Hadoop/MapReduce: Phases + System

Readings : Hadoop - The Definitive Guide (Chapters 2, 8, 9, 16)

Open-Box versus Black-Box Model

- **DBMSs have an Open-Box Model**
  - Data are known (DB Schema)
  - Queries are known (written in SQL)

- **Hadoop has a Black-Box Model**
  - Data are not known (files of unknown structure)
  - Jobs are also unknown (written in java)

Hadoop offers very limited optimizations

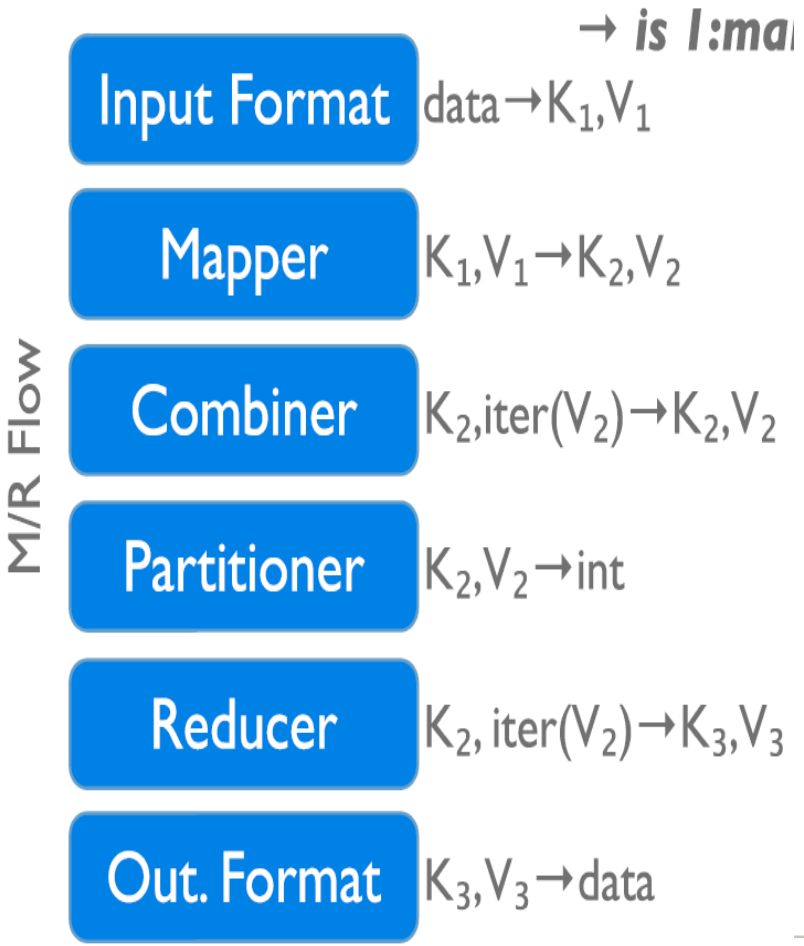
Simplicity Comes From…

- Jobs are read-only (do not change or modify actual data)
- Intermediate data between mappers & reducers is materialized until job finishes

More About Execution Phases

Execution Phases

- InputFormat
- Map function
- Partitioner
- Sorting & Merging
- Combiner
- Shuffling
- Merging
- Reduce function
- OutputFormat



Partitioners

- **The output of the mappers need to be partitioned**
  - # of partitions = # of reducers
  - **The same key in all mappers must go to the same partition (and hence the same reducer)**
- **Default partitioning is hash-based**
- **Users could customize this, if desired**

