Aurelius Corporate Solutions

Course- Big Data Hadoop (Basic)

Aurelius Corporate Solutions
A-125, Sector 63, Noida, Uttar Pradesh – 201307, India
1000+ Unique Technologies Projects Delivered | 500+ Corporate Customers Worldwide | 50000+ Professionals
Trained on 40+ Domains in Over 30 Countries

Aurelius™ CORPORATE SOLUTIONS

Course Topics

- ✓ Welcome to Big Data World
  - ✓ Understanding Big Data
  - ✓ Hadoop Architecture
- ✓ HDFS
  - ✓ Deep dive in HDFS Architecture
  - ✓ HDFS APIs
  - ✓ Introduction to HDP Sandbox
  - ✓ HDFS Hands – 1 Hour
- ✓ Introduction to YARN & MR
  - ✓ Hadoop MapReduce framework
  - ✓ Programming in Map Reduce
- ✓ Advance Map Reduce
  - ✓ Understanding Counters
  - ✓ Differences between MR1 & MR2
  - ✓ Introduction to MR API
  - ✓ Overview of Map Side Join
  - ✓ Overview of Reduce Side Join
  - ✓ Map Reduce Hands On – 1 hour
- ✓ Hive
  - ✓ Analytics using Hive
  - ✓ Understanding HIVE QL
- ✓ Advanced Hive
  - ✓ Advance Hive
  - ✓ Hive Hands On – 1 Hour
- ✓ NoSQL & HBase
  - ✓ CAP Theorem
  - ✓ NoSQL Databases and HBASE
  - ✓ HBase Architecture
  - ✓ HBase Schema Design
  - ✓ Difference between Hive & Hbase
  - ✓ Hbase Hands On – 1 Hour
- ✓ Apache Spark
  - ✓ Introduction to Spark
  - ✓ Why Spark?
  - ✓ Spark Stack Overview
  - ✓ Overview of RDD, Data Frame & Data Set
  - ✓ Spark Actions & Transformation Overview

Aurelius™ CORPORATE SOLUTIONS

Topics for Day 1

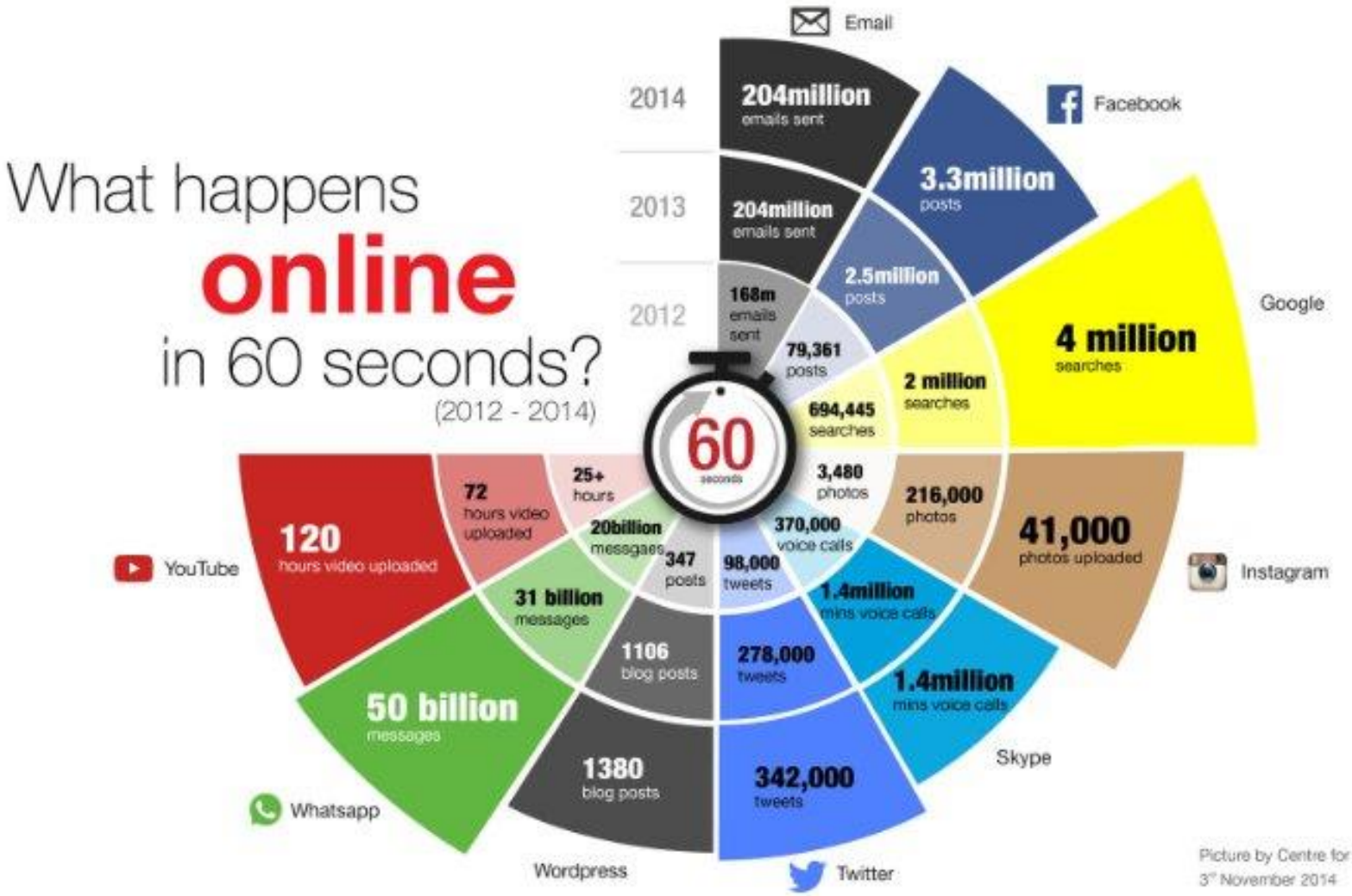
- ✓ Team Introduction
- ✓ Introduction to Big Data – Why and What?
- ✓ Characteristics of Big Data (4Vs)
- ✓ Overview of Big Data Ecosystem
- ✓ What is Hadoop?
- ✓ History of Hadoop

Tea Break

- ✓ Components of Hadoop
- ✓ Introduction to HDFS
- ✓ HDFS Architecture – Name Node / Data Node, Concept of Blocks
- ✓ File Formats in Hadoop
- ✓ HDFS API walk through
- ✓ Anatomy of a File Write and Read

Lunch Break

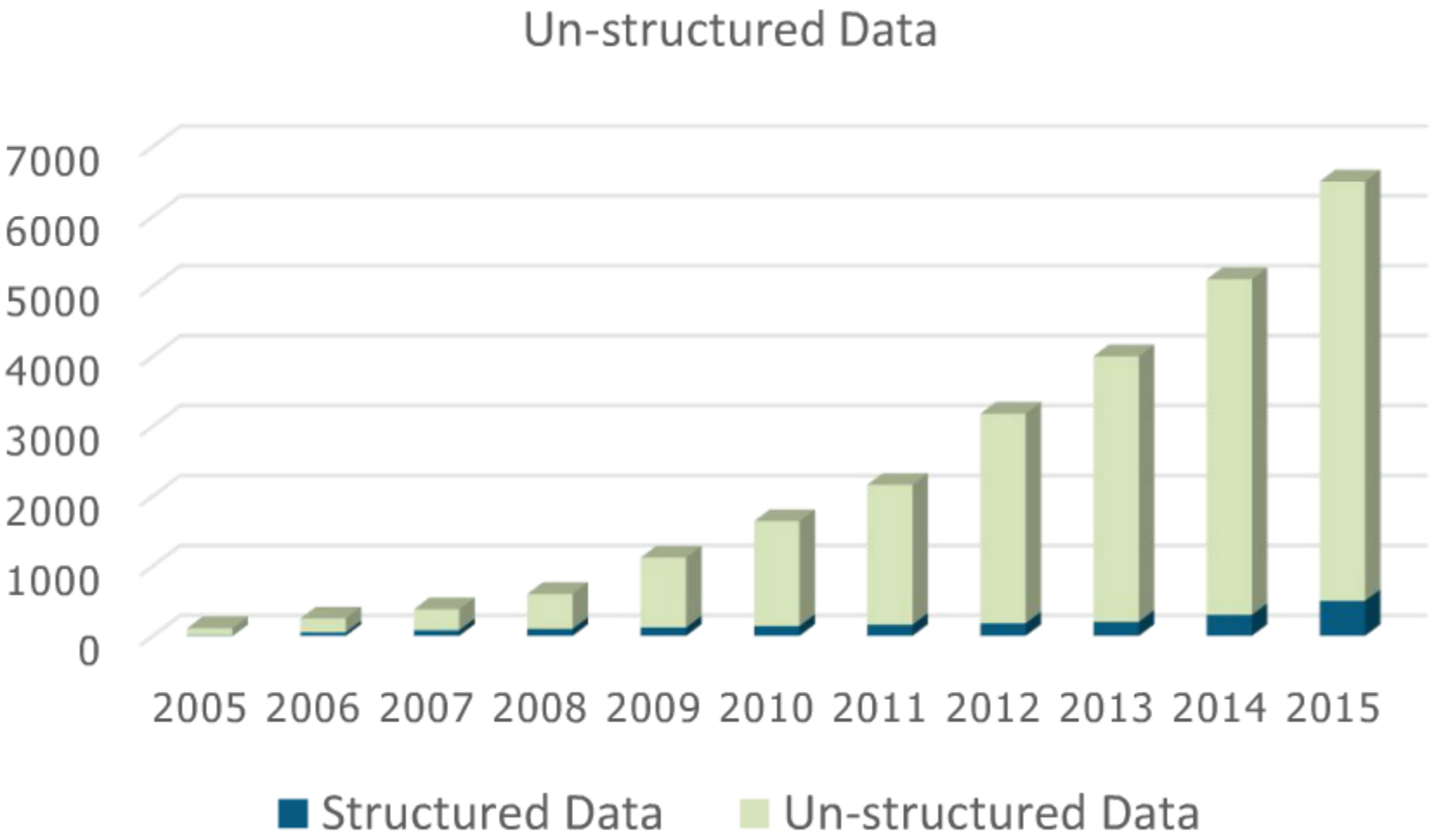
- ✓ Overview of Lab environment – HDP sandbox etc.
- ✓ HDFS Hands on – Getting Familiar with HDFS most commonly used commands
- ✓ Introduction to Map Reduce
- ✓ Map Reduce Phases – Map, Shuffle-Sort and Reduce
- ✓ Map Reduce Job Submission Flow



Picture by Centre for Learning and Teaching
3rd November 2014

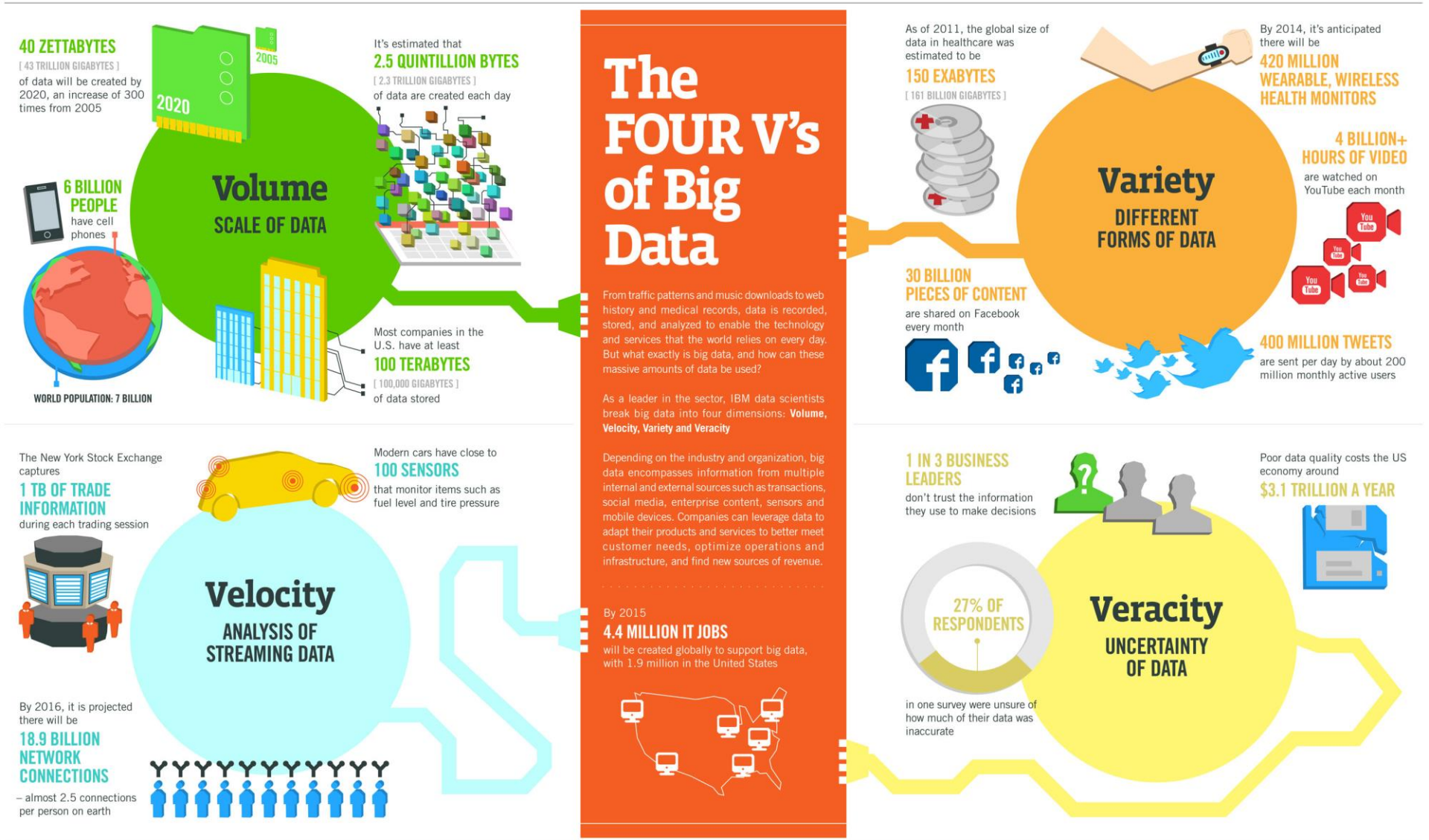
Aurelius™ CORPORATE SOLUTIONS

Un-Structured Data is Exploding



Aurelius™ CORPORATE SOLUTIONS

4V's of Big data



Sources: McKinsey Global Institute, Twitter, Cisco, Gartner, EMC, SAS, IBM, MEPTEC, QAS

Question

Map the following to corresponding data type:

- XML Files
- Word Docs, PDF files, Text files
- E-Mail body
- Data from Enterprise systems (ERP, CRM etc.)

Aurelius™ CORPORATE SOLUTIONS

Answer

XML Files **-> Semi-structured data**

Word Docs, PDF files, Text files -> **Unstructured Data**

E-Mail body -> **Unstructured Data**

Data from Enterprise systems (ERP, CRM etc.) -> **Structured Data**

Aurelius™ CORPORATE SOLUTIONS

Common Big Data Customer Scenarios

- ✓ Web and e-tailing
  - ✓ Recommendation Engines
  - ✓ Ad Targeting
  - ✓ Search Quality
  - ✓ Abuse and Click Fraud Detection

- ✓ Telecommunications
  - ✓ Customer Churn Prevention
  - ✓ Network Performance Optimization
  - ✓ Calling Data Record (CDR) Analysis
  - ✓ Analyzing Network to Predict Failure

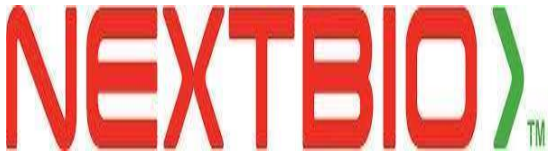
http://wiki.apache.org/hadoop/PoweredBy

Aurelius™ CORPORATE SOLUTIONS

Common Big Data Customer Scenarios (Contd.)

- ✓ Government
  - ✓ Fraud Detection And Cyber Security
  - ✓ Welfare schemes
  - ✓ Justice

- ✓ Healthcare & Life Sciences
  - ✓ Health information exchange
  - ✓ Gene sequencing
  - ✓ Serialization
  - ✓ Healthcare service quality improvements
  - ✓ Drug Safety



http://wiki.apache.org/hadoop/PoweredBy

Common Big Data Customer Scenarios (Contd.)

- ✓ Banks and Financial services
  - ✓ Modeling True Risk
  - ✓ Threat Analysis
  - ✓ Fraud Detection
  - ✓ Trade Surveillance
  - ✓ Credit Scoring And Analysis

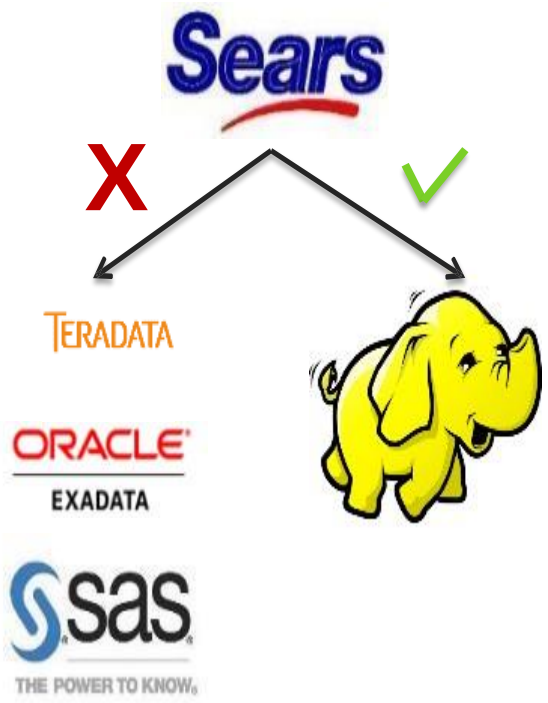
- ✓ Retail
  - ✓ Point of sales Transaction Analysis
  - ✓ Customer Churn Analysis
  - ✓ Sentiment Analysis

http://wiki.apache.org/hadoop/PoweredBy

Hidden Treasure

Case Study: Sears Holding Corporation

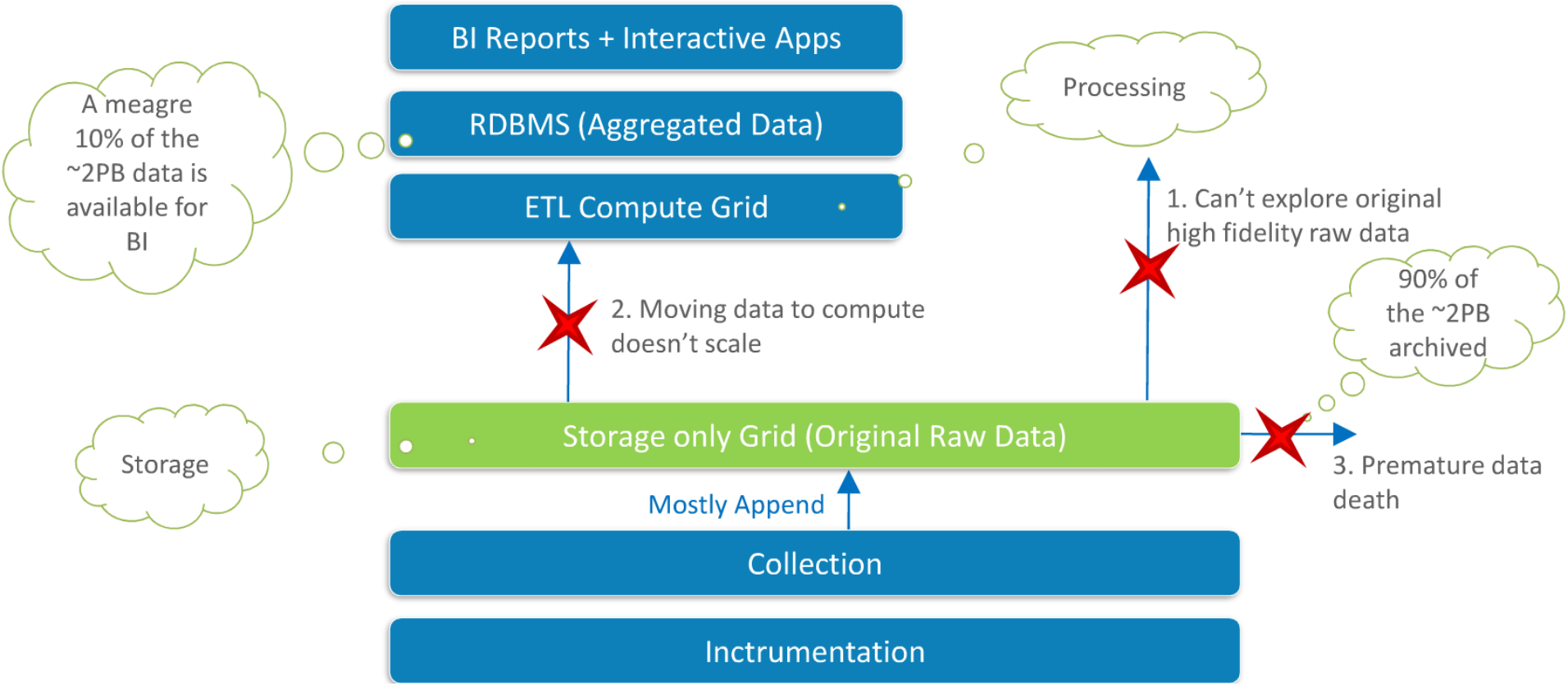
- ✓ Insight into data can provide **Business Advantage**.
- ✓ Some key early indicators can mean **Fortunes to Business**.
- ✓ **More Precise Analysis** with more data.



*Sears was using traditional systems such as Oracle Exadata, Teradata and SAS etc. to store and process the customer activity and sales data.

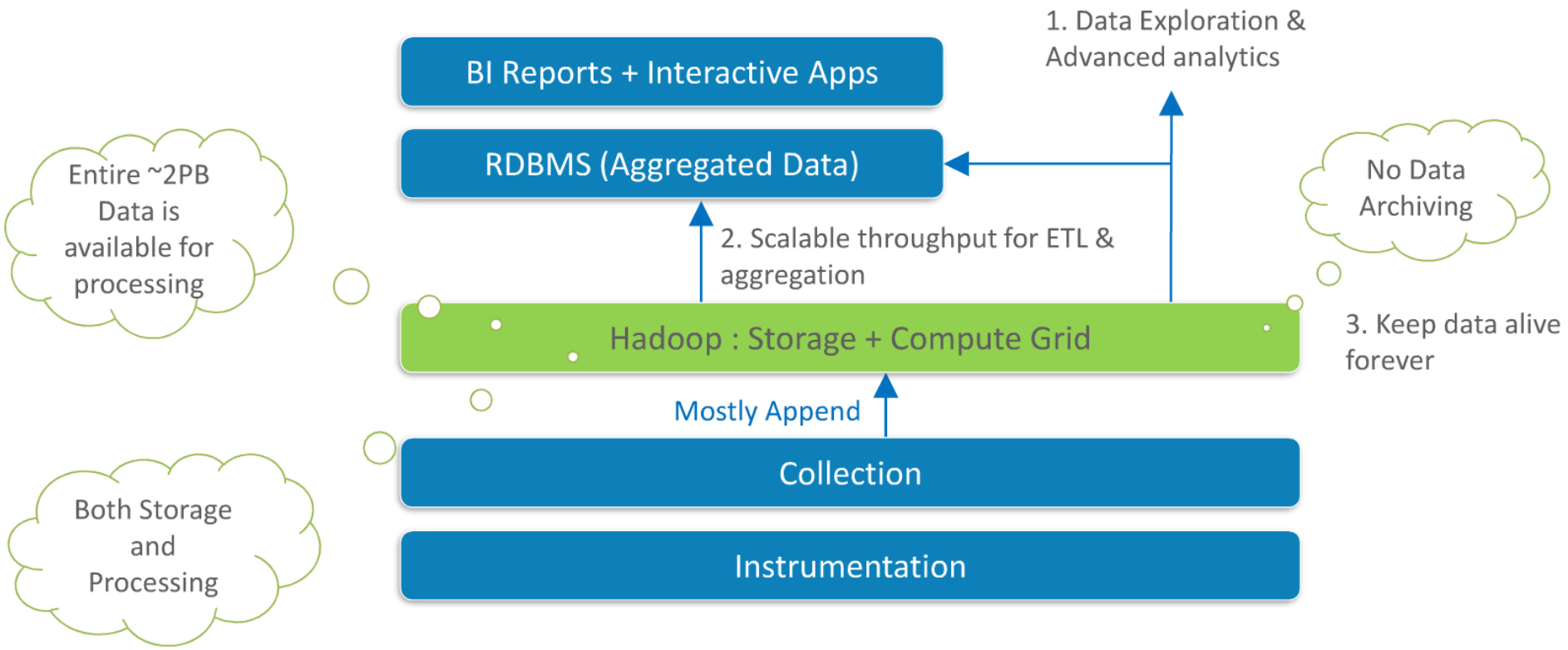
Aurelius™ CORPORATE SOLUTIONS

Limitations of Existing Data Analytics Architecture



http://www.informationweek.com/it-leadership/why-sears-is-going-all-in-on-hadoop/d/d-id/1107038?

