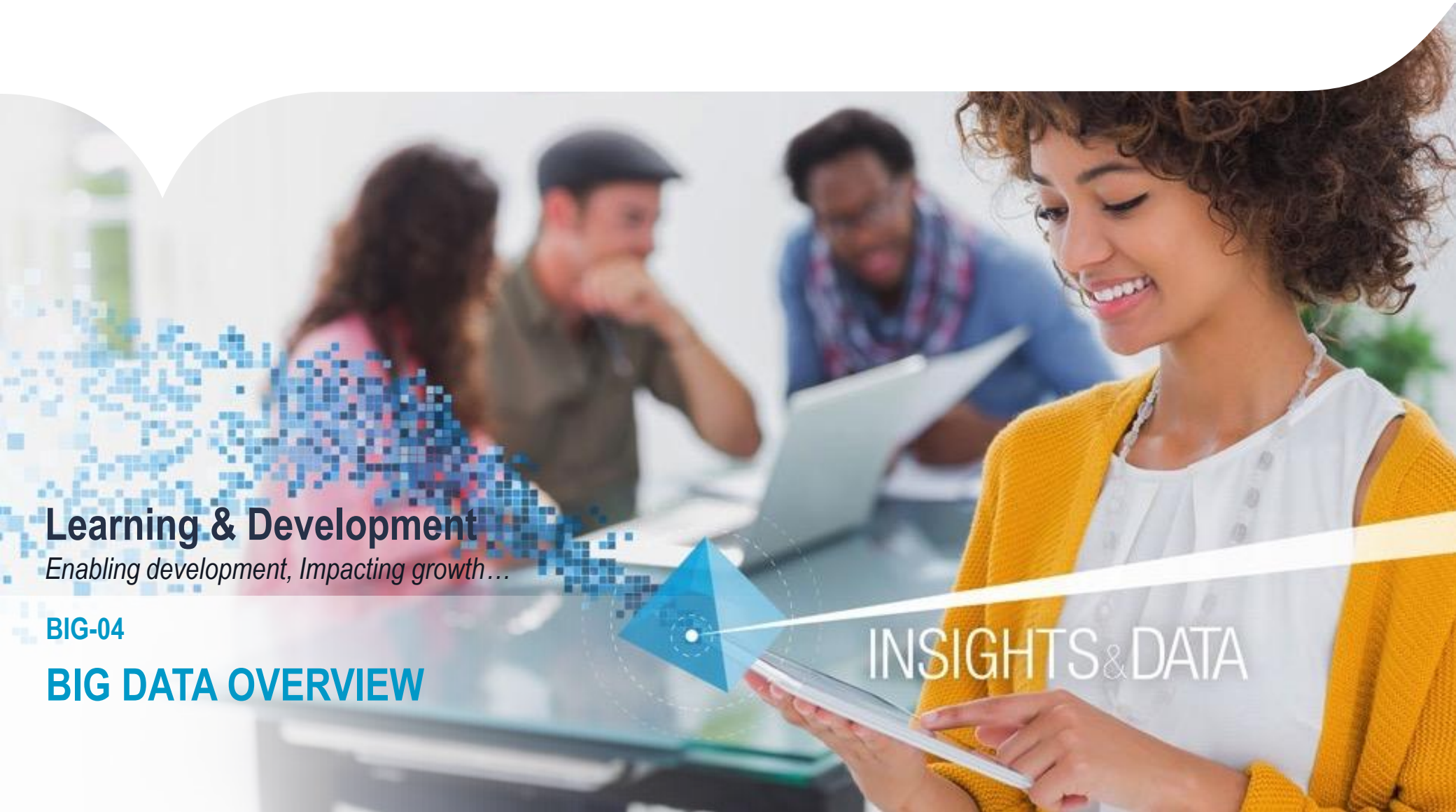
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

People matter, results count.



Learning & Development

Enabling development, Impacting growth…

BIG-04

BIG DATA OVERVIEW

Data Everywhere…

We're generating more data than ever

– Financial transactions
– Sensor networks
– Call Detail Records
– e-mail and text messages
– Social media

For example, every day

– Twitter processes 340 million messages
– Amazon S3 storage adds more than one billion objects
– Facebook generate more than 2.7 billion comments and "Likes"

We must extract the **value** of this data for business benefits

What is Big Data?

"Big Data" is a broad term for any situation involving data that is produced faster than it typically can be consumed into standard data warehousing and analytics operations.

This data can be divided into two major categories, based on the tools used to approach the situation: High Performance Structured Data and Massively Paralleled Captured Data.

Category	High Performance Structured Data	Massively Parallel Captured Data
Context	"Need to Know" the exact answer	"Need to Understand" trends and direction
Target Usage	Event-Driven Processing – "When X and Y happen, Do Z"	Ad-Hoc Analytics – "We Don't Know What We Don't Know"
Tools / Products	SAP HANA, Oracle Exadata/ExaAnalytics, Netezza, Teradata	Cloudera Hadoop , etc
Major Challenge	Data Size / Scalability	Association Accuracy
Major Questions	Do we have the right model for the data? Do we know the quality of the data? Can we scale the performance?	Do we understand the data we are capturing, and can we relate it to other content?

What's Big Data (Cont..)?

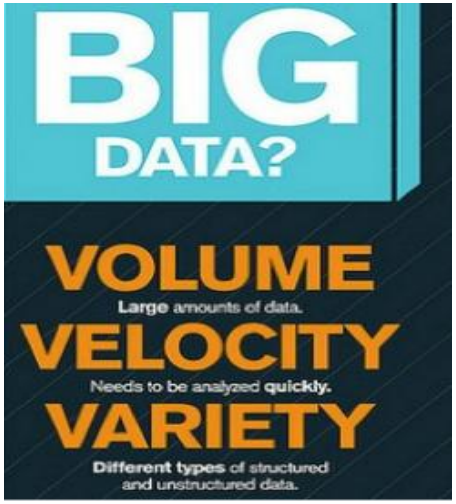
No single definition; here is from Wikipedia:

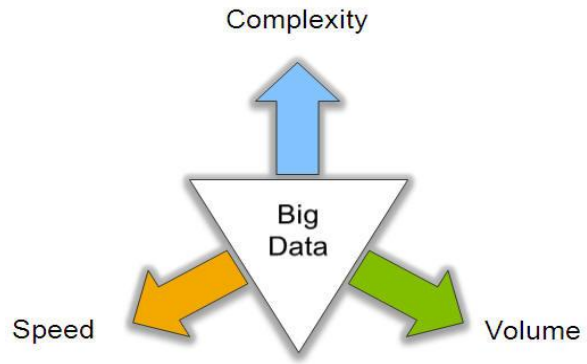
- **Big data** is the term for a collection of data sets so large and complex that it becomes difficult to process using on-hand database management tools or traditional data processing applications.

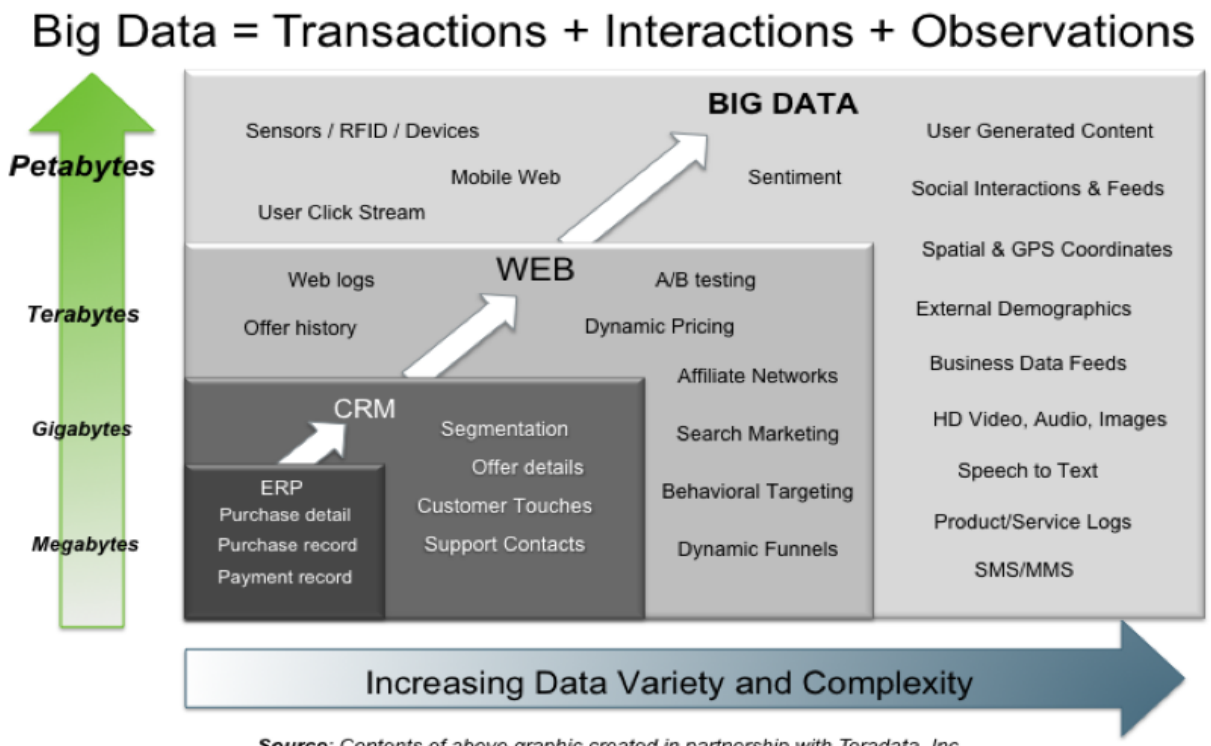
- The challenges include capture, curation, storage, search, sharing, transfer, analysis, and visualization.

- The trend to larger data sets is due to the additional information derivable from analysis of a single large set of related data, as compared to separate smaller sets with the same total amount of data, allowing correlations to be found to "spot business trends, determine quality of research, prevent diseases, link legal citations, combat crime, and determine real-time roadway traffic conditions."

Big Data: 3V's



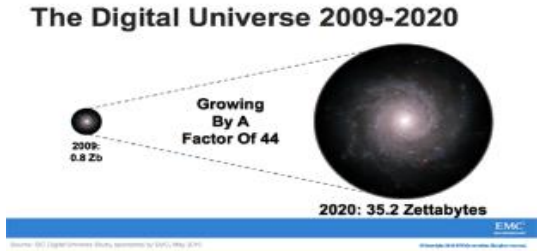


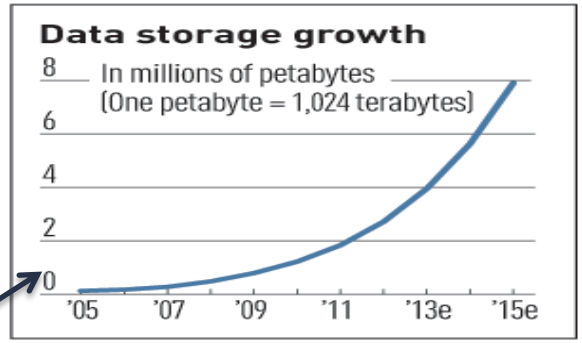


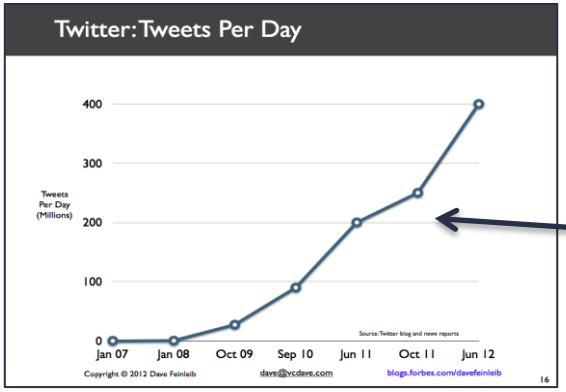
***Source**: Contents of above graphic created in partnership with Teradata, Inc.*

Volume (Scale)

- **Data Volume**
  - 44x increase from 2009 2020
  - From 0.8 zettabytes to 35zb
- Data volume is increasing exponentially

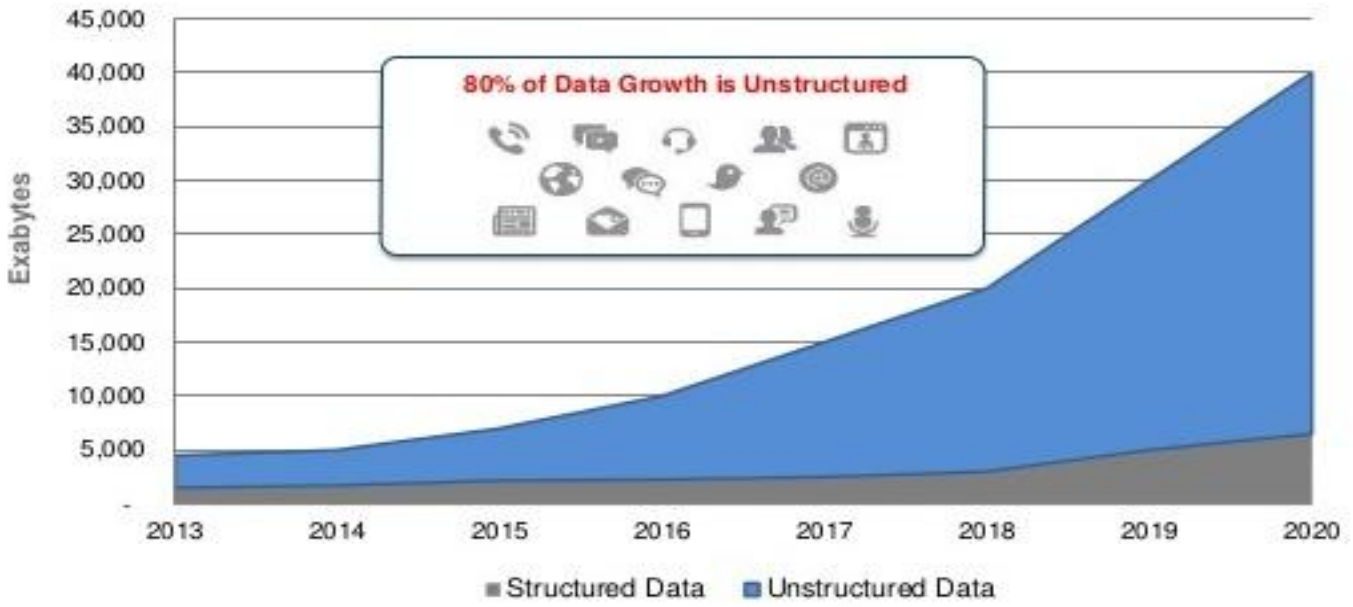






Exponential increase in collected/generated data

Volume (continued)



- By 2020, International Data Corporation predicts the number will reach 40,000 EB, or 40 Zettabytes (ZB) .
- The world's information is doubling every two years. By 2020, there will be 5,200 GB of data for every person on Earth.
- By 2020, the amount of high-value data worth analyzing will double and 60% of information delivered to decision makers will be actionable.

Data Sources for large volume

12+ TBs
of tweet data every day

? TBs of data every day

25+ TBs **of log data every day**

30 billion RFID tags today (1.3B in 2005)

4.6 billion camera phones world wide

100s of millions of GPS enabled devices sold annually

2+ billion people on the Web by end 2011

Velocity(continued)…Real-time/Fast Data

Social media and networks
(all of us are generating data)

Scientific instruments
(collecting all sorts of data)

Mobile devices
(tracking all objects all the time)

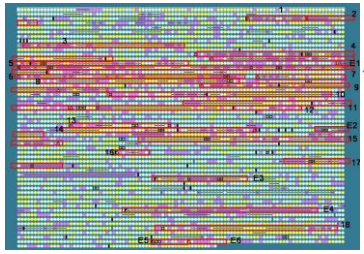
Sensor technology and networks
(measuring all kinds of data)

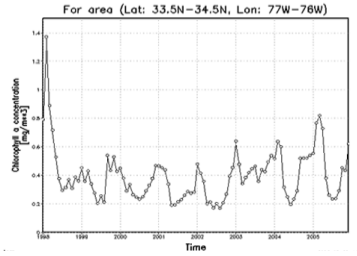
- The progress and innovation is no longer hindered by the ability to collect data
- But, by the ability to manage, analyze, summarize, visualize, and discover knowledge from the collected data in a timely manner and in a scalable fashion

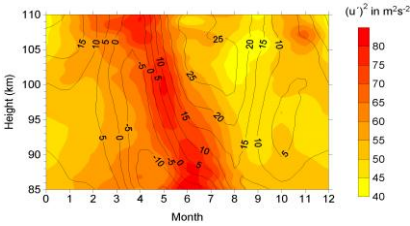
Variety (Complexity)

- Relational Data (Tables/Transaction/Legacy Data)
- Text Data
- XML Data
- Streaming Data
  - Data changing within fraction of seconds
- Audio Data
- Video Data
- Logs Data
- Graph Data
  - Social Network
- A single application may generate/collect different types of data

- Big Public Data (online, weather, finance, etc)





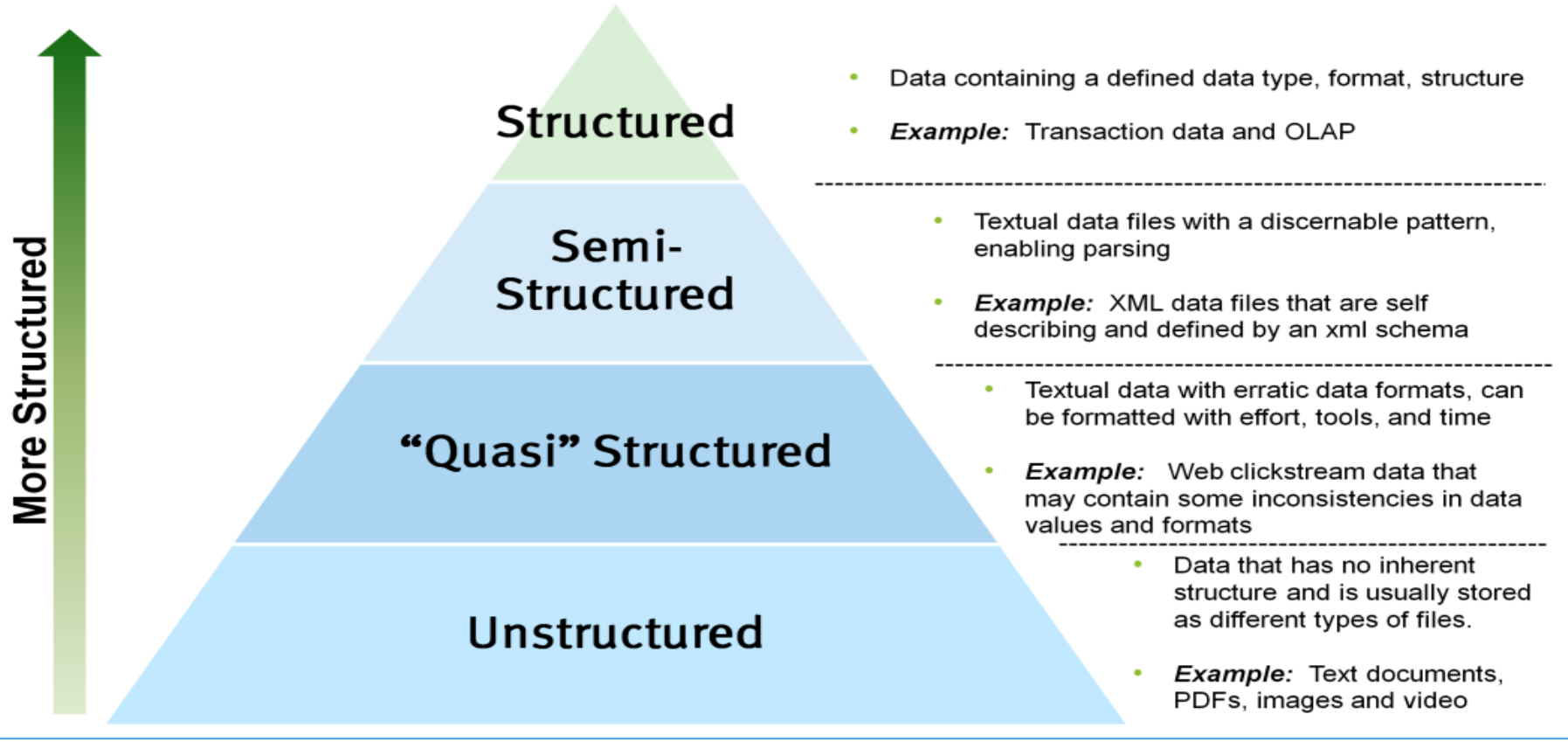


To extract knowledge➔ All these types of data need to linked together

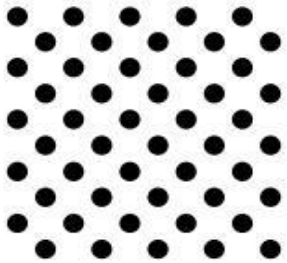
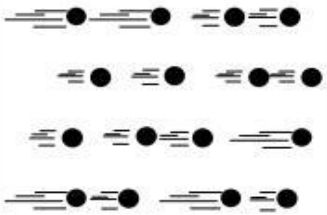
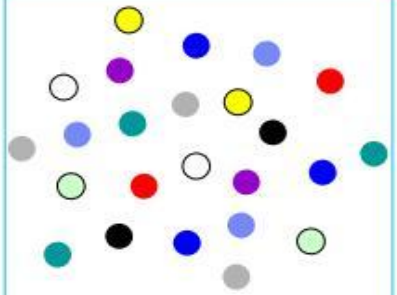
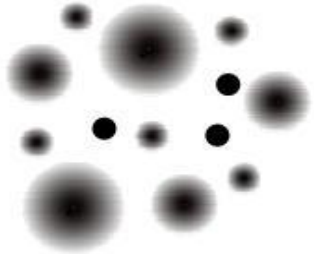
Variety(continued)…Types of Data in Big Data

Big Data Characteristics: Data Structures

Data Growth is Increasingly Unstructured



Some Make it 4V's

Volume	Velocity	Variety	Veracity*
			
Data at Rest	**Data in Motion**	**Data in Many Forms**	**Data in Doubt**
Terabytes to exabytes of existing data to process	Streaming data, milliseconds to seconds to respond	Structured, unstructured, text, multimedia	Uncertainty due to data inconsistency & incompleteness, ambiguities, latency, deception, model approximations

Challenges !

