Introducing

Big Data

Author:

AmirHossein Mohammadi

Data GROWTH ON THE INTERNET PER MINUTE

HGST
a Western Digital company


YouTube

YouTube

72 Hours per Minute


Gmail™
by Google™

EMAIL USERS SEND

240,166,667

MESSAGES

HGST
a Western Digital company

240,166,667


facebook

facebook®

User share 684,478


Google

GOOGLE RECEIVES OVER

2,283,105

HGST
a Western Digital company

twitter

Send over 236,111

WordPress

WordPress

340 New blog posts


YOUR AD
YOUR AD
YOUR AD

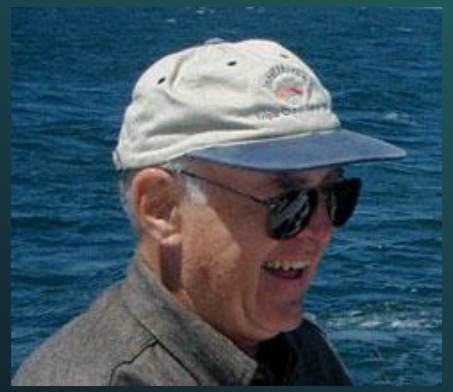
571 New Website


Instagram

3600 User Share New Photo

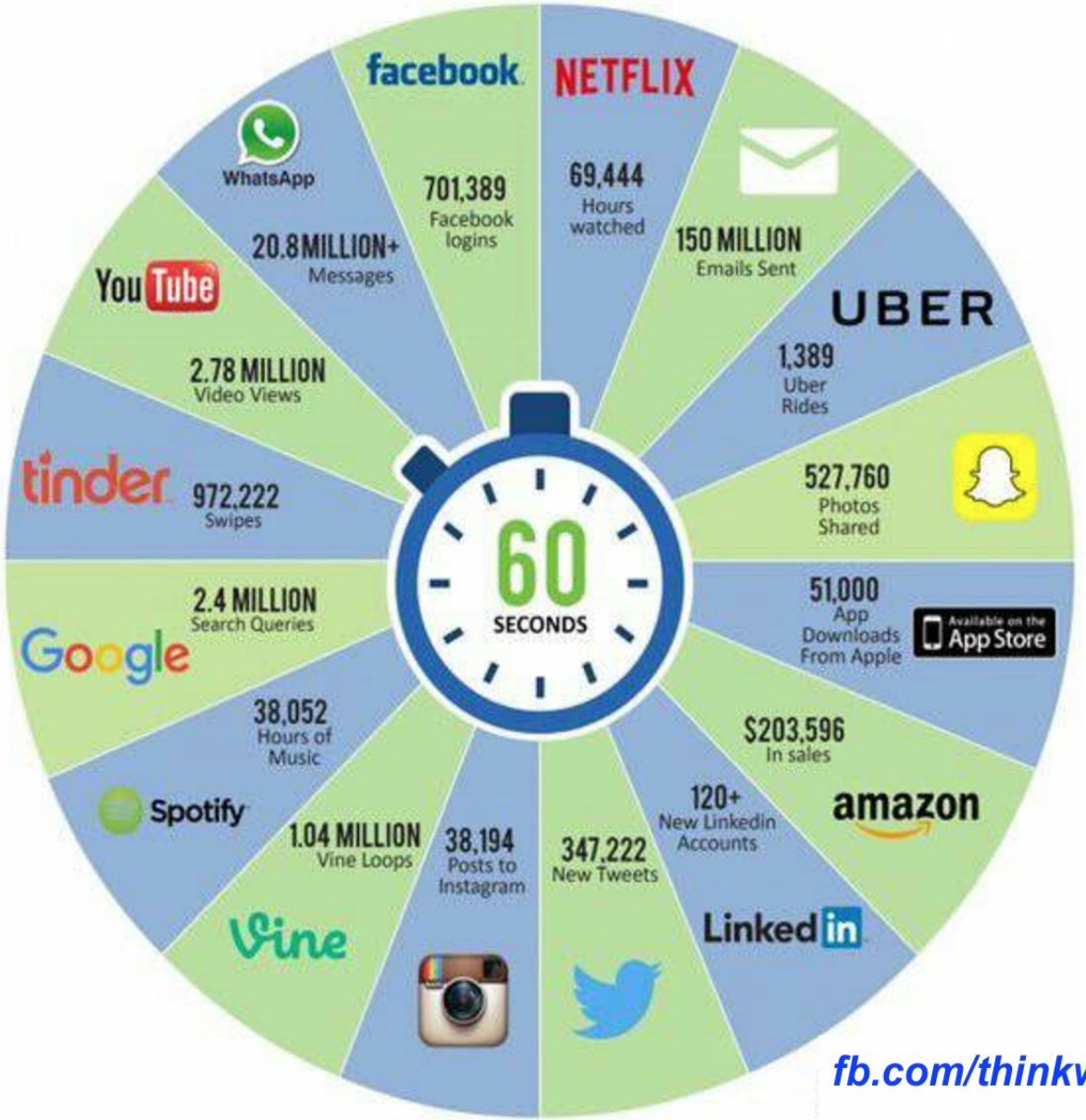
This is the statistics for 2013

Moore's law is:



the observation that the number of transistors in a dense integrated circuit doubles about every two years.