Solution: A Combined Storage Computer Layer

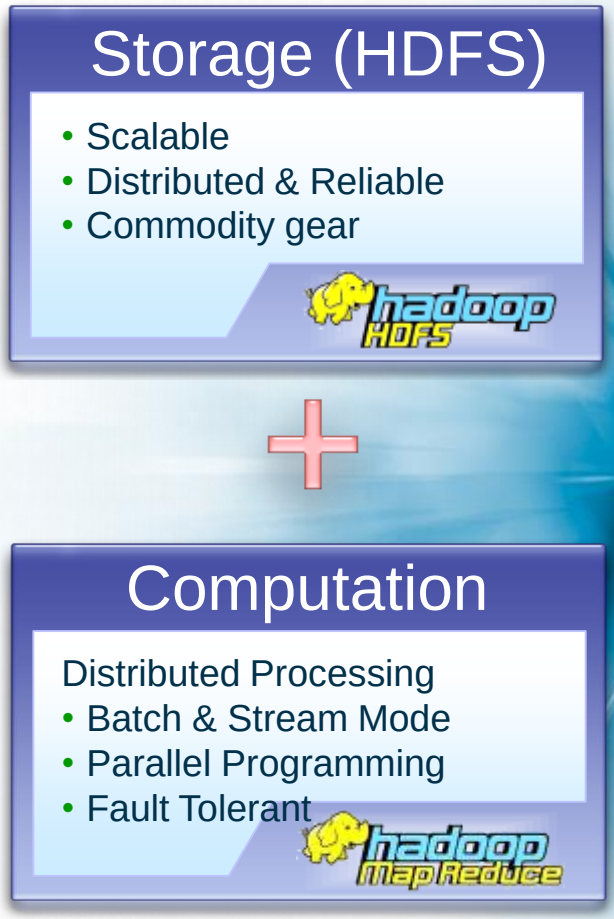


**Sears moved to a 300-Node Hadoop cluster to keep 100% of its data available for processing rather than a meagre 10% as was the case with existing Non-Hadoop solutions.*

Aurelius™ CORPORATE SOLUTIONS

How Hadoop Solves the problem?

A scalable fault-tolerant distributed system for data storage and processing (open source under the Apache license).



Scalable

- New nodes can be added on the fly

Distributed Fault Tolerant

- Through software framework

Affordable

- Massively parallel computing on commodity servers (instead of high-cost hardware(e.g. IBM POWER7 or Sun-Oracle's Sparc[4] RISC), go for vendor of your choice. Scaling Hadoop to 4000 nodes at Yahoo!)

Flexible

- Hadoop is schema-less – can absorb any type of data

Map Reduce

- Resource management and scheduling coupled with a scalable data programming abstraction

Confused? Which solution to use?

Relational Database

Use when:
- **Interactive OLAP Analytics (<1sec)**
- **Data volume is manageable (in TBs)**
- **Multistep ACID Transactions**
- **100% SQL Compliance**

Hadoop

Use when:
- **Structured or Not (Flexibility)**
- **Scalability of Storage & Compute**
- **Complex Data Processing**

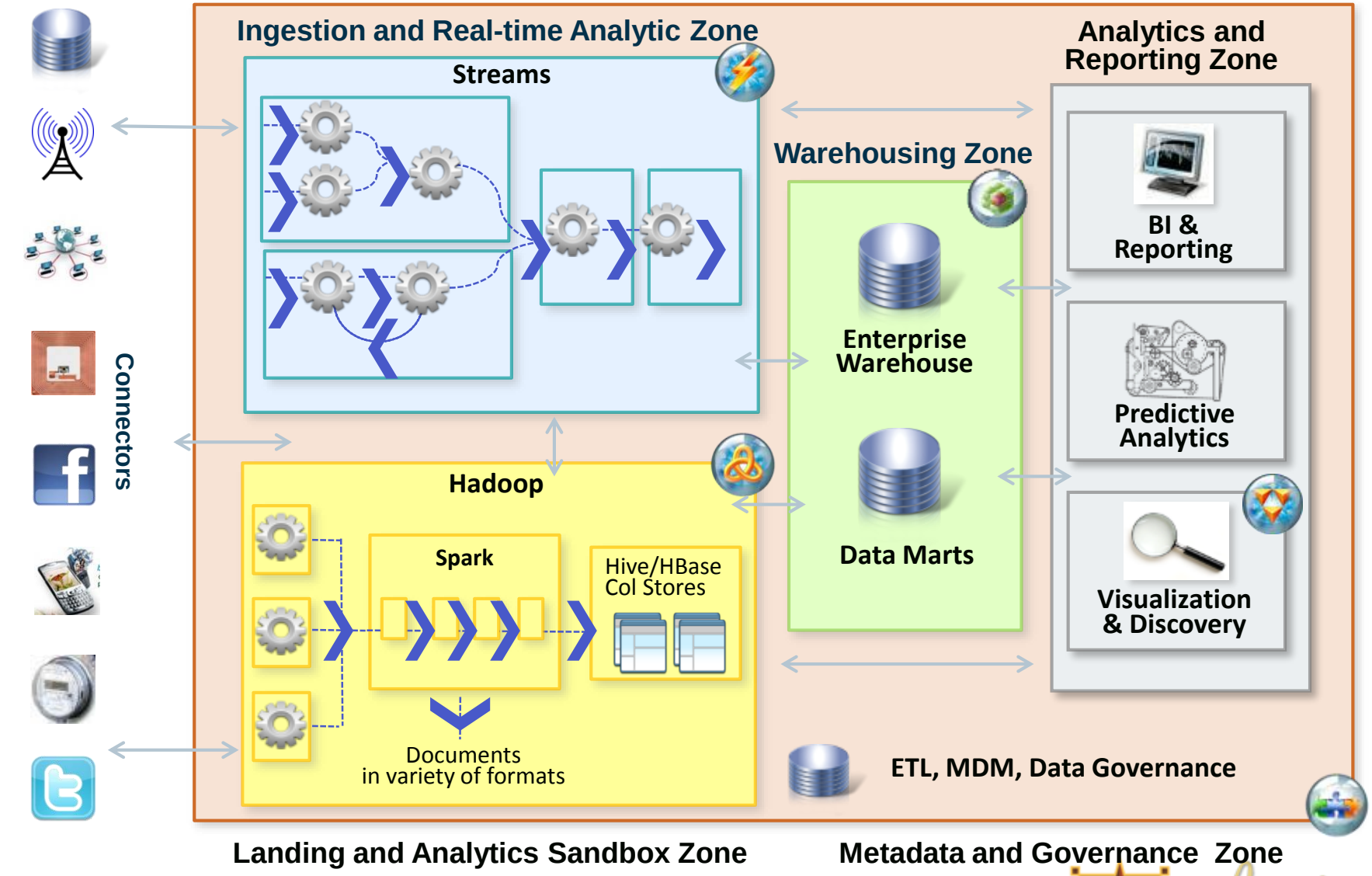
Aurelius™ CORPORATE SOLUTIONS

RDBMS Vs Hadoop

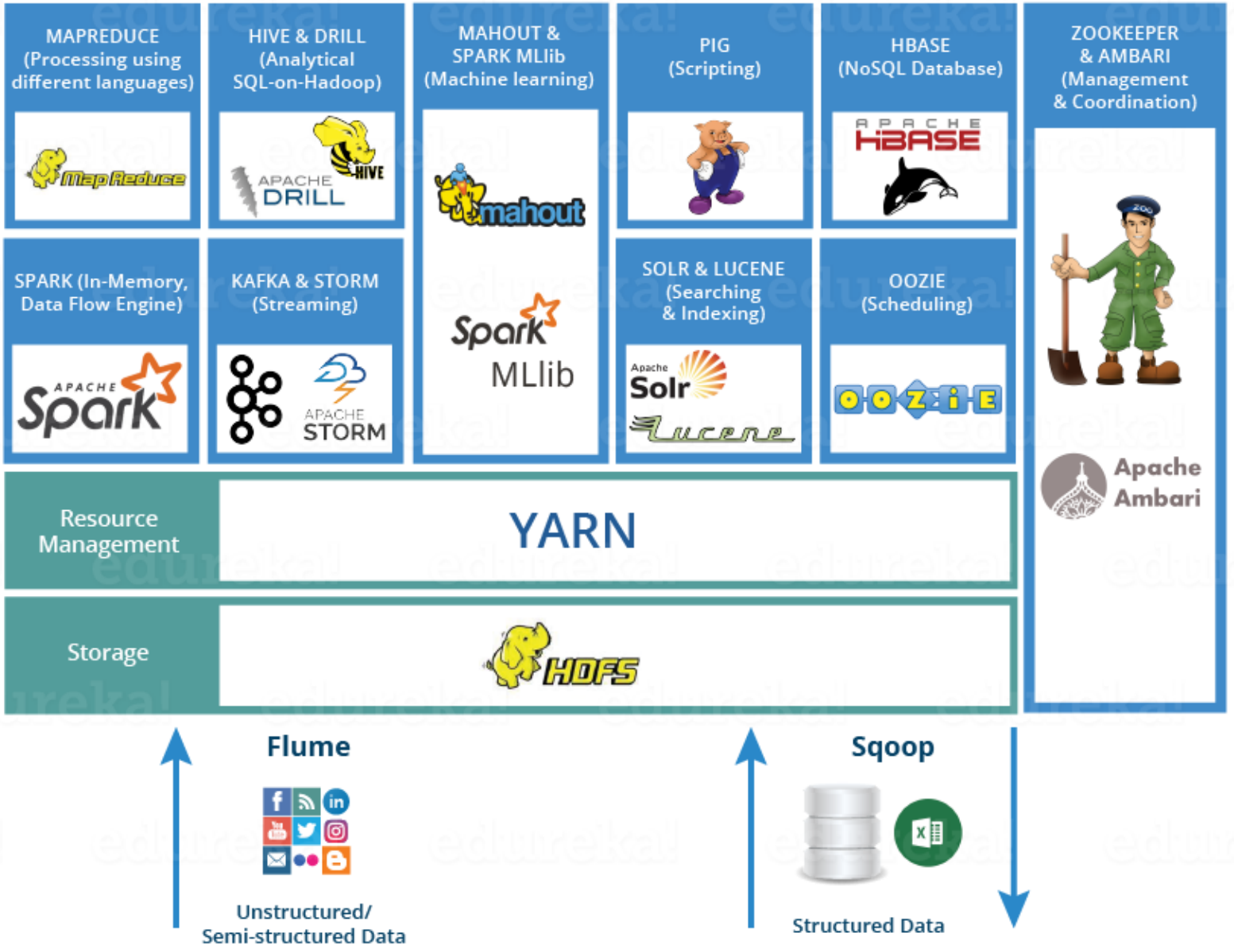
Structured	Data Types	Semi and Un-structured
Limited, No Data Processing	Processing	Processing coupled with Data
Standards & Structured	Governance	Loosely Structured
Required On Write	Schema	Required On Read
Reads are Fast	Speed	Writes are Fast
Software License	Cost	Support Only
Known Entity	Resources	Growing, Complexities, Wide
OLTP Complex ACID Transactions Operational Data Store	Best Fit Use	Data Discovery Processing Unstructured Data Massive Storage/Processing

Aurelius™ CORPORATE SOLUTIONS

An example of the big data platform in practice



Hadoop Ecosystem



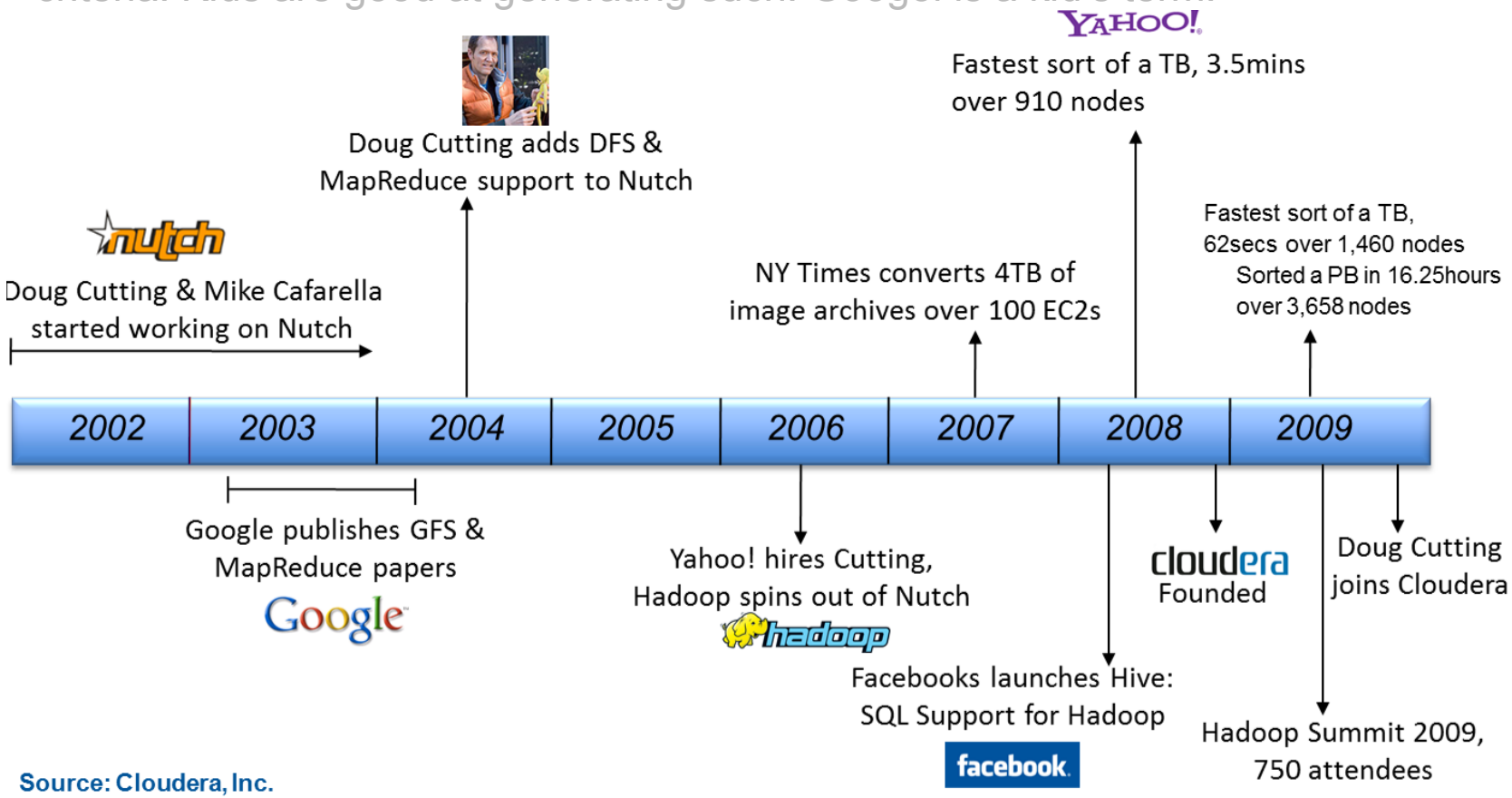
Aurelius™
CORPORATE SOLUTIONS

A Brief History of Hadoop & Timelines

Created by Doug Cutting, the creator of Apache Lucene.

- It's not an acronym; The project's creator, Doug Cutting, explains:

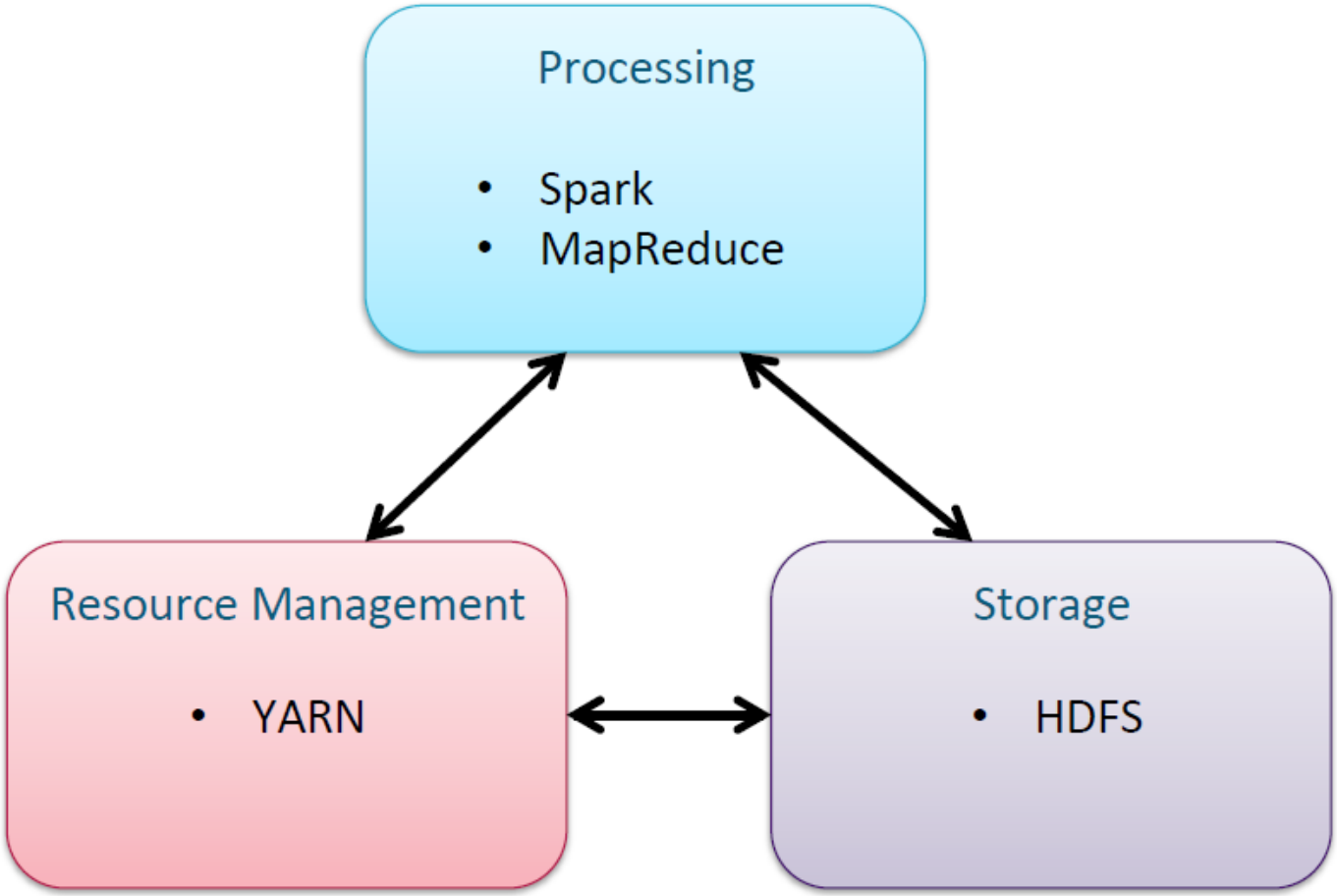
  "The name my kid gave a stuffed yellow elephant. Short, relatively easy to spell and pronounce, meaningless, and not used elsewhere: those are my naming criteria. Kids are good at generating such. Googol is a kid's term."



Source: Cloudera, Inc.

