CPSC 4330: Big Data Analytics

MapReduce

Features of MapReduce

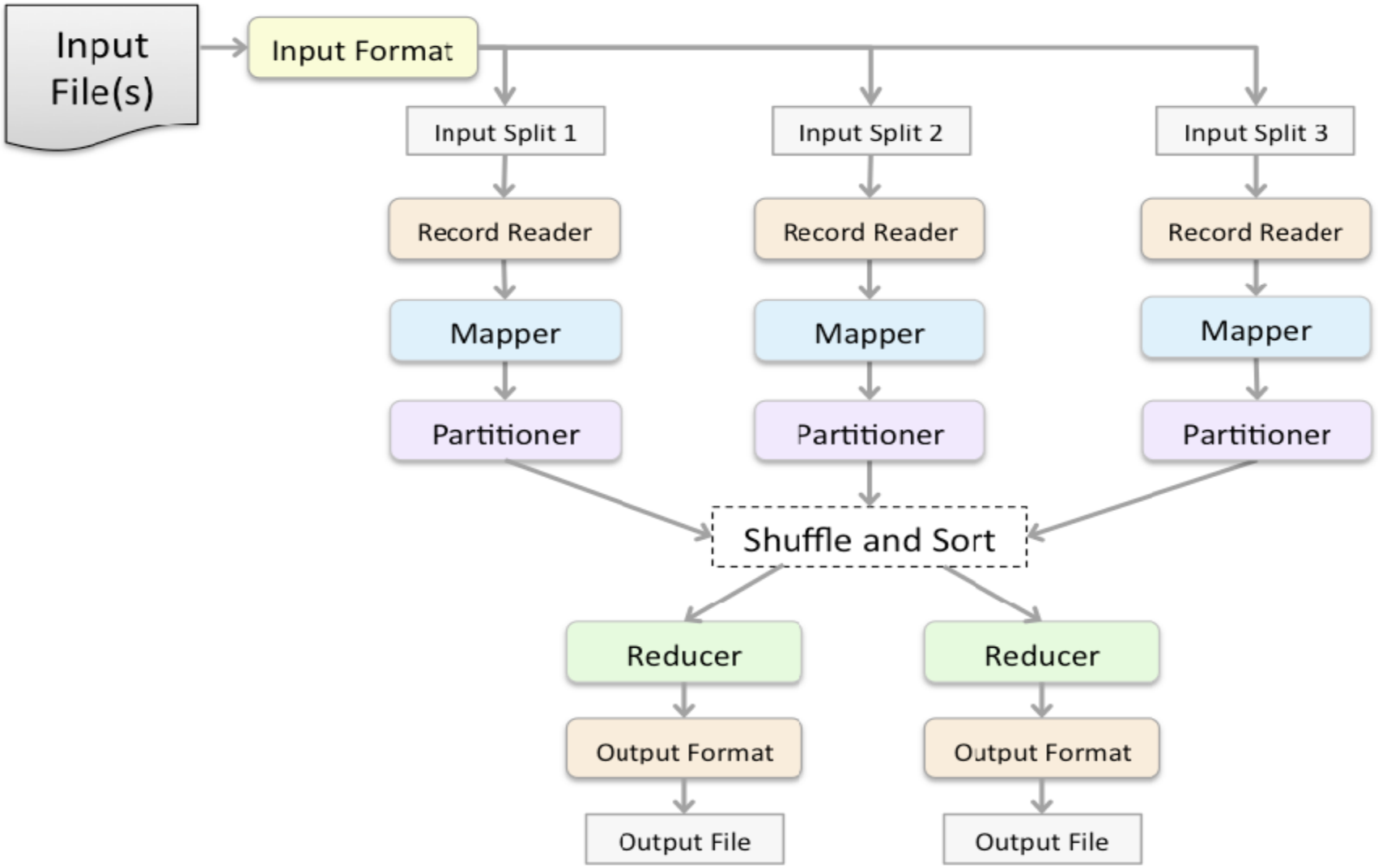
- Automatic parallelization and distribution
- A clean abstraction for programmers
- MapReduce abstracts all the ‘housekeeping’ away from the developer
- Programmers only need to concentrate on business logic

Key MapReduce Stages

- The Mapper
  - Each Map task (typically) operates on a single HDFS block
  - Map tasks (usually) run on the node where the block is stored

- Shuffle and Sort
  - Sorts and consolidates intermediate data from all mappers
  - Happens after all Map tasks are complete and before Reduce tasks start
- The Reducer
  - Operates on shuffled/sorted intermediate data (Map task output)
  - Produces final output

The MapReduce Flow



- The Input File is your input data that you loaded into HDFS
- The InputFormat splits the file up into multiple input splits and it defines how input files are split and read. One map task is created for each split (block).