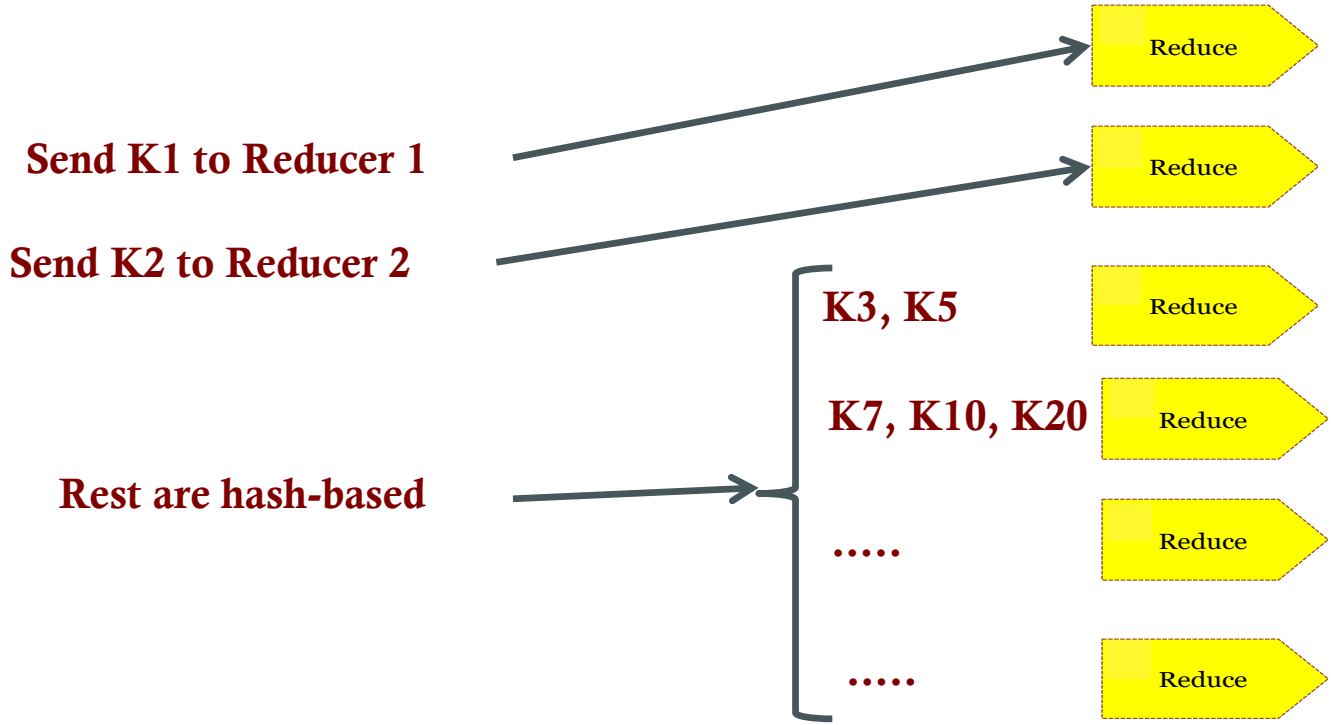
Customized Partitioner

```
1  package org.apache.hadoop.examples.textpair;
2
3  import org.apache.hadoop.io.Writable;
4  import org.apache.hadoop.mapred.JobConf;
5  import org.apache.hadoop.mapred.Partitioner;
6
7  /**
8   * the hash partitioner
9   *
10  * @author yingyib
11  *
12  */
13 public class FirstPartitioner implements Partitioner<TextPair, Writable> {
14
15         @Override
16         public void configure(JobConf job) {
17         }
18
19         @Override
20         public int getPartition(TextPair key, Writable value, int numPartitions) {
21                 return Math.abs(key.getFirst().hashCode()) % numPartitions;
22         }
23 }
```

Returns a partition Id

Optimization: Balance Load among Reducers

- Assume we have N reducers but many more keys {K1, K2, …, Km}
- Distribution could be skewed, e.g., assume K1 and K2 have many records

