Big Data Analytics

UNIT - I

INTRODUCTION TO BIG DATA: Big Data Analytics, Characteristics of Big Data – The Four Vs, importance of Big Data, Different Use cases, Data-Structured, Semi-Structured, Un-Structured.

INTRODUCTION TO HADOOP : Hadoop and its use in solving big data problems, Comparison Hadoop with RDBMS, Brief history of Hadoop, Apache Hadoop EcoSystem, Components of Hadoop, The Hadoop Distributed File System (HDFS):, Architecture and design of HDFS in detail, Working with HDFS (Commands)

INTRODUCTION

Data

- Data is nothing but facts and statistics stored or free flowing over a network, **generally it's raw and unprocessed**.
- When data are processed, organized, structured or presented in a given context so as to make them useful, they are called Information.

What is Big Data? –

➢ Big Data is a term used to describe a collection of data that is huge in size and yet growing exponentially with time.

➢ In short such data is so large and complex that none of the traditional data management tools are able to store it or process it efficiently.

Other Definition: **Big data** is a term that is used to describe data that is high volume, high velocity, and/or high variety, requires new technologies and techniques to capture, store, and analyze it, and is used to enhance decision making.

- **Data analytics** converts raw data into actionable insights. It includes a range of tools, technologies, and processes used to find trends and solve problems by using data.

- Data analytics can shape business processes, improve decision- making, and business growth.

Example of Big Data:

Following are some of the Big Data examples-

- The **New York Stock Exchange** is an example of Big Data that generates about **one terabyte** of new trade data per day.

- **Social Media**: The statistic shows that **500+terabytes** of new data get ingested into the databases of social media site **Facebook**, every day. This data is mainly generated in terms of photo and video uploads, message exchanges, putting comments etc.

- A single **Jet engine** can generate **10+terabytes** of data in **30 minutes** of flight time. With many thousand flights per day, generation of data reaches up to many **Petabytes.**

DATA RANGE


Normal Range
Kilo Byte (KB) 10^3
Mega Byte (MB) 10^6
Giga Byte (GB) 10^9
Tera Byte (TB) 10^{12}
Big Data Range
Peta Byte (PB) 10^{15}
Exa Byte (EB) 10^{18}
Zetta Byte (ZB) 10^{21}
Yottabyte (YB) 10^{24}

Applications:

- **Healthcare:** With the help of predictive analytics, medical professionals are now able to provide personalized healthcare services to individual patients.
- **Academia:** Big Data is also helping enhance education today. there are numerous online educational courses to learn from. Academic institutions are investing in digital courses powered by Big Data technologies to aid the all-round development of budding learners.
- **Banking:** The banking sector relies on Big Data for fraud detection. Big Data tools can efficiently detect fraudulent acts in real-time such as misuse of credit/debit cards.
- **IT :** One of the largest users of Big Data, IT companies around the world are using Big Data to optimize their functioning, enhance employee productivity, and minimize risks in business operations. By combining Big Data technologies with ML and AI, the IT sector is continually powering innovation to find solutions even for the most complex of problems.

Big Data Challenges:

The major challenges associated with big data are as follows −

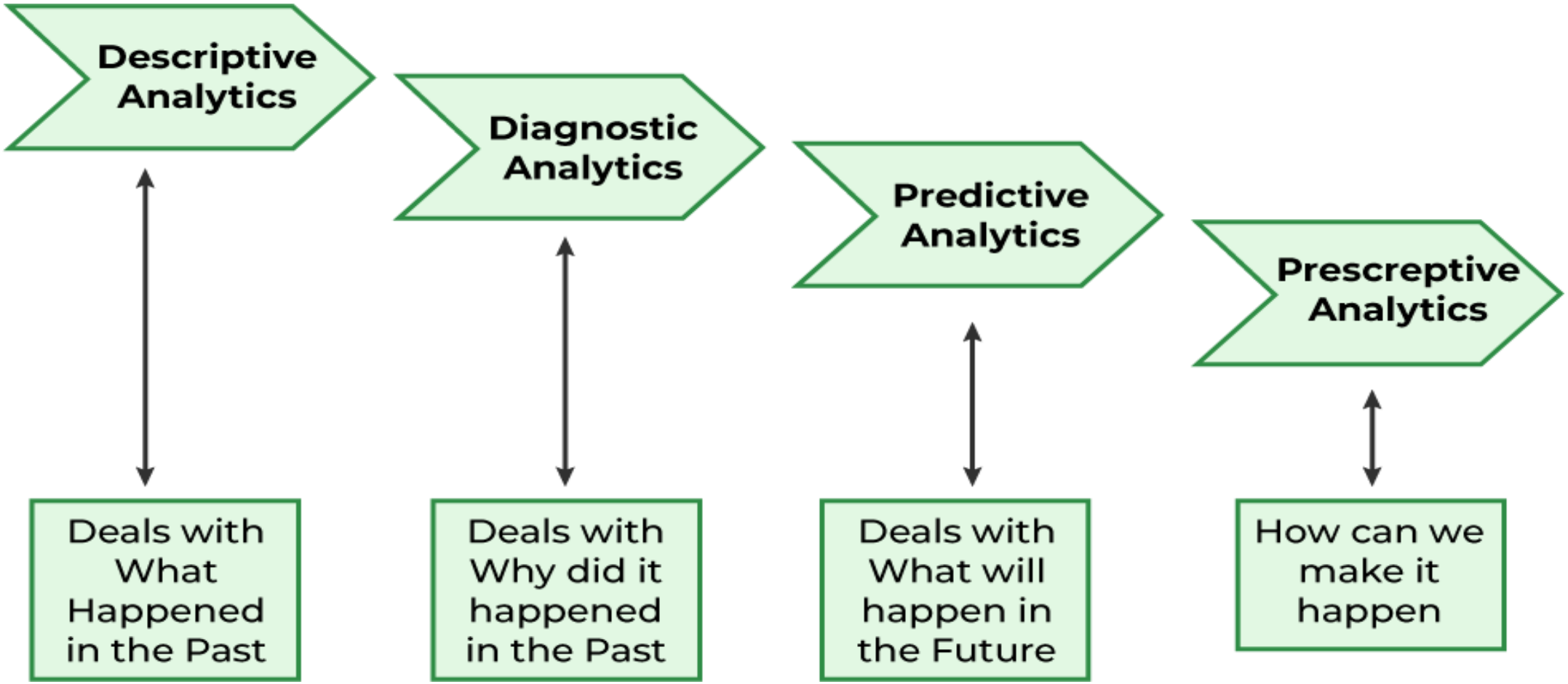
- Capturing data
- Storage
- Security
- Analysis

To fulfill the above challenges, organizations normally take the help of enterprise servers.

BIG DATA ANALYTICS

- The Process of Analysis of large volumes of Diverse data sets using Advanced Analytical Techniques is referred as Big Data Analytics.

There are **mainly 4 types of analytics**:

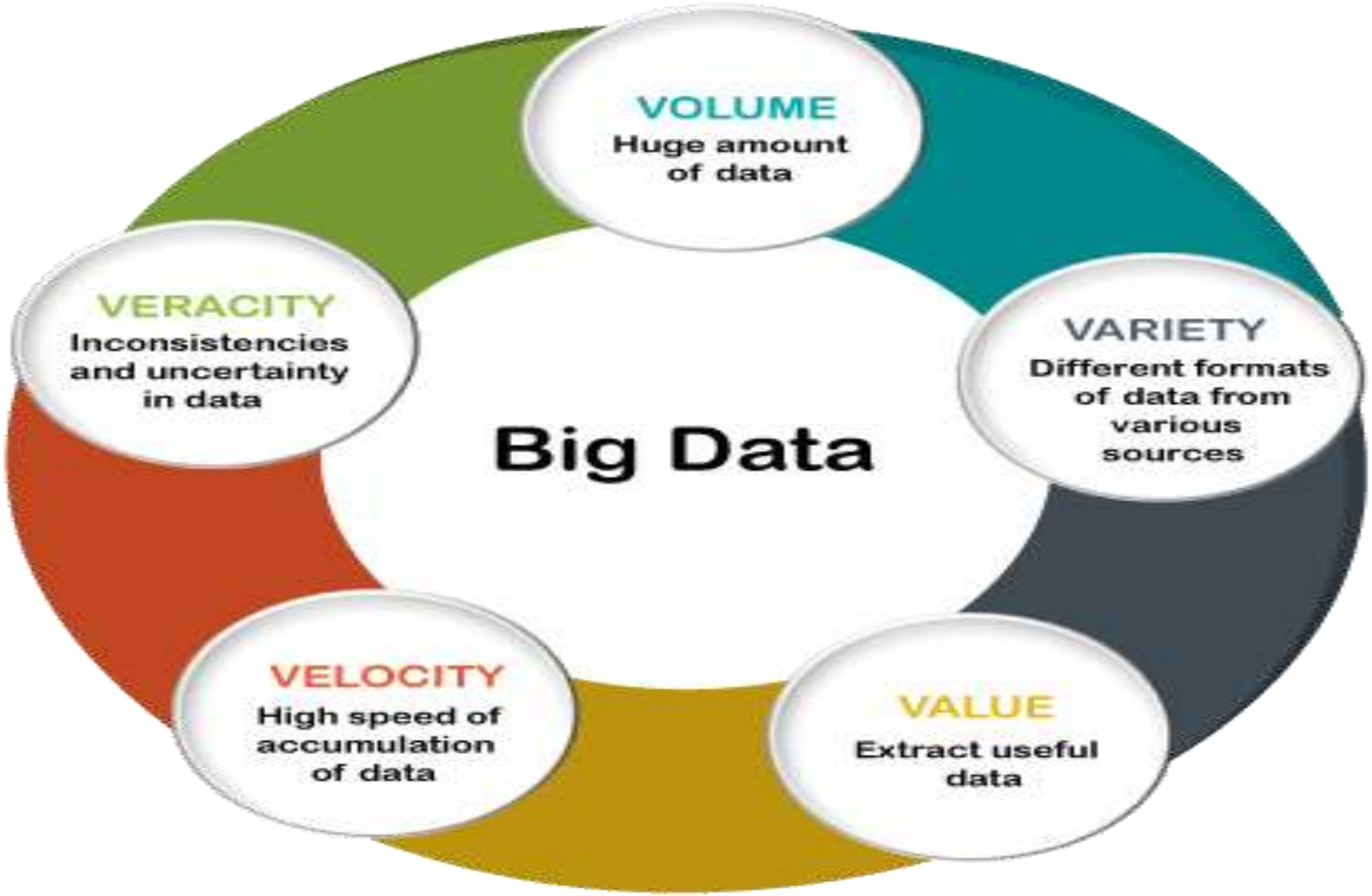


- **Descriptive Analytics:** Descriptive Analytics is focused solely on historical data.describing without judgment. It tells what happen in the past and uses some statistics functions and returns answers**(“What happened?”)**
  Ex: Google Analytics and Netflix

- **Predictive Analytics** : States that some specified event will happen in future i.e. what might happen in future, to reduce loses for the users. **(“What might happen in the future?”).** Predictive Analytics is a statistical method that utilizes algorithms and machine learning to identify
  trends in data and predict future behaviors.

- **Prescriptive Analytics:** Imposition of a rule/method. What we should do like what kind of action should be performed. Eg: self driving cars which analyzes environment and moves on the road**.(“What should we do next?”)**
- **Diagnostic analytics:** It finds out the root cause for happened thing in medical diagnosis**.(why Did this happen.)**

Characteristics of Big Data – The Five V's



➢Volume:

Volume refers to the unimaginable amounts of information generated every second from social media, cell phones, cars, credit cards, M2M sensors, images, video, and whatnot. The amount of data which we deal with is of very large size of Peta bytes.According to sources each day 2.3 trillion gigabytes of new data is being created.

➢Velocity: (https://www.javatpoint.com/big-data-characteristics)

The term **'velocity'** refers to the speed of generation of data in real time. How fast the data is generated and processed to meet the demands, determines real potential inthedata

➢Variety:

Big Data is generated in multiple varieties**. Variety** of Big Data refers to structured, unstructured, and semi-structured data that is gathered from multiple sources. While in the past, data could only be collected from spreadsheets anddatabases, today data comes in an array of forms such as emails, PDFs, photos, videos, audios, and so much more.

➢Veracity:

Veracity refers to theTrustworthiness in terms of quality and accuracy .

➢ Value:

It is **valuable** and **reliable** data that we **store, process**, and also **analyze**.

IMPORTANCE OF BIG DATA:

- Big Data importance doesn't revolve around the amount of data a company has. Its importance lies in the fact that how the company utilizes the gathered data.
- Every company uses its collected data in its own way. More effectively the company uses its data, more rapidly it grows.

The companies in the present market need to collect it and analyze it because:

1.Cost Savings:

Big Data tools like Apache Hadoop, Spark, etc. bring cost-saving benefits to businesses when they have to store large amounts of data. These tools help organizations in identifying more effective ways of doing business.

2.Time-Saving:

Real-time in-memory analytics helps companies to collect data from various sources. Tools like Hadoop help them to analyze data immediately thus helping in making quick decisions based on the learnings.

3.Understand the market conditions:

Big Data analysis helps businesses to get a better understanding of market situations.

For example, analysis of customer purchasing behavior helps companies to identify the products sold most and thus produces those products accordingly. This helps companies to get ahead of their competitors.

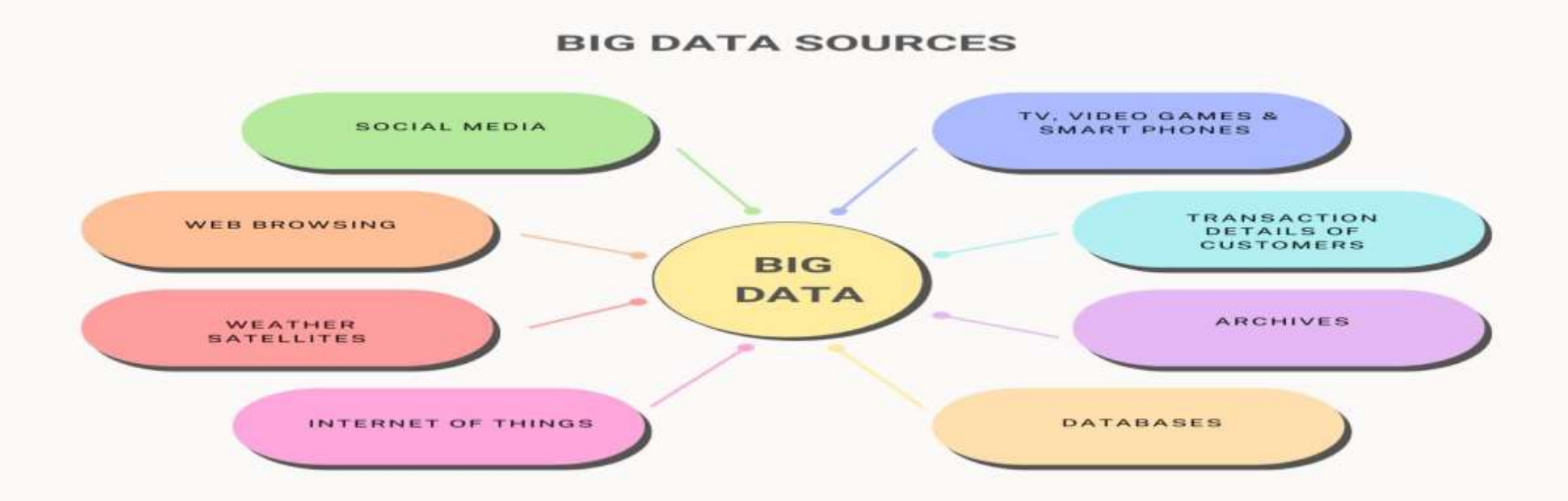
4.Social Media Listening/Control online reputation:

Companies can perform sentiment analysis using Big Data tools. These enable them to get feedback about their company, that is, who is saying what about the company.