SequenceFileInputFormat

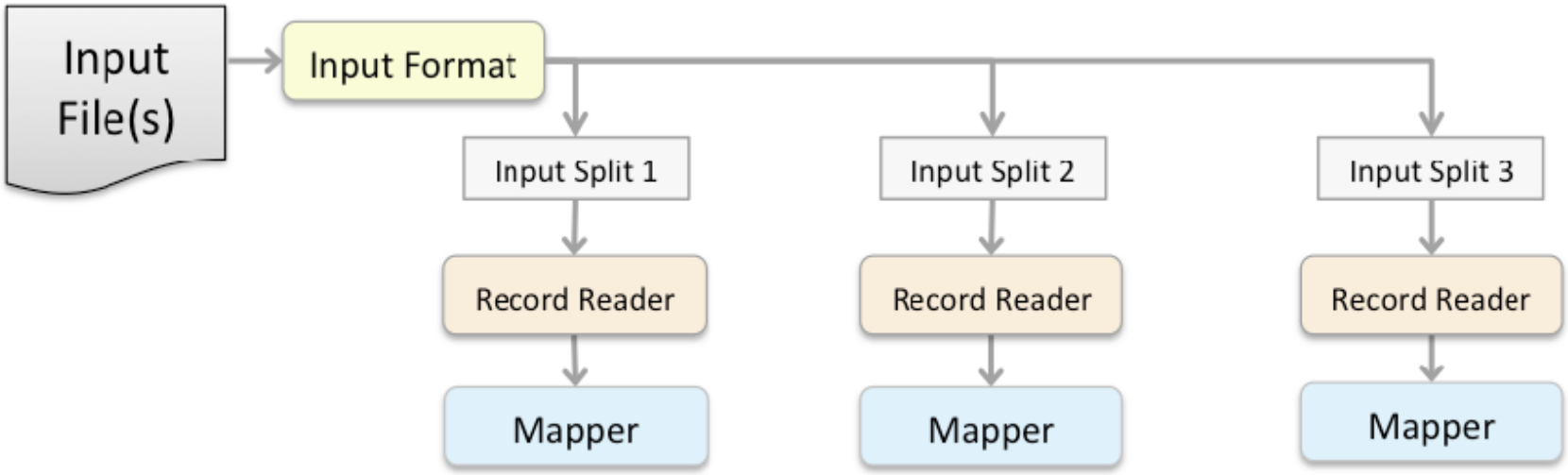
TextInputFormat

FileInputFormat

DBInputFormat

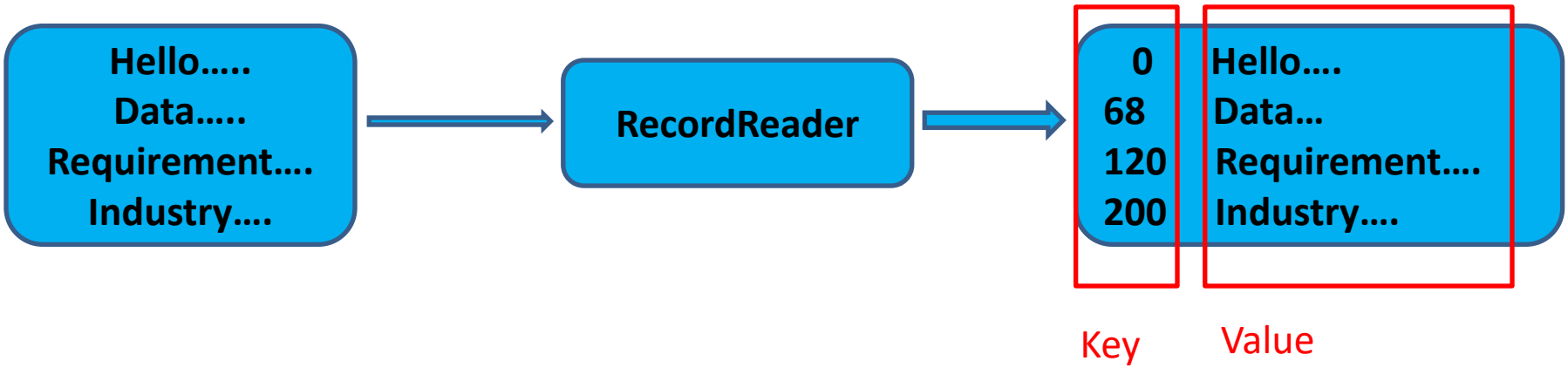
KeyValueTextInputFormat

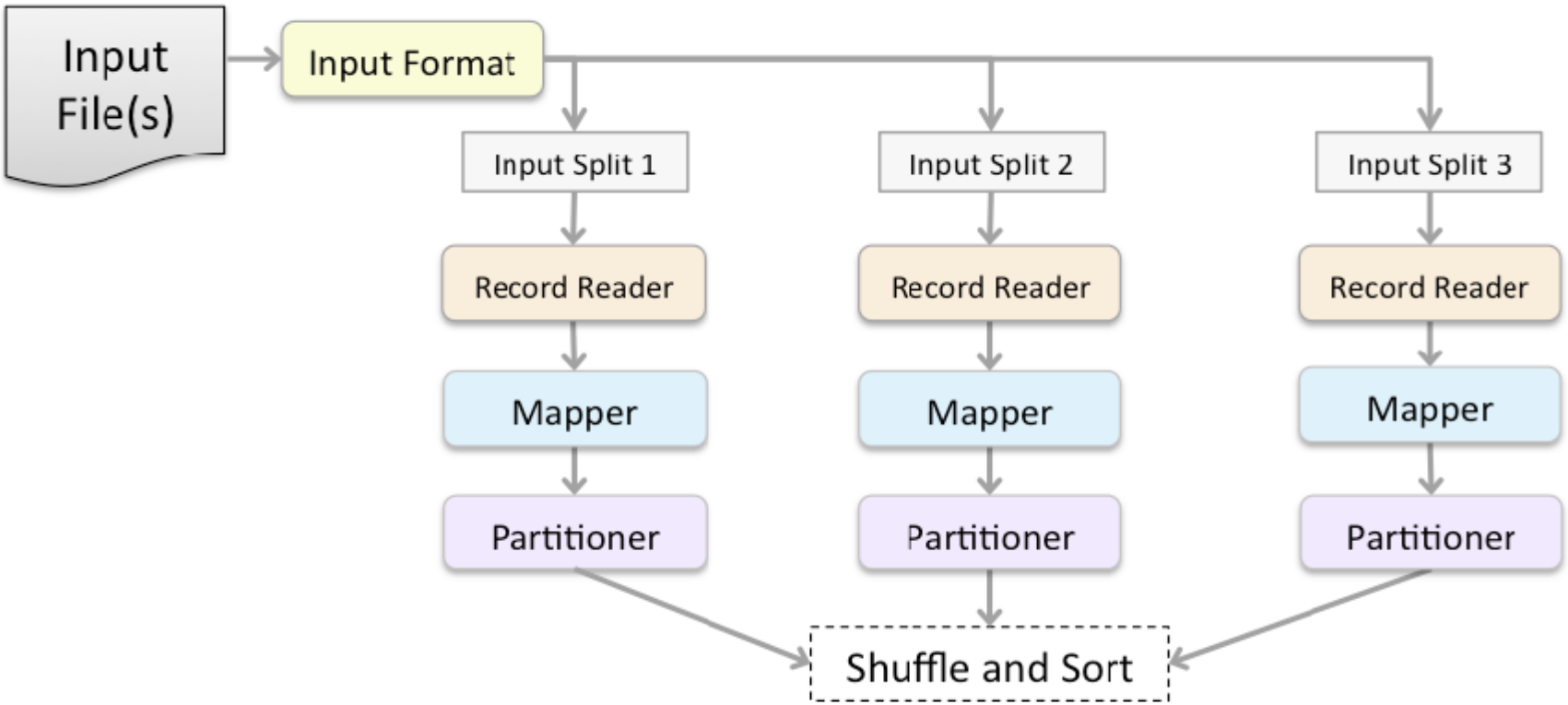
.....



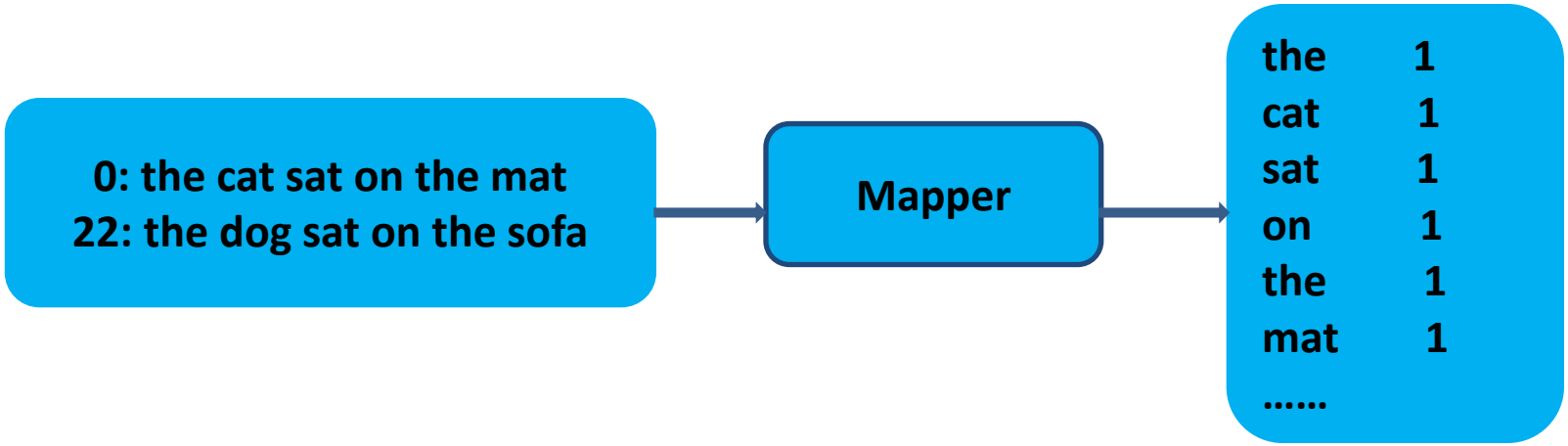
InputFormat defines the Record Reader, which is responsible for reading actual records from the input files.