Scalability Issues

How do we process all this information,
Large Scale Data Storage
Large Scale Data Processing / Analytics

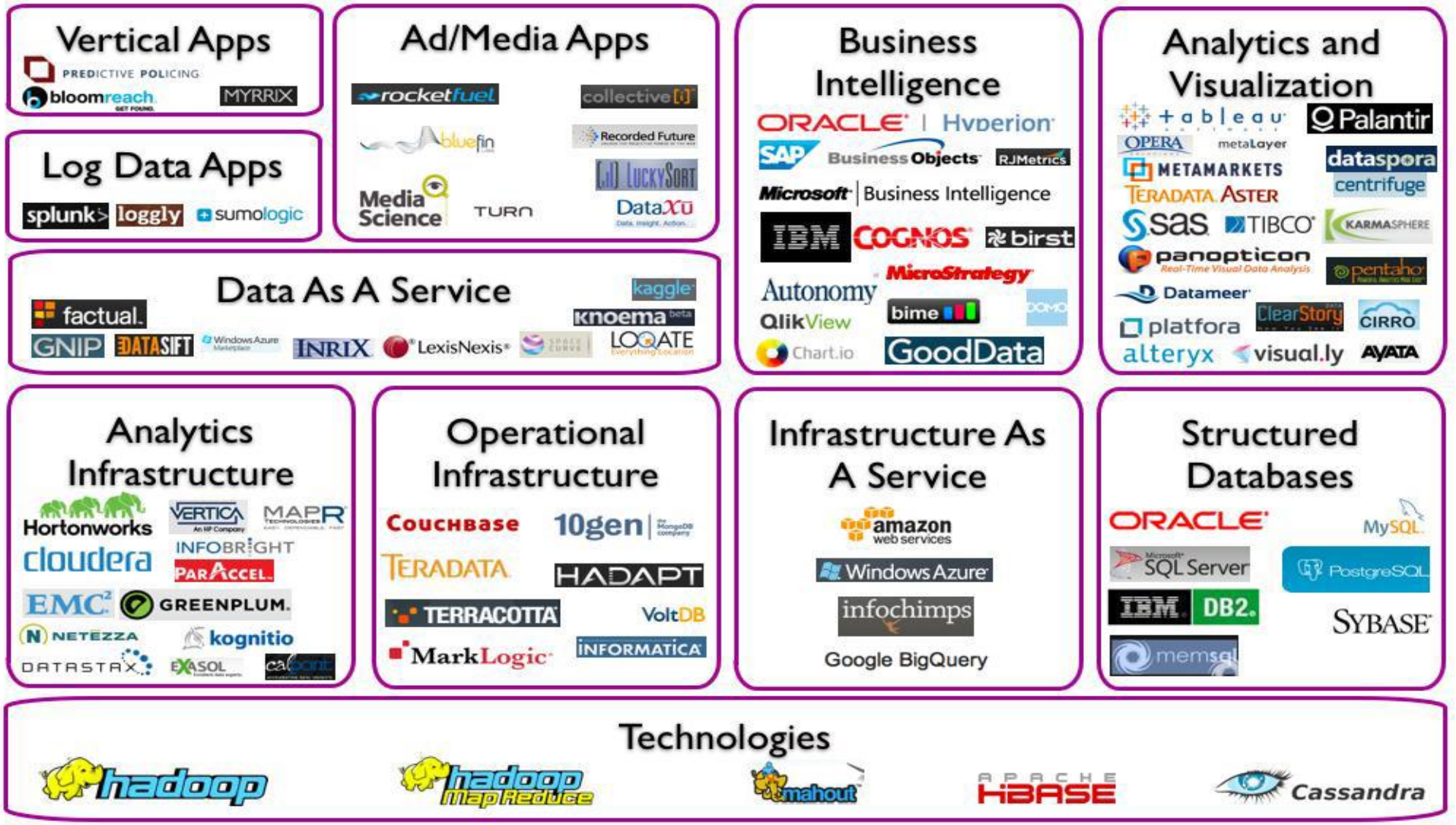
Storage

Over the period (decade) Size and cost of storage became affordable
E.g. in 2012 Storage space cost for storing 3000 GB data is approx $0.05 per GB

Performance

However the disk transfer rate , did not scaled up
To read 3000 GB data, we end up 210 MB/s and disk read time will be approx 4 hours.

The Big Data Landscape



Quiz

1. What type of data Big Data deals with?

A. Only Structured data
B. Only Unstructured data
C. All types of Data

2. Which of following characteristic of big data deals with Speed?

A. Volume
B. Variety
C. Velocity

Quiz-Answers

1. What type of data Big Data deals with?

A. Only Structure data
B. Only Unstructured data
C. All types of Data

C: All types of Data

2. Which of following charactistics of big data deals with Speed?

A. Volume
B. Variety
C. Velocity

C: Velocity

Quiz(continued)

3. Map each of the below data as structured ,semi structured or unstructured.

A. PDF
B. E-mail
C. Database table
D. XML

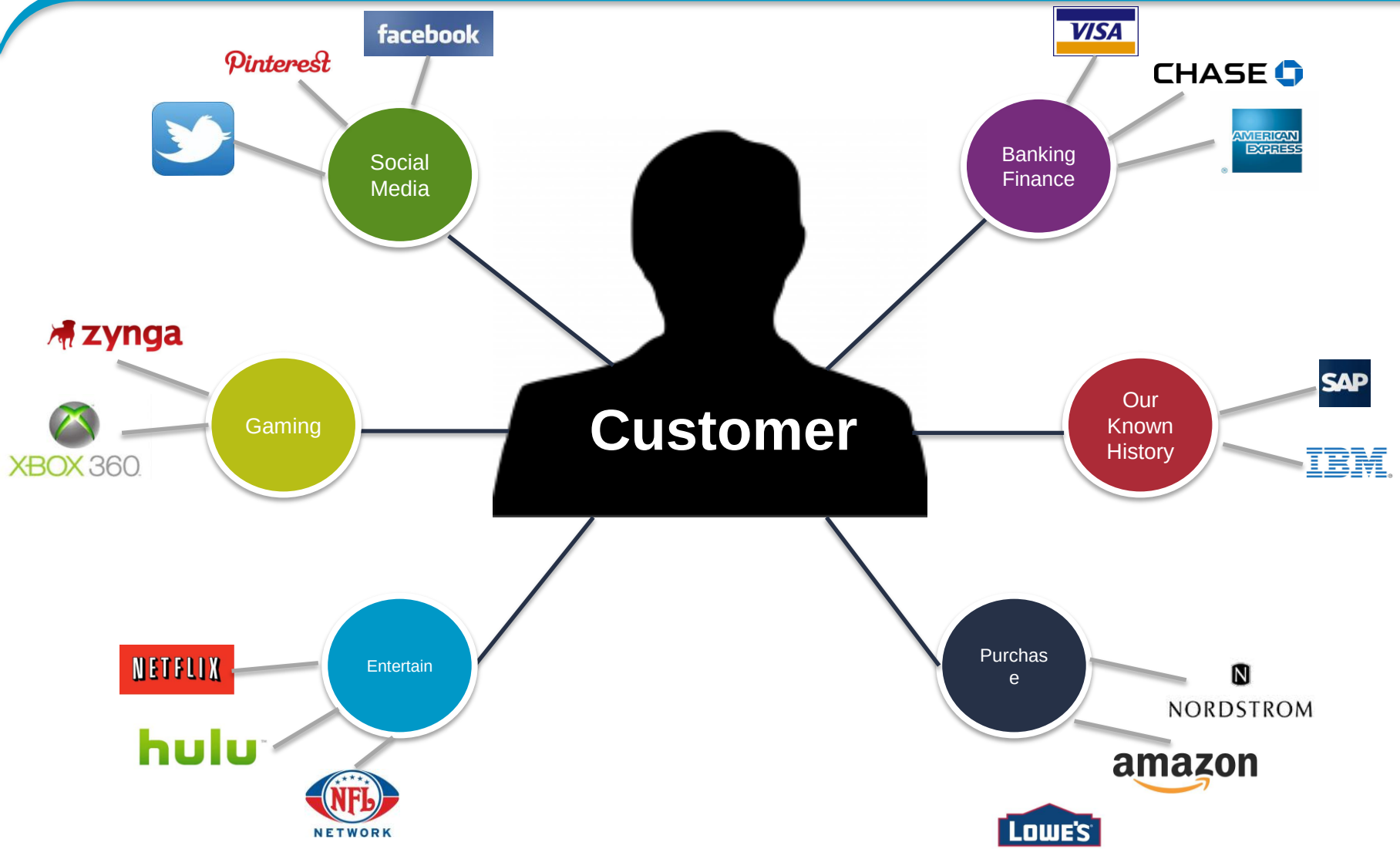
Quiz-Answer

3. Map each of the below data as structured ,semi structured or unstructured.

A. PDF
B. E-mail
C. Database tables
D. XML

PDF→ Unstructured
E-mail → Unstructured
Database tables → **Structured**
XML File → **Semi structured**

Explosion of Data- A Single View to the Customer



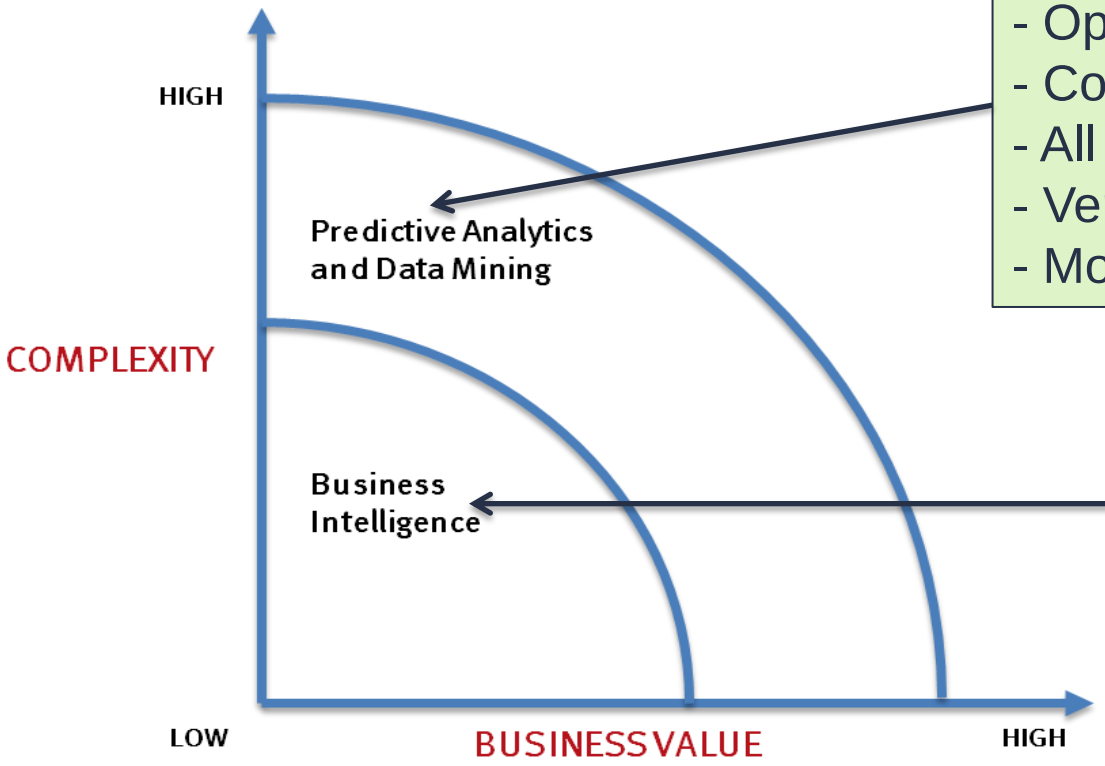
Explosion of Data(continued)- The Model Has Changed…

- **The Model of Generating/Consuming Data has Changed**

Old Model: Few companies are generating data, all others are consuming data

New Model: all of us are generating data, and all of us are consuming data

What's driving Big Data



- Optimizations and predictive analytics
- Complex statistical analysis
- All types of data, and many sources
- Very large datasets
- More of a real-time

- Ad-hoc querying and reporting
- Data mining techniques
- Structured data, typical sources
- Small to mid-size datasets

What is really innovative behind the Big Data buzz word ?

Hadoop & Cloud : Low cost ways to store, manage and analyze massive volumes of data

No SQL : New ways to organize and analyze non-structured data

Cost

Event Processing tools : Ability to analyze and detect trends in real time streaming events (monitoring, Next Best Action, Fraud...)

In-memory technologies : A new way to guarantee response time even for very complex calculations

Speed

Explosion of Analytics usage : R is open source, High performance statistics make them usable by every needed process

Multiple external data sources : Easier to tackle new data sources e.g. social media, traffic, GPS sensors, open data.

Insights

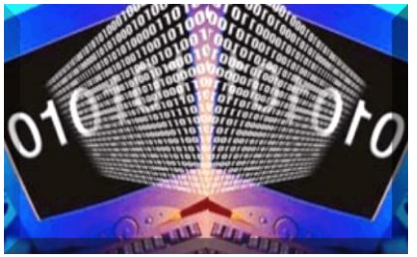
Applications for Big Data Analytics

Smarter Healthcare

Multi-channel

Finance

Log Analysis



Homeland Security

Traffic Control

Telecom

Search Quality

Manufacturing

Trading Analytics

Fraud and Risk

Retail: Churn

Who uses Hadoop?

- 1&1
- A9.com
- Amazon.com
- AOL
- Apple
- Booz Allen Hamilton
- EHarmony
- eBay
- Facebook
- Fox Interactive Media
- Freebase
- Hewlett-Packard
- IBM
- ImageShack
- ISI
- Joost
- Last.fm
- LinkedIn
- Meebo
- Metaweb
- NetFlix
- The New York Times
- Ning
- Powerset (now part of Microsoft)
- Rackspace
- Razorfish
- StumbleUpon
- Twitter

Big Data Use Cases(continued)

- Retail
  - Customer churn prevention
  - Point of sales transaction analysis
  - 360 degree customer view

- E-commerce
  - Click stream analysis
  - Recommendation engine
  - Ad targeting

- Government Sector
  - UID enrollment
  - Social welfare schemes

Quiz

1. Name different types of data in big-data.

A. Structured and Semi-structured

B. Quasi-Structured and Unstructured

C. None of the above

D. A&B

Quiz-Answer

1. Name different types of data in big-data.

A. Structured and Semi-structured

B. Quasi-Structured and Unstructured

C. None of the above

D. A&B

D: A & B

History of Hadoop

- Oct 2003: Google File system paper published
- Dec 2004: Jeffrey Dean & Sanjay Ghemawat from Google published MapReduce paper called “MapReduce: Simplified Data Processing on Large Clusters”
- Jan 2006: Above MapReduce Paper inspired Doug cutting, a yahoo employee then to develop an open source implementation of MapReduce framework
- Jan 2006: Hadoop subproject created as extension of Apache Nutch project, created by Doug Cutting.
- Apr 2006: Hadoop 0.1.0 released
- May 2006: Yahoo deploys 300 machine Hadoop cluster
- 2008: Cloudera, one of the major distributor of Hadoop founded

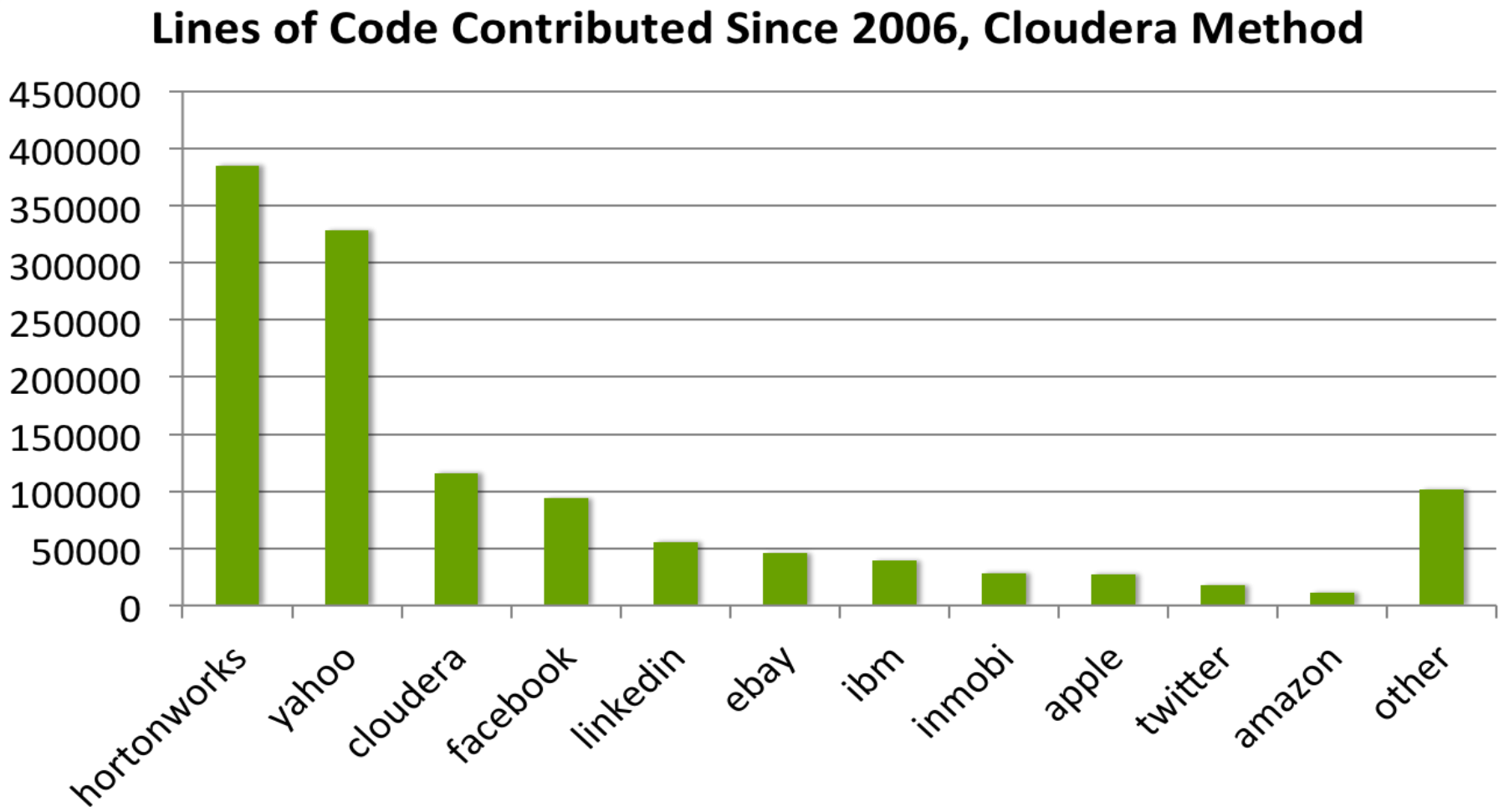
Hadoop was named after Doug Cutting’s son’s toy elephant.

History of Hadoop(continued)

- Apr 2007: Yahoo runs 2 clusters of 1,000 machines
- Jul 2008: Hadoop wins TeraByte sort benchmark
  (1st time a Java program won this competition)
- Jun 2010: Yahoo 4,000 nodes/70 petabytes
- Jun 2010: Facebook 2,300 clusters/40 petabytes
- Dec 2011: Apache Hadoop release 1.0.0 available
- 2011: Hortonworks, another major Hadoop distributor founded
- Oct 2013: Apache Hadoop release 2.2.0 (YARN)
- Dec 2015: Apache Hadoop release 2.6.3 available
- Feb 2016: Apache Hadoop release 2.6.4 available

- Source: http://hadoop.apache.org/#News
  - https://en.wikipedia.org/wiki/Apache_Hadoop#Papers

Contributions 2006 - 2011



(Source: . http://hortonworks.com/blog/reality-check-contributions-to-apache-hadoop/)

Quiz

1. Who was the creator of Hadoop?
   A. Sanjay Ghemawat
   B. Michael Franklin
   C. Doug Cutting
   D. Jeffrey Dean

2. What was Hadoop named after?
   A. Creator's favorite circus act
   B. The toy elephant of creator's son
   C. Creator's high school rock band
   D. A sound Creator's laptop made during Hadoop development

Quiz-Answers

1. Who was the creator of Hadoop?

   A. Sanjay Ghemawat
   B. Michael Franklin
   C. Doug Cutting
   D. Jeffrey Dean

C: Doug Cutting

2. What was Hadoop named after?

   A. Creator's favorite circus act
   B. The toy elephant of creator's son
   C. Creator's high school rock band
   D. A sound Creator's laptop made during Hadoop development

B: The toy elephant

What is Hadoop?