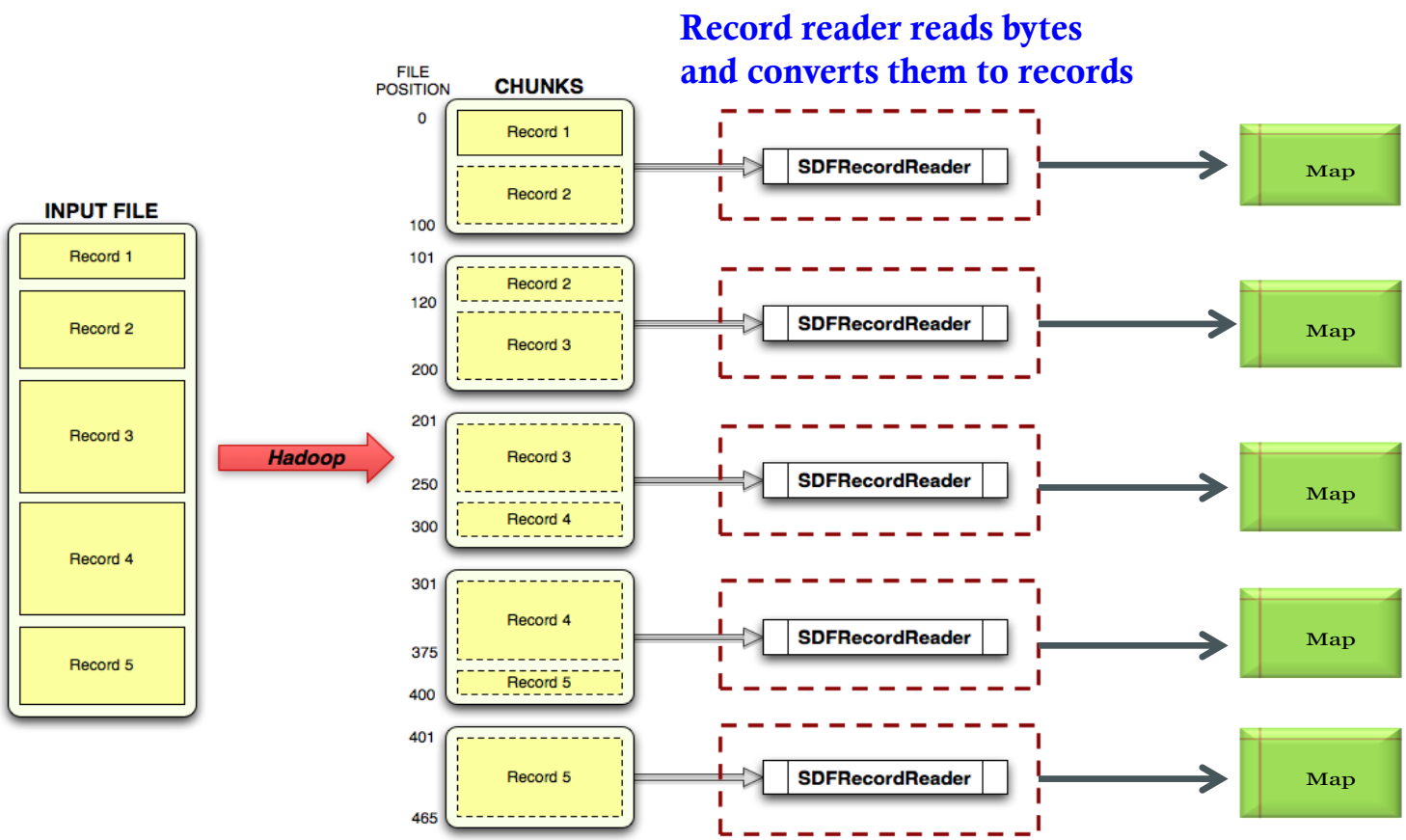


Input/Output Formats

- **Hadoop's data model ← Any data in any format will fit**
  - Text or binary in a certain structure
- **How does Hadoop understand and read data ?**
- ***"Input format" class corresponds to* code that understands how to read data of a particular built-in format:**
  - Hadoop has built-in input formats to make use of
  - Most common are Text files and Binary sequence files

Input Formats

Module that constructs a ***record***, and passes it to a mapper



HDFS Blocks & Splits

- ***HDFS Block*** is a physical thing re "disk"
  - blind to record structure
- ***Split*** is a logical concept: Start offset & End offset
- Split can be one block, portion of block, or multiple blocks
  - Splits aligns with the record structure

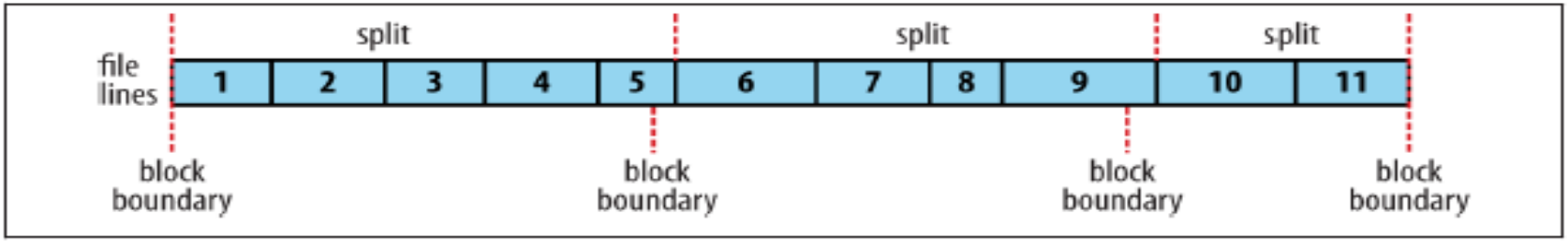


Figure 8-3. Logical records and HDFS blocks for TextInputFormat

(Hadoop: The Definitive Guide)

