Distributed Computing

Prof. Dr. Abeer Mahmoud

Professor of Computer Science -
Head of computer science Department-
Faculty of Computer and Information Sciences-
Ain Shams University
Abeer.mahmoud@cis.asu.edu.eg

Lecture 12:

Big data and distributed Systems

Introduction to Big Data analytics

Upon completing this module, you should be able to:

- ✓ Define Big Data and its characteristics.
- ✓ Identify the various sources of Big Data.
- ✓ Cite the business drivers for Big Data.
- ✓ Explain the evolving analytical architecture.
- ✓ Describe the role of data scientist.

Lesson: Big Data and its characteristics

In this lesson we discuss:

- The definition of Big Data
- Big Data characteristics and structure
- Sources of Big Data
- Understanding the business drivers for Big Data

What are your thoughts on Big Data?

Is there a threshold at which data becomes Big Data?

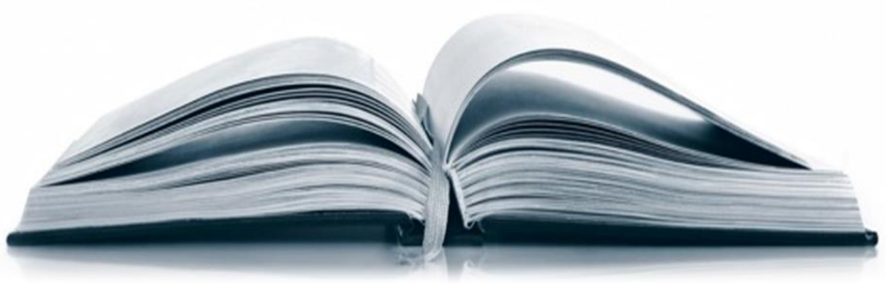
How much does the complexity of its structure influence the designation as Big Data?

Are you using any new or novel analytical techniques and tools to handle Big Data?

What is Big Data?

Big Data:

Datasets so large they break traditional IT infrastructures



- **Big Data** not only signifies a huge volume of data, but also presents complex data types and structure, with an increasing volume of unstructured data.
- Data gets generated and **changes rapidly**, and also comes from diverse sources.

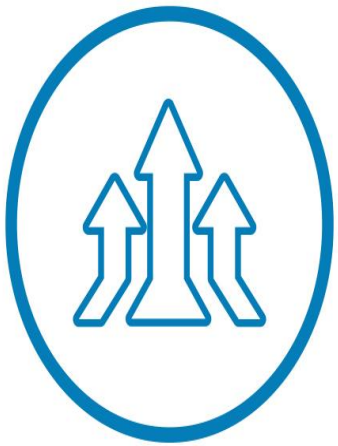
Big Data Defined

- ***"Big Data" is data whose scale, distribution, diversity, and/or timeliness require the use of new technical architectures and analytics to enable insights that unlock new sources of business value."***
  - Requires new data architectures, analytic sandboxes
  - New tools
  - New analytical methods
  - Integrating multiple skills into new role of data scientist

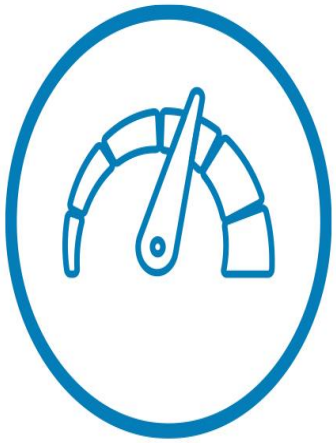
- Organizations are deriving business benefit from analyzing ever larger and more complex data sets that increasingly require real-time or near-real time capabilities

Source: McKinsey May 2011 article *Big Data: The next frontier for innovation, competition, and productivity*

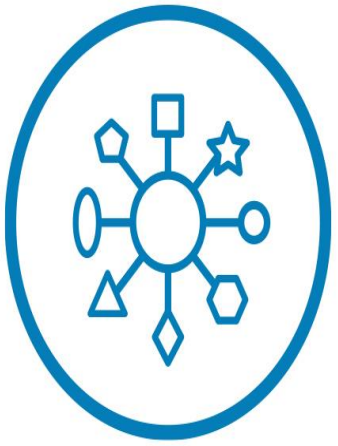
Characteristics of Big Data—(the 3 V's)



Volume



Velocity



Variety

Characteristics of Big Data—volume

- **2.5 quintillion bytes** of data are created daily: 44x increase from 2009–2020.

  This would fill 10 million blue ray discs, the size of which would measure 4 Eiffel towers, one on top of another.

- An estimated 40 Zettabytes (43 trillion Gigabytes) of data will be created by 2020, an increase of 300 times from 2005. That is, 5,247 GB of machine data for every person on the planet.

- The population of the world is 7 billion; 6 billion people have cell phones: a source of huge volumes of data.

Characteristics of Big Data—velocity

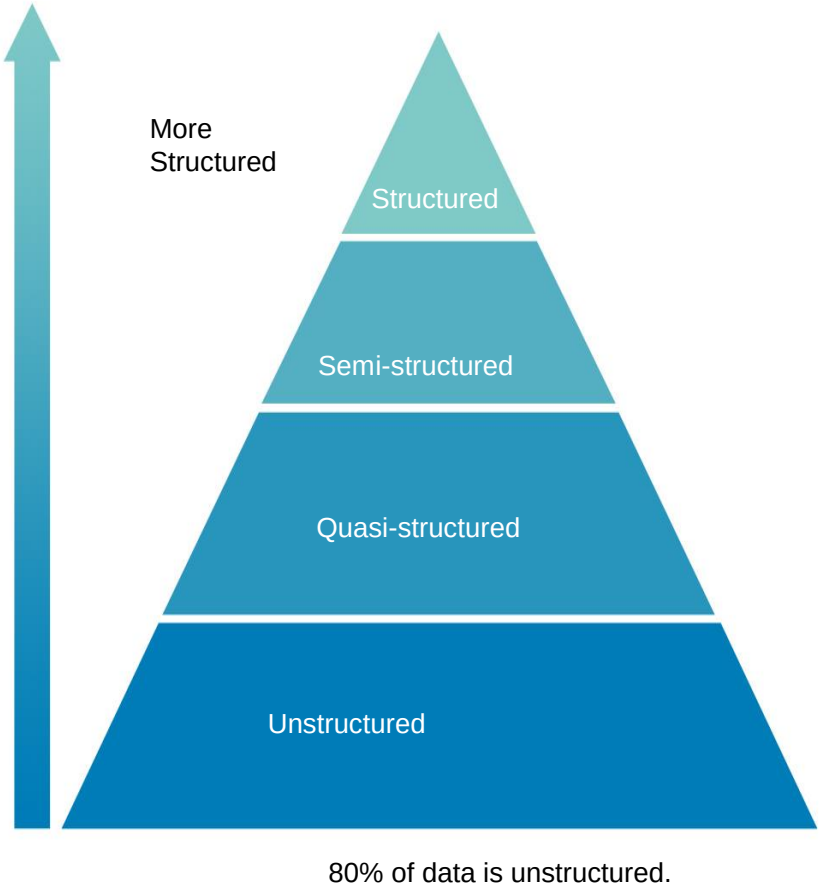
- Every 60 seconds, there are:
  - 98,000+ tweets.
  - 695,000 status updates on Facebook.
  - 698,445 Google searches.
- NYSE captures 1 TB of trade-related information during a trading session.
- The estimated rate of global Internet traffic by 2018 is 50,000 GB/sec.

Characteristics of Big Data—variety

- **Data comes from social media in the form of tweets, feeds, status updates, and videos, structured and unstructured**.
- Cisco estimates a total of 578 million wearables by 2019.

  As per estimate from VNI, wearables data traffic forecast for 2014 to 2019 will reach 292 EBs per year.
- Others **varieties** of data include data from:
  - **Sensors in cars.**
  - **The healthcare industry.**
  - **Smart homes.**
  - **Air travel.**

Big Data characteristics—data structures



80% of data is unstructured.