Consider a requirement where you need to perform pattern matching over a text file.

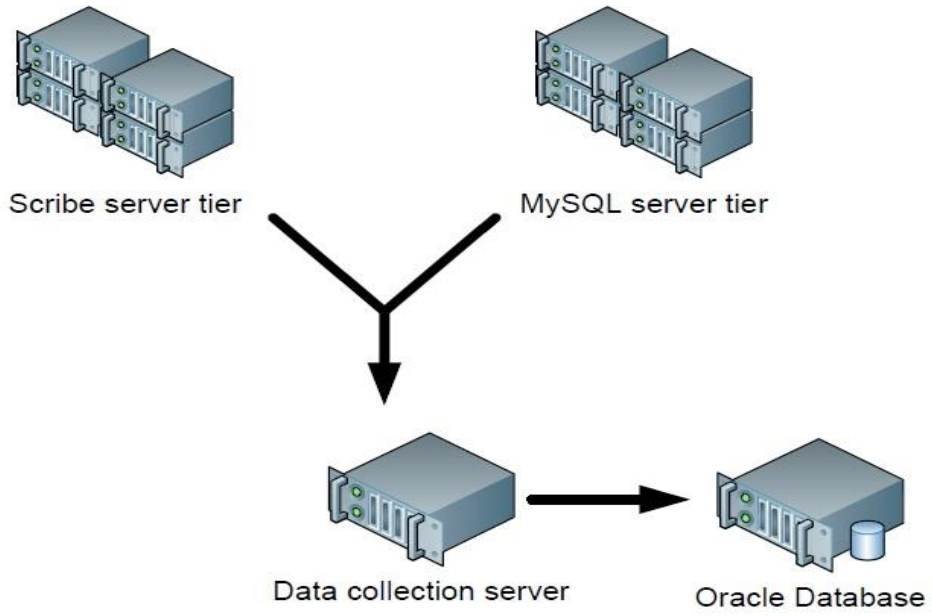
Soln: This seems very easy and Can be achieved in few seconds.

Now imagine the some problem to be executed over a crawled data which is a few 100TB in size spread over 10 nodes ....

Soln: Run the same program over 10 nodes.

Facebook – A practical example

Consider the example of facebook, their data has grown upto 15TB/day by 2012 and in future it will produce a data of higher magnitude.
They have many web servers and huge MySql(profile,friends etc.) servers to hold the user data.



Facebook – A practical example

Now the main aim was to run various reports on these huge data

eg: 1) Ratio of men vs. women users for a period.
2) No of users who commented on a particular day.

Soln: For this requirement they had scripts written in python which uses ETL processes.

But as the size of data increased to this extent these scripts did not work.
Hence their main aim at this point of time was to handle data warehousing and their home ground solutions were not working.

This is when Hadoop came into the picture..

Distributed File System (DFS)

Read 1 TB Data

1 Machine

50 Minutes

10 Machines

5 Minutes

What is Hadoop

- Hadoop is not a :
  - Database
  - Big Data
  - Networking Concept
  - Data warehouse
  - Programming Language

Then What Hadoop is.....??

What is Hadoop(continued)

- **Hadoop** is a framework that allows distributed processing of large data sets across clusters of commodity computers using simple programming models.

  Definition In Depth:

  - Distributed Processing :
    - Data is processed in multiple machines in a distributed manner
  - Large Data sets:
    - Large data sets in this context means files that are hundreds of megabytes, gigabytes, or terabytes in size
  - Clusters of commodity computers :
    - Cheap hardware (not expensive servers) are used to create a cluster
  - Simple Programming Model:
    - Map Reduce/Spark is used as a programming model to manipulate/process the data

Characteristics of Hadoop

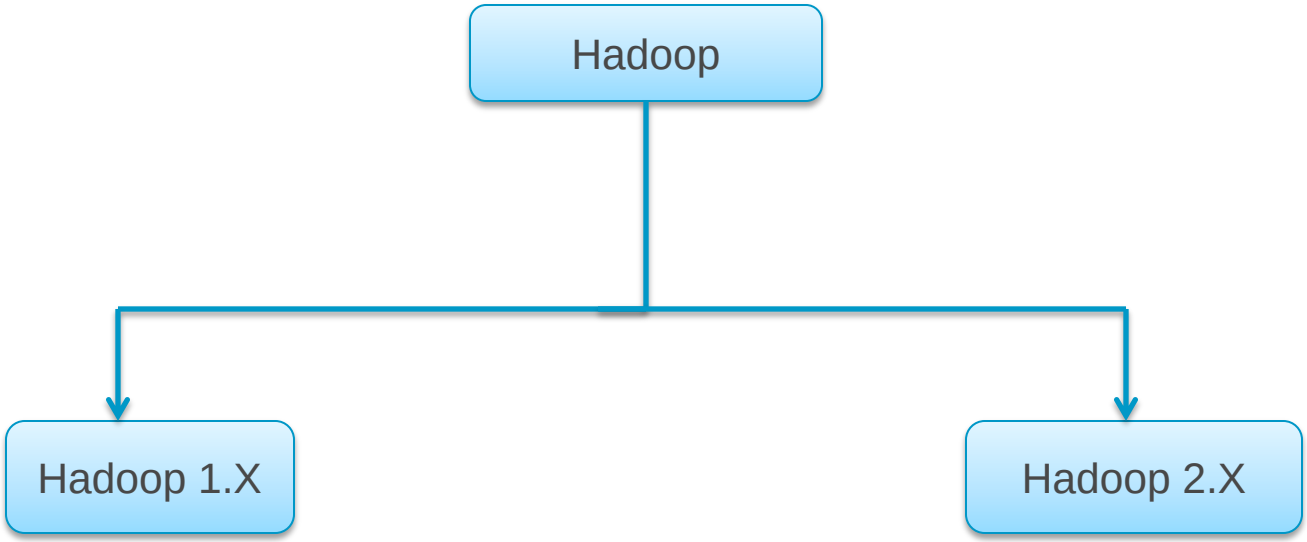
- **Scalable:** It can reliably store and process petabytes of data and can be scaled up anytime whenever required without any adverse impact of cluster.
- **Economical:** It distributes the data and processing across clusters of commonly available computers (in thousands).
- **Efficient:** By distributing the data, it can process it in parallel on the nodes where the data is located.
- **Reliable:** It automatically maintains multiple copies of data and automatically redeploys computing tasks based on failures.

Relational DB vs. Hadoop

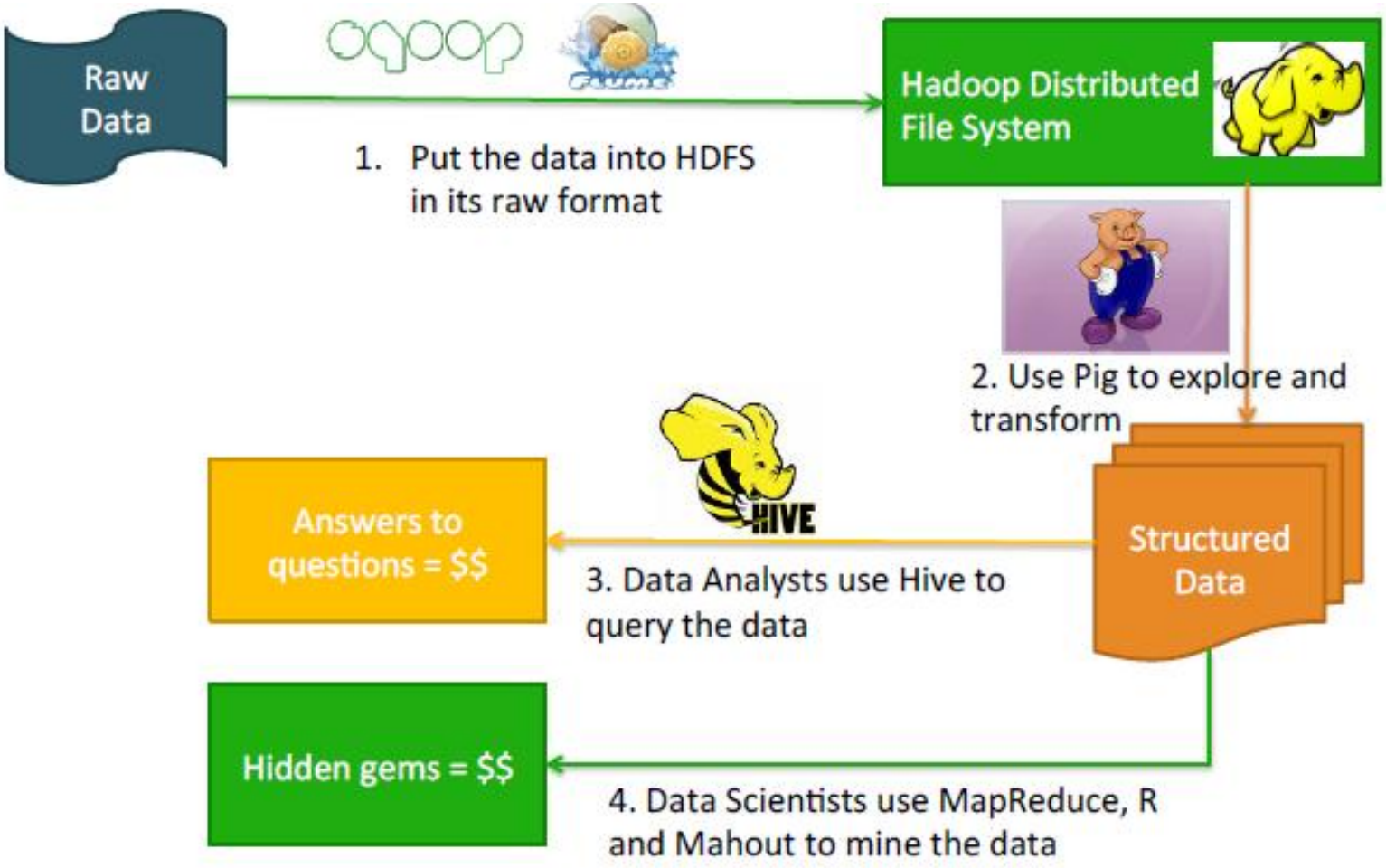
- Expensive dedicated HW
- Built for **performance**
- Designed for **high volumes** (eg 10s of TB)
- High availability
- Initially developed using Relational Data bases
- Supports only **modelled and structured data**
- Business As Usual ways to design, build and deliver
- Very mature solutions (skills, SW, HW, administration)
- Teradata, Oracle Exadata, IBM Netezza, ...

- Uses commodity PCs
- Built for **extreme scalability**
- Designed for **extreme volumes** (10s of PB and more)
- Very high availability (clouds like Amazon distributed all around the world)
- Initially developed by Google for storing Petabytes of web pages for ranking
- Not yet fully mature
- Hadoop = Data is distributed over many machines
- MapReduce = Computing is distributed and executed where data is (grid solution)
- Works on **Write Once read many times** approach

Generations of Hadoop



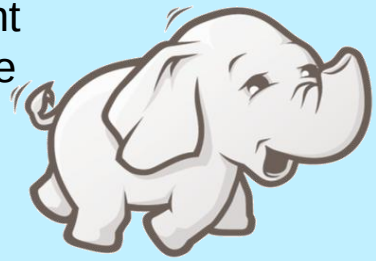
Usage of Hadoop and its eco-systems



Components of Hadoop

Hadoop is a platform for data storage and processing that is…

- ✓ Scalable
- ✓ Fault tolerant
- ✓ Open source



CORE HADOOP COMPONENTS

Hadoop Distributed File System (HDFS)

File Sharing & Data Protection Across Physical Servers

+

MapReduce

Distributed Computing Across Physical Servers

Components of Hadoop(continued)

- Two core components of Hadoop :
  1. Hadoop Distributed File System (HDFS)
  2. Map Reduce

- HDFS :
  - Is the storage part of Hadoop framework
  - Data is stored in files under directories
  - Files are always divided into blocks

- Map Reduce :
  - Is the programming/processing model of Hadoop framework
  - Is responsible for manipulation or processing on data
  - Preferebly written as java programs

HDFS Blocks & Replication

- Data files are divided into blocks and distributed across multiple nodes in the cluster
  - Each block is typically multiple of 64 MB in size in Hadoop 1.X and 128 MB in Hadoop 2.X

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

Hadoop 2.x

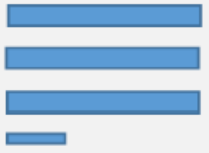
200mb – abc.txt

128mb – Block 1
72mb – Block 2

Hadoop 1.x

200mb – emp.dat

64mb – Block1
64mb – Block2
64mb – Block3
8mb – Block4



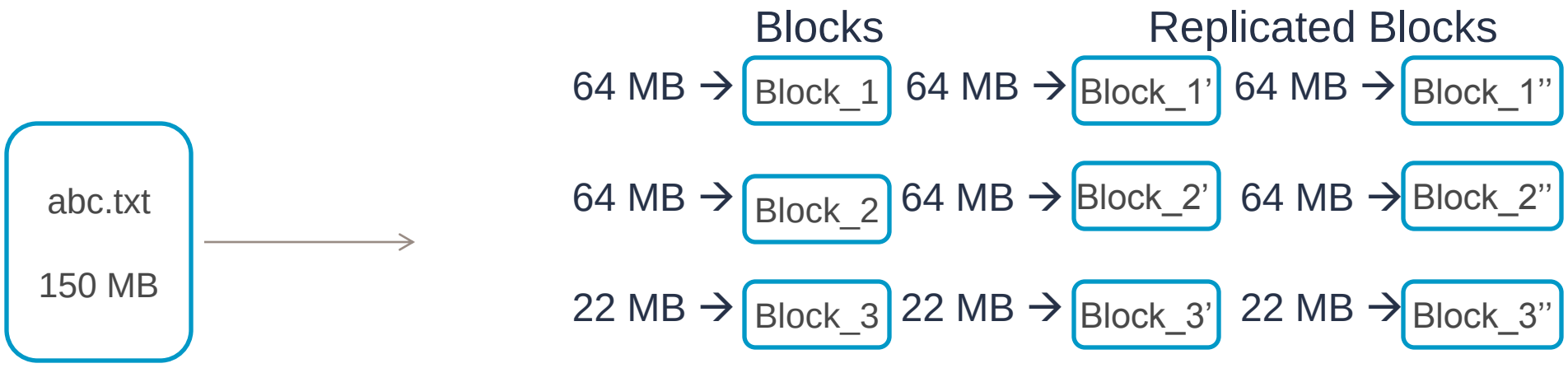
→ Why block size is large?

» 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.

HDFS Blocks & Replication(continued)

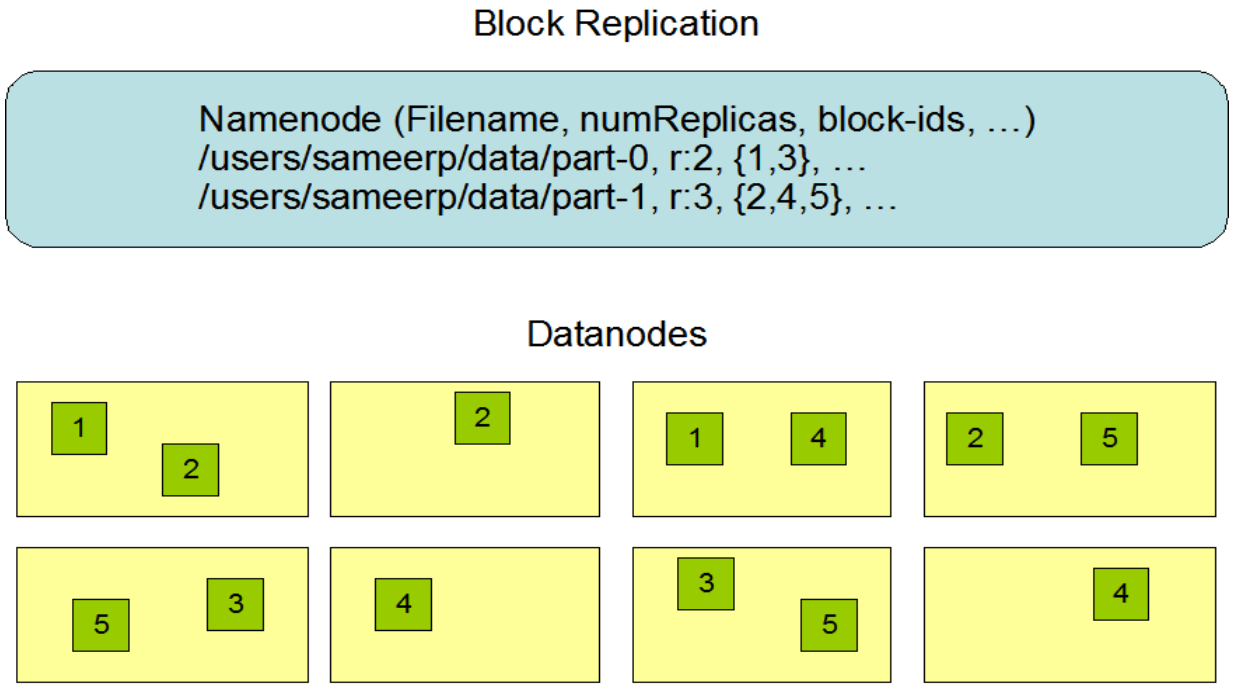
- Each block is replicated multiple times
  - Default replication factor is 3 (configurable)
  - Replicas are stored on different nodes
  - This ensures both reliability and availability



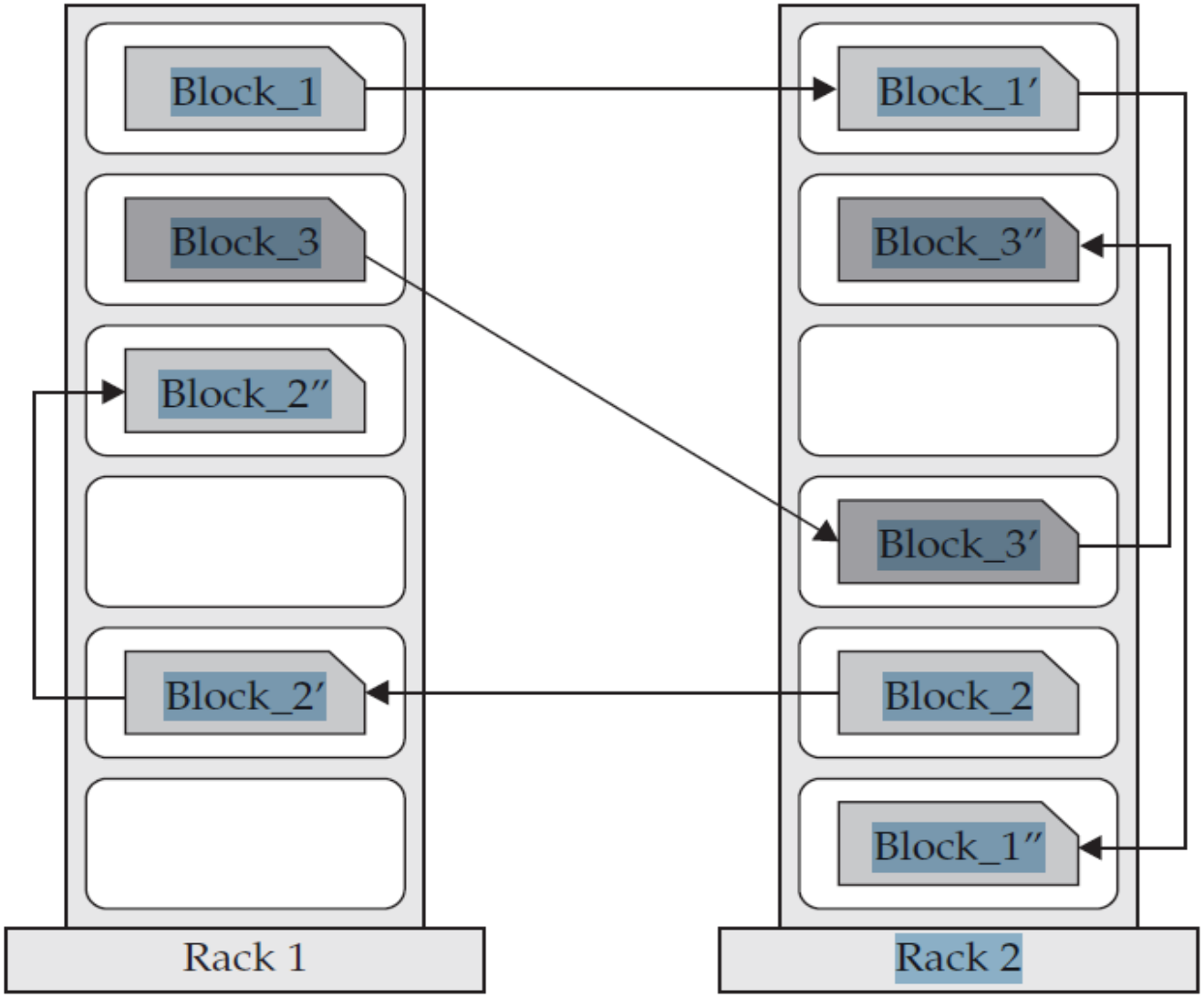
Considered Hadoop 1.X in above example.