Example

On the top of the Crumpetty Tree
The Quangle Wangle sat,
But his face you could not see,
On account of his Beaver Hat.
However, the offset within the file is known
The records are interpreted

HDFS Blocks

On the top of the Crumpetty Tree\nThe Quangle Wangle sat,\nBut his face you could

not see,**n**On account of his Beaver Hat.**n**However, the offset within the file is Known

The records are interpreted\n

Example

On the top of the Crumpetty Tree
The Quangle Wangle sat,
But his face you could not see,
On account of his Beaver Hat.
However, the offset within the file is known
The records are interpreted

Splits

On the top of the Crumpetty Tree\nThe Quangle Wangle sat,\nBut his face you could

not see,**n**On account of his Beaver Hat.**n**However, the offset within the file is Known\n

The records are interpreted\n

Tell Hadoop which Input/Output Formats

File Edit Options Buffers Tools Java Help

```
public class WordCount {

  public static class Map extends MapReduceBase implements
                Mapper<LongWritable, Text, Text, IntWritable> {
    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();

    public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable>
                    output, Reporter reporter) throws IOException {
      String line = value.toString();
      StringTokenizer tokenizer = new StringTokenizer(line);
      while (tokenizer.hasMoreTokens()) {
        word.set(tokenizer.nextToken());
        output.collect(word, one);
  }}}

  public static class Reduce extends MapReduceBase implements
                Reducer<Text, IntWritable, Text, IntWritable> {
    public void reduce(Text key, Iterator<IntWritable> values, OutputCollector<Text,
                       IntWritable> output, Reporter reporter) throws IOException {
      int sum = 0;
      while (values.hasNext()) { sum += values.next().get(); }
      output.collect(key, new IntWritable(sum));
  }}

  public static void main(String[] args) throws Exception {
    JobConf conf = new JobConf(WordCount.class);
    conf.setJobName("wordcount");
    conf.setOutputKeyClass(Text.class);
    conf.setOutputValueClass(IntWritable.class);
    conf.setMapperClass(Map.class);
    conf.setCombinerClass(Reduce.class);
    conf.setReducerClass(Reduce.class);
    conf.setInputFormat(TextInputFormat.class);
    conf.setOutputFormat(TextOutputFormat.class);
    FileInputFormat.setInputPaths(conf, new Path(args[0]));
    FileOutputFormat.setOutputPath(conf, new Path(args[1]));

    JobClient.runJob(conf);
  }}
```

--:--- **mapreduce.java** All L9 (Java/l Abbrev)----------

Wrote /home/shivnath/Desktop/mapreduce.java

Define the formats

All Execution Phases

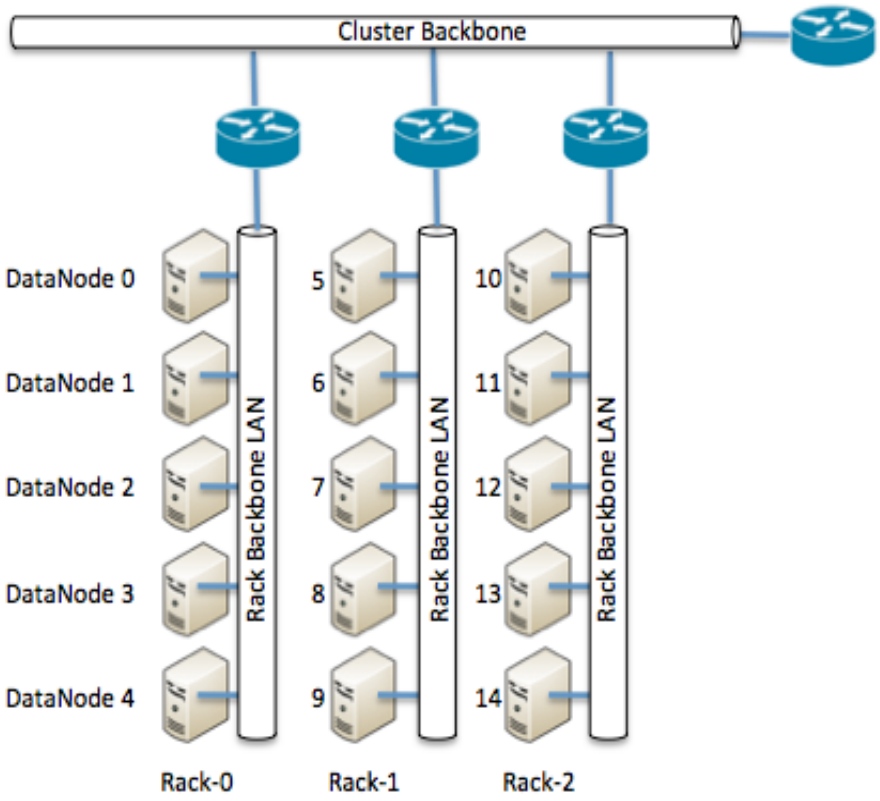
- **InputFormat**
- **Map function**
- **Partitioner**
- **Sorting & Merging**
- **Combiner**
- **Shuffling**
- **Merging**
- **Reduce function**
- **OutputFormat**