- **Structured:**
  - Data of a well-defined data type, format, or structure
  - *Examples:* Relational database tables and **CSV files**
- **Semi-structured:**
  - Textual data files with a discernable pattern, enabling parsing
  - *Example:* XML files
- **Quasi-structured:**
  - Textual data with **erratic** data formats: can be formatted with effort, tools, and time
  - *Example:* **Web clickstream data**
- **Unstructured:**
  - Data that has no inherent structure
  - *Examples:* **Text documents, images, and video**

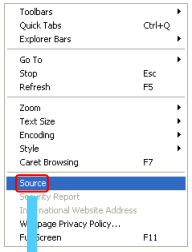
Four Main Types of Data Structures

Structured Data

SUMMER FOOD SERVICE PROGRAM 1] (Data as of August 01, 2011)				
Fiscal Year	Number of Sites	Peak (July) Participation	Meals Served	Total Federal Expenditures 2]
	----------Thousands----------		--Mil.--	---Million $---
1969	1.2	99	2.2	0.3
1970	1.9	227	8.2	1.8
1971	3.2	569	29.0	8.2
1972	6.5	1,080	73.5	21.9
1973	11.2	1,437	65.4	26.6
1974	10.6	1,403	63.6	33.6
1975	12.0	1,785	84.3	50.3
1976	16.0	2,453	104.8	73.4
TQ 3]	22.4	3,455	198.0	88.9
1977	23.7	2,791	170.4	114.4
1978	22.4	2,333	120.3	100.3
1979	23.0	2,126	121.8	108.6
1980	21.6	1,922	108.2	110.1

Semi-Structured Data

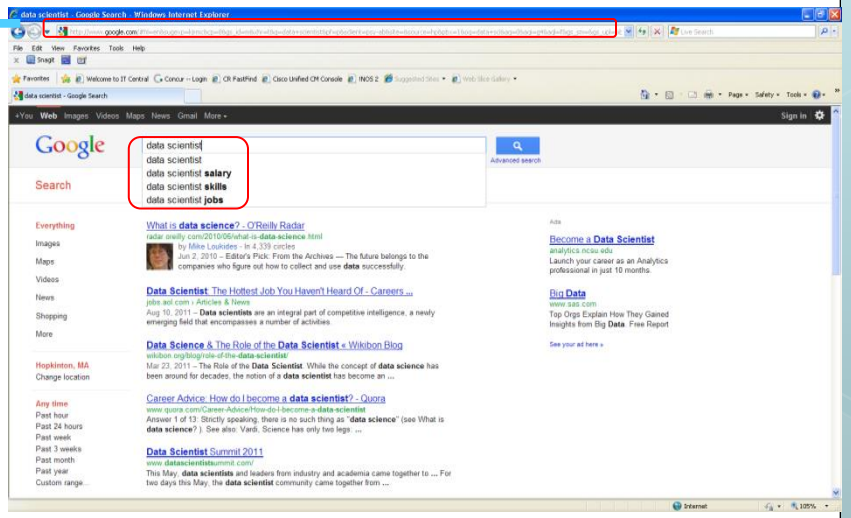
View → Source




```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-trans
<html xmlns="http://www.w3.org/1999/xhtml">

        <head>
                <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
                <META name="y_key" content="859b4020e1c9acec">
                        <link rel="canonical" href="http://www.emc.com/index.htm" />
                                <META NAME="verify-v1" CONTENT="yiZt9V0P4eV0jFdIPeVVIfRP32g4qtwFE0I2UvTMfSU
                <title>EMC - Data Recovery, Cloud Computing, and Storage Hardware</title>
                <META NAME="description" CONTENT="EMC is a leading provider of storage hardware solutions th
 data recovery and improve cloud computing." />
                        <META NAME="keywords" CONTENT="emc,network storage,data recovery,information manager
 software,nas storage,information protection,information management" />
                <!-- Start :stylehseet includes -->
<link rel="stylesheet" href="/_admin/css/styles.css" />
<link rel="stylesheet" href="/_admin/css/styles_nav.css" />
<!--[if IE]>
```

Quasi-Structured Data



http://www.google.com/#hl=en&sugexp=kjrmc&cp=8&gs_id=2m&xhr=t&q=data+scientist&pq=big+data&pf=p&sclient=psyb&source=hp&pbx=1&oq=data+sci&aq=0&aqi=g4&aql=f&gs_sm=&gs_upl=&bav=on.2,or.r_gc.r_pw.,cf.osb&fp=d566e0fbd09c8604&biw=1382&bih=651

Unstructured Data

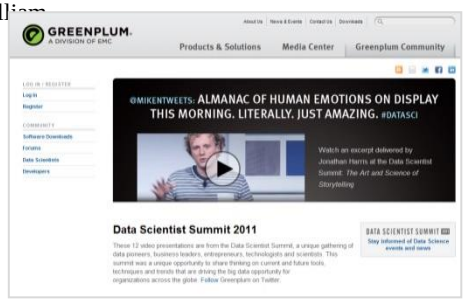
The Red Wheelbarrow, by William Carlos Williams

so much depends
upon

a red wheel
barrow

glazed with rain
water

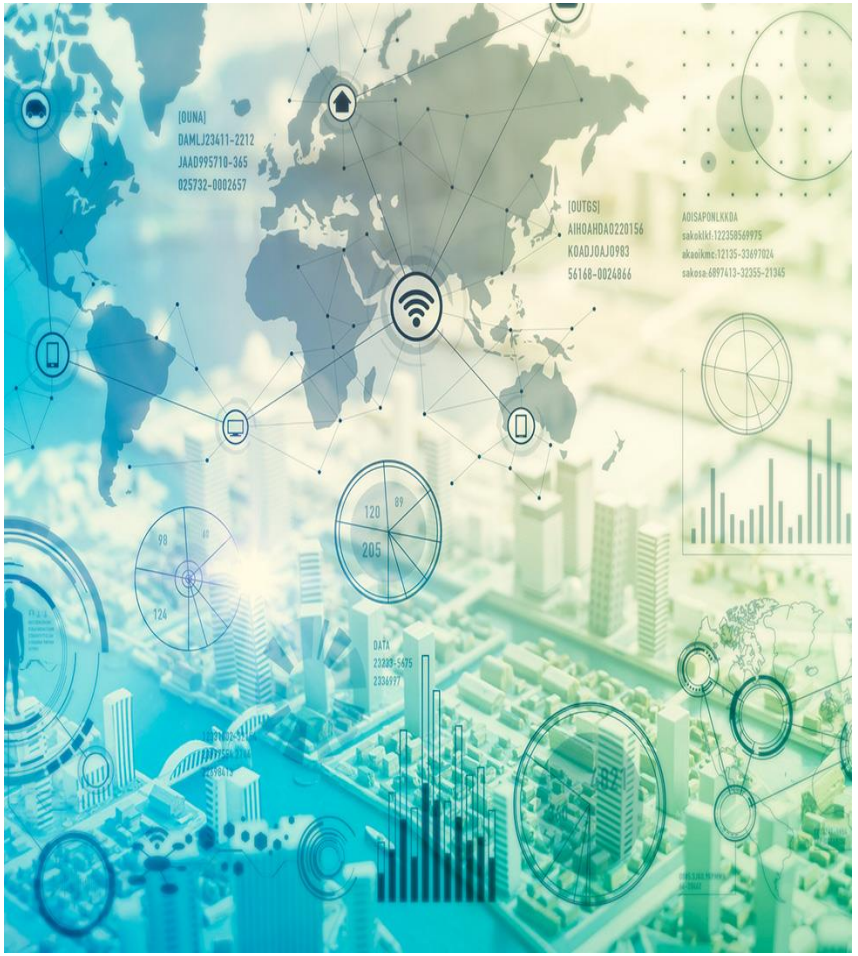
beside the white
chickens.



Big Data ecosystems

As the new ecosystem takes shape, there are four main groups of players within this interconnected web:

1. Data devices
2. Data collectors
3. Data aggregators
4. Data users/buyers



1-Big Data ecosystem—data devices

1 Data devices


Cell phone


GPS


iPod


eBook


Video game


Cable box

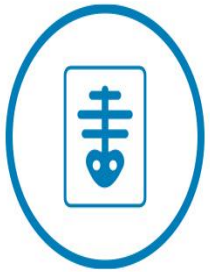

ATM


Credit card reader


Computer


RFID


Video surveillance

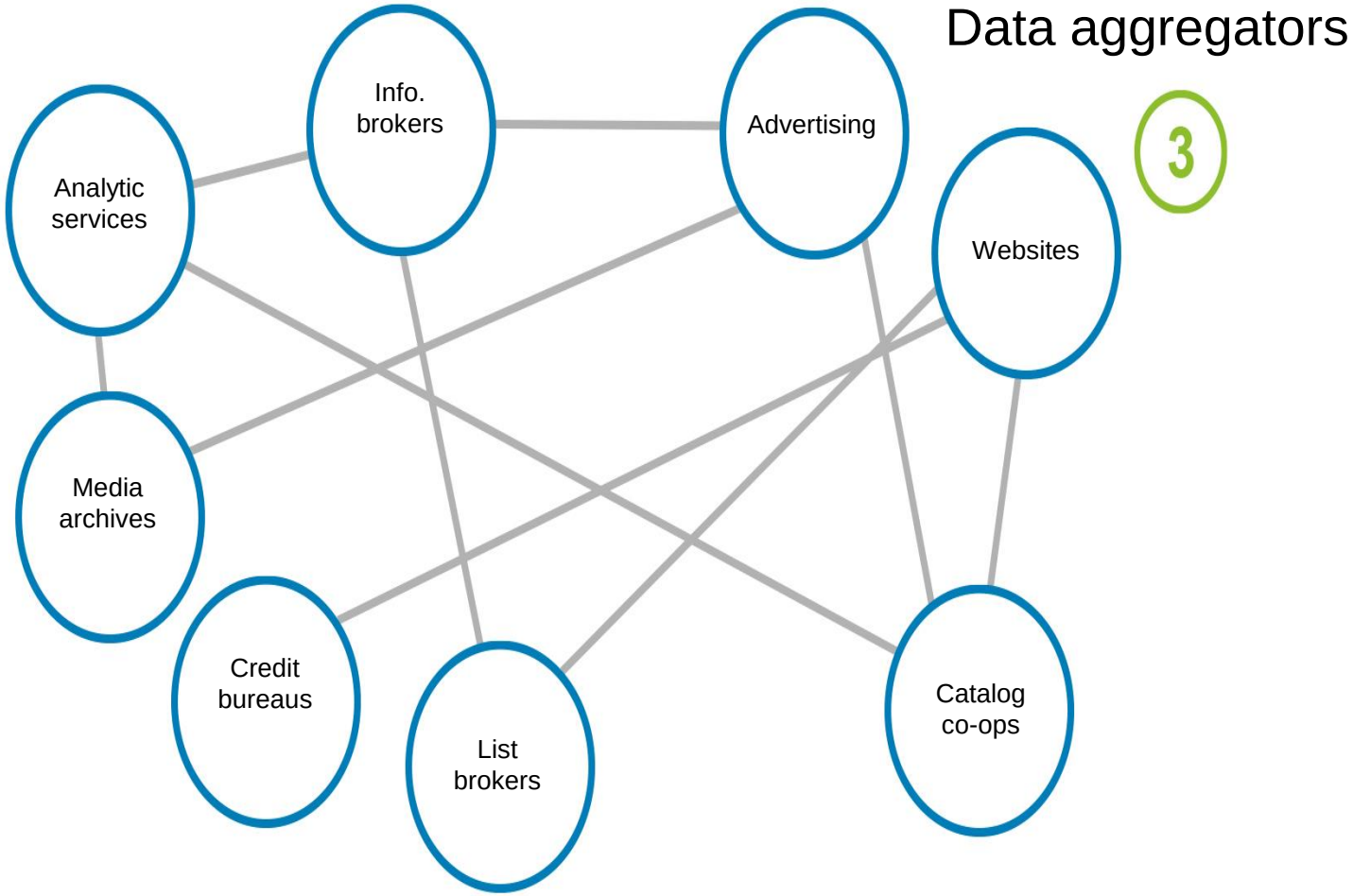

Medical imaging

2-Big Data ecosystem—data collectors

2

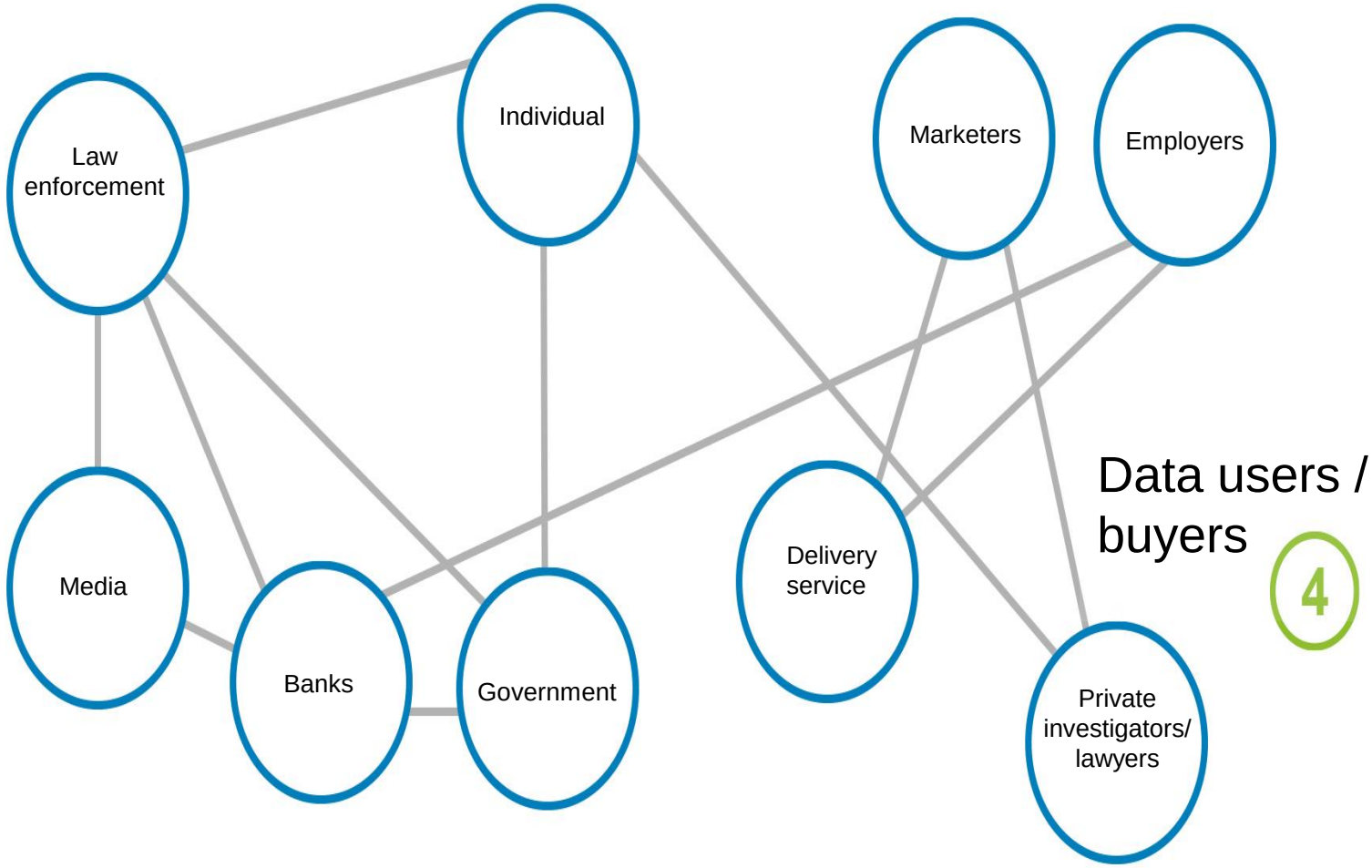
Data collectors

3-Big Data ecosystem—data aggregators



Data aggregators are data mining systems that spread business information online. They collect and share business data with a multitude of sources including search engines like Google.

Big Data ecosystem—data users and buyers



Sources of Big Data

Communications, media, and entertainment

Financial services

Healthcare

Social media

Internet of Things (IoT)

Sources of Big Data—communication, media, and entertainment

- Customer feedback
- Contracts
- Network performance data
- Network traffic
- Network bandwidth usage
- User demographics
- Customer call records
- Social networks
- Viewing or usage habits

Sources of Big Data—financial services

- Transaction records
- Trade messages
- World news
- Audio recordings
- Governance and regulatory data
- Customer feedback

Sources of Big Data—healthcare

- Genomic sequencing and diagnostic imaging
- Medical billing records
- Patient-specific data with socio-demographic
- Hospital care path
- Post-discharge information

Sources of Big Data—social media

- Facebook
- Twitter
- Emails
- Blogs
- LinkedIn
- WhatsApp
- YouTube

Sources of Big Data—Internet of Things (IoT)

- Satellite communications
- Transmitters
- Receiver
- Tracking devices
- Smart phones
- Smart watches
- Public Web

Why Big Data matters?

Organizations that utilize Big Data have:

5X more likelihood of making decisions faster than their competitors.

2X more likelihood of having top-quartile financial performance.

50% more revenue growth than companies that don't use Big Data.

Organizations that don't use Big Data may have:

12% less revenue generated.

Organizations can use Big Data to:

- Enhance customer experience and sales by providing personalized recommendations.
- Detect and prevent cybersecurity threats in real time.
- Make decisions faster by analyzing real-time information.

Test yourself

Which characteristic of big data refers to the diversity in the formats and types of data?

A. Variety

C. Value

B. Variability

D. Volume

Test yourself

Which data asset is an example of unstructured data?