Aurelius™ CORPORATE SOLUTIONS

Hadoop Components

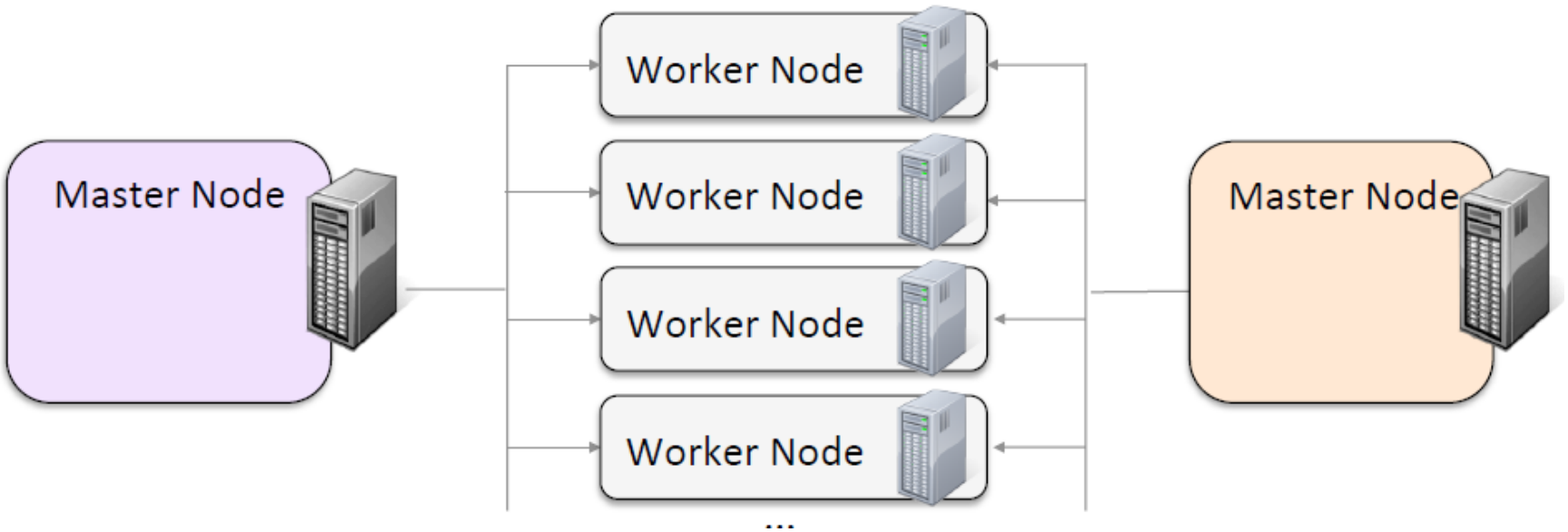


A Hadoop Cluster

Aurelius™ CORPORATE SOLUTIONS

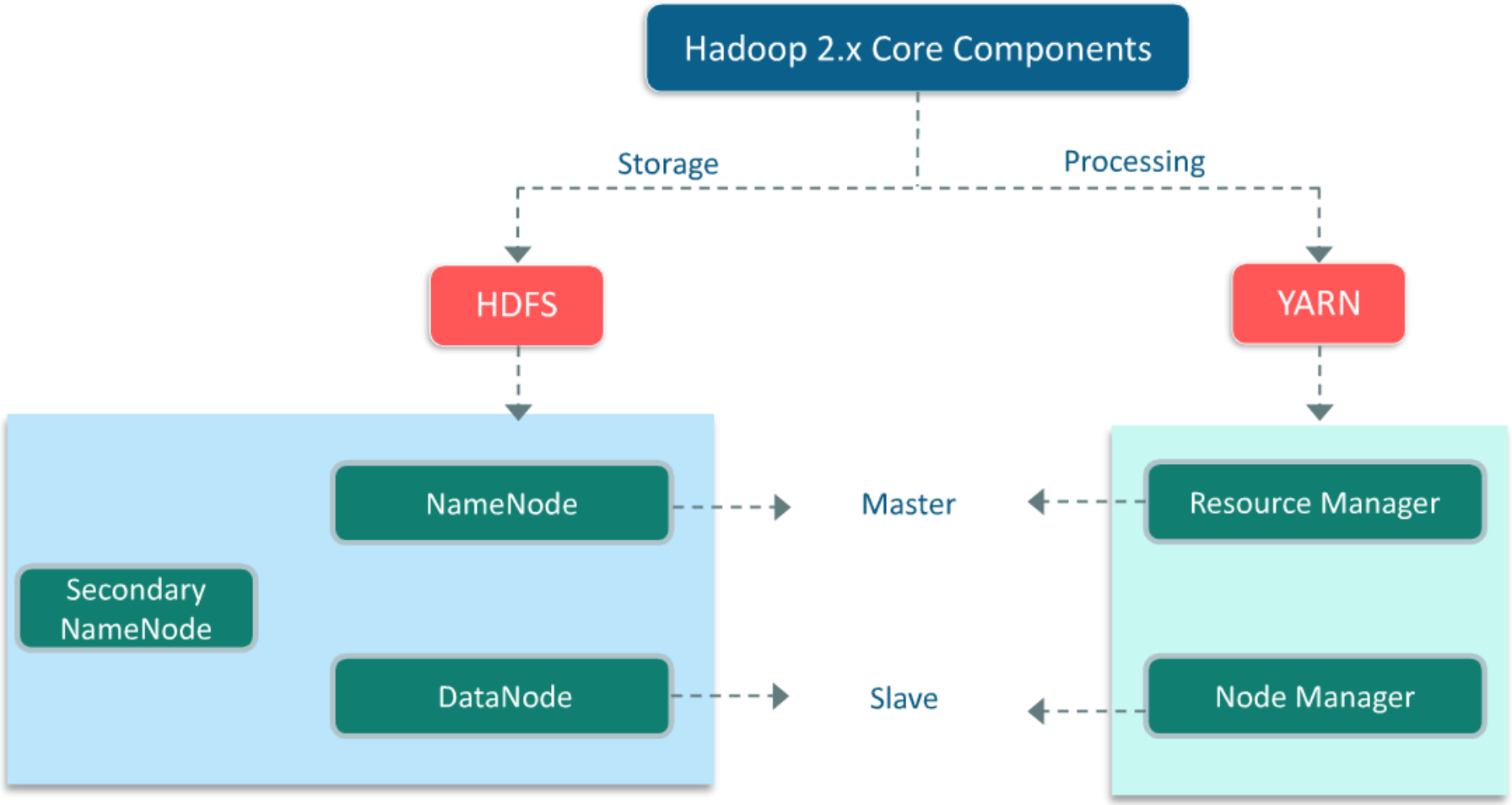
Hadoop Terminologies

- ***Cluster:* Group of computers working together**
  - *Provides data storage, data processing, and resource management*
- ***Node: A* n individual computer in the cluster**
  - *Master* nodes manage distribution of work and data to *worker* or *slave* nodes
- ***Daemon: A* program running on a node**
  - *Each performs different functions in the cluster*



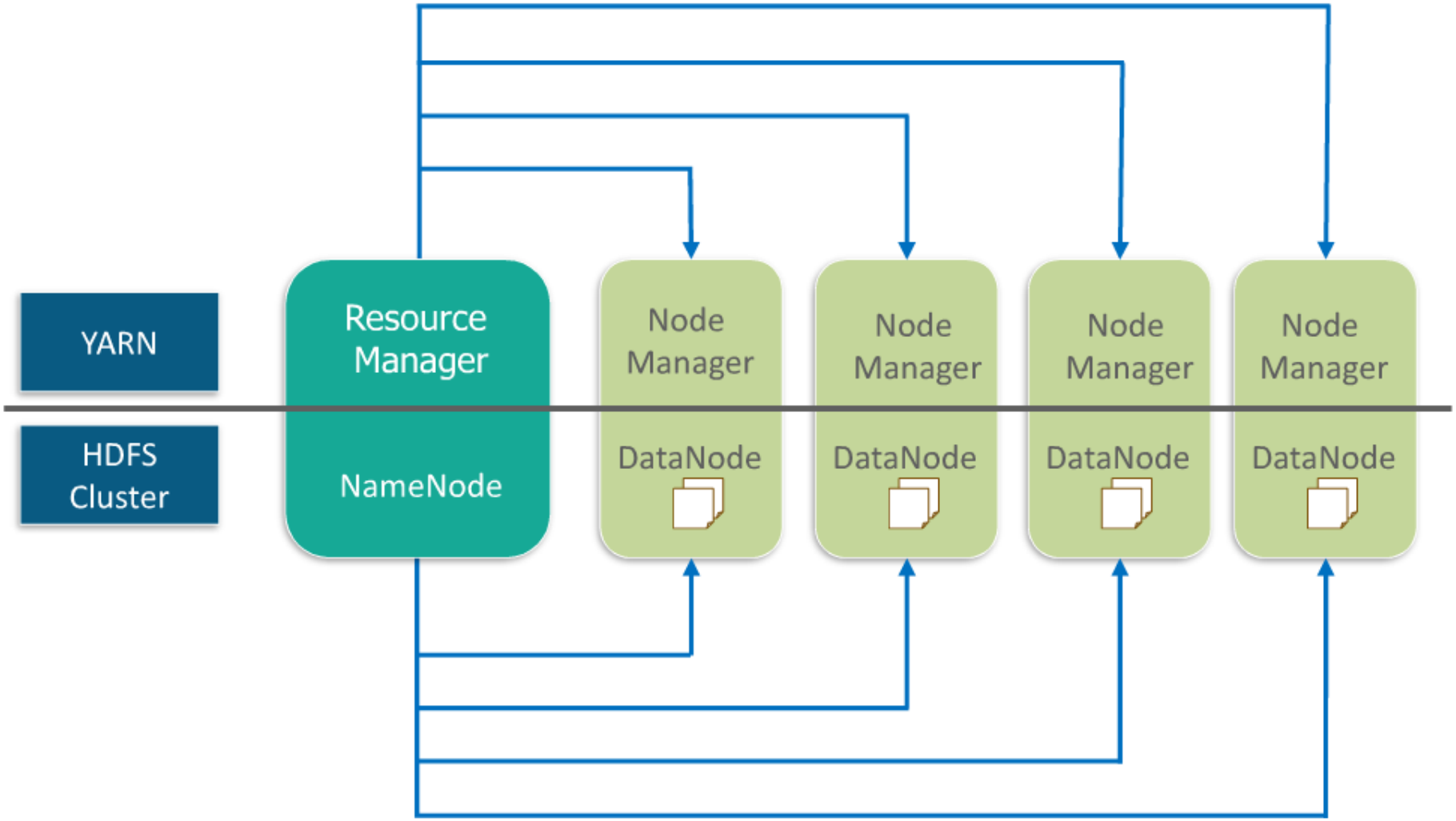
Aurelius™ CORPORATE SOLUTIONS

Hadoop 2.x Core Components



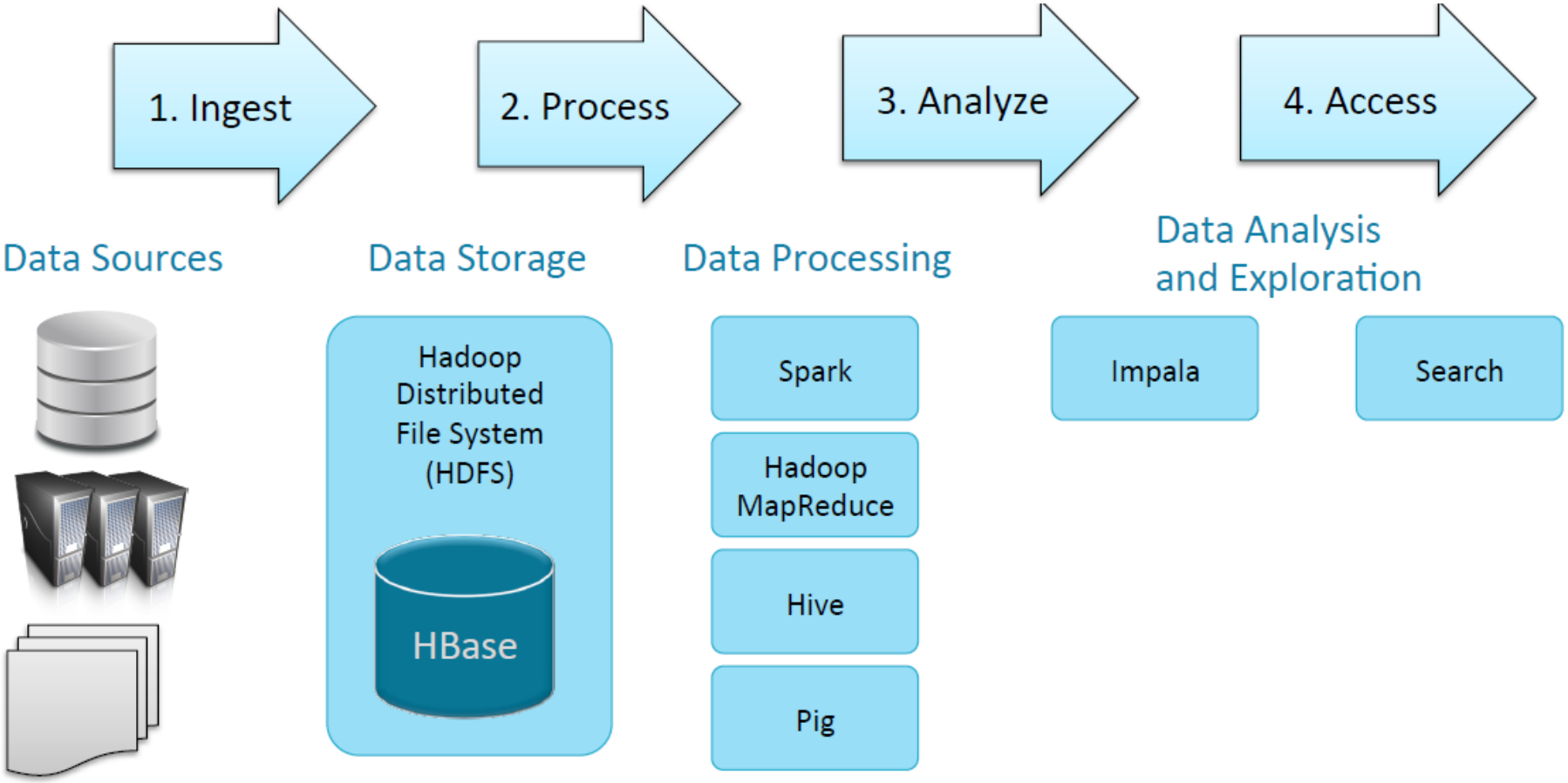
Aurelius™ CORPORATE SOLUTIONS

Hadoop 2.x Core Components Cont.



Aurelius™ CORPORATE SOLUTIONS

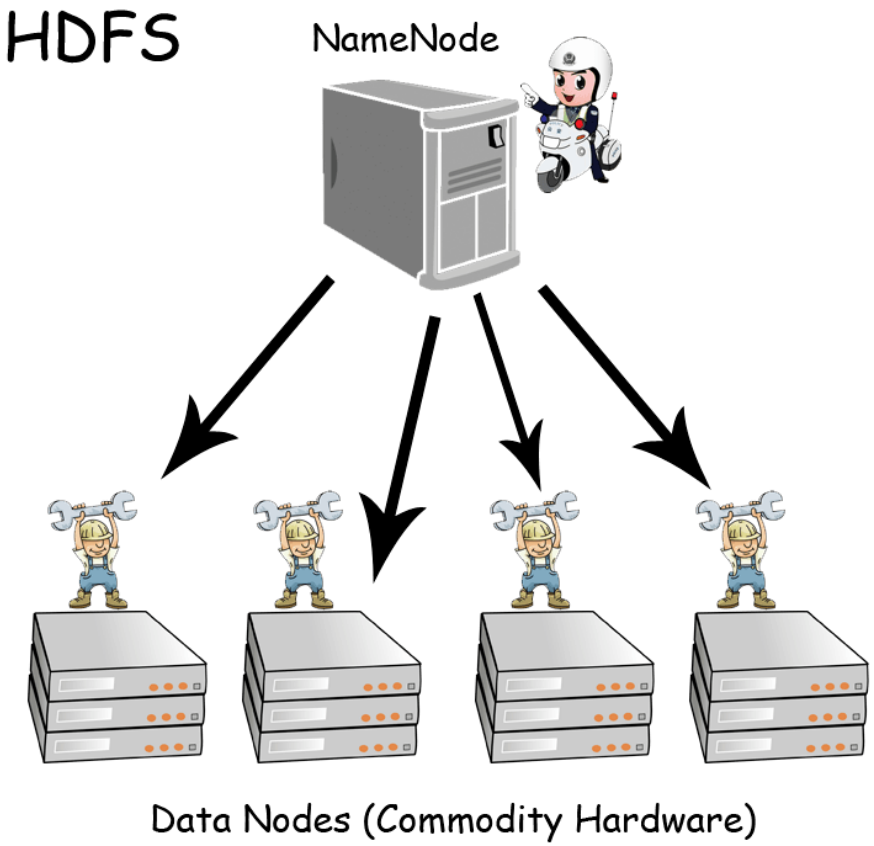
Big Data Processing



Aurelius™ CORPORATE SOLUTIONS

Storage: HDFS Basics

- **HDFS is a filesystem written in Java**
  - Based on Google's GFS
- **Sits on top of a native filesystem**
  - Such as ext3, ext4, or xfs
- **Provides redundant storage for massive amounts of data**
  - Using readily-available, industry-standard computers

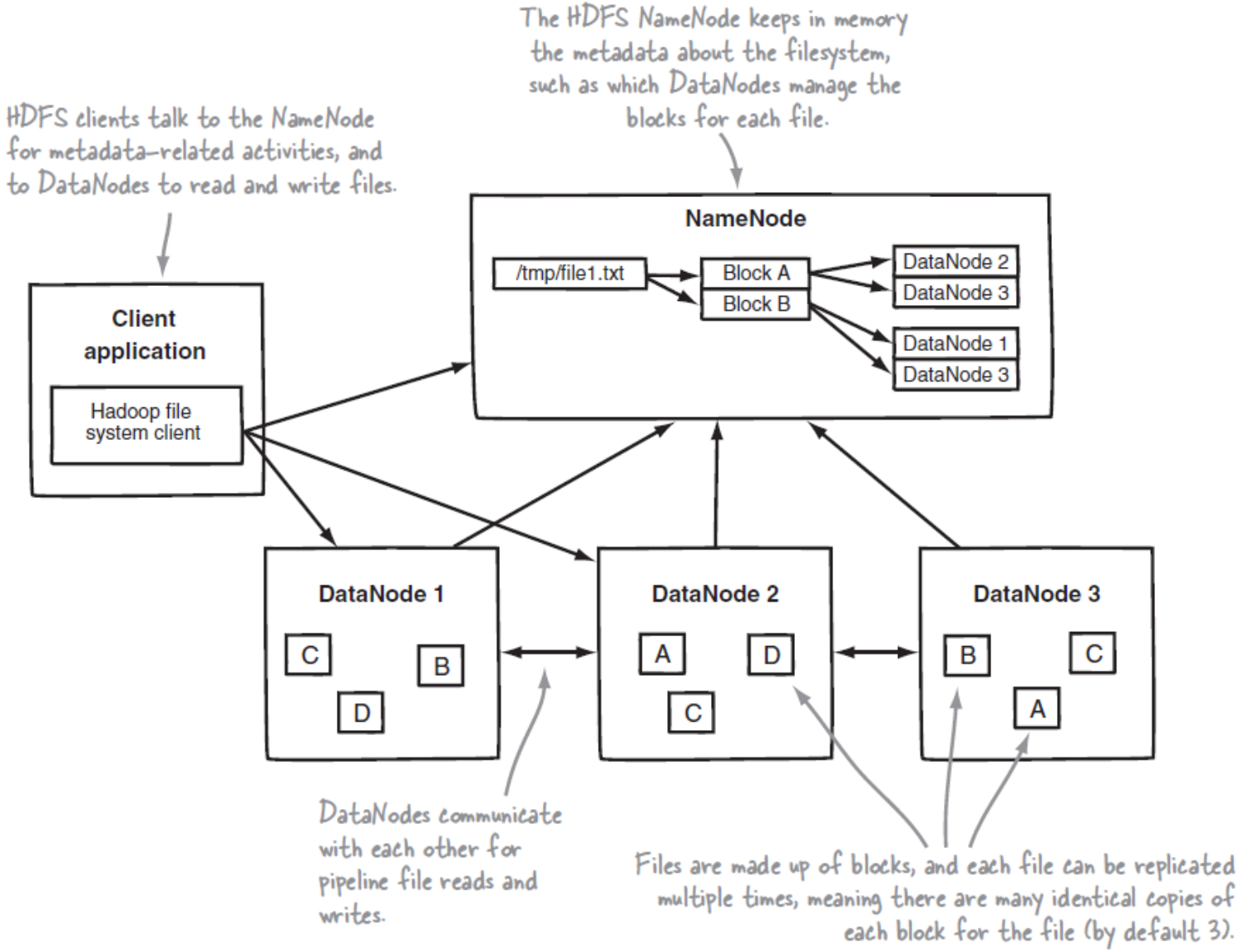


How it Stores:

- Data files are split into blocks (default 128MB) which are distributed at load time
- Each block is replicated on multiple data nodes (default 3x)
- Data Nodes stores data while Name Node stores metadata of files

Aurelius™ CORPORATE SOLUTIONS

Storage: HDFS Architecture



Aurelius™ CORPORATE SOLUTIONS

HDFS Architecture Cont.

NameNode

- Mater Node
- *Maintains and manages the blocks which are present on the DataNodes*

DataNode

- Slave nodes which are deployed on each machine and is responsible for storing the data
- *Responsible for serving read and write requests for the clients*

Name Node Metadata

Meta-data in Memory

- The entire metadata is in the main memory
- *No demand paging of FS meta-data*

Types of Metadata

- List of files
- *List of blocks for each file*
- *List of DataNode for each block*
- *File attributes, e.g. access time,replication factor, etc.*

A Transaction Log

- *Records file creations, file deletions etc.*

NameNode
(Stores metadata only)

METADATA:
/user/doug/hinfo -> 1 3 5
/user/doug/pdetail -> 4 2

NameNode:

Keeps track of overall file directory structure and the placement of Data Block

File Blocks

- By Default, block size is **128MB** in Hadoop 2.x and **64MB** in Hadoop 1.x

Hadoop 2.x

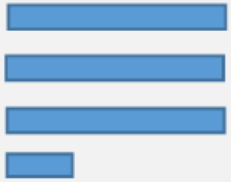
200MB – abc.txt

128MB – Block 1
72MB – Block 2

Hadoop 1.x

200MB – emp.dat

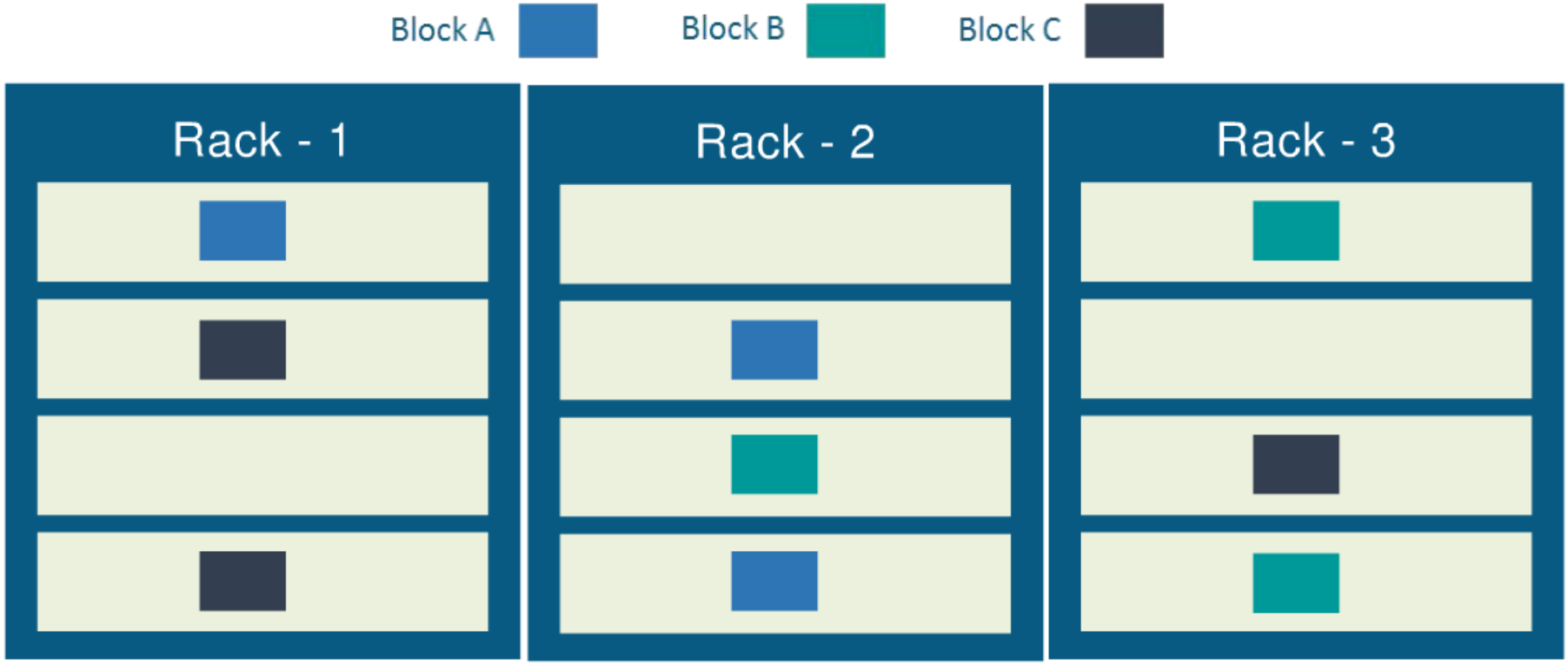
64MB – Block1
64MB – Block2
64MB – Block3
8MB – Block4



- Why the default block size is so large?
  - ❑ The main reason for having the HDFS blocks in large size is to reduce the cost of seek time.
  - ❑ The large block size is to account for proper usage of storage space while considering the limit on the memory of name node.

