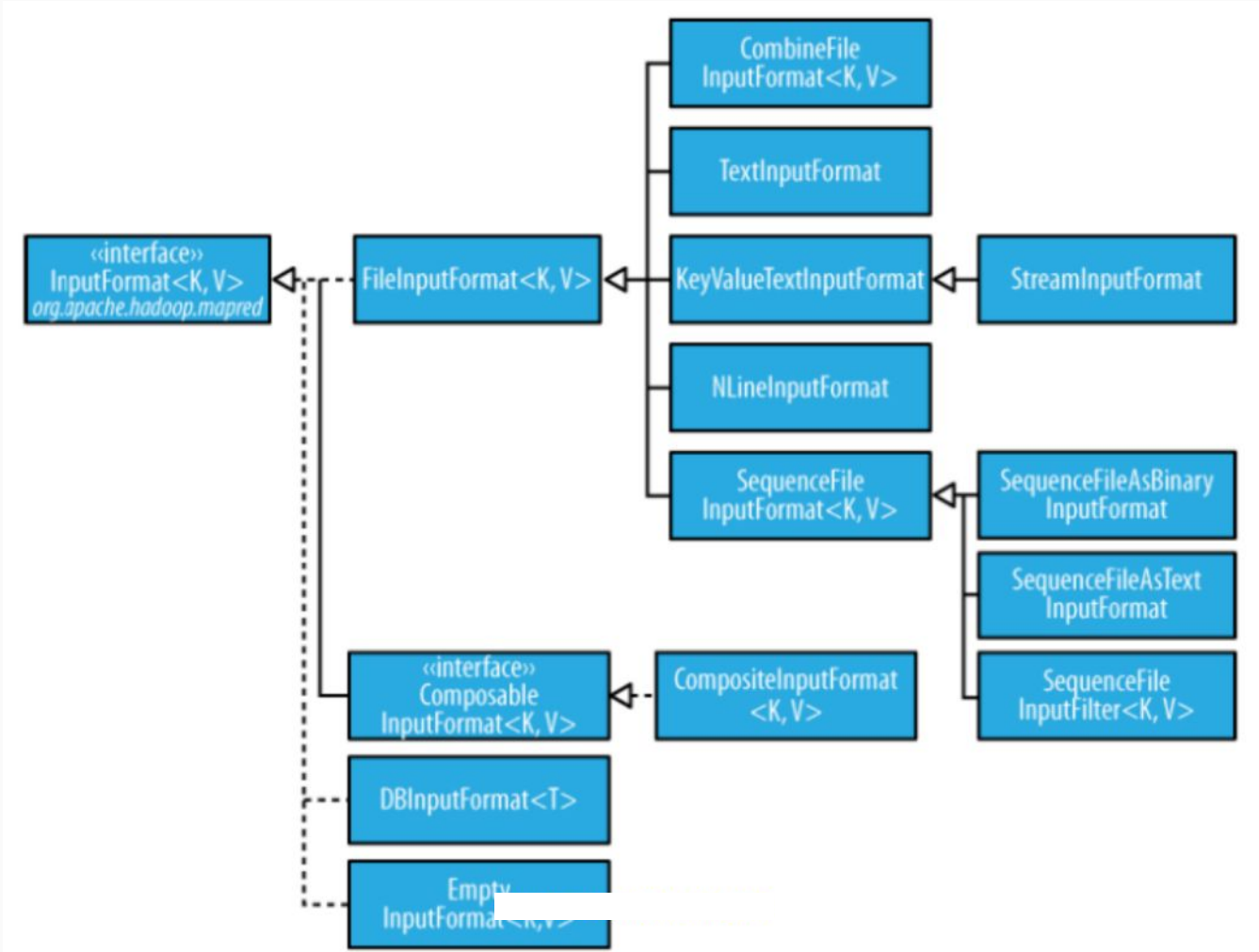
InputFormat

Validate inputs

Input files into logical InputSplits

RecordReader implementation to extract logical records



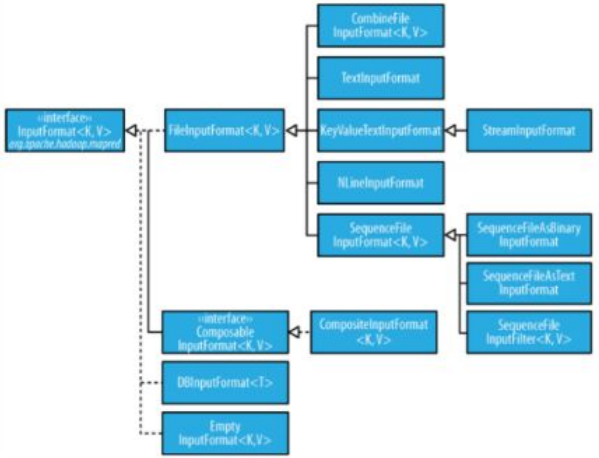

«interface»
InputFormat<K, V>
org.apache.hadoop.mapred
FileInputFormat<K, V>
CombineFile
InputFormat<K, V>
TextInputFormat
KeyValueTextInputFormat
StreamInputFormat
NLineInputFormat