More on HDFS

HDFS and Placement Policy

Default Placement Policy

- ***First copy*** is written to the node creating the file
- ***Second copy*** is written to a data node within same rack
- ***Third copy*** is written to a data node in different rack
- **Objective: Load balancing & fault tolerance**



Rack-aware replica placement

Safemode Startup

- On startup, NameNode enters Safemode (few seconds).
- Each DataNode checks in with Heartbeat and BlockReport.
- NameNode verifies each block has acceptable # of replicas

- If things are fine ➔ NameNode exits Safemode
- If some blocks are under replicated
  - Replicate these blocks to other DataNodes
  - Then, exit Safemode

Communication Protocols

- HDFS communication protocols layered on top of TCP/IP protocol
- A client establishes a connection to a configurable TCP port on the NameNode machine using "ClientProtocol"
- A Datanode talks to the Namenode using the "Datanode" protocol
- **File transfers between nodes ?**
  - **Directly between DataNodes**
  - **Does not go though the NameNode**

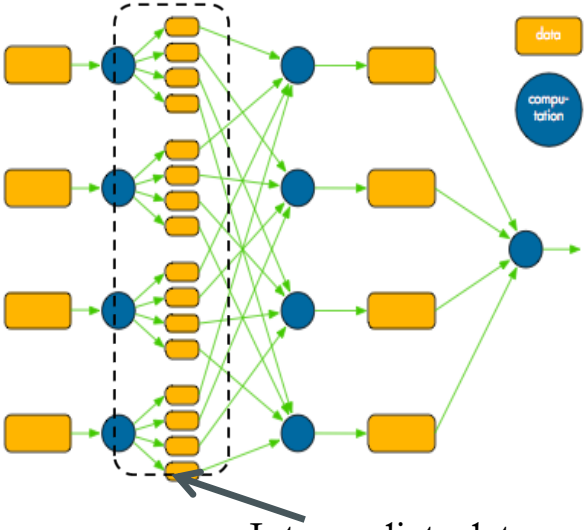
Hadoop Fault Tolerance

- **Intermediate data between mappers and reducers are *materialized* for achieving simple & straightforward fault tolerance**

- **What if a task fails (map or reduce)?**
  - Tasktracker detects the failure
  - Sends message to the jobtracker
  - Jobtracker re-schedules the task

- **What if a datanode fails?**
  - Both namenode and jobtracker detect the failure
  - All tasks on the failed node are re-scheduled
  - Namenode replicates the users' data to another node

- **What if a namenode or jobtracker fails?**
  - The entire cluster is down



Intermediate data
(materialized)

Simplicity Comes From…

- Jobs are read-only (do not change or modify actual data)
- Intermediate data between mappers & reducers is materialized until job finishes

Configuration

- Several files control Hadoop's cluster configurations
  - **Mapred-site.xml:** map-reduce parameters
  - **Hdfs-site.xml:** HDFS parameters
  - **Masters:** Which node(s) are the masters
  - **Slaves: W**hich nodes are the slaves
- Hadoop has around 190 parameters
  - Mostly 10-20 are the used

Web Interface

HDFS Interface

NameNode ' mach1:8000 '