2016 What happens in an INTERNET MINUTE?

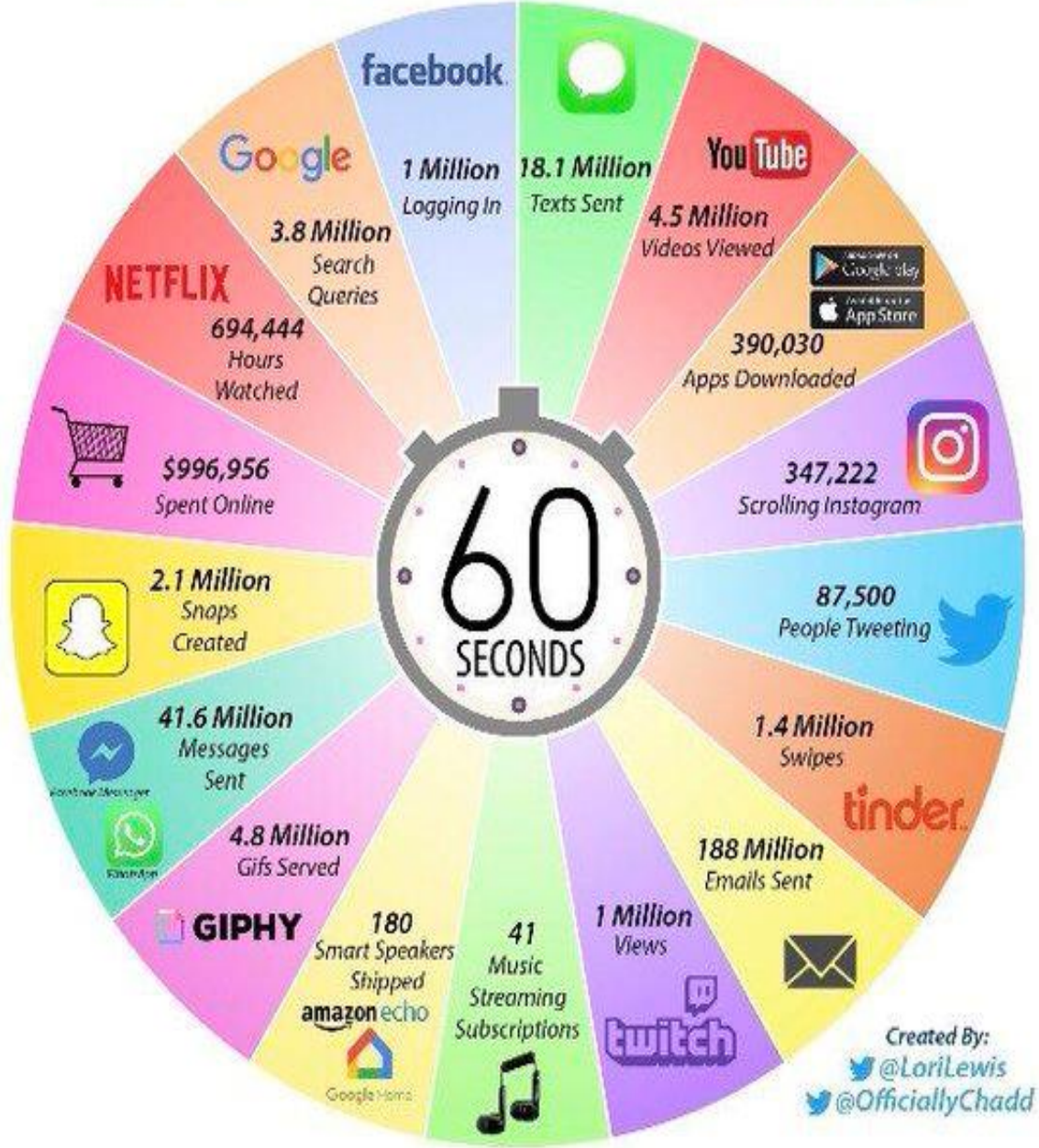


2017 This Is What Happens In An Internet Minute

fb.com/thinkwittynow

2018 This Is What Happens In An Internet Minute

2019 This Is What Happens In An Internet Minute



Global Mobile Data Traffic, 2015 to 2020

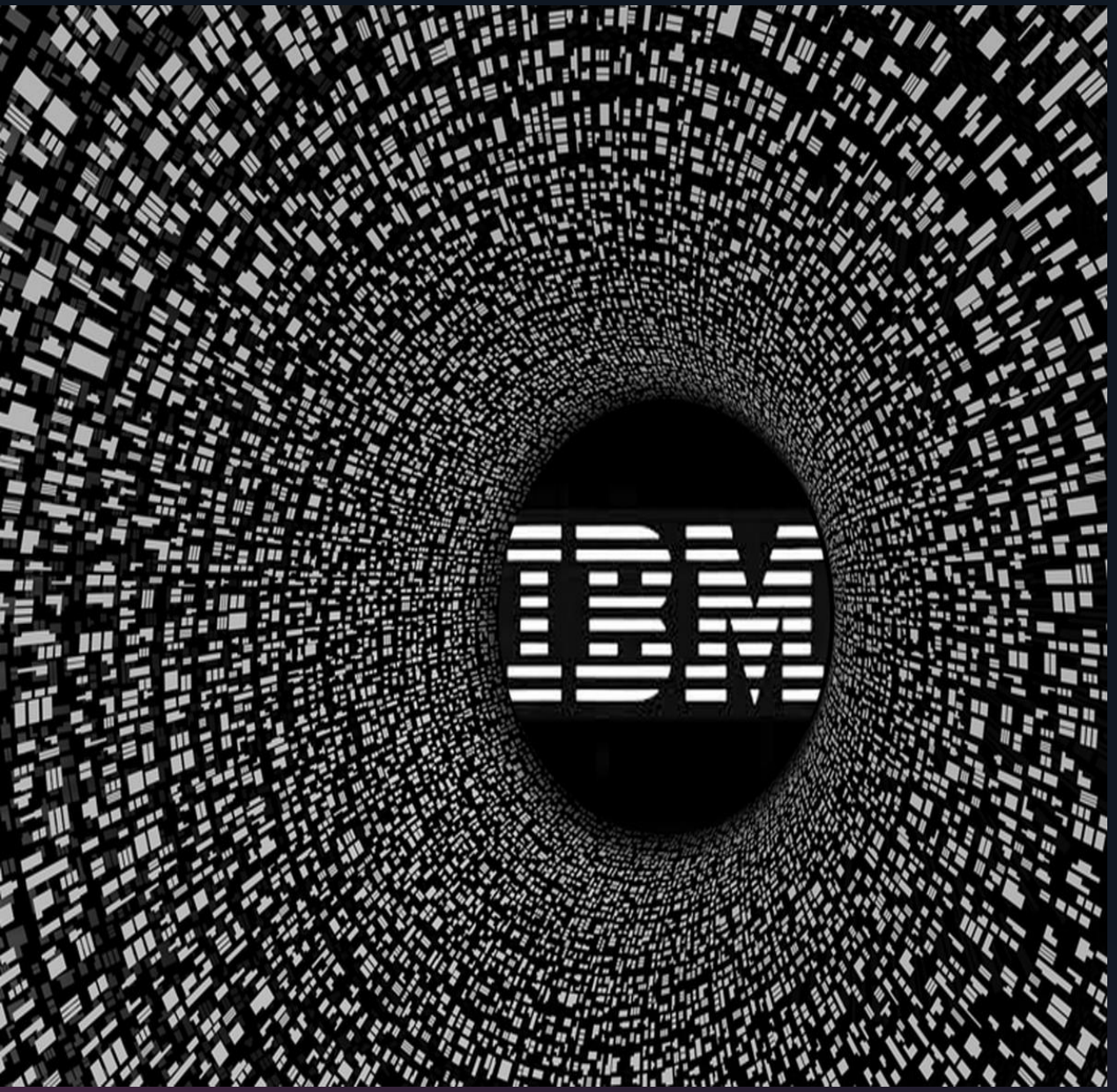
edureka!

Cisco Forecasts 30.6 Exabytes per Month of Mobile Data Traffic by 2020

3 major trends contributing to the growth of mobile data traffic:

- Adapting to Smarter Mobile Devices
- Defining Cell Network Advances—2G, 3G, and 4G (5G Perspectives)
- Reviewing Tiered Pricing—Unlimited Data and Shared Plans

Source: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html



IBM

نشان داد که هر روز ۲٫۵ اگزا بایت داده تولید میشود و همچنین ۹۰% از داده ها در ۲ سال اخیر تولید شده است.

Large Data Means?

- 1000 **kilobytes** = 1 Megabyte
- 1000 **Megabytes** = 1 Gigabyte
- 1000 **Gigabytes** = 1 Terabyte
- 1000 **Terabytes** = 1 Petabyte
- 1000 **Petabytes** = 1 Exabyte
- 1000 **Exabytes** = 1 Zettabyte
- 1000 **Zettabytes** = 1 Yottabyte

What is Big Data?