HDFS Blocks & Replication(continued)

- Application can specify number of replicas for a file to be made.
- Default is 3
- Each file is stored as sequence of block of same size except last size.
- These block are replicated over the cluster for fault tolerance
- NameNode is responsible for replication.



Rack Awareness

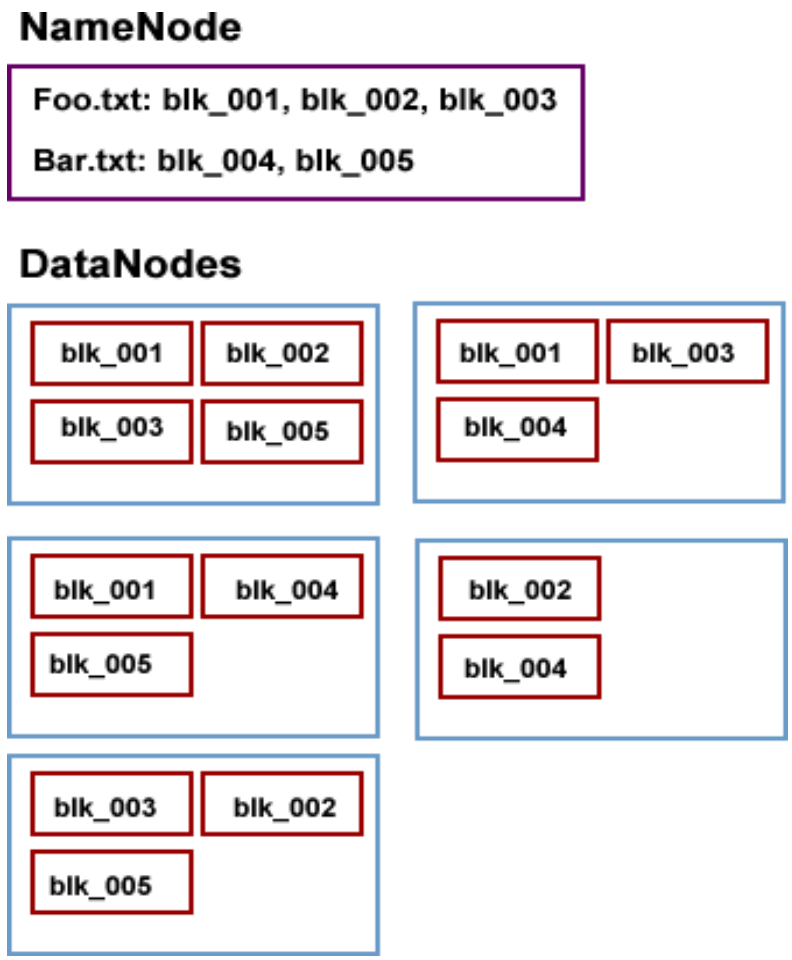


How Files Are Stored

- Files are split into blocks
- 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.
- Blocks are replicated across multiple machines, known as Data Nodes
  - Default replication is three-fold meaning that each block exists on three different machines
- A master node called the Name Node keeps track of which blocks make up a file, and where those blocks are located
  - Known as the metadata

How Files Are Stored: Example

- Name Node holds metadata for the two files (Foo.txt and Bar.txt)
- Data Nodes hold the actual blocks
  - Each block will be 64 MB or 128 MB in size
  - Each block is replicated three times on the cluster



Data correctness

- **Use Checksums to validate data**
  - Use CRC32
- **File Creation**
  - Client computes checksum per 512 byte
  - DataNode stores the checksum
- **File access**
  - Client retrieves the data and checksum from DataNode
  - If Validation fails, Client tries other replicas

Rebalancer

- **Goal: % of disk occupied on Datanodes should be similar**
  - Usually run when new Datanodes are added
  - Cluster is online when Rebalancer is active
  - Rebalancer is throttled to avoid network congestion
  - Command line tool

Fault Tolerance in Hadoop

- If a data node fails, the master(Name Node) will detect that failure and re-assign the task to a different node on the cluster
- Restarting a task does not require communication with nodes working on other portions of the data
- If a failed node recovers, it is automatically added back to the cluster 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’

Quiz

1. What is Hadoop?
   A. Database
   B. Framework
   C. Programming Language
   D. File

2. What is a Hadoop cluster?
   A. Group of machines connected in a network
   B. Group of machines sharing same software
   C. Group of connected machines running HDFS & Map Reduce
   D. Group of machines sharing same printer

Quiz-Answers

1. What is Hadoop?

   A. Database
   B. Framework
   C. Programming Language
   D. File

B: Framework

2. What is a Hadoop cluster?

   A. Group of machines connected in a network
   B. Group of machines sharing same software
   C. Group of connected machines running HDFS & Map Reduce
   D. Group of machines sharing same printer

C: Group of Machines Running HDFS & MR

Quiz(continued)

3. Default block size in Hadoop 2.X is :

A. 64 MB
B. 68 MB
C. 188 MB
D. 128 MB

4. Data Blocks are stored in ?

A. Name Node
B. Data Node
C. Database
D. A&B

Quiz-Answers(continued)

3. Default block size in Hadoop 2.X is :

A. 64 MB
B. 68 MB
C. 188 MB
D. 128 MB

D: 128 MB

4. Data Blocks are stored in ?

A. Name Node
B. Data Node
C. Database
D. A&B

B: Data Node

Quiz(continued)

5. How many number of blocks will be created for a file xyz.txt of size 200 MB considering Hadoop 1.x environment?

A. 4

B. 5

C. 2

D. 3

Quiz-Answers(continued)

5. How many number of blocks will be created for a file xyz.txt of size 200 MB considering Hadoop 1.x environment?

A. 4
B. 5
C. 2
D. 3

A: 4

HDFS Commands

- HDFS Commands are used to
  - access Hadoop Distributed File System (HDFS)
  - list down files and directories present in HDFS
  - create and remove directory in HDFS
  - push the data present in landing zone(local machine) to HDFS
  - copy the data back to landing zone(local machine) from HDFS

HDFS Commands(continued)

- Get a directory listing of the HDFS root directory

```
hadoop fs –ls /
```

- Create a directory called input under the user directory

```
hadoop fs –mkdir /user/input
```

- Copy file foo.txt from local disk to the input directory in HDFS

```
hadoop fs -copyFromLocal foo.txt /user/input/foo.txt
```


HDFS Commands(continued)

- Display the contents of the HDFS file **/user/input/*foo.txt***

```
hadoop fs –cat /user/input/foo.txt
```

- Copy the file foo.txt from HDFS to local

```
hadoop fs –copyToLocal /user/input/foo.txt  foo.txt
```

- Delete the file bar.txt in HDFS

```
hadoop fs –rm /user/input/bar.txt
```

HDFS Commands(continued)

- Upload a file in HDFS :

```
hadoop fs -put <localsrc> ... <hdfs_dest_path>
```

- Download a file from HDFS :

```
hadoop fs -get <hdfs_src> …<localdst>
```

- Display last few lines of a file

```
hadoop fs -tail <path[filename]>
```


HDFS Commands(continued)

- Delete the input directory in HDFS

```
hadoop fs –rm [–r ] /user/input/
```

→ -r used for recursive, deletes directory even if some file is present under that.

For more commands please refer:

http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/FileSystemShell.html

Hadoop Daemons

- Hadoop Works on Master-Slave pattern

- There are 5 daemons in Hadoop 1.x

1. Name Node → Master Node
2. Secondary Name Node → Master Node
3. Job Tracker → Master Node
4. Task Tracker → Slave Node
5. Data Node → Slave Node

Hadoop Daemons(continued)

- **Name Node** stores metadata of cluster
  - Block details
  - Replication details
  - Load Balancing
  - High configuration machine, 1 Name Node per cluster

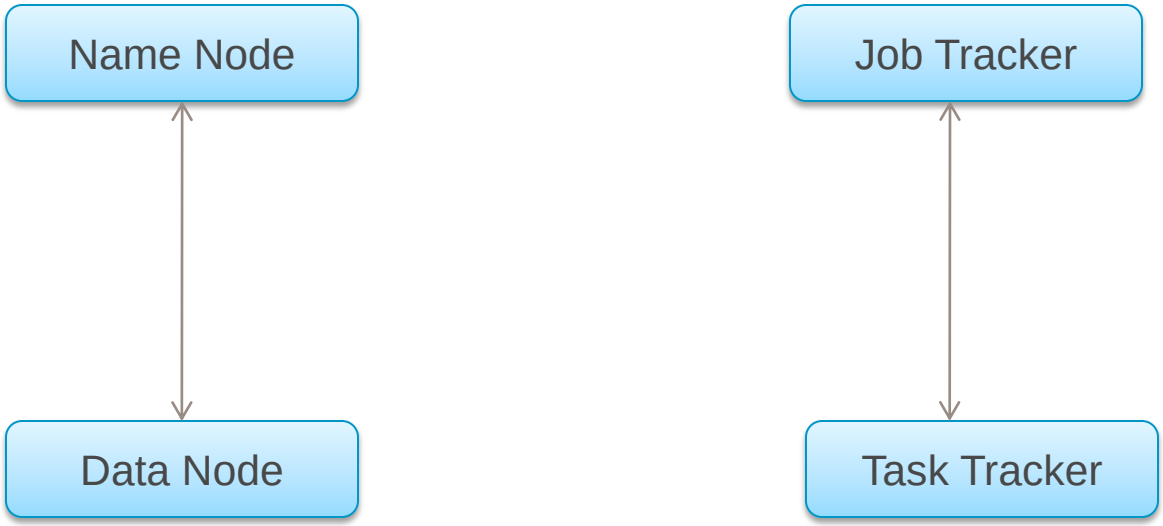
- **Secondary Name Node** is the backup of Name Node
  - Takes backup of Name Node in a regular interval
  - Not called as complete backup of Name Node

- **Job Tracker** is the daemon where hadoop jobs are submitted

- **Task Tracker** accomplish the task which are submitted to job tracker

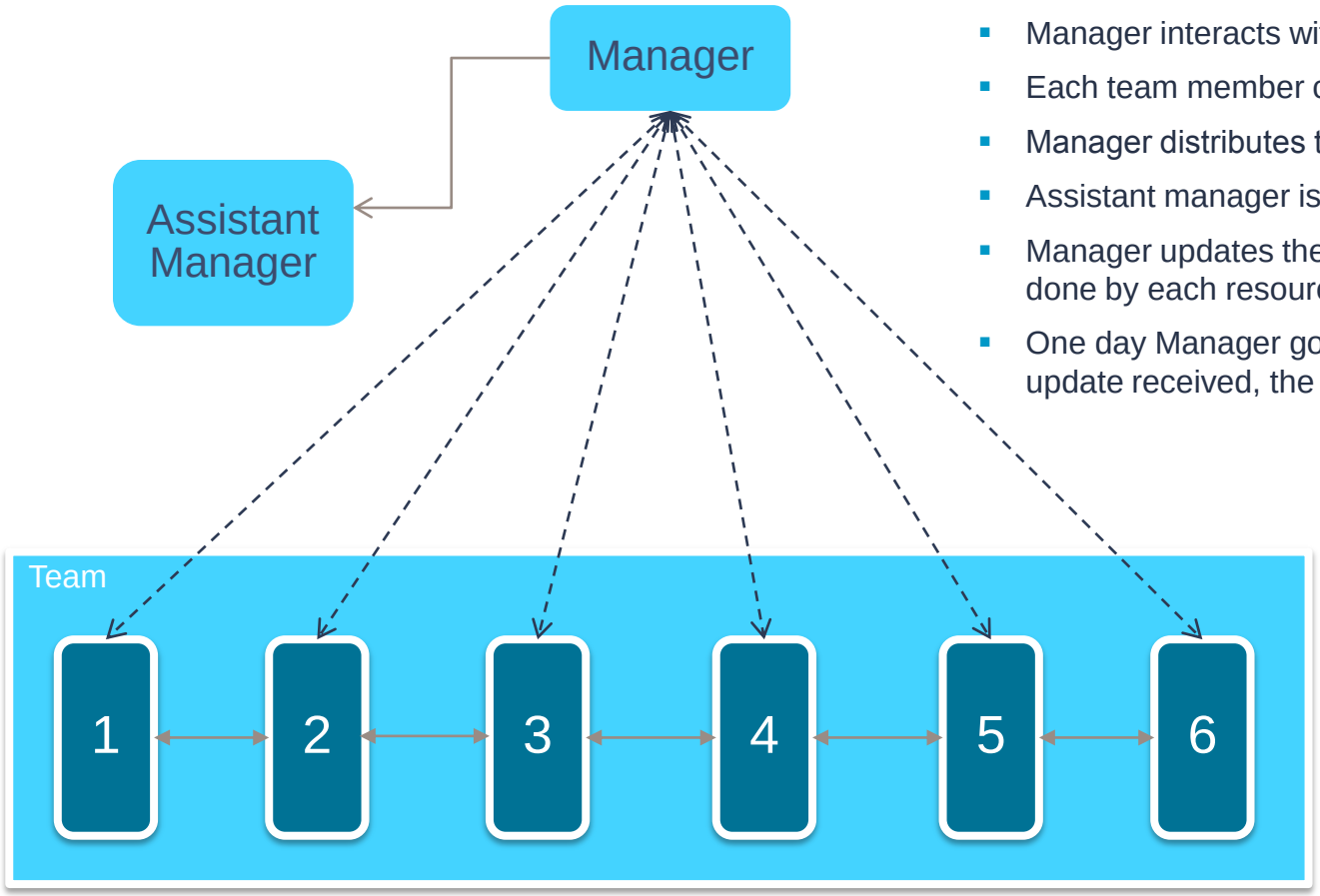
- **Data Node** is the daemon where actual data in the form of blocks reside

Hadoop Daemons(continued)



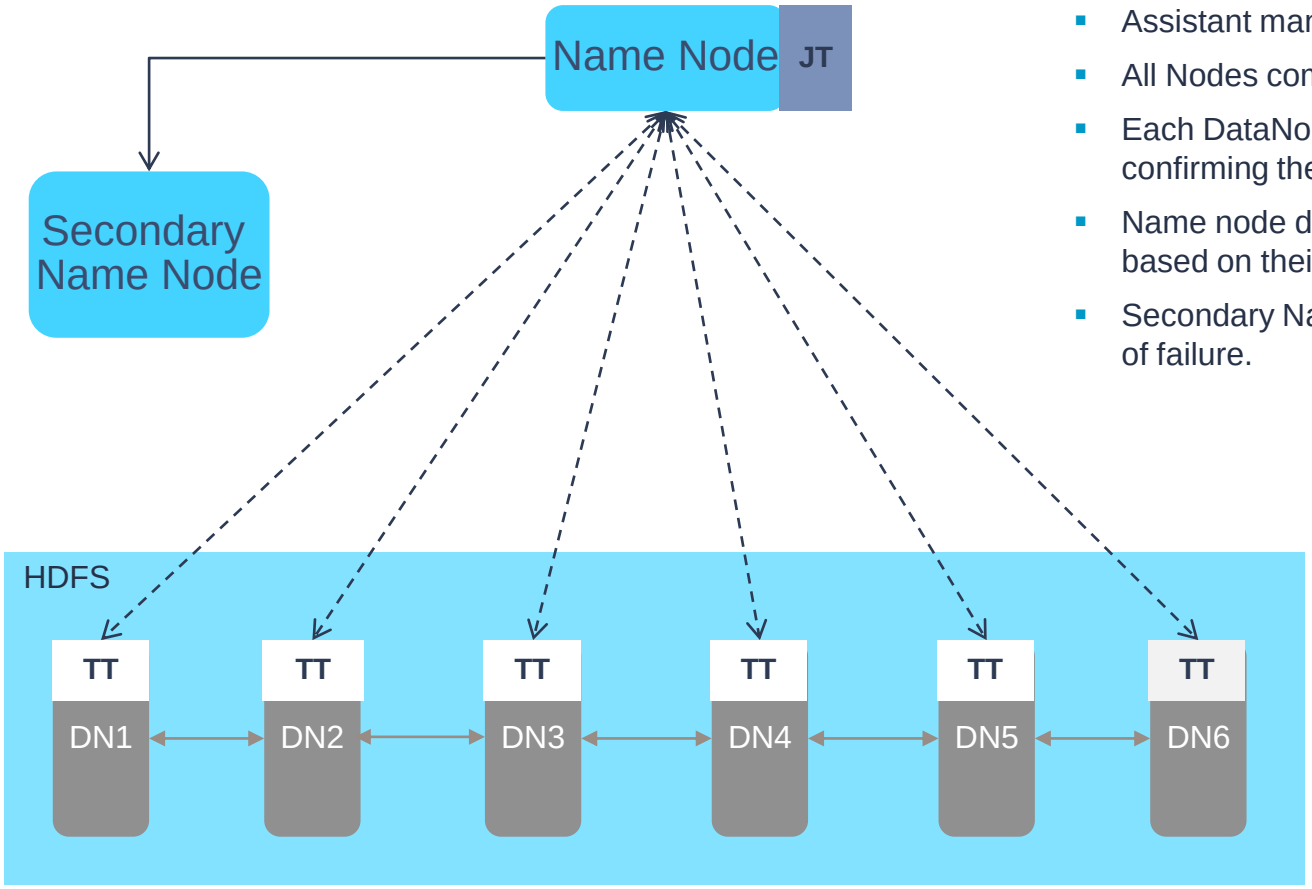
- Name Node coordinates with Data Nodes
  - In turn, all Data Nodes send heart beats to Name Node periodically
- Job Tracker coordinates with Task Tracker

Hadoop Daemons– Corporate Analogy



- Manager is appointed who will govern the project.
- Manager recruits 6 Members to Join the project team.
- They all communicate with each other.
- Manager interacts with each of the 6 Team members.
- Each team member confirms his presence to the manger everyday
- Manager distributes the work volume based on Team's utilization.
- Assistant manager is provisioned as a backup manager.
- Manager updates the Assistant manager every day about activities done by each resource.
- One day Manager goes on unplanned leave and based on the last update received, the asst. manager takes over the activities.

Hadoop Daemons(continued)



- Manager : Name node
- 6 Team Members : 6 data Nodes.
- Assistant manager : Secondary Name Node
- All Nodes communicate via SSH password less protocol.
- Each DataNode sends heartbeat signals to the Name node confirming they are live.
- Name node distributes the data Blocks across Datanodes based on their memory utilization and alive status.
- Secondary NameNode acts as a failover to avoid single point of failure.

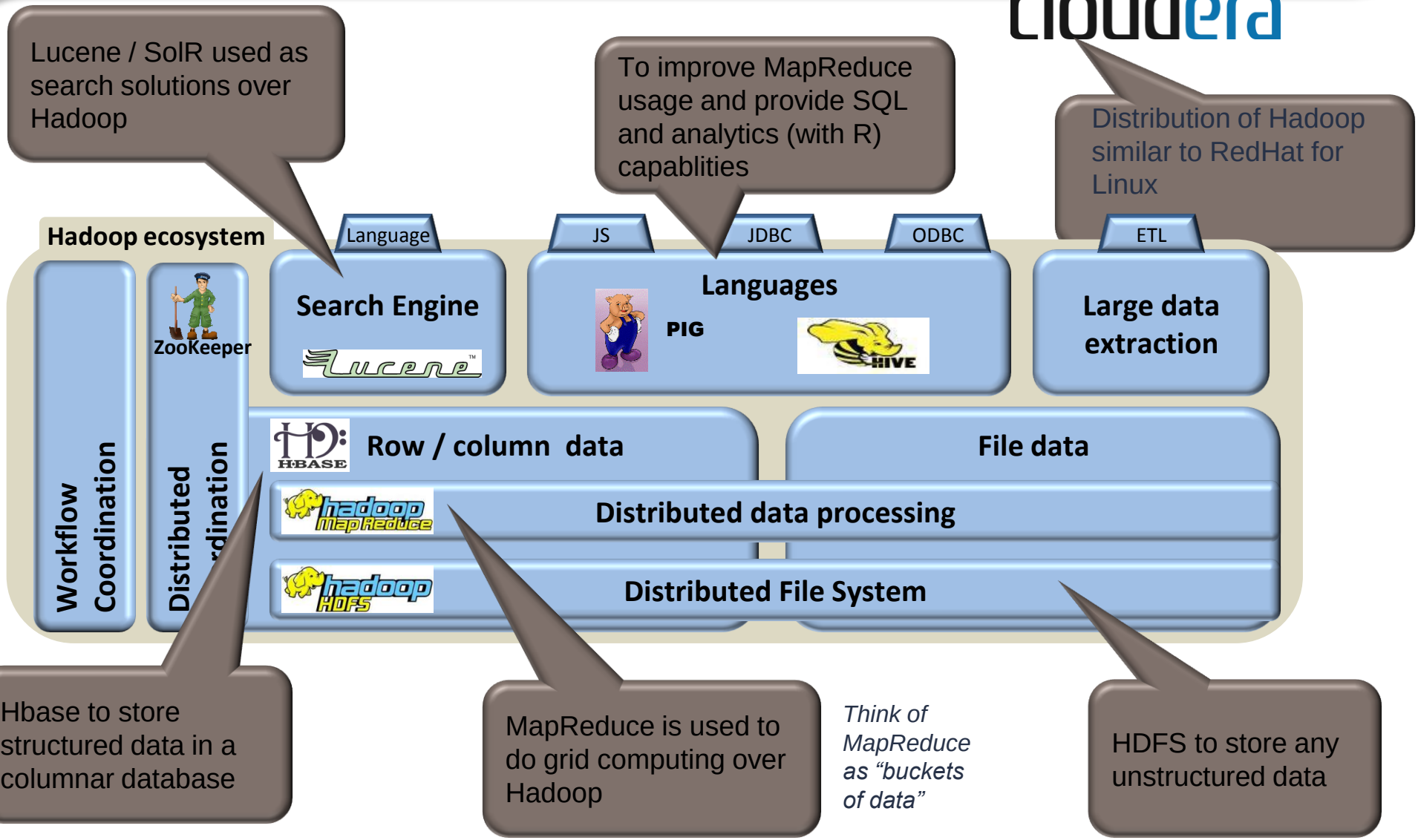
1. When a MR program is submitted it goes to the Job Tracker(JT)
2. JT talks with NameNode to understand which Data Nodes have the required data blocks of the source files.
3. Once identified, JT talks to the TTs on those Nodes and use them to perform the assigned Tasks.
4. On task completion, JT updates the NameNode on the new Block location where the output is stored.