Started: Fri Aug 22 10:50:32 GMT 2008
Version: 0.17.1.r81212
Compiled: Mon Aug 11 03:42:21 GMT 2008 by rs3e24
Upgrades: There are no updates in progress

Applications Places Sun Jun 10, 10:43 AM T V Ganesh

localhost Hadoop Map/Re... | Running Hadoop On Ubun... | Blogger: Giga thoughts... | My Stats — WordPress.co...

localhost:50030/jobtracker.jsp

Gmail Facebook Welcome, Tinniam ... Blogger: Dashboard

localhost Hadoop Map/Reduce Ad

State: RUNNING
Started: Sun Jun 10 10:10:10 IST 2012
Version: 1.0.3, r1335192
Compiled: Tue May 8 20:16:59 UTC 2012 by hortonfo
Identifier: 201206101010

Cluster Summary (Heap Size is 15.06 MB/989.

Running Map Tasks	Running Reduce Tasks	Total Submissions	Nodes	Occupied Map Slots	Occupied Reduce Slots
0	0	2	1	0	0

Scheduling Information

Queue Name	State	Scheduling Information
default	running	N/A

Filter (Jobid, Priority, User, Name)
Example: 'user:smith 3200' will filter by 'smith' only in the user field and '3200' in all fields

Running Jobs

none

Completed Jobs

Jobid	Priority	User	Name	Map % Complete	M... To...
job_201206101010_0003	NORMAL	root	grep-search	100.00%	9

Hadoop job_200709211549_0003 on localhost

User: hadoop
Job Name: streamjob34453.jar
Job File: /usr/local/hadoop-datastore/hadoop-hadoop/mapred/system/job_200709211549_0003/job.xml
Status: Succeeded
Started at : Fri Sep 21 16:07:10 CEST 2007
Finished at: Fri Sep 21 16:07:26 CEST 2007
Finished in: 16sec

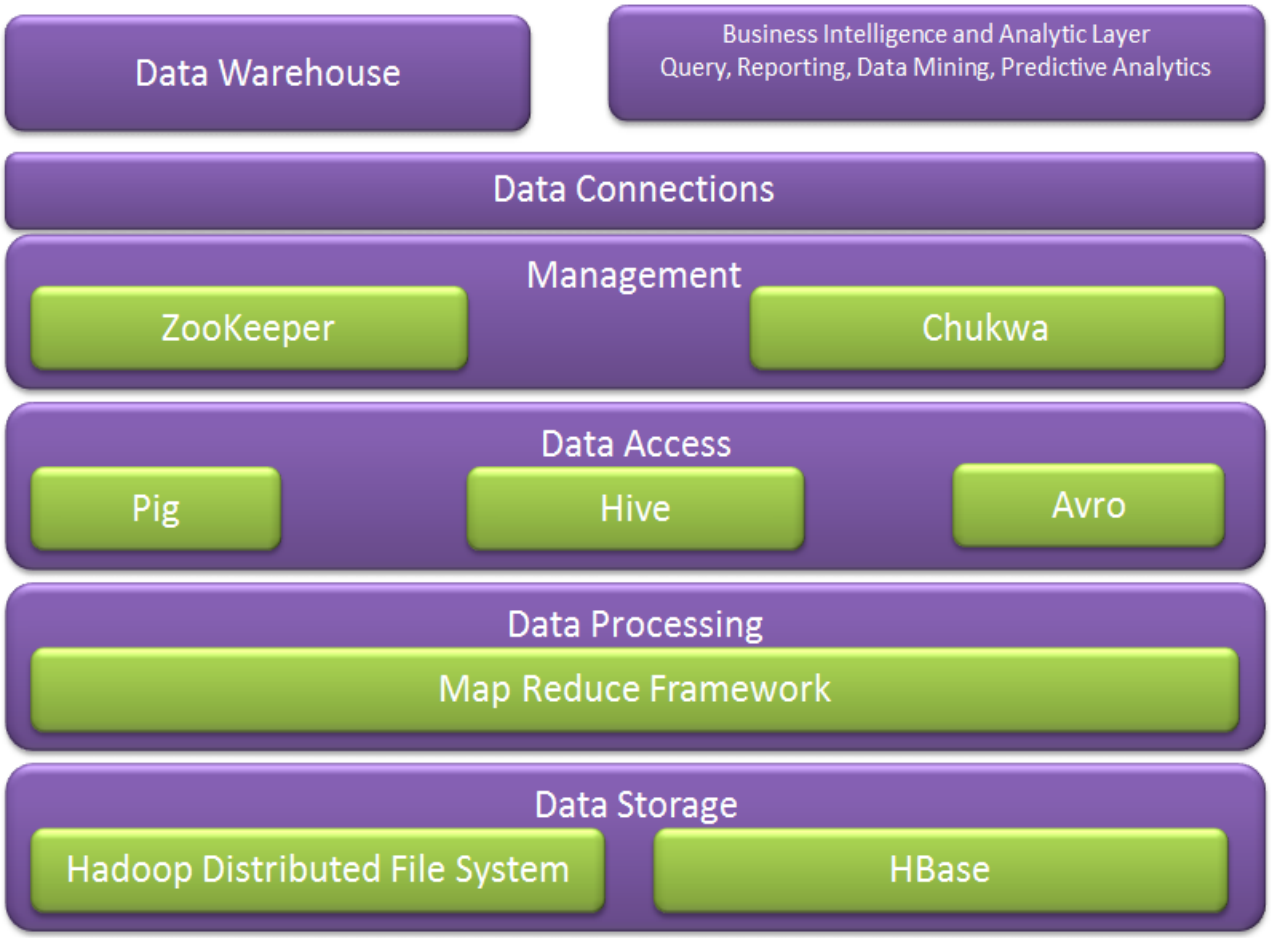
Kind	% Complete	Num Tasks	Pending	Running	Complete	Killed	Failed/Killed Task Attempts
map	100.00%	3	0	0	3	0	0 / 0
reduce	100.00%	1	0	0	1	0	0 / 0