What is Big Data?

edureka!

"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"

Volume

Processing increasing huge data sets

Variety

Processing different types of data

Velocity

Data is being generated at an alarming rate

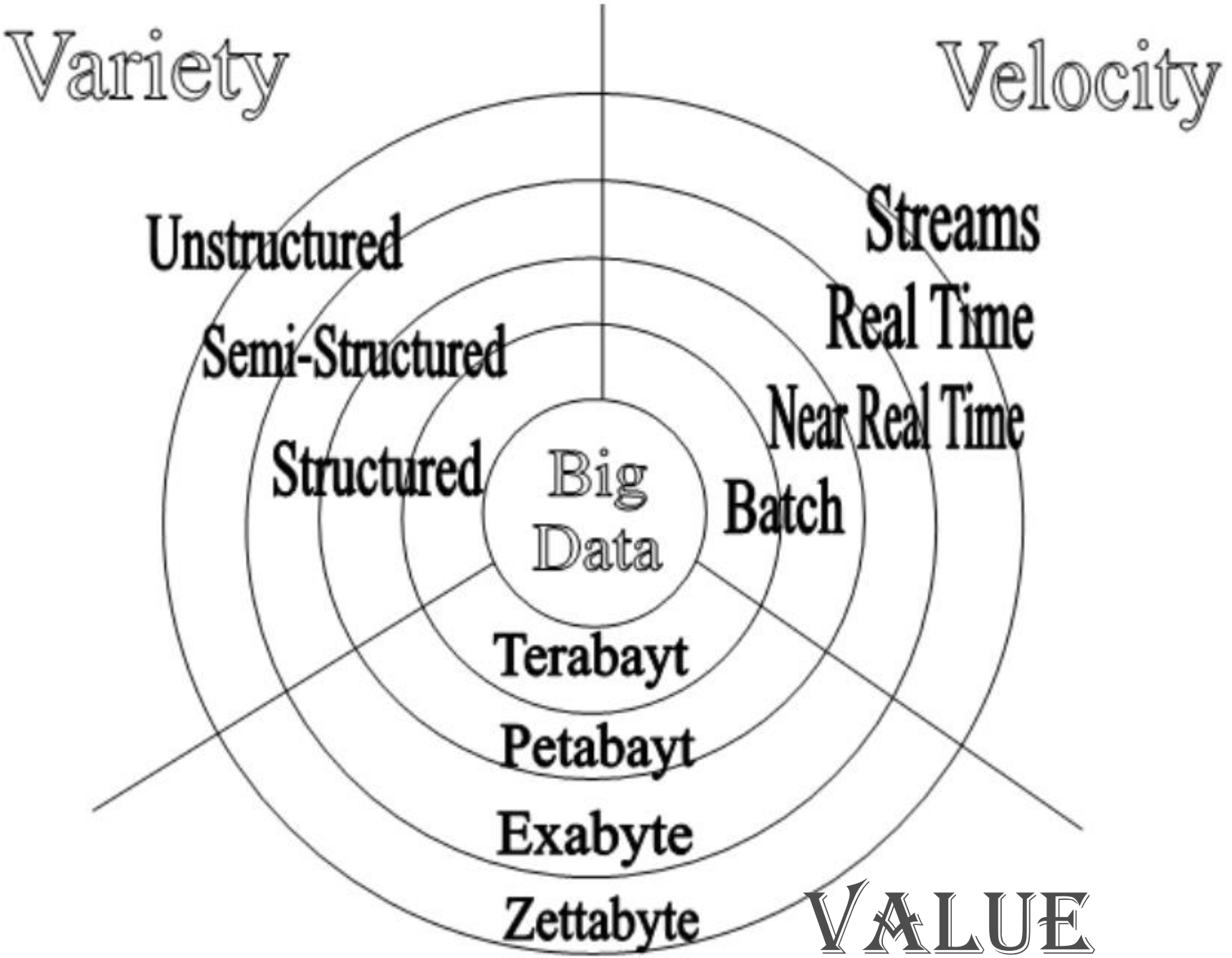
Value

Finding correct meaning out of the data

Veracity

Min	Max	Mean	SD
4.3	?	5.84	0.83
2.0	4.4	3.05	[illegible]
15000	7.9	1.20	0.43
0.1	2.5	?	0.76

Uncertainty and inconsistencies in the data


Variety
Velocity
Unstructured
Semi-Structured
Structured
Streams
Real Time
Near Real Time
Batch
Big Data
Terabayt
Petabayt
Exabyte
Zettabyte
VALUE

Big Data.....