5.Big Data Analytics in Product Development:

Another huge advantage of big data is the ability to help companies innovate and redevelop their products

SOURCES OF BIG DATA



Data Sources

- Web logs
- Internet text and documents
- Social networks
- Internet search indexing
- RFID & sensor networks
- Military surveillance

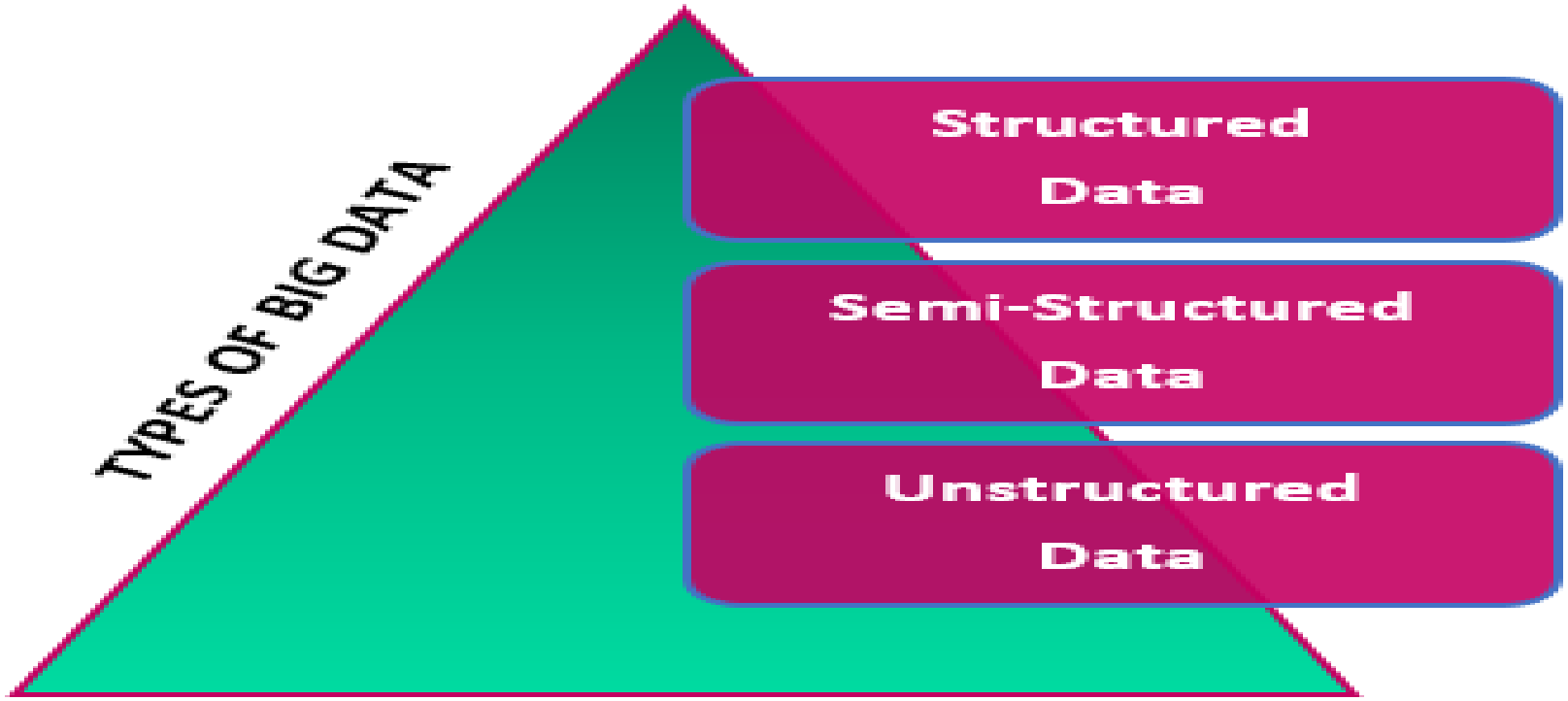
DIFFERENT USE CASES:

Big Data Use-Cases

Let’s discuss various use cases of Big data. Below are some of the Big data use cases from different domains:

- Netflix Uses Big Data to Improve Customer Experience
- Promotion and campaign analysis by Sears Holding
- Sentiment analysis
- Customer Churn analysis
- Predictive analysis
- Real-time ad matching and serving

BIG DATA / DIGITAL DATA TYPES:

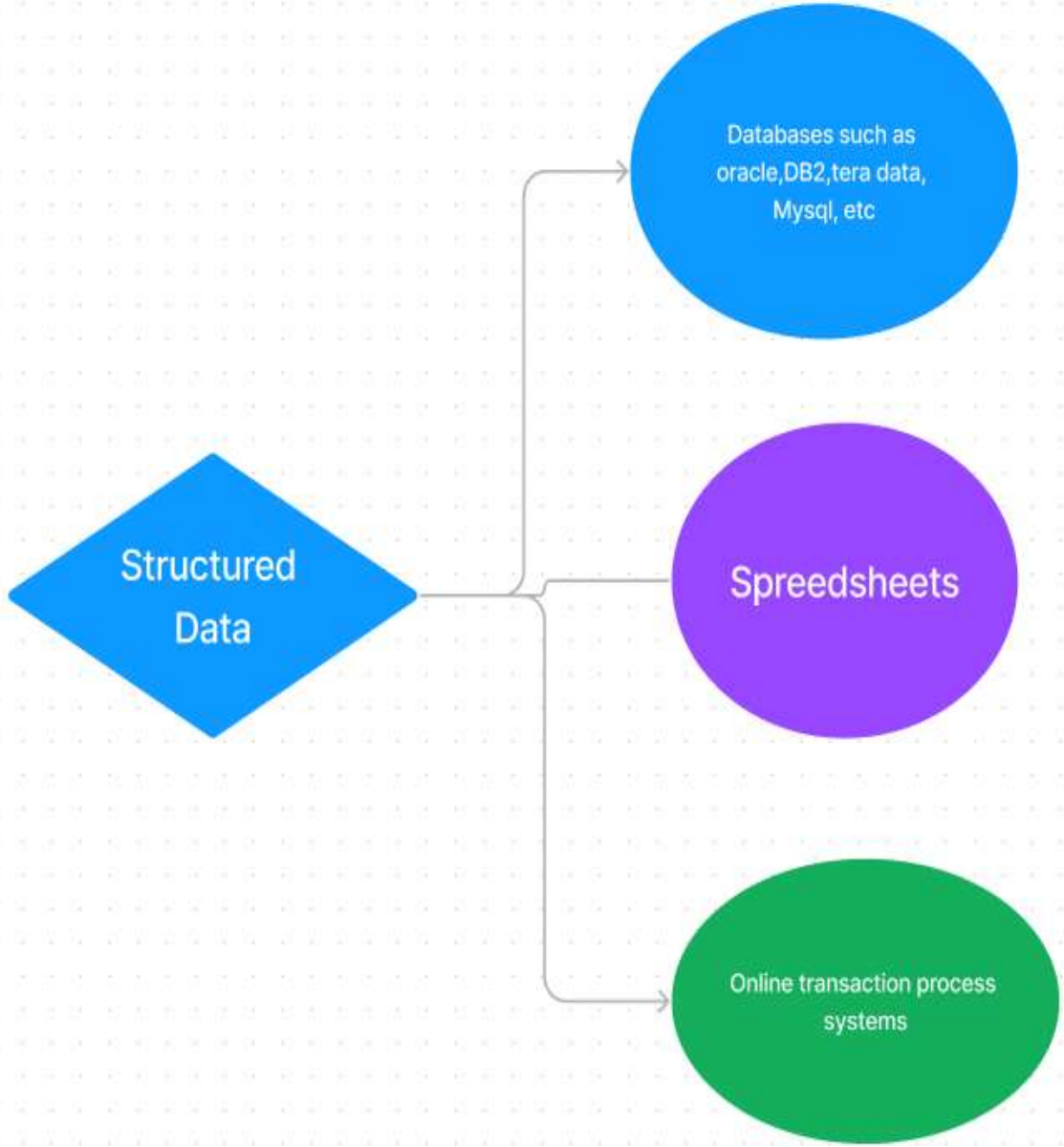


➢**STRUCTURED DATA** : Structured is one of the types of big data and By structured data, we mean data that can be processed, stored, and retrieved in a fixed format. It is represented in a Tabular Format.

➢**Eg:** An '**Employee'** table in a database is an example of Structured Data, spreadsheets data etc.

Employees Table

Employee ID	Last	First	Phone Number	Work Location	Project 1	Project 2	Project 3
AA123	Adams	Adam	222-333-4444	Smith Tower 22222			
BB234	Breen	Betty	333-444-5555	Nakatomi Plaza 33333			
CC456	Chen	Chao	444-555-6666	Smith Tower 22222	RDBMS Update May 2017	Network Security Audit May 2018	
DD789	Dickinson	Durah	555-666-7777	Nakatomi Plaza 33333	OS Update for PCs Sept 2017	OS Update for Macs Jan 2018	
EE012	Edinburgh	Elvis	666-777-8888	Tall Tower 22222	Network Security Audit May 2018	OS Update for Macs Jan 2018	
FF345	Fawzi	Farah	888-999-0000	Tall Tower 22222			
GG456	Giovanni	Georgio	000-111-2222	Nakatomi Plaza 33333	RDBMS Update May 2017	Network Security Audit May 2018	

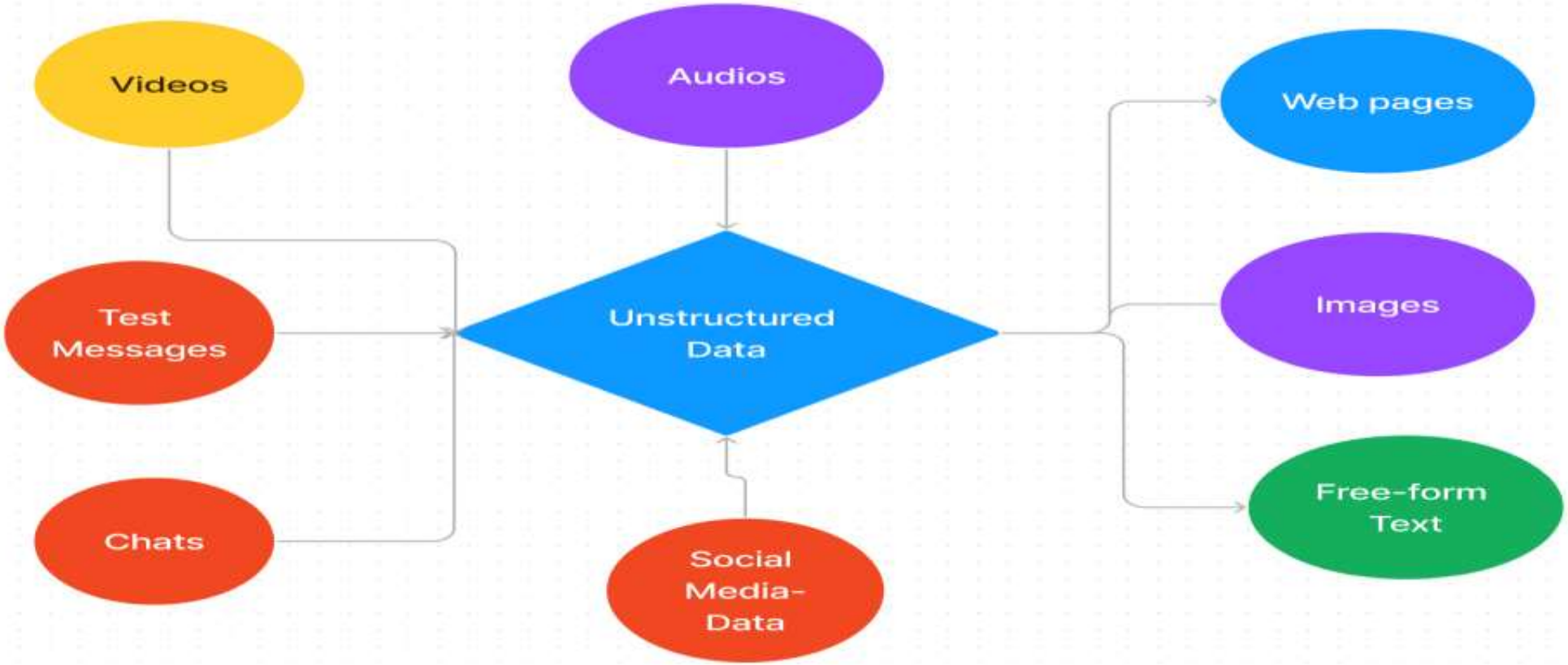

Structured Data
Databases such as oracle,DB2,tera data, Mysql, etc
Spreedsheets
Online transaction process systems

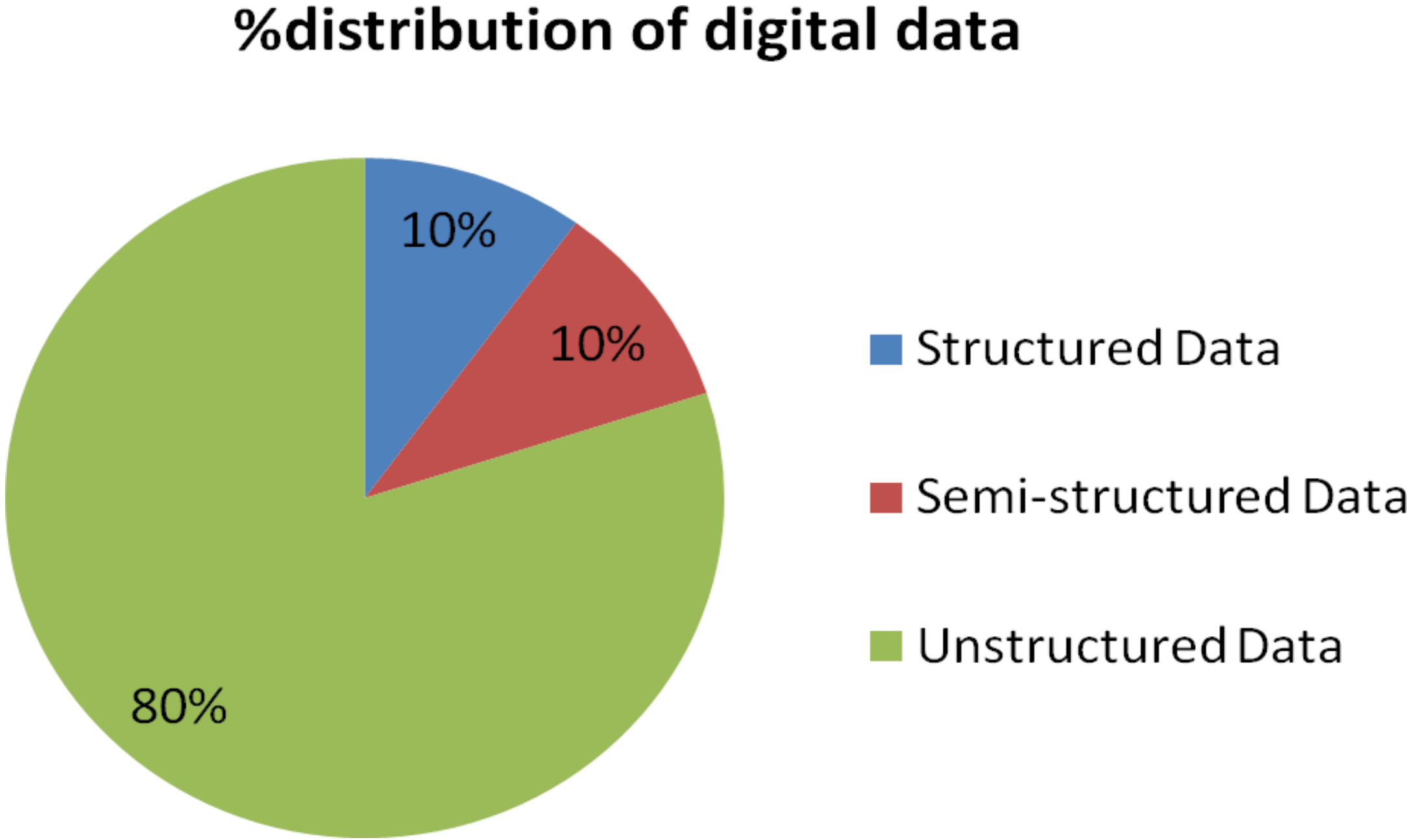
SEMI-STRUCTURED DATA:

- Semi-structured data can contain both the forms of data.
- The data is not in the relational format and is not neatly organized into rows and columns like that in a spreadsheet.
- Since semi-structured data doesn't need a structured query language, it is commonly called *NoSQL data*.Semi-structured content is often used to store metadata about a business process but it can also include files containing machine instructions for computer programs.
- This type of information typically comes from external sources such as social media platforms or other web-based data feeds.
  Eg: Zip files, e-mails, HTML
- xml files(extensible markup language, is a text-based markup language designed to store and transport data)etc

UN-STRUCTURED DATA:

- Unstructured data refers to the data that lacks any specific form or structure whatsoever. This makes it very difficult and time-consuming to process and analyze unstructured data.
- Additionally, Unstructured data is also known as “dark data” because it cannot be analyzed without the proper software tools
- Eg: Email, Audio, simple text files, images, videos , sensor data, Websites, logs etc.




%distribution of digital data
10%
10%
80%
Structured Data
Semi-structured Data
Unstructured Data

Evolution of Big Data

Data has evolved in the last 5 years like never before. Lots of data is being generated each day in every business sector

©Simplilearn. All rights reserved.

Evolution of Big Data

Data has evolved in the last 5 years like never before. Lots of data is being generated each day in every business sector

©Simplilearn. All rights reserved.

Evolution of Big Data

Here are some facts to convince you that data is exploding and needs your attention

Every minute, users send 31.25 million messages and watch 2.77 million videos on Facebook

Walmart handles more than 1 million customer transactions every hour

40,000 search queries are performed on Google per second, i.e. 3.46 million searches a day

300 hours of video are uploaded every minute on YouTube

IDC reports that by 2025, real time data will be more than a quarter of all the data

55 billion messages and 4.5 billion photos are sent each day on WhatsApp

By 2025, the volume of digital data will increase to 163 zettabytes

simplilearn

©Simplilearn. All rights reserved.

What is Big Data?

simplilearn

What is Big Data?

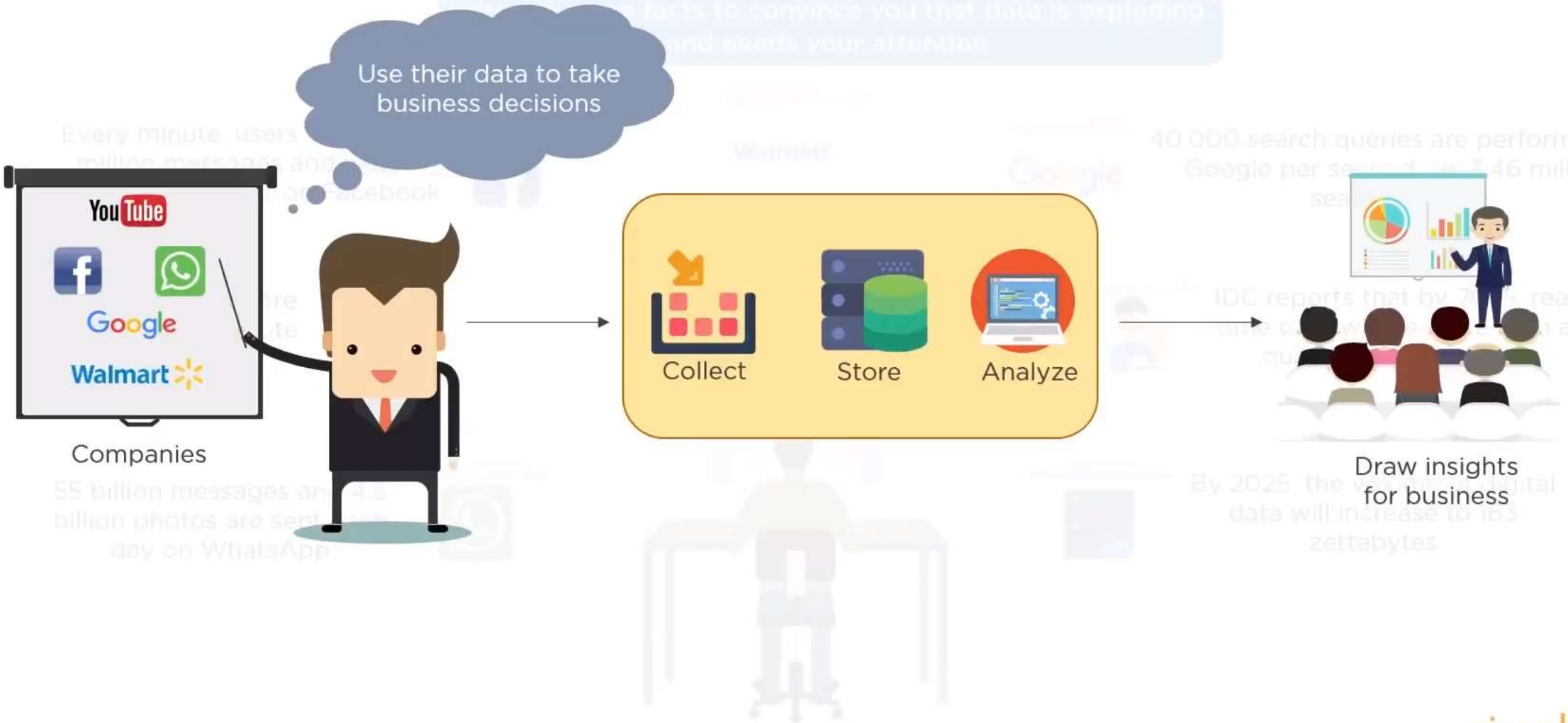
Big Data is a term used to describe data that is too large and complex to store in traditional databases

Used to

simplilearn

©Simplilearn. All rights reserved.

Why Big Data?



©Simplilearn. All rights reserved.

5 V's of Big Data

5 V's of Big Data

©Simplilearn. All rights reserved.

5 V's of Big Data

Size of the data

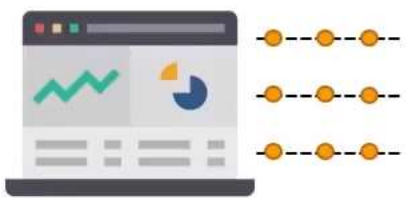
Volume

- Incredible amount of data
- Data generated each second
- From cell phones, social media, online transactions, etc.

©Simplilearn. All rights reserved.

5 V's of Big Data

Speed at which data is generated

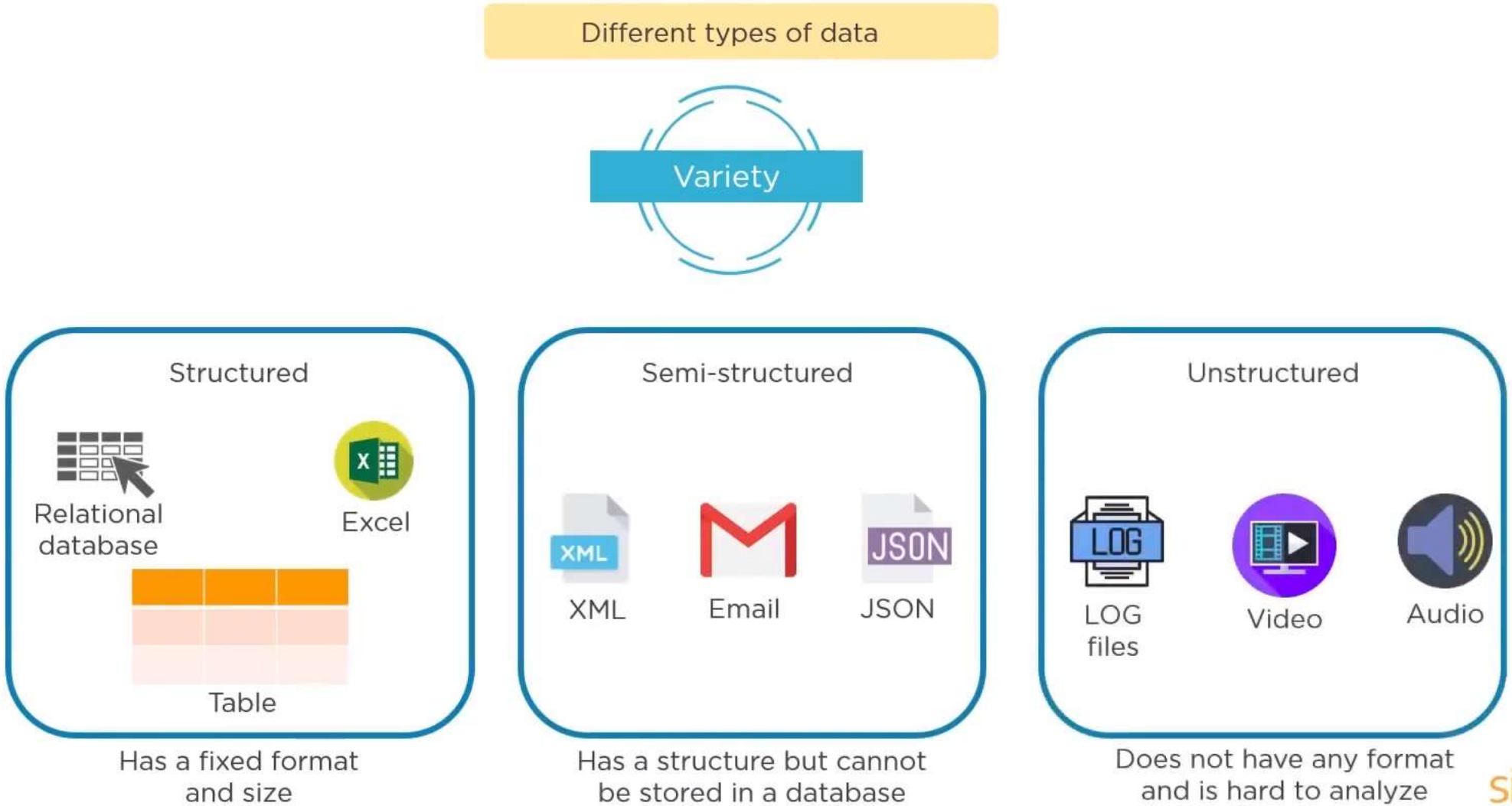


Speed at which data is:

- Generated
- Collected
- Analyzed

©Simplilearn. All rights reserved.

5 V's of Big Data



©Simplilearn. All rights reserved.

•Examples of Unstructured data

•Web pages
•Images (JPEG, GIF, PNG, etc.)
•Videos
•Memos
•Reports
•Word documents and PowerPoint presentations
•Surveys

application_log - Notepad

File Edit Format View Help

```
2015-12-31 02:16:59,929 WARN [main] org.apache.hadoop.metrics2.impl.MetricsConfig: Cannot locate configuration: tried hadoop-metrics2
-maptask.properties,hadoop-metrics2.properties
2015-12-31 02:17:00,004 INFO [main] org.apache.hadoop.metrics2.impl.MetricsSystemImpl: Scheduled snapshot period at 10 second(s).
2015-12-31 02:17:00,004 INFO [main] org.apache.hadoop.metrics2.impl.MetricsSystemImpl: MapTask metrics system started
2015-12-31 02:17:00,016 INFO [main] org.apache.hadoop.mapred.YarnChild: Executing with tokens:
2015-12-31 02:17:00,016 INFO [main] org.apache.hadoop.mapred.YarnChild: Kind: mapreduce.job, Service: job_1450565638170_12821, Ident:
(org.apache.hadoop.mapreduce.security.token.JobTokenIdentifier@14e50583)
2015-12-31 02:17:00,126 INFO [main] org.apache.hadoop.mapred.YarnChild: Sleeping for 0ms before retrying again. Got null now.
2015-12-31 02:17:00,562 INFO [main] org.apache.hadoop.mapred.YarnChild: mapreduce.cluster.local.dir for child:
/hd_data/disk1/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk2/hadoop/yarn/local/usercach
e/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk3/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_145056563
8170_12821,/hd_data/disk4/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk5/hadoop/yarn/loc
al/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk6/hadoop/yarn/local/usercache/hsgctcrp/appcache/applicatio
n_1450565638170_12821,/hd_data/disk7/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk8/hado
op/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk9/hadoop/yarn/local/usercache/hsgctcrp/appcache
/application_1450565638170_12821,/hd_data/disk10/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_dat
a/disk11/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk12/hadoop/yarn/local/usercache/hsg
ctcrp/appcache/application_1450565638170_12821,/hd_data/disk13/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170
_12821,/hd_data/disk14/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk15/hadoop/yarn/local
/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk16/hadoop/yarn/local/usercache/hsgctcrp/appcache/application
_1450565638170_12821,/hd_data/disk17/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk18/had
oop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk19/hadoop/yarn/local/usercache/hsgctcrp/appcac
he/application_1450565638170_12821,/hd_data/disk20/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_d
ata/disk21/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk22/hadoop/yarn/local/usercache/h
sgctcrp/appcache/application_1450565638170_12821,/hd_data/disk23/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_14505656381
70_12821,/hd_data/disk24/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821
2015-12-31 02:17:01,275 INFO [main] org.apache.hadoop.conf.Configuration.deprecation: session.id is deprecated. Instead, use
dfs.metrics.session-id
2015-12-31 02:17:01,787 INFO [main] org.apache.hadoop.mapred.Task:  Using ResourceCalculatorProcessTree : [ ]
2015-12-31 02:17:02,122 INFO [main] org.apache.hadoop.mapred.MapTask: Processing split:
Paths:/EDMBD/PROJECTS/GCT/OCODS/CLAIMS_XML/BIX_ODS/XML_DELTA_FEED/UELEMENTARY_CLAIM/part-m-
```