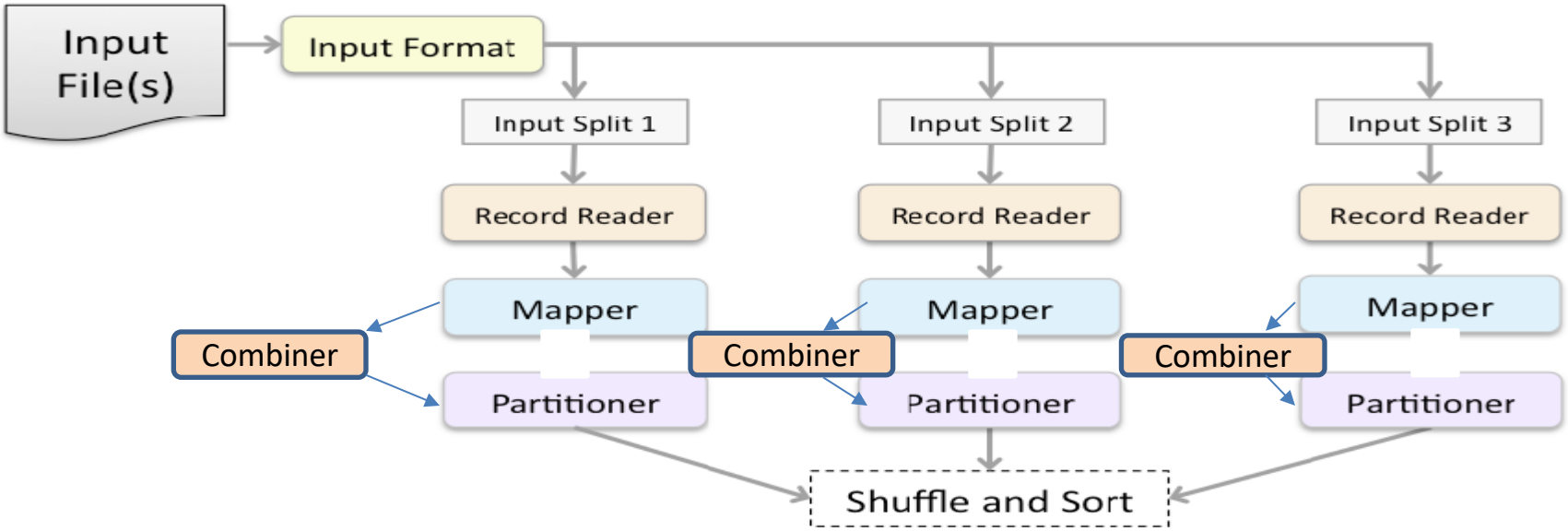
Record Reader: convert the data from InputSplit into key-value pairs suitable for reading by the mapper.





Mapper: processes each input record (from RecordReader) and generates new key-value pair.





Combiner (Mini-reducer): performs local aggregation on the mapper's output.

Example: Helps to minimize the data transfer between mapper and reducer.
Input:
Block 1: Hello World Bye World
Block 2: Hello Hadoop Goodbye Hadoop

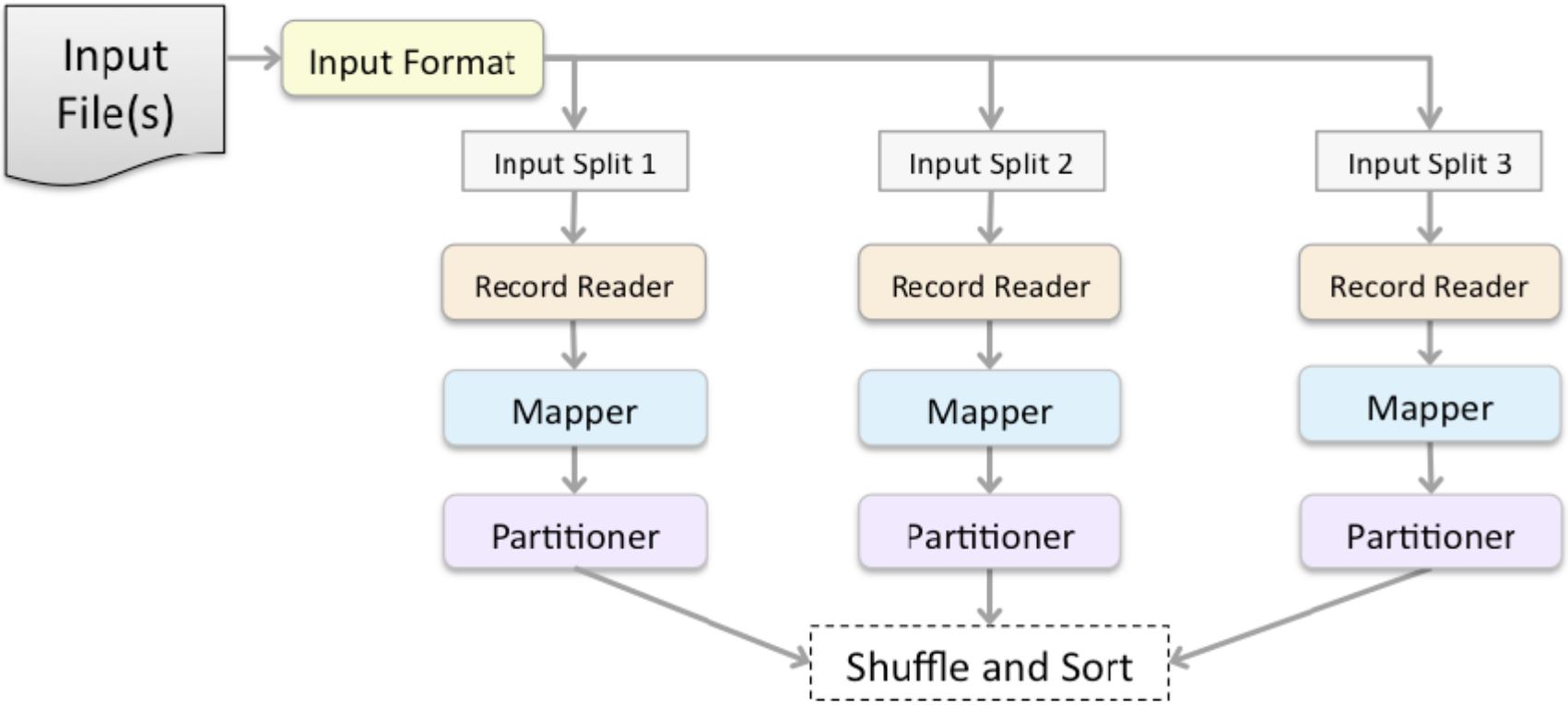
8 keys

Output of Mapper 1: <Hello, 1> <World, 1> <Bye, 1> <World, 1>
Output of Mapper 2: <Hello, 1> <Hadoop, 1> <Goodbye, 1> <Hadoop, 1>