برای مجموعه داده های حجیم که بزرگ ، متنوع ، با ساختار پیچیده و با دشواریهایی برای ذخیره سازی ، تحلیل و تصویرسازی (نمایش) ، پردازشهای بیشتر یا نتایج میباشد. پروسه تحقیق بر روی داده های حجیم جهت آشکارسازی الگوهای مخفی و راز همبستگی ها ، تجزیه و تحلیل big data نامیده میشود.

Story of Big Data & Traditional System

edureka!

Scenario:
Bob has opened a small restaurant in his city

Traditional Scenario

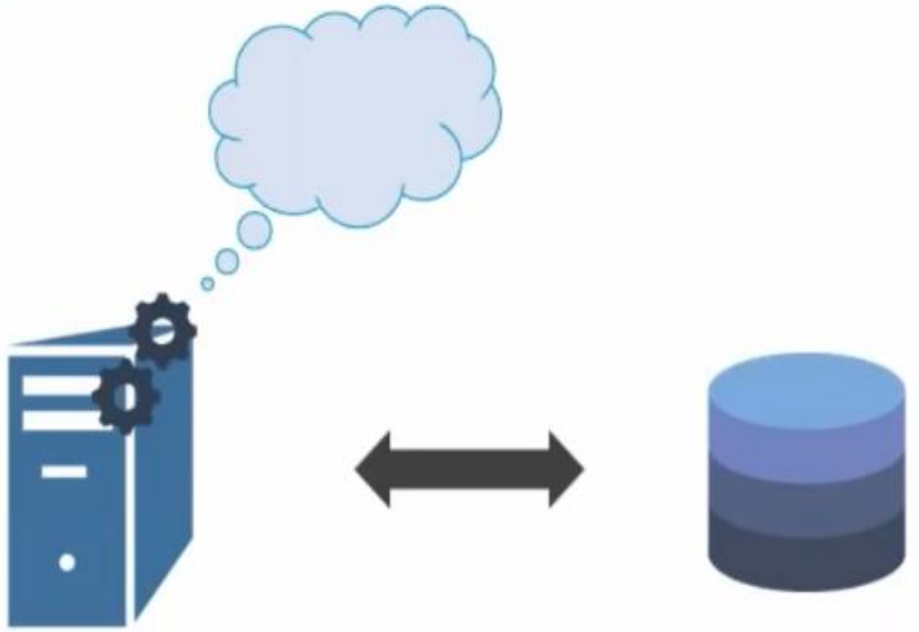
edureka!

Traditional Scenario:

2 orders per hour

Single Cook

Food Shelf



Traditional Processing System

RDBMS

Traditional Scenario

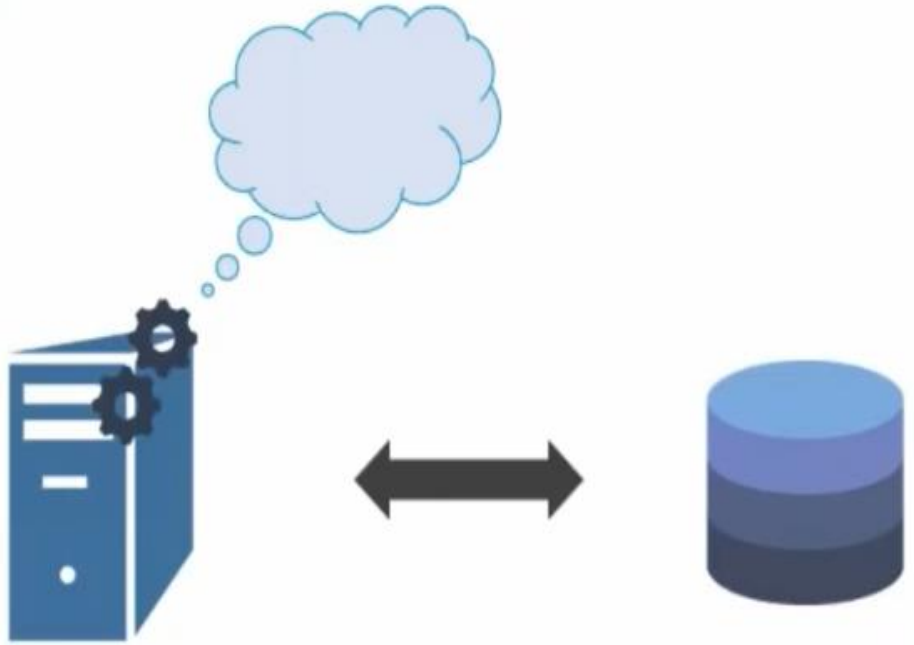
edureka!

Traditional Scenario:

2 orders per hour

Single Cook

Food Shelf



Traditional Processing System

RDBMS

Traditional Scenario

edureka!

