Unit-3

Part-3

Map Reduce Types And Formats

Map Reduce Types

- ✓ These types are used to represent the input, intermediate, and output data of the MapReduce job.

The following are the main types used in MapReduce:

Input Key And Value Types: These types represent the **Input Data To The MapReduce Job.**

- ✓ The key type is typically a unique identifier for each input record, and the value type is the data associated with the record.

Intermediate Key And Value Types: These types represent the **Output Of The Mapper And Input To The Reducer**.

- ✓ The intermediate key and value types can be different from the input key and value types.

Output Key And Value Types: These types represent the **Output Of The Reducer**.

- ✓ The output key and value types can be different from the input and intermediate key and value types.

Property	Job setter method	Input types		Intermediate types		Output types	
		K1	V1	K2	V2	K3	V3
Properties for configuring types:							
mapreduce.job.inputformat.class	setInputFormatClass()	•	•				
mapreduce.map.output.key.class	setMapOutputKeyClass()			•			
mapreduce.map.output.value.class	setMapOutputValueClass()				•		
mapreduce.job.output.key.class	setOutputKeyClass()					•	
mapreduce.job.output.value.class	setOutputValueClass()						•
Properties that must be consistent with the types:							
mapreduce.job.map.class	setMapperClass()	•	•	•	•		
mapreduce.job.combine.class	setCombinerClass()			•	•		
mapreduce.job.partitioner.class	setPartitionerClass()			•	•		
mapreduce.job.output.key.comparator.class	setSortComparatorClass()			•			
mapreduce.job.output.group.comparator.class	setGroupingComparatorClass()			•			
mapreduce.job.reduce.class	setReducerClass()			•	•	•	•
mapreduce.job.outputformat.class	setOutputFormatClass()					•	•

Configuration of Map Reduce types in new API

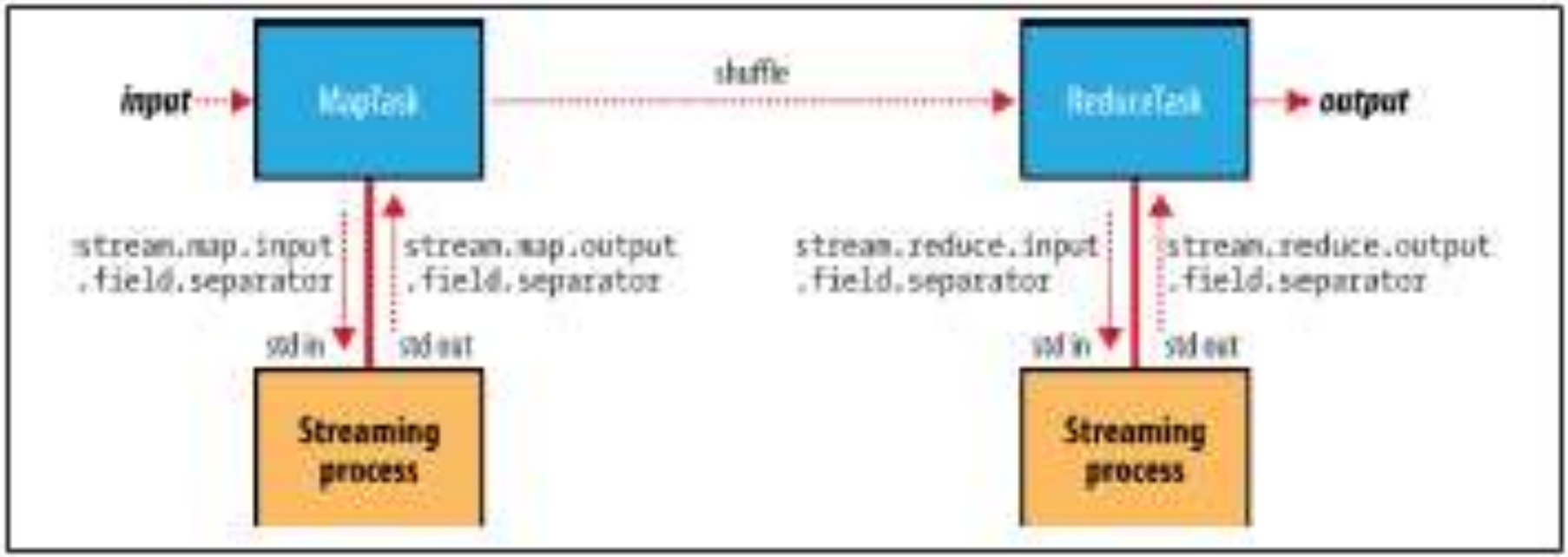
Property	JobConf setter method	Input types		Intermediate types		Output types	
		K1	V1	K2	V2	K3	V3
Properties for configuring types:							
mapred.input.format.class	setInputFormat()	•	•				
mapred.mapoutput.key.class	setMapOutputKeyClass()			•			
mapred.mapoutput.value.class	setMapOutputValueClass()				•		
mapred.output.key.class	setOutputKeyClass()					•	
mapred.output.value.class	setOutputValueClass()						•
Properties that must be consistent with the types:							
mapred.mapper.class	setMapperClass()	•	•	•	•		
mapred.map.runner.class	setMapRunnerClass()	•	•	•	•		
mapred.combiner.class	setCombinerClass()			•	•		
mapred.partitioner.class	setPartitionerClass()			•	•		
mapred.output.key.comparator.class	setOutputKeyComparatorClass()			•			
mapred.output.value.groupfn.class	setOutputValueGroupingComparator()			•			
mapred.reducer.class	setReducerClass()			•	•	•	•
mapred.output.format.class	setOutputFormat()					•	•