A. News article text

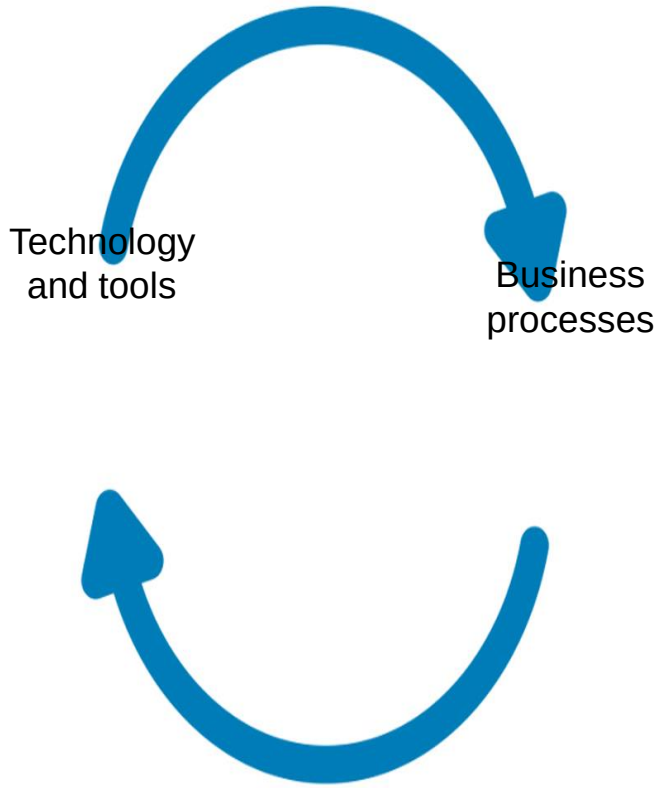
C. Webserver log

B. XML data file

D. Database table

Challenges with Big Data beyond analytics

- Infrastructure
  - Storage
  - Backups/restores
  - Compute
  - Network
- Architectural complexities
- Security
- Data quality
- Ethics



Big data require distributed systems

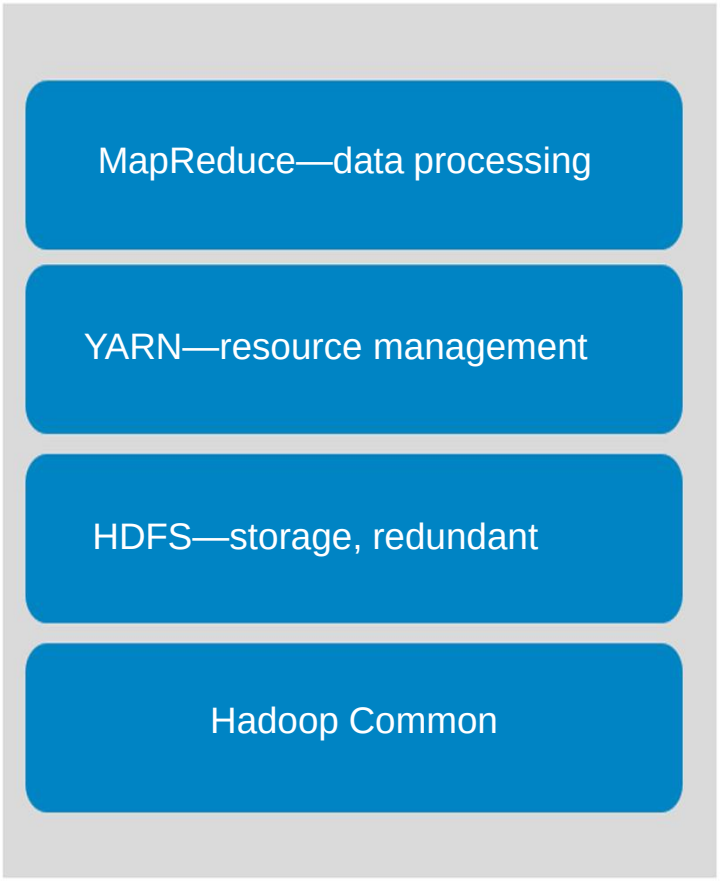
An example hadoop

- Apache™ Hadoop® project develops open-source software for reliable, scalable, distributed computing.*

What is Apache Hadoop?

- The Apache **Hadoop** software library is a **framework** that allows for the distributed processing of large datasets across clusters of computers using simple programming models.*
- This library enables us to use parallel processing capability to handle huge volumes of data using flexible infrastructure.
- To **summarize**, Hadoop offers:
  - A scalable, flexible, and reliable distributed computing Big Data framework for a cluster of systems.
  - Storage capacity.
  - Local computing power—applying commodity hardware.
- Hadoop is not:
  - A database.
  - Simply a data warehouse tool. But it can be used as one.
- Hadoop is written in **Java™**.

Four main components of Apache Hadoop



- MapReduce
- Yet Another Resource Negotiator (YARN™)
- Hadoop Distributed File System (HDFS™)
- Hadoop Common module is a Hadoop Base API —a jar file— for all Hadoop components. All other components work on top of this module.

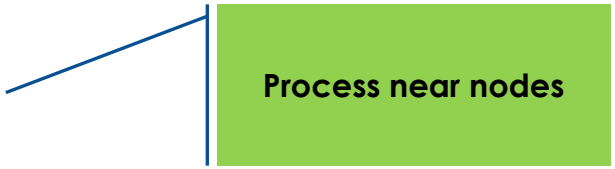
The components are explained in detail in the next few slides.

Hadoop Distributed File System

- Distributed file system designed to run on commodity hardware for storing large files of data with streaming data access patterns
- Highly fault tolerant
- Default storage for the Hadoop cluster
- File system namespace
- Data/File on HDFS is stored in chunks (128 MB default) called blocks

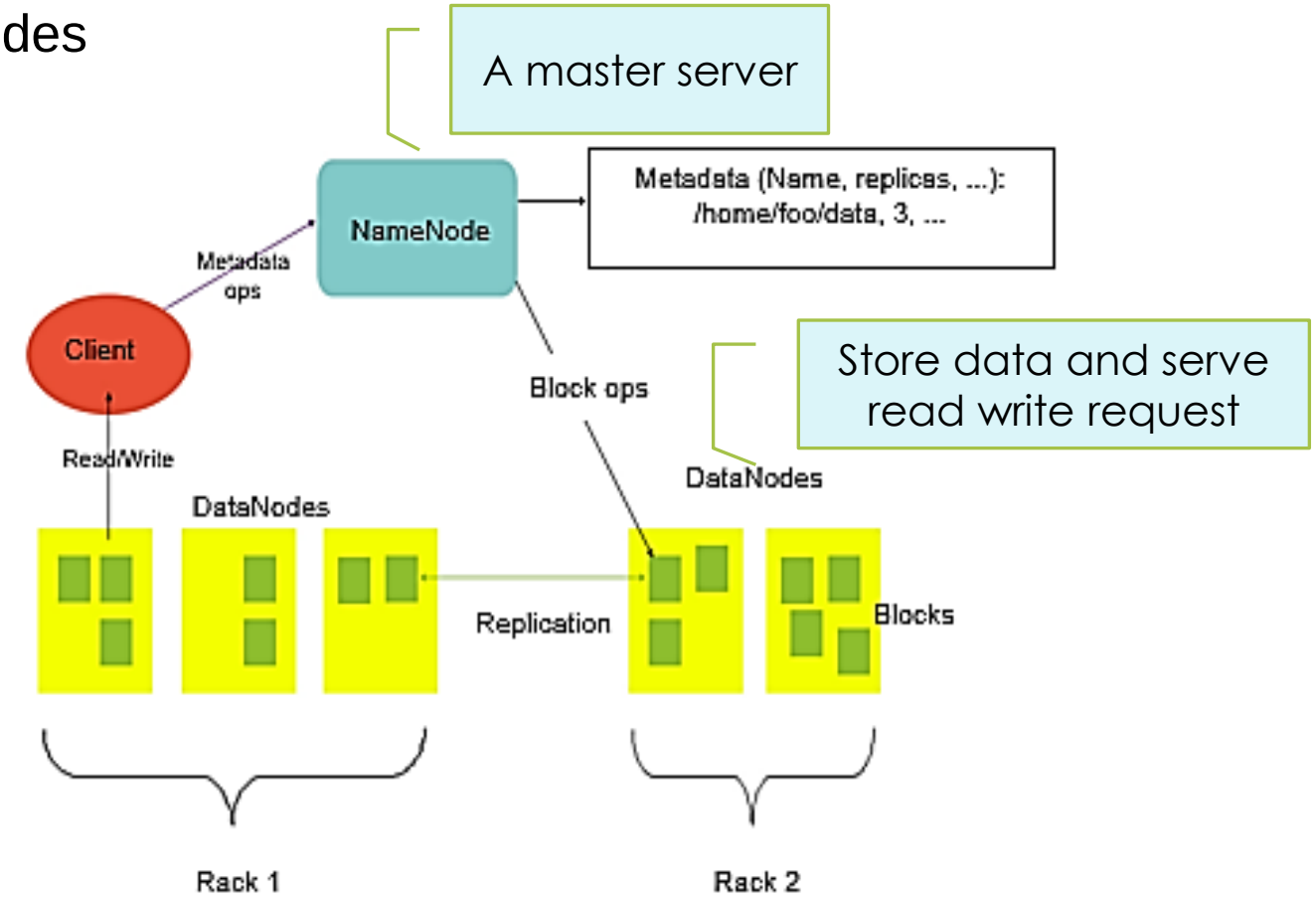
Hadoop Distributed File System—assumption/goals

- Hardware failure is a norm rather than an exception
- Streaming data access
- Large dataset processing
- Simple coherency model
- Moving computation is cheaper than moving data
- Portability across heterogeneous hardware and software platforms



Hadoop Distributed File System—architecture

- Uses a master/slave architecture
- HDFS clusters can contain multiple NameNodes and some DataNodes



NameNode

- **A master server** that manages the file system namespace and regulates access to clients.
- NameNode maps the entire file system and metadata structure—such as permissions, modification timestamp, and so on—into memory.
- It executes file system namespace operations such as opening, closing, and renaming files and directories. It also determines the mapping of blocks to DataNodes.
- Information is stored persistently on the disk in the form of two files: namespace image and edit log.
  - **Namespace image** file contains the Inodes and the list of blocks which define the metadata.
  - **Edit log** contains any modifications that have been performed on the content of the image file.