	Counter	Map	Reduce	Total
Job Counters	Launched map tasks	0	0	3
	Launched reduce tasks	0	0	1
	Data-local map tasks	0	0	3
Map-Reduce Framework	Map input records	77,637	0	77,637
	Map output records	103,909	0	103,909
	Map input bytes	3,659,910	0	3,659,910
	Map output bytes	1,083,767	0	1,083,767
	Reduce input groups	0	85,095	85,095
	Reduce input records	0	103,909	103,909
	Reduce output records	0	85,095	85,095

Change priority from NORMAL to: VERY_HIGH HIGH LOW VERY_LOW

localhost Hadoop Map... | tvganesh@localhost:/... | hadoop.doc - LibreOffi...

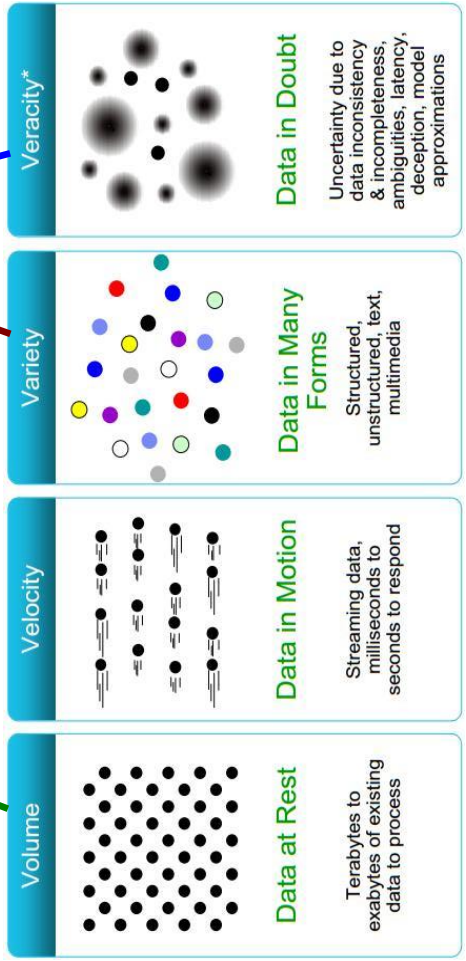
Hadoop Ecosystem



Recall…DBMS

- Data is nicely structured (known in advance)
- Data is correct & certain
- Data is relatively static
- Data is small-mid size

- Access pattern: Mix Read/Write
- Notion of transactions



What About Hadoop?

- Any structure will fit
- Data is correct & certain
- Data is static, <u>but scales to petabytes</u>

- Access pattern: <u>Read-Only</u>
- <u>Notion of jobs</u>

In Big Data: ***It is read only,***
No notion of transactions