Configuration of Map Reduce types in old API

The Default Map Reduce Job

Property name	Type	Default value	Description
stream.map.input.field.separator	String	\t	The separator to use when passing the input key and value strings to the stream map process as a stream of bytes
stream.map.output.field.separator	String	\t	The separator to use when splitting the output from the stream map process into key and value strings for the map output
stream.num.map.output.key.fields	int	1	The number of fields separated by stream.map.output.field.separator to treat as the map output key
stream.reduce.input.field.separator	String	\t	The separator to use when passing the input key and value strings to the stream reduce process as a stream of bytes
stream.reduce.output.field.separator	String	\t	The separator to use when splitting the output from the stream reduce process into key and value strings for the final reduce output
stream.num.reduce.output.key.fields	int	1	The number of fields separated by stream.reduce.output.field.separator to treat as the reduce output key

Streaming Separator Properties



Separators are used in a streaming Map Reduce Job

Map Reduce Types

MapReduce is a programming model and processing framework that can be categorized into different types based on variations and use cases.

Here are several types of MapReduce:

Batch MapReduce:

- ✓ This is the traditional form of MapReduce and is used for batch processing of large datasets.

Streaming MapReduce:

- ✓ Streaming MapReduce extends the MapReduce model to handle real-time data processing.

Recursive MapReduce:-

- ✓ Recursive MapReduce is used for problems that involve recursive subtasks.

Incremental MapReduce:-

- ✓ Incremental MapReduce processes only the new or changed data since the last run.

Multi-Step MapReduce:-

- ✓ Some complex data processing tasks require multiple MapReduce jobs in sequence.

Hybrid MapReduce:-

- ✓ Hybrid MapReduce combines the MapReduce model with other processing models, such as in-memory processing or data streaming.

MapReduce on Cloud Services:-

- ✓ Cloud-based MapReduce services, like Amazon EMR and Google Data prep, provide MapReduce capabilities in a cloud environment.
- ✓ They offer scalability, flexibility, and ease of use for running MapReduce jobs on cloud infrastructure.

Iterative Map Reduce: Continue Until One condition becomes false

- ✓ Some algorithms, like graph processing or machine learning algorithms, require multiple iterations over the same dataset.

