cloudera
Ask Bigger Questions

Apache Hadoop – A course for undergraduates

Lecture 1

201406

Introduction

Chapter 1.1

Chapter Topics

Introduction

- **Course Objectives**
- About Apache Hadoop

Course Objectives

- **Why is Hadoop needed?**
- **Learn concepts of the Hadoop Distributed File System and MapReduce**
- **Hadoop-able Problems**
- **Core Hadoop technologies and the Hadoop Ecosystem**
- **Developing MapReduce applications**
- **Common MapReduce algorithms**
- **Using Hive and Pig for rapid application development**

Chapter Topics

Introduction

- Course Objectives
- **About Apache Hadoop**

What is Apache Hadoop?

- **A software framework for storing, processing, and analyzing "big data"**
  - Distributed
  - Scalable
  - Fault-tolerant
  - Open source

Facts About Apache Hadoop

- **Open source**
- **Around 60 committers from ~10 companies**
  - Cloudera, Yahoo!, Facebook, Apple, and more
- **Hundreds of contributors writing features, fixing bugs**
- **Many related projects, applications, tools, etc.**

A Large (and Growing) Ecosystem

APACHE HBASE

HUE

Pig

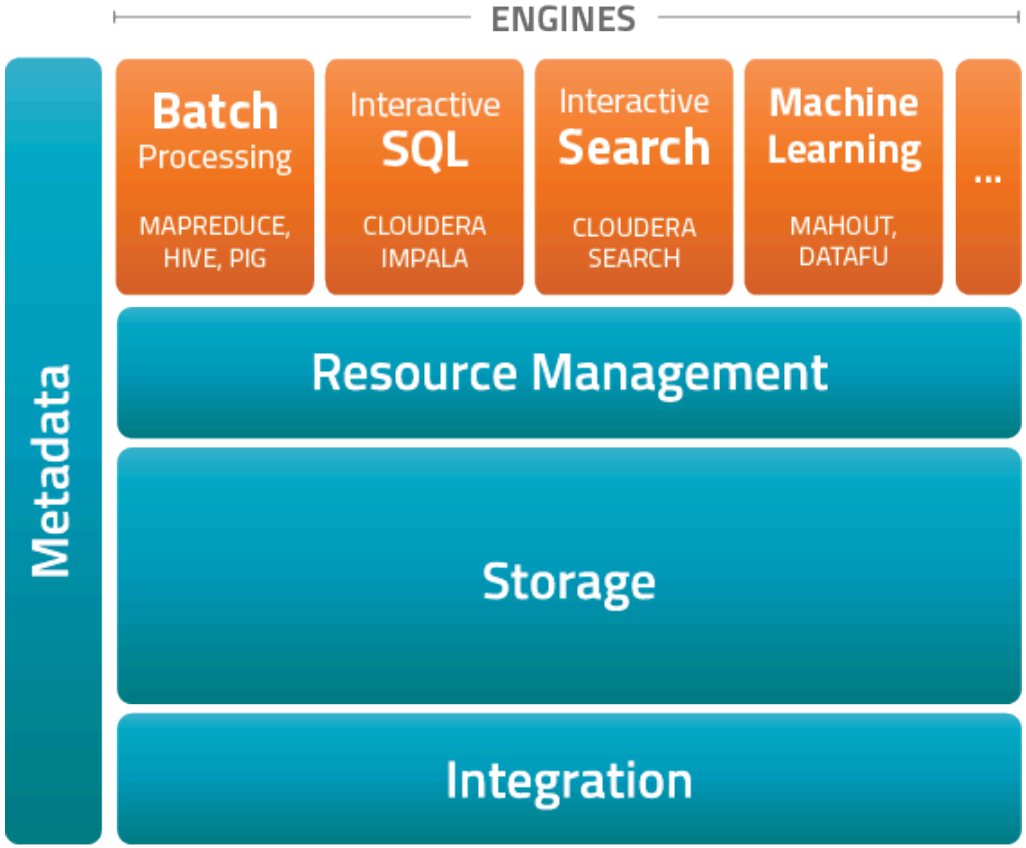
Zookeeper

Impala

Who uses Hadoop?

CDH

- **CDH (Cloudera's Distribution, including Apache Hadoop)**
  - 100% open source distribution of Hadoop and related projects
  - The most complete, tested, and widely-deployed distribution of Hadoop
  - Integrates all the key Hadoop ecosystem projects



Bibliography

The following offer more information on topics discussed in this chapter

- **What is Apache Hadoop?**
  - http://hadoop.apache.org/index.html
- **Who are the committers (i.e. programmers)?**
  - http://hadoop.apache.org/who.html

The Motivation for Hadoop

Chapter 1.2

The Motivation For Hadoop

- **What problems exist with traditional large-scale computing systems?**
- **How does Hadoop address those challenges?**
- **What kinds of data processing and analysis does Hadoop do best?**

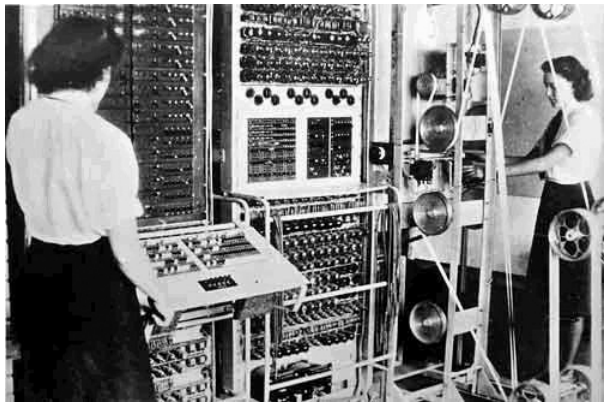
Chapter Topics

The Motivation for Hadoop

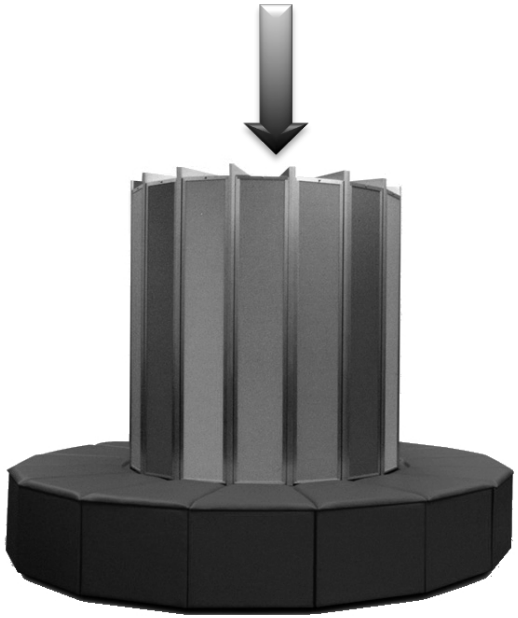
- **Problems with Traditional Large-Scale Systems**
- Requirements for a New Approach
- Hadoop!
- Hadoop-able Problems

Traditional Large-Scale Computation

- **Traditionally, computation has been processor-bound**
  - Relatively small amounts of data
  - Lots of complex processing



- **The early solution: bigger computers**
  - Faster processor, more memory
  - But even this couldn't keep up

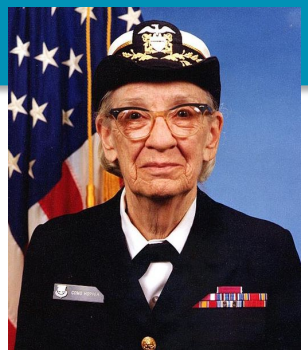


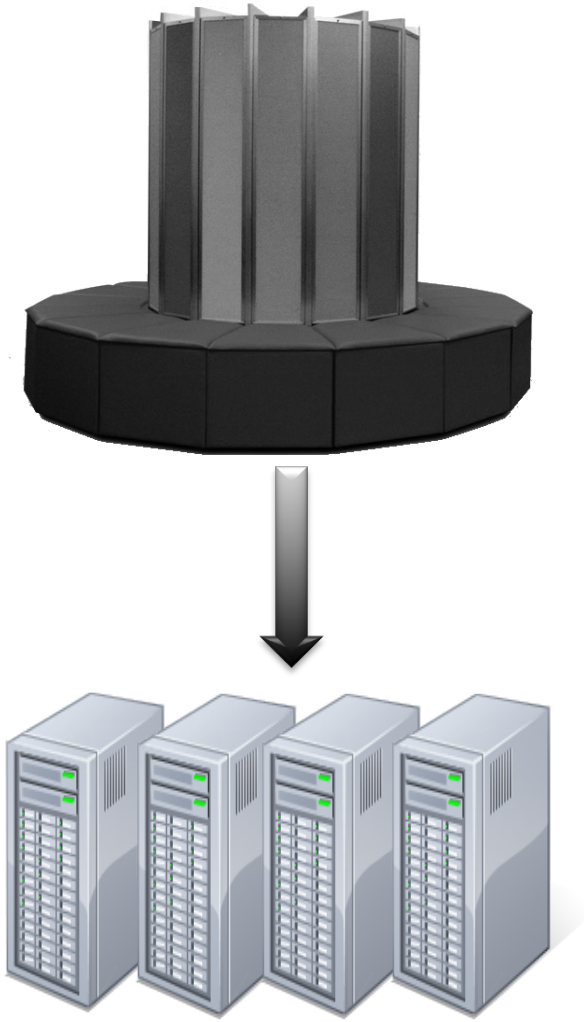
Distributed Systems

- **The better solution: more computers**
  - Distributed systems evolved
  - Use multiple machines for a single job
  - MPI, (Message Passing Interface), for example

"In pioneer days they used oxen for heavy pulling, and when one ox couldn't budge a log, we didn't try to grow a larger ox. We shouldn't be trying for bigger computers, but for *more systems* of computers."

– Grace Hopper





Distributed Systems: Problems

- **Programming for traditional distributed systems is complex**
  - Data exchange requires synchronization
  - Finite bandwidth is available
  - Temporal dependencies are complicated
  - It is difficult to deal with partial failures of the system
- **Ken Arnold, CORBA designer:**
  - "Failure is the defining difference between distributed and local programming, so you have to design distributed systems with the expectation of failure"
    - Developers spend more time designing for failure than they do actually working on the problem itself

CORBA: Common Object Request Broker Architecture

Distributed Systems: Challenges

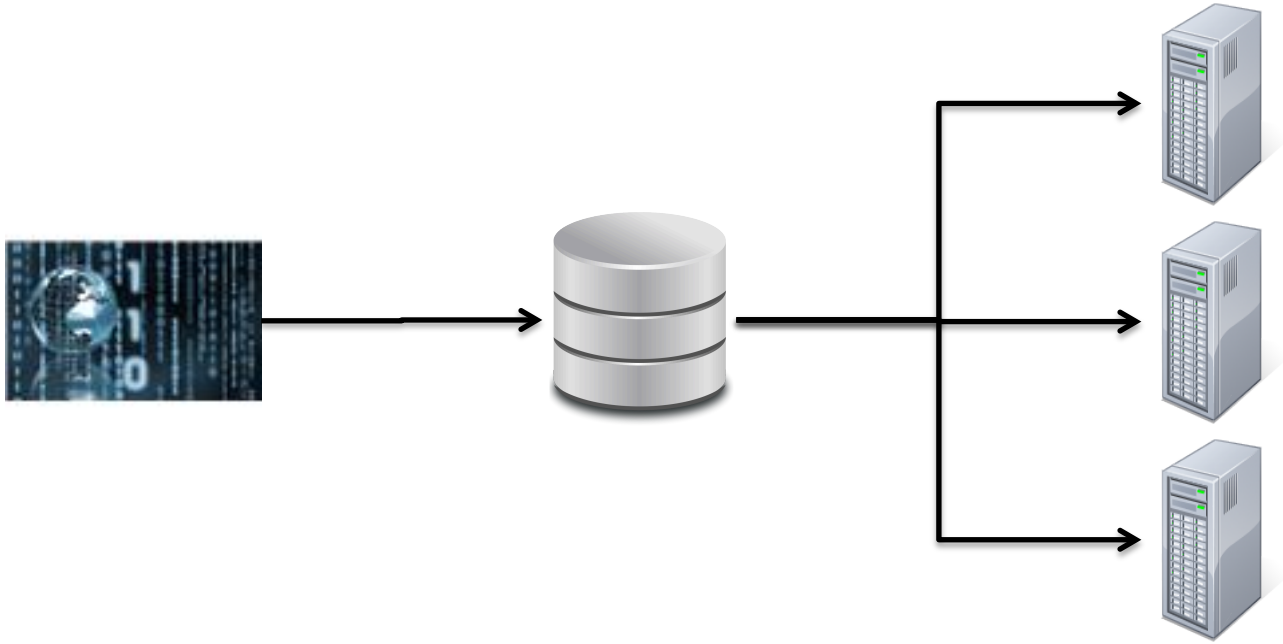
- **Challenges with distributed systems**
  - Programming complexity
    - Keeping data and processes in sync
  - Finite bandwidth
  - Partial failures

Distributed Systems: The Data Bottleneck

- **Moore's Law has held firm for over 40 years**
  - Processing power doubles every two years
  - Processing speed is no longer the problem
- **Getting the data to the processors becomes the bottleneck**
- **Quick calculation**
  - Typical disk data transfer rate: 75MB/sec
  - Time taken to transfer 100GB of data to the processor: approx 22 minutes!
    - Assuming sustained reads
    - Actual time will be worse, since most servers have less than 100GB of RAM available
- **A new approach is needed**

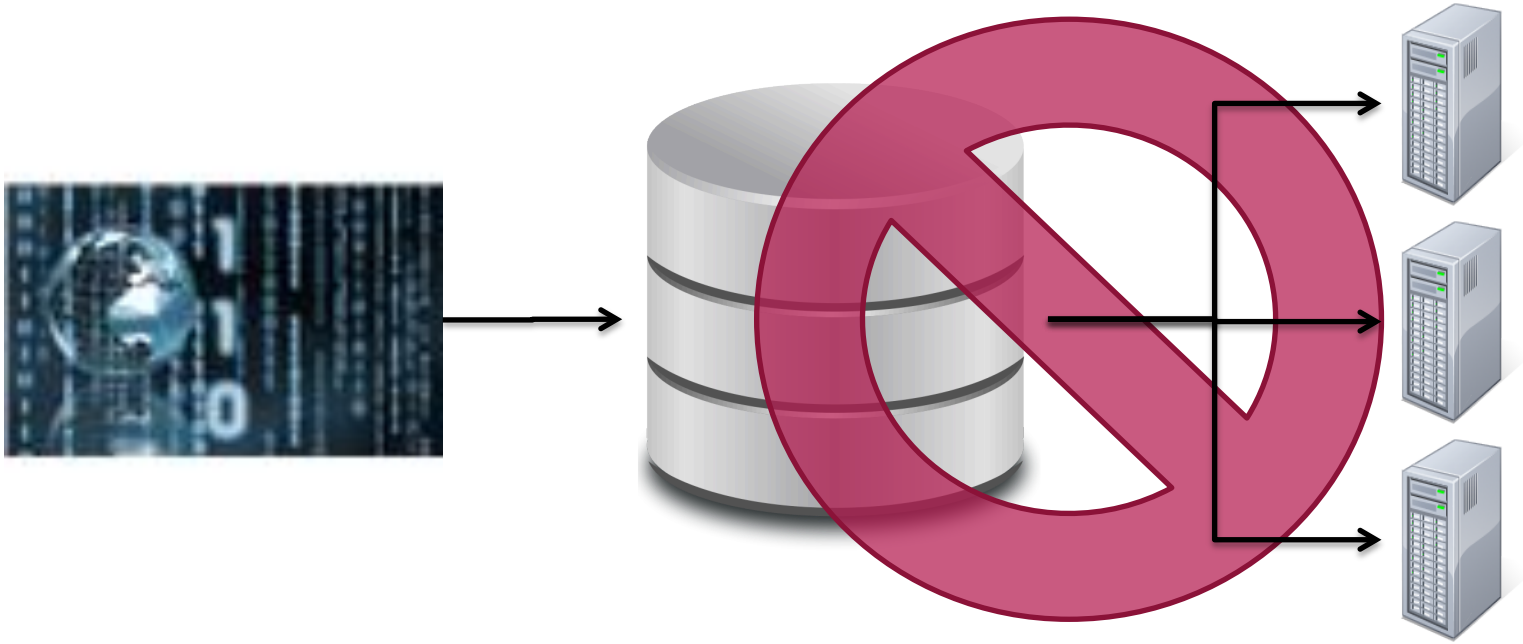
Distributed Systems: The Data Bottleneck (cont'd)

- **Traditionally, data is stored in a central location**
- **Data is copied to processors at runtime**
- **Fine for limited amounts of data**



Distributed Systems: The Data Bottleneck (cont'd)

- **Modern systems have much more data**
  - terabytes+ per day
  - petabytes+ total
- **We need a new approach…**



Chapter Topics

The Motivation for Hadoop

- Problems with Traditional Large-Scale Systems
- **Requirements for a New Approach**
- Hadoop!
- Hadoop-able Problems