SequenceFile
InputFormat<K, V>
SequenceFileAsBinary
InputFormat
SequenceFileAsText
InputFormat
SequenceFile
InputFilter<K, V>
«interface»
Composable
InputFormat<K, V>
CompositeInputFormat
<K, V>
DBInputFormat<T>

InputFormat

Validate inputs

Input files into logical InputSplits

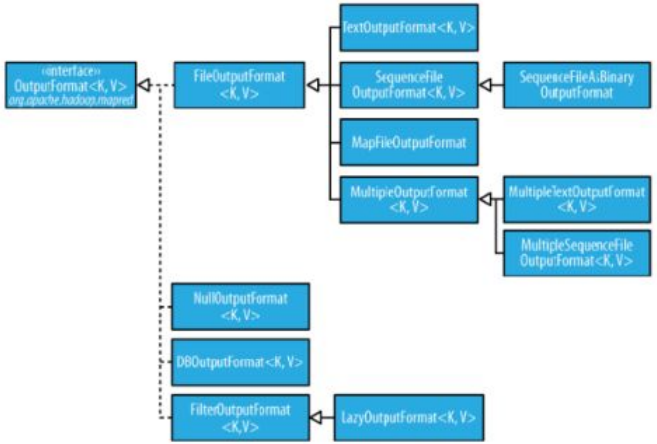
RecordReader implementation to extract logical records