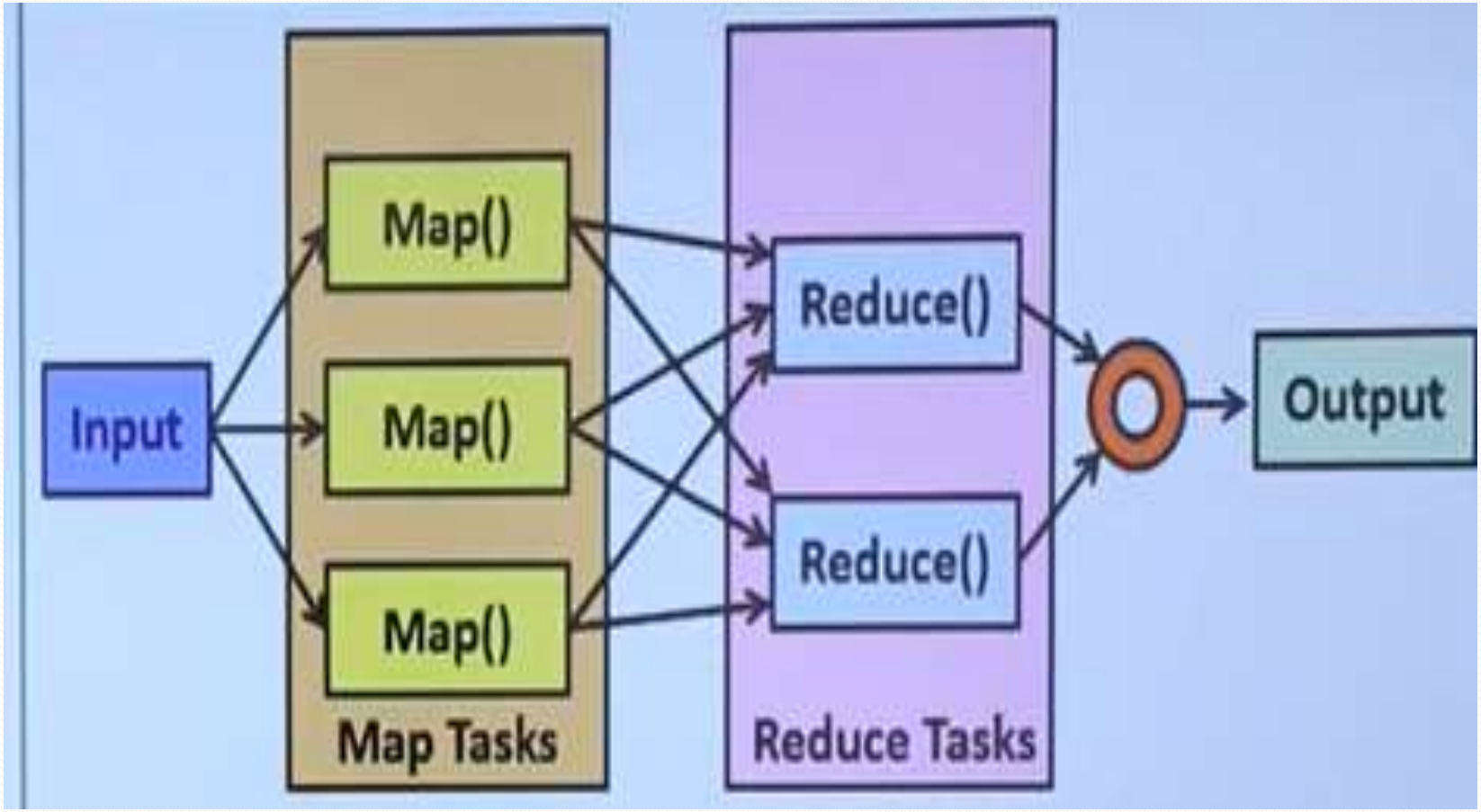
MapReduce types are the **Data Types** that are used for the input and output of the map and reduce functions.