Partial Failure Support

- **The system must support partial failure**
  - Failure of a component should result in a graceful degradation of application performance
    - Not complete failure of the entire system

Data Recoverability

- **If a component of the system fails, its workload should be assumed by still-functioning units in the system**
  - Failure should not result in the loss of any data

Component Recovery

- **If a component of the system fails and then recovers, it should be able to rejoin the system**
  - Without requiring a full restart of the entire system

Consistency

- **Component failures during execution of a job should not affect the outcome of the job**

Scalability

- **Adding load to the system should result in a graceful decline in performance of individual jobs**
  - Not failure of the system
- **Increasing resources should support a proportional increase in load capacity**

Chapter Topics

The Motivation for Hadoop

- Problems with Traditional Large-Scale Systems
- Requirements for a New Approach
- **Hadoop!**
- Hadoop-able Problems

Hadoop's History

- **Hadoop is based on work done by Google in the late 1990s/early 2000s**
  - Specifically, on papers describing the Google File System (GFS) published in 2003, and MapReduce published in 2004
- **This work takes a radical new approach to the problem of distributed computing**
  - Meets all the requirements we have for reliability and scalability
- **Core concept: distribute the data as it is initially stored in the system**
  - Individual nodes can work on data local to those nodes
    - No data transfer over the network is required for initial processing

Core Hadoop Concepts

- **Applications are written in high-level code**
  - Developers need not worry about network programming, temporal dependencies or low-level infrastructure
- **Nodes talk to each other as little as possible**
  - Developers should not write code which communicates between nodes
  - 'Shared nothing' architecture
- **Data is spread among machines in advance**
  - Computation happens where the data is stored, wherever possible
    - Data is replicated multiple times on the system for increased availability and reliability

Hadoop: Very High-Level Overview

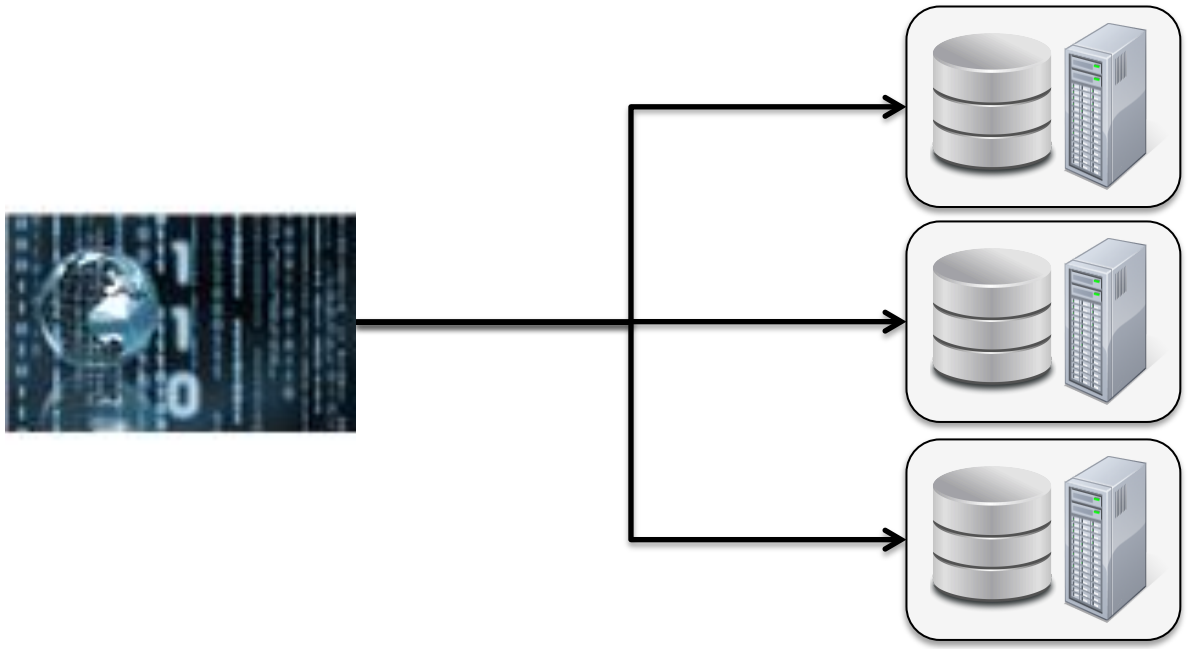
- **When data is loaded into the system, it is split into 'blocks'**
  - Typically 64MB or 128MB
- **Map tasks (the first part of the MapReduce system) work on relatively small portions of data**
  - Typically a single block
- **A master program allocates work to nodes such that a Map task will work on a block of data stored locally on that node whenever possible**
  - Many nodes work in parallel, each on their own part of the overall dataset

Fault Tolerance

- **If a node fails, the master will detect that failure and re-assign the work to a different node on the system**
- **Restarting a task does not require communication with nodes working on other portions of the data**
- **If a failed node restarts, it is automatically added back to the system and assigned new tasks**
- **If a node appears to be running slowly, the master can redundantly execute another instance of the same task**
  - Results from the first to finish will be used
  - Known as 'speculative execution'

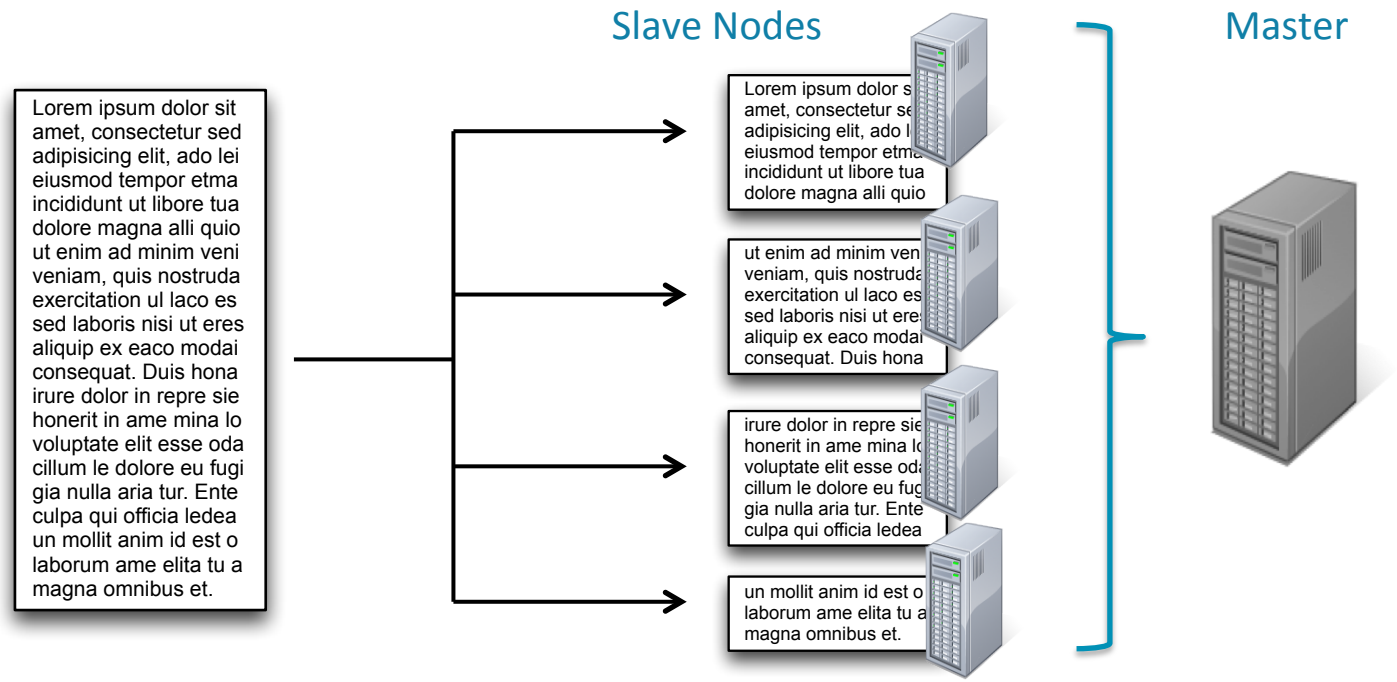
Hadoop

- **A radical new approach to distributed computing**
  - Distribute data when the data is being stored
  - Run computation where the data is stored



Hadoop: Very High-Level Overview

- **Data is split into "blocks" when loaded**
- **Map tasks typically work on a single block**
- **A master program manages tasks**

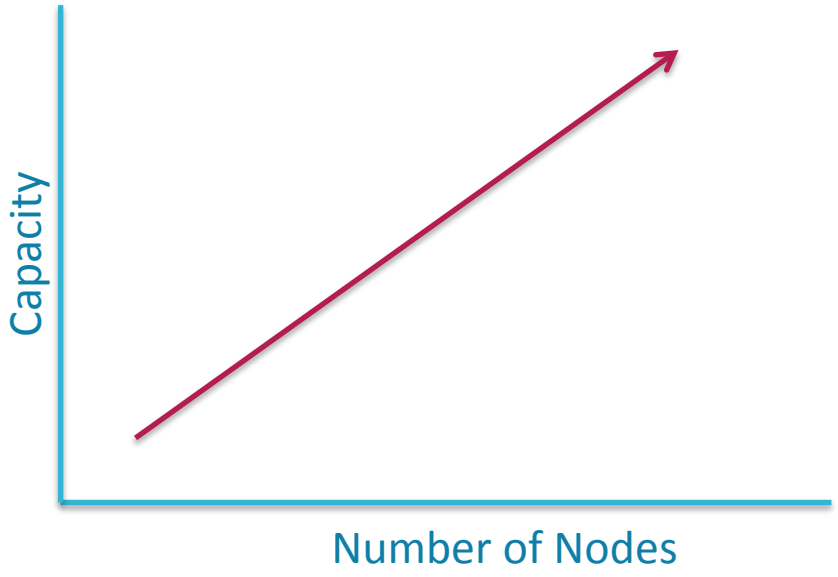


Core Hadoop Concepts

- **Applications are written in high-level code**
- **Nodes talk to each other as little as possible**
- **Data is distributed in advance**
  - Bring the computation to the data
- **Data is replicated for increased availability and reliability**
- **Hadoop is scalable and fault-tolerant**

Scalability

- **Adding nodes adds capacity proportionally**
- **Increasing load results in a graceful decline in performance**
  - Not failure of the system



Fault Tolerance

- **Node failure is inevitable**
- **What happens?**
  - System continues to function
  - Master re-assigns tasks to a different node
  - Data replication = no loss of data
  - Nodes which recover rejoin the cluster automatically

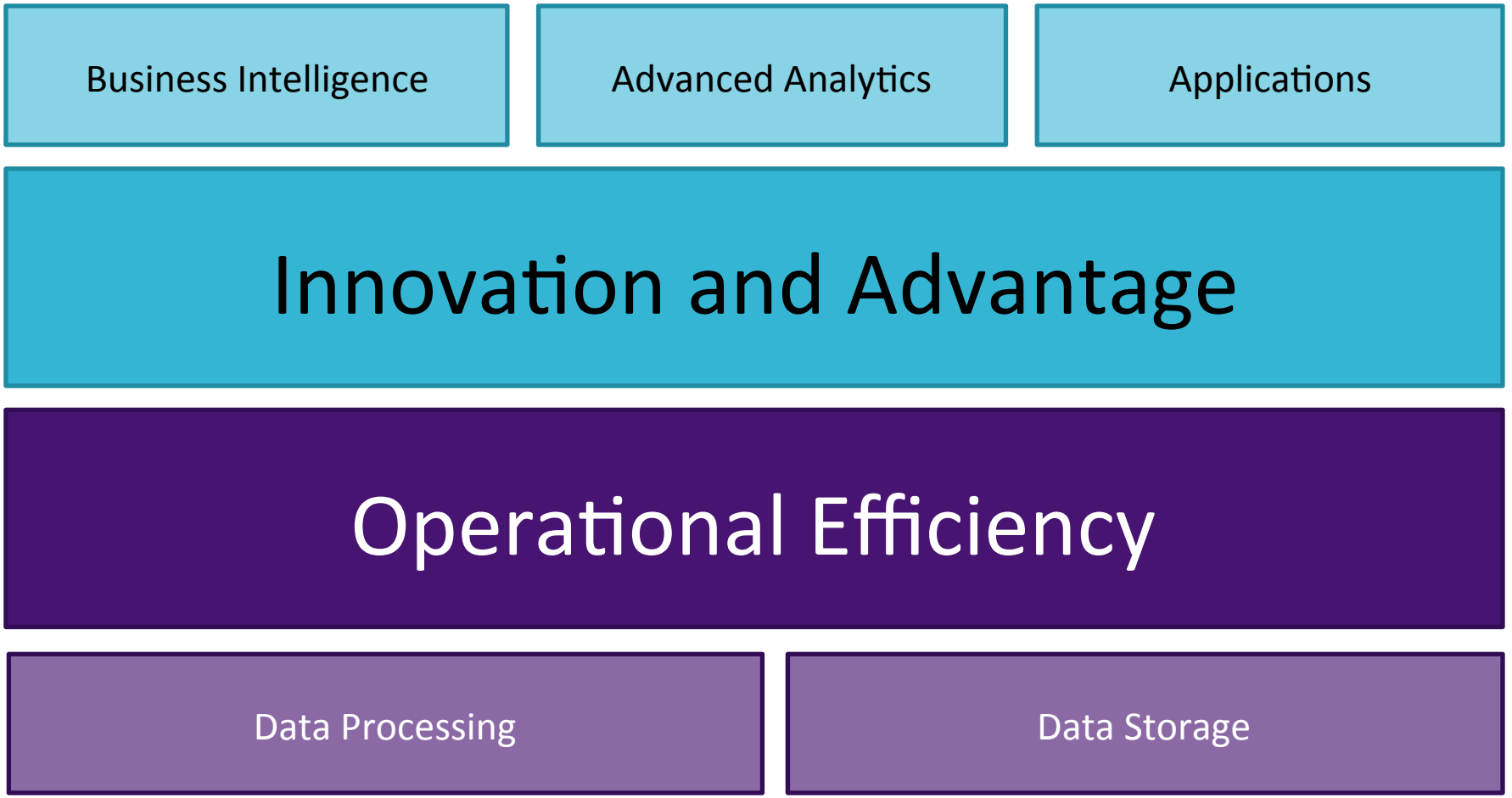
> "Failure is the defining difference between distributed and local programming, so you have to design distributed systems with the expectation of failure."
>
> – Ken Arnold
> (CORBA designer)

Chapter Topics

The Motivation for Hadoop

- Problems with Traditional Large-Scale Systems
- Requirements for a New Approach
- Hadoop!
- **Hadoop-able Problems**

What Can You Do With Hadoop?



Where Does Data Come From?

- **Science**
  - Medical imaging, sensor data, genome sequencing, weather data, satellite feeds, etc.
- **Industry**
  - Financial, pharmaceutical, manufacturing, insurance, online, energy, retail data
- **Legacy**
  - Sales data, customer behavior, product databases, accounting data, etc.
- **System Data**
  - Log files, health and status feeds, activity streams, network messages, web analytics, intrusion detection, spam filters

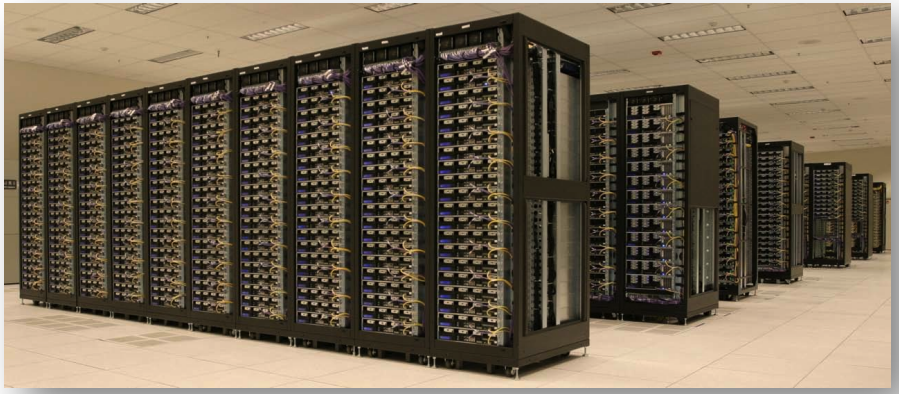
Common Types of Analysis with Hadoop

- Text mining
- Index building
- Graph creation and analysis
- Pattern recognition
- Collaborative filtering
- Prediction models
- Sentiment analysis
- Risk assessment

What is Common Across Hadoop-able Problems?