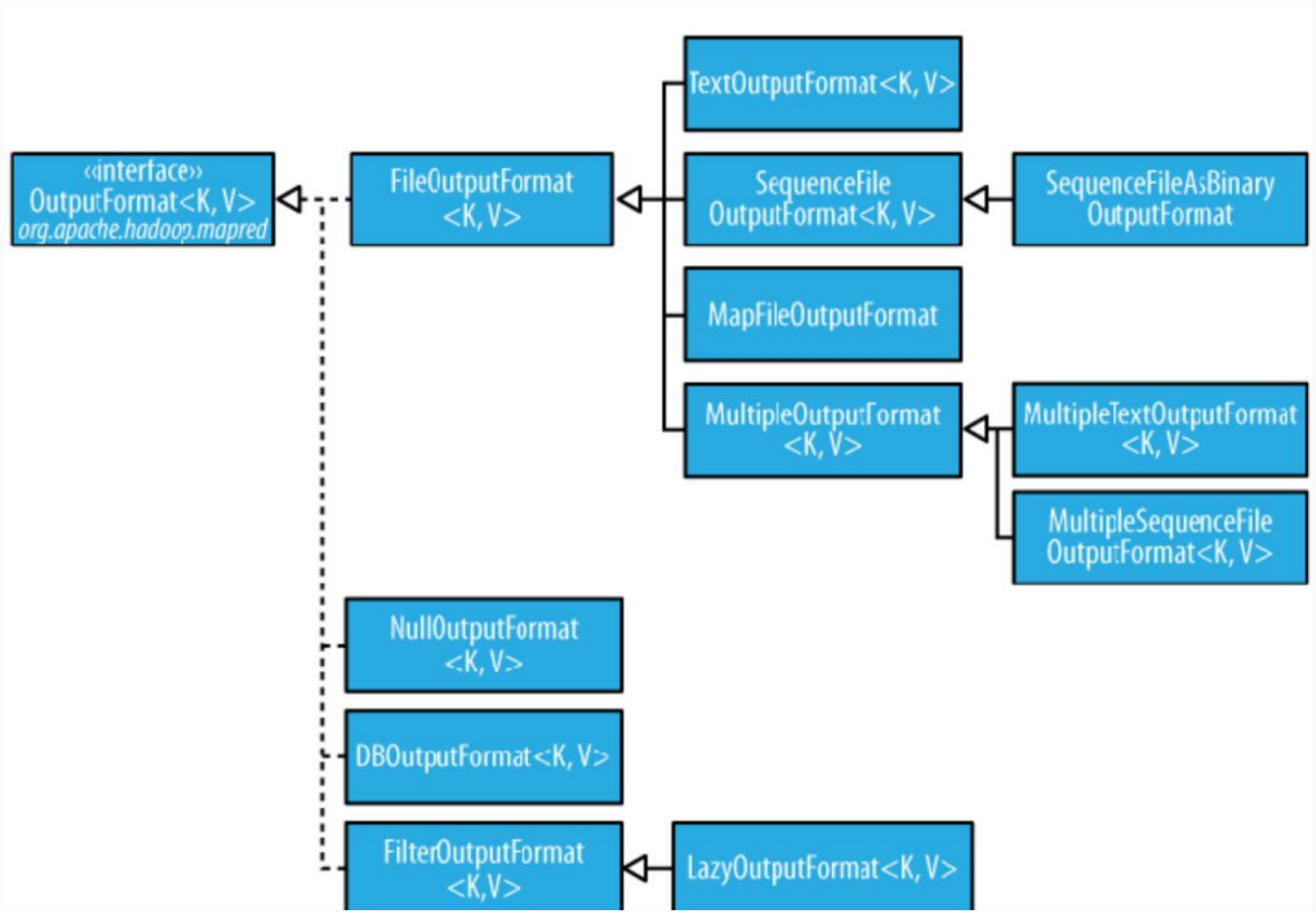


OutputFormat

Validate output specifications

RecordWriter implementation to write output files of the job

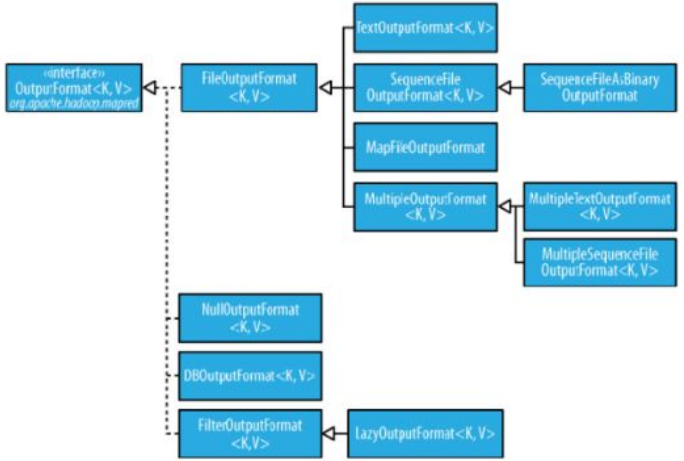



«interface»
OutputFormat<K, V>
org.apache.hadoop.mapred
FileOutputFormat
<K, V>
TextOutputFormat<K, V>
SequenceFile
OutputFormat<K, V>
SequenceFileAsBinary
OutputFormat
MapFileOutputFormat
MultipleOutputFormat
<K, V>
MultipleTextOutputFormat
<K, V>
MultipleSequenceFile
OutputFormat<K, V>
NullOutputFormat
<K, V>
DBOutputFormat<K, V>
FilterOutputFormat
<K,V>
LazyOutputFormat<K, V>