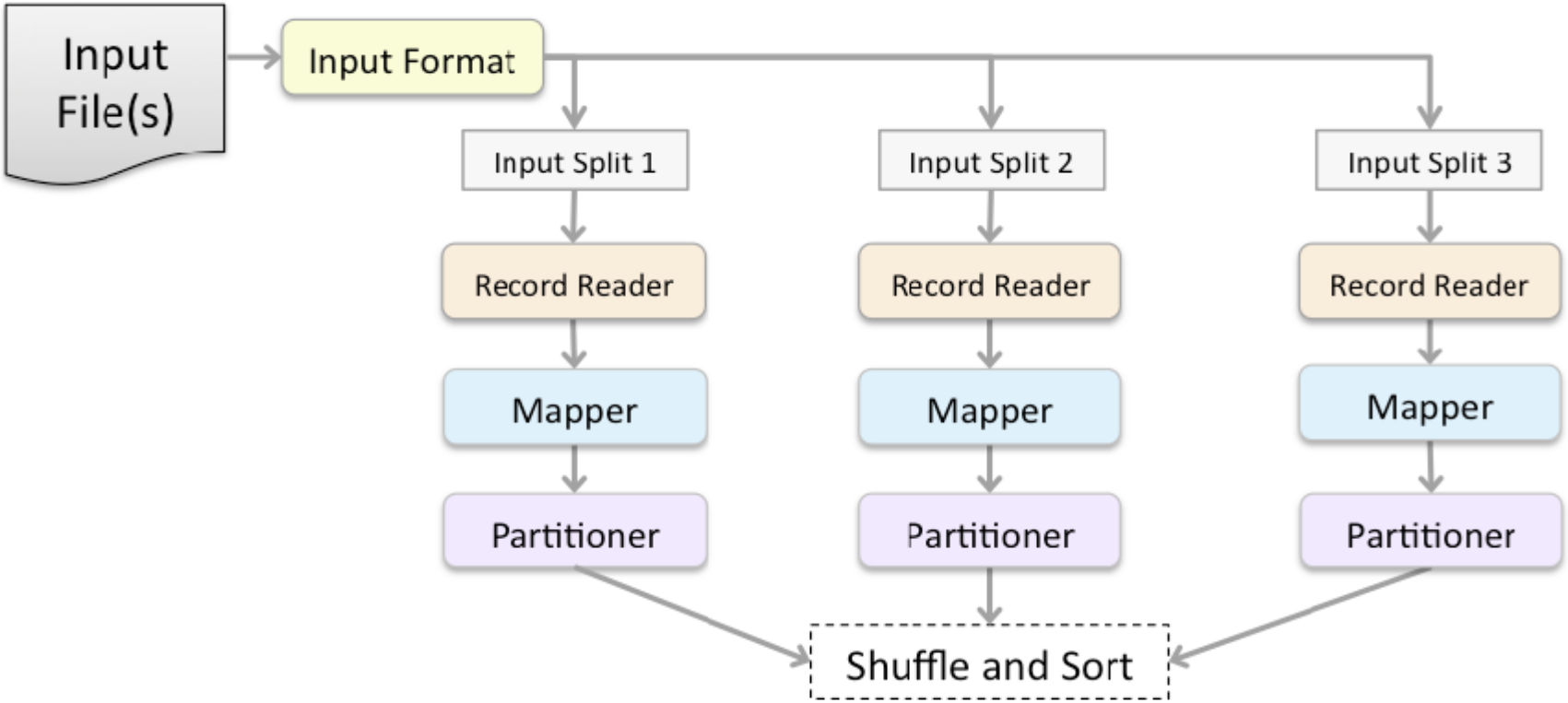
With Combiner:

6 keys

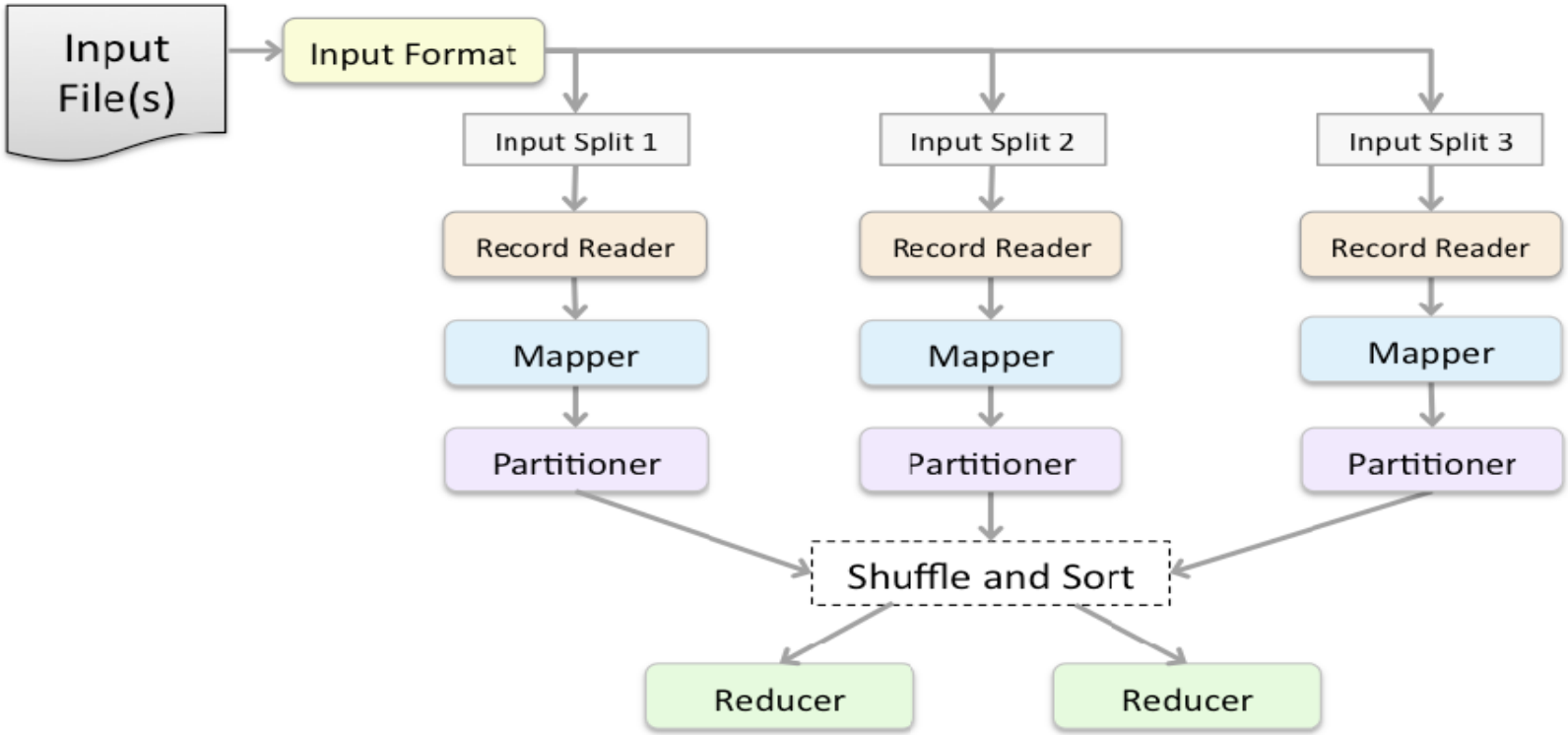
The output of Combiner 1: <Bye, 1> <Hello, 1> <World, 2>
The output of Combiner 2: <Goodbye, 1> <Hadoop, 2> <Hello, 1>