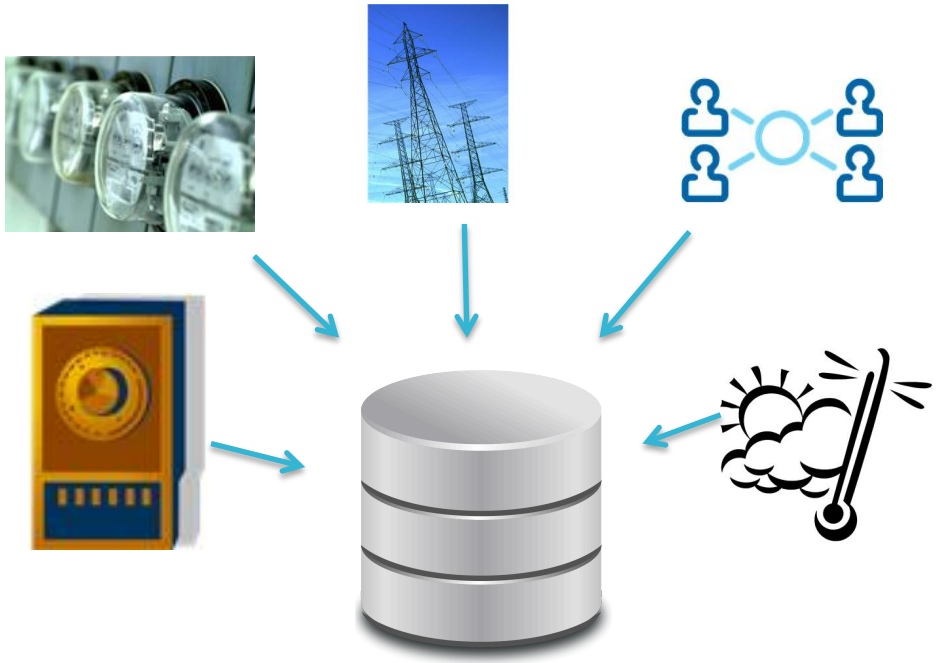
- **Nature of the data**
  - Volume
  - Velocity
  - Variety

- **Nature of the analysis**
  - Batch processing
  - Parallel execution
  - Distributed data



Use Case: Opower

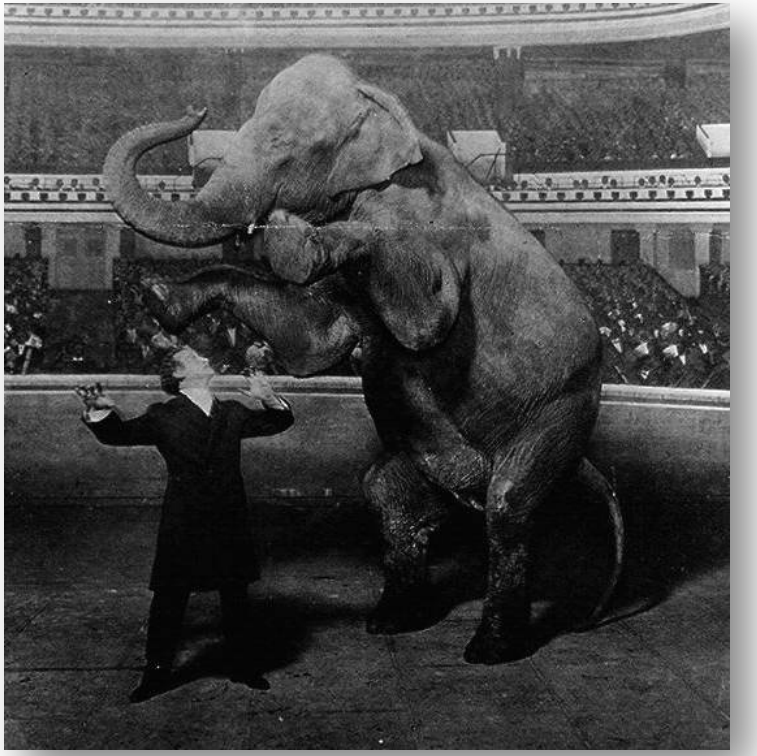
- **Opower**
  - SaaS for utility companies
  - Provides insights to customers about their energy usage
- **The data**
  - Huge amounts of data
  - Many different sources
- **The analysis, e.g.**
  - Similar homes comparison
  - Heating and cooling usage
  - Bill forecasting



SaaS: Software as a Service

Benefits of Analyzing with Hadoop

- **Previously impossible or impractical analysis**
- **Lower cost**
- **Less time**
- **Greater flexibility**
- **Near-linear scalability**
- **Ask Bigger Questions**



Bibliography

The following offer more information on topics discussed in this chapter

- **Google File System (2003)**
  - http://research.google.com/archive/gfs.html
- **MapReduce (2004)**
  - http://research.google.com/archive/mapreduce.html
  - http://www.cloudera.com/content/cloudera/en/resources/library/presentation/panel-it-trends-changing-the-way-organizations-do-business.html
  - http://opower.com/

Bibliography (cont'd)

The following offer more information on topics discussed in this chapter

- **"Data Science with Hadoop at Opower"**
  - http://strataconf.com/stratany2012/public/schedule/detail/25736
  - http://www.cloudera.com/content/cloudera/en/resources/library/hbasecon/hbasecon-2012--overcoming-data-deluge-with-hbase-to-help-save-the-environment.html

Hadoop Basic Concepts

Chapter 1.3

Hadoop: Basic Concepts

- **What is Hadoop?**
- **What features does the Hadoop Distributed File System (HDFS) provide?**
- **What are the concepts behind MapReduce?**
- **How does a Hadoop cluster operate?**

Chapter Topics

Hadoop Basic Concepts

- **What is Hadoop?**
- The Hadoop Distributed File System (HDFS)
- How MapReduce Works

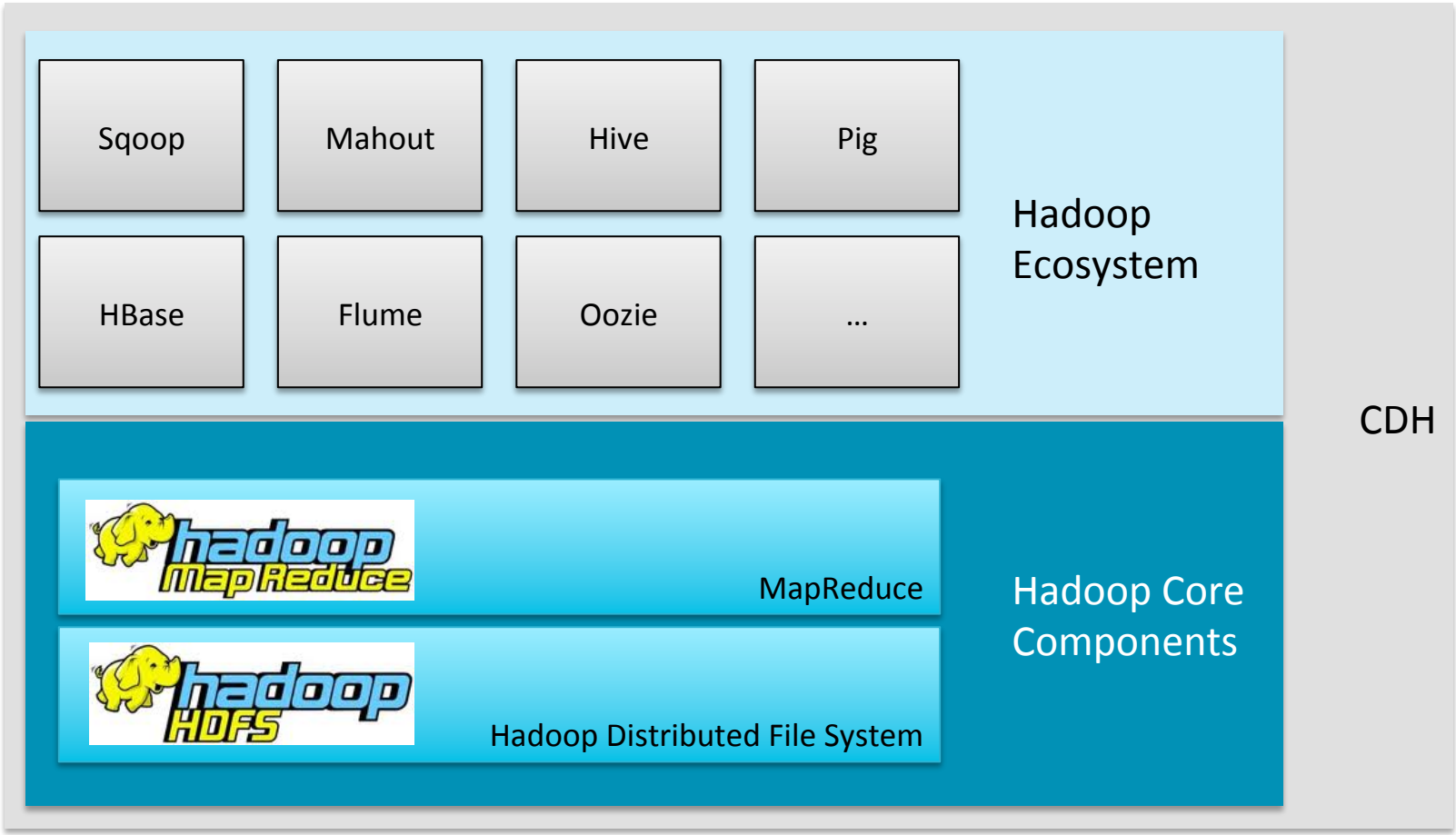
The Hadoop Project

- **Hadoop is an open-source project overseen by the Apache Software Foundation**
- **Originally based on papers published by Google in 2003 and 2004**
- **Hadoop committers work at several different organizations**
  - Including Cloudera, Yahoo!, Facebook, LinkedIn

Hadoop Components

- **Hadoop consists of two core components**
  - The Hadoop Distributed File System (HDFS)
  - MapReduce
- **There are many other projects based around core Hadoop**
  - Often referred to as the 'Hadoop Ecosystem'
  - Pig, Hive, HBase, Flume, Oozie, Sqoop, etc
    - Many are discussed later in the course
- **A set of machines running HDFS and MapReduce is known as a *Hadoop Cluster***
  - Individual machines are known as *nodes*
  - A cluster can have as few as one node, as many as several thousand
    - More nodes = better performance!

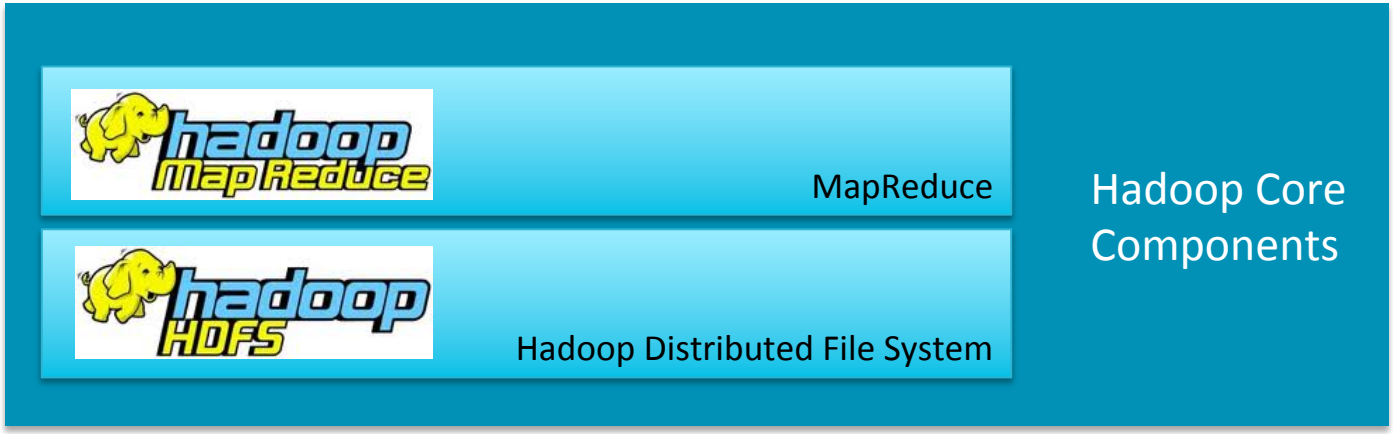
Hadoop Components (cont'd)



Note: CDH is Cloudera's open source Apache Hadoop distribution.

Core Components: HDFS and MapReduce

- **HDFS (Hadoop Distributed File System)**
  - Stores data on the cluster
- **MapReduce**
  - Processes data on the cluster



Hadoop Components: HDFS

- **HDFS, the Hadoop Distributed File System, is responsible for storing data on the cluster**
- **Data is split into blocks and distributed across multiple nodes in the cluster**
  - Each block is typically 64MB or 128MB in size
- **Each block is replicated multiple times**
  - Default is to replicate each block three times
  - Replicas are stored on different nodes
    - This ensures both reliability and availability

Hadoop Components: MapReduce

- **MapReduce is the system used to process data in the Hadoop cluster**
- **Consists of two phases: Map, and then Reduce**
  - Between the two is a stage known as the *shuffle and sort*
- **Each Map task operates on a discrete portion of the overall dataset**
  - Typically one HDFS block of data
- **After all Maps are complete, the MapReduce system distributes the intermediate data to nodes which perform the Reduce phase**
  - Much more on this later!

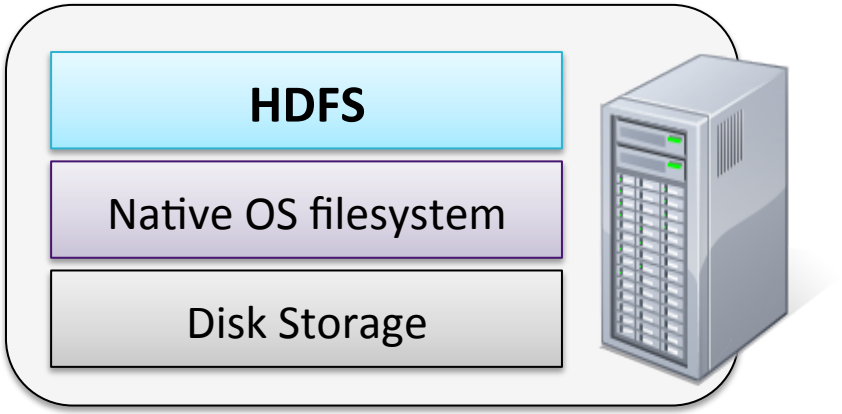
Chapter Topics

Hadoop Basic Concepts

- What is Hadoop?
- **The Hadoop Distributed File System (HDFS)**
- How MapReduce Works

HDFS Basic Concepts

- **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

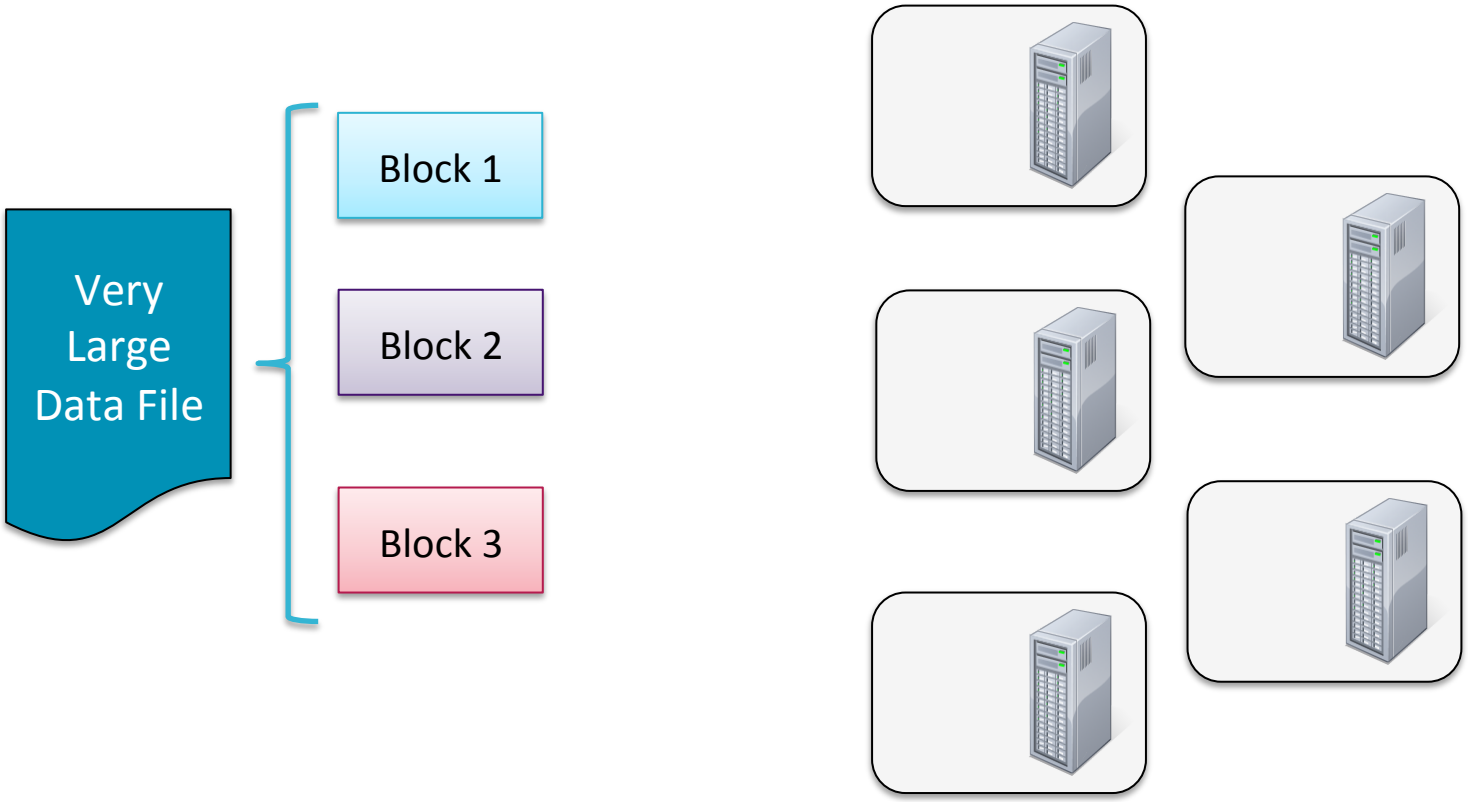


How Files Are Stored in HDFS

- **Files are split into blocks**
  - Each block is usually 64MB or 128MB
- **Data is distributed across many machines at load time**
  - Different blocks from the same file will be stored on different machines
  - This provides for efficient MapReduce processing (see later)
- **Blocks are replicated across multiple machines, known as *DataNodes***
  - Default replication is three-fold
    - Meaning that each block exists on three different machines
- **A master node called the *NameNode* keeps track of which blocks make up a file, and where those blocks are located**
  - Known as the *metadata*