Products in Hadoop Ecosystem

- MapReduce job (with multiple map and reduce functions) is challenging, and the development framework provides tools to enable increased programmer productivity.
- The framework consists of two main architectural components:
  - **Pig - ** A scripting platform for processing and analyzing large data sets
  - **Hive -** to query, summarize, explore and analyze that data in Hadoop

- **Other products**
  - **Hbase** – A non-relational (NoSQL) database, Columnar Data base
  - **Sqoop** – Efficiently transfers bulk data between Hadoop and structured datastores
  - **Zookeeper** - To coordinate distributed processes
  - **Oozie** - to schedule Apache Hadoop jobs
  - **Flume -** Flume lets Hadoop users ingest high-volume streaming data into HDFS for storage
  - **Kafka** - A fast, scalable, fault-tolerant Pub/Sub messaging system

Hadoop Ecosystem



Quiz

1. Which of the below HDFS command is used to create a directory in hdfs

   A. hadoop fs -dir /user/abc
   B. hadoop fs -crdir /user/abc
   C. hadoop fs -mkdir /user/abc
   D. Hadoop fs -mdir /user/abc

2. Which of the below HDFS command is used to copy file from local to HDFS?

   A. hadoop fs -copy 1.txt /user/abc/1.txt
   B. hadoop fs -copyfromlocal 1.txt /user/abc/1.txt
   C. hadoop fs -copyFromLocal 1.txt /user/abc/1.txt
   D. Hadoop fs -CopyToLocal 1.txt /user/abc/1.txt

Quiz-Answers(continued)

1. Which of the below HDFS command is used to create a directory in hdfs

A. hadoop fs -dir /user/abc
B. hadoop fs -crdir /user/abc
C. hadoop fs -mkdir /user/abc
D. Hadoop fs -mdir /user/abc

C: mkdir

2. Which of the below HDFS command is used to copy file from local to HDFS?

A. hadoop fs -copy 1.txt /user/abc/1.txt
B. hadoop fs -copyfromlocal 1.txt /user/abc/1.txt
C. hadoop fs -copyFromLocal 1.txt /user/abc/1.txt
D. Hadoop fs -CopyToLocal 1.txt /user/abc/1.txt

C: copyFromLocal

Quiz

3. Which of the below HDFS command is incorrect?

A. hadoop fs -copytolocal /user/abc/1.txt 1.txt
B. hadoop fs -mkdir /user/abc
C. hadoop fs -ls /user/abc/
D. Hadoop fs -cat /user/abc.txt

4. Name Node stores _____ of cluster :

A. Actual data
B. Blocks
C. Metadata
D. Replicated blocks

Quiz-Answers(continued)

3. Which of the below HDFS command is incorrect?

A. hadoop fs -copytolocal /user/abc/1.txt 1.txt
B. hadoop fs -mkdir /user/abc
C. hadoop fs -ls /user/abc/
D. Hadoop fs -cat /user/abc.txt

A:
Reason: copyToLocal Command is case sensitive

4. Name Node stores _____ of cluster :

A. Actual data
B. Blocks
C. Metadata
D. Replicated blocks

C: Metadata

Quiz

5. Secondary Name Node is complete Backup of Name Node?

A. True

B. False

6. Which of the following are Master Nodes?

A. Name Node and Data Node

B. Name Node and Job Tracker

C. Secondary Name Node

D. Both A & C

E. Both B & C

Quiz-Answers(continued)

5. Secondary Name Node is complete Backup of Name Node?

A. True

B. False

B: False
It takes backup in regular intervals only.

6. Which of the following are Master Nodes?

A. Name Node and Data Node

B. Name Node and Job Tracker

C. Secondary Name Node

D. Both A & C

E. Both B & C

E: Both B & C

Final Quiz

1. Hadoop is a framework that works with a variety of related tools. Common cohorts include:
   A. MapReduce, Hive and HBase
   B. MapReduce, MySQL and Google Apps
   C. MapReduce, Hummer and Iguana
   D. MapReduce, Heron and Trumpet

2. What technology might you use to stream Twitter feeds into Hadoop?
   A. Sqoop
   B. Flume
   C. Pig
   D. Hive

Final Quiz-Answers

1. Hadoop is a framework that works with a variety of related tools. Common cohorts include:

   A. MapReduce, Hive and HBase
   B. MapReduce, MySQL and Google Apps
   C. MapReduce, Hummer and Iguana
   D. MapReduce, Heron and Trumpet

A: MR ,Hive & HBase

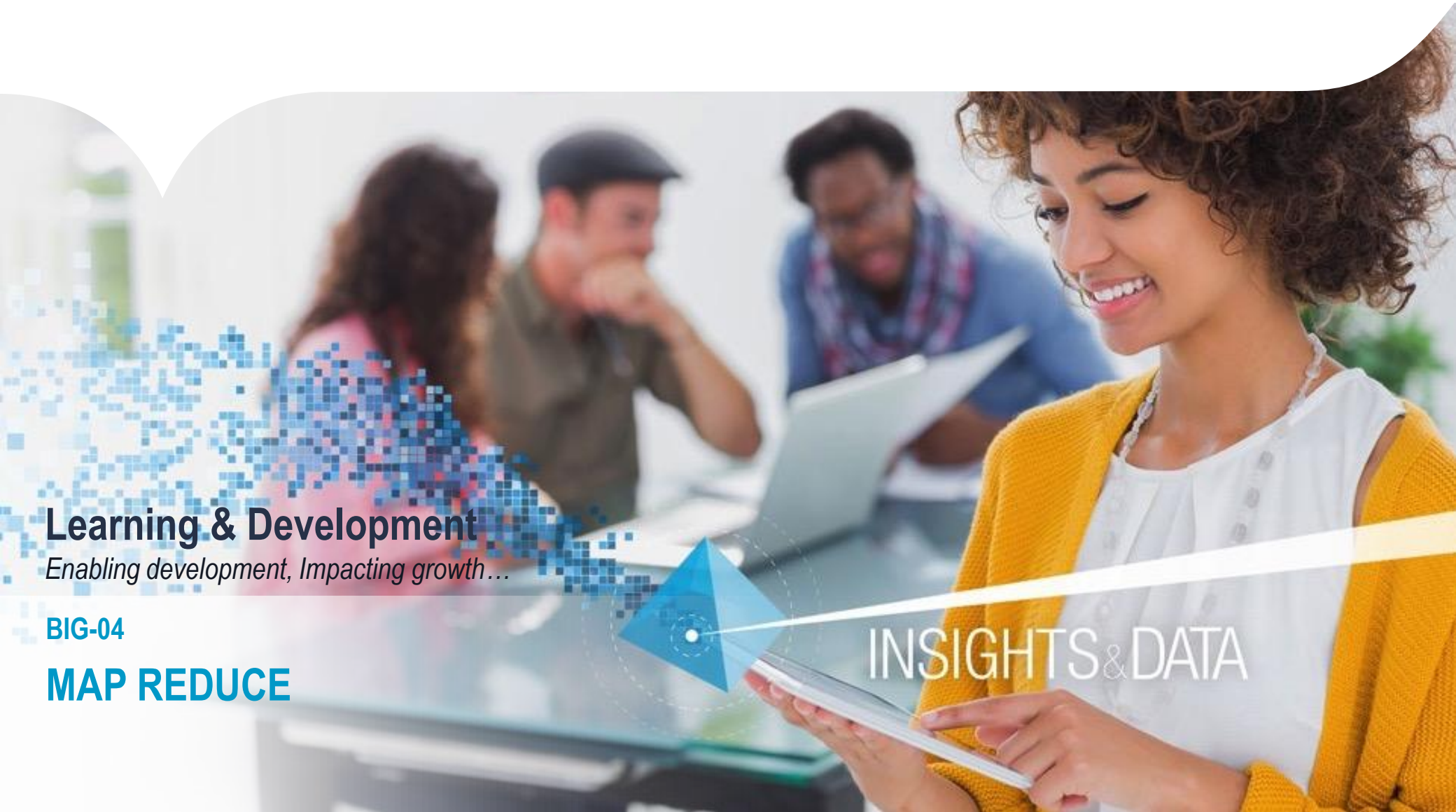
2. What technology might you use to stream Twitter feeds into Hadoop?

   A. Sqoop
   B. Flume
   C. Pig
   D. Hive

B: Flume

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

People matter, results count.



Learning & Development

Enabling development, Impacting growth…

BIG-04

MAP REDUCE

What is Map Reduce

- Programming model for data processing for clusters of commodity machines

- Pioneered by Google
  - Processes 20 PB of data per day
- Popularized by open-source Hadoop project
  - Used by Yahoo, Facebook, Amazon etc
- Simple Implementation model
  - Implementation can be done in Java, ruby, python, C++ etc.
  - No need to worry about shipment, rack, distribution, JVM etc.

Map Reduce(continued)

- "Map" step:
  - Is just a data preparation step
  - Program split into pieces
  - Worker(Slave) nodes process individual pieces in parallel (under global control of the Job Tracker node)
  - Each worker node stores its result in its local file system where a reducer is able to access it

- "Reduce" step:
  - Data is aggregated (‘reduced” from the map steps) by worker nodes (under control of the Job Tracker)
  - Multiple reduce tasks can parallelize the aggregation
  - Aggregation logic is applied in this phase

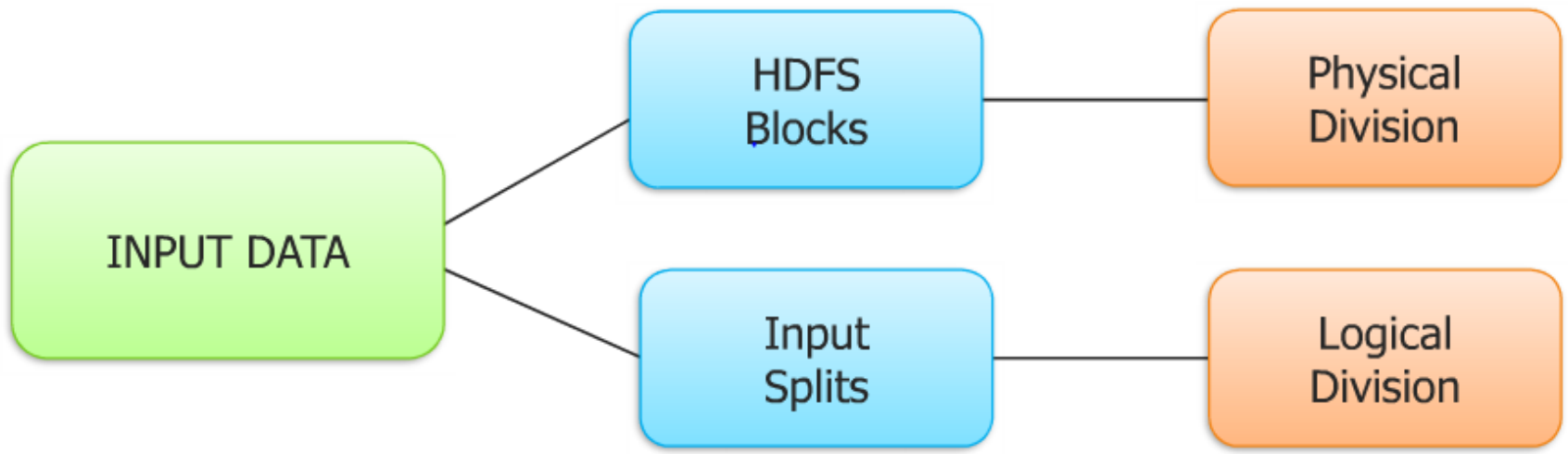
Map Reduce(continued)

- Input to Mapper (map phase) is always an input split.
- An input split is a logical division of data.
- Input and output of map reduce is always a Key-Value pair.
- Mapper is just a data preparation phase.
- Actual aggregation logic is applied in Reducer phase.
- Merge and sort is done by framework after map phase.

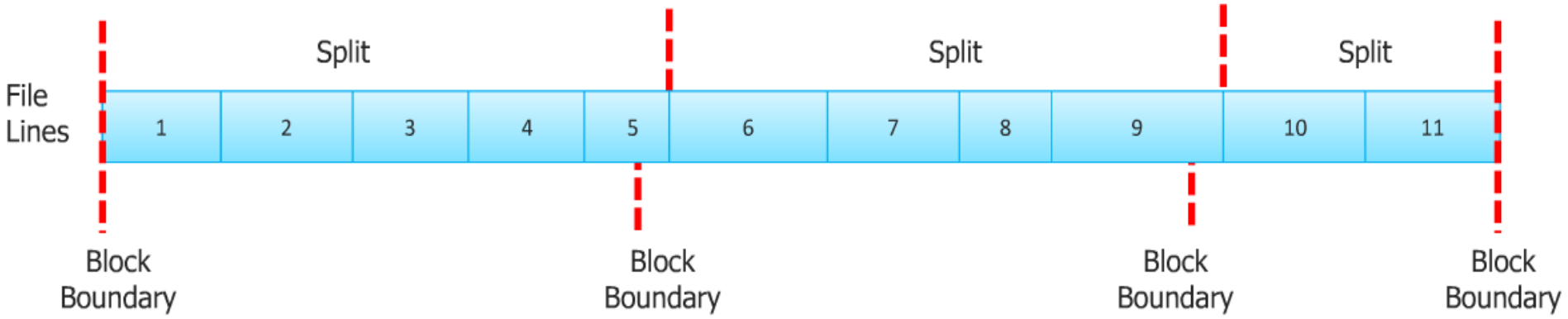
MapReduce: The JobTracker

- **Software deamons**
  - JobTracker
  - TaskTracker
- **JobTracker**
  - It controls MapReduce job
  - The JobTracker resides on a ‘master node’
  - Clients submit MapReduce jobs to the JobTracker
  - The JobTracker assigns Map and Reduce tasks to other nodes
    - on the cluster
  - These nodes each run a software daemon known as the TaskTracker
- ***TaskTracker***
  - The TaskTracker is responsible for actually instantiating the Map
    - or Reduce task and reporting progress back to the JobTracker

Input Splits



Relation between blocks and input splits



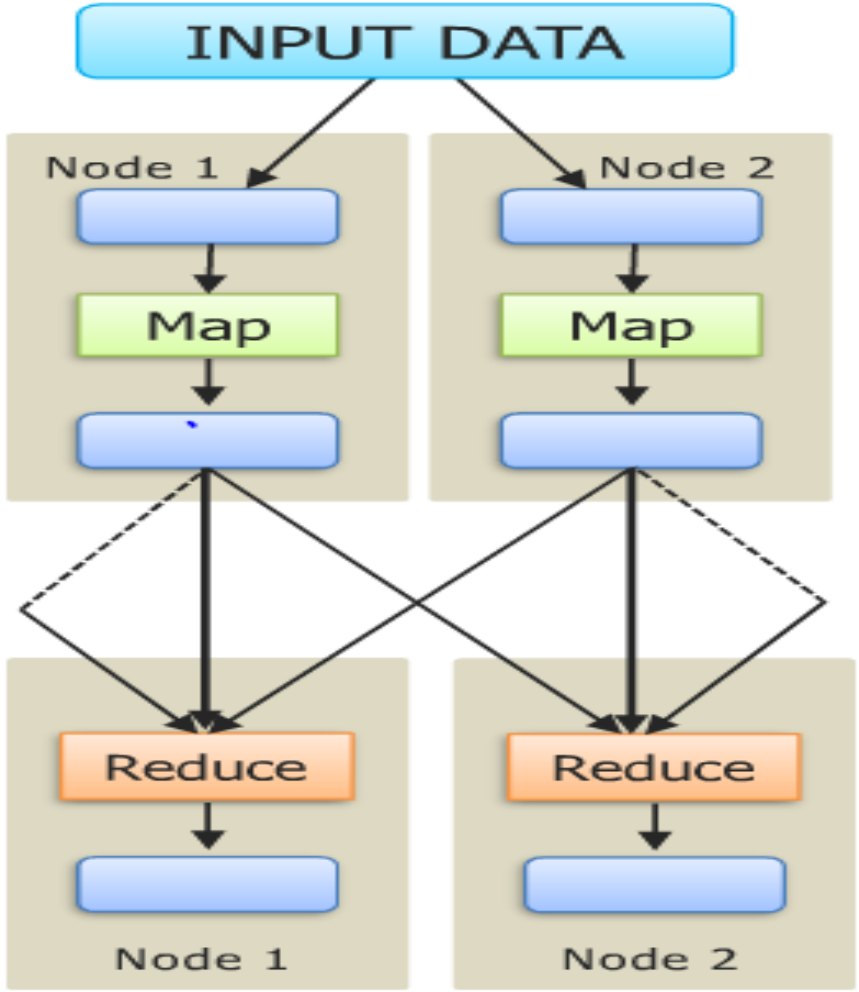
→ Logical records do not fit neatly into the HDFS blocks.

→ Logical records are lines that cross the boundary of the blocks.

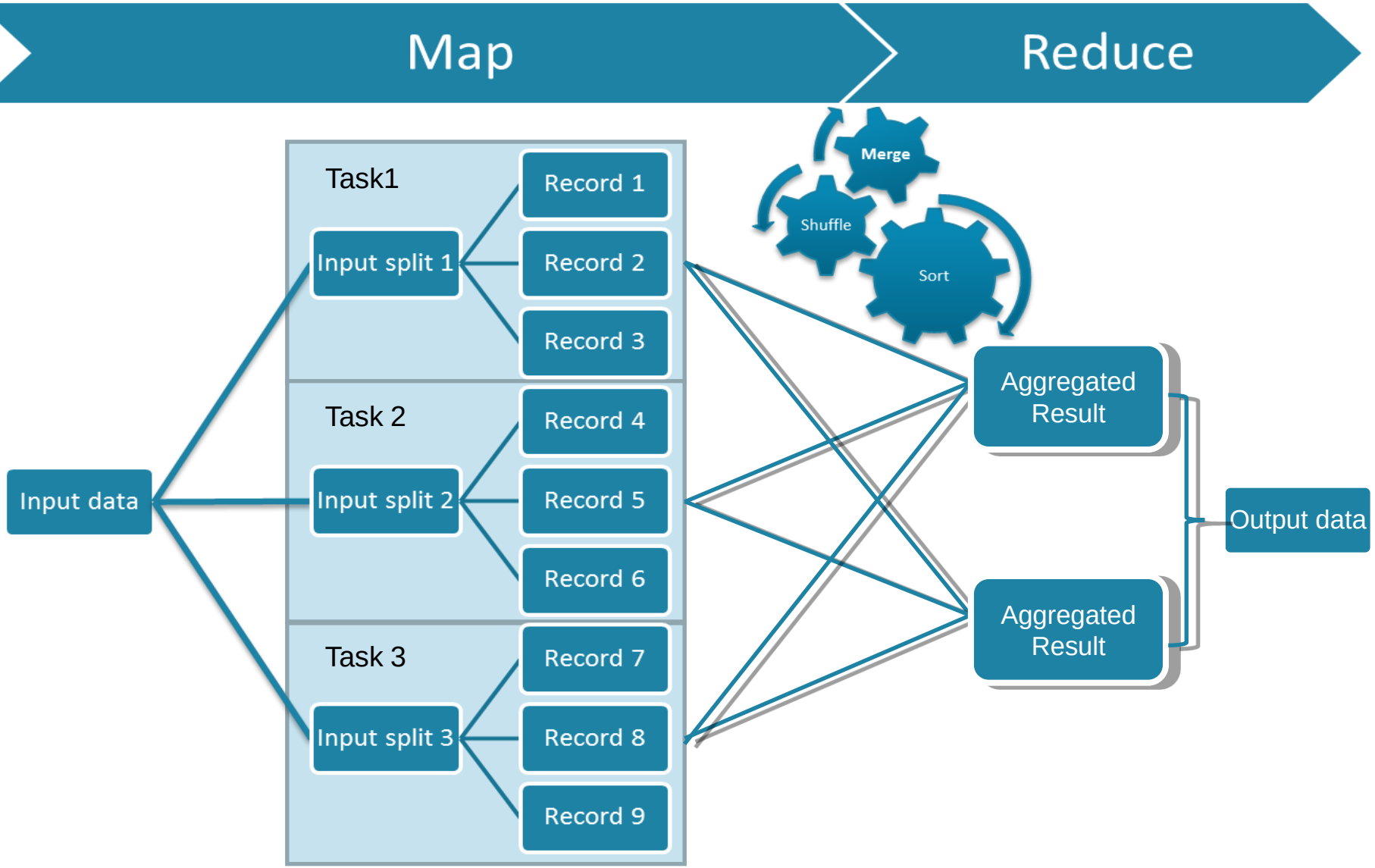
→ First split contains line 5 although it spans across blocks.