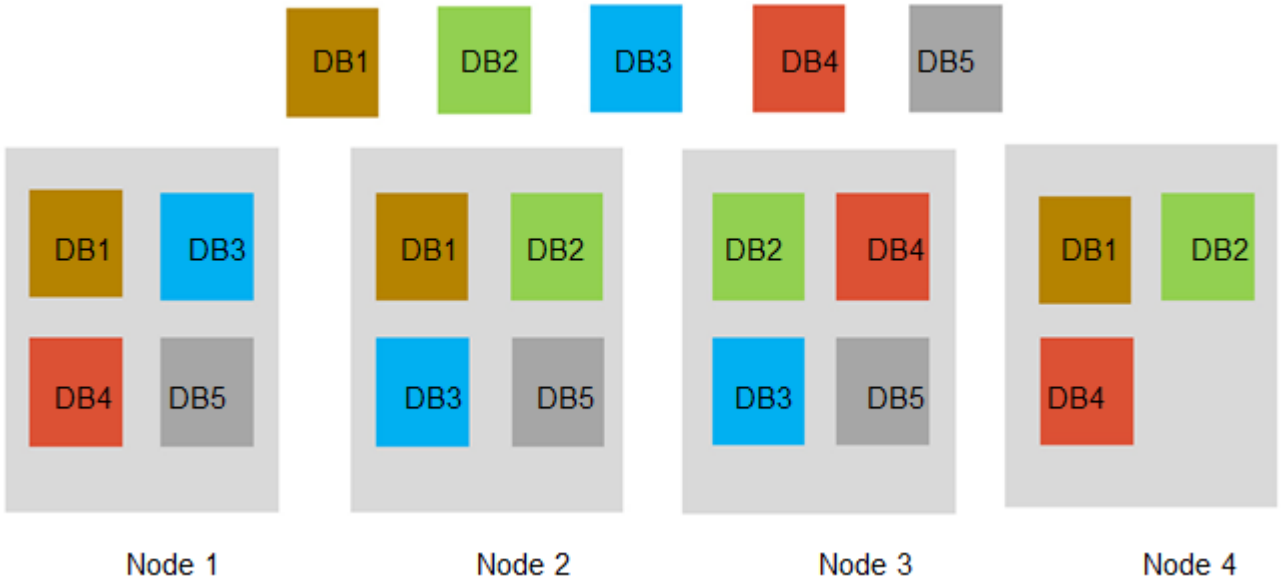
DataNodes

- A file is split into one or more blocks, and these blocks are stored in multiple DataNodes.
- DataNodes are responsible for *serving read* and *write requests* from the file system's clients.
- The DataNodes perform block creation, deletion, and replication upon instruction from the NameNode.
- Reports back to NameNode with the list of stored blocks.
- Bringing computation to data is often more efficient than the reverse.
- DataNodes perform most **CPU-intensive and I/O**-intensive jobs.

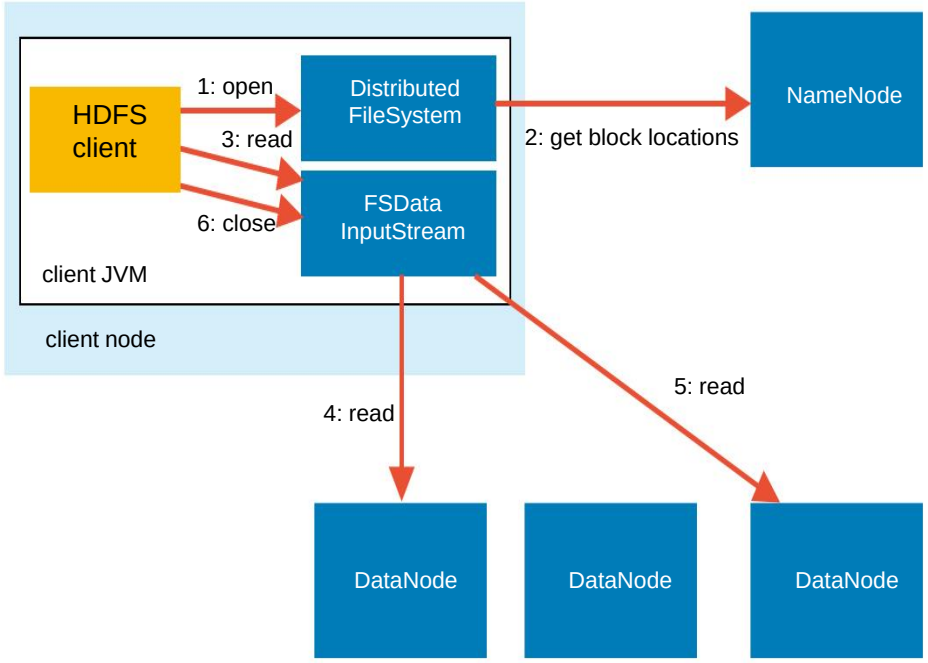
Block replication

Data is replicated more than once in a Hadoop cluster for fault tolerance and availability.

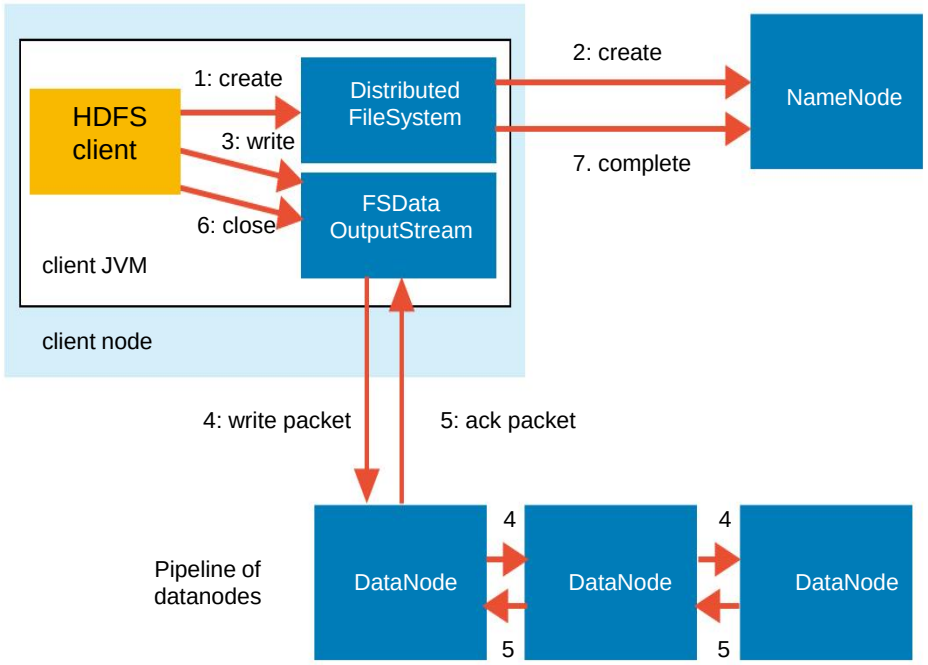
- ✓ Every block of data is replicated on more than one node so, even if a node fails, the data is available on another node.
- ✓ **The replication factor is the number of times a block is replicated. The default is 3 for HDFS, which means every block is replicated three times on three different nodes—see example below; DB stands for data block.**