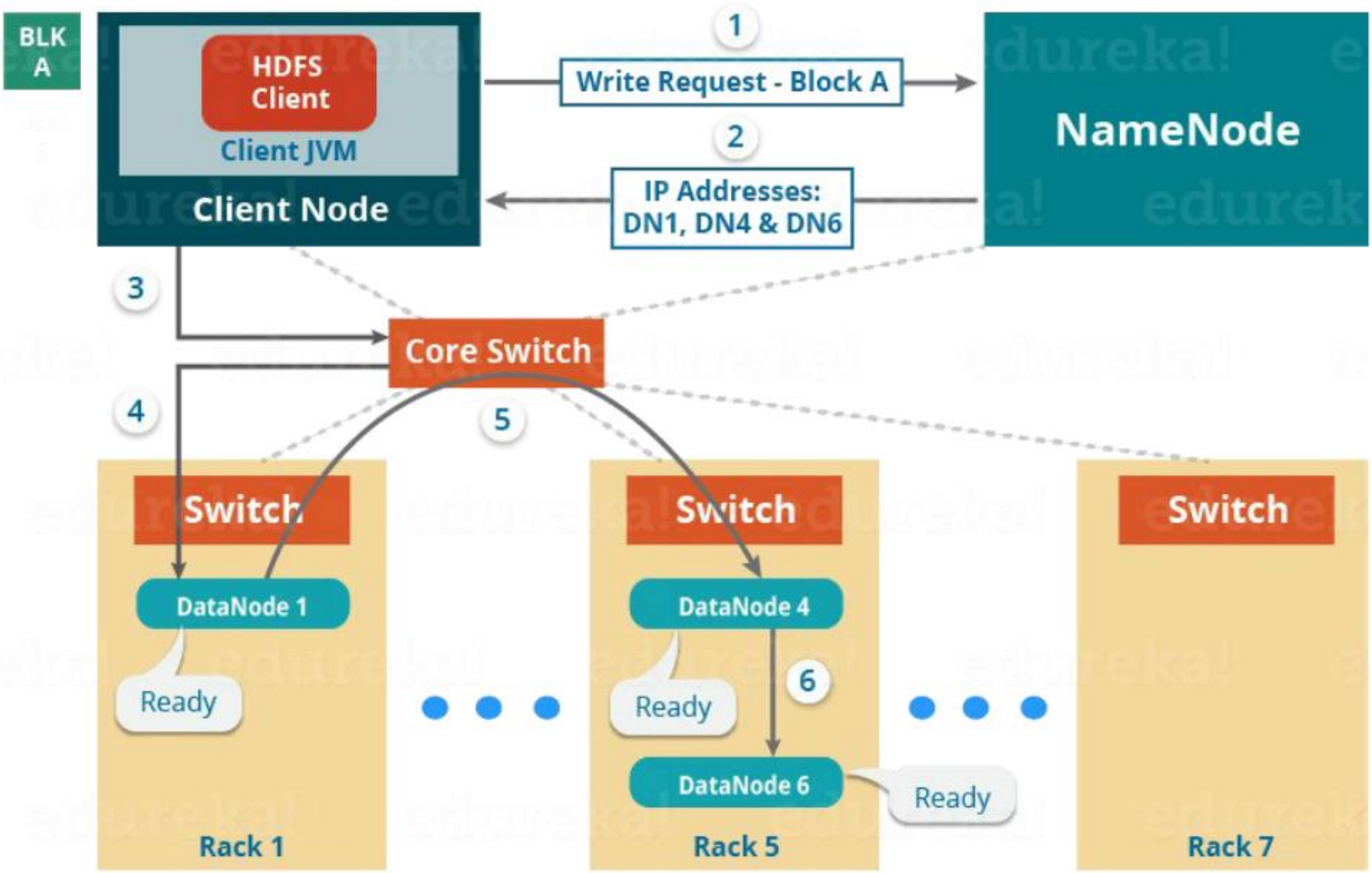
Rack Awareness

- *Rack Awareness Algorithm reduces latency* as well as *provide fault tolerance* by replicating data block
- Rack Awareness Algorithm says that the *first replica of a block will be stored on a local rack & the next two replicas will be stored on a different (remote) rack*



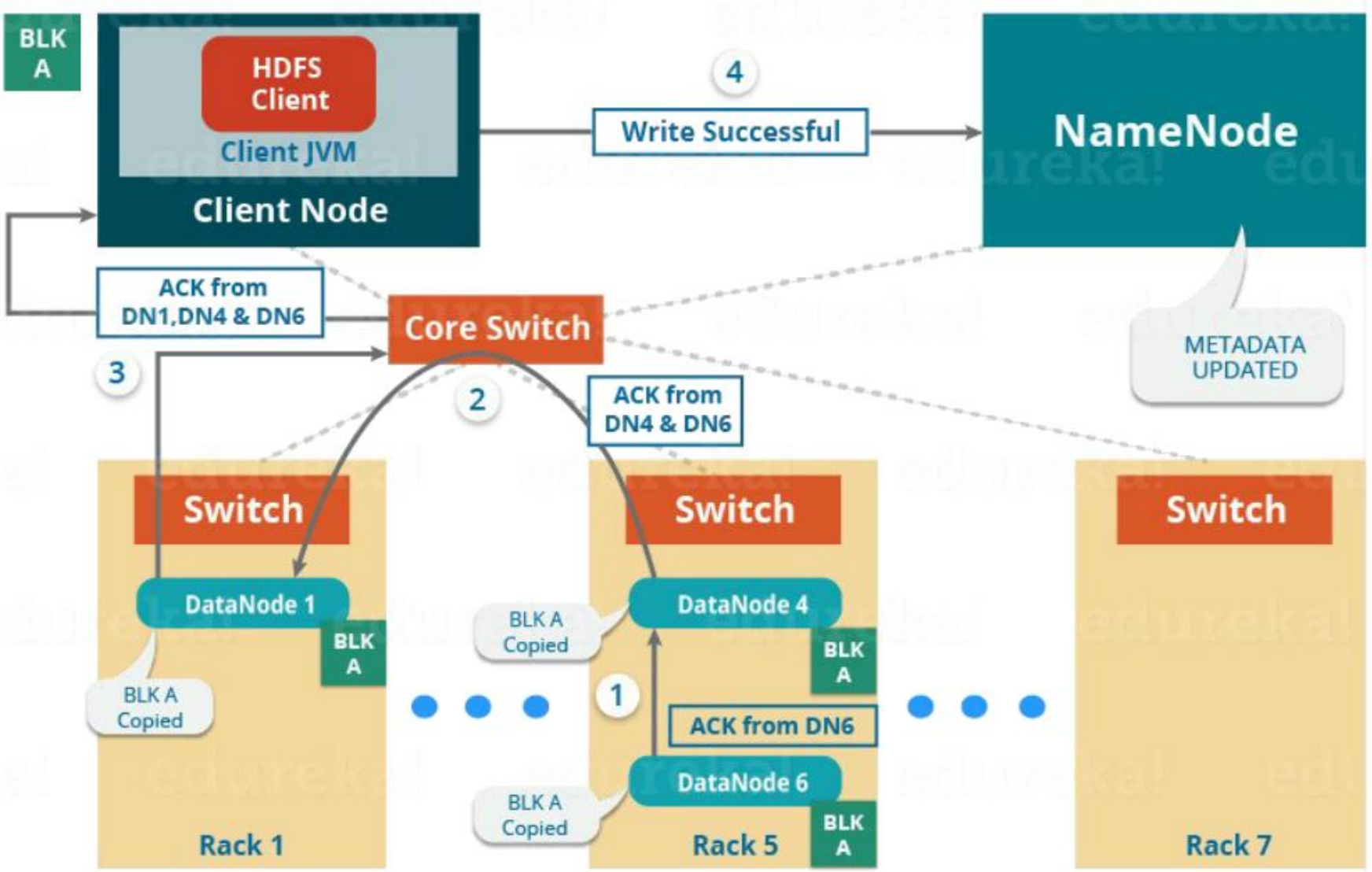
Aurelius™ CORPORATE SOLUTIONS

HDFS Write Mechanism – Setting up pipeline



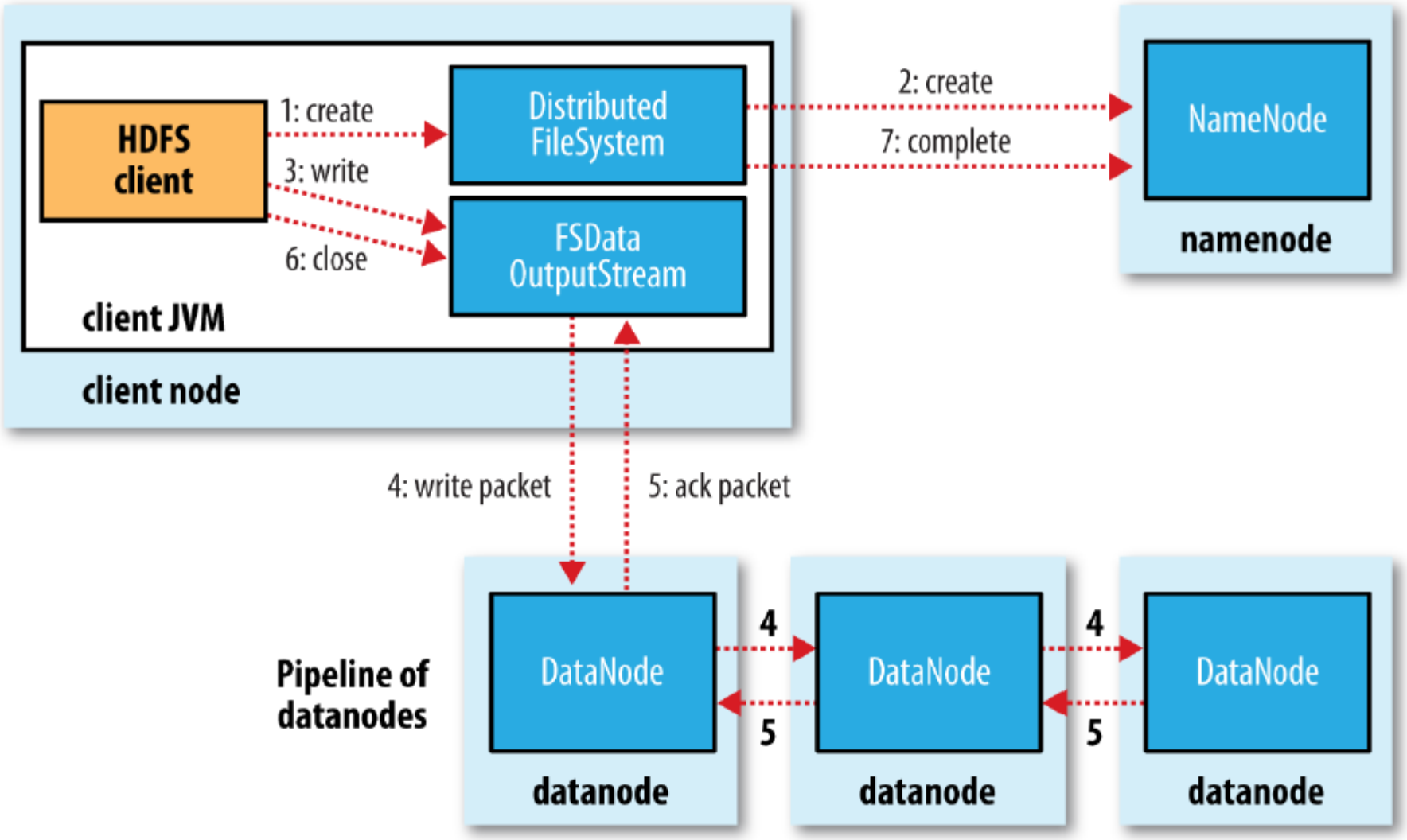
Aurelius™ CORPORATE SOLUTIONS

Write Acknowledgement in HDFS



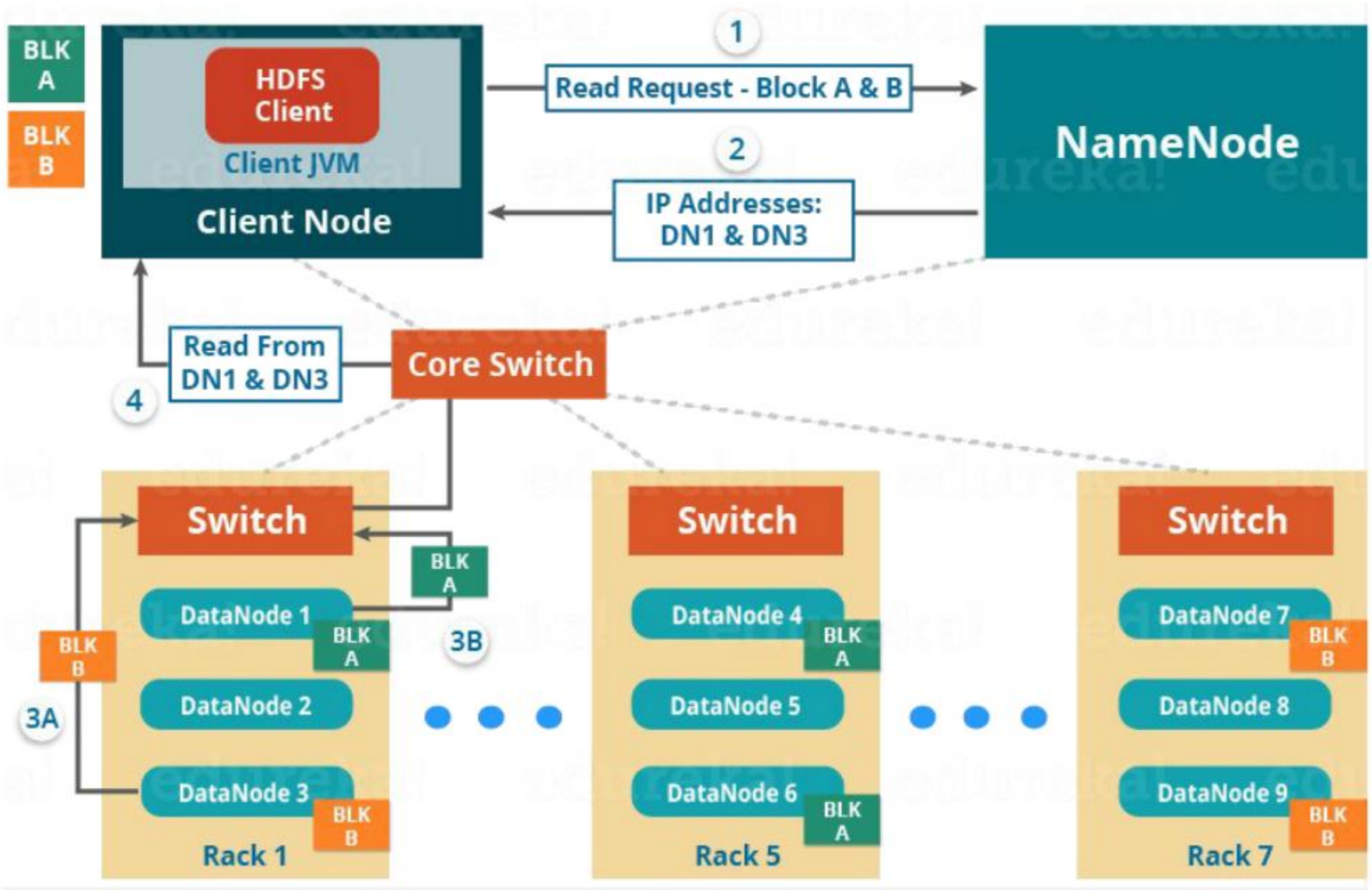
Aurelius™ CORPORATE SOLUTIONS

HDFS FileSystem: Write a file



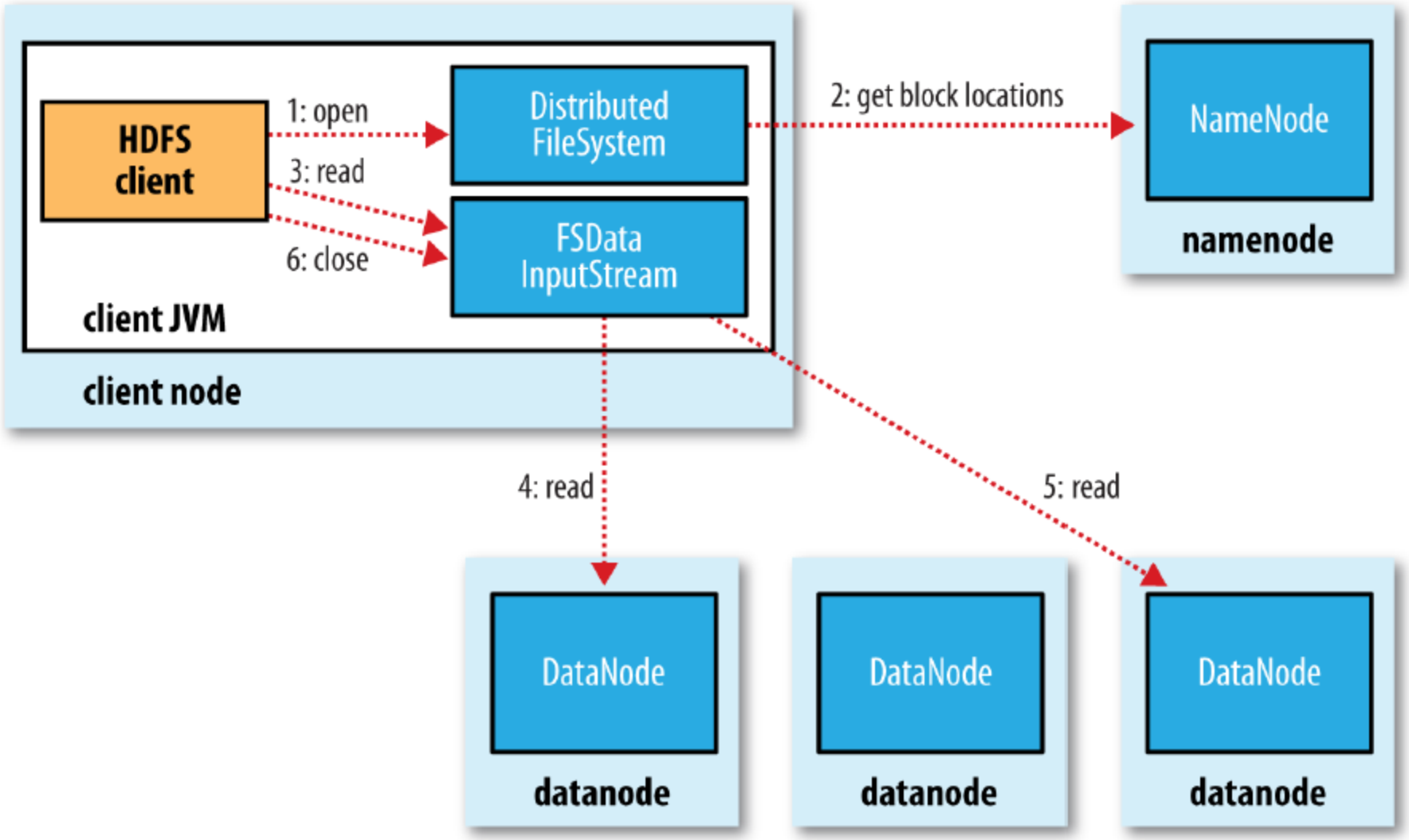
Aurelius™ CORPORATE SOLUTIONS

HDFS: Read Architecture



Aurelius™ CORPORATE SOLUTIONS

HDFS File System: Read a file



Aurelius™ CORPORATE SOLUTIONS

How to access data from HDFS

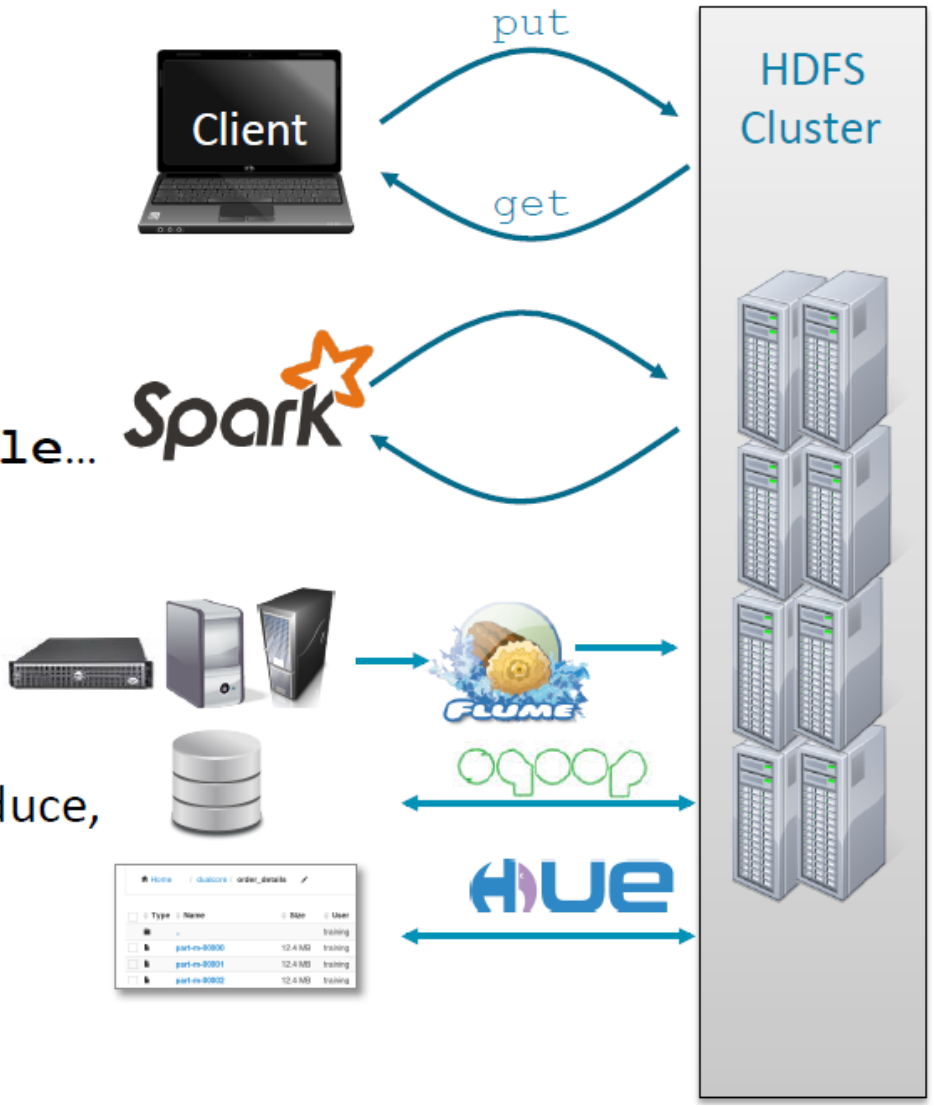
- **From the command line**
  - FsShell:

```
$ hdfs dfs
```

- **In Spark**
  - By URI, e.g.