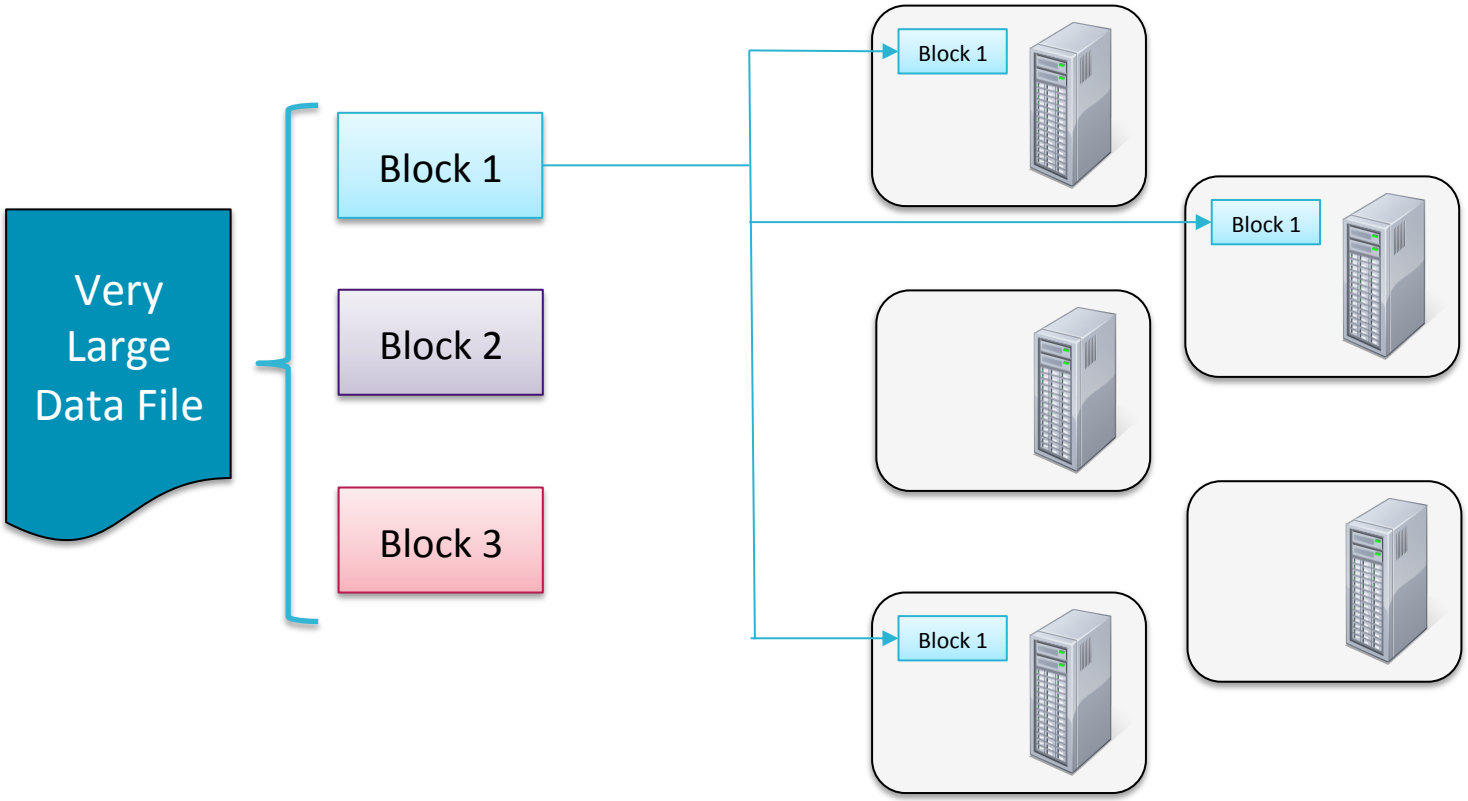
How Files are Stored (1)

- **Data files are split into blocks and distributed to data nodes**



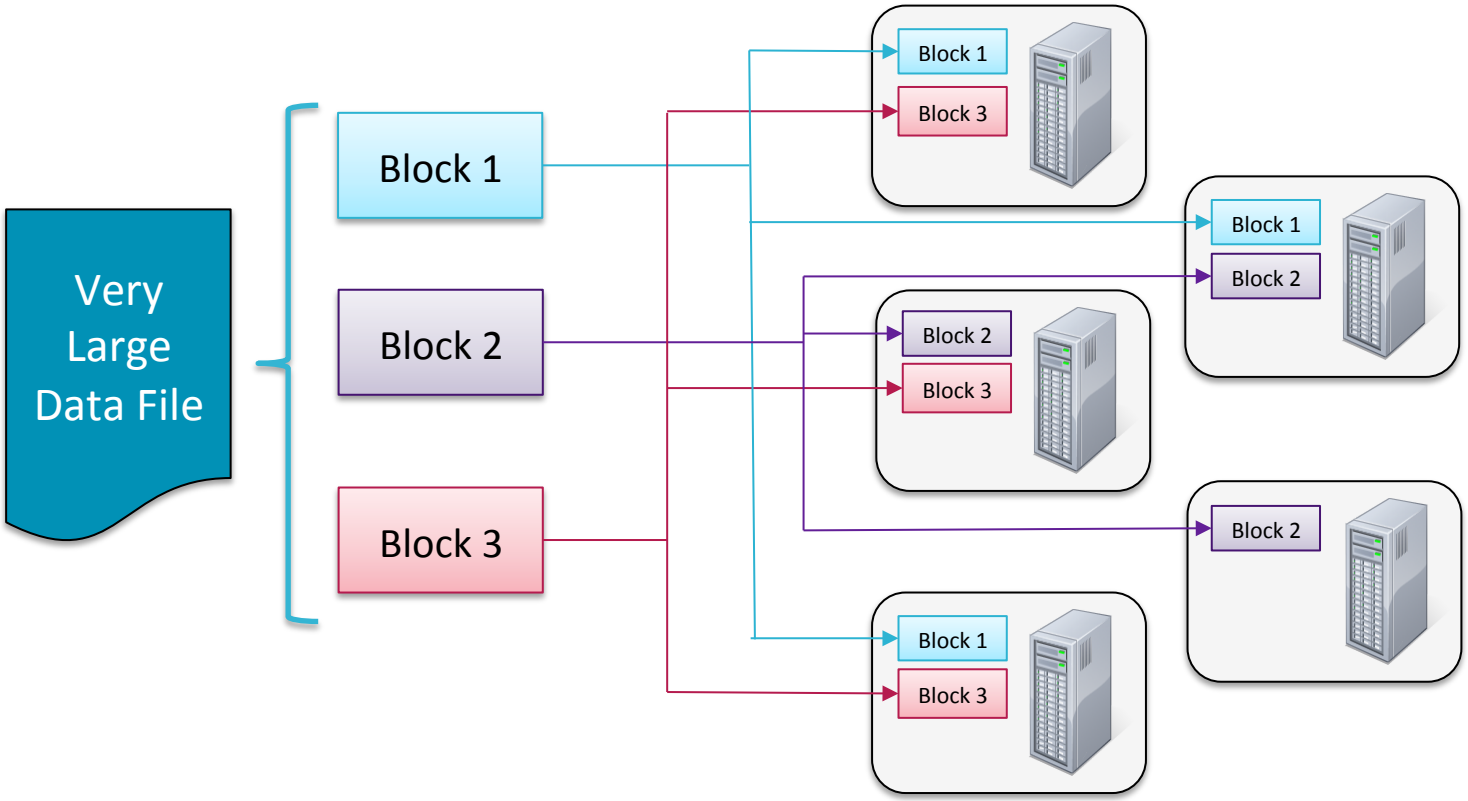
How Files are Stored (2)

- **Data files are split into blocks and distributed to data nodes**



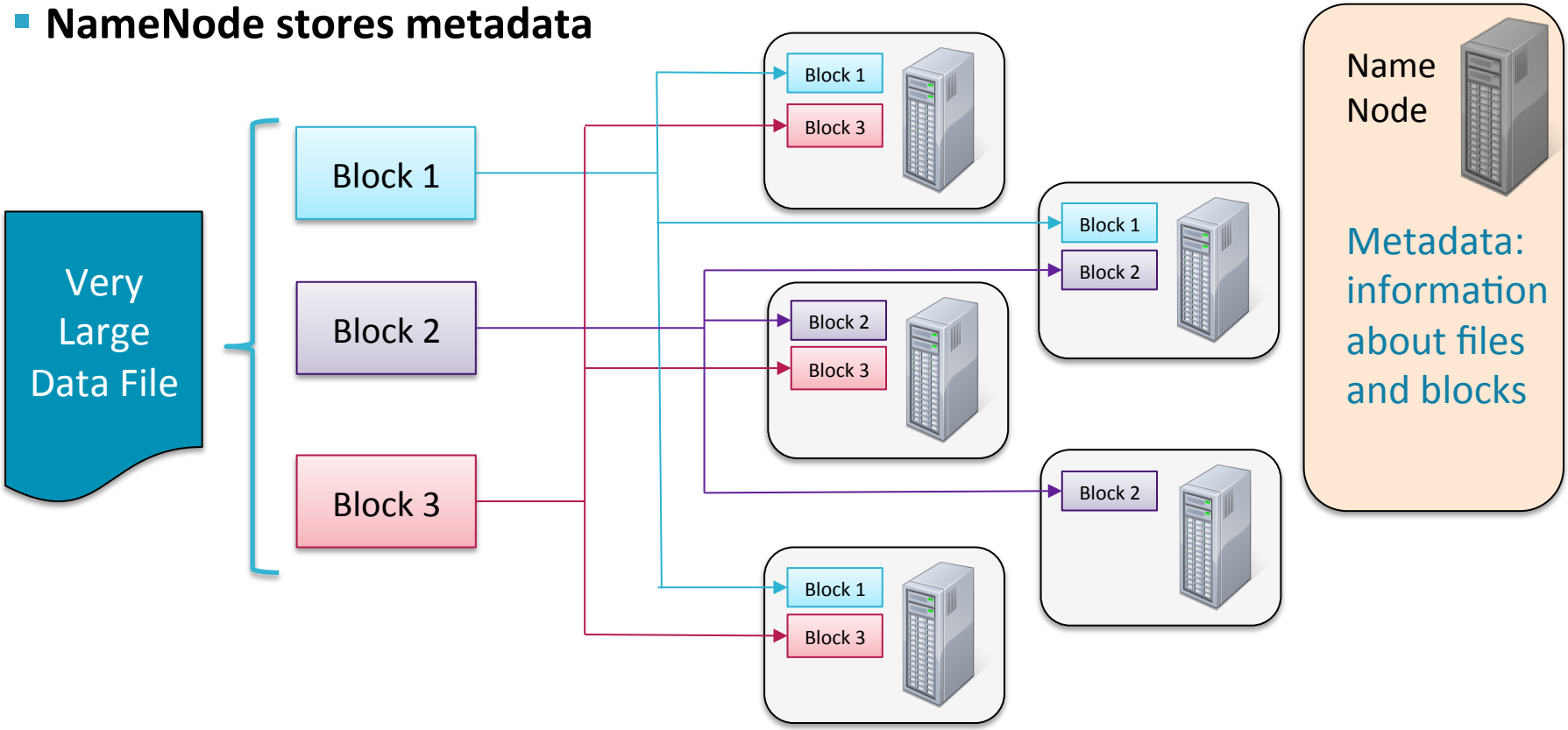
How Files are Stored (3)

- **Data files are split into blocks and distributed to data nodes**
- **Each block is replicated on multiple nodes (default 3x)**



How Files are Stored (4)

- **Data files are split into blocks and distributed to data nodes**
- **Each block is replicated on multiple nodes (default 3x)**
- **NameNode stores metadata**



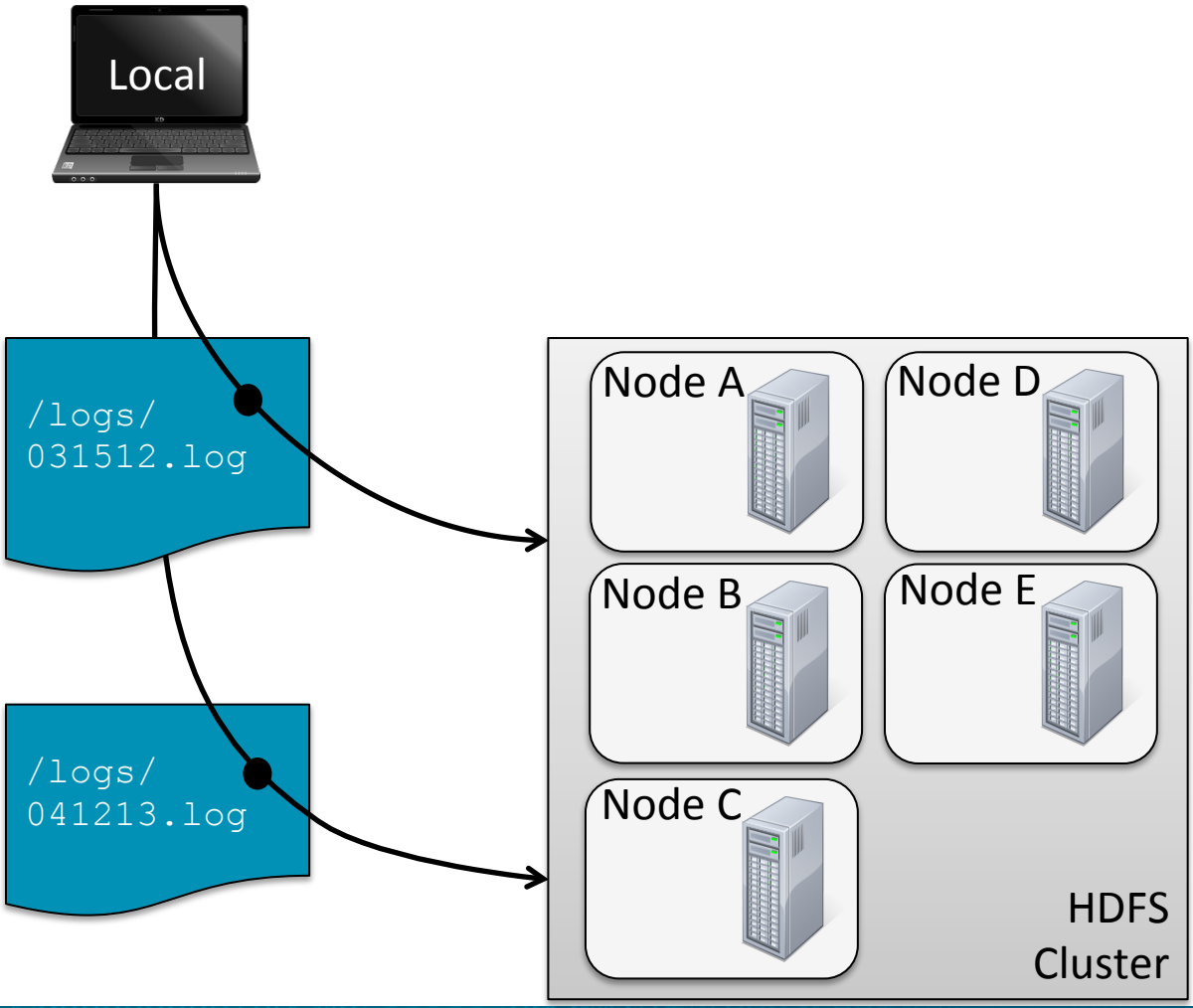
HDFS: Points To Note

- **Although files are split into 64MB or 128MB blocks, if a file is smaller than this the full 64MB/128MB will not be used**
- **Blocks are stored as standard files on the DataNodes, in a set of directories specified in Hadoop's configuration files**
- **Without the metadata on the NameNode, there is no way to access the files in the HDFS cluster**
- **When a client application wants to read a file:**
  - It communicates with the NameNode to determine which blocks make up the file, and which DataNodes those blocks reside on
  - It then communicates directly with the DataNodes to read the data
  - The NameNode will not be a bottleneck