Map Reduce Job Submission

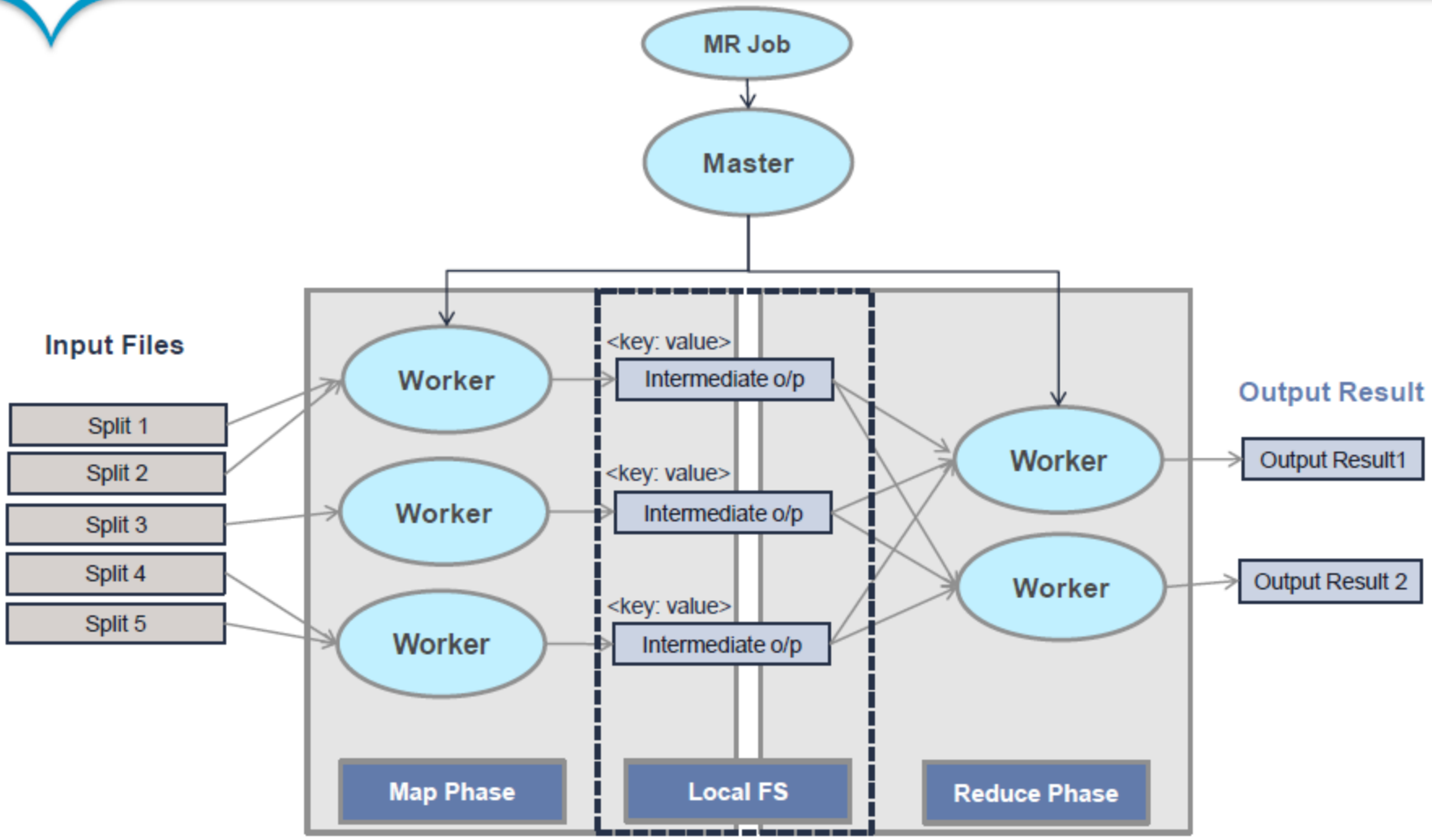
- Each Map task works on a split data
- Mapper outputs intermediate data
- Sort and merge is performed based on key
- Reducer output is stored in different nodes



Map Reduce(continued)

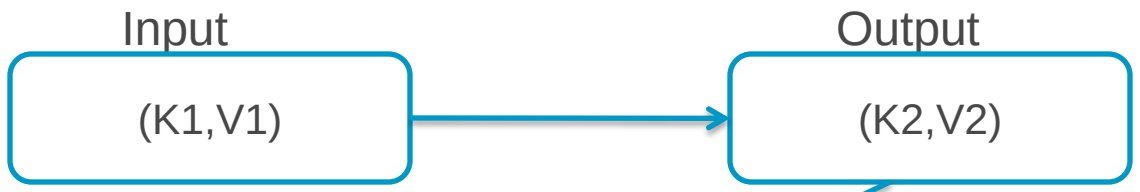


Map Reduce(continued)



Map Reduce Functions

- Map Function:



  - Reduce Function:

Map Reduce in Hadoop-1

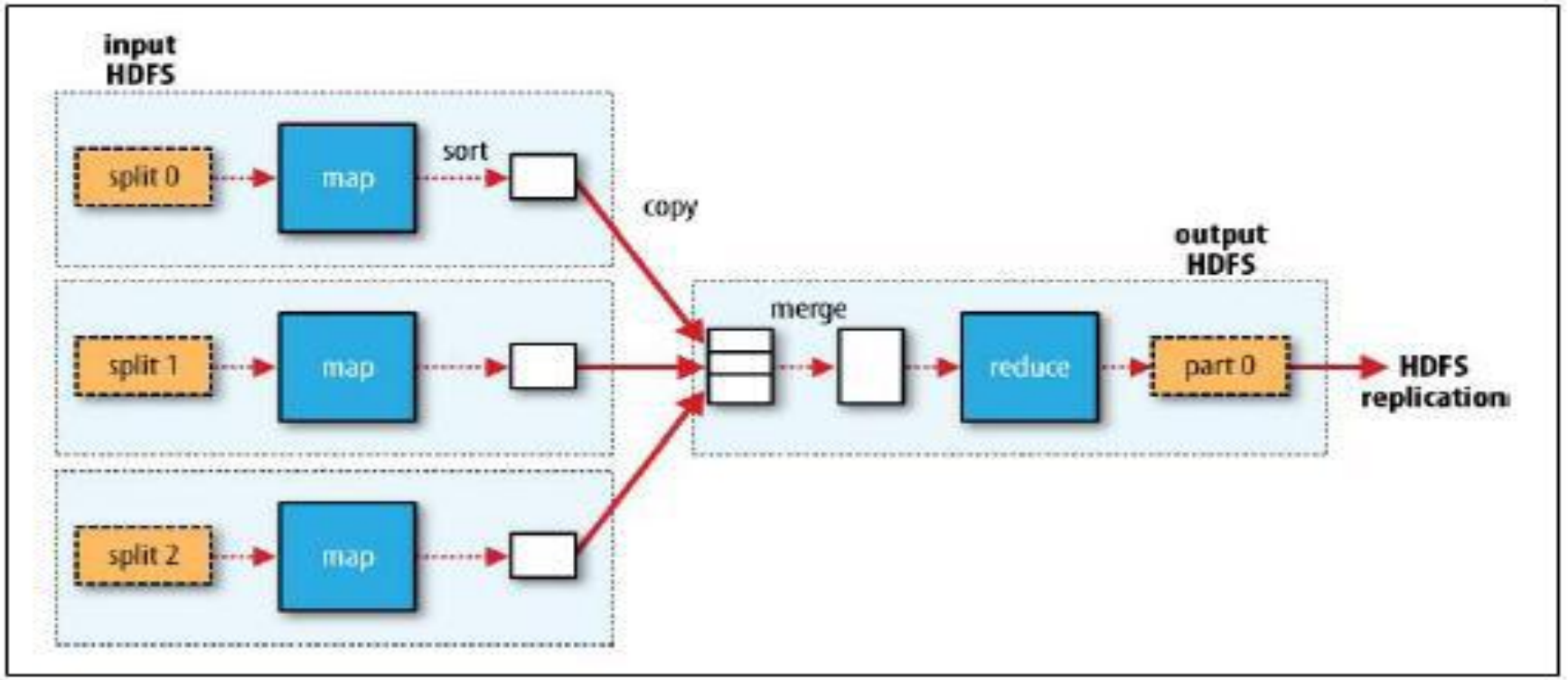
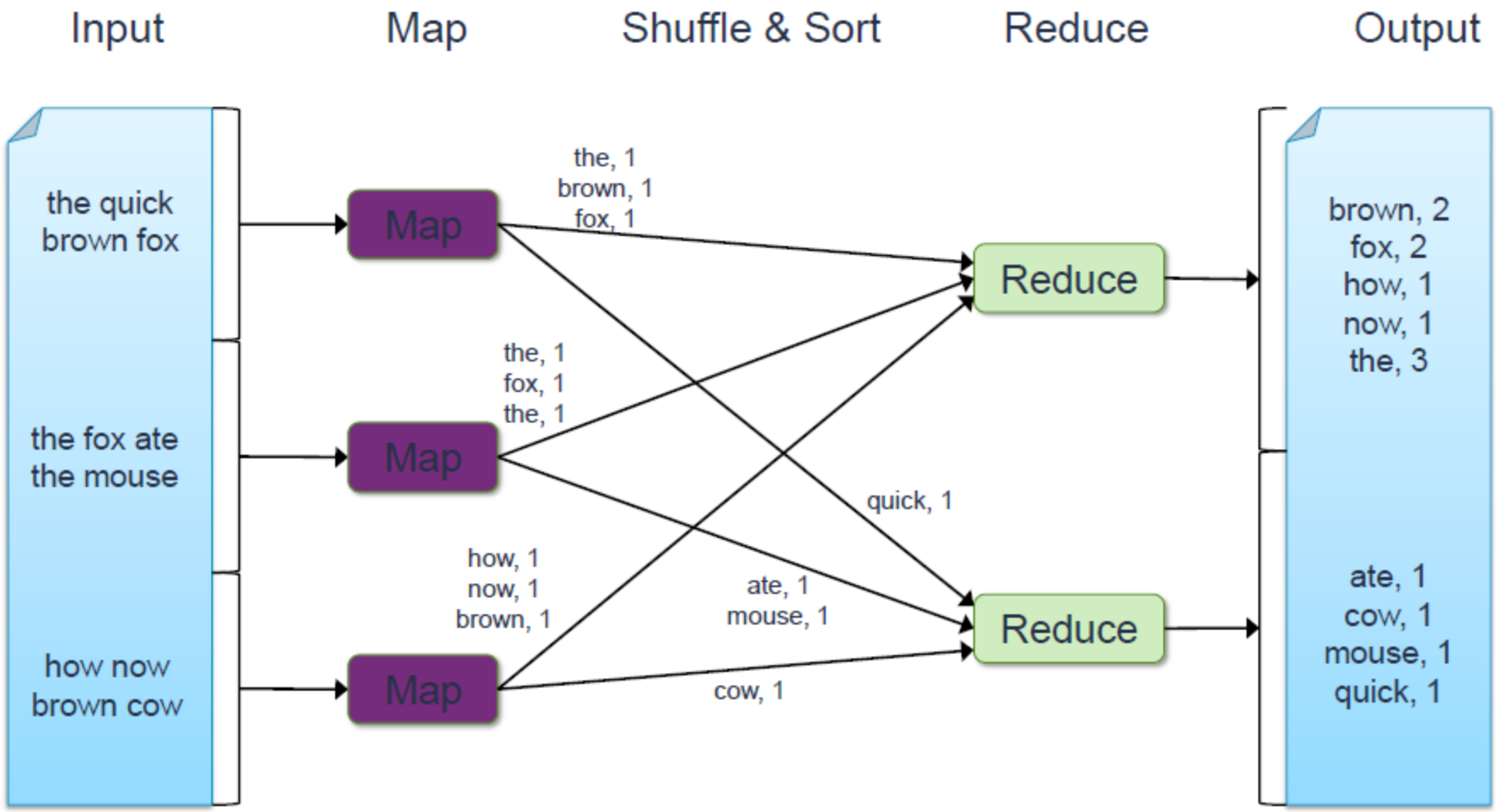


Figure 2-2. MapReduce data flow with a single reduce task

Word Count Execution



Example Mapper: Upper Case Mapper

- **Turn input into upper case (pseudo-code):**
  - *Let map(k, v) = emit(k.toUpper(), v.toUpper())*
  - *('foo', 'bar') -> ('FOO', 'BAR')*
  - *('foo', 'other') -> ('FOO', 'OTHER')*
  - *('baz', 'more data') -> ('BAZ', 'MORE DATA')*

Example Mapper: Explode Mapper

- **Output each input character separately (pseudo-code):**
  - *let map(k, v) =foreach char c in v:emit (k, c)*
  - *('foo', 'bar') -> ('foo', 'b'), ('foo', 'a'),*
  - *('foo', 'r')*
  - *('baz', 'other') -> ('baz', 'o'), ('baz', 't'),*
  - *('baz', 'h'), ('baz', 'e'),*
  - *('baz', 'r')*

Example Mapper: Filter Mapper

- **Only output key/value pairs where the input value is a prime**
- **number (pseudo-code):**
  - *let map(k, v) = if (isPrime(v))*
    - *then emit(k, v)*
    - *('foo', 7) -> ('foo', 7)*
    - *('baz', 10) -> nothing*

MapReduce: The Reducer

- **After the Map phase is over, all the 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
  - This is specified as part of the job configuration (see later)
  - 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

Example Reducer: Sum Reducer

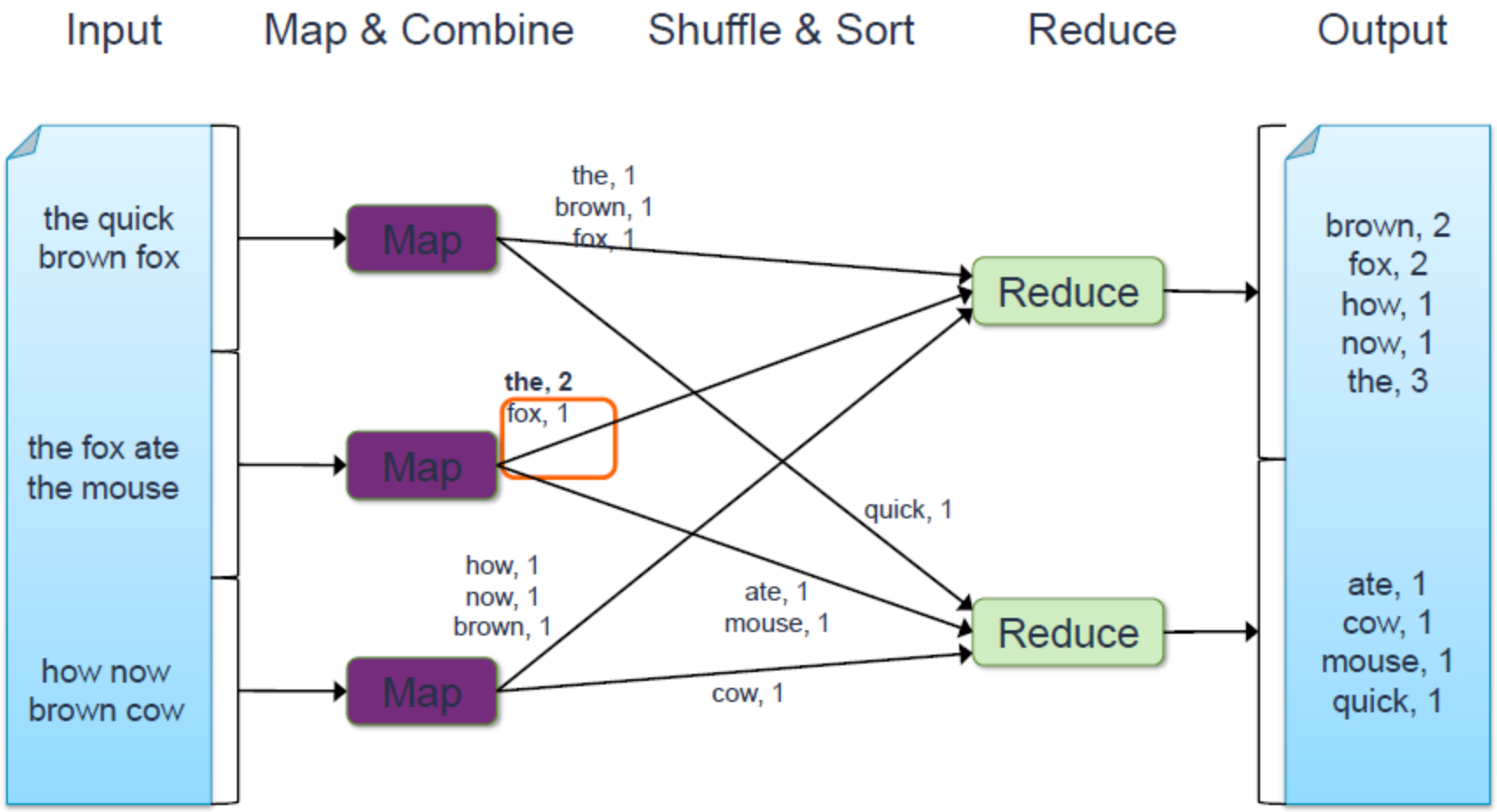
- **Add up all the values associated with each intermediate key**
- **(pseudo-code):**
  - *let reduce(k, vals) = sum = 0*
    - *foreach int i in vals:*
    - *sum += i*
    - *emit(k, sum)*

- *('bar', [9, 3, -17, 44]) -> ('bar', 39)*
- *('foo', [123, 100, 77]) -> ('foo', 300)*

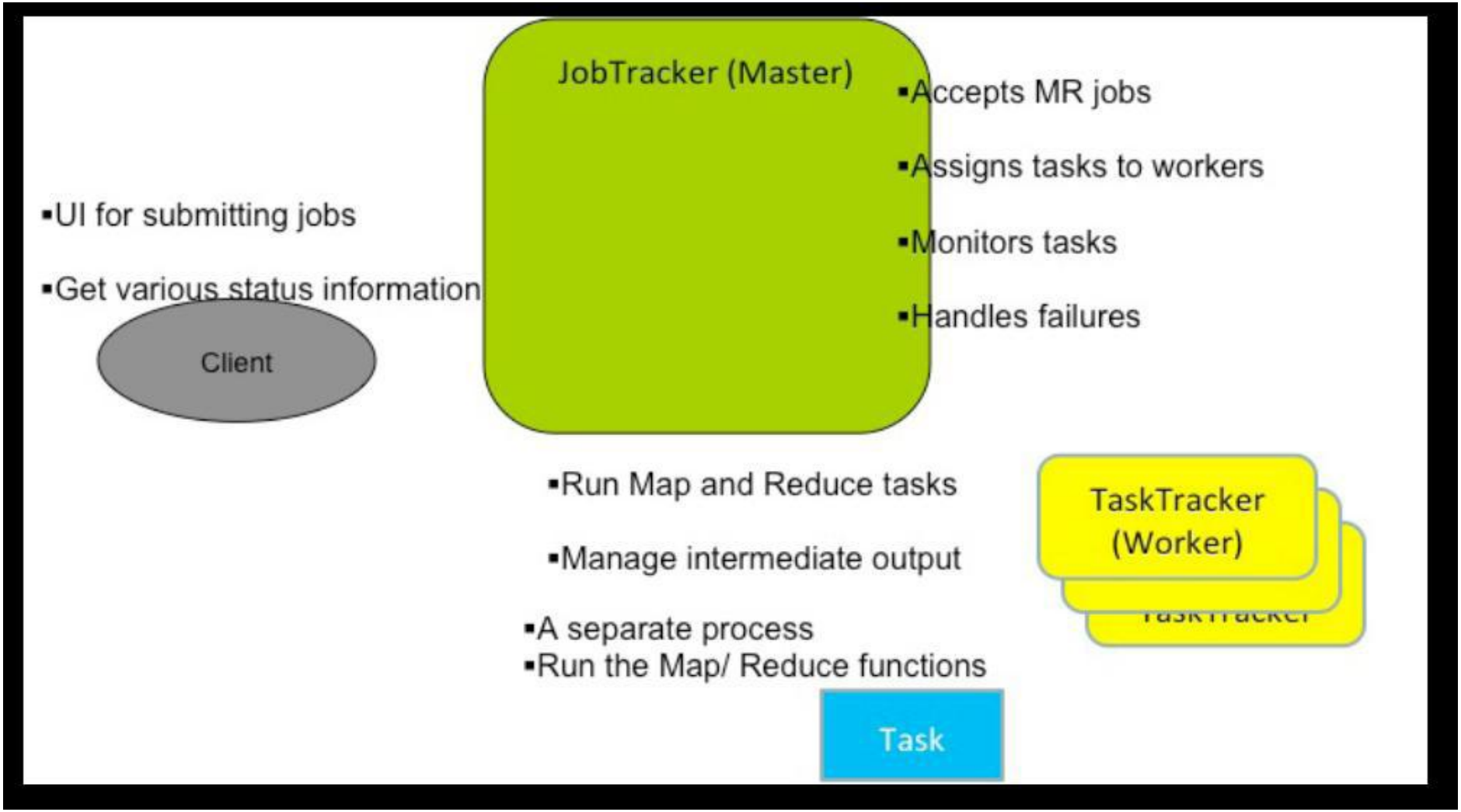
An Optimization: The Combiner

- A combiner is a local aggregation function for repeated keys produced by same map
- For associative ops. like sum, count, max
- Decreases size of intermediate data

Word Count with Combiner



Components in a Hadoop MR Workflow



Quiz

1. Input to Map Reduce is in the form of:
   A. List of values
   B. Array of values
   C. Key-Value pair
   D. None of the above

2. Input splits are:
   A. Logical division of data
   B. Physical division of data
   C. Both A & B
   D. None of the above

Quiz-Answers

1. Input to Map Reduce is in the form of:
   - A. List of values
   - B. Array of values
   - C. Key-Value pair
   - D. None of the above

C: Key-Value Pair

2. Input splits are:
   - A. Logical division of data
   - B. Physical division of data
   - C. Both A & B
   - D. None of the above

A: Logical division

Quiz

3. Combiners are also called:

A. Mini mappers

B. Mini reducers

C. Map phase

D. Key-Value pairs

4. True or False? MapReduce can best be described as a programming model used to develop Hadoop-based applications that can process massive amounts of unstructured data.