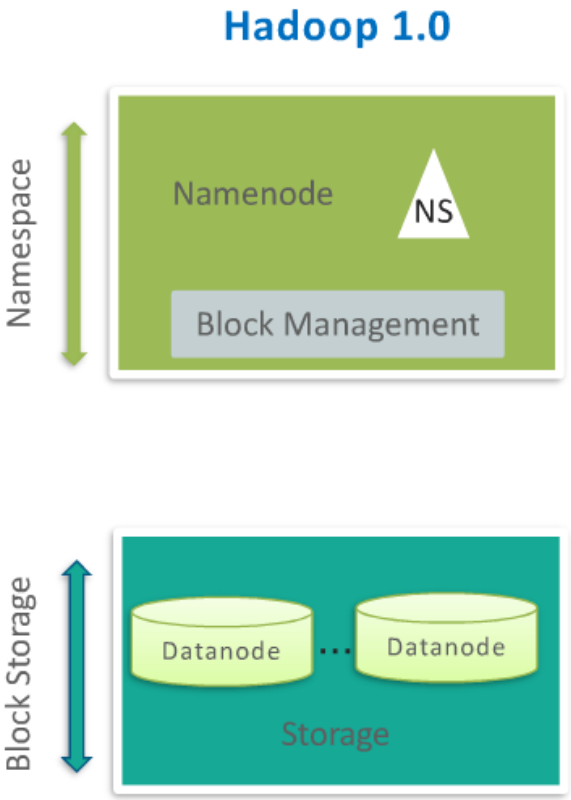
```
hdfs://nnhost:port/file…
```

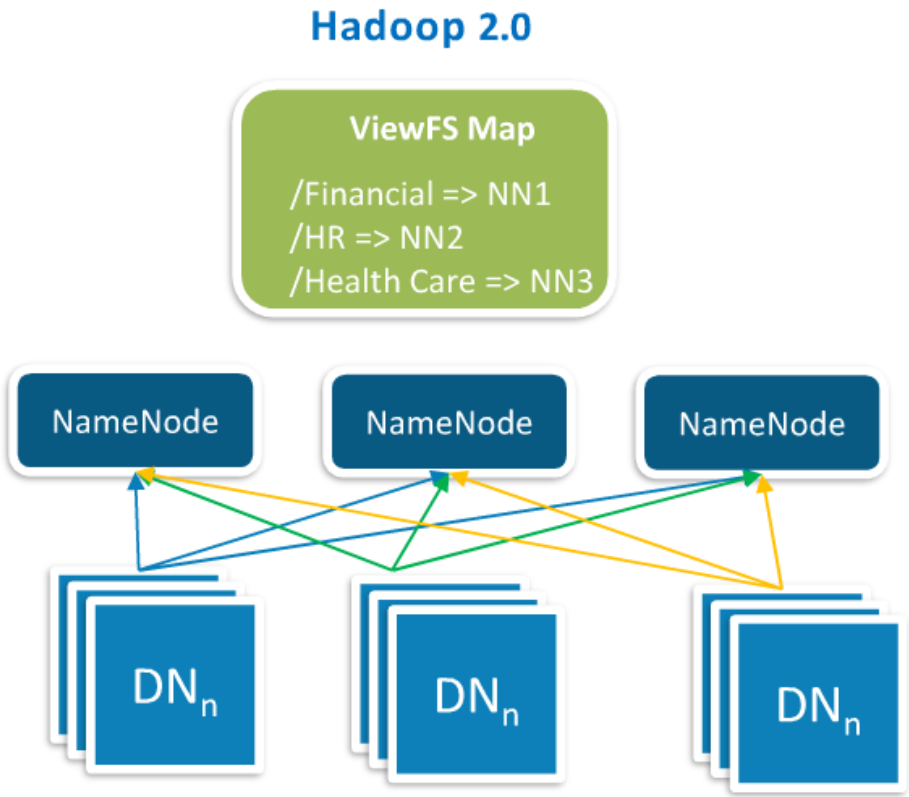
- **Other programs**
  - Java API
    - Used by Hadoop MapReduce, Impala, Hue, Sqoop, Flume, etc.
  - RESTful interface



Aurelius™ CORPORATE SOLUTIONS

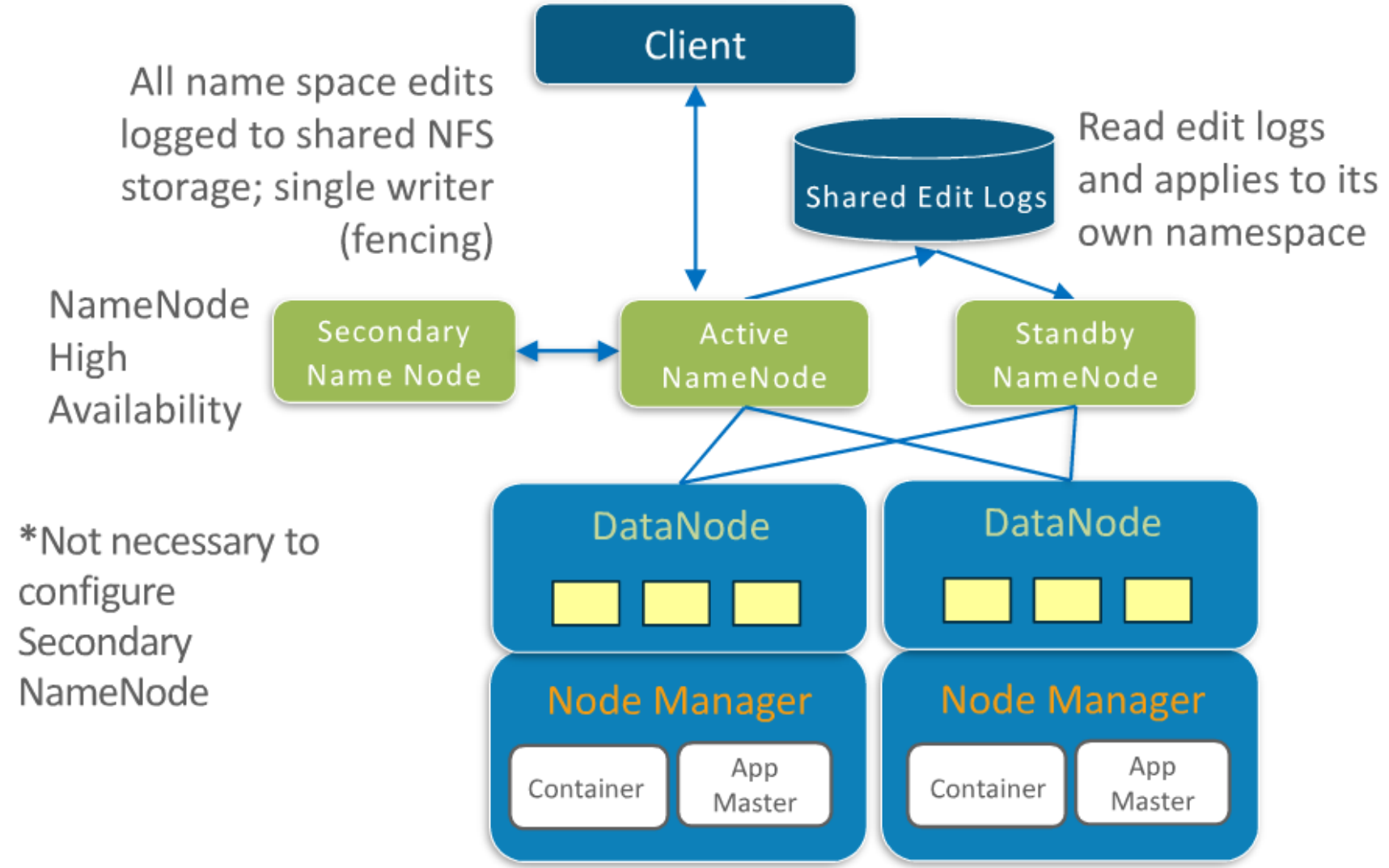
HDFS Federation





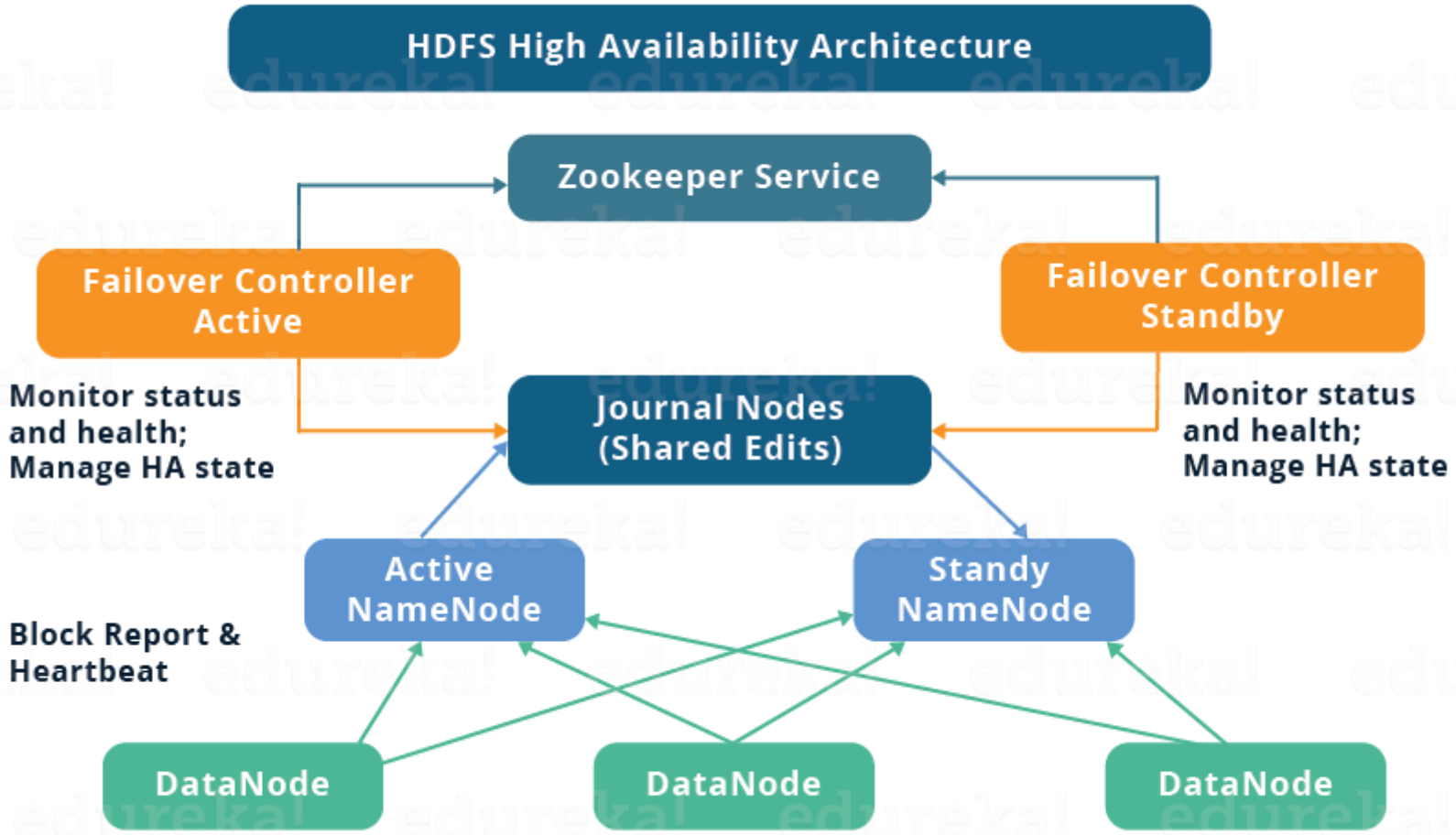
Aurelius™ CORPORATE SOLUTIONS

HDFS High Availability - NFS



Aurelius™ CORPORATE SOLUTIONS

HDFS High Availability – Zookeeper Quorum



Aurelius™ CORPORATE SOLUTIONS

Aurelius™
CORPORATE SOLUTIONS

Lab Environment – HDP Sandbox

- Prerequisites for LAB
  - **Virtualization enabled in all Machines**: Please need to make sure it is enabled in all machines for Virtual machine set up.
  - Download VMware /Virtual Box: Download the HDP Sandbox from Hortonrworks website (https://hortonworks.com/downloads/):
  - Putty and Winscp: Please download and installed these software's in all machines , to access VM and transfer data from the local system into VM.
- I am using: HDP 2.6.3 Sandbox

  (HDP_2.6.3_virtualbox_16_11_2017.ova)

Aurelius™ CORPORATE SOLUTIONS

Terminal Commands

Listing of files present on HDFS

```
cloudera@cloudera-vm:/usr/lib/hadoop-0.20/conf$ hadoop dfs -ls /
Found 3 items
drwxrwxrwx   - hue    supergroup          0 2013-04-20 03:01 /tmp
drwxr-xr-x   - hue    supergroup          0 2013-04-20 03:22 /user
drwxr-xr-x   - mapred supergroup          0 2011-04-12 06:20 /var
cloudera@cloudera-vm:/usr/lib/hadoop-0.20/conf$
```

Listing of files present in bin Directory

```
cloudera@cloudera-vm:~$ cd /usr/lib/hadoop-0.20/bin/
cloudera@cloudera-vm:/usr/lib/hadoop-0.20/bin$ ls
fuse_dfs_wrapper.sh  hadoop-daemons.sh  start-all.sh       stop-all.sh
hadoop               rcc                start-balancer.sh  stop-balancer.sh
hadoop-config.sh     README             start-dfs.sh       stop-dfs.sh
hadoop-daemon.sh     slaves.sh          start-mapred.sh    stop-mapred.sh
cloudera@cloudera-vm:/usr/lib/hadoop-0.20/bin$
```

Files Formats

- Key factor in Big Data processing and query performance
- Schema Evolution
- Compression and Splittability
- Data Processing
  - Write performance
  - Partial read
  - Full read

Available File Formats

- Text / CSV
- JSON / XML
- SequenceFile
  - binary key/value pair format
- Avro
- Parquet
- ORC
  - optimized row columnar format

AVRO

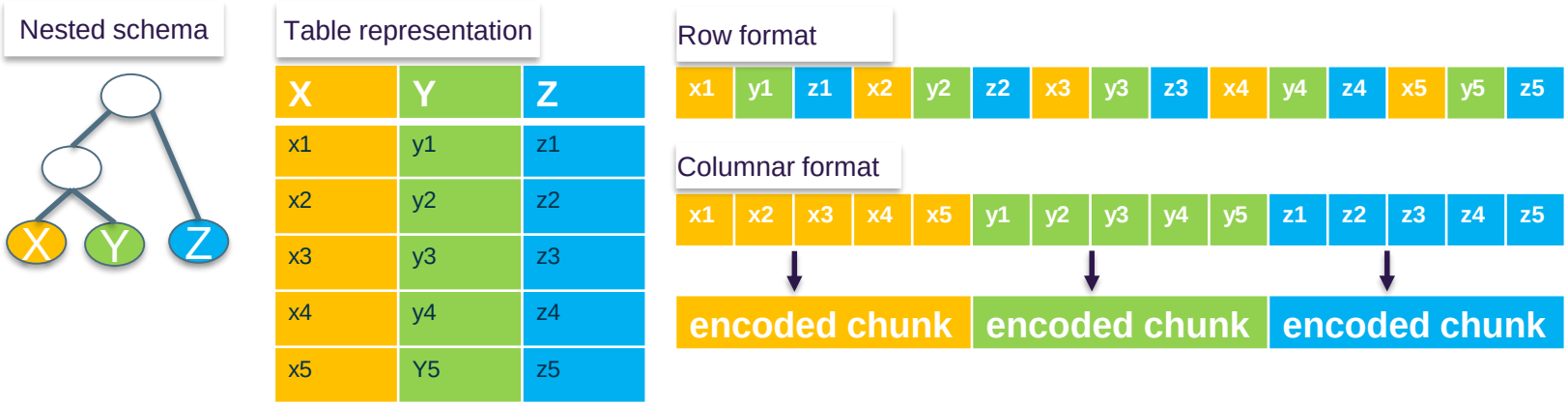
- Language neutral data serialization system
  - Write a file in python and read it in C
- AVRO data is described using language independent schema
- AVRO schemas are usually written in JSON and data is encoded in binary format
- Supports schema evolution
  - producers and consumers at different versions of schema
- Supports compression and are splittable

Sample AVRO schema in JSON format

```
{
  "type" : "record",
  "name" : "tweets",
  "fields" : [ {
    "name" : "username",
    "type" : "string",
}, {
    "name" : "tweet",
    "type" : "string",
}, {
    "name" : "timestamp",
    "type" : "long",
} ],
  "doc:" : "schema for storing tweets"
}
```

Parquet

- columnar storage format
- key strength is to store nested data in truly columnar format using definition and repetition levels[1]



(1) Dremel made simple with parquet - https://blog.twitter.com/2013/dremel-made-simple-with-parquet

YARN Basics

- **YARN** stands for **Yet Another Resource Negotiator**

- **YARN is the Hadoop processing layer that contains**
  - A resource manager
  - A job scheduler

- **YARN allows multiple data processing engines to run on a single Hadoop cluster**
  - Batch programs (e.g. Spark, MapReduce)
  - Interactive SQL (e.g. IBM BigSQL, Impala)
  - Advanced analytics (e.g. Spark)
  - Streaming (e.g. Spark Streaming, Storm)

YARN Components

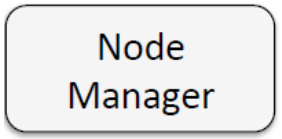
YARN Daemons

Resource Manager (RM)

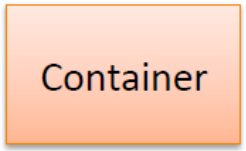
- Global resource scheduler, runs on master node
- Arbitrates system resources between competing applications
- Has a pluggable scheduler to support different algorithms (capacity, fair scheduler, etc.)

Node Manager (NM)

- Runs on slave nodes
- Communicates with RM



Slave Thread for a running Application

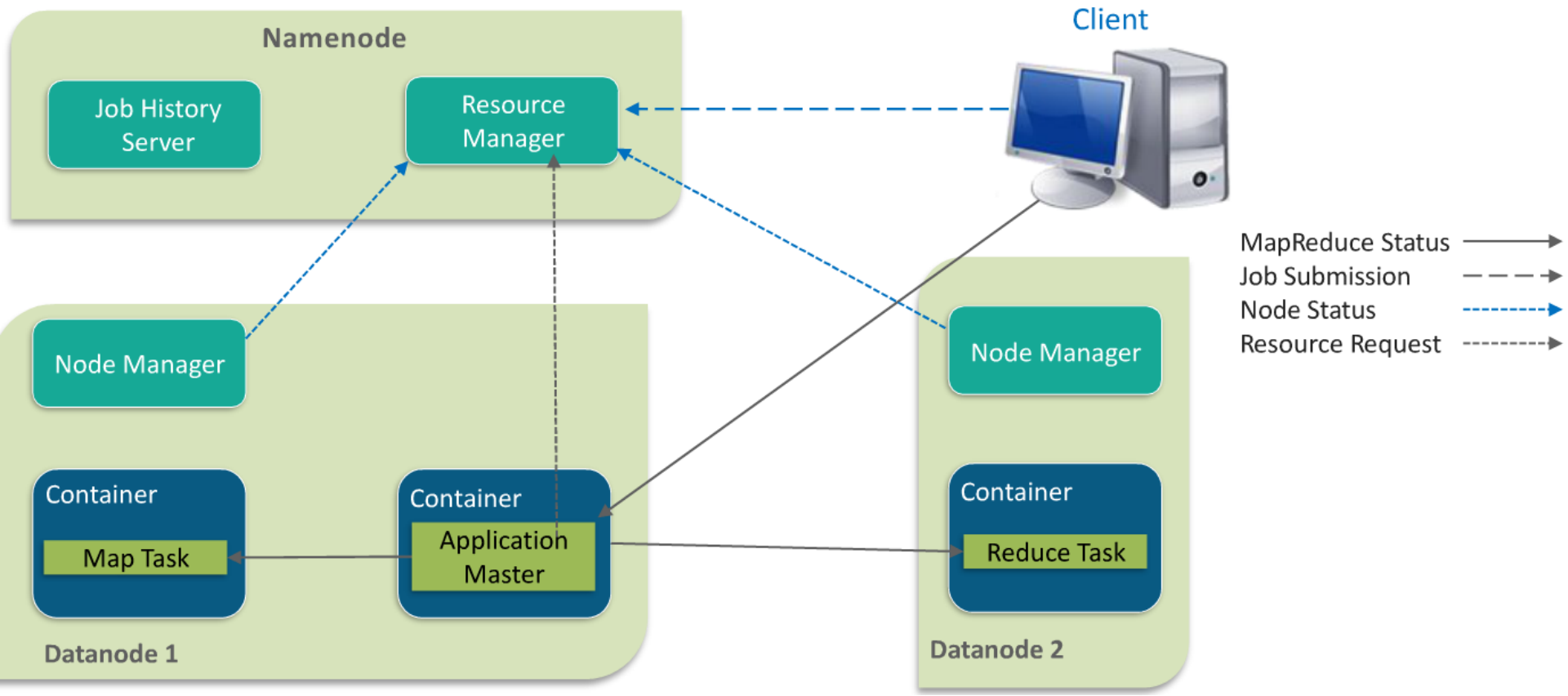


Containers

- Created by the RM upon request
- Allocate a certain amount of resources (memory, CPU) on a slave node
- Applications run in one or more containers

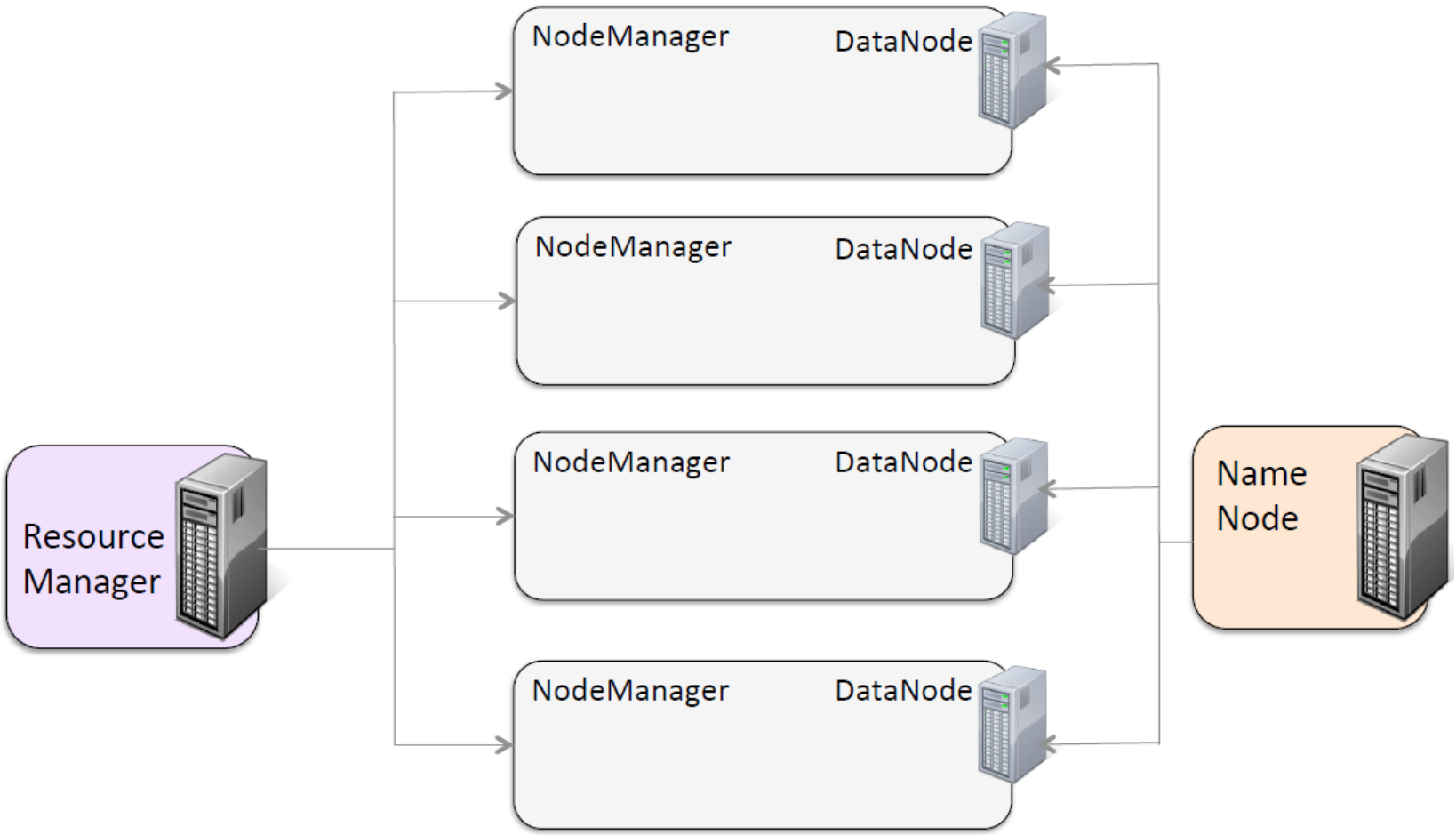
Application Master (AM)

- One per application, runs in a container
- Framework/application specific
- Requests more containers to run application tasks