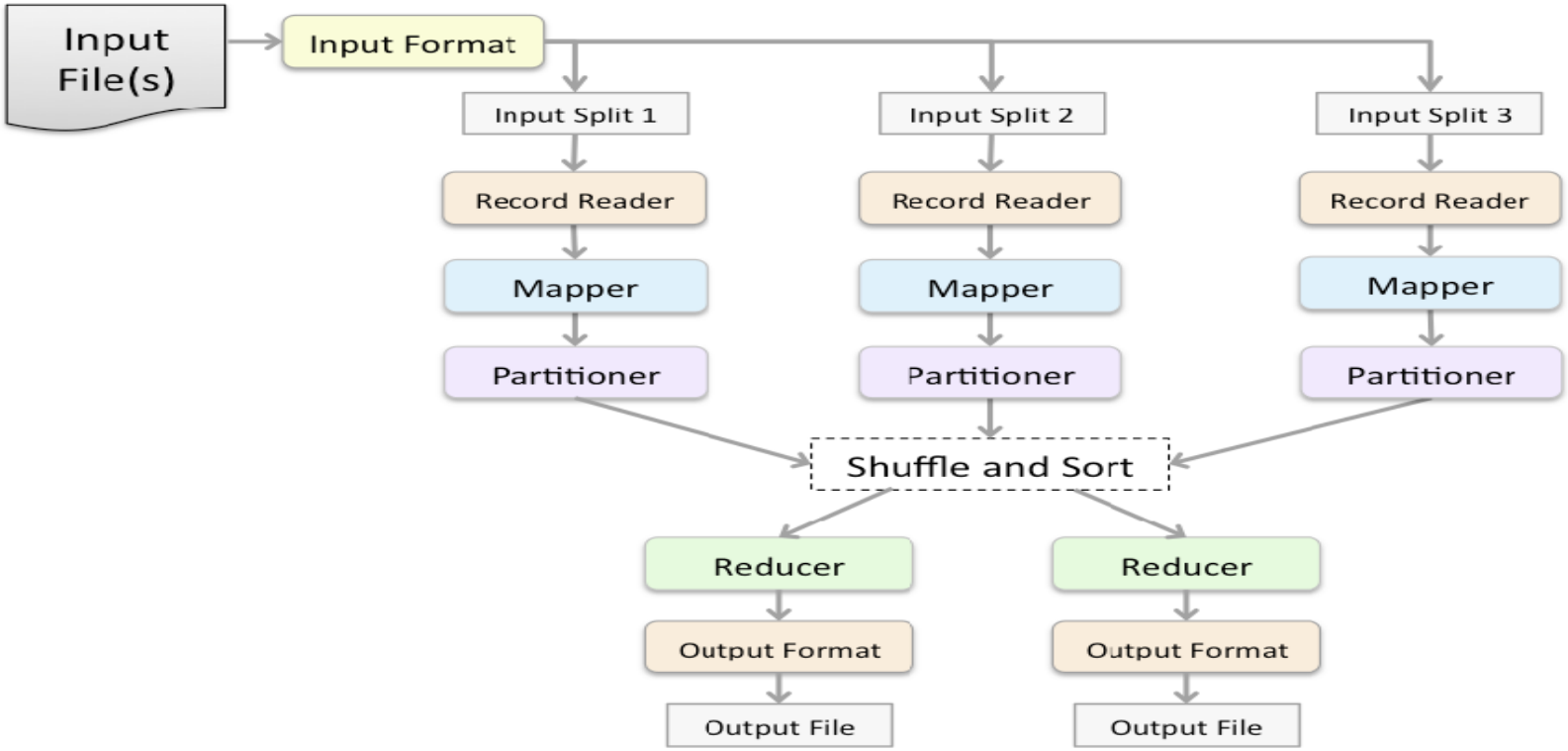
- The Partitioner is the class that determines which Reducer a given key should go to.
- Records having the same key go into the same partition, and then each partition is sent to a reducer.
- The default Partitioner class in Hadoop is HashPartitioner which computes a hash value for the key and assigns the partition based on the result.
- Partitioner is needed if more than one reducer is used. If only one reducer is used, partitioner is not needed.



- Shuffle: intermediate output from mappers are transferred to reducers.
- Sort: Intermediate key-value pairs generated by mapper are merged and sorted automatically by key.
- Shuffle and sort is not performed at all if zero reducer is used (map-only job). The output of mapper is written to local disk before sending to reducer but in map only job, this output is directly written to HDFS.



- Reducer: takes the set of intermediate key-value pairs (produced by mappers) and runs a reducer function on each of them to generate output.
- The output of reducer is stored on HDFS.
- The user decides the number of reducers.
- By default, the number of reducers is 1.



- OutputFormat: determines how the output of Reducer is written to files in HDFS. Provides the RecordWriter implementation to write the output files.

TextOutputFormat

DBOutputFormat

SequenceFileOutputFormat

……

Example: Movie Rating

- Given files containing movie rating information, we want to know how many movies get 5 stars, how many movies get 4 stars,… etc.

User ID	Movie ID	Rating	Timestamp
196	242	3	881250949
186	302	3	891717742
196	377	1	878887116
244	51	2	880660923
166	346	1	886397596
186	474	4	884182806
186	265	2	881171488

How to make it a mapreduce problem?

Example: Movie Rating

- Map:

User ID	Movie ID	Rating	Timestamp
196	242	3	881250949
186	302	3	891717742
196	377	1	878887116
244	51	2	880660923
166	346	1	886397596
186	474	4	884182806
186	265	2	881171488

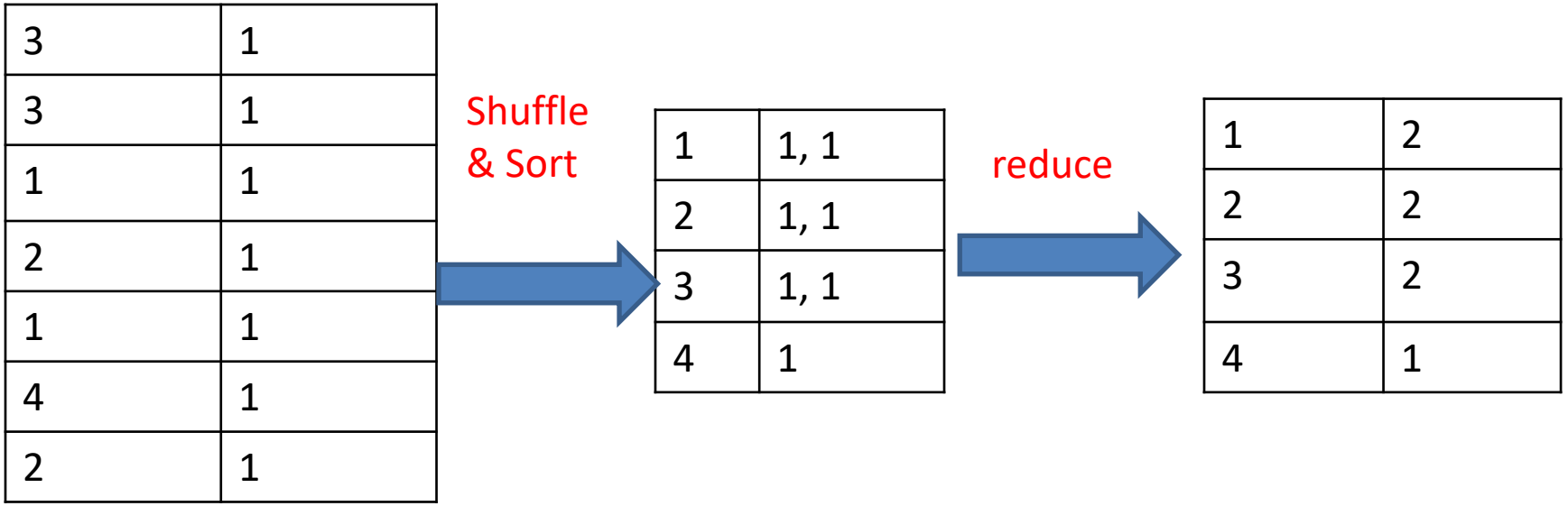
Map

3	1
3	1
1	1
2	1
1	1
4	1
2	1

Key: Rating
Value: 1

Example: Movie Rating

- Reduce:



Example: Soccer games

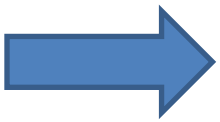
- Input: score sheet of soccer games for a team

Game =17	Date = 230515	Goals = 3	Ben = 2	Tom =1
Game = 18	Date = 240515	Goals = 1	Mike = 1	
Game = 19	Date = 250515	Goals = 2	Ben = 1	Mike = 1
Game = 20	Date = 260515	Goals = 1	Ben = 1	
Game = 21	Date = 270515	Goals = 4	Tom = 3	Mike = 1
Game = 22	Date = 280515	Goals = 1	Mike = 1	

How to find total goals of each player?