©Simplilearn. All rights reserved.

simplilearn

application_log - Notepad

File Edit Format View Help

```
2015-12-31 02:16:59,929 WARN [main] org.apache.hadoop.metrics2.impl.MetricsConfig: Cannot locate configuration: tried hadoop-metrics2
-maptask.properties,hadoop-metrics2.properties
2015-12-31 02:17:00,004 INFO [main] org.apache.hadoop.metrics2.impl.MetricsSystemImpl: Scheduled snapshot period at 10 second(s).
2015-12-31 02:17:00,004 INFO [main] org.apache.hadoop.metrics2.impl.MetricsSystemImpl: MapTask metrics system started
2015-12-31 02:17:00,016 INFO [main] org.apache.hadoop.mapred.YarnChild: Executing with tokens:
2015-12-31 02:17:00,016 INFO [main] org.apache.hadoop.mapred.YarnChild: Kind: mapreduce.job, Service: job_1450565638170_12821, Ident:
(org.apache.hadoop.mapreduce.security.token.JobTokenIdentifier@14e50583)
2015-12-31 02:17:00,126 INFO [main] org.apache.hadoop.mapred.YarnChild: Sleeping for 0ms before retrying again. Got null now.
2015-12-31 02:17:00,562 INFO [main] org.apache.hadoop.mapred.YarnChild: mapreduce.cluster.local.dir for child:
/hd_data/disk1/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk2/hadoop/yarn/local/usercach
e/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk3/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_145056563
8170_12821,/hd_data/disk4/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk5/hadoop/yarn/loc
al/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk6/hadoop/yarn/local/usercache/hsgctcrp/appcache/applicatio
n_1450565638170_12821,/hd_data/disk7/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk8/hado
op/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk9/hadoop/yarn/local/usercache/hsgctcrp/appcache
/application_1450565638170_12821,/hd_data/disk10/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_dat
a/disk11/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk12/hadoop/yarn/local/usercache/hsg
ctcrp/appcache/application_1450565638170_12821,/hd_data/disk13/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170
_12821,/hd_data/disk14/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk15/hadoop/yarn/local
/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk16/hadoop/yarn/local/usercache/hsgctcrp/appcache/application
_1450565638170_12821,/hd_data/disk17/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk18/had
oop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk19/hadoop/yarn/local/usercache/hsgctcrp/appcac
he/application_1450565638170_12821,/hd_data/disk20/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_d
ata/disk21/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821,/hd_data/disk22/hadoop/yarn/local/usercache/h
sgctcrp/appcache/application_1450565638170_12821,/hd_data/disk23/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_14505656381
70_12821,/hd_data/disk24/hadoop/yarn/local/usercache/hsgctcrp/appcache/application_1450565638170_12821
2015-12-31 02:17:01,275 INFO [main] org.apache.hadoop.conf.Configuration.deprecation: session.id is deprecated. Instead, use
dfs.metrics.session-id
2015-12-31 02:17:01,787 INFO [main] org.apache.hadoop.mapred.Task:  Using ResourceCalculatorProcessTree : [ ]
2015-12-31 02:17:02,122 INFO [main] org.apache.hadoop.mapred.MapTask: Processing split:
Paths:/EDMBD/PROJECTS/GCT/OCODS/CLAIMS_XML/BIX_ODS/XML_DELTA_FEED/UELEMENTARY_CLAIM/part-m-
```

©Simplilearn. All rights reserved.

simplilearn

5 V's of Big Data

How much data is useful and meaningful

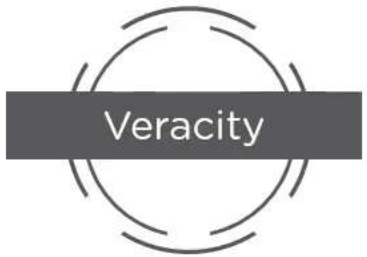
Value refers to the ability to turn your data useful for business

simplilearn

©Simplilearn. All rights reserved.

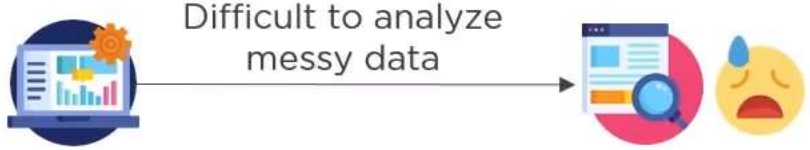
5 V's of Big Data

Trustworthiness of data in terms of quality and accuracy



Extracting loads of data is not useful if the data is messy and poor in quality

Twitter posts with abbreviations, spelling mistakes, etc.



©Simplilearn. All rights reserved.

Importance of Big data

- Cost Saving
- Time Saving
- Social media Listening

- Customer acquisition
- Marketing insights
- Innovation

It helps organizations:

- To understand Where, When and Why their customers buy
- Protect the company’s client base with improved loyalty programs
- Seizing cross-selling and upselling opportunities
- Provide targeted promotional information
- Optimize Workforce planning and operations
- Improve inefficiencies in the company’s supply chain
- Predict market trends
- Predict future needs
- Make companies more innovative and competitive
- It helps companies to discover new sources of revenue

Challenges of Big Data

1 Storing huge volume of data

Data is growing at a rapid rate

Unstructured data cannot be stored in traditional databases

simplilearn

©Simplilearn. All rights reserved.

Challenges of Big Data

2 Processing massive data

Organizations don't just store their big data – they use that data to achieve business goals

Processing and extracting insights from big data takes time

©Simplilearn. All rights reserved.

Challenges of Big Data

3 Securing data

For any organization, encryption of big data is difficult to perform

If the data is not safe, hackers can exploit it at any time

Providing user authentication for every team member can be dangerous

©Simplilearn. All rights reserved.

Use case

An e-commerce site XYZ (having 100 million users) wants to offer a gift voucher of 100$ to its top 10 customers who have spent the most in the previous year. Moreover, they want to find the buying trend of these customers so that company can suggest more items related to them.

Issues

Huge amount of unstructured data which needs to be stored, processed and analyzed.

Solution

Storage: This huge amount of data, Hadoop uses HDFS (Hadoop Distributed File System) which uses commodity hardware to form clusters and store data in a distributed fashion. It works on Write once, read many times principle.

Processing: Map Reduce paradigm is applied to data distributed over network to find the required output.

Analyze: Pig, Hive can be used to analyze the data.

Cost: Hadoop is open source so the cost is no more an issue.

Introduction to Hadoop

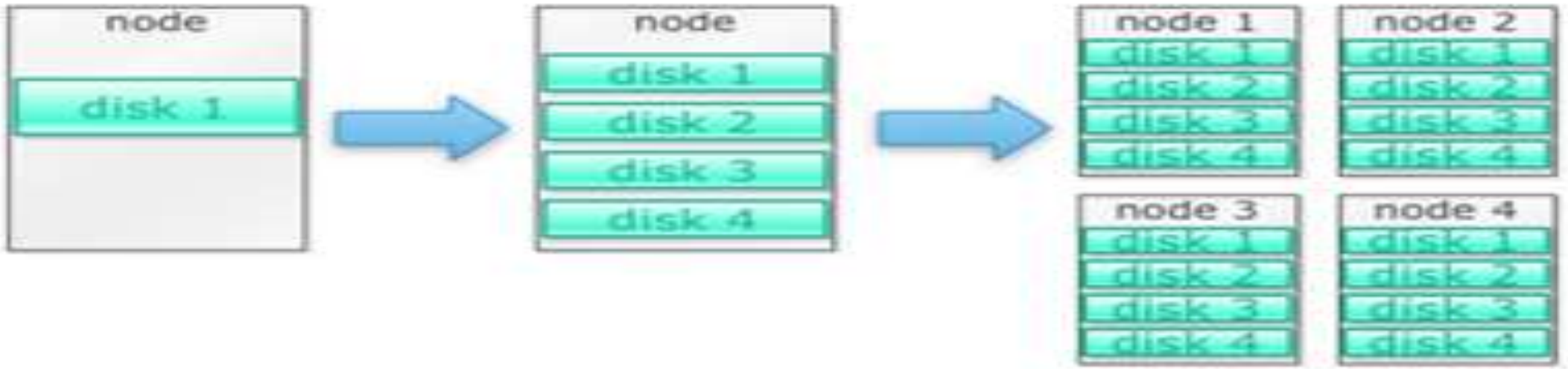
- Hadoop is an open-source software framework that is used for storing and processing large amounts of data in a distributed computing environment.
- It is designed to handle big data and is based on the MapReduce programming model, which allows for the parallel processing of large datasets.

How Hadoop Solves the Big Data Problem

Hadoop is built to run on a cluster of machines

Lets start with an example. Let's say that we need to store lots of photos. We will start with a single disk. When we exceed a single disk, we may use a few disks stacked on a machine. When we max out all the disks on a single machine, we need to get a bunch of machines, each with a bunch of disks.

This is exactly how Hadoop is built. Hadoop is designed to run on a cluster of machines from the get go.



Hadoop clusters scale horizontally

More storage and compute power can be achieved by adding more nodes to a Hadoop cluster. This eliminates the need to buy more and more powerful and expensive hardware.

Hadoop can handle unstructured/semi-structured data

Hadoop doesn't enforce a schema on the data it stores. It can handle arbitrary text and binary data. So Hadoop can digest any unstructured data easily.

Hadoop clusters provides storage and computing

We saw how having separate storage and processing clusters is not the best fit for big data. Hadoop clusters, however, provide storage and distributed computing all in one.

Hadoop provides storage for big data at reasonable cost

Storing big data using traditional storage can be expensive. Hadoop is built around commodity hardware, so it can provide fairly large storage for a reasonable cost. Hadoop has been used in the field at petabyte scale.
One study by Cloudera suggested that enterprises usually spend around $25,000 to $50,000 per terabyte per year. With Hadoop, this cost drops to a few thousand dollars per terabyte per year. As hardware gets cheaper and cheaper, this cost continues to drop.

Hadoop allows for the capture of new or more data

Sometimes organizations don't capture a type of data because it was too cost prohibitive to store it. Since Hadoop provides storage at reasonable cost, this type of data can be captured and stored.
One example would be website click logs. Because the volume of these logs can be very high, not many organizations captured these. Now with Hadoop it is possible to capture and store the logs.

With Hadoop, you can store data longer

To manage the volume of data stored, companies periodically purge older data. For example, only logs for the last three months could be stored, while older logs were deleted. With Hadoop it is possible to store the historical data longer. This allows new analytics to be done on older historical data.
For example, take click logs from a website. A few years ago, these logs were stored for a brief period of time to calculate statistics like popular pages. Now with Hadoop, it is viable to store these click logs for longer period of time.

Hadoop provides scalable analytics

There is no point in storing all this data if we can't analyze them. Hadoop not only provides distributed storage, but also distributed processing as well, which means we can crunch a large volume of data in parallel. The compute framework of Hadoop is called MapReduce. MapReduce has been proven to the scale of petabytes.

Hadoop provides rich analytics

Native MapReduce supports Java as a primary programming language. Other languages like Ruby, Python and R can be used as well.

Of course, writing custom MapReduce code is not the only way to analyze data in Hadoop. Higher-level Map Reduce is available. For example, a tool named Pig takes English like data flow language and translates them into MapReduce. Another tool, Hive, takes SQL queries and runs them using MapReduce.

Business intelligence (BI) tools can provide even higher level of analysis. There are tools for this type of analysis as well.

Comparison HADOOP with RDBMS

RDMS (Relational Database Management System):

- RDBMS is an information management system, which is based on a data model.
- In RDBMS tables are used for information storage.
- Each row of the table represents a record and column represents an attribute of data.
- Organization of data and their manipulation processes are different in RDBMS from other databases.
- RDBMS ensures ACID (atomicity, consistency, integrity, durability) properties required for designing a database.
- The purpose of RDBMS is to store, manage, and retrieve data as quickly and reliably as possible.

Hadoop:

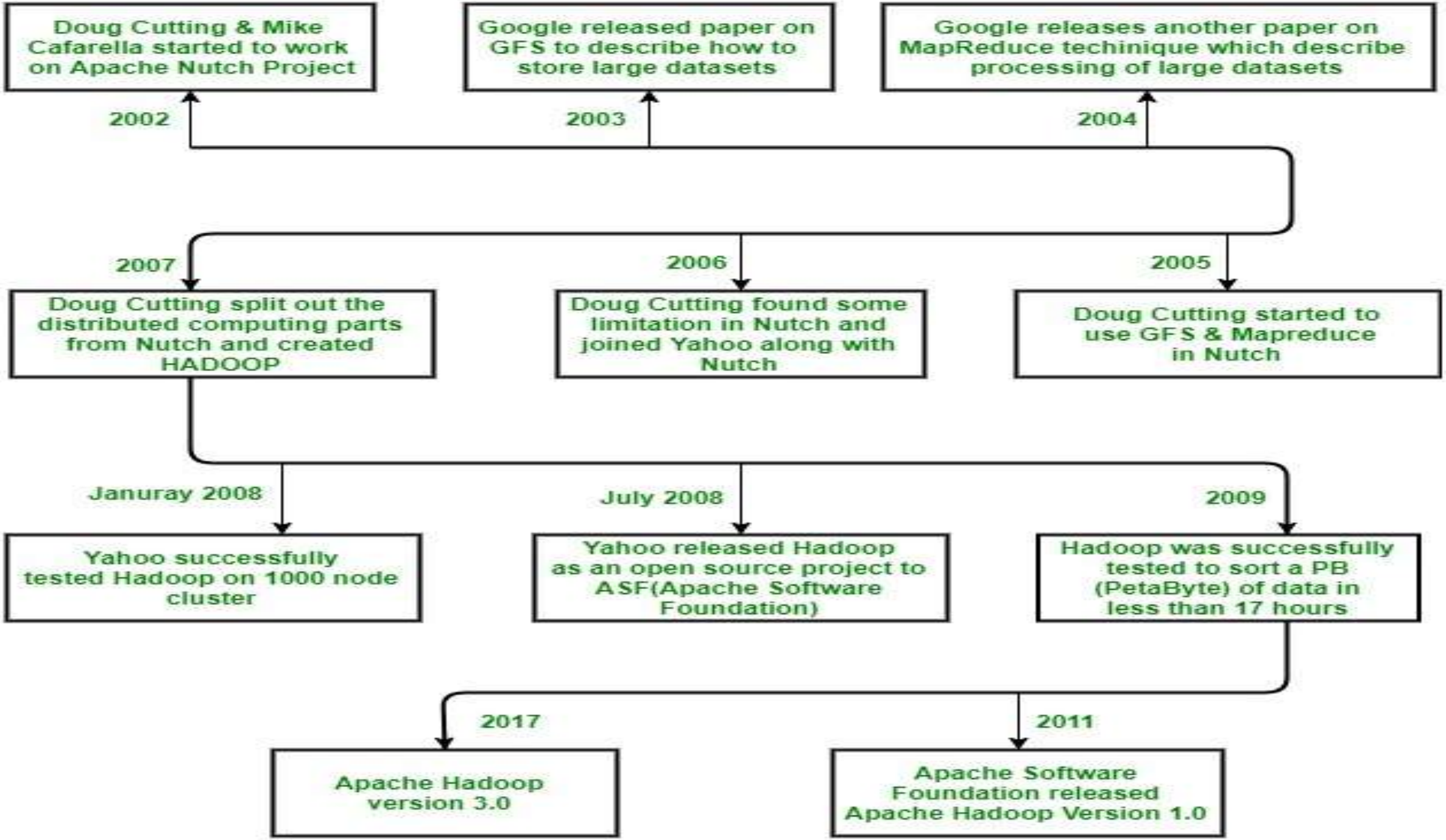
- It is an open-source software framework used for storing data and running applications on a group of commodity hardware.
- It has large storage capacity and high processing power.
- It can manage multiple concurrent processes at the same time.
- It is used in predictive analysis, data mining and machine learning.
- It can handle both structured and unstructured form of data.
- It is more flexible in storing, processing, and managing data than traditional RDBMS.
- Unlike traditional systems, Hadoop enables multiple analytical processes on the same data at the same time.
- It supports scalability very flexibly.