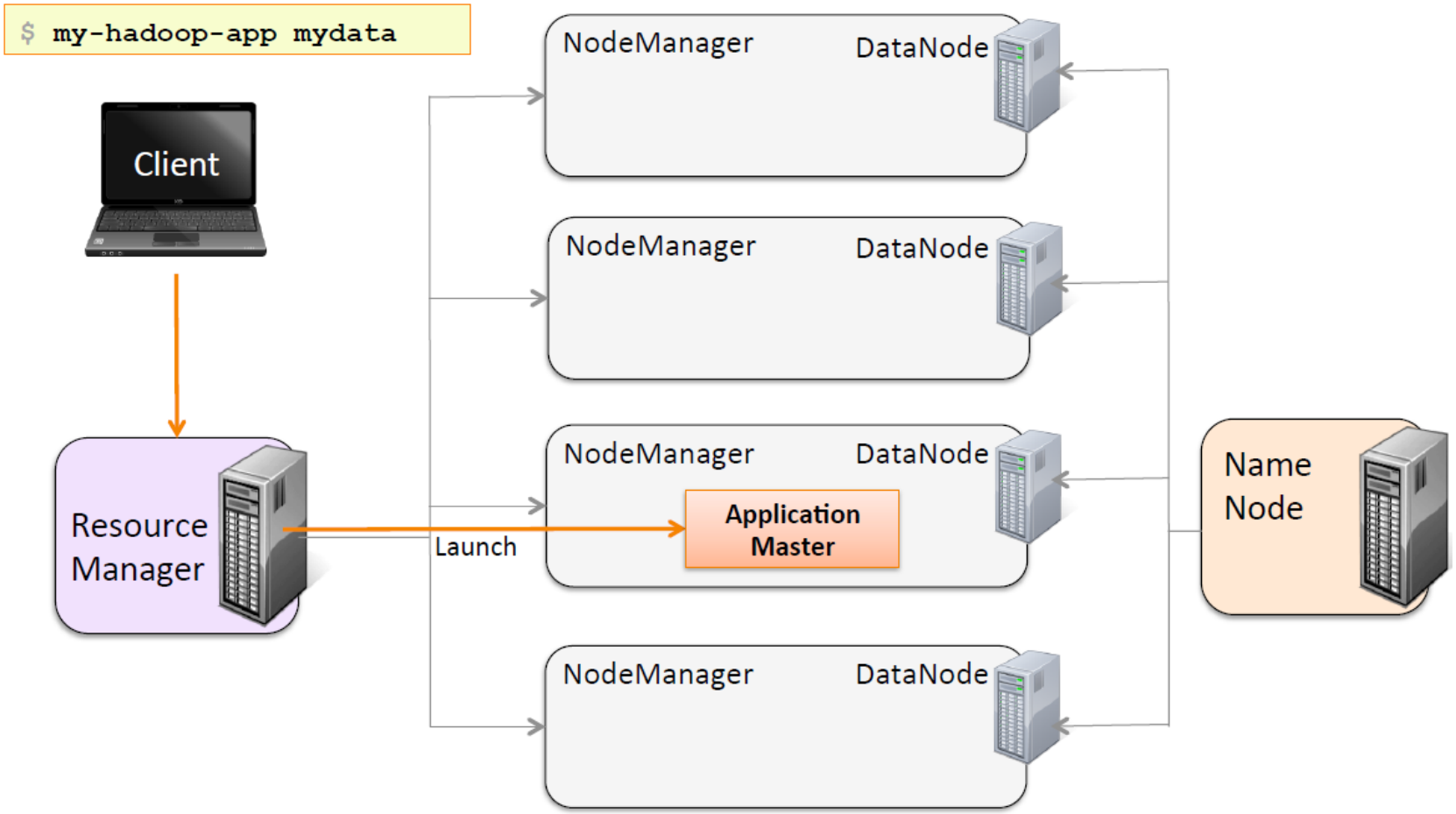


Aurelius™ CORPORATE SOLUTIONS

Running a YARN Application: 1

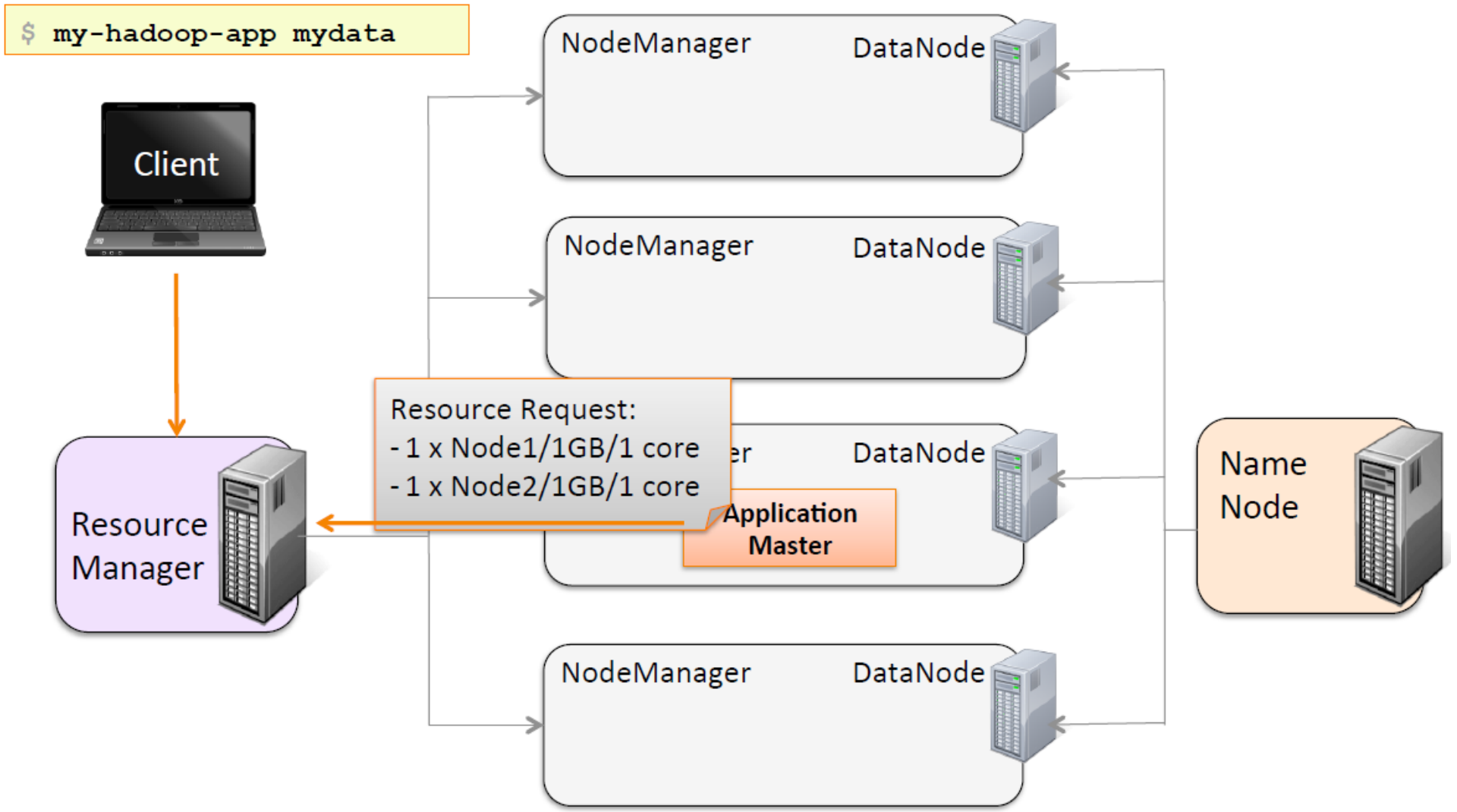


Running a YARN Application: 2



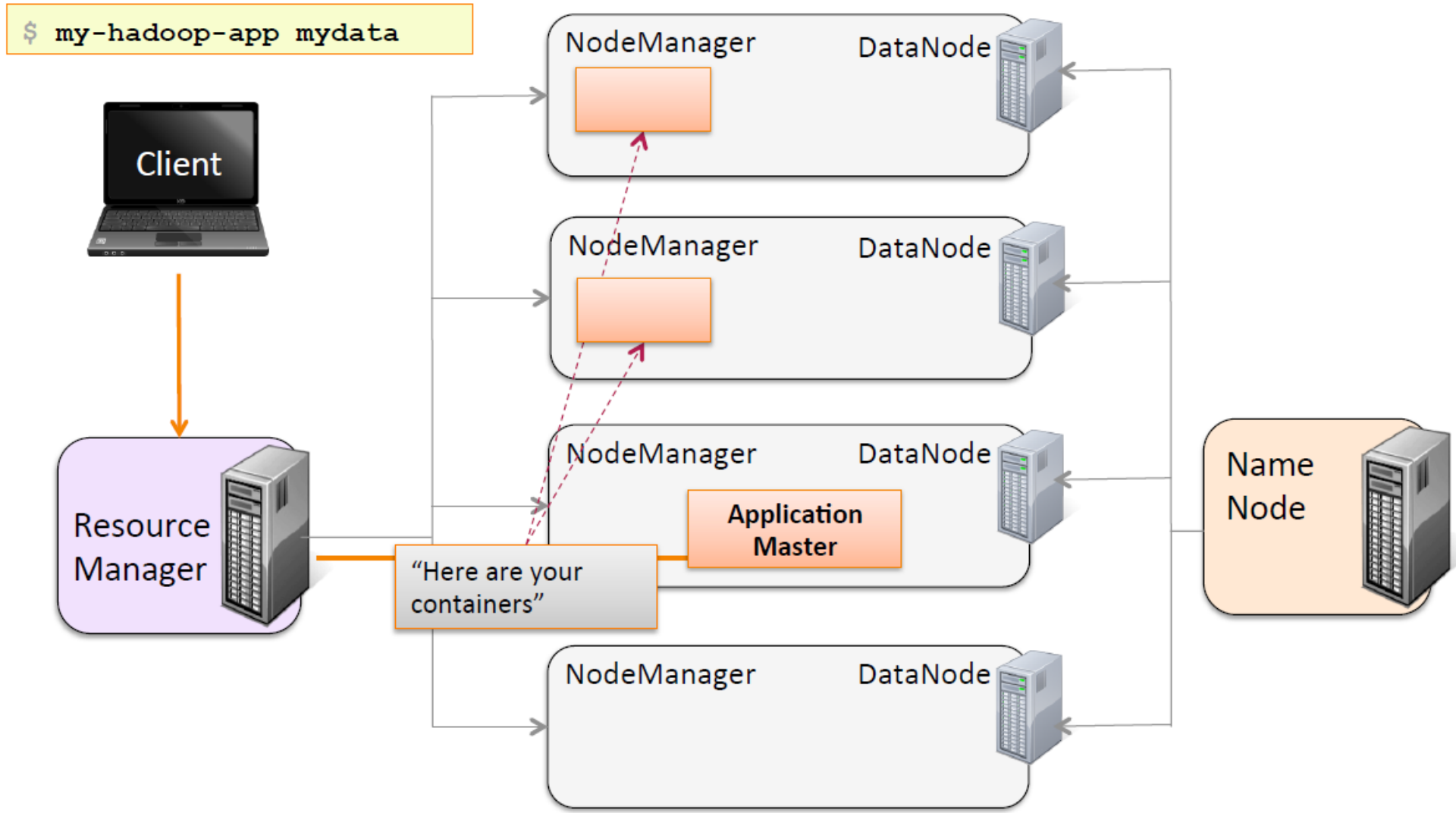
Aurelius™ CORPORATE SOLUTIONS

Running a YARN Application: 3

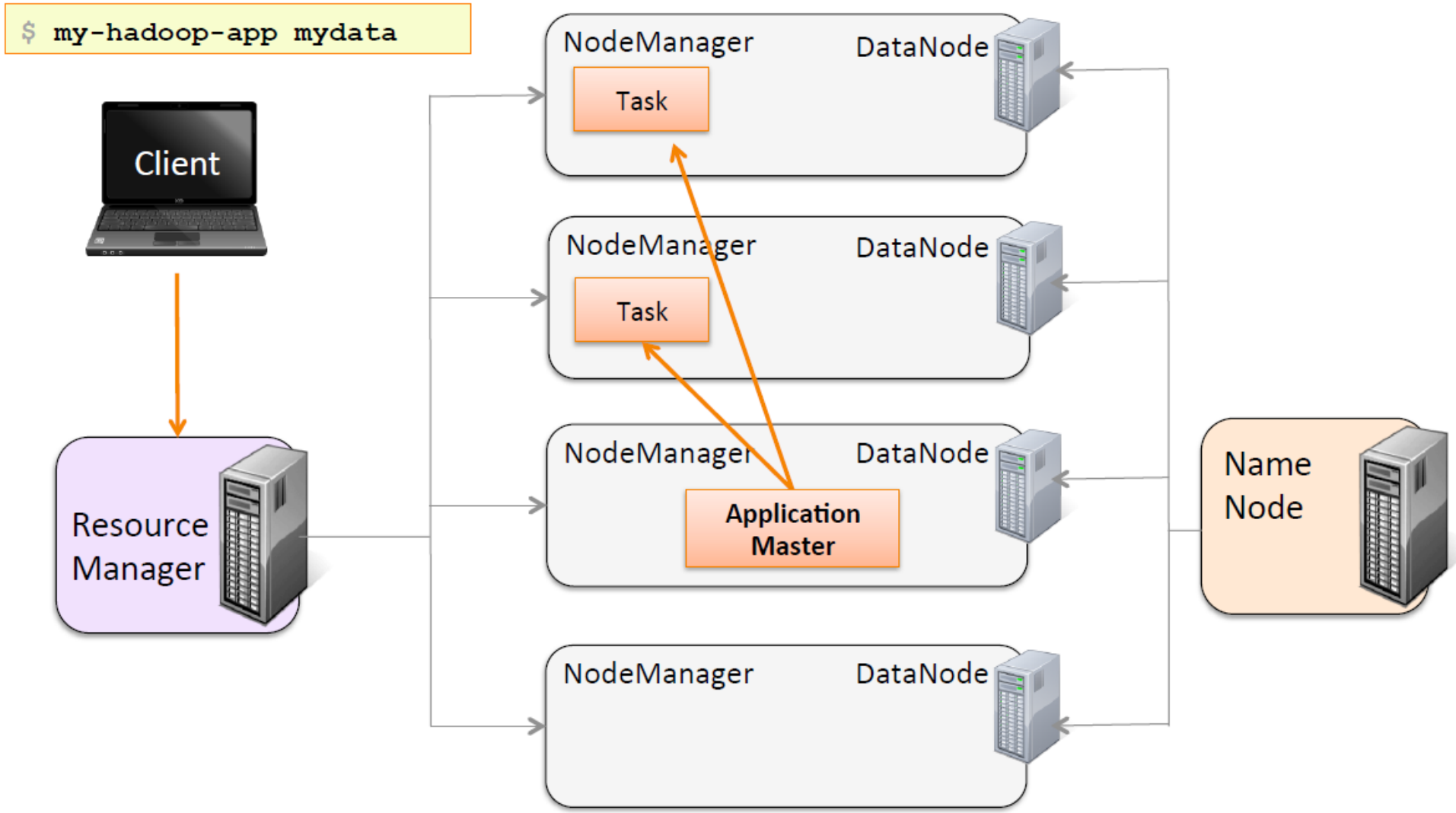


Aurelius™ CORPORATE SOLUTIONS

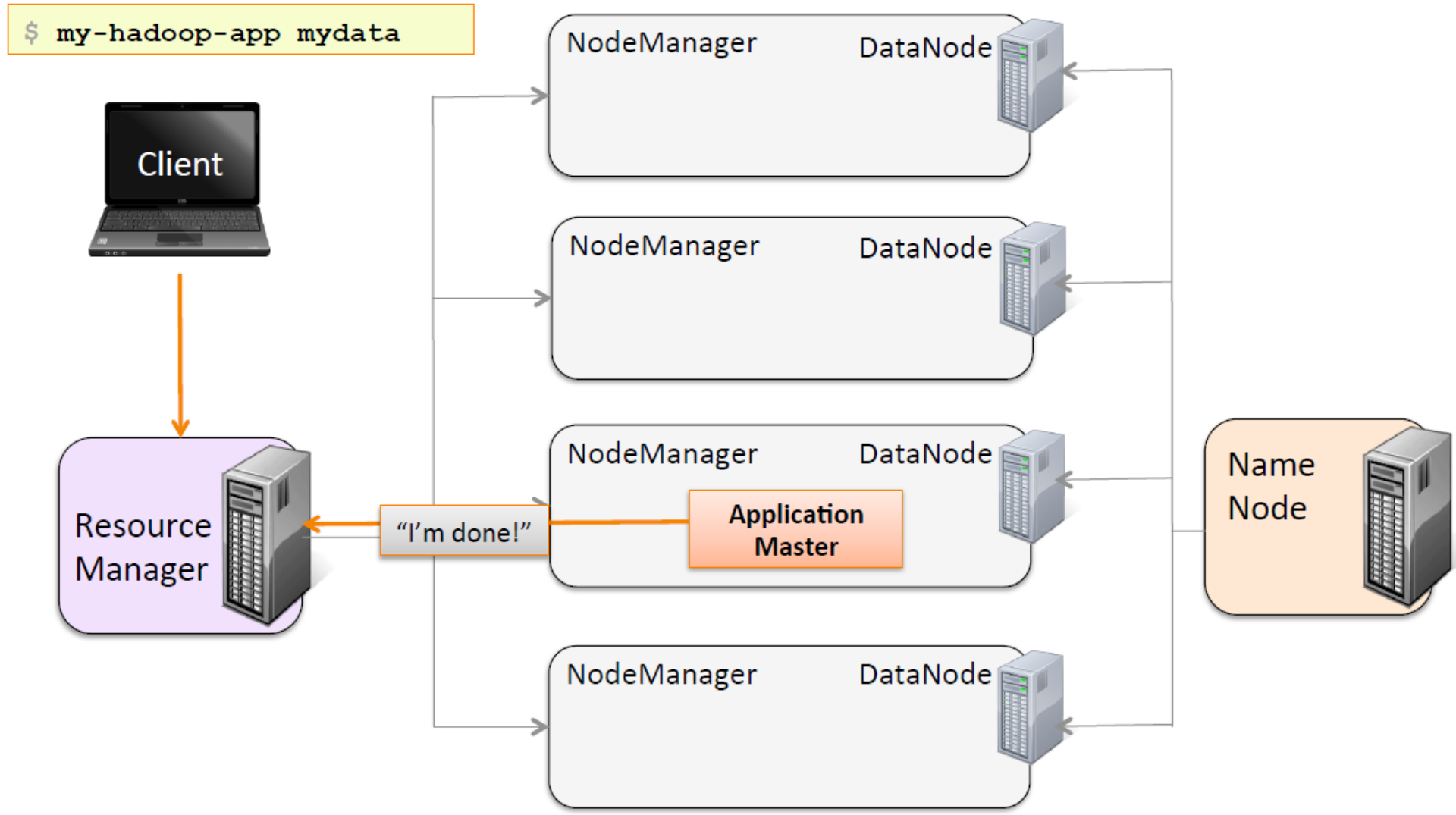
Running a YARN Application: 4



Running a YARN Application: 5

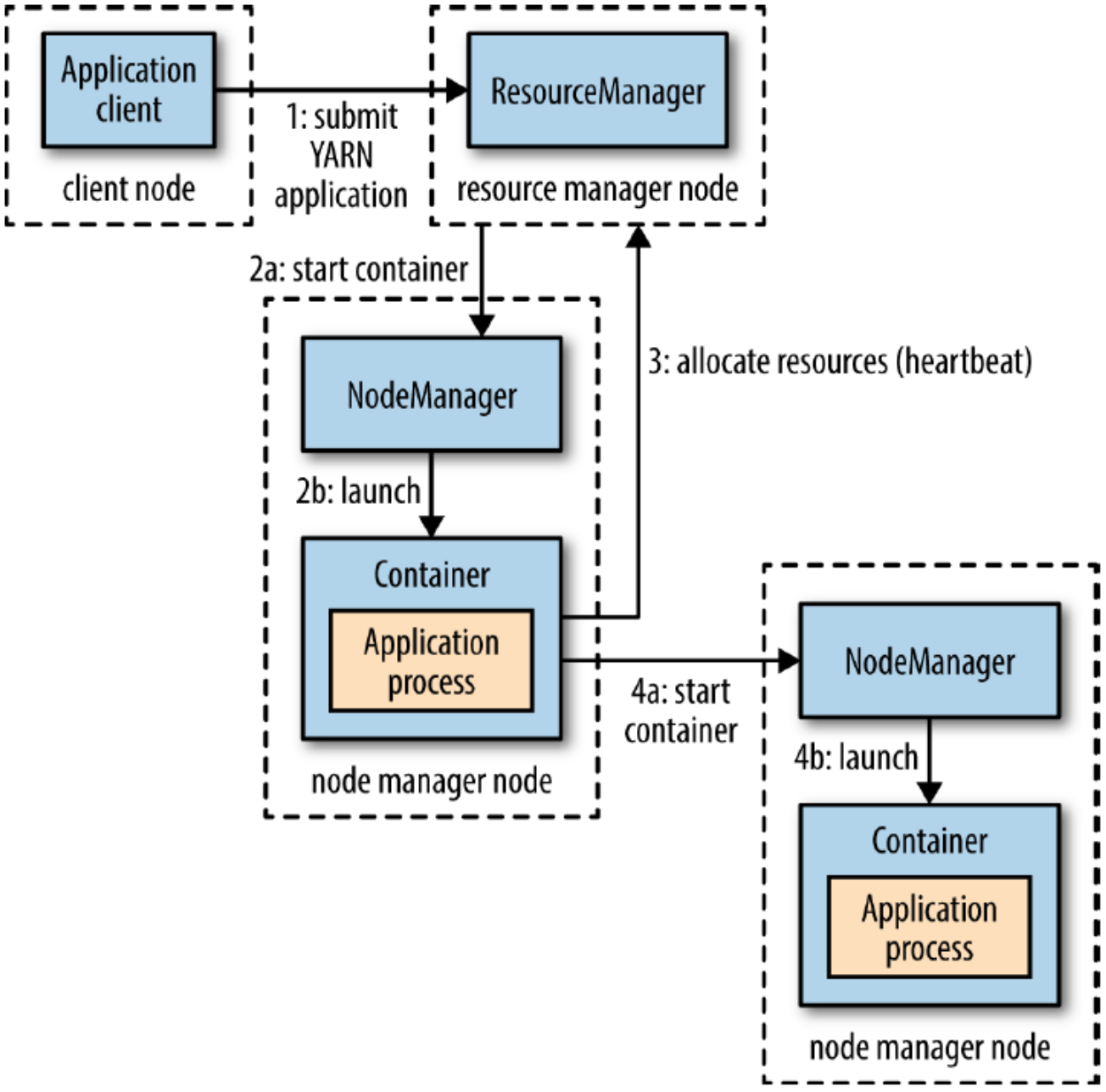


Running a YARN Application: 6

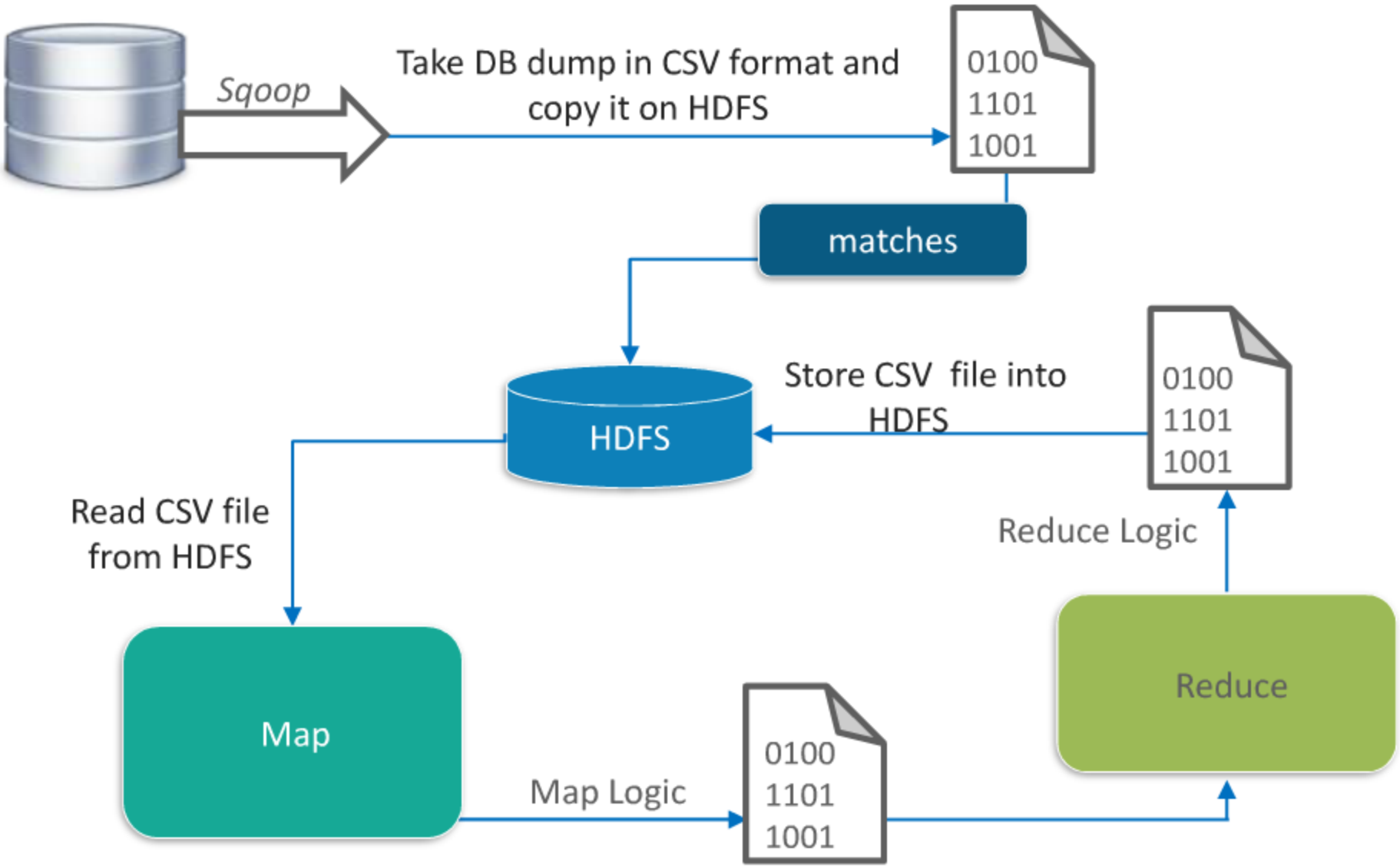


Aurelius™ CORPORATE SOLUTIONS

Running a YARN Application: Summary

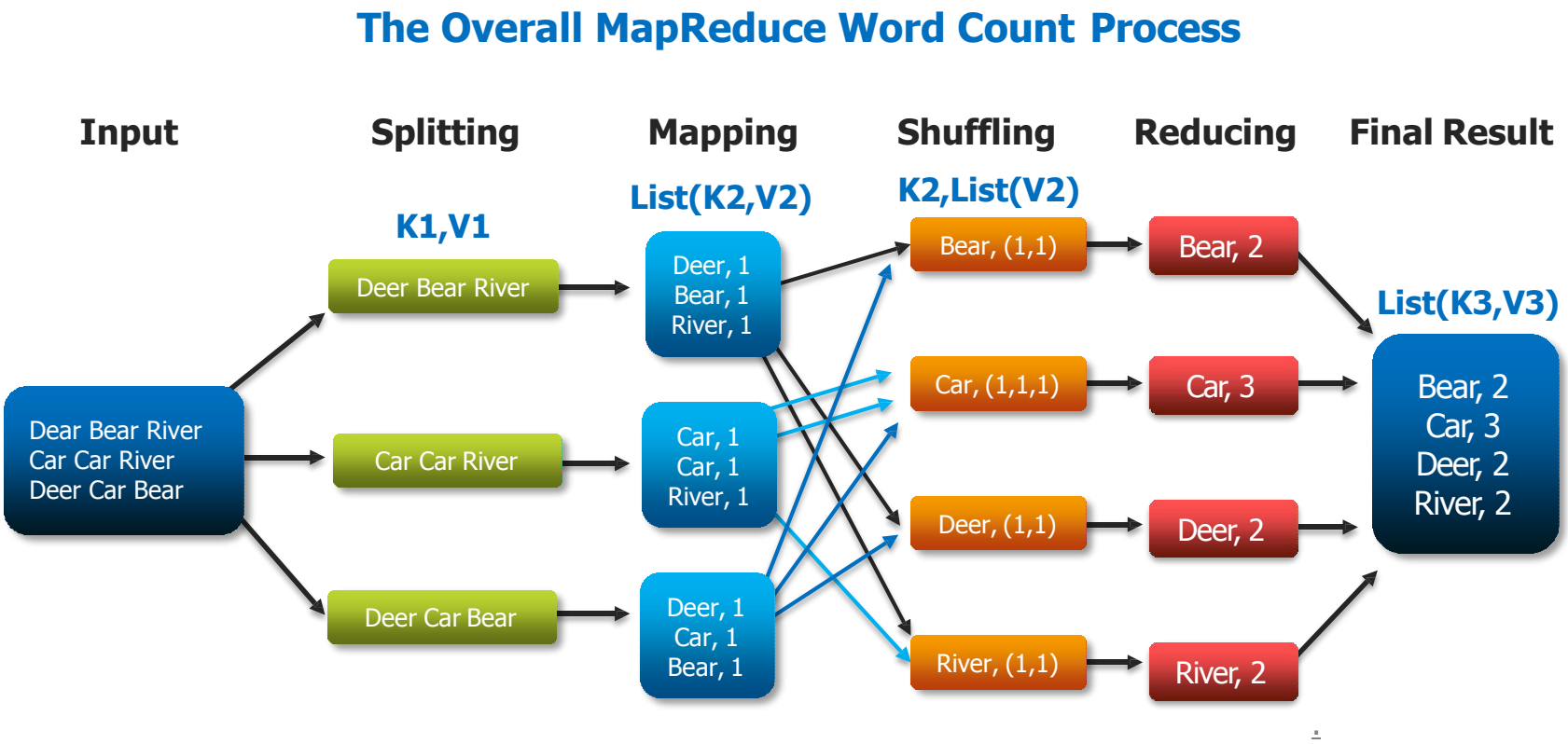


Typical Use Case for Map Reduce

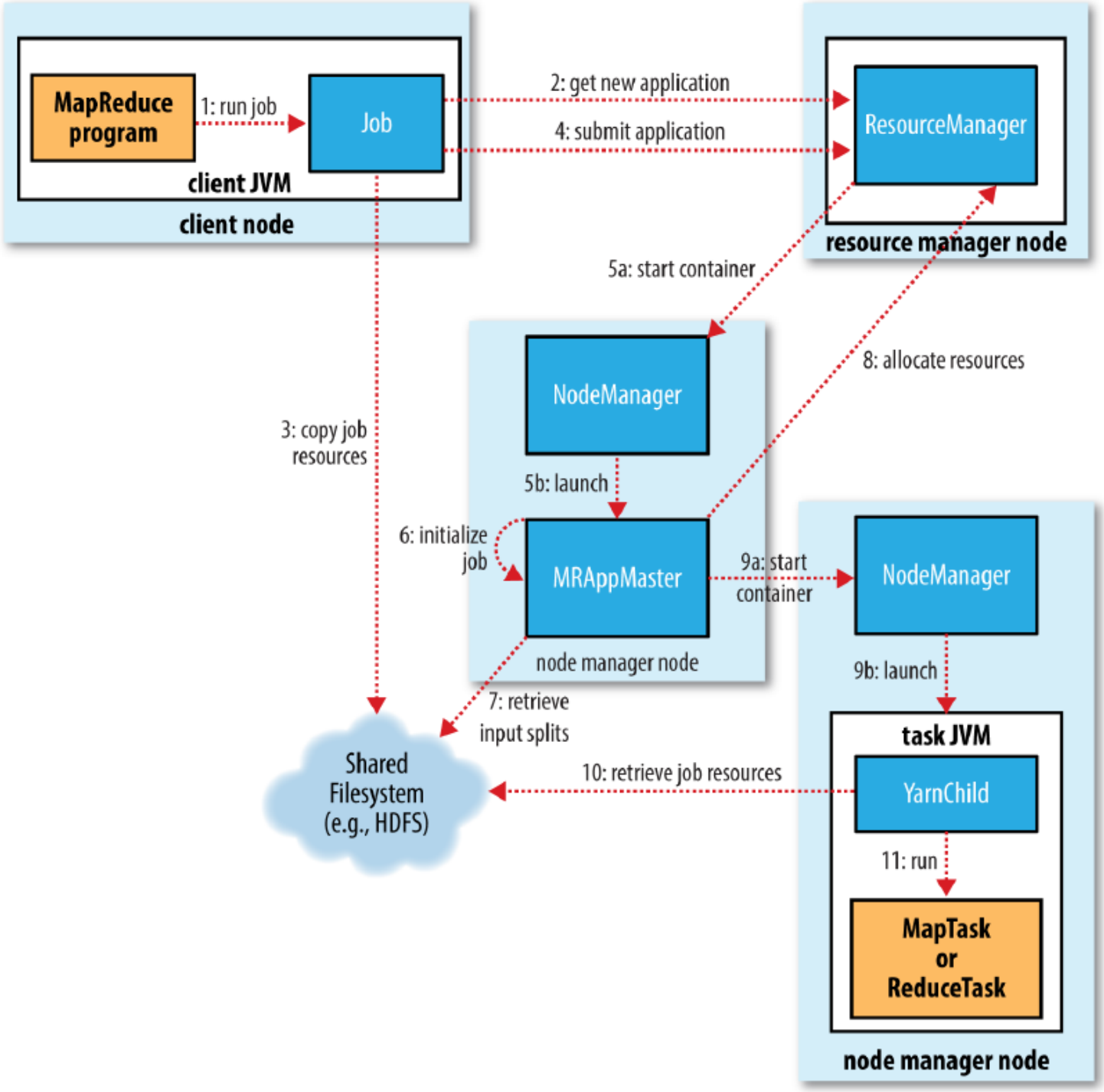


Aurelius™ CORPORATE SOLUTIONS

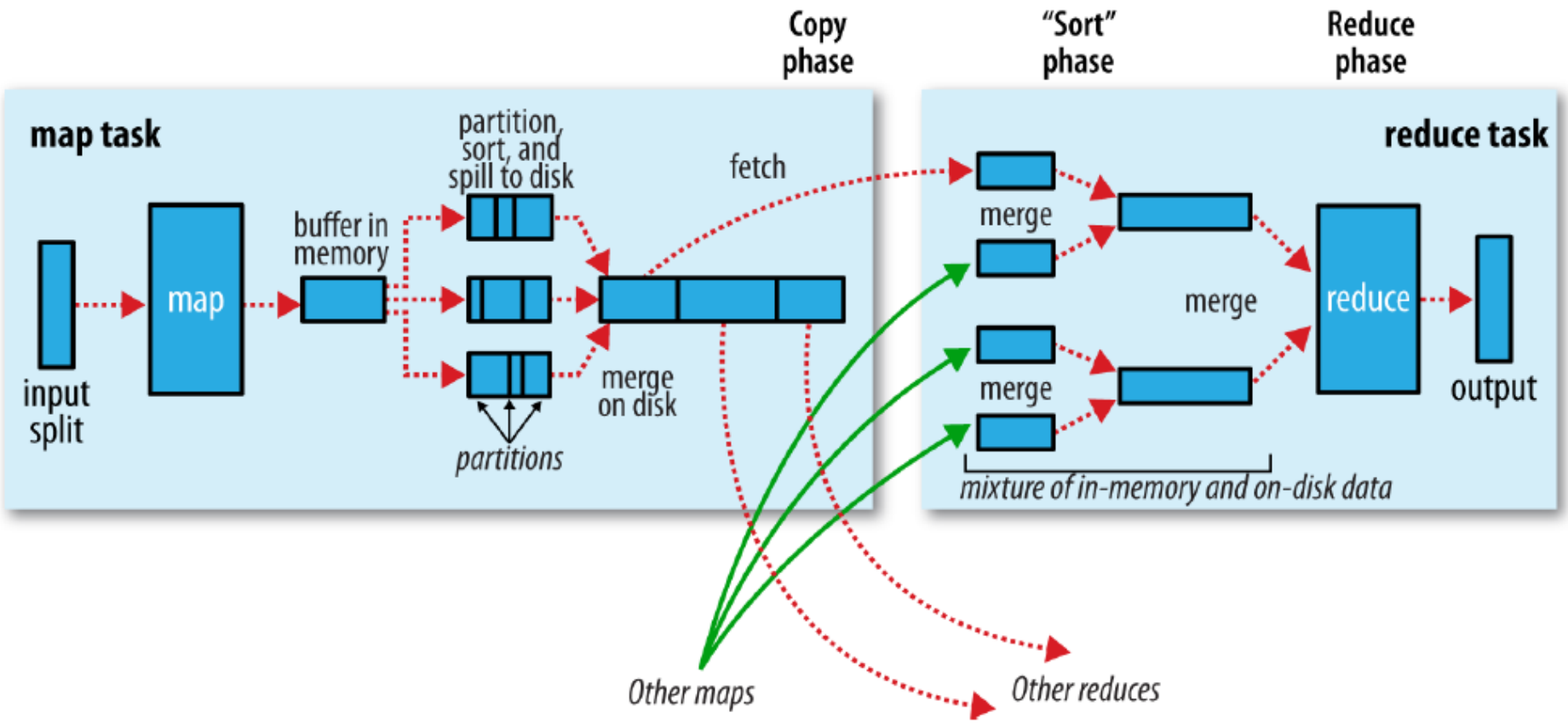
MapReduce Paradigm



Aurelius™ CORPORATE SOLUTIONS

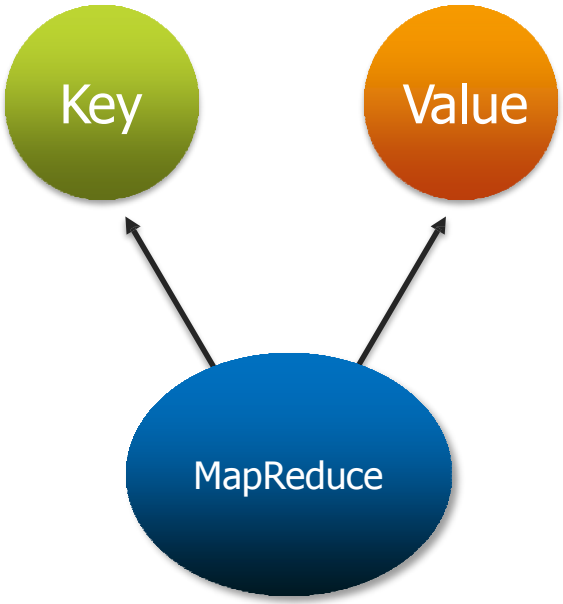

MapReduce program
1: run job
Job
client JVM
client node
2: get new application
4: submit application
ResourceManager
resource manager node
5a: start container
NodeManager
8: allocate resources
3: copy job resources
5b: launch
6: initialize job
MRAppMaster
node manager node
9a: start container
NodeManager