Hadoop Distributed File System—file read



Hadoop Distributed File System—file write



Hadoop Distributed File System—not ideal in the following situations

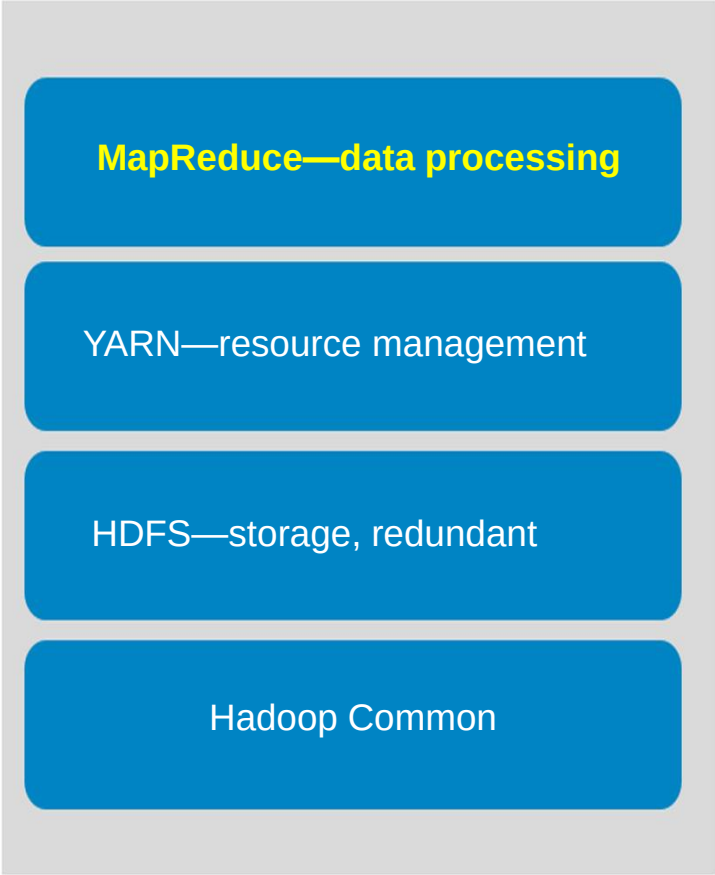
a very high volume of data messages with minimal delay (latency). These networks are designed to support operations that require near real-time access to rapidly changing data.

- Low-latency reads
  - High throughput rather than low latency for small chunks of data

Processing versus time

- Large number of small files
  - Better for large files but not for millions of small files
- Multiple writes
  - Single write per file
  - Writes only at the end of the file

Four main components of Apache Hadoop



- MapReduce
- Yet Another Resource Negotiator (YARN™)
- Hadoop Distributed File System (HDFS™)
- Hadoop Common module is a Hadoop Base API —a jar file— for all Hadoop components. All other components work on top of this module.

The components are explained in detail in the next few slides.

Introduction to MapReduce

What MapReduce is

- A software paradigm for writing applications that process vast amounts of data, multi-terabyte datasets, in-parallel on large clusters—thousands of nodes—of commodity hardware in a reliable, fault-tolerant manner
- Java-based programming paradigm
- A combination of the Map and Reduce models that can be applied to wide variety of business cases
- Handles scheduling and fault tolerance

When to use MapReduce

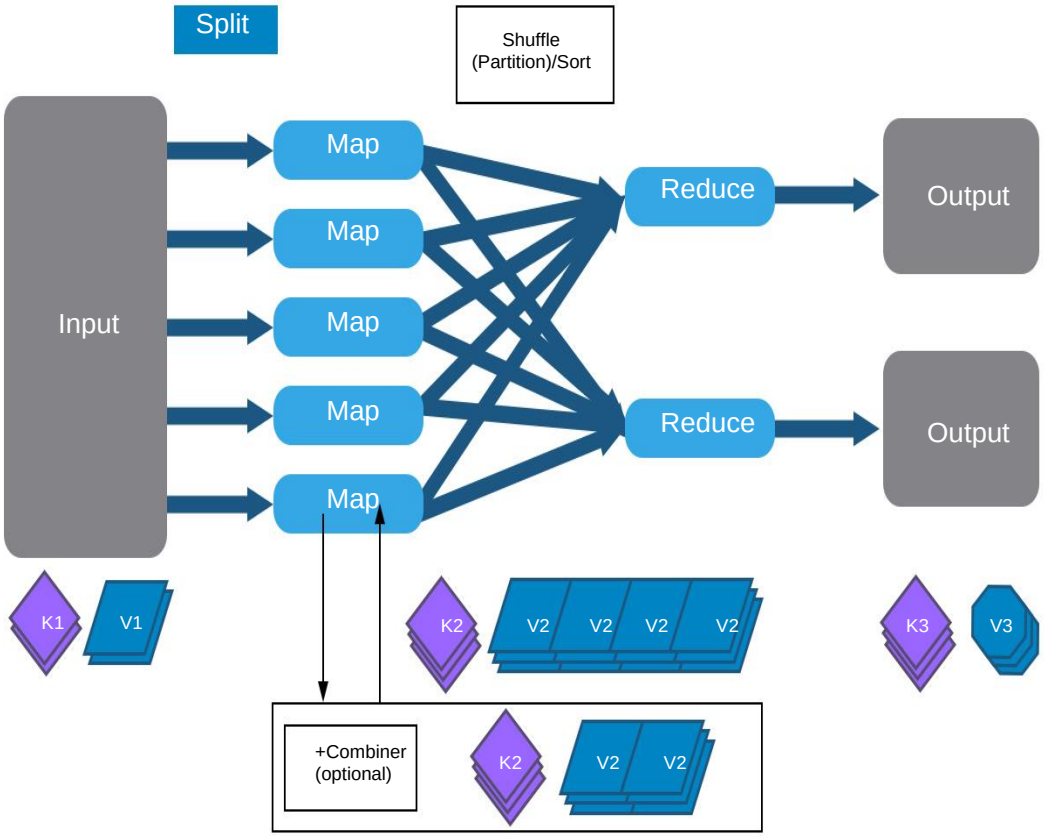
- Problems that are "embarrassingly parallel"
- Examples
  - Word count
  - Reverse index
  - tf-idf
  - Distributed object recognition
  - .....

Term Frequency - Inverse Document Frequency (TF-IDF) is a widely used statistical method in natural language processing and information retrieval. It measures how important a term is within a document relative to a collection of documents (i.e., relative to a corpus).

MapReduce Steps

?

Make sure that each correlated keys are grouped and sorted

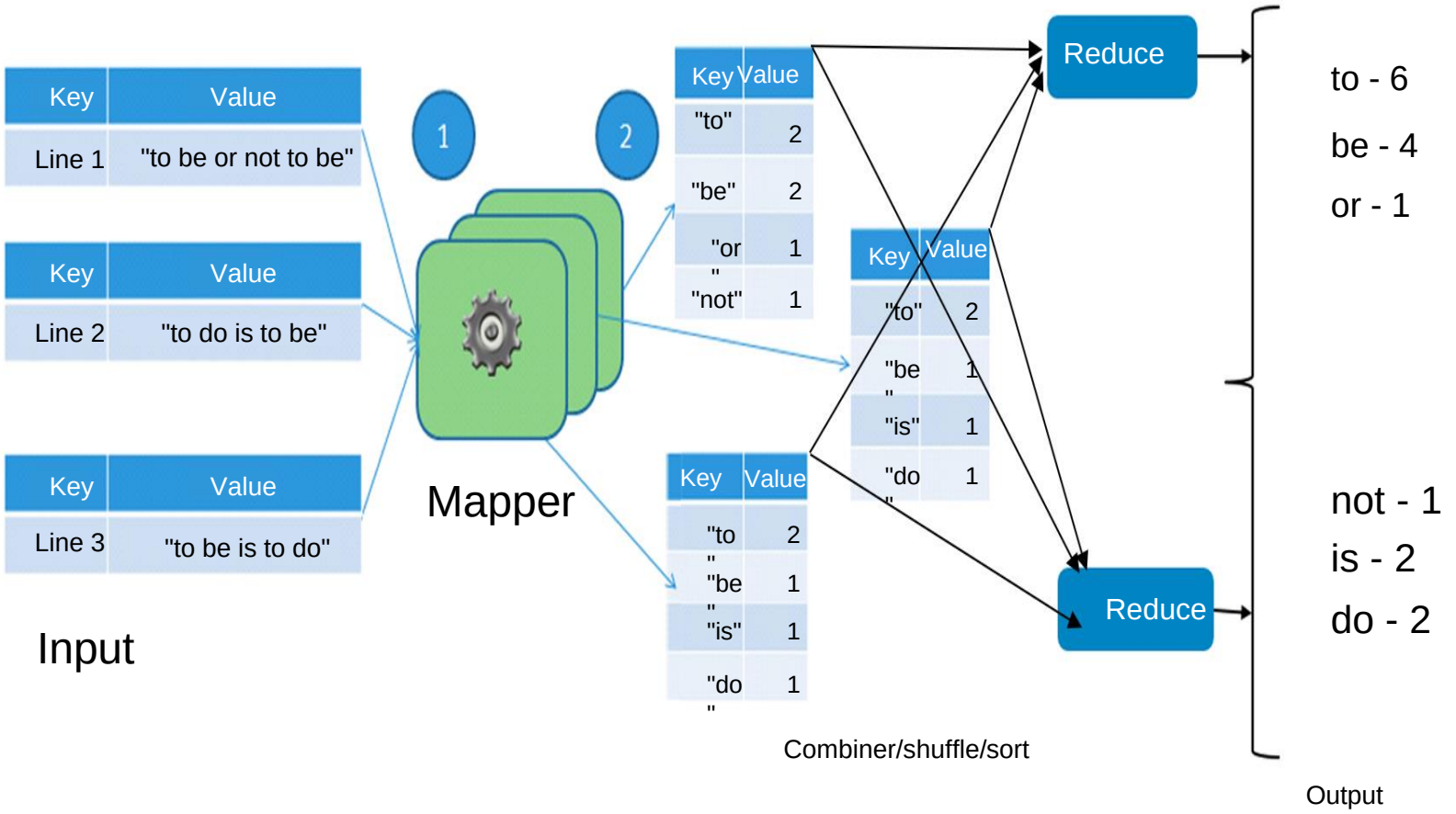


MapReduce paradigm

- Data processing system with two phases—Map and Reduce
- **Map**
  - Performs a map function on key input key-value pairs to generate **intermediate** key-value pairs
- **Reduce**
  - Performs a reduce function on intermediate key-value groups to generate **output** key-value pairs

- Process
  - Input: a set of key-value pairs
  - User supplies two functions
    - Map $(K_1,V_1) \rightarrow list(K_2,V_2)$
    - Reduce $(K_2,list(V_2)) \rightarrow (K_3,V_3)$
  - (K_2,V_2) is an intermediate key-value pair
    - Intermediate key-value groups are created by sorting map output which is input to combiner or reduce function.