A. True

B. False

Quiz-Answers

3. Combiners are also called:

A. Mini mappers
B. Mini reducers
C. Map phase
D. Key-Value pairs

B: Mini Reducers

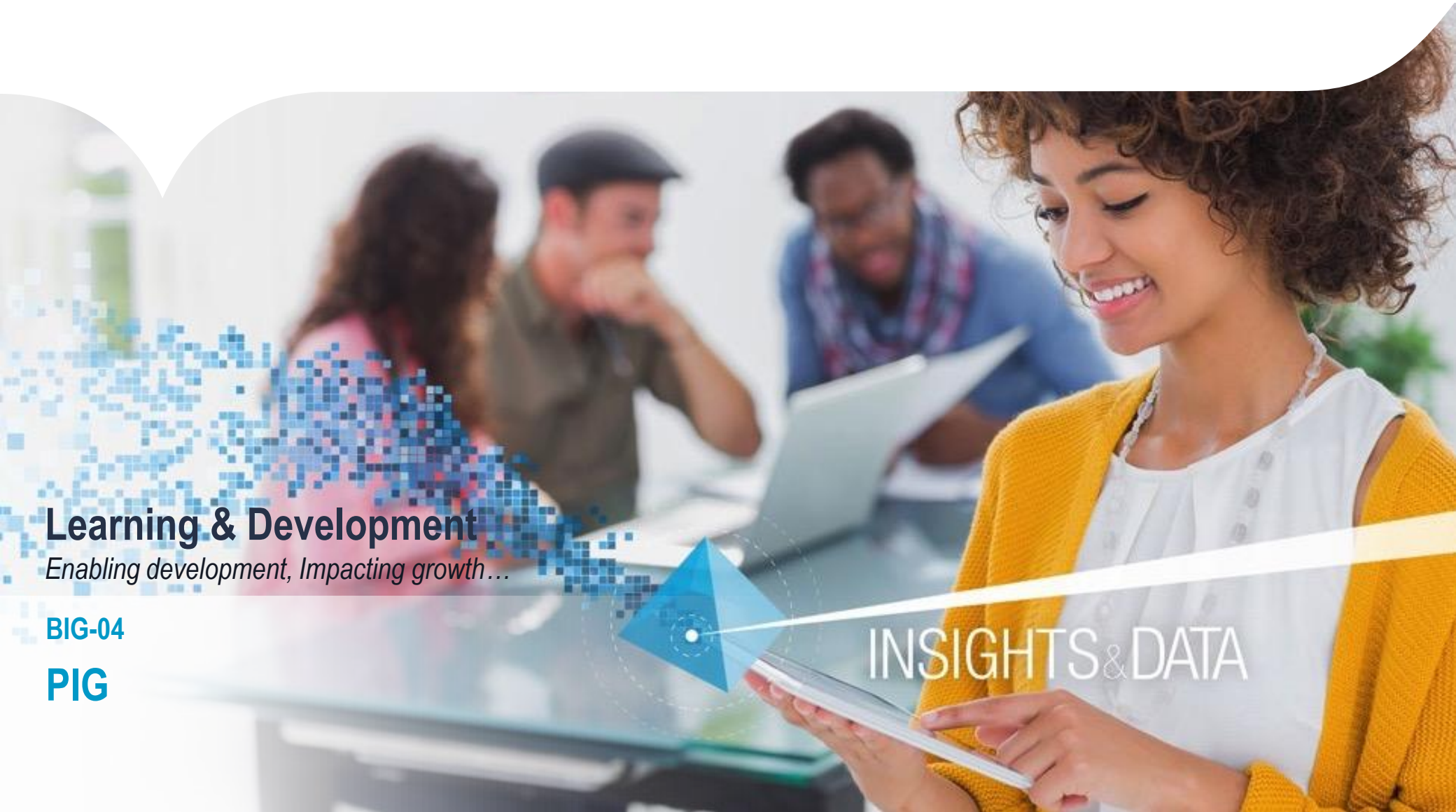
4. True or False? MapReduce can best be described as a programming model used to develop Hadoop-based applications that can process massive amounts of unstructured data.

A. True
B. False

A: True

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

People matter, results count.



Learning & Development

Enabling development, Impacting growth…

BIG-04

PIG

How are Organizations using Pig?

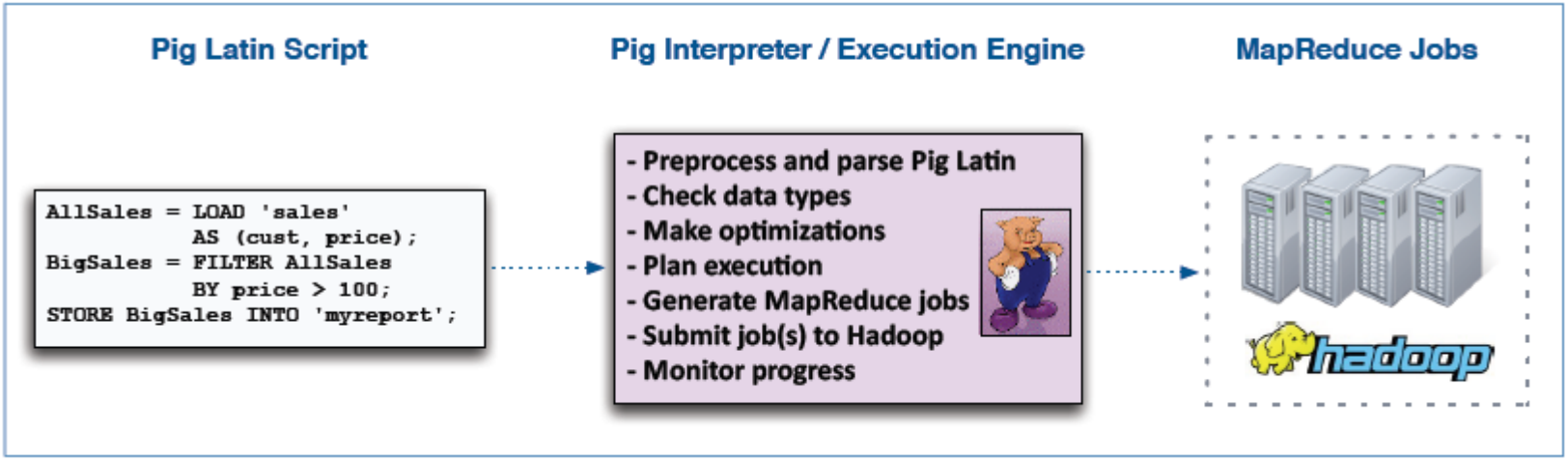
- Usage of Pig
  - Many Organization use Pig for data analyses to
  - Find relevant records in a massive data
  - Querying multiple data sets
  - Calculating values from input data
  - Pig also frequently used for data processing that includes
  - Reorganizing an existing data set
  - Joining dataset from multiple sources to produce a new data set.
- However, Organization uses 80% of Pig features for data processing purpose

Pig: Introduction

- **Pig was originally created at Yahoo! to answer a similar need to Hive**
  - Many developers did not have the Java and/or MapReduce knowledge required to write standard MapReduce programs
  - But still needed to query data
- **Pig is a dataflow language**
  - Language is called PigLatin
  - Relatively simple syntax
  - Under the covers, PigLatin scripts are turned into MapReduce jobs and executed on the cluster

The Pig Latin framework

- Main components of Pig
  - The data flow language (Pig Latin)
  - The Interactive shell where you can type Pig Latin Statements (Grunt shell)



The Pig Latin framework (cont..)

- **Pig Latin"** language is used to define data flows in Pig transformation. Pig Latin is called as data-flow language. Dataflow is a collection of **data pipes** wherein each pipe is an operation.

- The Operations can be loading data, transformation, data sorting, grouping of data, filtering data, aggregation etc

Pig - Execution Modes

- **Local Mode**
- **Mapreduce Mode**

Pig - Execution Modes (cont..)

- **Examples**
- This example shows how to run Pig in local and mapreduce mode using the pig command.
  - **local mode**

    $ pig -x local ...

  - **mapreduce mode**

    $ pig ... or
    $ pig -x mapreduce

Pig - Concepts

- In pig, a collection of elements – such as a row, or a partial row – is a tuple
- Tuples are collected together into bags
- Typically, a PigLatin script starts by loading one or more datasets into bags, and then creates new bags by modifying those it already has

Pig - Features

- Pig Latin is a data flow language rather than procedural or declarative.
- Pig Latin provides support for types (e.g., long, float, and chararray), schemas, and functions.
- Pig Latin is extensible and supports the definition of user defined functions.
- Metadata not required, but used when available.
- Operates on files in HDFS.

Pig Interacting with HDFS

```
grunt> fs -mkdir sales/;
grunt> fs -put europe.txt sales/;
grunt> fs -get europt.txt ;
```

Interacting with Linux/ Unix

grunt> sh date → display unix default date and time

grunt> sh ls ----> listing files from local file system

grunt> fs ls ----> listing files from default HDFS path

grunt> sh mkdir test ----> creating test directory in default local file systems

grunt> fs mkdir test ---> creating test directory in default HDFS

Running Pig in Interactive Mode

- You can run Pig in interactive mode using the Grunt shell.
- Invoke the Grunt shell using the "pig" command (as shown below) and then enter your Pig Latin statements and Pig commands interactively at the command line.
- The DUMP operator will display the results to your terminal screen.

grunt> A = load 'passwd' using PigStorage(':');

grunt> B = foreach A generate $0 as id;

grunt> dump B;

Pig Scripts

- Use Pig scripts to place Pig Latin statements and Pig commands in a single file
- It is good practice to create the file using the .pig extension, however it is not mandatory.

Comments in Script:

- You can include comments in Pig scripts:
- For multi-line comments use

/**myscript.pig My script is simple. It includes three Pig Latin statements* .... */

- For single-line comments use

--

Eg: A = LOAD 'student' USING PigStorage() AS (name:chararray, age:int, gpa:float); *-- loading data*

Data Types (cont..)

- fields default to type bytearray

- Implicit conversions are applied to the data depending on the context in which t

  - For example, in relation B, f1 is converted to integer because 5 is integer.
  - In relation C, f1 and f2 are converted to double because we don't know the type of either f1 or f2.hat data is used

```
A = LOAD ‘salaries.txt ' AS (gender,age,income);

B = FOREACH A GENERATE age + 5;

C = FOREACH A generate  age + income;
```

Data Types (cont..)

- If a schema is defined as part of a load statement, the load function will attempt to enforce the schema.

- If the data does not conform to the schema, the loader will generate a null value or an error.

  A = LOAD 'salaries.txt' AS (gender:chararray, age:int, income:float,zip:int);

Referencing Relations

- Relations are referred to by name (or alias). Names are assigned by you as part of the Pig Latin statement.

- The Below example shows the name (alias) of the relation is A.

```
A = LOAD ‘salaries.txt' USING PigStorage() AS (gender:chararray, age:int,
income:float,zip:int);
```

```
DUMP A;
        (M,40,76000,95102)
        (F,58,95000,95103)
        (F,68,60000,95105)
        (M,67,99000,94040)
        (F,37,65000,94040)
```

Referencing Fields

- Fields are referred to by positional notation or by name (alias).
- Positional notation is generated by the system. Positional notation is indicated with the dollar sign ($) and begins with zero (0); for example, $0, $1, $2.
- Names are assigned by you using schemas
  - for example, gender, age,income or $1,$3 etc

```
grunt> sal= load 'salaries.txt' USING PigStorage(',') as (gender,age,income,zip);

grunt>X= foreach sal generate age,$2;
grunt>DUMP X;

    (44,96000)
    (73,12000)
    (55,32000)
    (82,10000)
```

Relation without schema

- Can be used refential position of the column using $ and column postion (starting with 0)
- Example shows grouping records using 3rd column

```
grunt> salariesgroup= group sal by $2;

grunt> describe salariesgroup;

salariesgroup: {group: bytearray,sal: {(gender: bytearray,age: bytearray,income:
bytearray,zip: bytearray)}}
```

Viewing Schema definition using DESCRIBE

- The DESCRIBE Command shows the structure of the data including names and types

- The following grunt session shows an example

```
grunt> sal =  load 'salaries.txt AS (gender,age,income , zip);

grunt> DESCRIBE sal

sal: {gender: bytearray,age: bytearray,income: bytearray,zip:
bytearray}
```

Loading Data to Pig

- Use the LOAD operator to load data from the file system to **relation**.

  Syntax : *LOAD 'data' [USING function] [AS schema];*

- Pig's Default loading function is called PigStorage
  - The name of the function is implicit when calling LOAD
  - PigStorage assumes text format with tab separated
  - Consider the following file in HDFS called **Salary.txt**

Loading Data to Pig (cont..)

- Example to load data
  - The two fields are separated by tab-characters

M	44	96000	94040
F	73	12000	95102
M	55	32000	94040
F	82	10000	95102

```
grunt> sal=LOAD 'salaries.txt' As (gender,age)
```

- The above Command loads file from the default HDFS directory

Viewing of Output using DUMP

- The Command used to handle output depends on its destination
  - DUMP: Sends output to the screen
  - STORE: Sends output to disk (HDFS)

- Example of DUMP

```
grunt> dump sal;
```

Storing Data with Pig - STORE

- The STORE commands is used to store data to HDFS
  - Similar to LOAD, but writes data instead of reading it
  - The directory must be present in the HDFS

  Syntax

  STORE alias INTO 'directory' [USING function];

- As with LOAD the use of PigStorage is implicit
  - The field delimiter also has default value (tab)

    STORE bigsales INTO 'myreport' ---> writes in default HDFS directory

- You may specify an alternate delimeter using PigStorage

  STORE bigsales INTO 'myreport' USING PigStorage('|');

Sorting Records and Limiting Output - Order By & Limit

- Use ORDER .....BY to sort the records in a bag in ascending order
  - Add DESC to sort in descending order instead
  - Take care to specify a schema - data type affects how data is sorted!

Syntax

alias = ORDER alias BY { * [ASC|DESC] | field_alias [ASC|DESC] [, field_alias [ASC|DESC] …] } [PARALLEL n];

```
grunt> sortedsal=  order sal by income desc;
```

```
grunt>dump sortedsal;

(M,67,99000,94040)
(M,44,96000,94040)
(F,58,95000,95103)
(M,31,95000,94041)
(M,48,91000,95102)
```

Using Alternative Column Delimiters

- You can specify an alternate delimeter as an argument to PigStorage
- This example shows how to load comma-delimited data

```
grunt>sal= LOAD 'salaries.csv' USING PigStorage(',') AS (gender,age,income,
zip)
```

- To load (|) delimited file without specifying column name

```
grunt>sal= LOAD 'salaries.txt' USING PigStorage('|');
```

Arithmetic Comparison Operators - Examples

- Example the modulo operator is used with fields f1 and f2.

```
grunt >X = FOREACH A GENERATE f1, f2, f1%f2;


grunt> DUMP X;
    (10,1,0)
    (10,3,1)
    (10,6,4)
```

- Example: numeric

```
grunt > X = FILTER A BY (f1 == 8);
```

- Example: string

```
Grunt > X = FILTER A BY (f2 == 'apache');
```

- Example: matches

```
Grunt > X = FILTER A BY (f1 matches '.*apache.*');
```

Expressions

- Expressions are language constructs used with the FILTER, FOREACH, GROUP, and SPLIT operators as well as the eval functions.

- Example of an arithmetic expression:

```
X = GROUP A BY f2*f3;
```

- An Example of a string expression: *Note a and b are both chararrays:*

```
X = FOREACH A GENERATE CONCAT(a,b);
```

- Example of A boolean expression:

```
X = FILTER A BY (f1==8) OR (NOT (f2+f3 > f1));
```

Group Operator

Use GROUP BY to do this in Pig Latin

- The new relation has one record per unique group

```
grunt> salariesbyage = group sal by age;
```

```
grunt> dump salariesbyage;

(58,{(F,58,95000,95103)})
(65,{(F,65,70000,95102)})
(66,{(F,66,41000,95103),(M,66,84000,95103)})
(67,{(M,67,99000,94040),(M,67,81000,95101)})
(68,{(F,68,60000,95105),(M,68,15000,95103)})
(71,{(M,71,0,94041)})
```

A Sample Pig Script

```
emps = LOAD 'people.txt' AS (id, name, salary);
rich = FILTER emps BY salary > 100000;
srtd = ORDER rich BY salary DESC;
STORE srtd INTO 'rich_people';
```

- **Here, we load a file into a bag called emps**
- **Then we create a new bag called rich which contains just those records where the salary portion is greater than 100000**
- **Finally, we write the contents of the srtd bag to a new directory in HDFS**
  - –By default, the data will be written in tab-separated format
- **Alternatively, to write the contents of a bag to the screen, say**

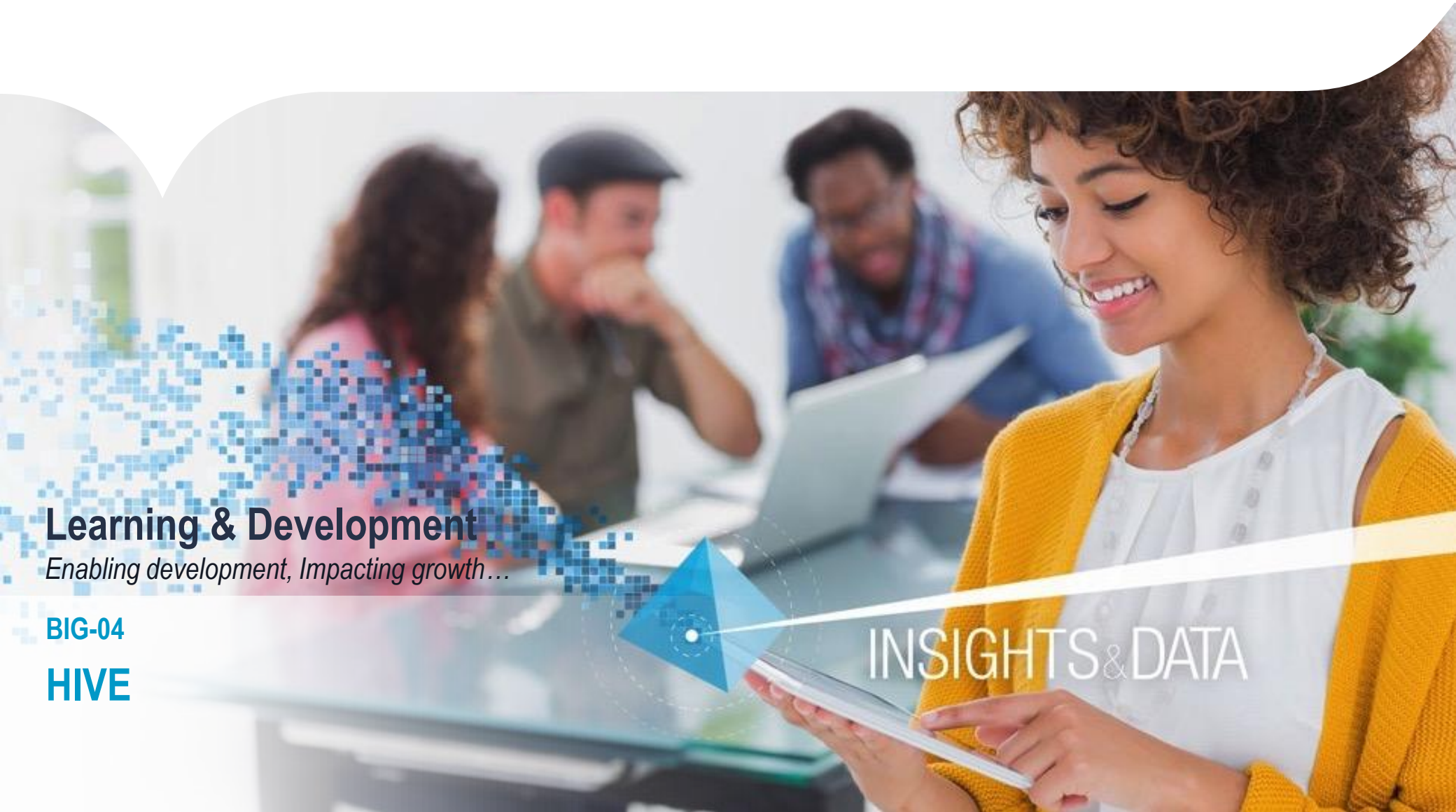
```
DUMP srtd;
```

Pig Operators

Pig Command	What it does
load	Read data from file system.
store	Write data to file system.
Foreach	Apply expression to each record and output one or more records.
filter	Apply predicate and remove records that do not return true.
group/cogroup	Collect records with the same key from one or more inputs.
join	Join two or more inputs based on a key.
order	Sort records based on a key.
distinct	Remove duplicate records.
union	Merge two data sets.
split	Split data into 2 or more sets, based on filter conditions.
stream	Send all records through a user provided binary.
dump	Write output to console.
limit	Limit the number of records.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

People matter, results count.