Traditional Scenario:

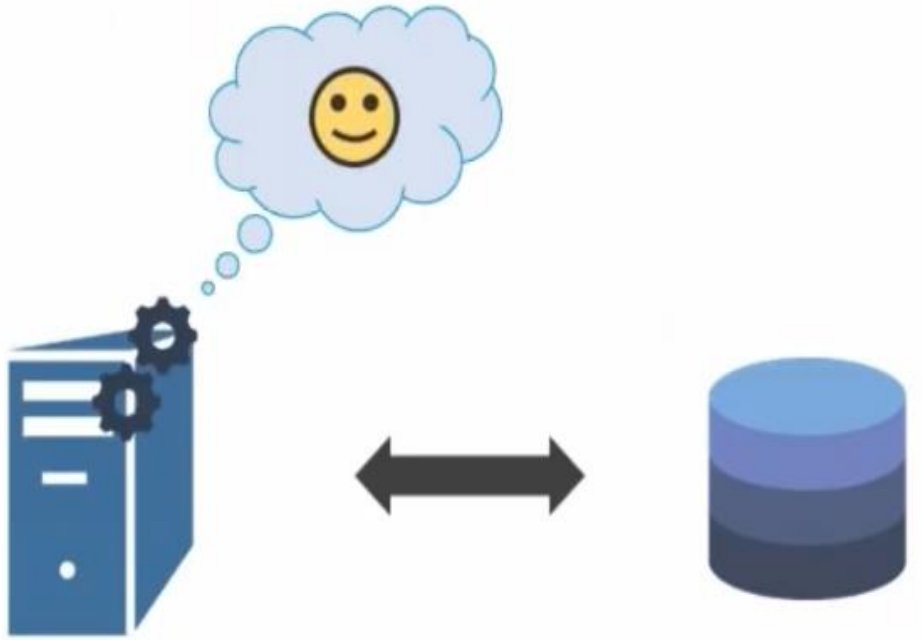
2 orders per hour

Single Cook

Food Shelf

Traditional Scenario:

Data is generated at a steady rate and is structured in nature



Traditional Processing System

RDBMS

Failure of Traditional System

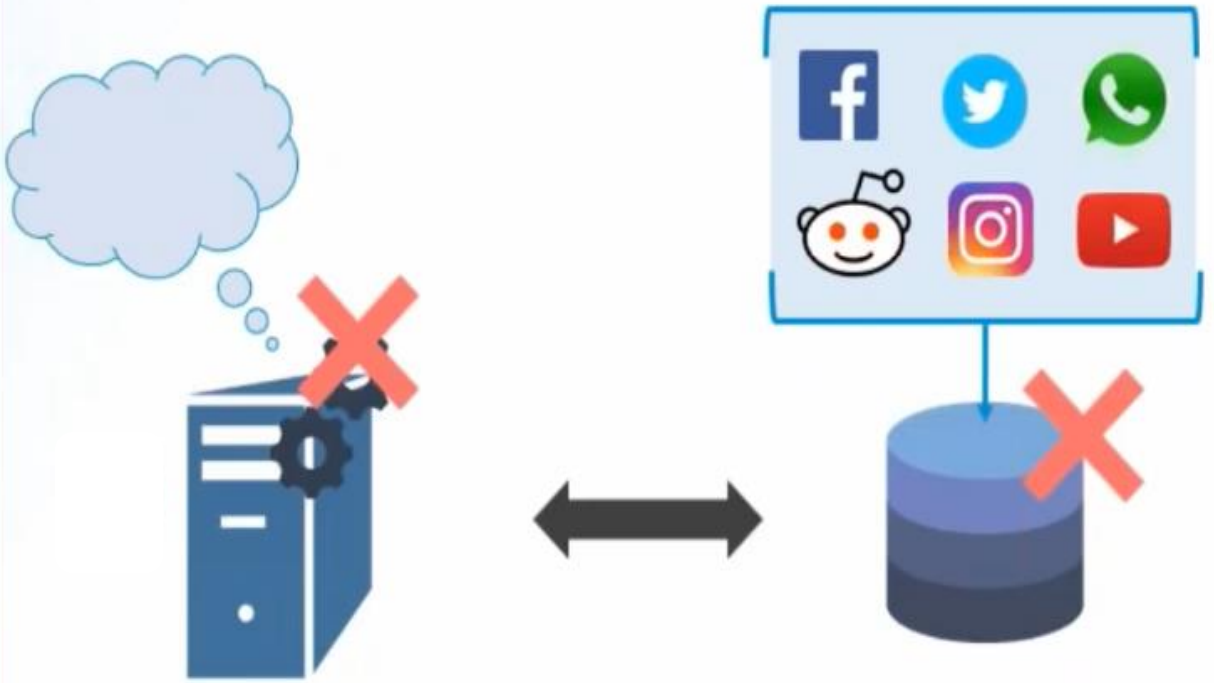
edureka!