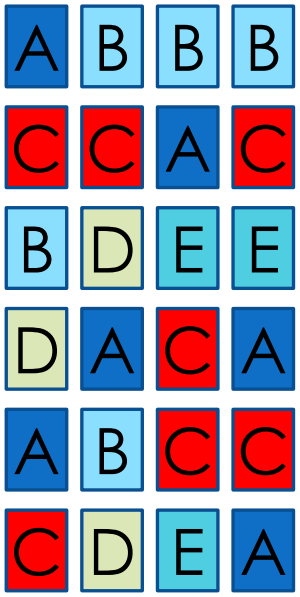
MapReduce—count words in document



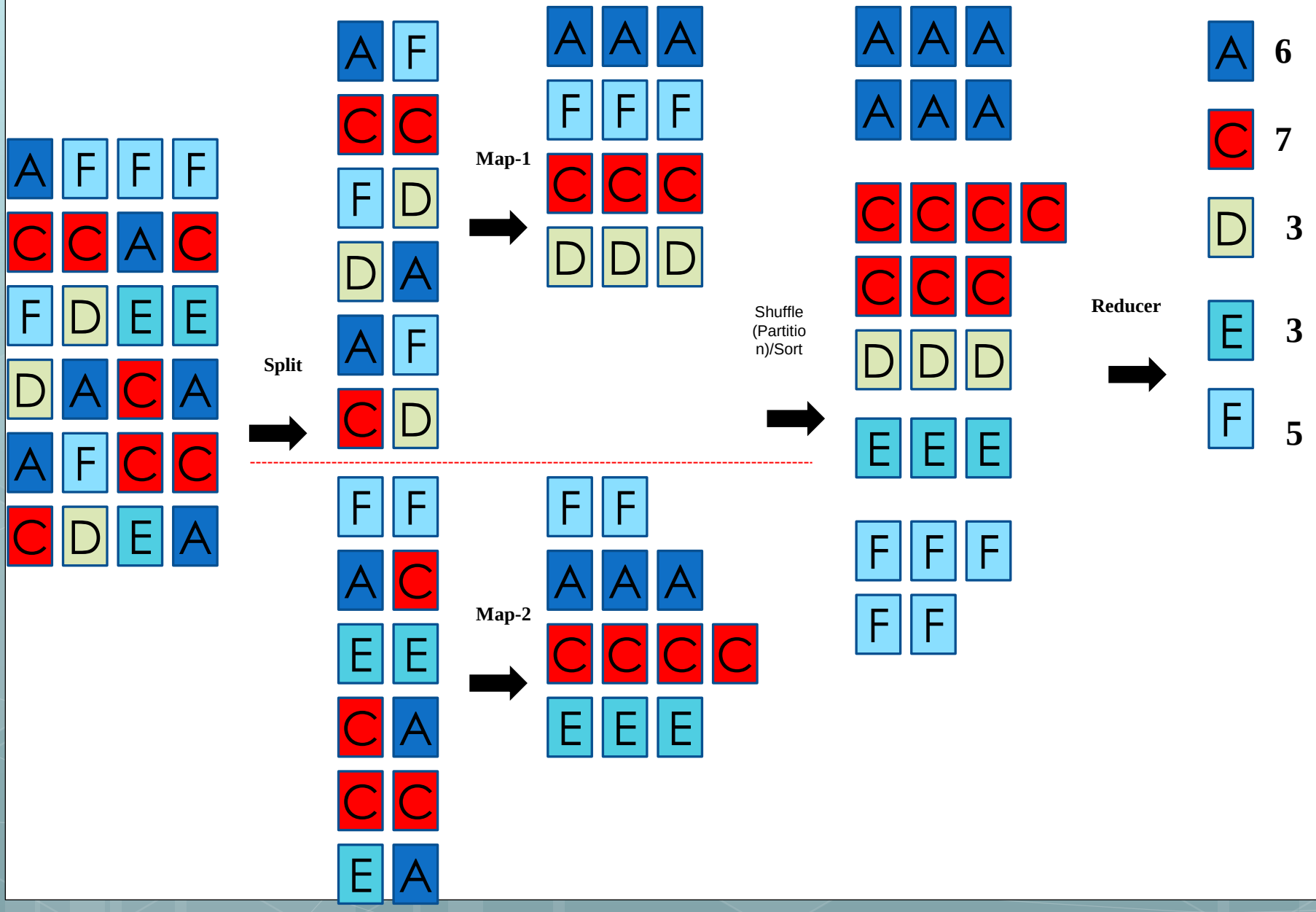
Where MapReduce is used—some examples

- Index building in search engines
- Article clustering for news
- Statistical machine translation
- Data mining
- Spam detection
- Ad optimization

Example



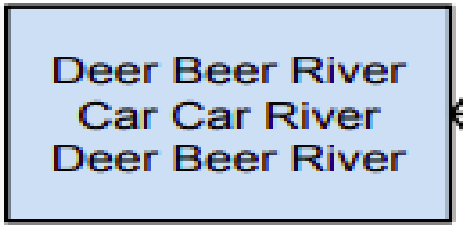
Apply Map Reduce using two mapper vertically and one reduce and output the result


Split
Map-1
Map-2
Shuffle (Partition)/Sort
Reducer
A 6
C 7
D 3
E 3
F 5

Your Turn

Input

Apply Map Reduce using three mapper and four reduce and output the result



Your Turn

input

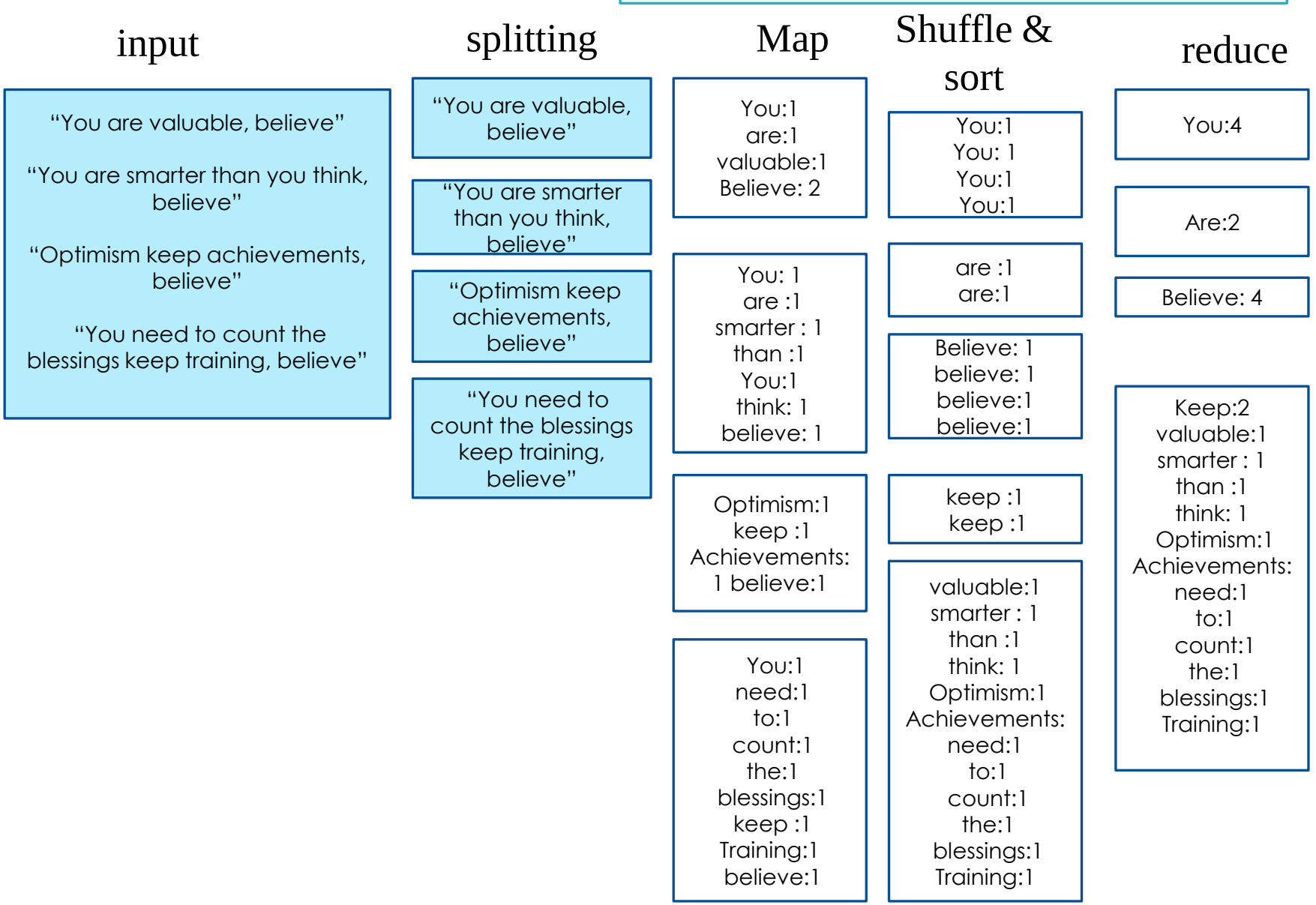
"You are valuable, believe"