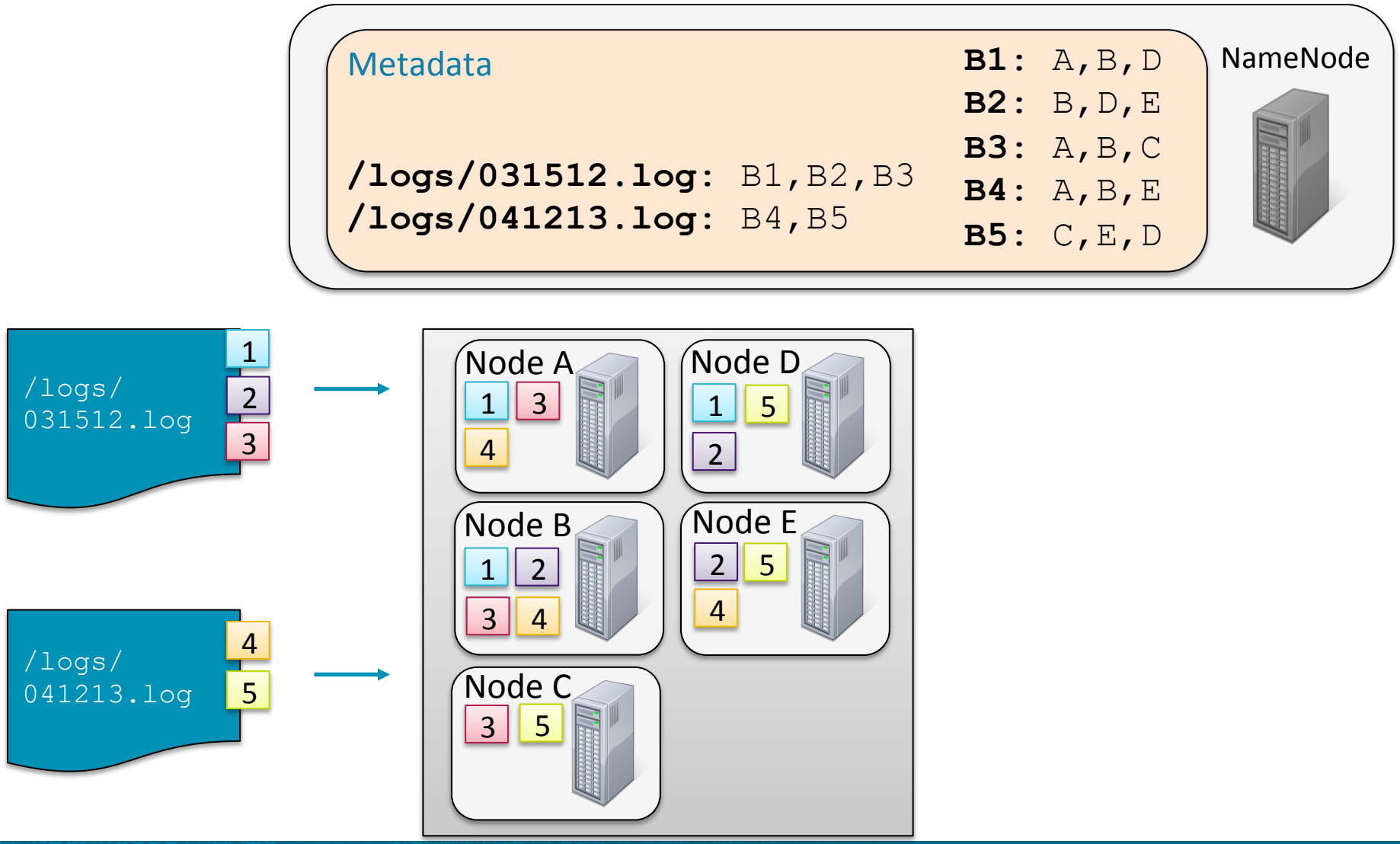
Options for Accessing HDFS

- **FsShell Command line: `hadoop fs`**
- **Java API**
- **Ecosystem Projects**
  - Flume
    - Collects data from network sources (e.g., system logs)
  - Sqoop
    - Transfers data between HDFS and RDBMS
  - Hue
    - Web-based interactive UI. Can browse, upload, download, and view files

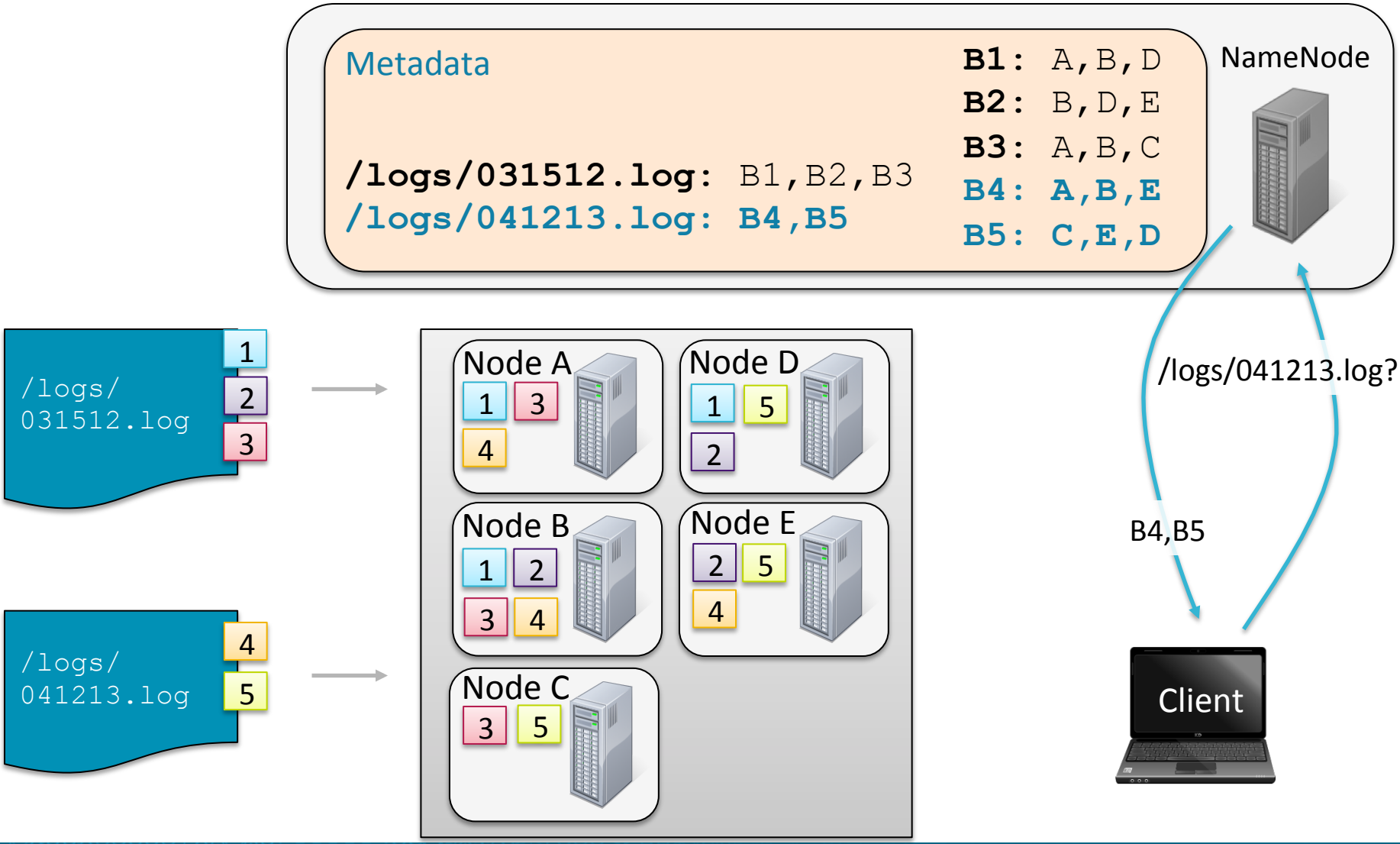
Example: Storing and Retrieving Files (1)



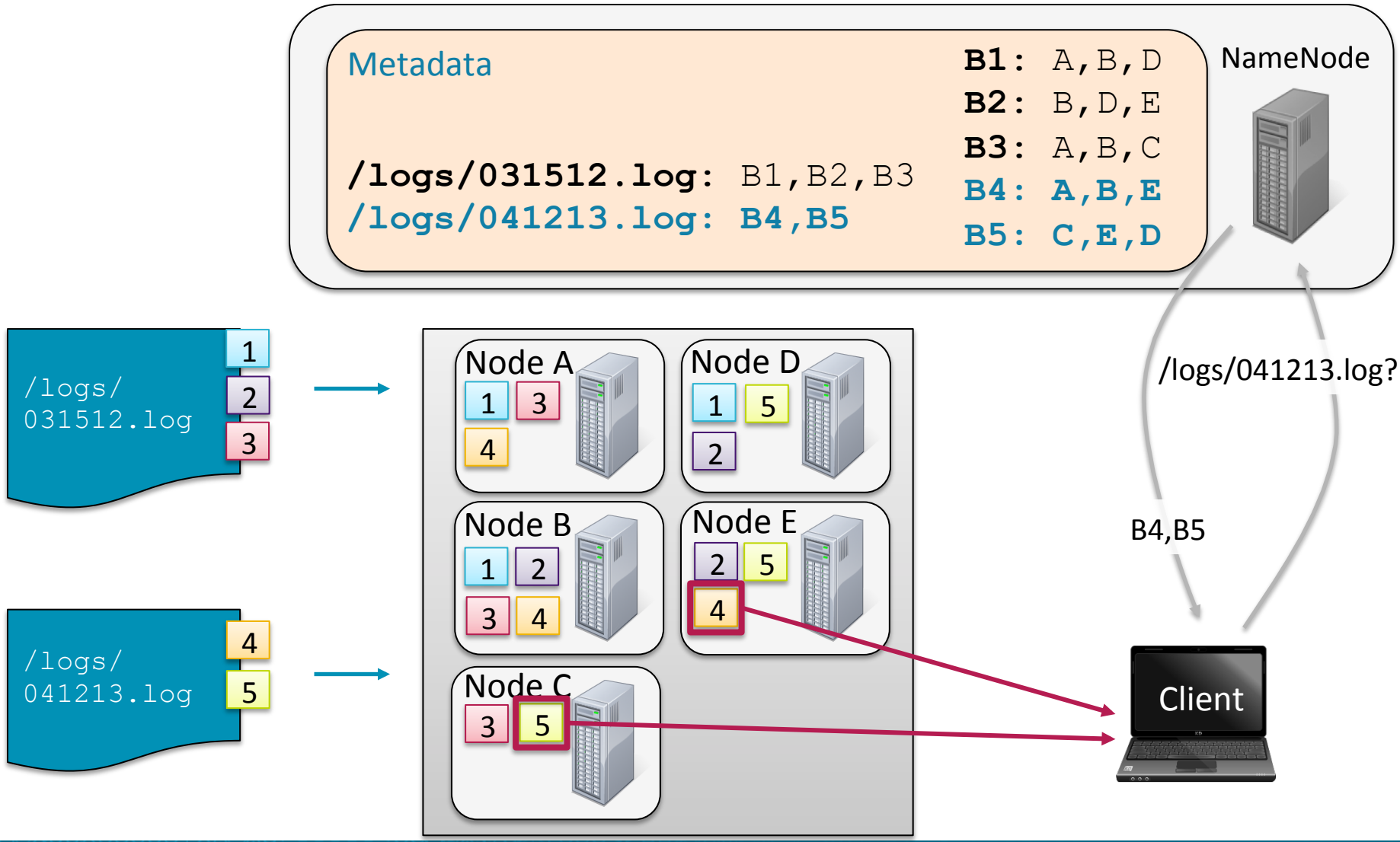
Example: Storing and Retrieving Files (2)



Example: Storing and Retrieving Files (3)



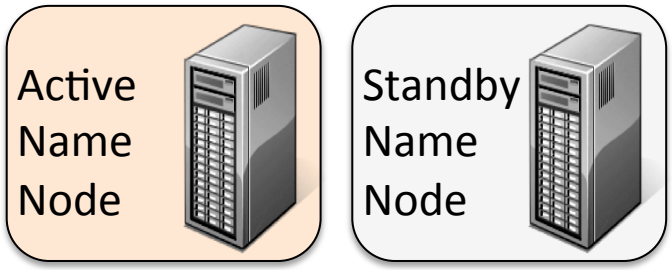
Example: Storing and Retrieving Files (4)



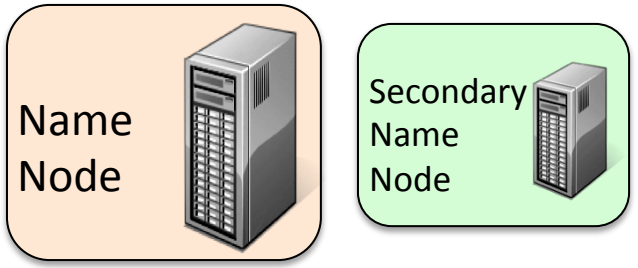
HDFS NameNode Availability

- **The NameNode daemon must be running at all times**
  - If the NameNode stops, the cluster becomes inaccessible

- **High Availability mode (in CDH4 and later)**
  - Two NameNodes: Active and Standby



- **Classic mode**
  - One NameNode
  - One "helper" node called SecondaryNameNode
    - Bookkeeping, not backup



hadoop fs Examples (1)

- **Copy file `foo.txt` from local disk to the user's directory in HDFS**

```
$ hadoop fs -put foo.txt foo.txt
```

  - This will copy the file to `/user/username/foo.txt`

- **Get a directory listing of the user's home directory in HDFS**

```
$ hadoop fs -ls
```

- **Get a directory listing of the HDFS root directory**

```
$ hadoop fs -ls /
```

hadoop fs Examples (2)

- **Display the contents of the HDFS file `/user/fred/bar.txt`**

```
$ hadoop fs -cat /user/fred/bar.txt
```

- **Copy that file to the local disk, named as `baz.txt`**

```
$ hadoop fs -get /user/fred/bar.txt baz.txt
```

hadoop fs Examples (3)

- **Create a directory called `input` under the user's home directory**

```
$ hadoop fs -mkdir input
```

- **Delete the directory `input_old` and all its contents**

```
$ hadoop fs -rm -r input_old
```

Important Notes About HDFS

- **HDFS performs best with a modest number of large files**
  - Millions, rather than billions, of files
  - Each file typically 100MB or more
- **How HDFS works**
  - Files are divided into blocks, which are replicated across nodes
- **Command line access to HDFS**
  - FsShell: `hadoop fs`
  - Sub-commands: `-get`, `-put`, `-ls`, `-cat`, etc
- **HDFS is optimized for large, streaming reads of files**
  - Rather than random reads

Chapter Topics

Hadoop Basic Concepts

- What is Hadoop?
- The Hadoop Distributed File System (HDFS)
- **How MapReduce Works**

What Is MapReduce?