Comparison / Differences between RDBMS and Hadoop:

S.No.	RDBMS	Hadoop
1.	Traditional row-column based databases, basically used for data storage, manipulation and retrieval.	An open-source software used for storing data and running applications or processes concurrently.
2.	In this structured data is mostly processed.	In this both structured and unstructured data is processed.
3.	It is best suited for OLTP environment.	It is best suited for BIG data.
4.	It is less scalable than Hadoop.	It is highly scalable.
5.	Data normalization is required in RDBMS.	Data normalization is not required in Hadoop.

S.No.	RDBMS	Hadoop
6.	It stores transformed and aggregated data.	It stores huge volume of data.
7.	It has no latency in response.	It has some latency in response.
8.	The data schema of RDBMS is static type.	The data schema of Hadoop is dynamic type.
9.	High data integrity available.	Low data integrity available than RDBMS.
10.	Cost is applicable for licensed software.	Free of cost, as it is an open source software.

Brief History of HADOOP


Doug Cutting & Mike Cafarella started to work on Apache Nutch Project
Google released paper on GFS to describe how to store large datasets
Google releases another paper on MapReduce techinique which describe processing of large datasets
2002
2003
2004
2007
2006
2005
Doug Cutting split out the distributed computing parts from Nutch and created HADOOP
Doug Cutting found some limitation in Nutch and joined Yahoo along with Nutch
Doug Cutting started to use GFS & Mapreduce in Nutch
Januray 2008
July 2008
2009
Yahoo successfully tested Hadoop on 1000 node cluster
Yahoo released Hadoop as an open source project to ASF(Apache Software Foundation)
Hadoop was successfully tested to sort a PB (PetaByte) of data in less than 17 hours
2017
2011
Apache Hadoop version 3.0
Apache Software Foundation released Apache Hadoop Version 1.0

Hadoop as a Solution

simplilearn

Hadoop as a Solution

Hadoop is an open-source framework for storing data and running applications on clusters of commodity hardware

2 main components

hadoop HDFS

Solves the issue of storing rapidly increasing data

hadoop Map Reduce

Helps you to process and analyze big data faster

©Simplilearn. All rights reserved.

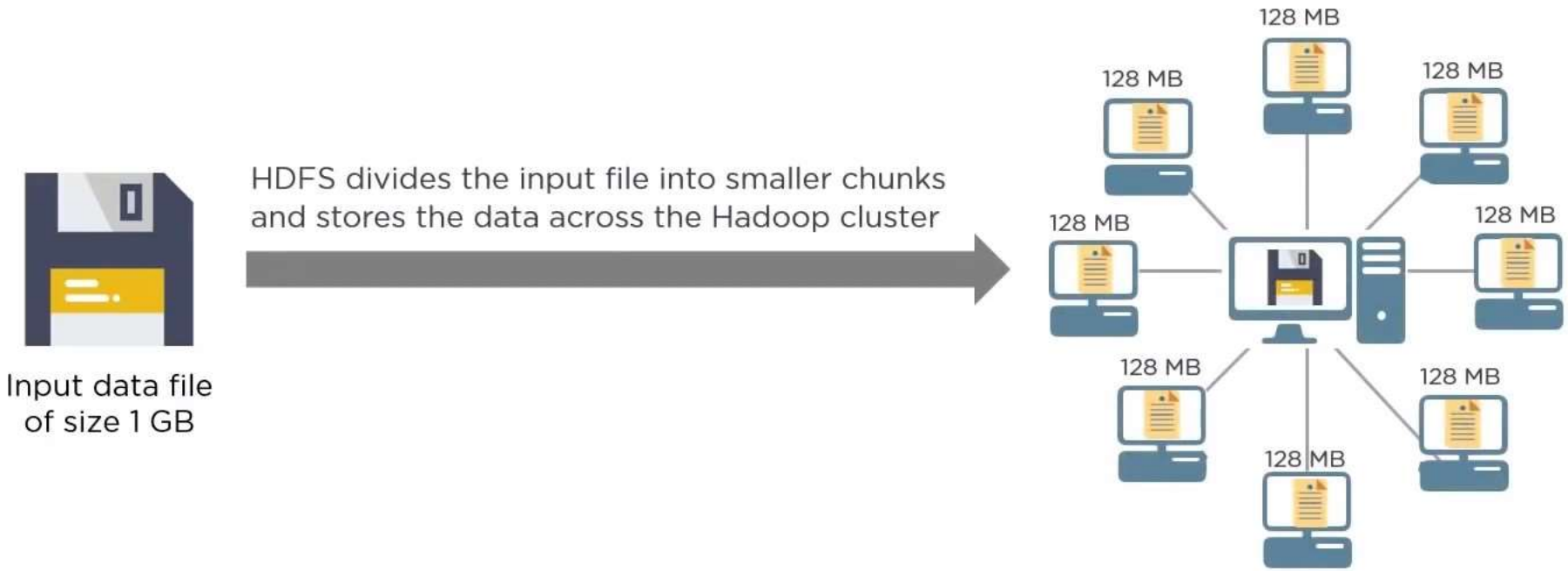
Hadoop as a Solution

Hadoop Distributed File System (HDFS) is the storage unit of Hadoop that stores big data in multiple server machines instead of a central server

©Simplilearn. All rights reserved.

Hadoop as a Solution

Hadoop Distributed File System (HDFS) is the storage unit of Hadoop that stores big data in multiple server machines instead of a central server



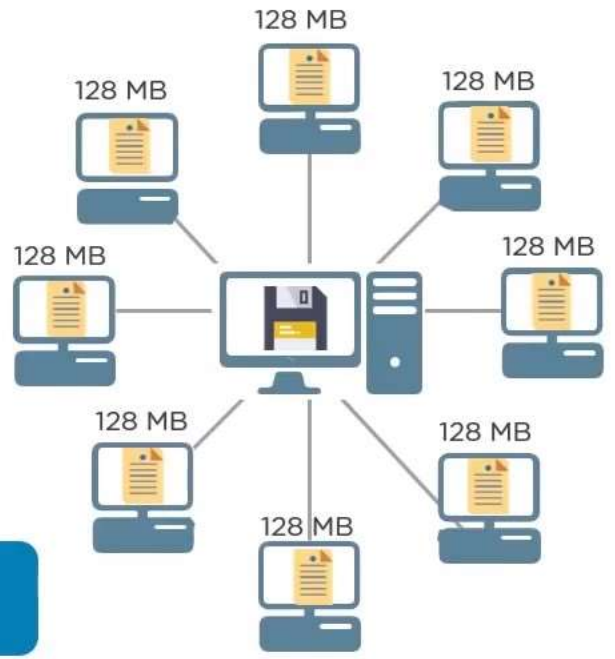
©Simplilearn. All rights reserved.

Hadoop as a Solution

Hadoop Distributed File System (HDFS) is the storage unit of Hadoop that stores big data in multiple server machines instead of a central server

Input data file of size 1 GB

HDFS divides the input file into smaller chunks and stores the data across the Hadoop cluster

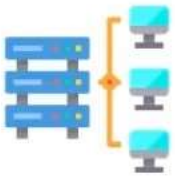


Files are split by Hadoop framework into blocks of default block size of 128 mb

The machines in the Hadoop cluster have disks that store these blocks

©Simplilearn. All rights reserved.

Hadoop as a Solution



Distributed Computing

Distributed computing allows you to perform distributed parallel processing on large volumes of data quickly and efficiently

Consider the following scenario

1 machine
4 I/O channels
Each channel – 100 MB/s

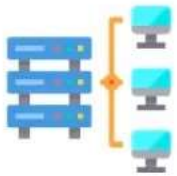
Data File
of 1 TB

It will take 43 minutes for one machine to process 1 TB of data

simplilearn

©Simplilearn. All rights reserved.

Hadoop as a Solution



Distributed Computing

Distributed computing allows you to perform distributed parallel processing on large volumes of data quickly and efficiently

Consider the following scenario

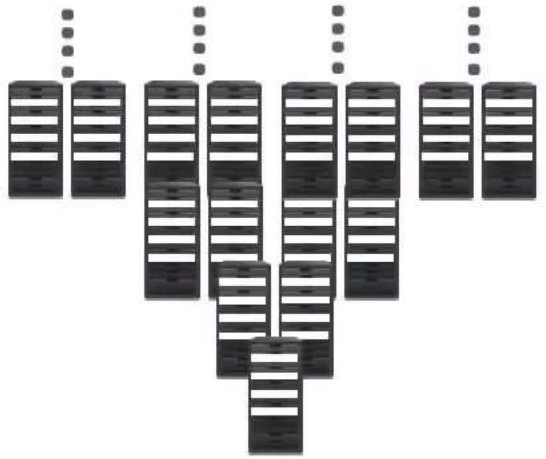
1 machine
4 I/O channels
Each channel – 100 MB/s

Data File
of 1 TB

It will take 43 minutes for one machine to process 1 TB of data

Using Distributed Computing

100 machine
4 I/O channels
Each channel – 100 MB/s



Data File
of 1 TB

simplilearn

©Simplilearn. All rights reserved.

Hadoop as a Solution

Hortonworks distribution of Hadoop has resources that provides **security** to your big data

APACHE KNOX REST API that supports monitoring, authorization management, auditing and policy enforcement on Hadoop clusters

Apache Ranger is a framework to enable, monitor and manage comprehensive data security across the Hadoop platform

Provides end-to-end encryption that protects data while it is at rest within the Hadoop cluster and in motion across the network

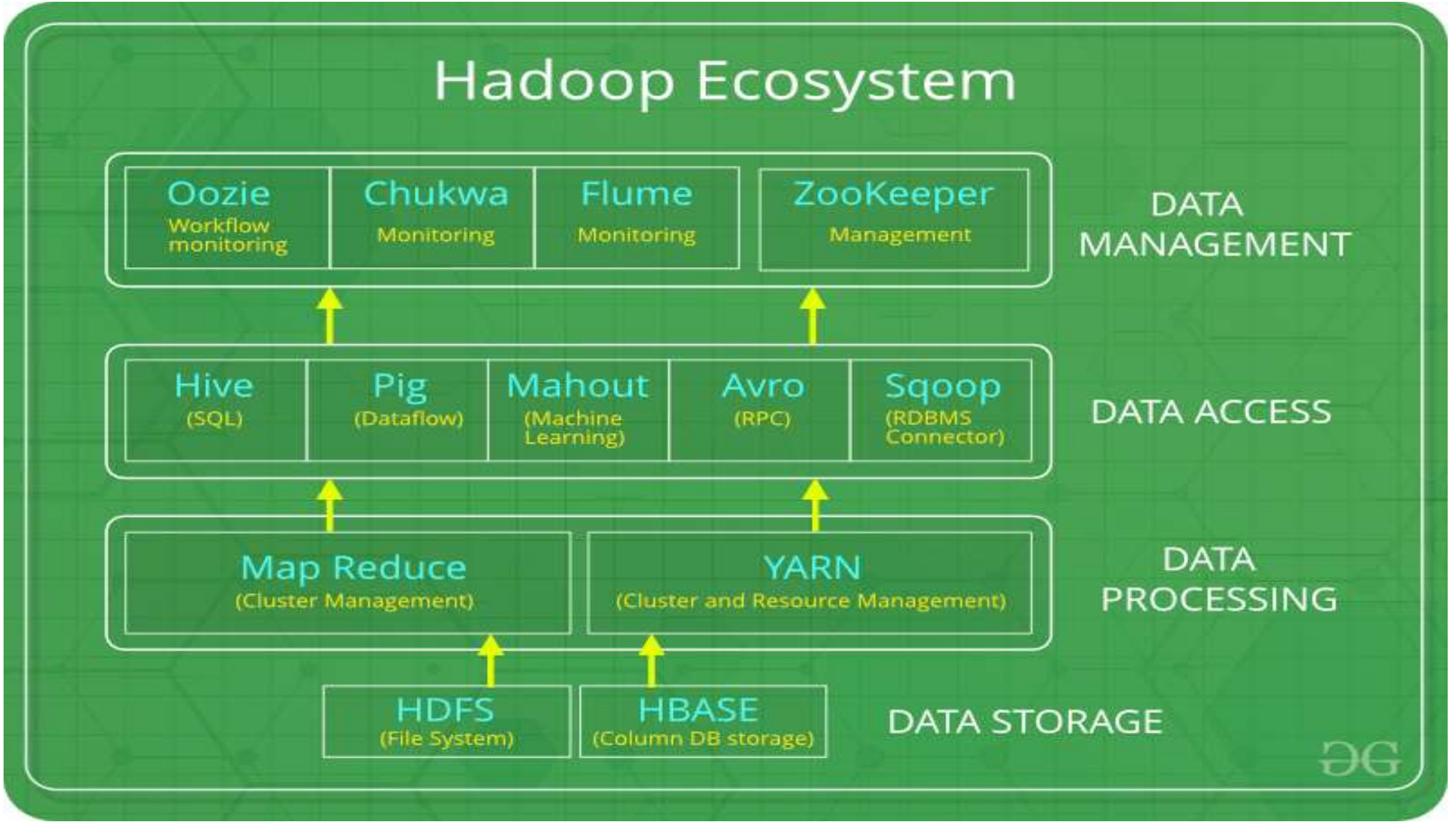
©Simplilearn. All rights reserved.

Hadoop Ecosystem

Introduction:

- *Hadoop Ecosystem* is a platform or a suite which provides various services to solve the big data problems.
- It includes Apache projects and various commercial tools and solutions.

- There are *four major elements of Hadoop* i.e. **HDFS, MapReduce, YARN, and Hadoop Common**.
- Most of the tools or solutions are used to supplement or support these major elements.
- All these tools work collectively to provide services such as absorption, analysis, storage and maintenance of data etc.