"You are smarter than you think, believe"

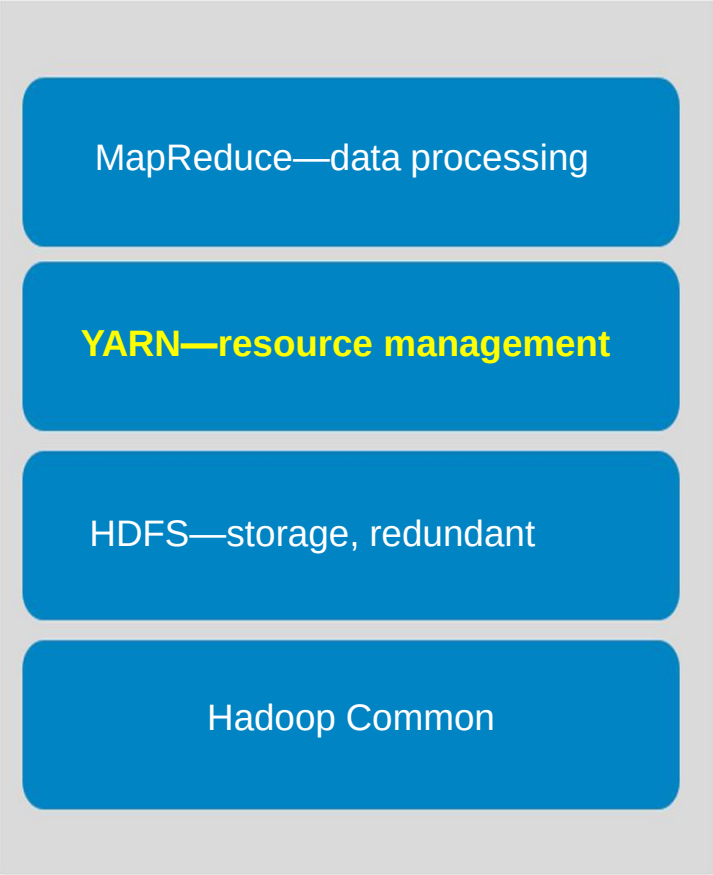
"Optimism keep achievements, believe"

"You need to count the blessings keep training, believe"

Apply Map Reduce using four mapper and four reduce and output the result


input
splitting
Map
Shuffle & sort
reduce
"You are valuable, believe"
"You are smarter than you think, believe"
"Optimism keep achievements, believe"
"You need to count the blessings keep training, believe"
"You are valuable, believe"
"You are smarter than you think, believe"
"Optimism keep achievements, believe"
"You need to count the blessings keep training, believe"
You:1 are:1 valuable:1 Believe: 2
You: 1 are :1 smarter : 1 than :1 You:1 think: 1 believe: 1
Optimism:1 keep :1 Achievements: 1 believe:1
You:1 need:1 to:1 count:1 the:1 blessings:1 keep :1 Training:1 believe:1
You:1 You: 1 You:1 You:1
are :1 are:1
Believe: 1 believe: 1 believe:1 believe:1
keep :1 keep :1
valuable:1 smarter : 1 than :1 think: 1 Optimism:1 Achievements: need:1 to:1 count:1 the:1 blessings:1 Training:1
You:4
Are:2
Believe: 4
Keep:2 valuable:1 smarter : 1 than :1 think: 1 Optimism:1 Achievements: need:1 to:1 count:1 the:1 blessings:1 Training:1

Four main components of Apache Hadoop



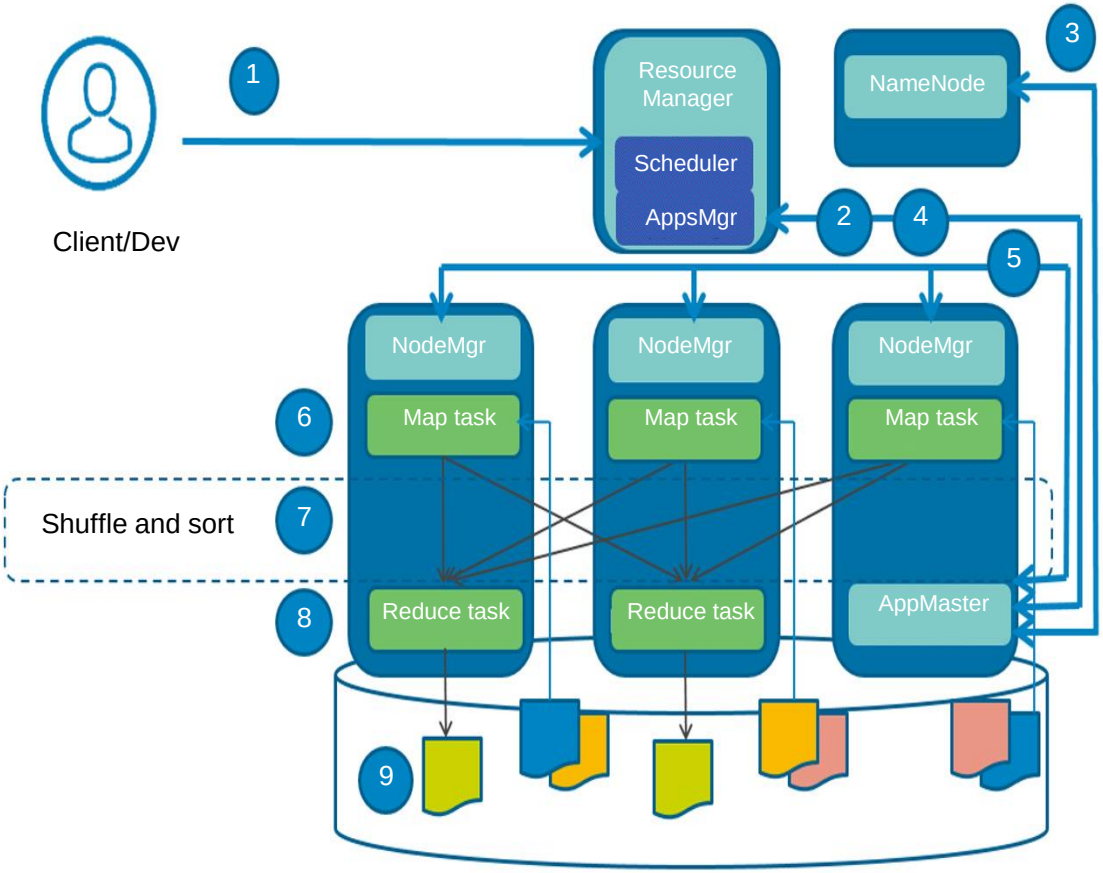
- MapReduce
- Yet Another Resource Negotiator (YARN™)
- Hadoop Distributed File System (HDFS™)
- Hadoop Common module is a Hadoop Base API —a jar file— for all Hadoop components. All other components work on top of this module.

The components are explained in detail in the next few slides.

YARN—Yet Another Resource Negotiator

- **Y**et **A**nother **R**esource **N**egotiator (YARN) is a Hadoop ecosystem component that provides the resource management.
- YARN is also one the most important component of the Hadoop ecosystem. YARN is called as the **operating** system of Hadoop, as it is responsible for managing and monitoring workloads.
- It allows multiple data processing engines such as real-time streaming and batch processing to handle data stored on a single platform.
- Main features of YARN are:
  - Flexibility
  - Efficiency
  - Shared
  - Scalability

YARN—architecture/components



Thank you