Learning & Development

Enabling development, Impacting growth…

BIG-04

HIVE

Motivation to Hive

- MapReduce code is typically written in Java
  - Although it can be written in other languages using Hadoop Streaming API

- Requires:
  - A programmer
  - Who is a good Java programmer
  - Who understands how to think in terms of MapReduce
  - Who understands the problem they're trying to solve
  - Who has enough time to write and test the code
  - Who will be available to maintain and update the code in the future as requirements change

What is Hive?

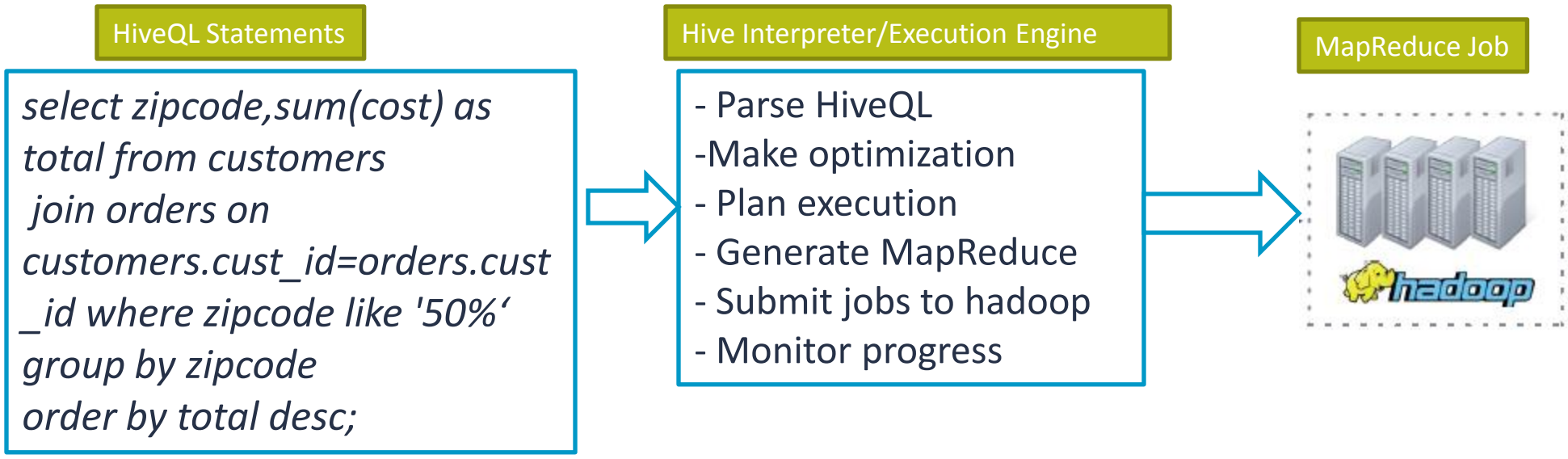
- A data warehousing system to store structured data on Hadoop file system
- Provide an easy query these data by execution Hadoop MapReduce plans

- **Intuitive**

  - Make the unstructured data looks like tables regardless how it really lay out
  - SQL based query can be directly against these tables
  - Generate specify execution plan for this query

What is Hive? (cont..)

- Hive runs on the client machine
  - Turns HiveQL queries into MapReduce jobs
  - Submit those jobs to the cluster



Why do we use Apache Hive?

- More productive than writing MR jobs
  - Five lines of HiveQL might be equivalent to 100 lines or more of Java
- Brings large-scale data analysis to a broader audience
  - No Software development experience required
  - Leverage existing knowledge of SQL
- Offers interpretability with other systems
  - Extensible through java and external scripts
  - Many BI tools supports hive

Hive Vs RDBMS

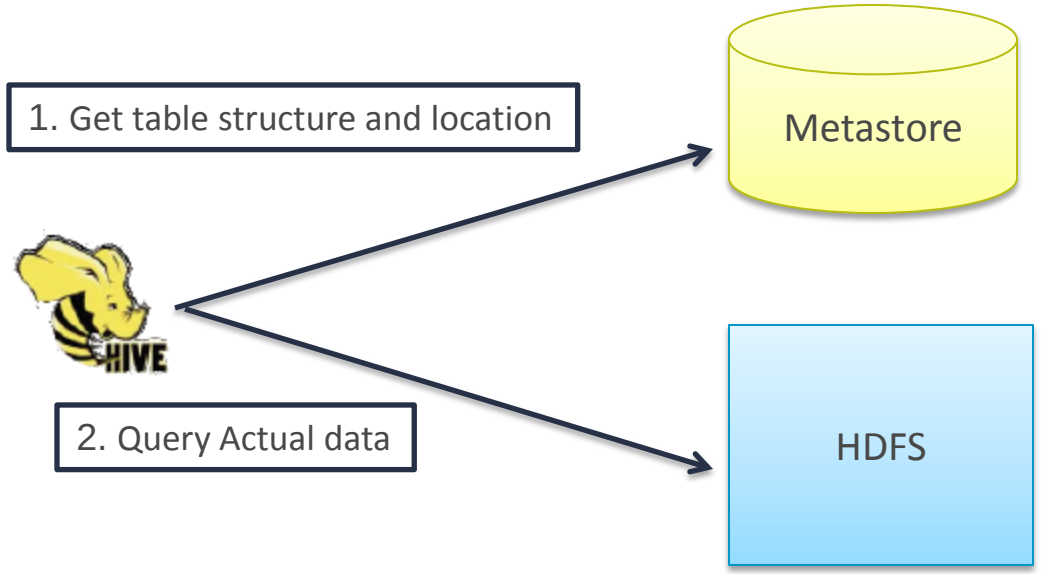
Hive	RDBMS
SQL Interface.	SQL Interface.
Focus on analytics.	May focus on online or analytics.
No transactions.	Transactions usually supported.
Partition adds, no random INSERTs. In-Place updates not natively supported (but are possible).	Random INSERT and UPDATE supported.
Distributed processing via map/reduce.	Distributed processing varies by vendor (if available).
Scales to hundreds of nodes.	Seldom scale beyond 20 nodes.
Built for commodity hardware.	Often built on proprietary hardware (especially when scaling out).
Low cost per petabyte.	What's a petabyte?

A High level comparison of SQL and HiveQL

Feature	SQL	HiveQL
Updates	UPDATE, INSERT, DELETE	INSERT OVERWRITE TABLE (populates whole table or partition)
Transactions	Supported	Not supported
Indexes	Supported	Not supported
Latency	Sub-second	Minutes
Data types	Integral, floating point, fixed point, text and binary strings, temporal	Integral, floating point, boolean, string, array, map, struct
Functions	Hundreds of built-in Functions	Dozens of built-in functions
Multitable inserts	Not supported	Supported
Joins	Join tables in the FROM clause, join condition in the WHERE clause	Inner joins, outer joins, semi joins, map joins.
Subqueries	In any clause. Correlated or noncorrelated	Only in the FROM clause. Correlated subqueries not supported

Hive Data Model

- How Hive Stores loads data and store?
- Hive consults the metastore to determine data format and location
- The query itself operates on data stored on a filesystem (typically HDFS



The Hive Metastore

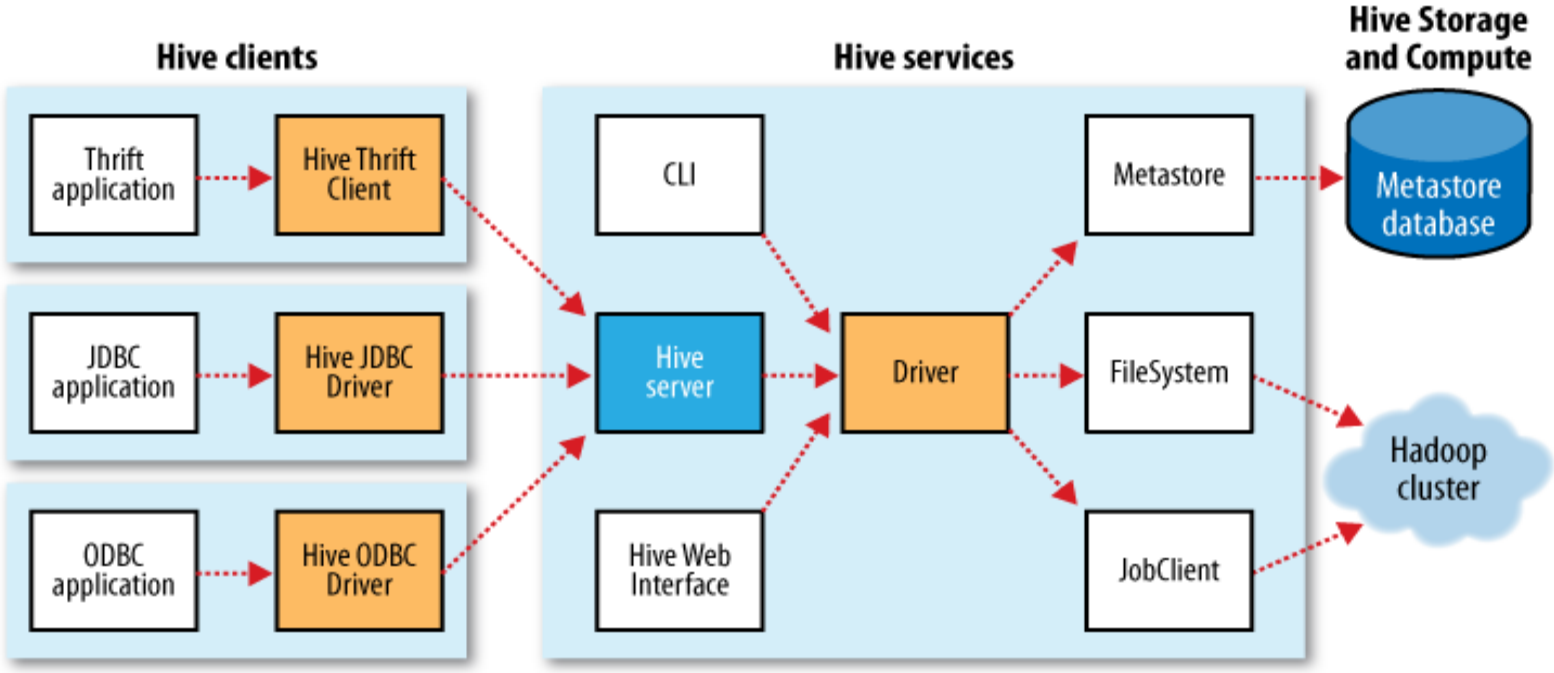
- **Hive's Metastore is a database containing table definitions and other metadata**
  - By default, stored locally on the client machine in a Derby database
  - If multiple people will be using Hive, the system administrator should create a shared Metastore
  - Usually in MySQL or some other relational database server

Hive Data Types

Hive SQL Datatypes	Hive SQL Semantics
INT	SELECT, LOAD, INSERT from query
TINYINT/SMALLINT/BIGINT	Expressions in WHERE and HAVING
BOOLEAN	GROUP BY, ORDER BY, SORT BY
FLOAT	Sub-queries in FROM clause
DOUBLE	GROUP BY, ORDER BY
STRING	CLUSTER BY, DISTRIBUTE BY
TIMESTAMP	ROLLUP and CUBE
BINARY	UNION
ARRAY, MAP, STRUCT, UNION	LEFT, RIGHT and FULL INNER/OUTER JOIN
DECIMAL	CROSS JOIN, LEFT SEMI JOIN
CHAR	Windowing functions (OVER, RANK, etc.)
VARCHAR	INTERSECT, EXCEPT, UNION DISTINCT
DATE	Sub-queries in WHERE (IN/NOT IN, EXISTS/NOT EXISTS
	Sub-queries in HAVING

Hive 0.10
Hive 0.11
Future

Hive Architecture



Starting The Hive Shell

- **To launch the Hive shell, start a terminal and run**
- **$ hive**
- **Results in the Hive prompt:**
- **hive>**

Hive Basics: Creating Tables

```
hive> SHOW TABLES;

hive> CREATE TABLE shakespeare (freq INT, word STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'
STORED AS TEXTFILE;

hive> DESCRIBE shakespeare;
```

Storing Output Results

- **The SELECT statement on the previous slide would write the data to the console**
- **To store the results in HDFS, create a new table then write, for example:**

```
INSERT OVERWRITE TABLE newTable
        SELECT s.word, s.freq, k.freq FROM
        shakespeare s JOIN kjv k ON
        (s.word = k.word)
        WHERE s.freq >= 5;
```

  - Results are stored in the table
  - Results are just files within the newTable directory
    - Data can be used in subsequent queries, or in MapReduce jobs

Hive Data: Physical Layout

- Hive tables are stored in Hive's 'warehouse' directory in HDFS
  - By default, /user/hive/warehouse

Loading Data into a Hive Table

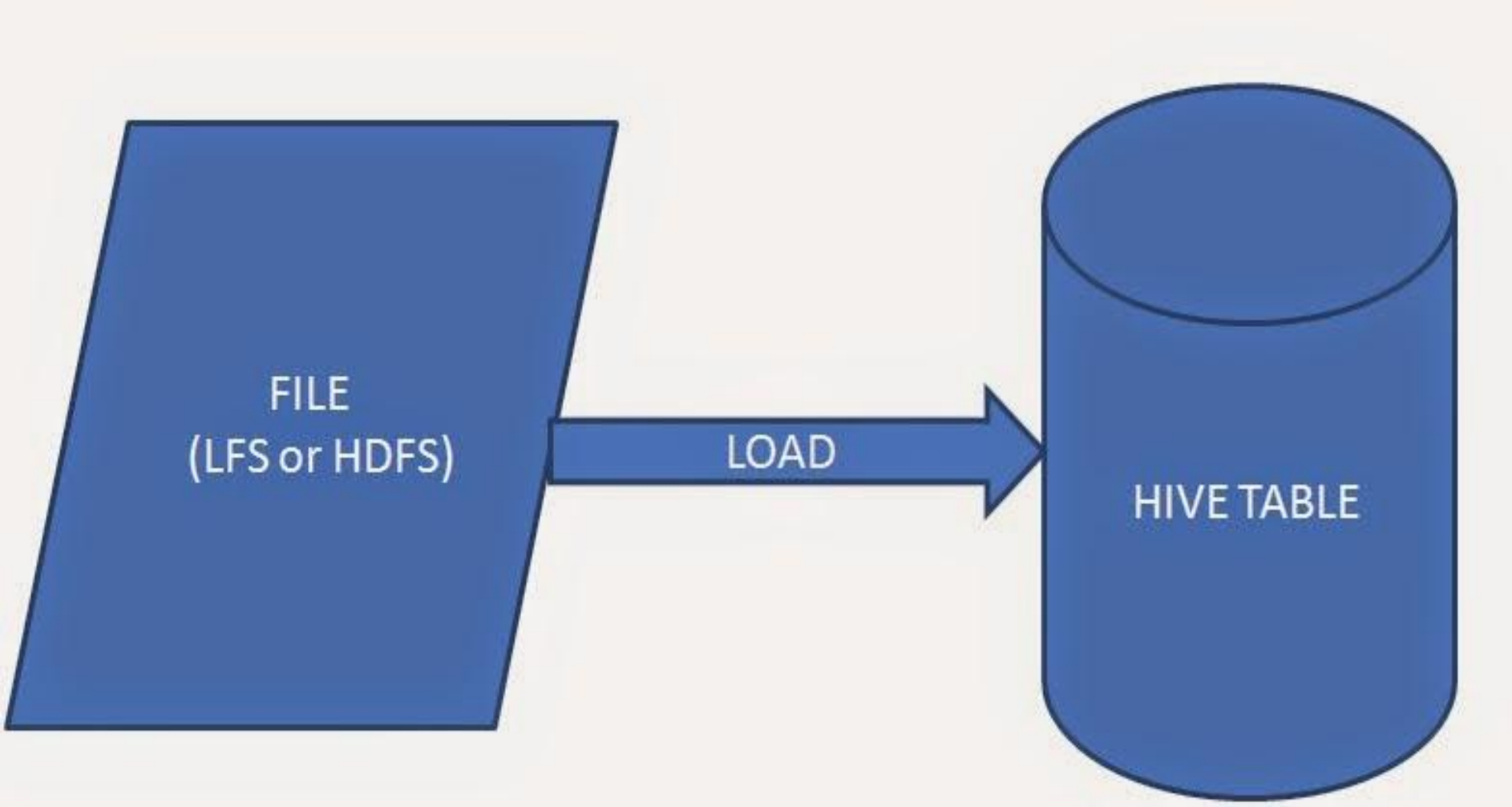
- Hive does not do any transformation while loading data into tables.
- Load operations are currently pure copy/move operations that move data files into locations corresponding to Hive tables.
- The load command will look for file path in the local file system
- If the keyword LOCAL is not specified, then the load command will look for file path in the HDFS
- Data is loaded into Hive with the LOAD DATA INPATH statement

  - assumes that the data is already in HDFS

- If the data is on the local filesystem, use LOAD DATA LOCAL INPATH

  - Automatically loads it into HDFS

Loading files into Hive tables



Basic SELECT Queries

- **Hive supports most familiar SELECT syntax**

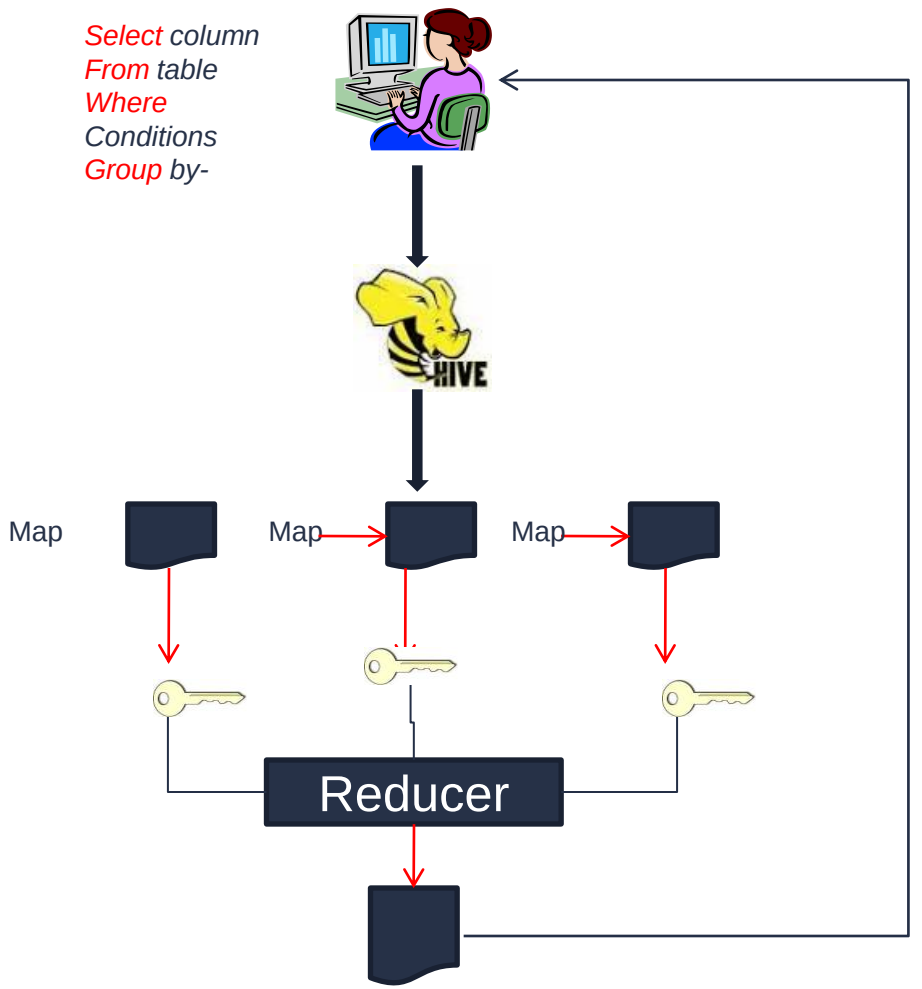
```
hive> SELECT * FROM shakespeare LIMIT 10;

hive> SELECT * FROM shakespeare
WHERE freq > 100 ORDER BY freq ASC
LIMIT 10;
```

Joining tables

```
hive> SELECT s.word, s.freq, k.freq FROM
          shakespeare s JOIN kjv k ON
          (s.word = k.word)
          WHERE s.freq >= 5;
```

Hive Workflow diagram

HQL statements are broken down by Hive service into MapReduce jobs and executed across the Hadoop cluster.



Hive Limitations

- **Not all 'standard' SQL is supported**
  - No correlated subqueries, for example
- **No support for UPDATE or DELETE prior to Hive 0.14**
- **No support for INSERTing single rows**
- **Relatively limited number of built-in functions**
- **No datatypes for date or time**
  - Use the STRING datatype instead

Thank You