- **MapReduce is a method for distributing a task across multiple nodes**
- **Each node processes data stored on that node**
  - Where possible
- **Consists of two phases:**
  - Map
  - Reduce

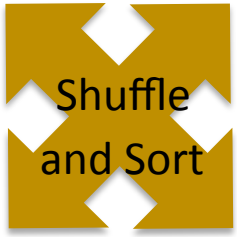
MapReduce: Terminology

- **A *job* is a 'full program'**
  - A complete execution of Mappers and Reducers over a dataset
  - In MapReduce 2, the term *application* is often used in place of 'job'
- **A *task* is the execution of a single Mapper or Reducer over a slice of data**
- **A *task attempt* is a particular instance of an attempt to execute a task**
  - There will be at least as many task attempts as there are tasks
  - If a task attempt fails, another will be started by the JobTracker
  - *Speculative execution* (see later) can also result in more task attempts than completed tasks

Hadoop Components: MapReduce

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

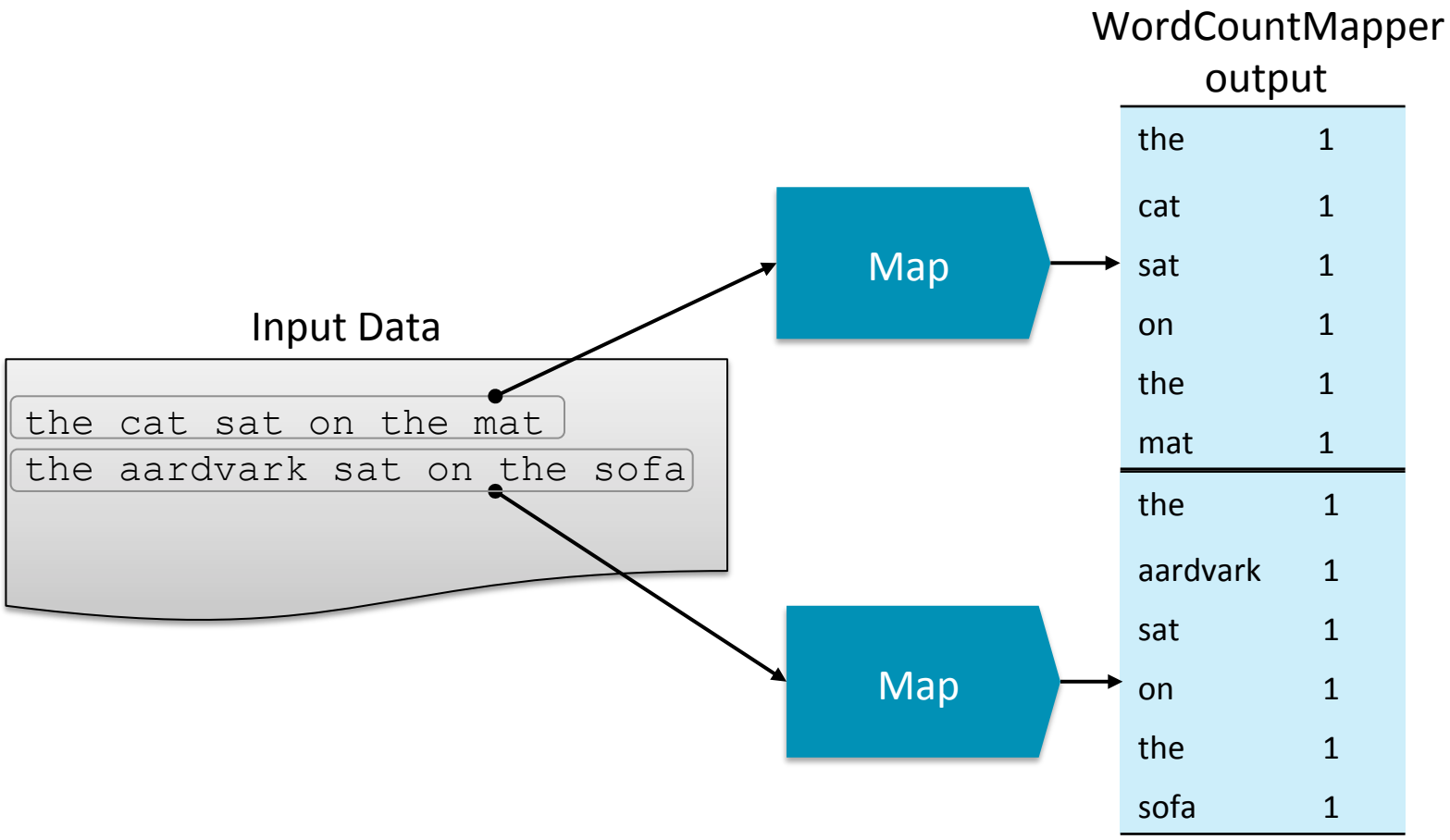
- **Shuffle and Sort**
  - Sorts and consolidates intermediate data from all mappers
  - Happens as Map tasks complete and before Reduce tasks start



- **The Reducer**
  - Operates on shuffled/sorted intermediate data (Map task output)
  - Produces final output

Example: Word Count

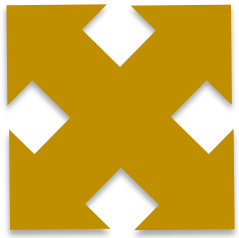
Example: The WordCount Mapper



Example: Shuffle & Sort

Mapper Output

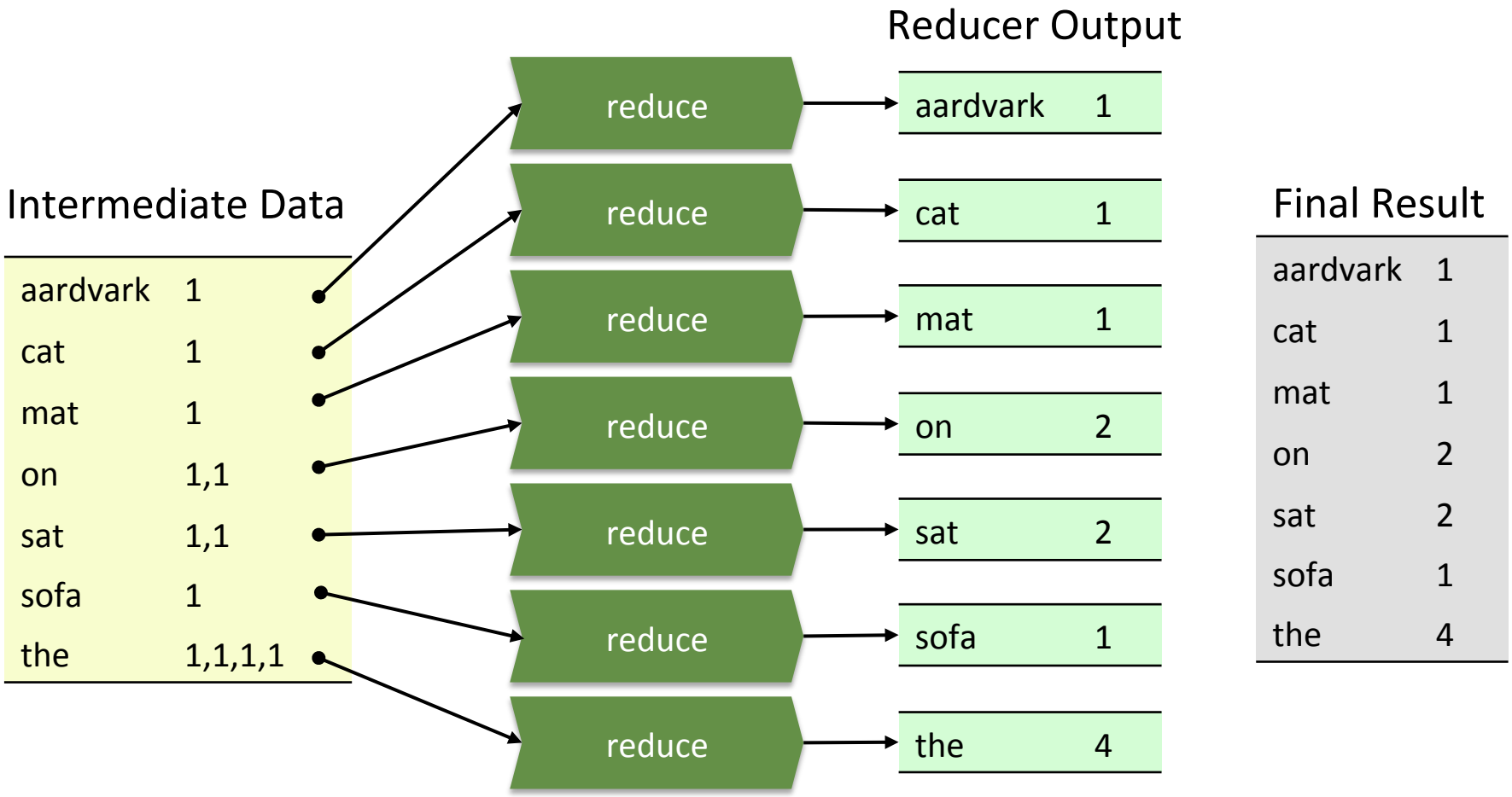
the	1
cat	1
sat	1
on	1
the	1
mat	1
the	1
aardvark	1
sat	1
on	1
the	1
sofa	1



Intermediate Data

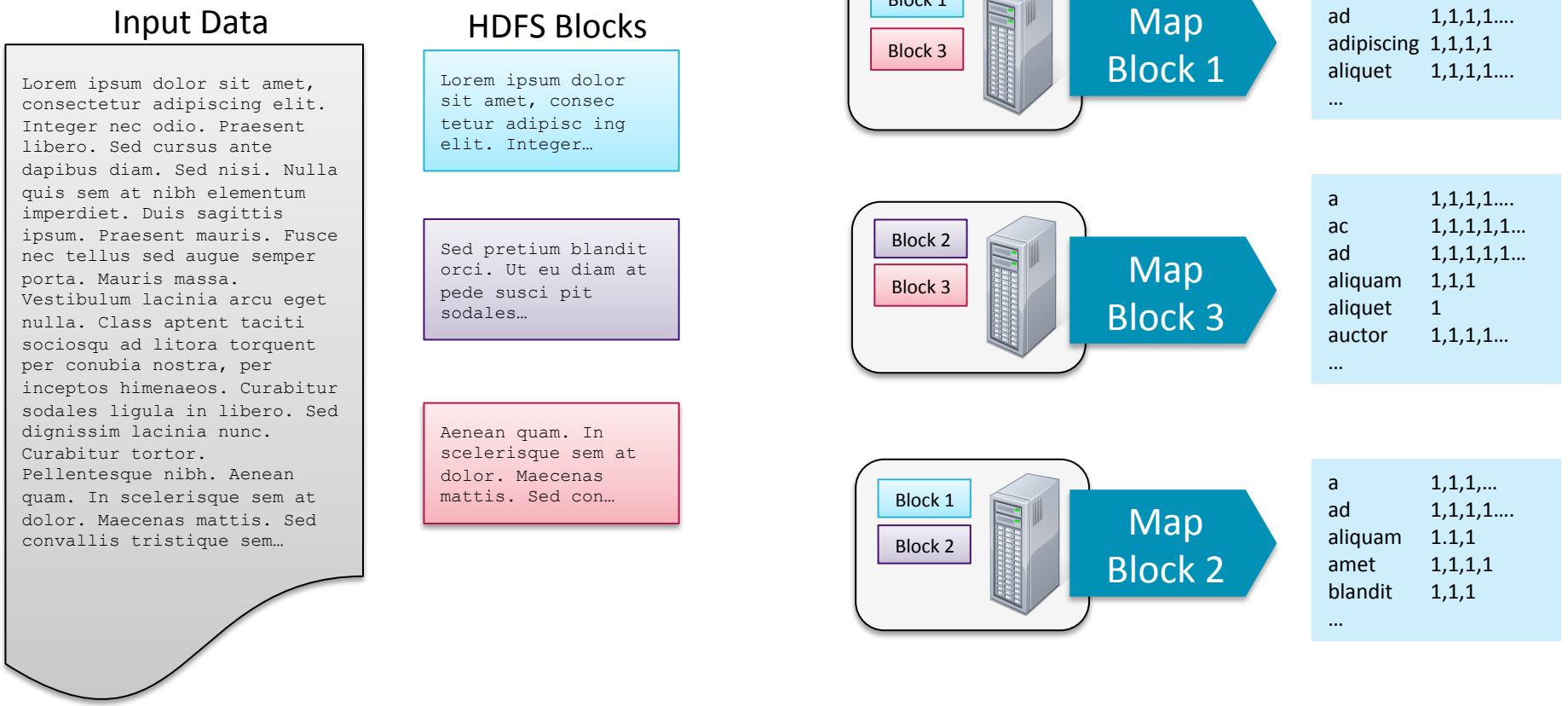
aardvark	1
cat	1
mat	1
on	1,1
sat	1,1
sofa	1
the	1,1,1,1

Example: SumReducer



Mappers Run in Parallel

- **Hadoop runs Map tasks on the slave node where the block is stored (when possible)**
  - Many Mappers can run in parallel
  - Minimizes network traffic



MapReduce: The Mapper

- **Hadoop attempts to ensure that Mappers run on nodes which hold their portion of the data locally, to avoid network traffic**
  - Multiple Mappers run in parallel, each processing a portion of the input data
- **The Mapper reads data in the form of key/value pairs**
  - 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
    - Typically the key is considered irrelevant
- **If the Mapper writes anything out, the output must be in the form of key/value pairs**

MapReduce: The Reducer

- **After the Map phase is over, all intermediate values for a given intermediate key are combined together into a list**
- **This list is given to a Reducer**
  - There may be a single Reducer, or multiple Reducers
  - All values associated with a particular intermediate key are guaranteed to go to the same Reducer
  - The intermediate keys, and their value lists, are passed to the Reducer in sorted key order
  - This step is known as the 'shuffle and sort'
- **The Reducer outputs zero or more final key/value pairs**
  - These are written to HDFS
  - In practice, the Reducer usually emits a single key/value pair for each input key

Why Do We Care About Counting Words?

- **Word count is challenging over massive amounts of data**
  - Using a single compute node would be too time-consuming
  - Using distributed nodes requires moving data
  - Number of unique words can easily exceed available memory
    - Would need to store to disk
- **Statistics are simple aggregate functions**
  - Distributive in nature
  - e.g., max, min, sum, count
- **MapReduce breaks complex tasks down into smaller elements which can be executed in parallel**
- **Many common tasks are very similar to word count**
  - e.g., log file analysis

Another Example: Analyzing Log Data

Input Data

```
...
2013-03-15 12:39 - 74.125.226.230 /common/logo.gif 1231ms - 2326
2013-03-15 12:39 - 157.166.255.18 /catalog/cat1.html 891ms - 1211
2013-03-15 12:40 - 65.50.196.141 /common/logo.gif 1992ms - 1198
2013-03-15 12:41 - 64.69.4.150 /common/promoex.jpg 3992ms - 2326
...
```

Map

FileTypeMapper output

...	...
gif	1231
html	891
gif	1992
jpg	3992
html	788
gif	3997
...	...

Intermediate Data after Shuffle and Sort

html	891,788,344,2990...
gif	1231,1992,3997,872...
jpg	3992,7881,2999...
png	919,890,3441,444...
txt	344,325,444,421...

Reduce

AverageReducer output

html	888.6
gif	1886.4
jpg	888.6
png	1201.0
txt	399.1

Features of MapReduce

- **Automatic parallelization and distribution**
- **Built-in fault-tolerance**
- **A clean abstraction for programmers**
  - MapReduce hides all of the "housekeeping" away from the developer
  - Developers concentrate simply on writing the Map and Reduce functions
- **Status and monitoring tools**

Hadoop Environments

- **Where to develop Hadoop solutions?**
  - Cloudera's Quickstart VM
    - Hadoop and Hadoop ecosystem tools are already installed and configured - works right out of the box, free of charge
    - Very useful for testing code before it is deployed to the real cluster
  - Alternately, configure a machine to run in Hadoop *pseudo-distributed mode*
    - Must install Hadoop ecosystem tools one by one
- **Where to run tested Hadoop solutions?**
  - Once testing is completed on a small data sample, the Hadoop solution can be run on a Hadoop cluster over all data
    - A Hadoop cluster is usually managed by the system administrator
    - It is useful to understand the components of a cluster, this will be covered in a future lecture

Bibliography

The following offer more information on topics discussed in this chapter

- **NameNode block size**
  - http://www.mail-archive.com/hdfs-user@hadoop.apache.org/msg00815.html
- **Basic information about the NameNode can be found in Hadoop: The Definitive Guide, third edition, by Tom White (TDG 3e).**
- **"File System Shell Guide"**
  - http://archive.cloudera.com/cdh/3/hadoop-0.20.2-cdh3u3/file_system_shell.html
- **The Hadoop "trash" directory for recently-deleted files**
  - http://archive.cloudera.com/cdh/3/hadoop-0.20.2-cdh3u3/hdfs_design.html#Space+Reclamation

Hadoop Solutions

Chapter 1.4

Hadoop Solutions

- **What types of problems are commonly addressed with Hadoop?**
- **What are some practical applications of Hadoop?**