Game =17	Date = 230515	Goals = 3	Ben = 2	Tom =1
Game = 18	Date = 240515	Goals = 1	Mike = 1	

map

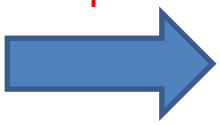


Ben	2
Tom	1
Mike	1

Node 1

Game = 19	Date = 250515	Goals = 2	Ben = 1	Mike = 1
Game = 20	Date = 260515	Goals = 1	Ben = 1	

map

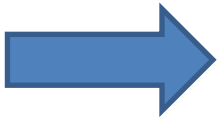


Ben	1
Mike	1
Ben	1

Node 2

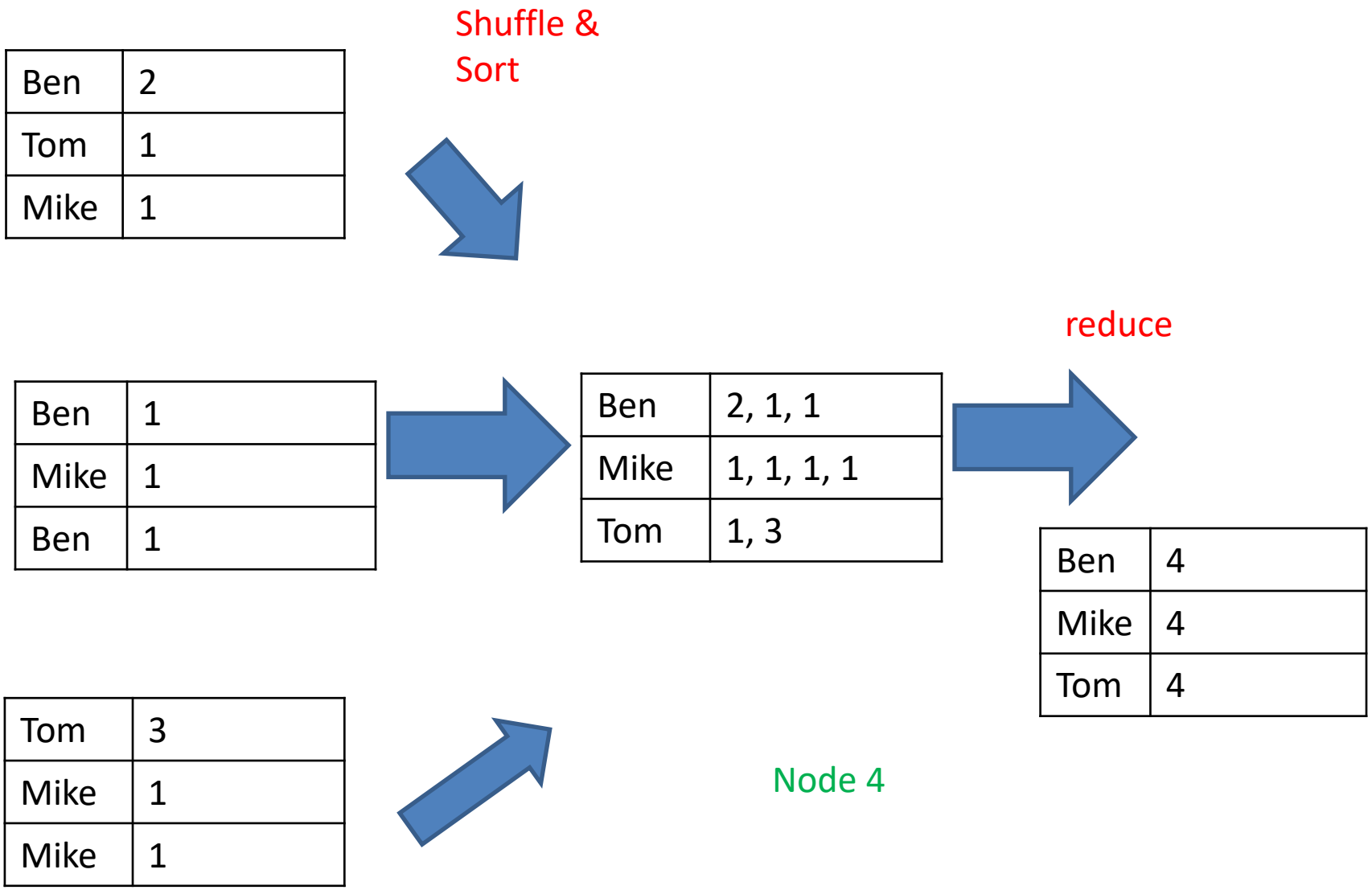
Game = 21	Date = 270515	Goals = 4	Tom = 3	Mike = 1
Game = 22	Date = 280515	Goals = 1	Mike = 1	

map



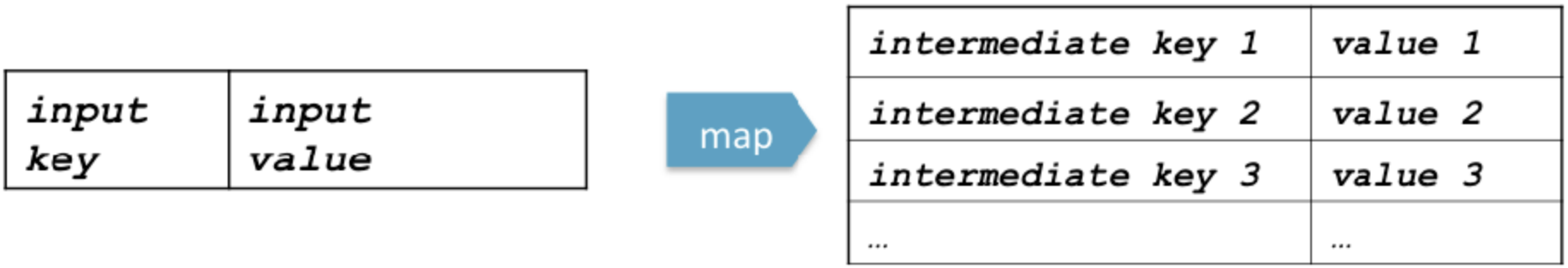
Tom	3
Mike	1
Mike	1

Node 3


Shuffle &
Sort
Ben 2
Tom 1
Mike 1
Ben 1
Mike 1
Ben 1
Ben 2, 1, 1
Mike 1, 1, 1, 1
Tom 1, 3
reduce
Ben 4
Mike 4
Tom 4
Tom 3
Mike 1
Mike 1
Node 4

MapReduce: The Mapper

- The Mapper
  - Input: key/value pair
  - Output: A list of key value pairs



Example Mapper 1: Ignoring the input key

- The Mapper may use or completely ignore the input key.
  - For example, a standard pattern is to read one line of a file at a time
  - The key is the byte offset into the file at which the line starts
  - The value is the contents of the line itself

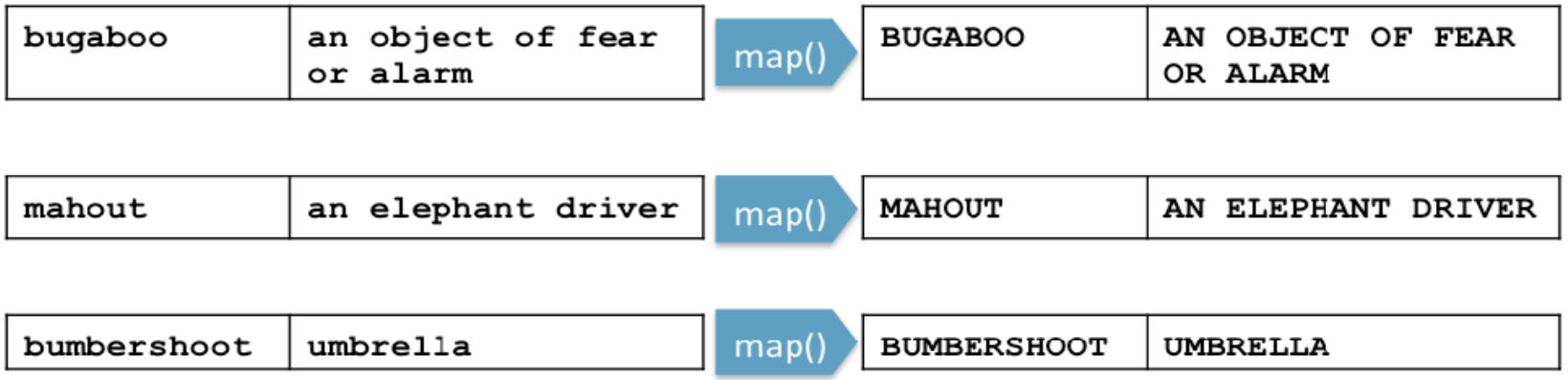
23	the aardvark sat on the sofa

map

the	1
aardvark	1
sat	1
on	1
the	1
sofa	1

Example Mapper 2: Transforming the input key

- Example: Turn input into upper case:



key: the word being defined

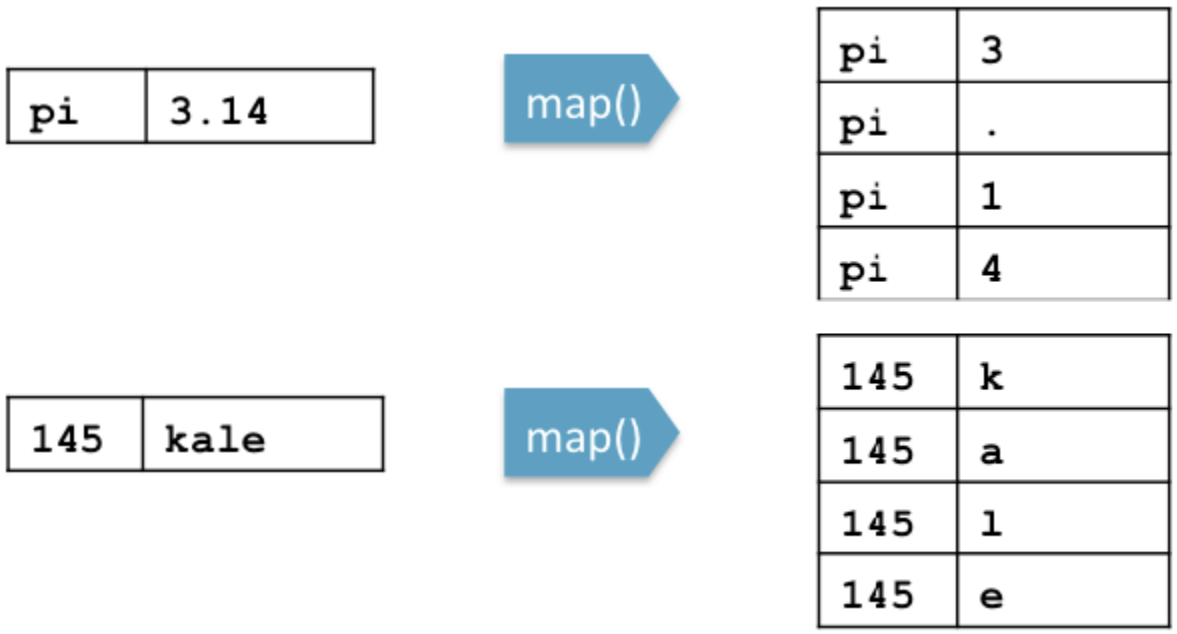
value: the definition of the word

Other examples:

1. The key could be a product ID. The mapper takes product ID and transforms them to product name.
2. The key could be an IP address. The mapper takes IP address and transforms to a hostname or geographic region.

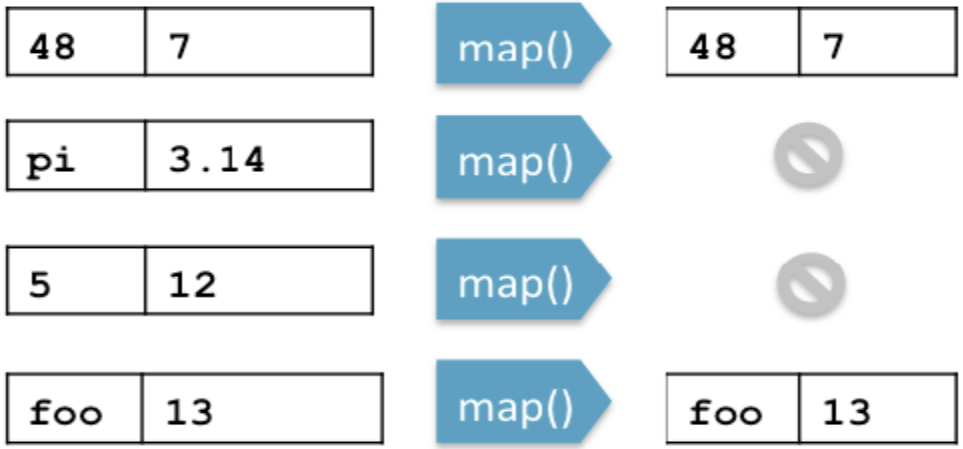
Example Mapper 3: ‘Explode’ Mapper

- Output each character of input values separately



Example Mapper 4: 'Filter' Mapper

- Example: Only output key/value pairs where the input value is a prime number :

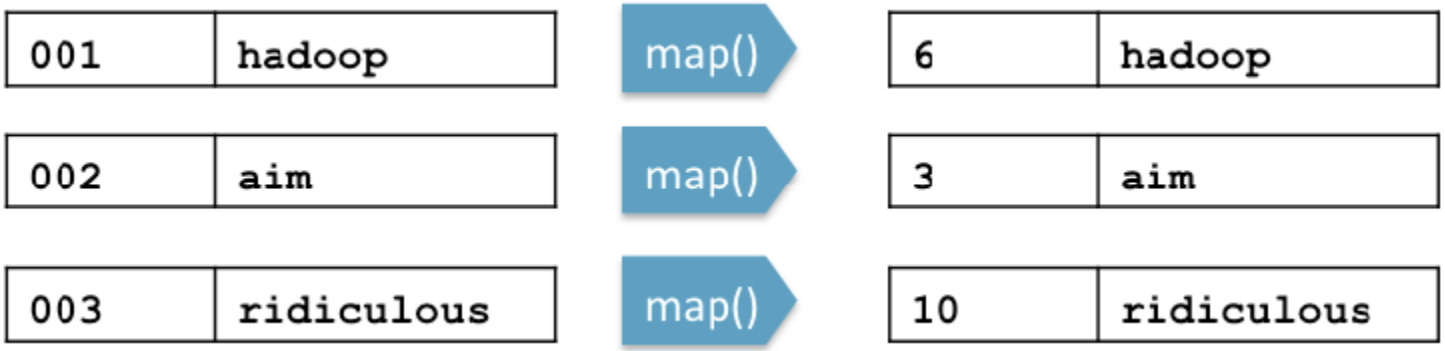


Other examples:

1. Filter out text which does (or does not) match some pattern
2. Take every Nth (e.g. 1,000) record from input to produce a sample of input data.

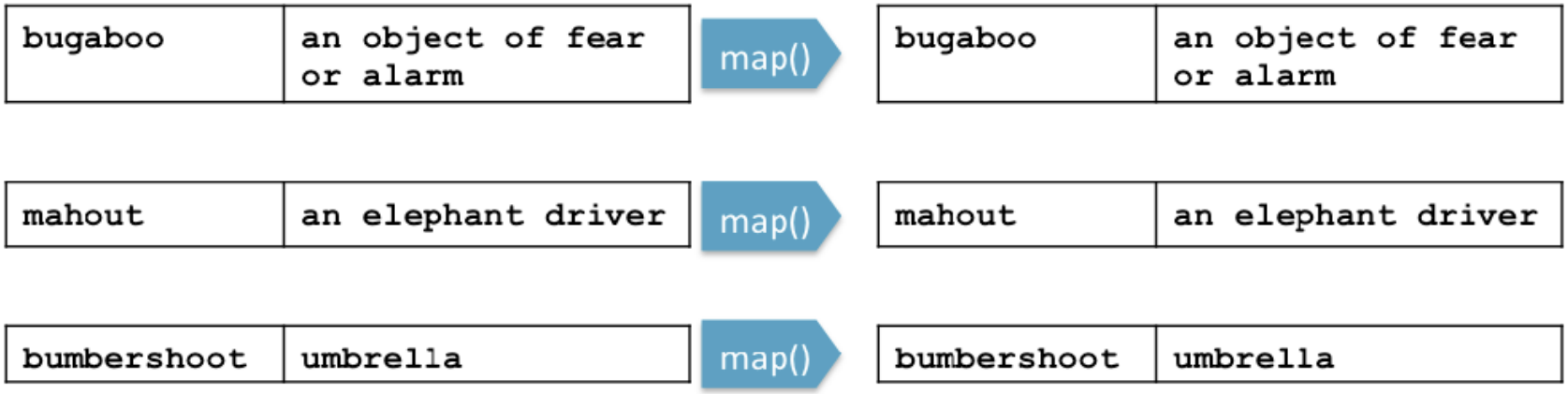
Example Mapper 5: Changing Keyspaces

- The type of the input key does not have to be the same type as the output key
- Example: output the length of input value as the key