OutputFormat

Validate output specifications

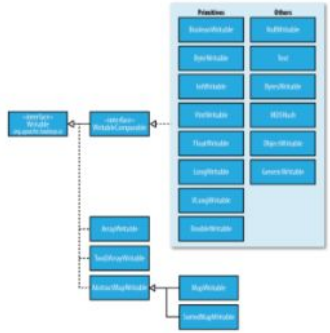
RecordWriter implementation to write output files of the job

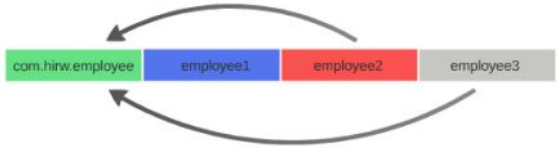


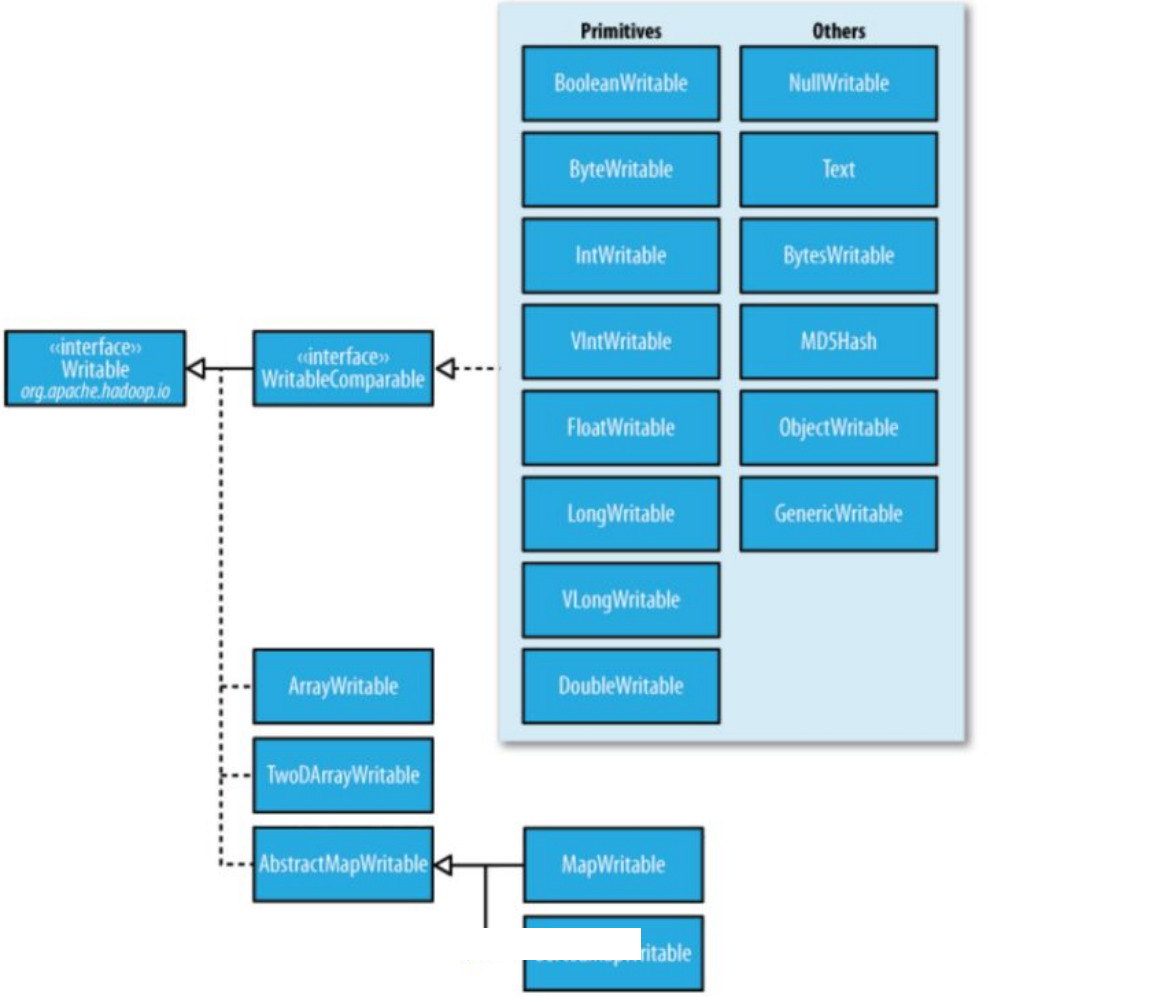
Writable

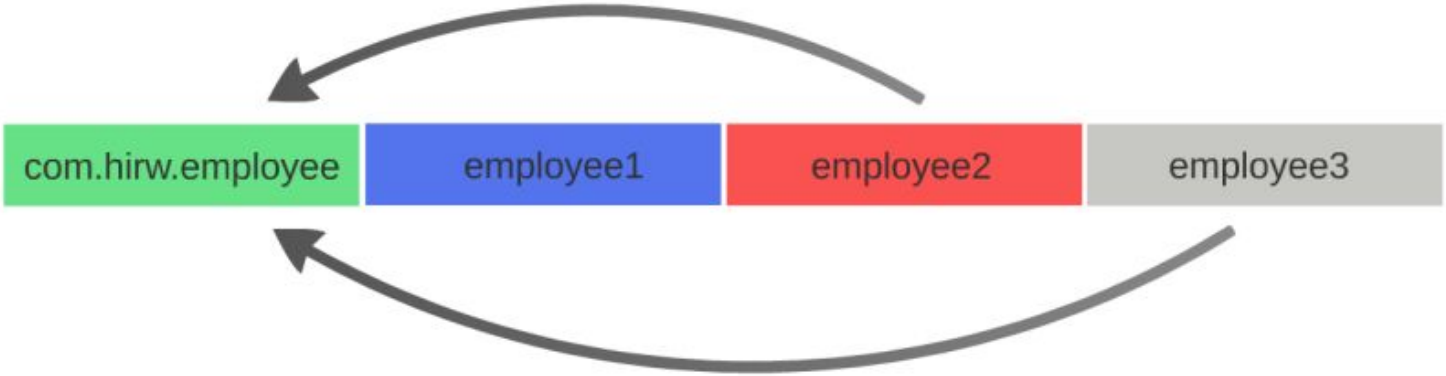
A serializable object which implements a simple, efficient, serialization protocol

Fast, compact & effective





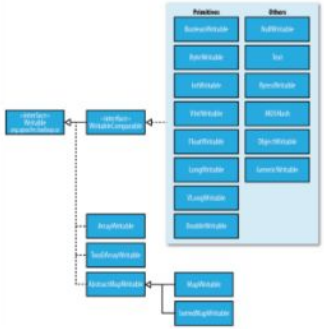

Primitives
Others
BooleanWritable
NullWritable
ByteWritable
Text
IntWritable
BytesWritable
VIntWritable
MD5Hash
FloatWritable
ObjectWritable
LongWritable
GenericWritable
VLongWritable
DoubleWritable
«interface»
Writable
org.apache.hadoop.io
«interface»
WritableComparable
ArrayWritable
TwoDArrayWritable
AbstractMapWritable
MapWritable

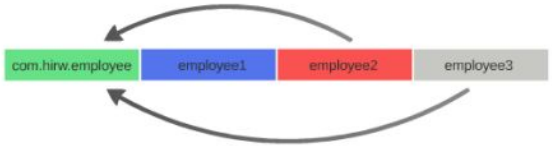

com.hirw.employee
employee1
employee2
employee3

Writable

A serializable object which implements a simple, efficient, serialization protocol

Fast, compact & effective





MAPPER

- Dataset is divided in to multiple parts - Input Splits
- Each Mapper process an Input Split
- Each Mapper can be called multiple times depending on the content of Input Split
- Mapper will emit Key Value pairs as output
- There will be one or more Mapper in a MapReduce job

REDUCER

- Reduce function will take Key Value pairs from multiple Map functions as input and Reduce them to output
- Keys are grouped with values. Reduce function is called once per key and its values.
- There could be 0, 1 or more Reduce function for a MapReduce job