Chapter Topics

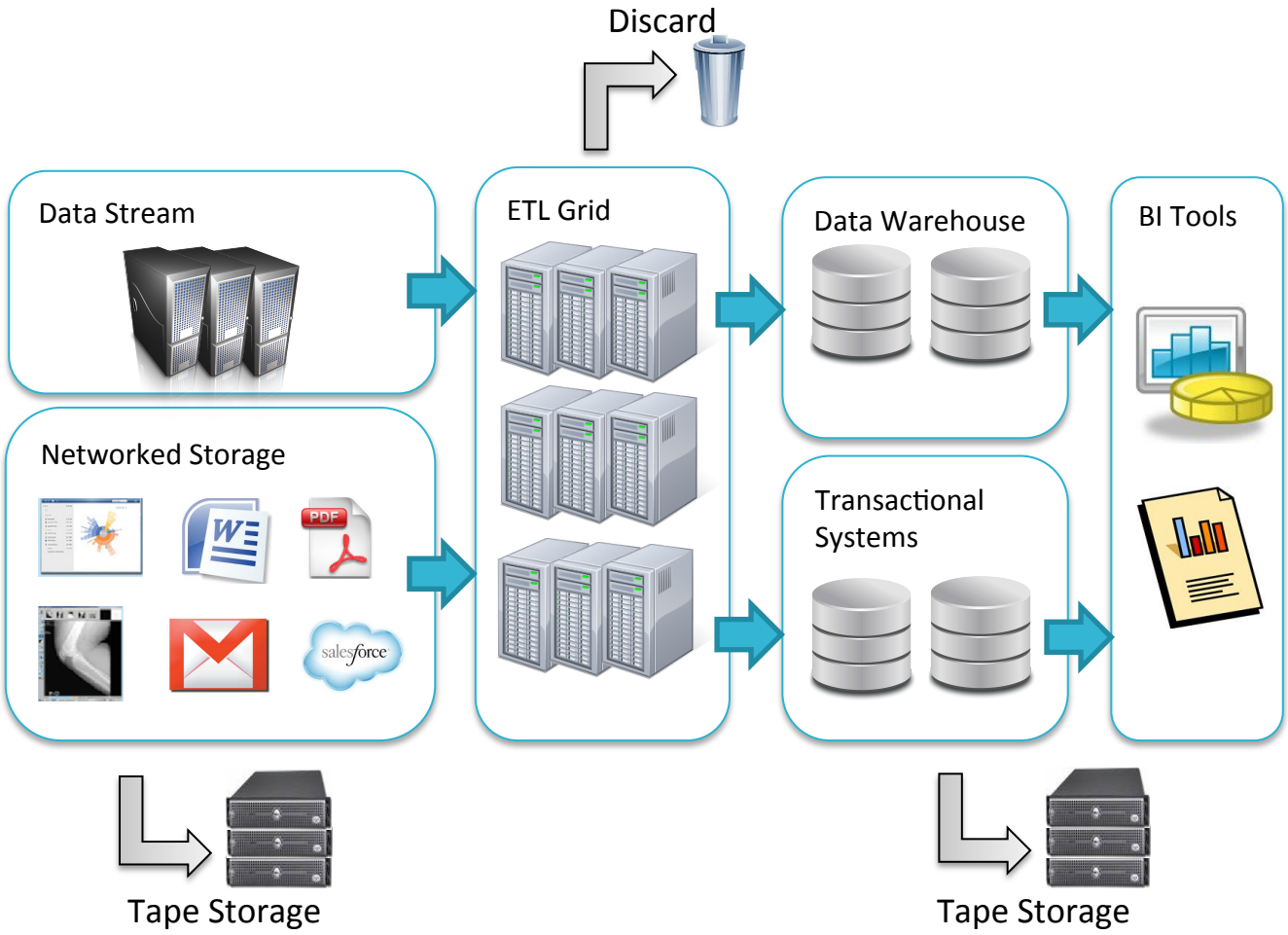
Hadoop Solutions

- **Some Common Hadoop Applications**
- Other Interesting Hadoop Use Cases

Some Common Hadoop Applications

- **Data Processing/ETL offload**
- **Data Warehouse offload**
- **Telemetry**
- **360-degree customer view**
- **Enterprise Data Hub**

Data Processing/ETL Offload: Traditional Approach



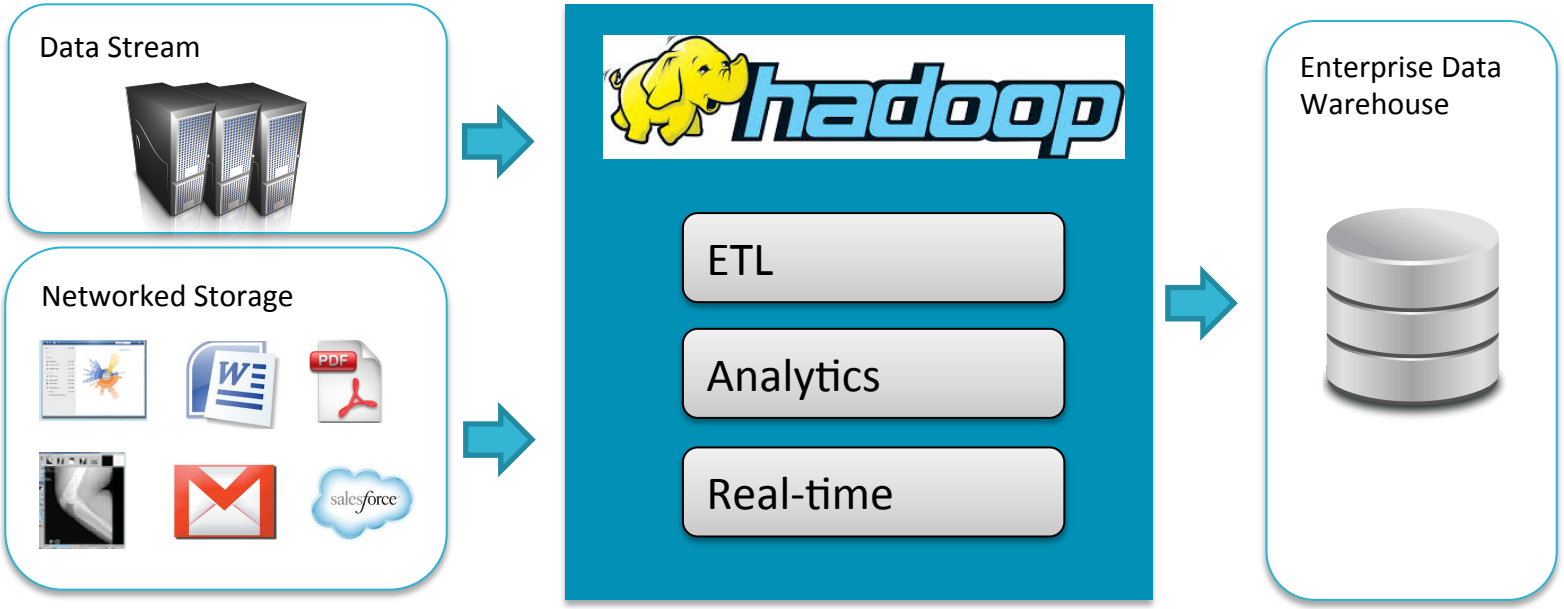
Challenges for existing data lifecycle:

- **Too much time**
- **Too much data**
- **Too costly**
- **Data quality**

ETL: Extract/Transform/Load

Data Processing/ETL Offload (cont'd)

With Hadoop, you can use *all* of your data

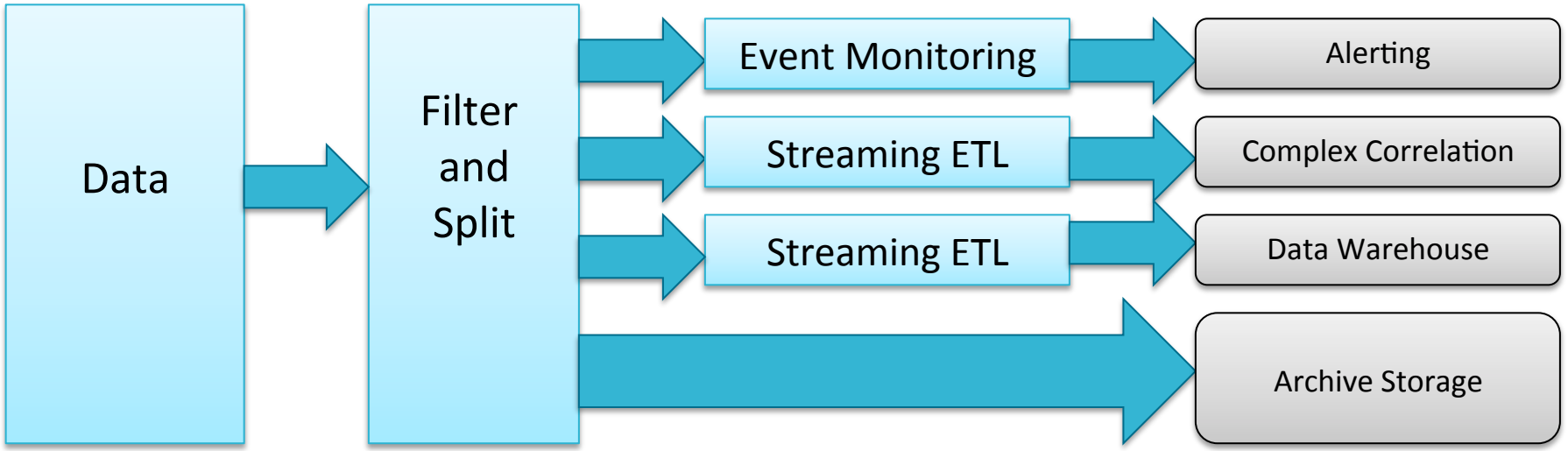


Data Processing/ETL Offload Use Case: BlackBerry (1)

- **Blackberry: Mobile devices**
- **The data**
  - Instrumentation data from devices
  - 650TB daily, 100PB total
  - Growing due to increased subscriber base, and increased usage
  - Mostly unstructured
- **The problems**
  - Infrastructure couldn't keep up with growth of data
  - Business analysts couldn't make use of data
    - Ad hoc queries took weeks to answer

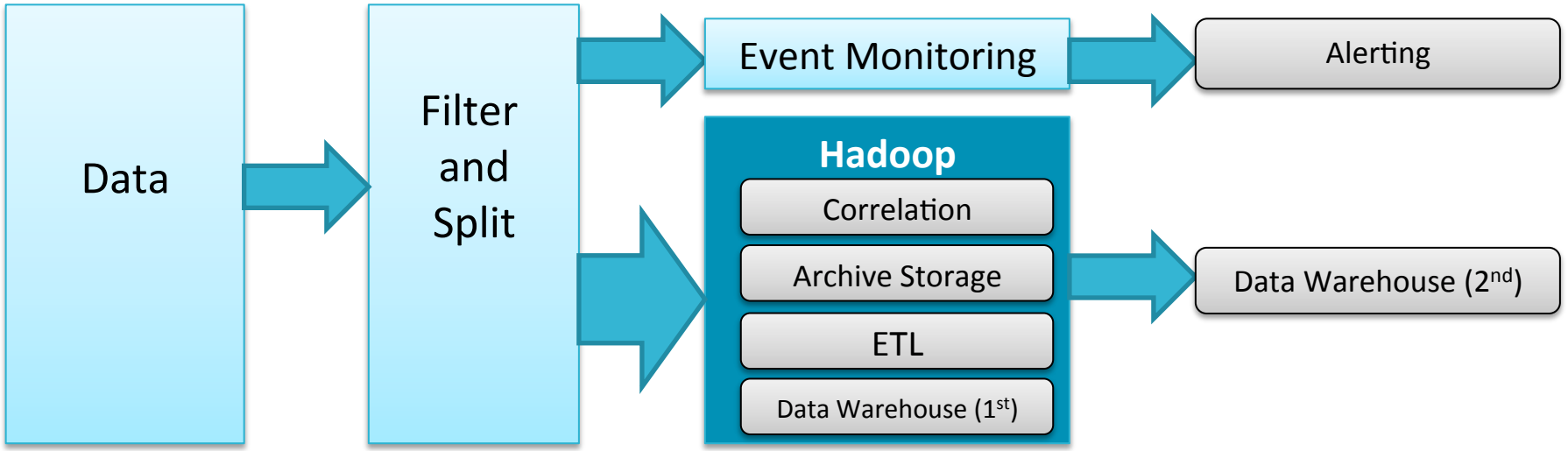
Data Processing/ETL Offload Use Case: BlackBerry (2)

- **The old way**
  - Focus on reducing data
  - Process data in the pipeline
  - Scalability issues at most stages



Data Processing/ETL Offload Use Case: BlackBerry (3)

- **The Hadoop way**
  - Archive storage in HDFS
  - ETL and correlations in Hadoop
  - Some data warehouse functions moved to Hadoop

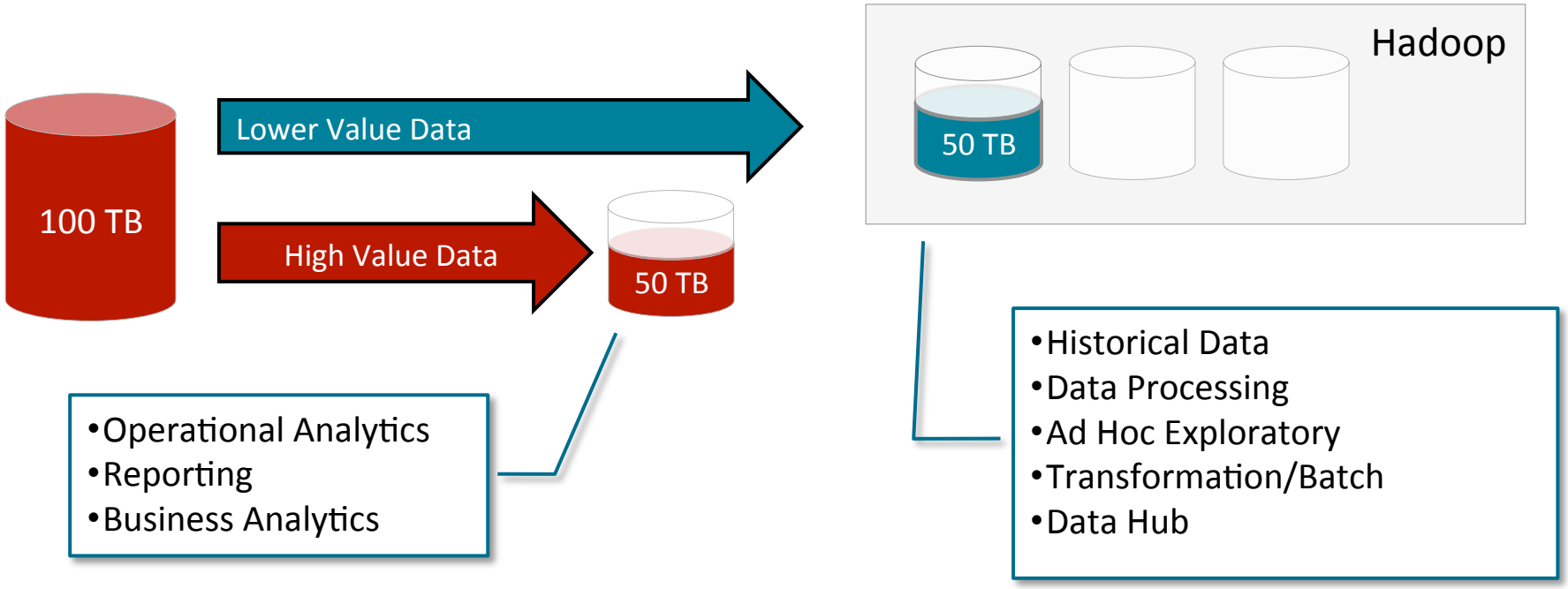


Data Processing/ETL Offload Use Case: BlackBerry (4)

- **The impact**
  - 90% code base reduction for ETL
    - Remaining code is business focused
  - Performance increases
    - e.g., previous ad hoc query that took four days now takes 53 minutes
  - Can perform queries and correlations that were previously impossible
  - Significant capital cost reduction

Data Warehouse Offload

- **Existing data warehouses often can't keep up with growing data**
  - Performance declines
  - Increasing capacity is very expensive
  - Less valuable data is archived – no longer available for analysis



Data Warehouse Offload Use Case: CBS Interactive

- **CBS Interactive: Online entertainment**
- **The data**
  - 1PB of content, clickstreams, weblogs, etc.
  - 1B events tracked daily
  - Web logs were overwhelming the proprietary system
  - Useful data was being discarded, historical data unusable
- **The analysis**
  - Cross-site and historical analysis, e.g., analysis across seasons
  - Identify user patterns, e.g., “high-value” users to target content
- **The impact**
  - Dramatic decrease in processing times, now meeting SLAs
  - Accommodates 50% data growth per year
  - Cost decrease for storage and processing

Telemetry

- **Telemetry**
  - Data collected at many points monitored in real time
  - Huge amounts of data – previously hard to store and analyze

- **Examples:**
  - Opower: SmartMeter readings
  - Chevron: Acoustic sampling from oil wells – 15TB per well
  - A major videogame company: player actions in multi-player online games

Telemetry Use Case: NetApp

- **NetApp: Network storage systems**
  - AutoSupport – real time device health monitoring
- **The data**
  - Device health data and logs
  - 240 billion records, >1PB and growing, 90% unstructured
- **The analysis, e.g.**
  - Compare current customer configuration to past configurations to proactively address problems
  - Automate RMAs (Return Merchandise Authorizations), use data to cross-sell and up-sell
- **The impact**
  - 64x improvement in query time on reports
  - Pattern matching to detect bugs was previously impossible, now takes 18 hours

360° Customer View

- **Customer information from many sources**
  - Customer Relationship Management systems (CRM)
  - Point-of-sale (POS)
  - Customer support
  - Website
  - etc.
- **The challenge: Bring existing data together into a 360° view**