The most common MapReduce types are:

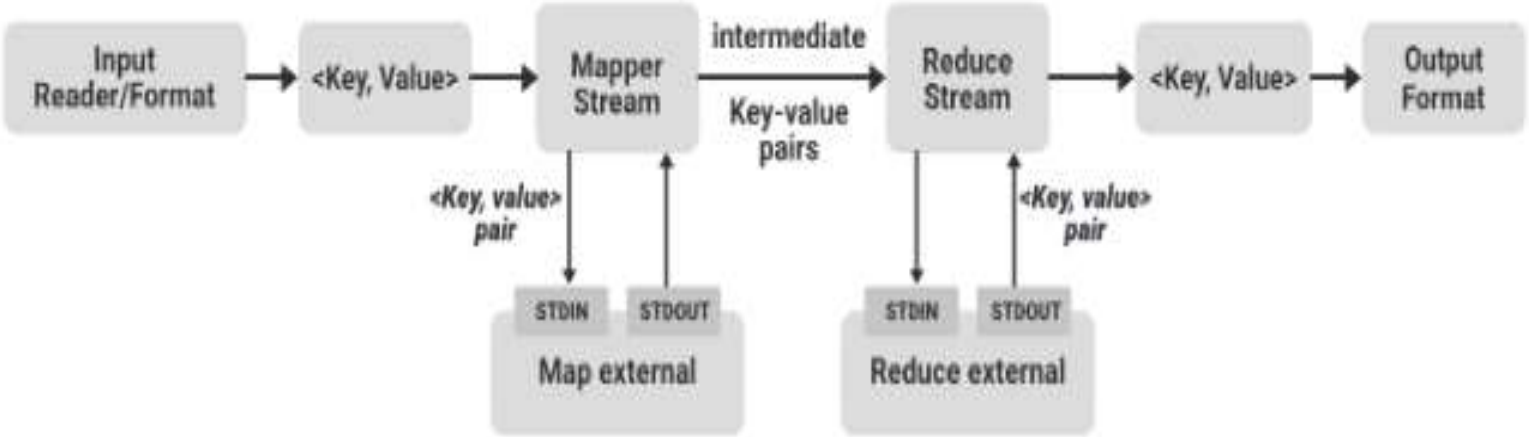
- ✓ **Text**: This type is used for text data, such as log files, web pages, and email messages.
- ✓ **Int**: This type is used for integer data, such as product IDs, customer counts, and transaction amounts.
- ✓ **Long**: This type is used for long integer data, such as timestamps and file sizes.
- ✓ **Float**: This type is used for floating-point data, such as product prices and customer ratings.
- ✓ **Double**: This type is used for double-precision floating-point data, such as scientific data and financial data.

MapReduce also supports custom types, which can be used for more complex data structures, such as JSON objects and XML documents.