9b: launch
task JVM
YarnChild
11: run
MapTask or ReduceTask
node manager node
7: retrieve input splits
Shared Filesystem (e.g., HDFS)
10: retrieve job resources

Shuffle & Sort



Aurelius™ CORPORATE SOLUTIONS

Anatomy of a MapReduce Program



Map:

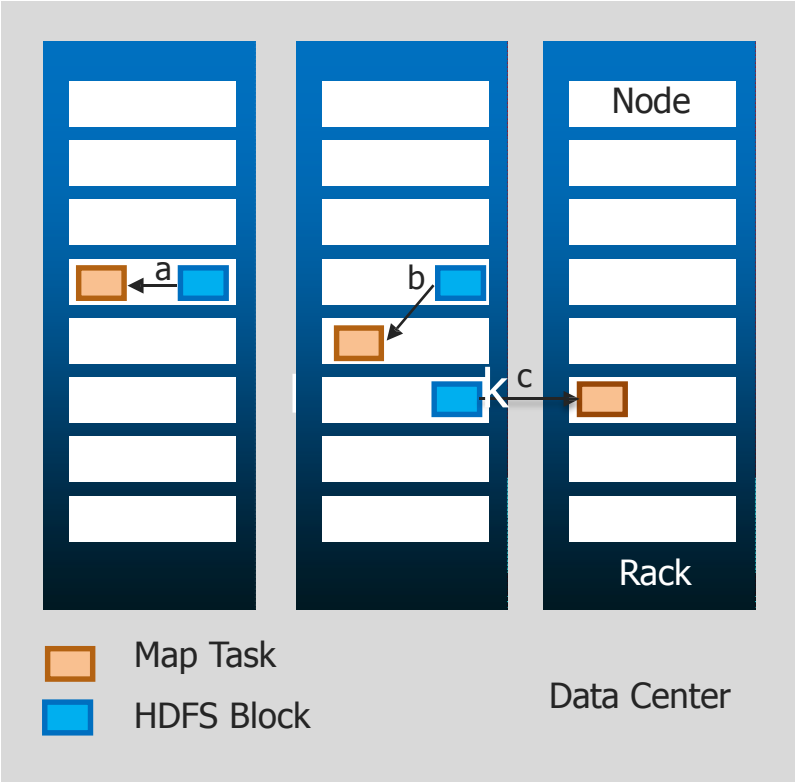
(K1, V1) → List (K2, V2)

Reduce:

(K2, list (V2)) → List (K3, V3)

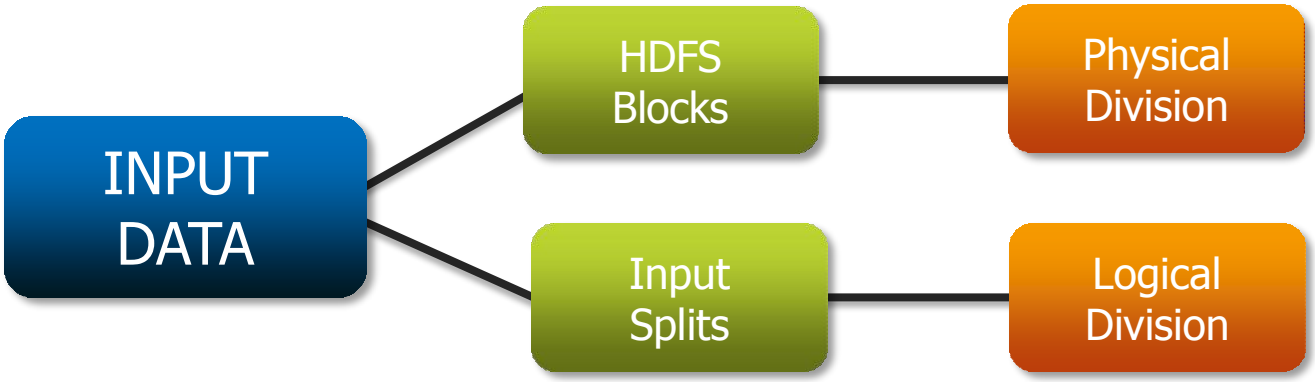
Why MapReduce?

- ✓ Two biggest Advantages:
  - ✓ Taking processing to the data
  - ✓ Processing data in parallel

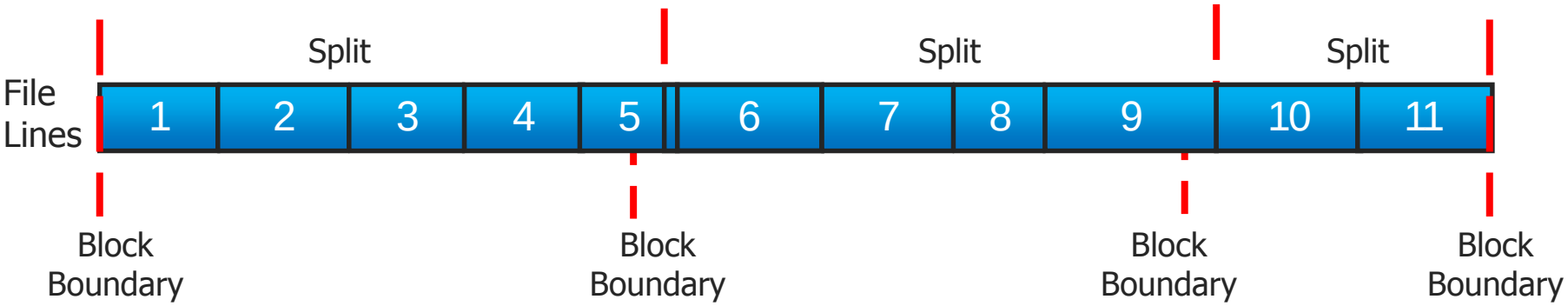


Aurelius™ CORPORATE SOLUTIONS

Input Splits



Relation Between Input Splits and HDFS Blocks



- ✓ Logical records do not fit neatly into the HDFS blocks.
- ✓ Logical records are lines that cross the boundary of the blocks.
- ✓ First split contains line 5 although it spans across blocks.

```
public abstract class InputSplit {
  public abstract long getLength() throws IOException, InterruptedException;
  public abstract String[] getLocations() throws IOException,
      InterruptedException;
}
```


Aurelius™ CORPORATE SOLUTIONS

MapReduce Job Submission Flow

Input data is distributed to nodes

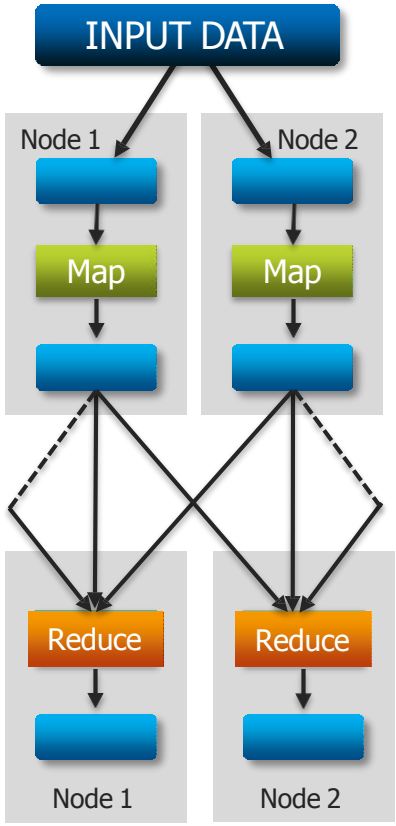
Each map task works on a "split" of data