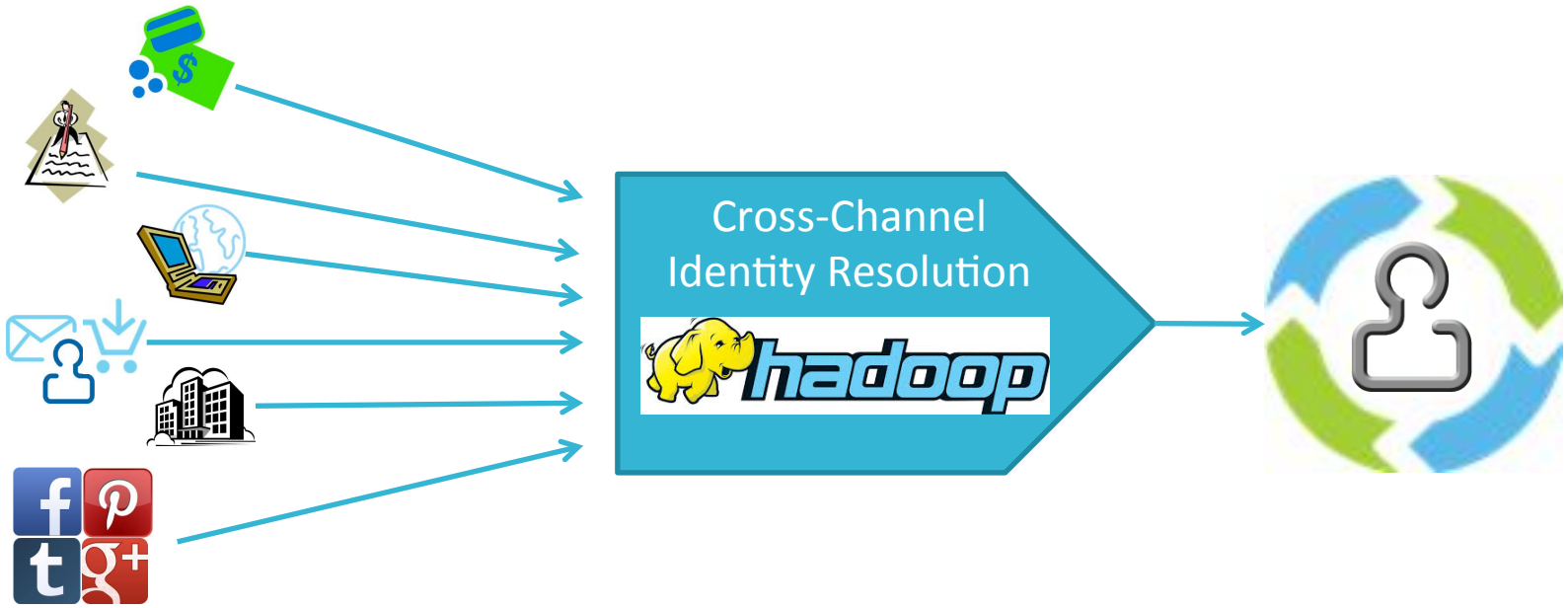
360° Customer View Use Case: Experian (1)

- **Experian: Digital Marketing**

- **The data:**
  - *Many* data channels (credit reporting, surveys, social media, website behavior, etc.)
  - 100M records per hour

360° Customer View Use Case: Experian (2)

- **The analysis**
  - Cross-Channel Identity Resolution (CCIR) engine built on Hadoop
  - Creates 360° view of customers
- **The impact**
  - 50x performance increase over prior data warehouse system

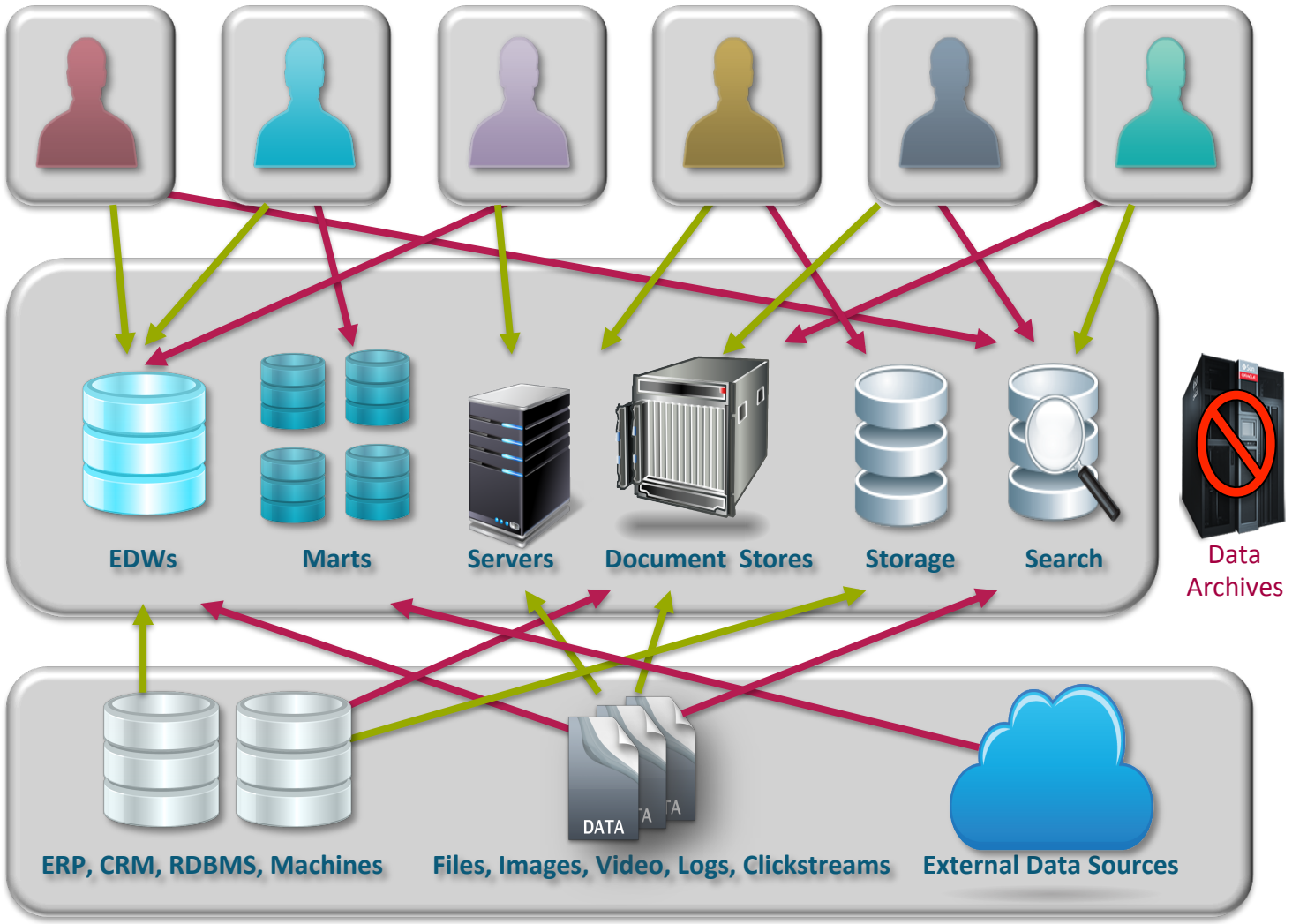


The Need for the Enterprise Data Hub

Thousands of Employees & Lots of Inaccessible Information

Heterogeneous Legacy IT Infrastructure

Silos of Multi-Structured Data Difficult to Integrate

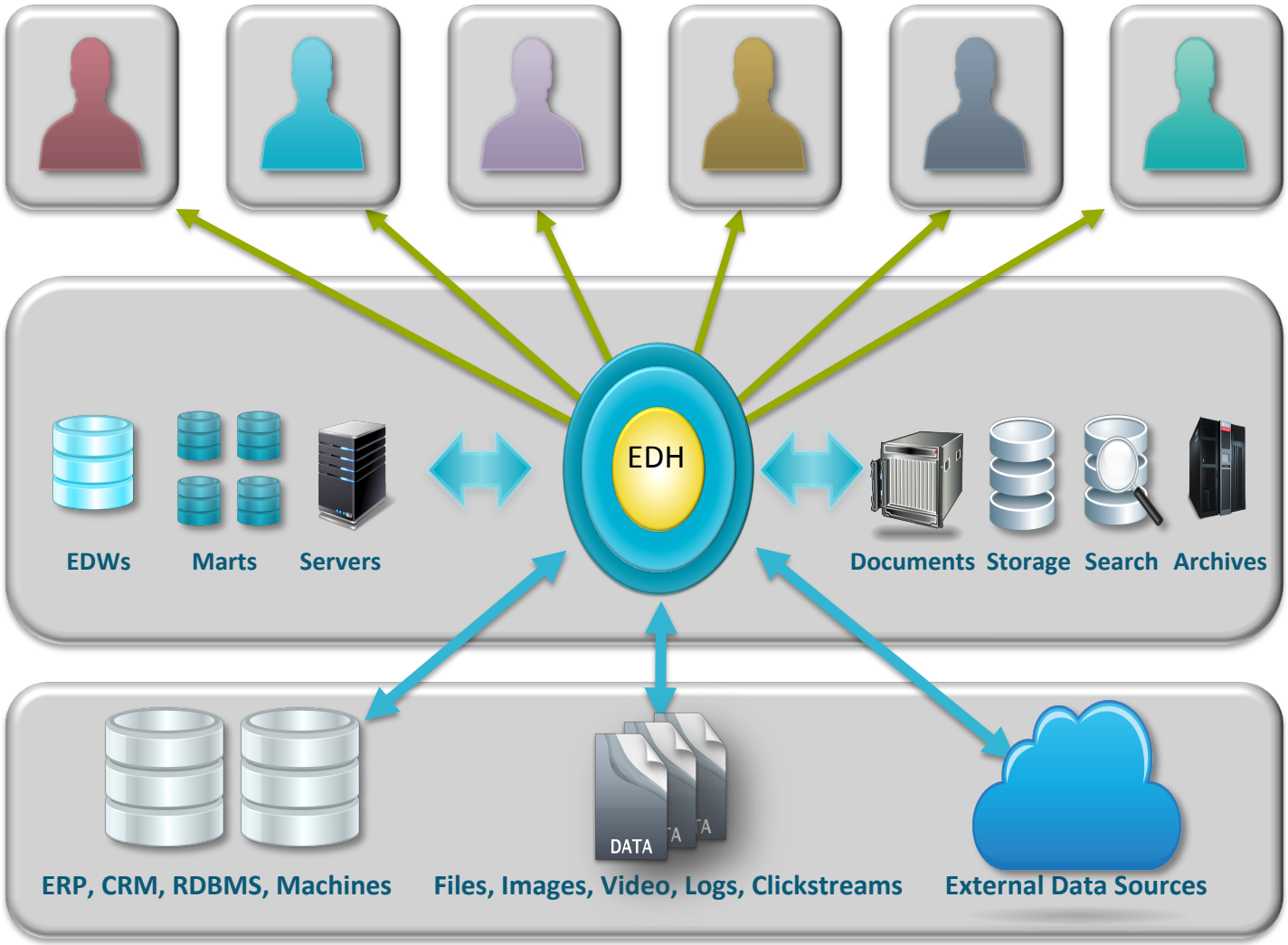


The Enterprise Data Hub: One Unified System

Information & data accessible by all for insight using leading tools and apps

Enterprise Data Hub
Unified Data Management Infrastructure

Ingest All Data
Any Type
Any Scale
From Any Source



Data Hub Use Case: Nokia

- **Nokia: Mobile phones and location services**
- **The data**
  - 100TB structured, 1+PB unstructured
  - Service usage, device streams, web logs, etc.
  - Before Hadoop: silos for different groups globally
- **The analysis**
  - 3D digital mapping
  - Usage patterns on mobile apps
- **The impact**
  - 10x reduction in storage cost
  - Use Hadoop to structure and export data

Other Common Applications

- **Predictive modeling**
- **Recommendations and ad targeting**
- **Time series analysis**
- **Point of Sale (PoS) analysis**

Chapter Topics

Hadoop Solutions

- Some Common Hadoop Applications
- **Other Interesting Hadoop Use Cases**

Some Other Interesting Use Case Examples

- **Orbitz**
- **A Financial Regulatory Body**
- **eBay**
- **A Leading North American Retailer**

Orbitz

- **Orbitz: Online travel booking**

- **The Data**
  - Millions of searches and bookings → hundreds of GB of data daily
  - Before Hadoop: only transactions stored (data warehouse)
  - After Hadoop: all data stored (HDFS + DW)
- **The analysis**
  - Hotel search rankings
  - Targeted recommendations
  - Web page performance tracking
  - Optimize search result cache performance
  - User segment analysis to drive personalization

Financial Regulatory Body

- **Stringent data reliability requirements**
  - Must store seven years of data
- **850TB of data collected from Wall Street trades each year**
  - Data volumes growing at 40% each year
- **Replacing EMC Greenplum + SAN with CDH**
  - Stores data for one year in DW, then years two to seven in Hadoop
  - 5PB of data in Hadoop by the end of 2013
- **Cost savings estimated to be tens of millions of dollars**
- **Performance gains of up to 20x**

eBay

- **eBay: Online marketplace**
  - Project Cassini: new search engine for ebay.com
- **The data**
  - 9PB total data
  - Per day: 250 million searches, 2 billion page views, 10 million new items for sale
- **The analysis**
  - Search results based on titles, descriptions, images, seller data, buyer data, behavioral data
  - Search 90 days of completed listings = 1B items

Leading North American Retailer

- **Storing 400TB of data in CDH cluster**
  - Capture and analysis of data on individual customers and SKUs across 4,000 locations
- **Using Hadoop for**
  - Loyalty program analytics and personal pricing
  - Fraud detection
  - Supply chain optimization
  - Marketing and promotions
  - Locating and pricing overstocked items for clearance

Key Points (1)

- **Hadoop has become a valuable business intelligence tool, and will become an increasingly important part of a BI infrastructure**
- **Hadoop won't replace the EDW**
  - But any organization with a large EDW should explore Hadoop as a complement to its BI infrastructure
- **Use Hadoop to offload the time- and resource-intensive processing of large data sets so you can free up your data warehouse to serve user needs**

Key Points (2)

- **Data is big and getting bigger**
- **Data is often unstructured or complex**
- **Hadoop is used to retrieve value out of data**
- **Examples can be found in every market sector**
- **Benefits of Hadoop**
  - Handles less structured/unstructured data
  - Significantly lower TCO
  - Can retain data indefinitely for much lower cost than traditional solutions

Bibliography

The following offer more information on topics discussed in this chapter

- **ETL Offload**
  - http://www.cloudera.com/content/cloudera/en/resources/library/recordedwebinar/the-business-advantage-of-hadoop-lessons-from-the-field-cloudera-summer-webinar-series-451-research-video-recording.html
- **CBS Data Warehouse Offload**
  - http://www.cloudera.com/content/dam/cloudera/Resources/PDF/2013-04-02_Ovum_whitepaper_Hadoop_Extending_Your_DW.pdf

Bibliography (cont'd)

The following offer more information on topics discussed in this chapter

- **CBS Data Warehouse Offload**
  - https://wiki.cloudera.com/display/~wmudge/CBS+Interactive
  - http://www.cloudera.com/content/cloudera/en/resources/library/recordedwebinar/how-cbs-interactive-uses-cloudera-manager-to-effectively-manage-their-hadoop-cluster-webinar.html
- **Nokia**
  - http://www.cloudera.com/content/dam/cloudera/documents/Cloudera-Nokia-case-study-final.pdf

Bibliography (cont'd)

The following offer more information on topics discussed in this chapter

- **360° Customer View - Experian**
  - http://www.cloudera.com/content/cloudera/en/solutions/applications/example-applications.html
  - http://www.cloudera.com/content/cloudera/en/resources/library/casestudy/Experian_Transforming_the_Marketing_Landscape_with_Cloudera_Case_Study.html
  - http://www.infoworld.com/d/big-data/experian-credits-big-data-real-time-marketing-209117
- **Orbitz**
  - http://blog.cloudera.com/blog/2010/08/improving-hotel-search-hadoop-orbitz-worldwide/

Bibliography (cont'd)

The following offer more information on topics discussed in this chapter

- **eBay**
  - http://www.cloudera.com/content/cloudera/en/resources/library/hadoopworld/hadoop-world-2011-keynote-presentation-video-ebay-keynote-hugh-e-williams-ebay.html
  - http://www.wired.com/business/2013/06/ebay-cassini-search-engine/

Bibliography (cont'd)

The following offer more information on topics discussed in this chapter

- **NetApp Use Case**
  - http://files.cloudera.com/pdf/whitepaper/Cloudera_Case_Study_NetApp.pdf
  - http://blogs.wsj.com/cio/2012/06/12/netapp-cio-uses-big-data-to-assess-product-performance/
  - http://www.cloudera.com/content/cloudera/en/resources/library/presentation/panel-it-trends-changing-the-way-organizations-do-business.html