Hadoop Ecosystem
Oozie
Workflow monitoring
Chukwa
Monitoring
Flume
Monitoring
ZooKeeper
Management
DATA MANAGEMENT
Hive
(SQL)
Pig
(Dataflow)
Mahout
(Machine Learning)
Avro
(RPC)
Sqoop
(RDBMS Connector)
DATA ACCESS
Map Reduce
(Cluster Management)
YARN
(Cluster and Resource Management)
DATA PROCESSING
HDFS
(File System)
HBASE
(Column DB storage)
DATA STORAGE

Following are the components that collectively form a Hadoop ecosystem:

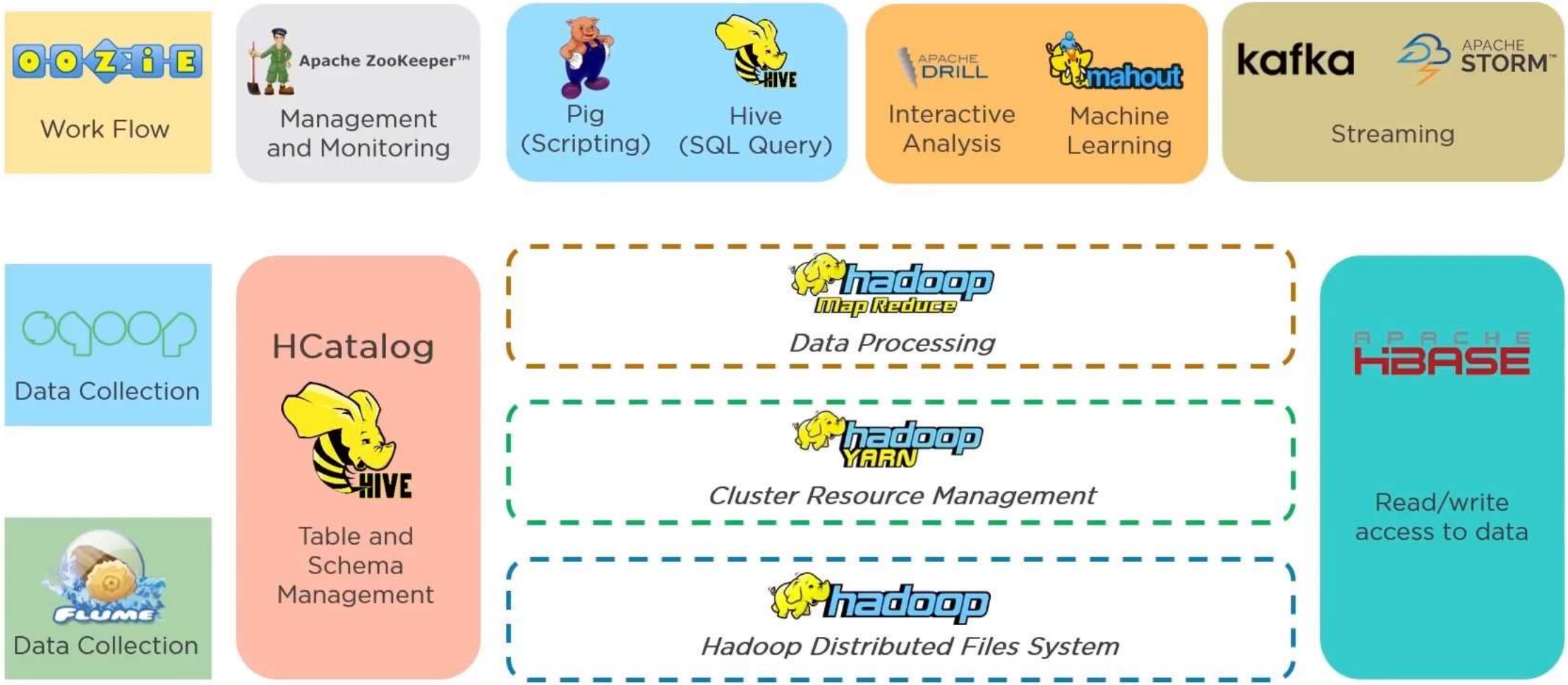
- **HDFS** : Hadoop Distributed File System
- **YARN** : Yet Another Resource Negotiator
- **MapReduce** : Programming based Data Processing
- **Spark** : In-Memory data processing
- **PIG, HIVE:** Query based processing of data services
- **Hbase** : NoSQL Database
- **Mahout, Spark MLLib** : Machine Learning algorithm libraries
- **Solar, Lucene** : Searching and Indexing
- **Zookeeper** : Managing cluster
- **Oozie** : Job Scheduling

Note:

Apart from the above-mentioned components, there are many other components too that are part of the Hadoop ecosystem.

All these toolkits or components revolve around one term i.e. *Data*. That’s the beauty of Hadoop that it revolves around data and hence making its synthesis easier.

Hadoop Ecosystem



©Simplilearn. All rights reserved.

HDFS:

- **HDFS** is the primary or major component of Hadoop ecosystem and is responsible for storing large data sets of structured or unstructured data across various nodes and thereby maintaining the metadata in the form of log files.

- **HDFS** consists of two core components i.e.
  1. Name node
  2. Data Node

- Name Node is the prime node which contains metadata (data about data) requiring comparatively fewer resources than the data nodes that stores the actual data. These data nodes are commodity hardware in the distributed environment. Undoubtedly, making Hadoop cost effective.

- HDFS maintains all the coordination between the clusters and hardware, thus working at the heart of the system.

❑ **HDFS** consists of two core components i.e.

1. **Name node**

- ✓ NameNode works as a *Master* in a Hadoop cluster that Guides the Datanode(Slaves).
- ✓ Namenode is mainly used for storing the Metadata i.e. nothing but the data about the data.
- ✓ Meta Data can be the transaction logs that keep track of the user's activity in a Hadoop cluster.
- ✓ Meta Data can also be the name of the file, size, and the information about the location(Block number, Block ids) of Datanode that Namenode stores to find the closest DataNode for Faster Communication.
- ✓ Namenode instructs the DataNodes with the operation like delete, create, Replicate, etc.
- ✓ As our NameNode is working as a Master it should have a high RAM or Processing power in order to Maintain or Guide all the slaves in a Hadoop cluster.
- ✓ Namenode receives heartbeat signals and block reports from all the slaves i.e. DataNodes.

- **HDFS** consists of two core components i.e.

1. **Data node**

- ✓ DataNodes works as a *Slave* DataNodes are mainly utilized for storing the data in a Hadoop cluster, the number of DataNodes can be from 1 to 500 or even more than that, the more number of DataNode your Hadoop cluster has More Data can be stored.

- ✓ So it is advised that the DataNode should have High storing capacity to store a large number of file blocks.

- ✓ Datanode performs operations like creation, deletion, etc. according to the instruction provided by the NameNode.

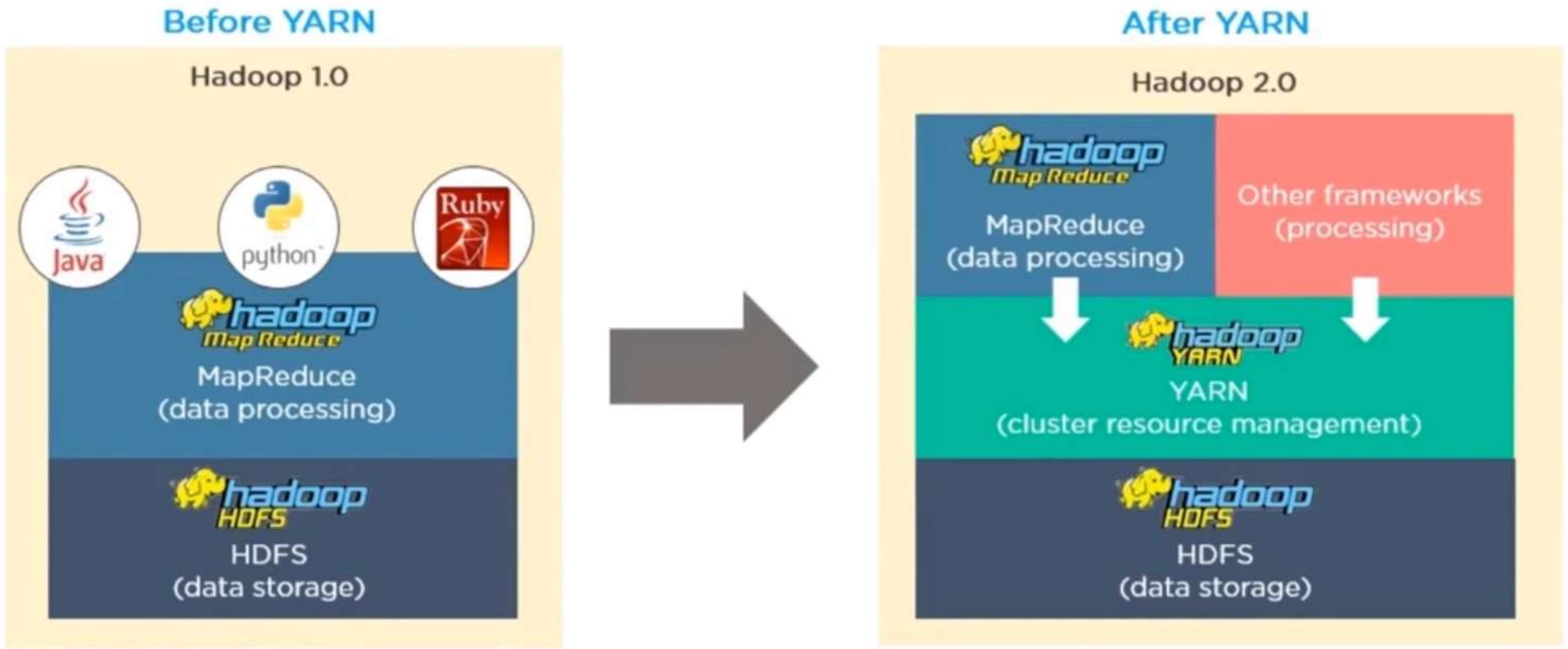
YARN:

- **Yet Another Resource Negotiator**, as the name implies, YARN is the one who helps to manage the resources across the clusters. In short, it performs scheduling and resource allocation for the Hadoop System.

- Consists of three major components i.e.
  1. Resource Manager
  2. Nodes Manager
  3. Application Manager

- Resource manager has the privilege of allocating resources for the applications in a system whereas Node managers work on the allocation of resources such as CPU, memory, bandwidth per machine and later on acknowledges the resource manager. Application manager works as an interface between the resource manager and node manager and performs negotiations as per the requirement of the two.

Need for YARN



Designed to run MapReduce jobs only and had issues in scalability, resource utilization, etc.

YARN solved those issues and users could work on multiple processing models along with MapReduce

simplilearn

©Simplilearn. All rights reserved.

MapReduce:

- By making the use of distributed and parallel algorithms, MapReduce makes it possible to carry over the processing's logic and helps to write applications which transform big data sets into a manageable one.

- **MapReduce makes the use of two functions i.e. Map() and Reduce() whose task is:**

    1. ***Map()*** performs sorting and filtering of data and thereby organizing them in the form of group. Map generates a key-value pair based result which is later on processed by the Reduce() method.

    2. ***Reduce(),*** as the name suggests does the summarization by aggregating the mapped data. In simple, Reduce() takes the output generated by Map() as input and combines those tuples into smaller set of tuples.

PIG:

Pig was basically developed by Yahoo which works on a pig Latin language, which is Query based language similar to SQL.

- It is a platform for structuring the data flow, processing and analyzing huge data sets.
- Pig does the work of executing commands and in the background, all the activities of MapReduce are taken care of. After the processing, pig stores the result in HDFS.
- Pig Latin language is specially designed for this framework which runs on Pig Runtime. Just the way Java runs on the JVM.
- Pig helps to achieve ease of programming and optimization and hence is a major segment of the Hadoop Ecosystem.

HIVE:

- With the help of SQL methodology and interface, HIVE performs reading and writing of large data sets. However, its query language is called as HQL (Hive Query Language).

- It is highly scalable as it allows real-time processing and batch processing both. Also, all the SQL datatypes are supported by Hive thus, making the query processing easier.

- Similar to the Query Processing frameworks, HIVE too comes with two components: *JDBC Drivers* and *HIVE Command Line*.

- JDBC, along with ODBC drivers work on establishing the data storage permissions and connection whereas HIVE Command line helps in the processing of queries.

Mahout:

- Mahout, allows Machine Learnability to a system or application. Machine Learning, as the name suggests helps the system to develop itself based on some patterns, user/environmental interaction or on the basis of algorithms.

- It provides various libraries or functionalities such as collaborative filtering, clustering, and classification which are nothing but concepts of Machine learning. It allows invoking algorithms as per our need with the help of its own libraries.

ApacheSpark:

- It’s a platform that handles all the process consumptive tasks like batch processing, interactive or iterative real-time processing, graph conversions, and visualization, etc.

- It consumes in memory resources hence, thus being faster than the prior in terms of optimization.

- Spark is best suited for real-time data whereas Hadoop is best suited for structured data or batch processing, hence both are used in most of the companies interchangeably.

ApacheHBase:

- It’s a NoSQL database which supports all kinds of data and thus capable of handling anything of Hadoop Database. It provides capabilities of Google’s BigTable, thus able to work on Big Data sets effectively.

- At times where we need to search or retrieve the occurrences of something small in a huge database, the request must be processed within a short quick span of time. At such times, HBase comes handy as it gives us a tolerant way of storing limited data

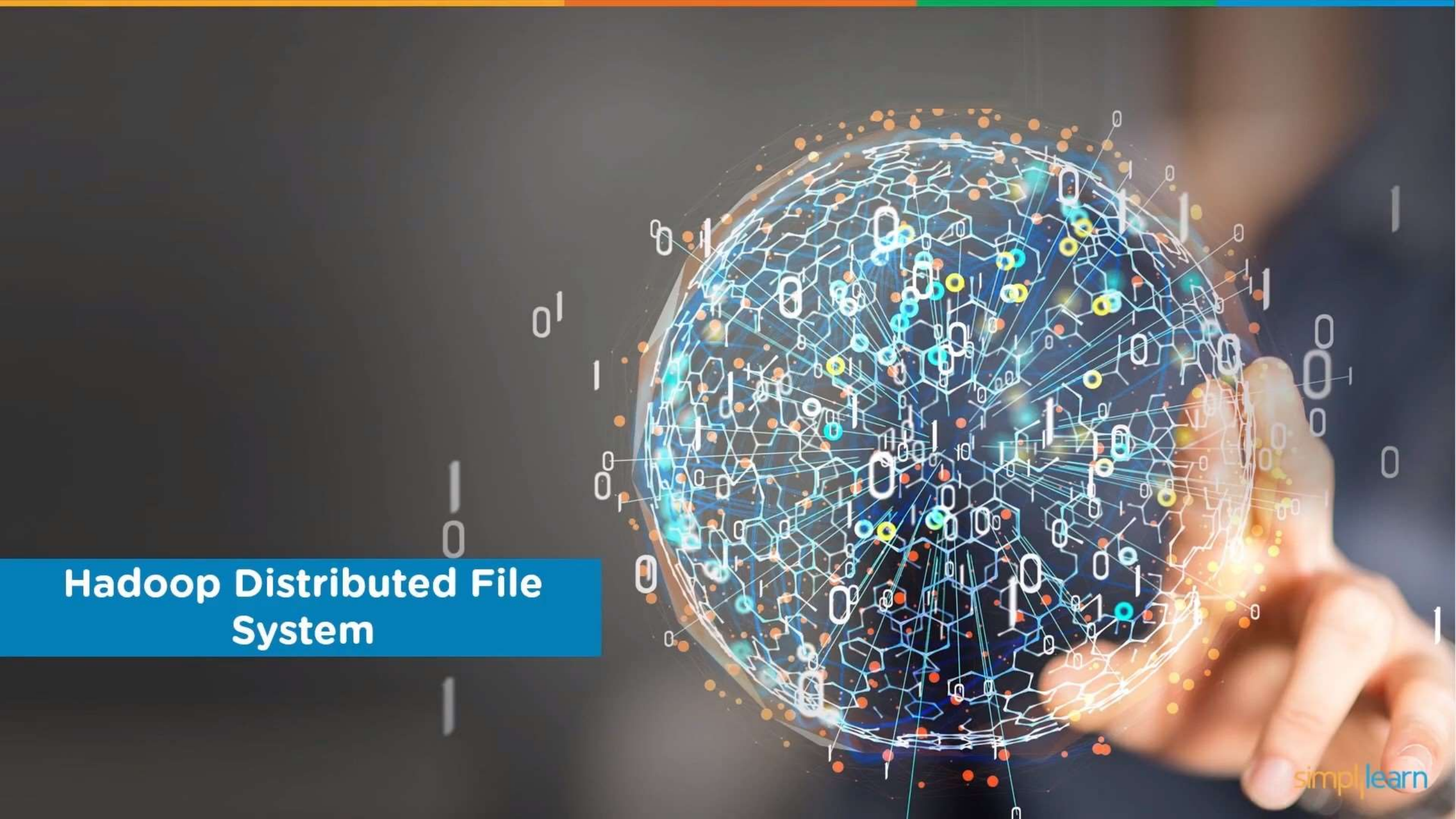
Other Components: Apart from all of these, there are some other components too that carry out a huge task in order to make Hadoop capable of processing large datasets. They are as follows:

- **Solr, Lucene:** These are the two services that perform the task of searching and indexing with the help of some java libraries, especially Lucene is based on Java which allows spell check mechanism, as well. However, Lucene is driven by Solr.