Mapper outputs intermediate data

Data exchange between nodes in a "shuffle" process

Intermediate data of the same key goes to the same reducer

Reducer output is stored



Aurelius™ CORPORATE SOLUTIONS

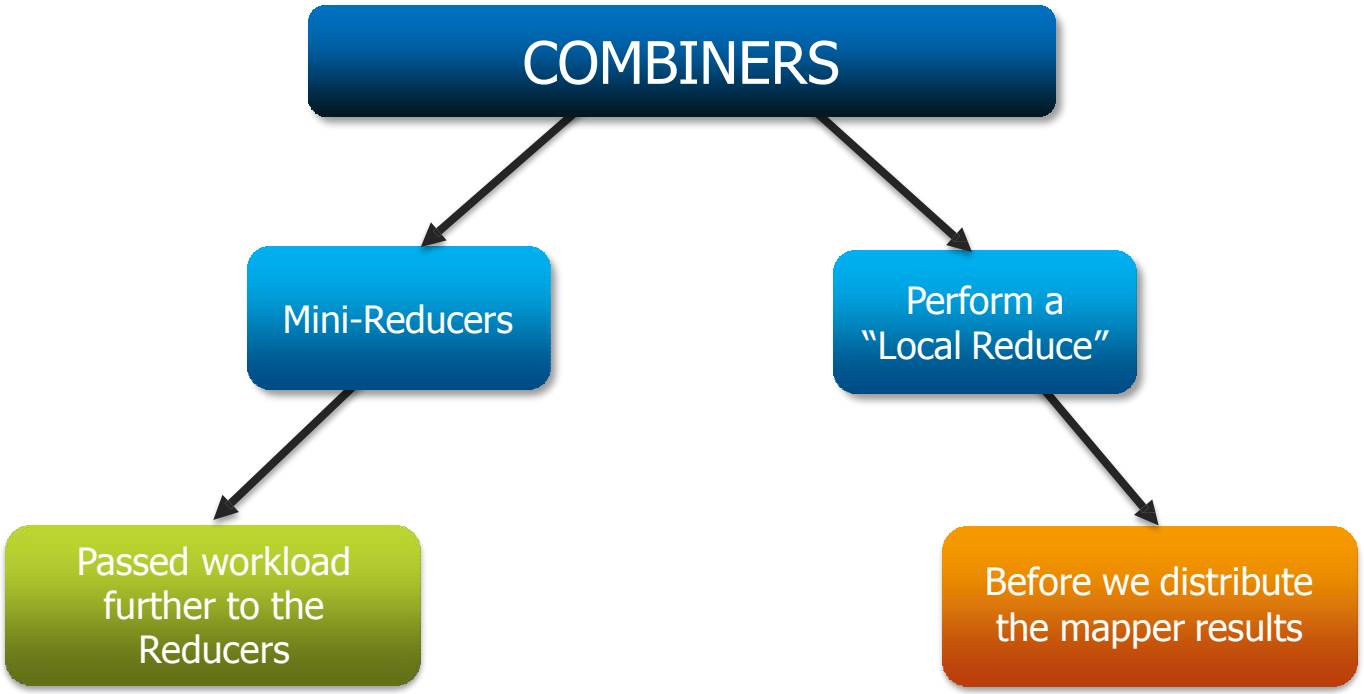
Overview Of MapReduce

Complete view of MapReduce, illustrating combiners and partitioners in addition to Mappers and Reducers

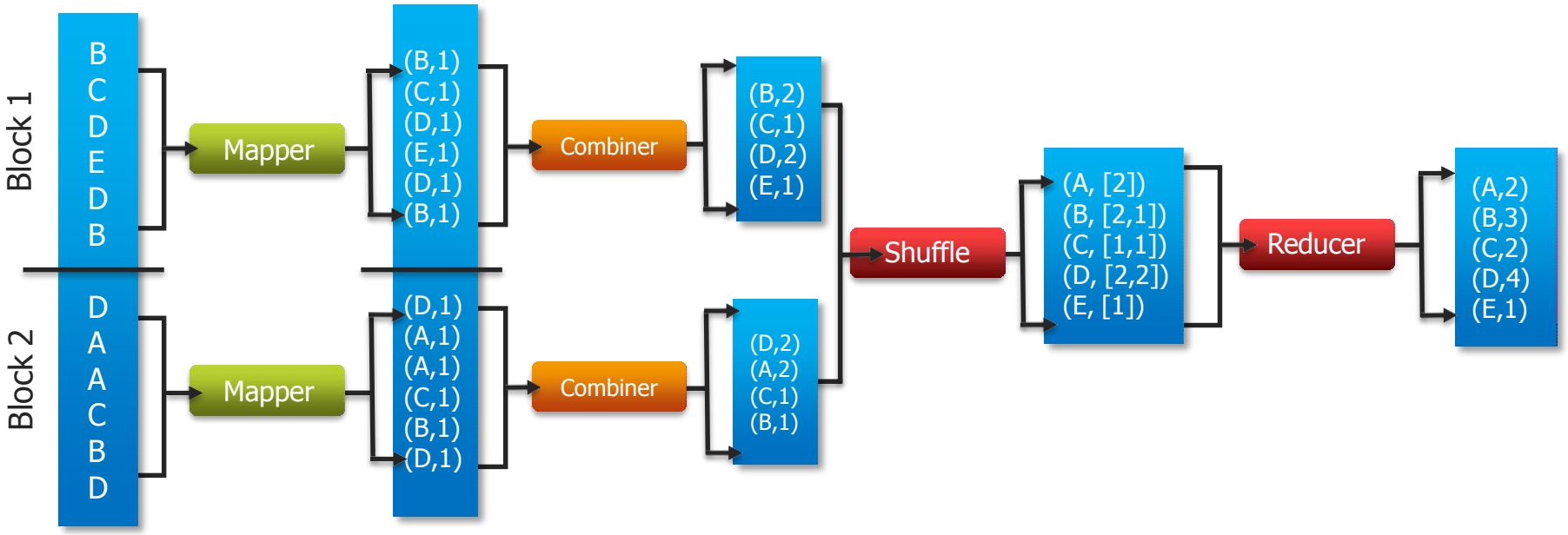
Combiners can be viewed as 'mini-reducers' in the Map phase.

Partitioners determine which reducer is responsible for a particular key.

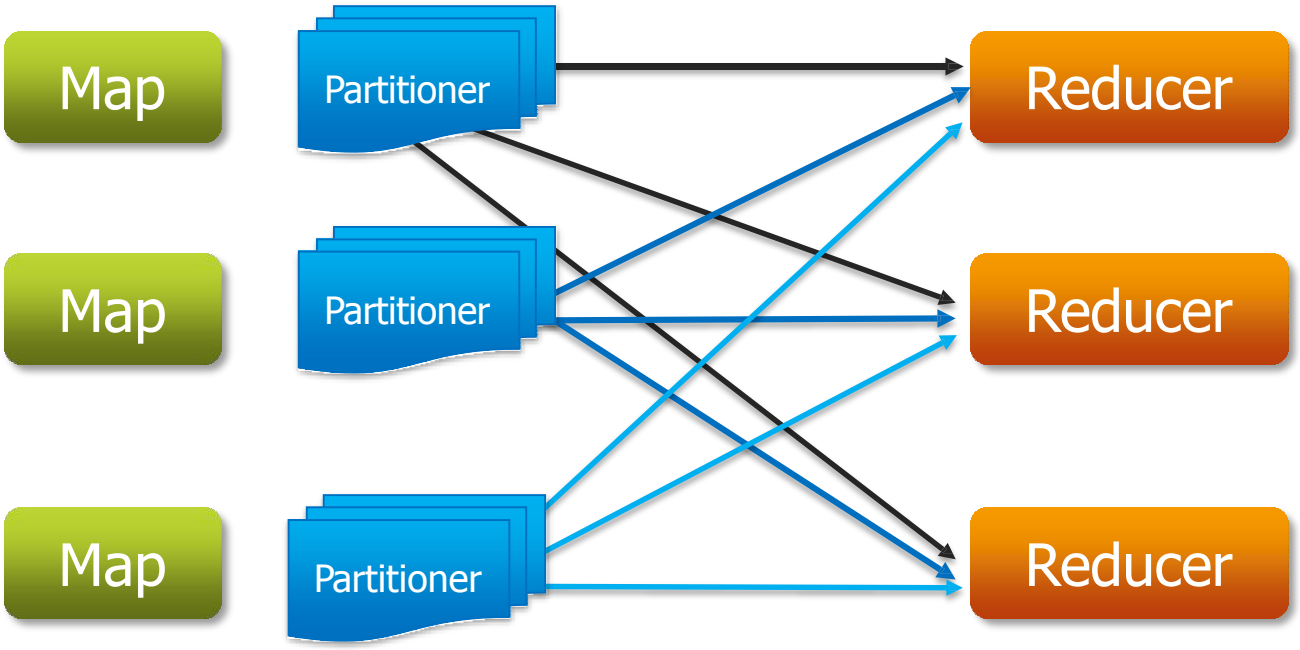
Combiner - Local Reduce



Combiner



Partitioner – Redirecting Output from Mapper



Aurelius™ CORPORATE SOLUTIONS

MapReduce In Course Project

Demo: MapReduce in Project

Word count program

- Code base is at: http://grepcode.com/file/repo1.maven.org/maven2/org.apache.hadoop/hadoop-mapreduce-examples/2.6.0/org/apache/hadoop/examples/WordCount.java

- Prebuilt JAR file is at: http://repo1.maven.org/maven2/org/apache/hadoop/hadoop-mapreduce-examples/2.6.0/hadoop-mapreduce-examples-2.6.0.jar

WordCount Program Execution

hadoop jar hadoop-mapreduce-examples-2.6.0.jar wordcount /user/root/samplemr/input.txt /user/root/samplemr/wordcountOuput4

```
[root@sandbox-hdp ~]# hadoop jar hadoop-mapreduce-examples-2.6.0.jar wordcount /user/root/samplemr/input.txt /user/root/samplemr/wordcountOuput4
18/03/15 13:43:07 INFO client.RMProxy: Connecting to ResourceManager at sandbox-hdp.hortonworks.com/172.17.0.2:8032
18/03/15 13:43:07 INFO client.AHSProxy: Connecting to Application History server at sandbox-hdp.hortonworks.com/172.17.0.2:10200
18/03/15 13:43:08 INFO input.FileInputFormat: Total input paths to process : 1
18/03/15 13:43:08 INFO mapreduce.JobSubmitter: number of splits:1
18/03/15 13:43:09 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1521118455171_0006
18/03/15 13:43:09 INFO impl.YarnClientImpl: Submitted application application_1521118455171_0006
18/03/15 13:43:09 INFO mapreduce.Job: The url to track the job: http://sandbox-hdp.hortonworks.com:8088/proxy/application_1521118455171_0006/
18/03/15 13:43:09 INFO mapreduce.Job: Running job: job_1521118455171_0006
18/03/15 13:43:16 INFO mapreduce.Job: Job job_1521118455171_0006 running in uber mode : false
18/03/15 13:43:16 INFO mapreduce.Job:  map 0% reduce 0%
18/03/15 13:43:22 INFO mapreduce.Job:  map 100% reduce 0%
18/03/15 13:43:30 INFO mapreduce.Job:  map 100% reduce 100%
18/03/15 13:43:31 INFO mapreduce.Job: Job job_1521118455171_0006 completed successfully
18/03/15 13:43:31 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=42
                FILE: Number of bytes written=305435
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=158
                HDFS: Number of bytes written=24
                HDFS: Number of read operations=6
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=3517
                Total time spent by all reduces in occupied slots (ms)=4948
                Total time spent by all map tasks (ms)=3517
                Total time spent by all reduce tasks (ms)=4948
                Total vcore-milliseconds taken by all map tasks=3517
                Total vcore-milliseconds taken by all reduce tasks=4948
                Total megabyte-milliseconds taken by all map tasks=879250
                Total megabyte-milliseconds taken by all reduce tasks=1237000
```

WordCount Program Execution Cont.

```
        Map-Reduce Framework
                Map input records=3
                Map output records=4
                Map output bytes=40
                Map output materialized bytes=42
                Input split bytes=133
                Combine input records=4
                Combine output records=3
                Reduce input groups=3
                Reduce shuffle bytes=42
                Reduce input records=3
                Reduce output records=3
                Spilled Records=6
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=271
                CPU time spent (ms)=1510
                Physical memory (bytes) snapshot=341209088
                Virtual memory (bytes) snapshot=4283138048
                Total committed heap usage (bytes)=151519232
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=25
        File Output Format Counters
                Bytes Written=24
[root@sandbox-hdp ~]#
```

WordCount Program Execution: Input & Output

[root@sandbox-hdp ~]# cat input.txt

```
[root@sandbox-hdp ~]# cat input.txt
hello world
hello again

[root@sandbox-hdp ~]#
```

hdfs dfs -ls /user/root/samplemr/wordcountOuput4

```
[root@sandbox-hdp ~]# hdfs dfs -ls /user/root/samplemr/wordcountOuput4
Found 2 items
-rw-r--r--   1 root hdfs          0 2018-03-15 13:43 /user/root/samplemr/wordcountOuput4/_SUCCESS
-rw-r--r--   1 root hdfs         24 2018-03-15 13:43 /user/root/samplemr/wordcountOuput4/part-r-00000
[root@sandbox-hdp ~]# hdfs dfs -cat /user/root/samplemr/wordcountOuput4/part-r-00000
again   1
hello   2
world   1
[root@sandbox-hdp ~]#
```

Dump of a MR Job

```
13/08/03 00:58:40 WARN mapred.JobClient: Use GenericOptionsParser for parsing th
e arguments. Applications should implement Tool for the same.
13/08/03 00:58:40 INFO mapred.FileInputFormat: Total input paths to process : 1
13/08/03 00:58:40 INFO mapred.JobClient: Running job: job_201308022025_0003
13/08/03 00:58:41 INFO mapred.JobClient:  map 0% reduce 0%
13/08/03 00:58:44 INFO mapred.JobClient:  map 100% reduce 0%
13/08/03 00:58:51 INFO mapred.JobClient:  map 100% reduce 11%
13/08/03 00:58:52 INFO mapred.JobClient:  map 100% reduce 66%
13/08/03 00:58:59 INFO mapred.JobClient:  map 100% reduce 100%
13/08/03 00:58:59 INFO mapred.JobClient: Job complete: job_201308022025_0003
13/08/03 00:58:59 INFO mapred.JobClient: Counters: 23
13/08/03 00:58:59 INFO mapred.JobClient:   Job Counters
13/08/03 00:58:59 INFO mapred.JobClient:     Launched reduce tasks=3
13/08/03 00:58:59 INFO mapred.JobClient:     SLOTS_MILLIS_MAPS=4053
13/08/03 00:58:59 INFO mapred.JobClient:     Total time spent by all reduces wai
ting after reserving slots (ms)=0
13/08/03 00:58:59 INFO mapred.JobClient:     Total time spent by all maps waitin
g after reserving slots (ms)=0
13/08/03 00:58:59 INFO mapred.JobClient:     Launched map tasks=2
13/08/03 00:58:59 INFO mapred.JobClient:     Data-local map tasks=2
13/08/03 00:58:59 INFO mapred.JobClient:     SLOTS_MILLIS_REDUCES=23684
13/08/03 00:58:59 INFO mapred.JobClient:   FileSystemCounters
13/08/03 00:58:59 INFO mapred.JobClient:     FILE_BYTES_READ=81770
13/08/03 00:58:59 INFO mapred.JobClient:     HDFS_BYTES_READ=136111
13/08/03 00:58:59 INFO mapred.JobClient:     FILE_BYTES_WRITTEN=429317
13/08/03 00:58:59 INFO mapred.JobClient:     HDFS_BYTES_WRITTEN=61194
13/08/03 00:58:59 INFO mapred.JobClient:   Map-Reduce Framework
13/08/03 00:58:59 INFO mapred.JobClient:     Reduce input groups=3586
13/08/03 00:58:59 INFO mapred.JobClient:     Combine output records=4027
13/08/03 00:58:59 INFO mapred.JobClient:     Map input records=2403
13/08/03 00:58:59 INFO mapred.JobClient:     Reduce shuffle bytes=81788
13/08/03 00:58:59 INFO mapred.JobClient:     Reduce output records=3586
13/08/03 00:58:59 INFO mapred.JobClient:     Spilled Records=8054
13/08/03 00:58:59 INFO mapred.JobClient:     Map output bytes=151013
13/08/03 00:58:59 INFO mapred.JobClient:     Map input bytes=132663
13/08/03 00:58:59 INFO mapred.JobClient:     Combine input records=11037
13/08/03 00:58:59 INFO mapred.JobClient:     Map output records=11037
13/08/03 00:58:59 INFO mapred.JobClient:     SPLIT_RAW_BYTES=146
13/08/03 00:58:59 INFO mapred.JobClient:     Reduce input records=4027
```

Dump of a MR Job

```
13/08/03 00:58:40 WARN mapred.JobClient: Use GenericOptionsParser for parsing th
e arguments. Applications should implement Tool for the same.
13/08/03 00:58:40 INFO mapred.FileInputFormat: Total input paths to process : 1
13/08/03 00:58:40 INFO mapred.JobClient: Running job: job_201308022025_0003
13/08/03 00:58:41 INFO mapred.JobClient:  map 0% reduce 0%
13/08/03 00:58:44 INFO mapred.JobClient:  map 100% reduce 0%
13/08/03 00:58:51 INFO mapred.JobClient:  map 100% reduce 11%
13/08/03 00:58:52 INFO mapred.JobClient:  map 100% reduce 66%
13/08/03 00:58:59 INFO mapred.JobClient:  map 100% reduce 100%
13/08/03 00:58:59 INFO mapred.JobClient: Job complete: job_201308022025_0003
13/08/03 00:58:59 INFO mapred.JobClient: Counters: 23
13/08/03 00:58:59 INFO mapred.JobClient:   Job Counters
13/08/03 00:58:59 INFO mapred.JobClient:     Launched reduce tasks=3
13/08/03 00:58:59 INFO mapred.JobClient:     SLOTS_MILLIS_MAPS=4053
13/08/03 00:58:59 INFO mapred.JobClient:     Total time spent by all reduces wai
ting after reserving slots (ms)=0
13/08/03 00:58:59 INFO mapred.JobClient:     Total time spent by all maps waitin
g after reserving slots (ms)=0
13/08/03 00:58:59 INFO mapred.JobClient:     Launched map tasks=2
13/08/03 00:58:59 INFO mapred.JobClient:     Data-local map tasks=2
13/08/03 00:58:59 INFO mapred.JobClient:     SLOTS_MILLIS_REDUCES=23684
13/08/03 00:58:59 INFO mapred.JobClient:   FileSystemCounters
13/08/03 00:58:59 INFO mapred.JobClient:     FILE_BYTES_READ=81770
13/08/03 00:58:59 INFO mapred.JobClient:     HDFS_BYTES_READ=136111
13/08/03 00:58:59 INFO mapred.JobClient:     FILE_BYTES_WRITTEN=429317
13/08/03 00:58:59 INFO mapred.JobClient:     HDFS_BYTES_WRITTEN=61194
13/08/03 00:58:59 INFO mapred.JobClient:   Map-Reduce Framework
13/08/03 00:58:59 INFO mapred.JobClient:     Reduce input groups=3586
13/08/03 00:58:59 INFO mapred.JobClient:     Combine output records=4027
13/08/03 00:58:59 INFO mapred.JobClient:     Map input records=2403
13/08/03 00:58:59 INFO mapred.JobClient:     Reduce shuffle bytes=81788
13/08/03 00:58:59 INFO mapred.JobClient:     Reduce output records=3586
13/08/03 00:58:59 INFO mapred.JobClient:     Spilled Records=8054
13/08/03 00:58:59 INFO mapred.JobClient:     Map output bytes=151013
13/08/03 00:58:59 INFO mapred.JobClient:     Map input bytes=132663
13/08/03 00:58:59 INFO mapred.JobClient:     Combine input records=11037
13/08/03 00:58:59 INFO mapred.JobClient:     Map output records=11037
13/08/03 00:58:59 INFO mapred.JobClient:     SPLIT_RAW_BYTES=146
13/08/03 00:58:59 INFO mapred.JobClient:     Reduce input records=4027
```

Question

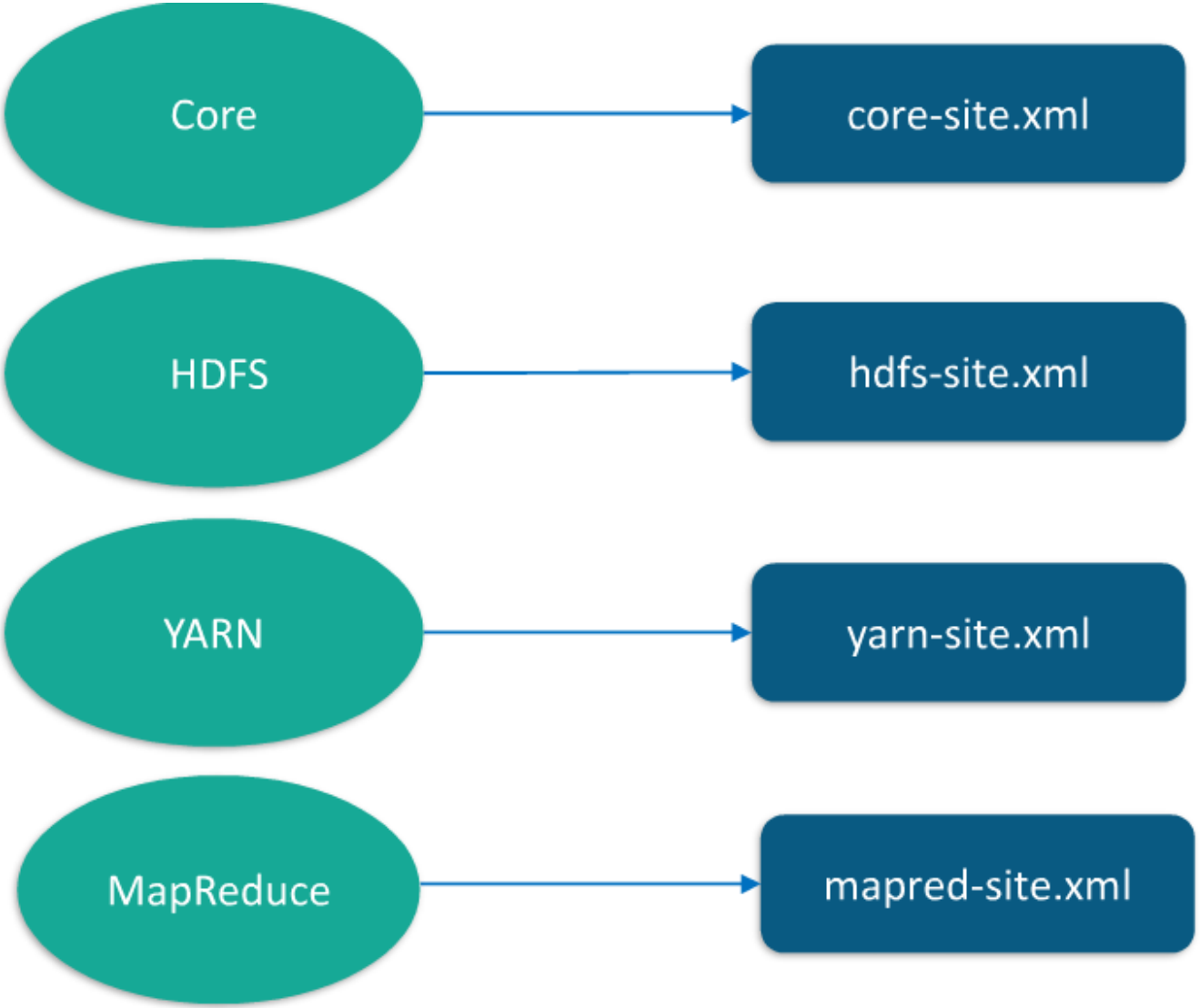
The output of a MR job will be stored on HDFS:

- TRUE
- FALSE

Answer

True. It is stored in different part files for eg – part-m-00000, part-m-00001 and so on. The part files are created on the basis of the block size.

Quick Overview of Hadoop 2.x Configuration files



Aurelius™ CORPORATE SOLUTIONS

core-site.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!-- core-site.xml -->
<configuration>
        <property>
            <name>fs.defaultFS</name>
            <value>hdfs://nameservice1</value>
        </property>
</configuration>
```

The name of the default file system. The url's authority is used to determine the host, port, etc. for a filesystem.

hdfs-site.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!-- hdfs-site.xml -->
<configuration>
        <property>
            <name>dfs.replication</name>
            <value>3</value>
        </property>
        <property>
            <name>dfs.blocksize</name>
            <value>134217728</value>
        </property>
</configuration>
```

Determines the number of replication of blocks allowed in the HDFS(here the specified value is 3).

Determines the size of data blocks in the HDFS(here, the specified value is in bytes – 128 MB).

mapred-site.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!-- mapred-site.xml -->
<configuration>
        <property>
            <name>mapreduce.framework.name</name>
            <value>yarn</value>
        </property>
</configuration>
```

The runtime framework for executing MapReduce jobs. Can be set to local, classic or yarn.

yarn-site.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!-- yarn-site.xml -->
<configuration>
        <property>
            <name>yarn.resourcemanager.ha.enabled</name>
            <value>true</value>
        </property>
</configuration>
```

Aurelius™ CORPORATE SOLUTIONS