Example Mapper 6: Identity Mapper

- Output the input key, value pair:



- Why do we need a mapper that outputs what we input?
  -- It's a way to get data into a Hadoop job so that other parts of Hadoop can operate on it (i.e. shuffle, sort, reduce). For example, can be used to filter out duplicates.

- The identity mapper is the default mapper. If you create a job and don't specify a mapper, the identity mapper will be used.

The Reducer

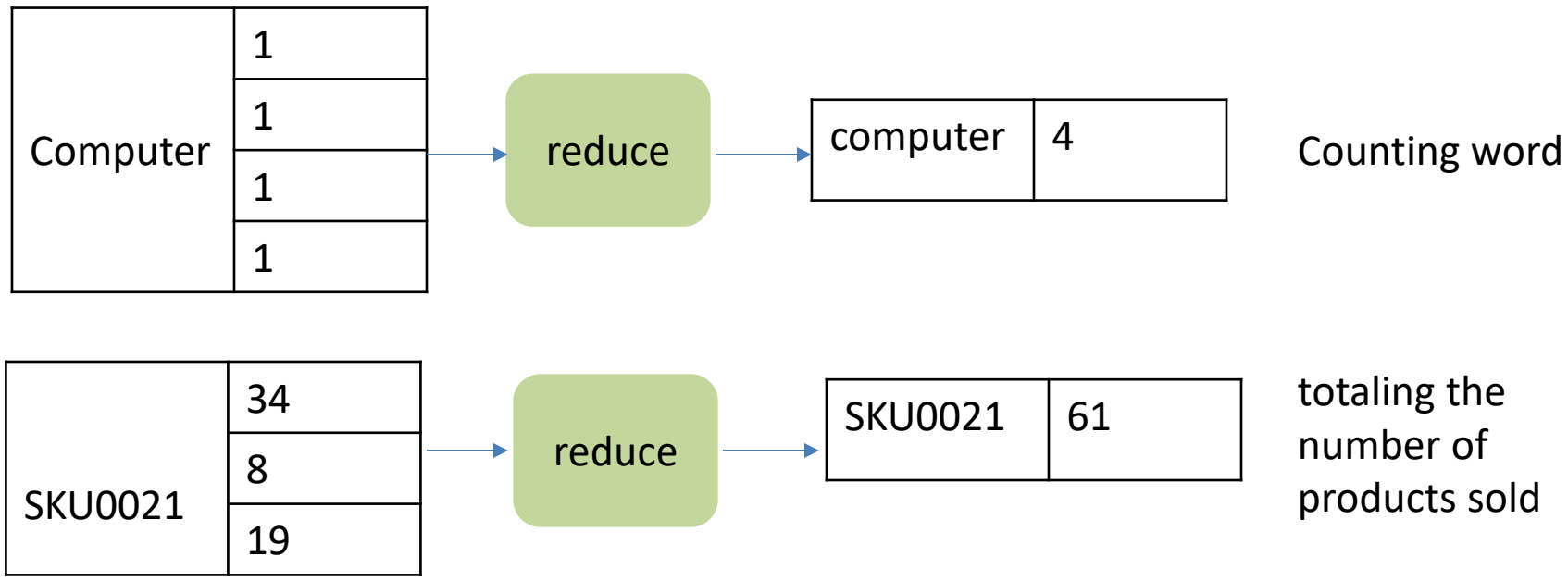
- The Reducer outputs final key/value pairs

  - In practice, usually outputs a single key/value pair for each input key

  - The output of reducer is written to HDFS

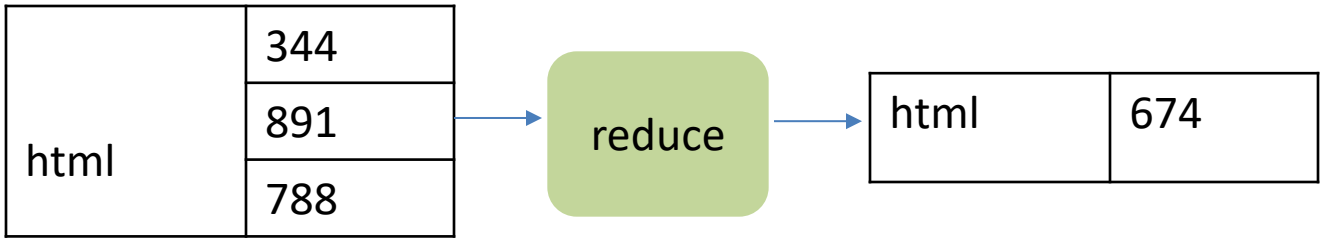
Example Reducer 1: Sum Reducer

- Add up all the values associated with each intermediate key



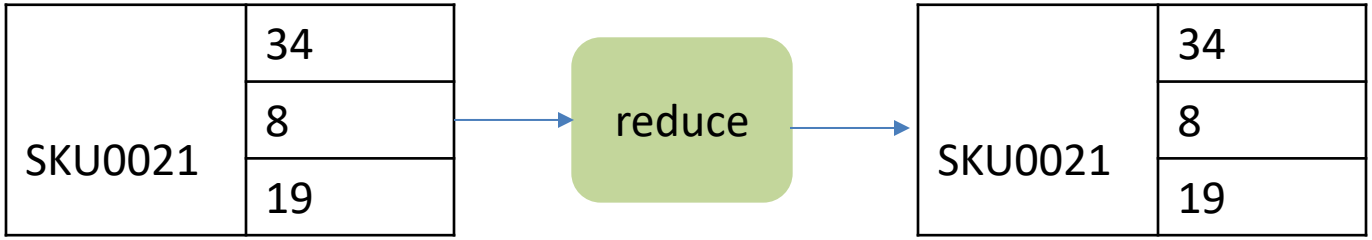
Example Reducer 2: Average Reducer

- Find the mean of all the values associated with each intermediate key



Example Reducer 3: Identity Reducer

- Output of reducer is the same as the input



- Why might an identity reducer be useful?
  -- For example, you can use it to group words. Each occurrence of a word (across all input files) with all its associated values will be grouped together and passed to a single Reducer, and written to a single output file.

- The identity reducer is the default reducer. If you create a job and don't specify a reducer, the identity reducer will be used.

Creating and Running a MapReduce Job

- Write the Mapper class
- Write the Reducer class
- Write a Driver class that configures the job and submits it to the cluster
- Compile the Mapper, Reducer, and Driver classes

```
#javac  -classpath   `hadoop classpath`  MyMapper.java
MyReducer.java  MyDriver.java
```

Creating and Running a MapReduce Job (Cont.)

- Create a jar file with the Mapper, Reducer, and Driver classes

```
#jar cvf  MyMR.jar MyMapper.class  MyReducer.class   MyDriver.class
```

- Run the *hadoop jar* command to submit the job to the Hadoop cluster

```
#hadoop jar MyMR.jar  MyDriver  in_file  out_dir
```

In-class Exercise: Running a mapreduce job