- **Zookeeper:** There was a huge issue of management of coordination and synchronization among the resources or the components of Hadoop which resulted in inconsistency, often. Zookeeper overcame all the problems by performing synchronization, inter-component based communication, grouping, and maintenance.

- **Oozie:** Oozie simply performs the task of a scheduler, thus scheduling jobs and binding them together as a single unit. There is two kinds of jobs .i.e Oozie workflow and Oozie coordinator jobs. Oozie workflow is the jobs that need to be executed in a sequentially ordered manner whereas Oozie Coordinator jobs are those that are triggered when some data or external stimulus is given to it.

Hadoop Distributed File System

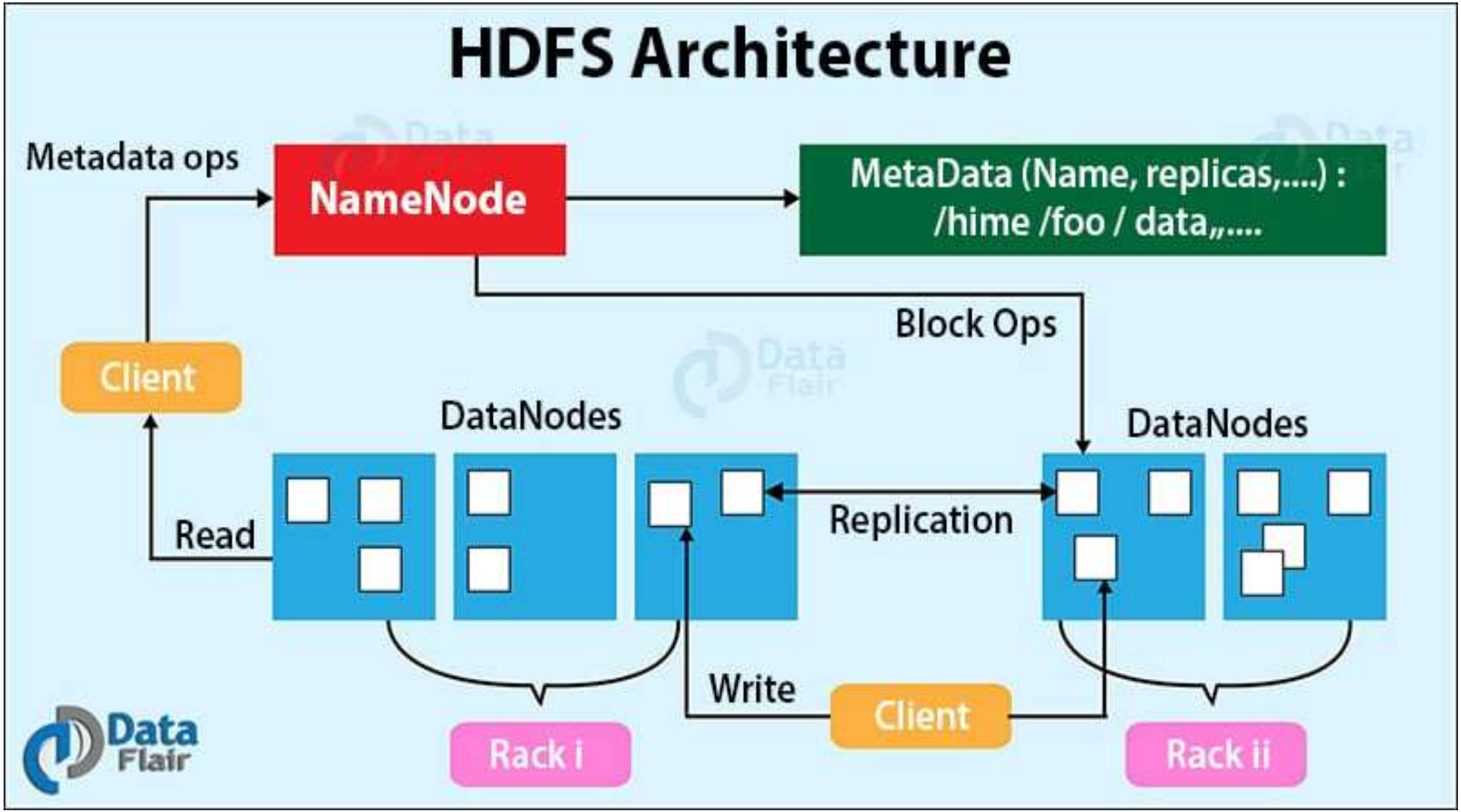


NameNode

Hadoop HDFS Architecture

- Hadoop Distributed File System follows the **master-slave architecture**.
- Each cluster comprises a **single master node** and **multiple slave nodes**. Internally the files get divided into one or more **blocks**, and each block is stored on different slave machines depending on the **replication factor**
- The master node stores and manages the file system namespace, that is information about blocks of files like block locations, permissions, etc. The slave nodes store data blocks of files.
- The Master node is the NameNode and DataNodes are the slave nodes.
- Let’s discuss each of the nodes in the Hadoop HDFS Architecture in detail.

HDFS Architecture



What is HDFS NameNode?

NameNode is the centerpiece of the Hadoop Distributed File System. It maintains and manages the **file system namespace** and provides the right access permission to the clients.

The NameNode stores information about blocks locations, permissions, etc. on the local disk in the form of two files:

- **Fsimage:** Fsimage stands for File System image. It contains the complete namespace of the Hadoop file system since the NameNode creation.
- **Edit log:** It contains all the recent changes performed to the file system namespace to the most recent Fsimage.

Functions of HDFS NameNode

1. It executes the file system namespace operations like opening, renaming, and closing files and directories.
2. NameNode manages and maintains the DataNodes.
3. It determines the mapping of blocks of a file to DataNodes.
4. NameNode records each change made to the file system namespace.
5. It keeps the locations of each block of a file.
6. NameNode takes care of the replication factor of all the blocks.
7. NameNode receives heartbeat and block reports from all DataNodes that ensure DataNode is alive.
8. If the DataNode fails, the NameNode chooses new DataNodes for new replicas.

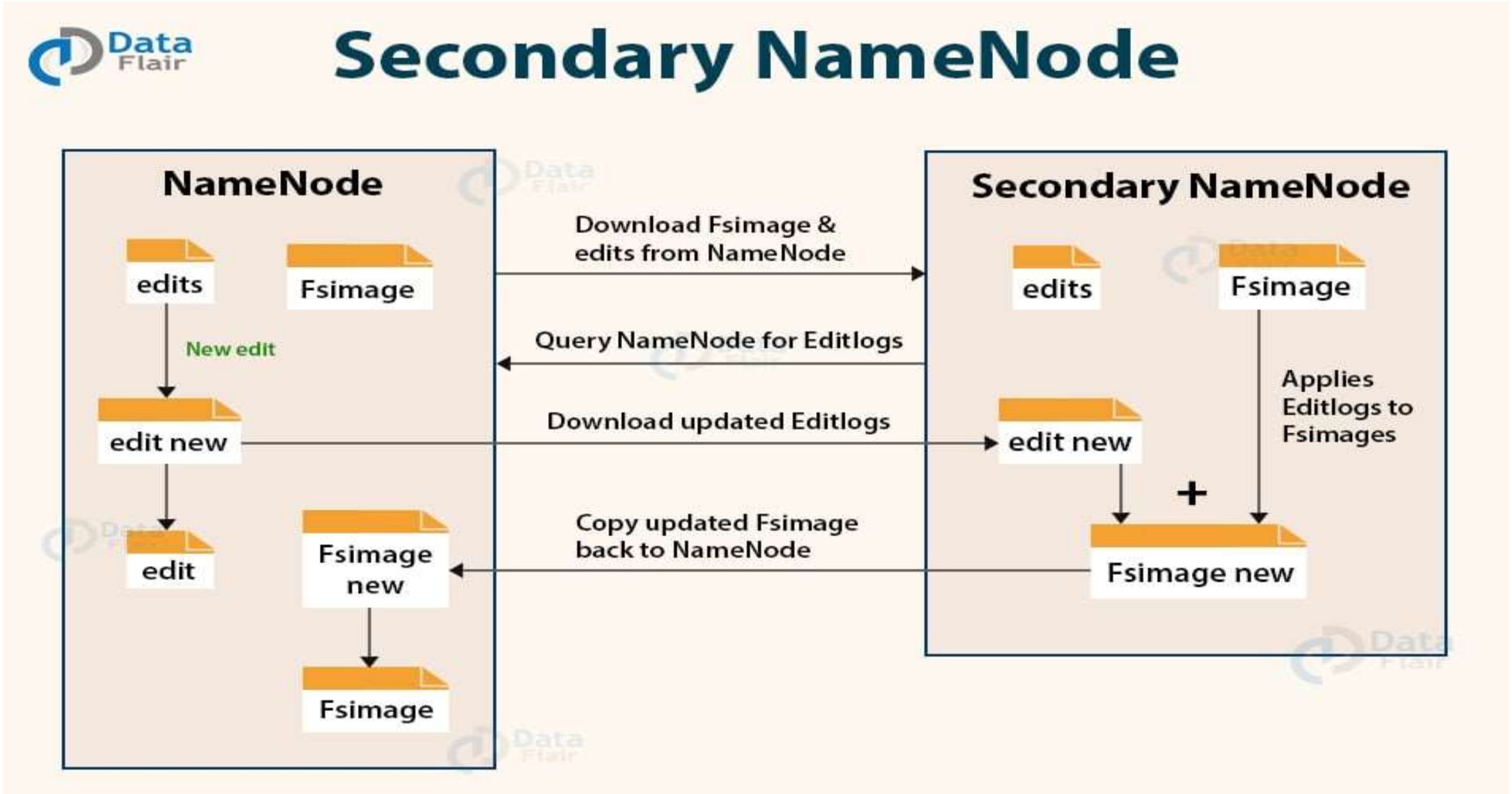
What is HDFS DataNode?

DataNodes are the slave nodes in Hadoop HDFS. DataNodes are **inexpensive commodity hardware**. They store blocks of a file.

Functions of DataNode

1. DataNode is responsible for serving the client read/write requests.
2. Based on the instruction from the NameNode, DataNodes performs block creation, replication, and deletion.
3. DataNodes send a heartbeat to NameNode to report the health of HDFS.
4. DataNodes also sends block reports to NameNode to report the list of blocks it contains.

What is Secondary NameNode?



- Apart from DataNode and NameNode, there is another daemon called the **secondary NameNode**.

- Secondary NameNode works as a helper node to primary NameNode but doesn't replace primary NameNode.

- When the NameNode starts, the NameNode merges the Fsimage and edit logs file to restore the current file system namespace.

- Since the NameNode runs continuously for a long time without any restart, the size of edit logs becomes too large.
- This will result in a long restart time for NameNode.

- Secondary NameNode solves this issue.

- Secondary NameNode downloads the Fsimage file and edit logs file from NameNode.

It periodically applies edit logs to Fsimage and refreshes the edit logs. The updated Fsimage is then sent to the NameNode so that NameNode doesn’t have to re-apply the edit log records during its restart. This keeps the edit log size small and reduces the NameNode restart time.

If the NameNode fails, the last save Fsimage on the secondary NameNode can be used to recover file system metadata. The secondary NameNode performs regular checkpoints in HDFS.

Working with HDFS Commands

HDFS is the primary or major component of the Hadoop ecosystem which is responsible for storing large data sets of structured or unstructured data across various nodes and thereby maintaining the metadata in the form of log files. To use the HDFS commands, first you need to start the Hadoop services using the following command:

sbin/start-all.sh

To check the Hadoop services are up and running use the following command:

jps

```
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ jps
2546 SecondaryNameNode
2404 DataNode
2295 NameNode
2760 ResourceManager
2874 NodeManager
4251 Jps
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$
```

Commands:

1. **ls:** This command is used to list all the files. Use *lsr* for recursive approach. It is useful when we want a hierarchy of a folder.

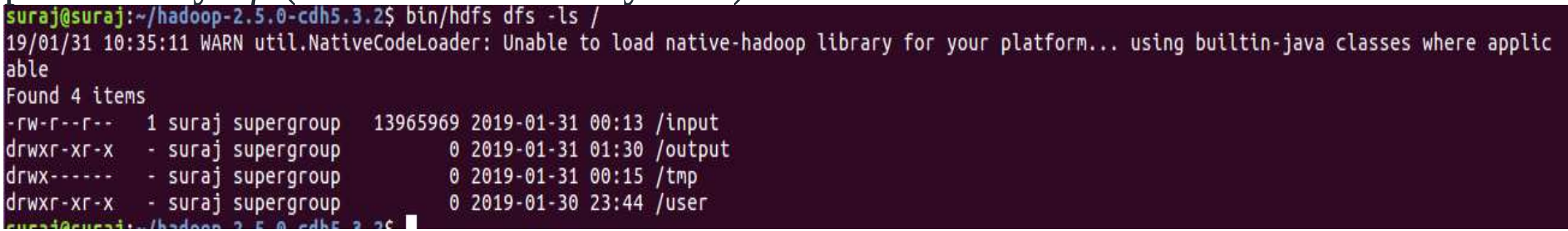
   Syntax:

   bin/hdfs dfs -ls <path>

   Example:

   bin/hdfs dfs -ls /

   It will print all the directories present in HDFS. bin directory contains executables so, *bin/hdfs* means we want the executables of hdfs particularly *dfs*(Distributed File System) commands.



mkdir: To create a directory. In Hadoop *dfs* there is no home directory by default. So let's first create it.