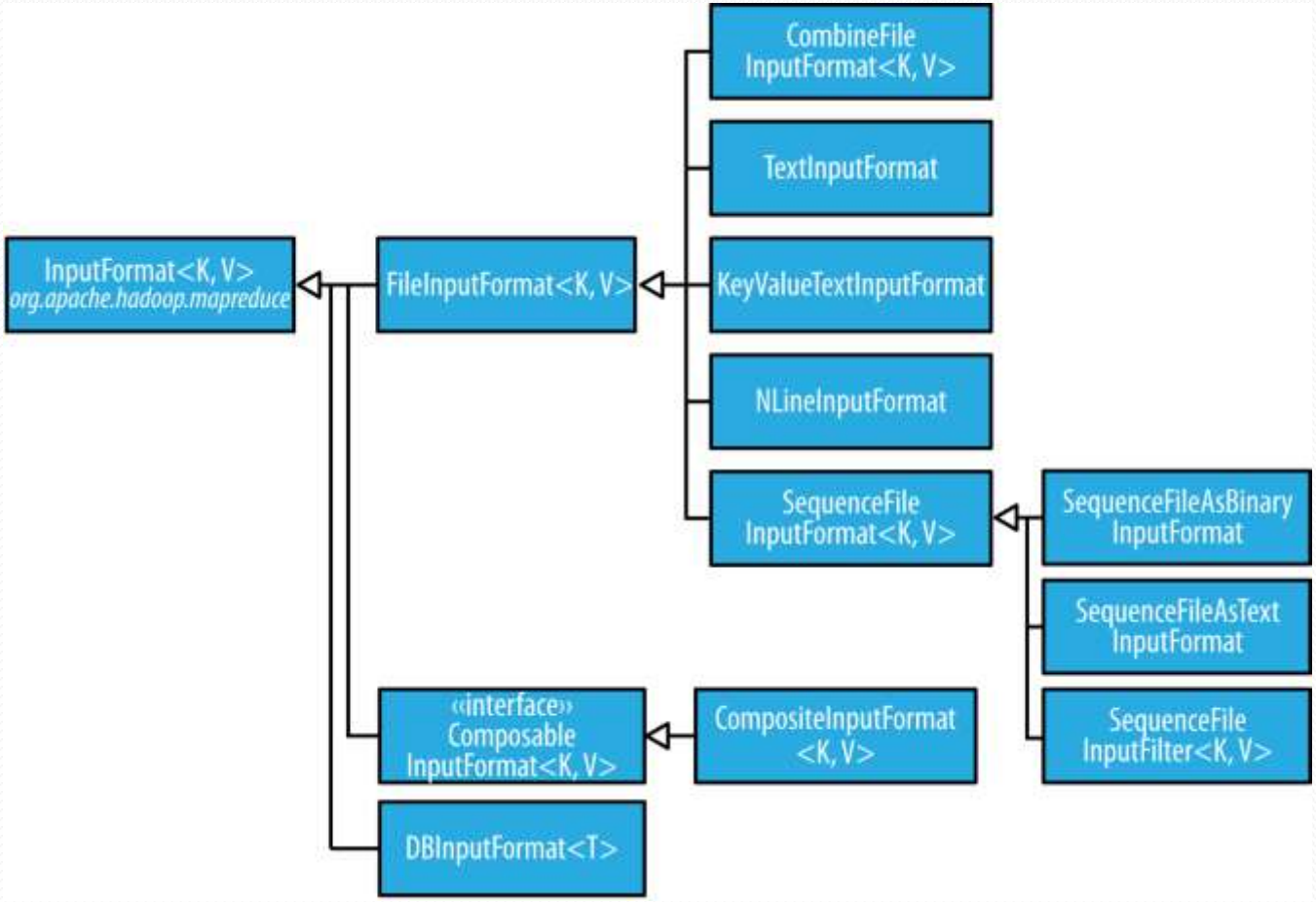
Map Reduce Working Mechanism



Map Reduce


Input Reader/Format
<Key, Value>
Mapper Stream
intermediate
Key-value pairs
Reduce Stream
<Key, Value>
Output Format
<Key, value> pair
<Key, value> pair
STDIN
STDOUT
Map external
STDIN
STDOUT
Reduce external

Input Format Class Hierarchy



Map Reduce - Input Formats

- In Hadoop MapReduce and similar distributed data processing frameworks, input data is typically read from various sources and formats.
- The choice of input format depends on the structure of the data and the requirements of the processing job.

Here are some common input formats used in Map Reduce:-

Text Input

1. Text Input Format:

- ✓ This is the default input format in Hadoop Map Reduce.

2. Key-Value Text Input Format:

- ✓ This format is similar to TextInputFormat but interprets each line as a key-value pair separated by a Tab Or A Comma.

3. NLine Input Format

4. XML

Binary Input:-

1. Sequence-File Input Format:

- ✓ Sequence files are **Binary Files** used to store key-value pairs efficiently.
- ✓ It allows for efficient **Serialization And Deserialization Of Complex Data Structures**.

2. Sequence File As Text Input Format

3. Sequence File As Binary Input Format

4. Fixed Length Input Format

Multiple Input:-

- ✓ Same Data in Different Formats

DB Input Format:-

- ✓ DB Input Format allows Map-Reduce jobs to read data **Directly From Relational Databases**.

Input Splits & Records:-

1. **FileInputFormat** - Base Class For All Implementations
2. **FileInputFormat Input Paths –** Offers 4 Static convenience methods for setting job's input paths
3. **FileInputFormat Input Splits –** Splits Only files larger than HDFS Blocks.
4. **Small Files & CombineFileInputFormat -**
5. **NonSplitTable TextInputFormat –** Some applications don't want the files to split
6. **WholeFileInputFormat** – Reading whole file as a record

Text-Input Format with Custom Record-Reader:

- ✓ You can create custom Record-Readers to parse and **Process Input Data In Specific Formats.**
- ✓ This approach is flexible and allows you to handle **Non-standard Data Formats** effectively.

Minimum split size	Maximum split size	Block size	Split size	Comment
1 (default)	Long.MAX_VALUE (default)	128 MB (default)	128 MB	By default, the split size is the same as the default block size.
1 (default)	Long.MAX_VALUE (default)	256 MB	256 MB	The most natural way to increase the split size is to have larger blocks in HDFS, either by setting `dfs.blocksize` or by configuring this on a per-file basis at file construction time.
256 MB	Long.MAX_VALUE (default)	128 MB (default)	256 MB	Making the minimum split size greater than the block size increases the split size, but at the cost of locality.
1 (default)	64 MB	128 MB (default)	64 MB	Making the maximum split size less than the block size decreases the split size.

Examples of how to control the split size

FileSplit method	Property name	Type	Description
`getPath()`	`mapreduce.map.input.file`	`Path/String`	The path of the input file being processed
`getStart()`	`mapreduce.map.input.start`	`long`	The byte offset of the start of the split from the beginning of the file
`getLength()`	`mapreduce.map.input.length`	`long`	The length of the split in bytes

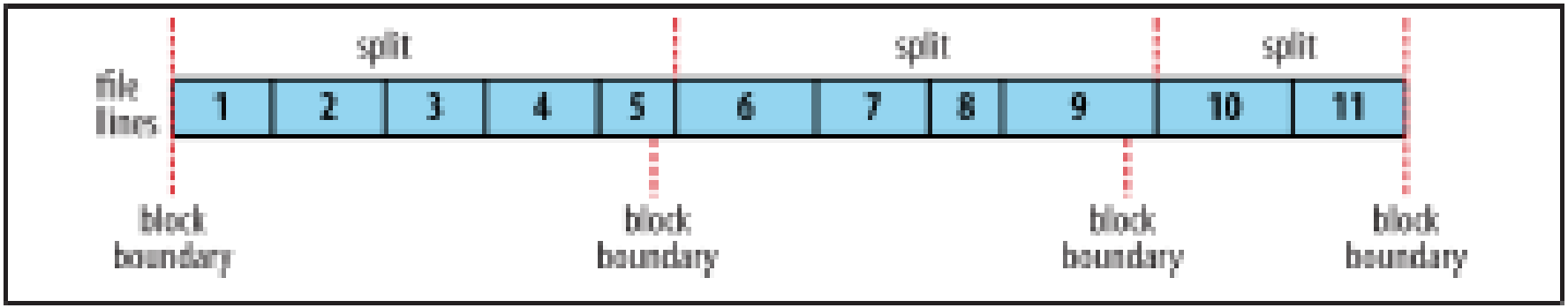
File Split Properties

Property name	Type	Default value	Description
mapreduce.input.fileinputformat.inputdir	Comma-separated paths	None	The input files for a job. Paths that contain commas should have those commas escaped by a backslash character. For example, the glob {a,b} would be escaped as {a\,b}.
mapreduce.input.pathfilter.class	PathFilter classname	None	The filter to apply to the input files for a job.