Syntax:

bin/hdfs dfs -mkdir <folder name>

creating home directory:

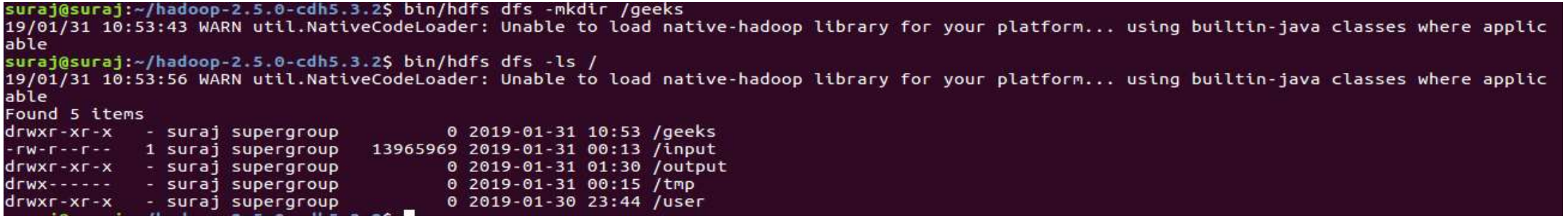
hdfs/bin -mkdir /user

hdfs/bin -mkdir /user/username -> write the username of your computer

Example:

bin/hdfs dfs -mkdir /geeks => '/' means absolute path

bin/hdfs dfs -mkdir geeks2 => Relative path -> the folder will be created relative to the home directory.



1. **touchz**: It creates an empty file.

   Syntax:

   bin/hdfs dfs -touchz <file_path>

   Example:

   bin/hdfs dfs -touchz /geeks/myfile.txt

```
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -touchz /geeks/myfile.txt
19/01/31 11:10:31 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -lsr /geeks
lsr: DEPRECATED: Please use 'ls -R' instead.
19/01/31 11:10:48 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
-rw-r--r--   1 suraj supergroup          0 2019-01-31 11:10 /geeks/myfile.txt
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$
```

1. **copyFromLocal (or) put:** To copy files/folders from local file system to hdfs store. This is the most important command. Local filesystem means the files present on the OS.
   Syntax:
   bin/hdfs dfs -copyFromLocal <local file path> <dest(present on hdfs)>

   Example: Let's suppose we have a file *AI.txt* on Desktop which we want to copy to folder *geeks* present on hdfs.
   bin/hdfs dfs -copyFromLocal ../Desktop/AI.txt /geeks

   (OR)

   bin/hdfs dfs -put ../Desktop/AI.txt /geeks

```
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -put ../Desktop/AI.txt /geeks
19/01/31 11:31:02 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -lsr /geeks
lsr: DEPRECATED: Please use 'ls -R' instead.
19/01/31 11:31:21 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
-rw-r--r--   1 suraj supergroup        205 2019-01-31 11:31 /geeks/AI.txt
-rw-r--r--   1 suraj supergroup          0 2019-01-31 11:10 /geeks/myfile.txt
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$
```

1. **cat:** To print file contents.

   Syntax:

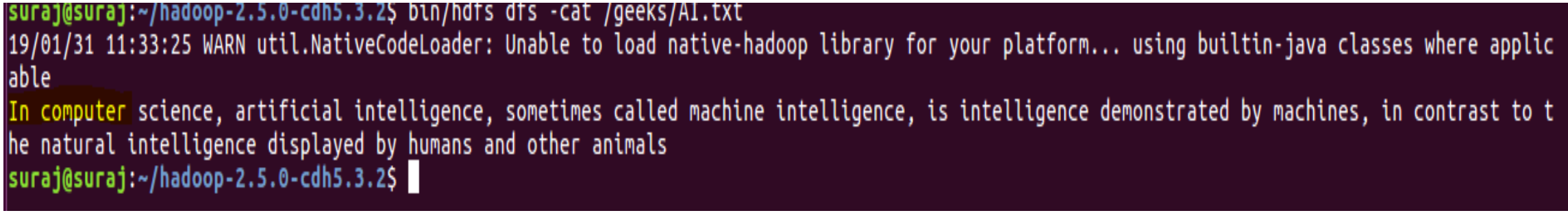
   bin/hdfs dfs -cat <path>

   Example:

   // print the content of AI.txt present

   // inside geeks folder.

   bin/hdfs dfs -cat /geeks/AI.txt ->



1. **copyToLocal (or) get:** To copy files/folders from hdfs store to local file system.
   Syntax:
   bin/hdfs dfs -copyToLocal <<srcfile(on hdfs)> <local file dest>

   Example:
   bin/hdfs dfs -copyToLocal /geeks ../Desktop/hero

   (OR)

   bin/hdfs dfs -get /geeks/myfile.txt ../Desktop/hero

   myfile.txt from *geeks* folder will be copied to folder *hero* present on *Desktop*.

```
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -get /geeks/myfile.txt  ../Desktop/hero
19/01/31 11:43:34 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ ls ../Desktop/hero
myfile.txt
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$
```

   Note: Observe that we don't write *bin/hdfs* while checking the things present on local filesystem.

1. **moveFromLocal:** This command will move file from local to hdfs.

   Syntax:

   bin/hdfs dfs -moveFromLocal <local src> <dest(on hdfs)>

   Example:

   bin/hdfs dfs -moveFromLocal ../Desktop/cutAndPaste.txt /geeks

```
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -moveFromLocal ../Desktop/cutAndPaste.txt /geeks
19/01/31 12:38:38 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -ls /geeks
19/01/31 12:38:56 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
Found 3 items
-rw-r--r--   1 suraj supergroup        205 2019-01-31 11:31 /geeks/AI.txt
-rw-r--r--   1 suraj supergroup         96 2019-01-31 12:38 /geeks/cutAndPaste.txt
-rw-r--r--   1 suraj supergroup          0 2019-01-31 11:10 /geeks/myfile.txt
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ ls ../Desktop
AFINN-111.txt  Deploy_hadoop_on_a_single_node_cluster_v02 (1).pdf  hero                  sentiment.jar
AI.txt         FlumeData.1440939532959                             json-simple-1.1.1.jar  worldBank
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$
```

1. **cp:** This command is used to copy files within hdfs. Lets copy folder *geeks* to *geeks_copied*.

   Syntax:

   bin/hdfs dfs -cp <src(on hdfs)> <dest(on hdfs)>

   Example:

   bin/hdfs -cp /geeks /geeks_copied

```
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -mkdir /geeks_copied
19/01/31 12:46:03 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -cp /geeks  /geeks_copied
19/01/31 12:46:26 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -ls /geeks_copied
19/01/31 12:47:10 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
Found 1 items
drwxr-xr-x   - suraj supergroup          0 2019-01-31 12:46 /geeks_copied/geeks
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$
```

1. **mv:** This command is used to move files within hdfs. Lets cut-paste a file *myfile.txt* from *geeks* folder to *geeks_copied*.

   Syntax:

   bin/hdfs dfs -mv <src(on hdfs)> <src(on hdfs)>

   Example:

   bin/hdfs -mv /geeks/myfile.txt /geeks_copied

```
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -ls /geeks
19/01/31 12:48:42 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
Found 3 items
-rw-r--r--   1 suraj supergroup        205 2019-01-31 11:31 /geeks/AI.txt
-rw-r--r--   1 suraj supergroup         96 2019-01-31 12:38 /geeks/cutAndPaste.txt
-rw-r--r--   1 suraj supergroup          0 2019-01-31 11:10 /geeks/myfile.txt
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -mv /geeks/myfile.txt  /geeks_copied
19/01/31 12:49:27 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -ls /geeks
19/01/31 12:49:42 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
Found 2 items
-rw-r--r--   1 suraj supergroup        205 2019-01-31 11:31 /geeks/AI.txt
-rw-r--r--   1 suraj supergroup         96 2019-01-31 12:38 /geeks/cutAndPaste.txt
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -ls /geeks_copied
19/01/31 12:50:00 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
Found 2 items
drwxr-xr-x   - suraj supergroup          0 2019-01-31 12:46 /geeks_copied/geeks
-rw-r--r--   1 suraj supergroup          0 2019-01-31 11:10 /geeks_copied/myfile.txt
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$
```

1. **rmr:** This command deletes a file from HDFS *recursively*. It is very useful command when you want to delete a *non-empty directory*.

   Syntax:

   bin/hdfs dfs -rmr <filename/directoryName>

   Example:

   bin/hdfs dfs -rmr /geeks_copied -> It will delete all the content inside the directory then the directory itself.

```
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -ls /geeks_copied
19/01/31 12:58:16 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
Found 2 items
drwxr-xr-x   - suraj supergroup          0 2019-01-31 12:46 /geeks_copied/geeks
-rw-r--r--   1 suraj supergroup          0 2019-01-31 11:10 /geeks_copied/myfile.txt
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -rmr /geeks_copied
rmr: DEPRECATED: Please use 'rm -r' instead.
19/01/31 12:58:40 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
19/01/31 12:58:42 INFO fs.TrashPolicyDefault: Namenode trash configuration: Deletion interval = 0 minutes, Emptier interval = 0 minutes.
Deleted /geeks_copied
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -ls /geeks_copied
19/01/31 12:59:35 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
ls: `/geeks_copied': No such file or directory
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$
```

1. **du:** It will give the size of each file in directory.

   Syntax:

   bin/hdfs dfs -du <dirName>

   Example:

   bin/hdfs dfs -du /geeks

```
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -du /geeks
19/01/31 13:01:57 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
205  205  /geeks/AI.txt
96   96   /geeks/cutAndPaste.txt
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$
```

1. **dus:**: This command will give the total size of directory/file.

Syntax:

bin/hdfs dfs -dus <dirName>

Example:

bin/hdfs dfs -dus /geeks

```
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$ bin/hdfs dfs -dus /geeks
dus: DEPRECATED: Please use 'du -s' instead.
19/01/31 13:02:34 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applic
able
301  301  /geeks
suraj@suraj:~/hadoop-2.5.0-cdh5.3.2$
```

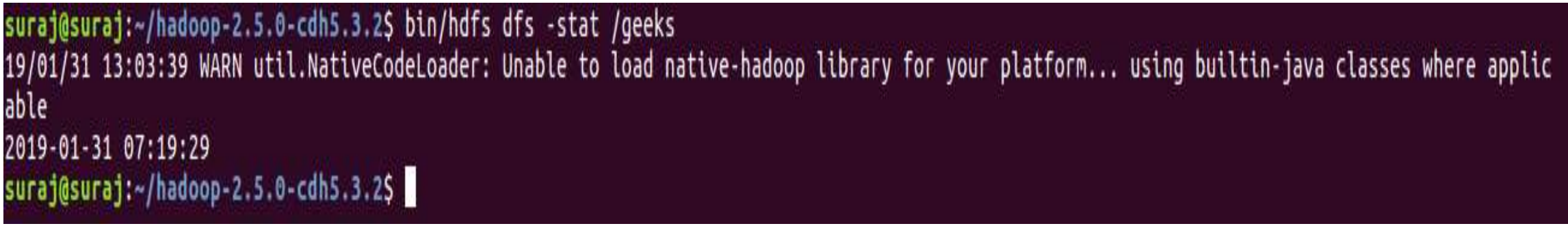
1. **stat:** It will give the last modified time of directory or path. In short it will give stats of the directory or file.

   Syntax:

   bin/hdfs dfs -stat <hdfs file>

   Example:

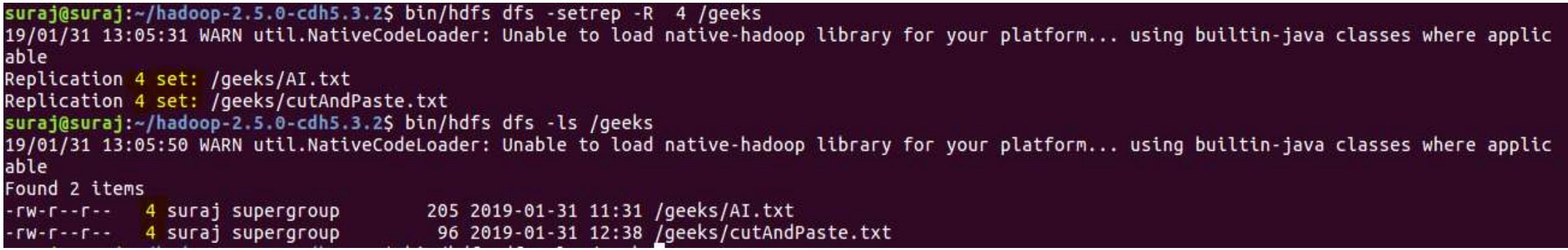
   bin/hdfs dfs -stat /geeks



1. **setrep:** This command is used to change the replication factor of a file/directory in HDFS. By default it is 3 for anything which is stored in HDFS (as set in hdfs *core-site.xml*).
   Example 1: To change the replication factor to 6 for *geeks.txt* stored in HDFS.
   bin/hdfs dfs -setrep -R -w 6 geeks.txt

   Example 2: To change the replication factor to 4 for a directory *geeksInput* stored in HDFS.
   bin/hdfs dfs -setrep -R 4 /geeks

   Note: The **-w** means wait till the replication is completed. And **-R** means recursively, we use it for directories as they may also contain many files and folders inside them.



Note: There are more commands in HDFS but we discussed the commands which are commonly used when working with Hadoop. You can check out the list of *dfs* commands using the following command:

bin/hdfs dfs

```
Usage: hadoop fs [generic options]
        [-appendToFile <localsrc> ... <dst>]
        [-cat [-ignoreCrc] <src> ...]
        [-checksum <src> ...]
        [-chgrp [-R] GROUP PATH...]
        [-chmod [-R] <MODE[,MODE]... | OCTALMODE> PATH...]
        [-chown [-R] [OWNER][:[GROUP]] PATH...]
        [-copyFromLocal [-f] [-p] <localsrc> ... <dst>]
        [-copyToLocal [-p] [-ignoreCrc] [-crc] <src> ... <localdst>]
        [-count [-q] <path> ...]
        [-cp [-f] [-p | -p[topax]] <src> ... <dst>]
        [-createSnapshot <snapshotDir> [<snapshotName>]]
        [-deleteSnapshot <snapshotDir> <snapshotName>]
        [-df [-h] [<path> ...]]
        [-du [-s] [-h] <path> ...]
        [-expunge]
        [-get [-p] [-ignoreCrc] [-crc] <src> ... <localdst>]
        [-getfacl [-R] <path>]
        [-getfattr [-R] {-n name | -d} [-e en] <path>]
        [-getmerge [-nl] <src> <localdst>]
        [-help [cmd ...]]
        [-ls [-d] [-h] [-R] [<path> ...]]
        [-mkdir [-p] <path> ...]
        [-moveFromLocal <localsrc> ... <dst>]
        [-moveToLocal <src> <localdst>]
        [-mv <src> ... <dst>]
        [-put [-f] [-p] <localsrc> ... <dst>]
        [-renameSnapshot <snapshotDir> <oldName> <newName>]
        [-rm [-f] [-r|-R] [-skipTrash] <src> ...]
        [-rmdir [--ignore-fail-on-non-empty] <dir> ...]
        [-setfacl [-R] [{-b|-k} {-m|-x <acl_spec>} <path>]|[--set <acl_spec> <path>]]
        [-setfattr {-n name [-v value] | -x name} <path>]
        [-setrep [-R] [-w] <rep> <path> ...]
        [-stat [format] <path> ...]
        [-tail [-f] <file>]
        [-test -[defsz] <path>]
        [-text [-ignoreCrc] <src> ...]
```

THE END

T2, T3, T6, T7, T9, U0, U6, U8, V1, V2, V5, V6, V7, W2, W4, W6, W8, Y0, Y1, Y4, Y6, Y9, Z0, 26, 27, 28, 30, 31, 32 , 554, 535