Input Path and Filter Properties

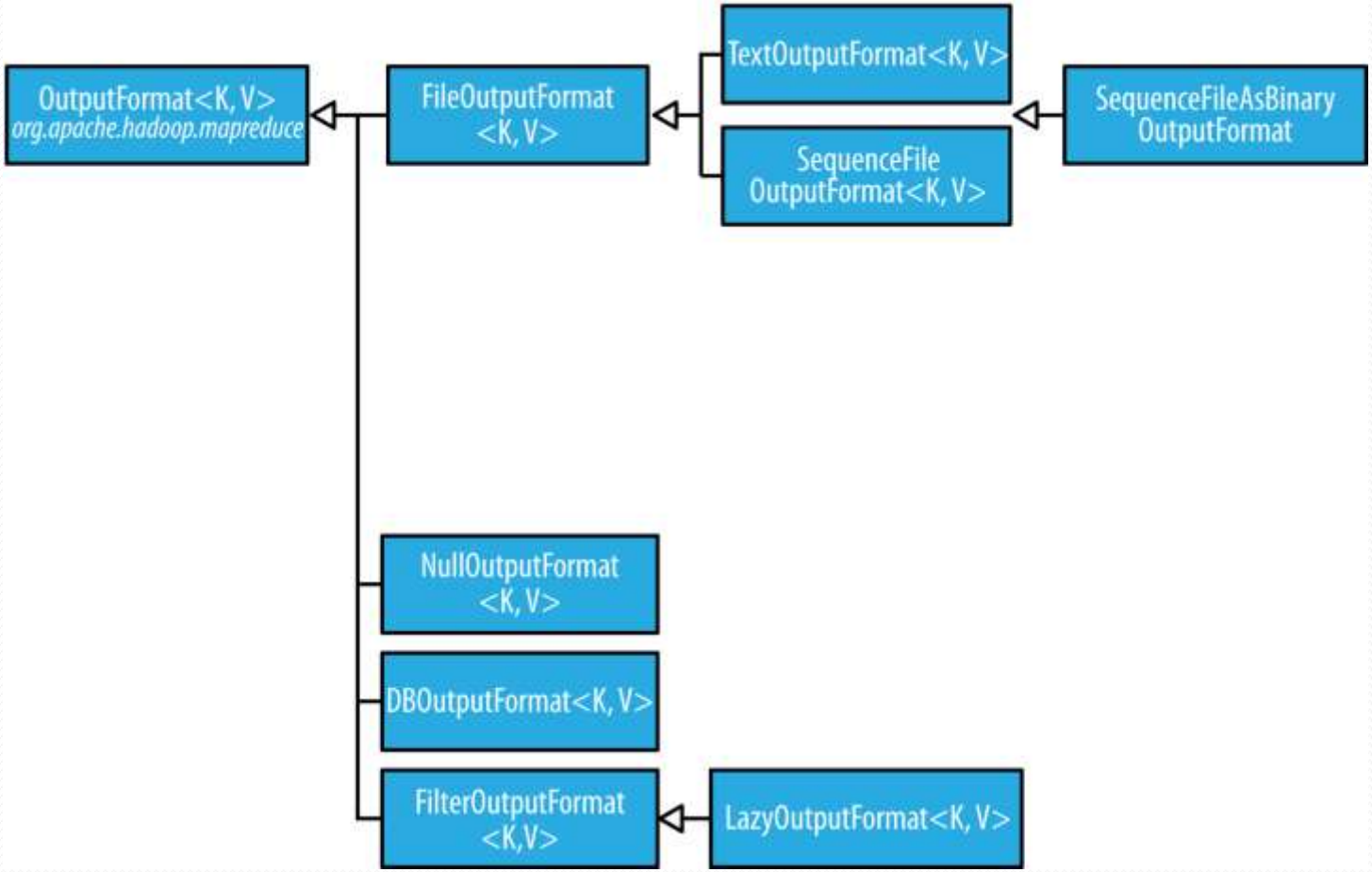
Property name	Type	Default value	Description
mapreduce.input.fileinputformat.split.minsize	int	1	The smallest valid size in bytes for a file split
mapreduce.input.fileinputformat.split.maxsize[a]	long	Long.MAX_VALUE (i.e., 9223372036854775807)	The largest valid size in bytes for a file split
dfs.blocksize	long	128 MB (i.e., 134217728)	The size of a block in HDFS in bytes

Properties for controlling split size



Logical records and HDFS blocks for TextInputFormat

Output Format Class Hierarchy



Map Reduce - Output formats

- The output of Map Reduce jobs can be stored in various formats to suit different use cases and requirements.

Here are some common output formats used in MapReduce:

Text Output Format:-

- ✓ This is the **Default Output Format** in Hadoop Map Reduce.

Binary Output:-

1. **SequenceFile Output Format**:-
   - ✓Sequence files are **Binary Files** used to store key-value pairs efficiently.

2. SequenceFile As Binary Output Format

3. Map File Output Format

Multiple Outputs:-

- ✓ It's useful when you want to produce **Multiple Outputs** with **Different Formats** or to separate data based on criteria.

Database Output (e.g., DB Output Format):-

- ✓ MapReduce jobs can write **Output Data Directly To Relational Databases** using output formats like DB Output Format.

Lazy Output:-

- ✓ Some application prefer that empty files not to be created

Custom Output Formats:-