Scenario 2:

- They started taking Online orders
- 10 orders per hour

Single Cook
(Regular Computing System)

Food Shelf
(Data)



Traditional Processing System

RDBMS

Failure of Traditional System

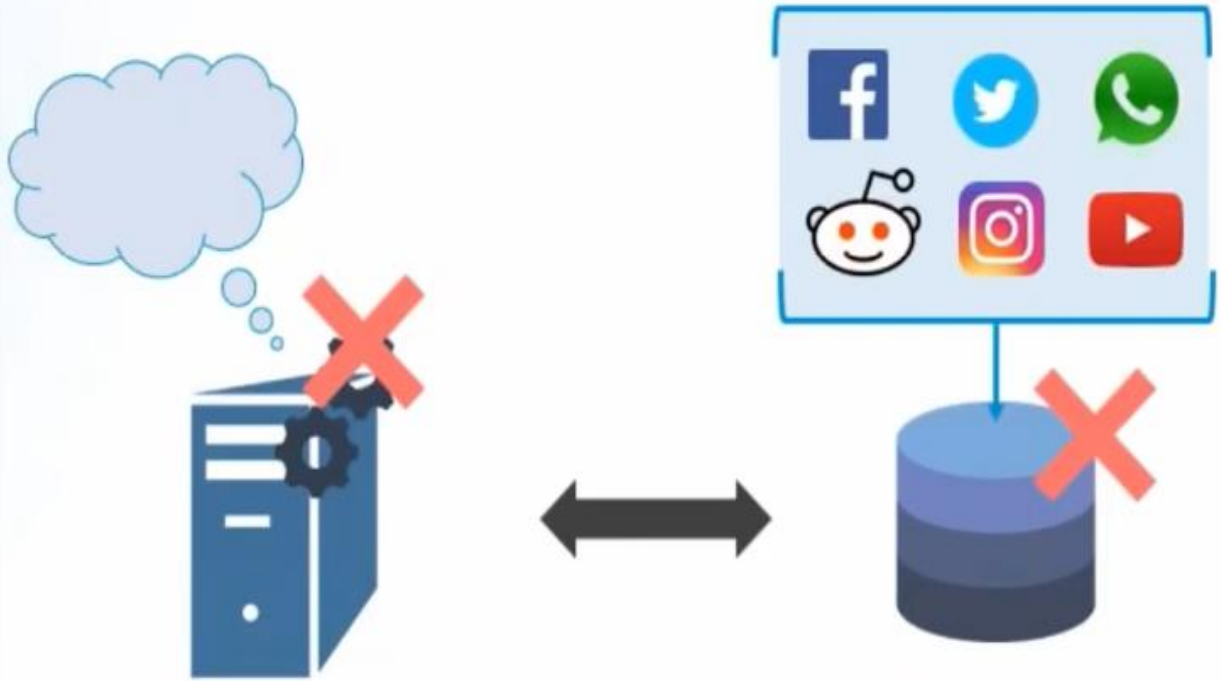
edureka!

Scenario 2:

- They started taking Online orders
- 10 orders per hour

Single Cook
(Regular Computing System)

Food Shelf
(Data)



Traditional Processing System

RDBMS

Failure of Traditional System

edureka!

Scenario 2:

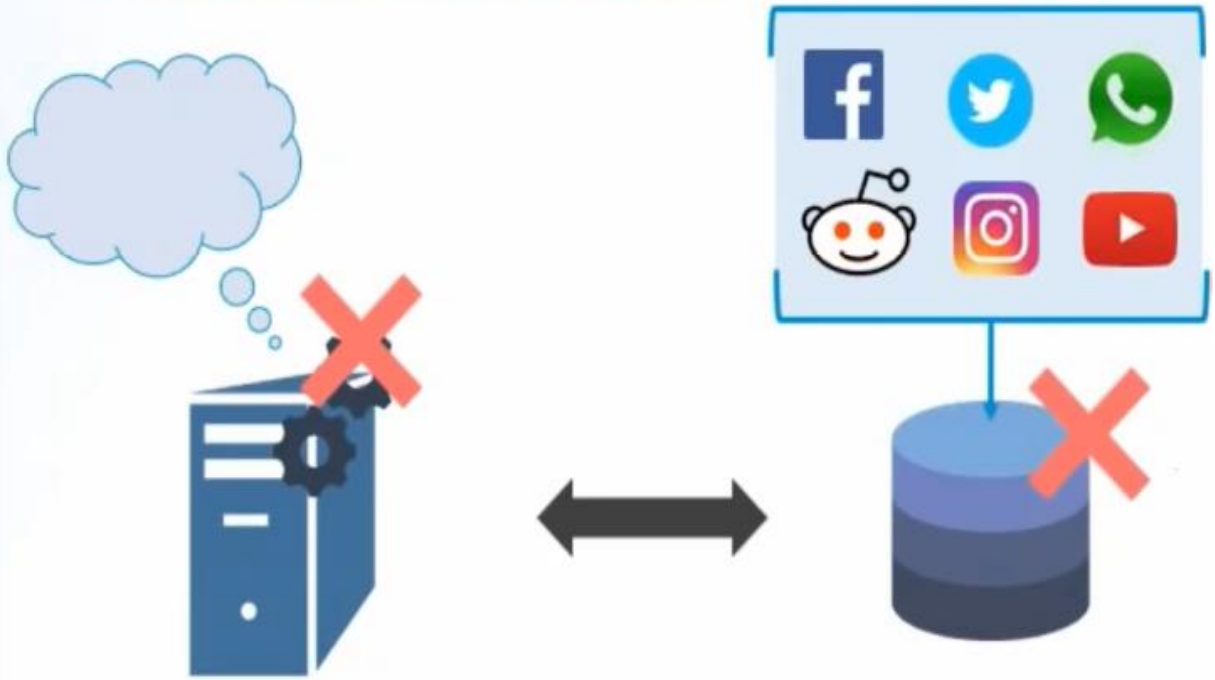
- They started taking Online orders
- 10 orders per hour

Single Cook
(Regular Computing System)

Food Shelf
(Data)

Big Data Scenario:

Heterogenous data is being generated at an alarming rate by multiple sources



Traditional Processing System

RDBMS

Failure of Traditional System

edureka!

Scenario 2:

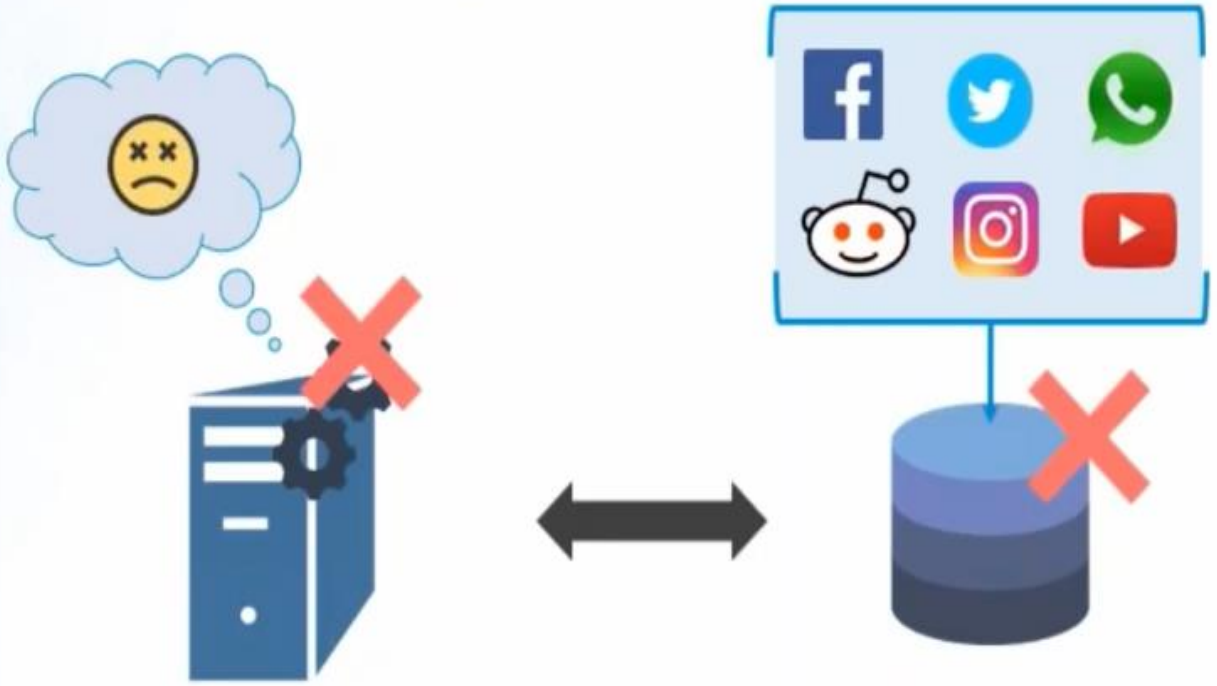
- They started taking Online orders
- 10 orders per hour

Single Cook
(Regular Computing System)

Food Shelf
(Data)

Big Data Scenario:

Heterogenous data is being generated at an alarming rate by multiple sources



Traditional Processing System

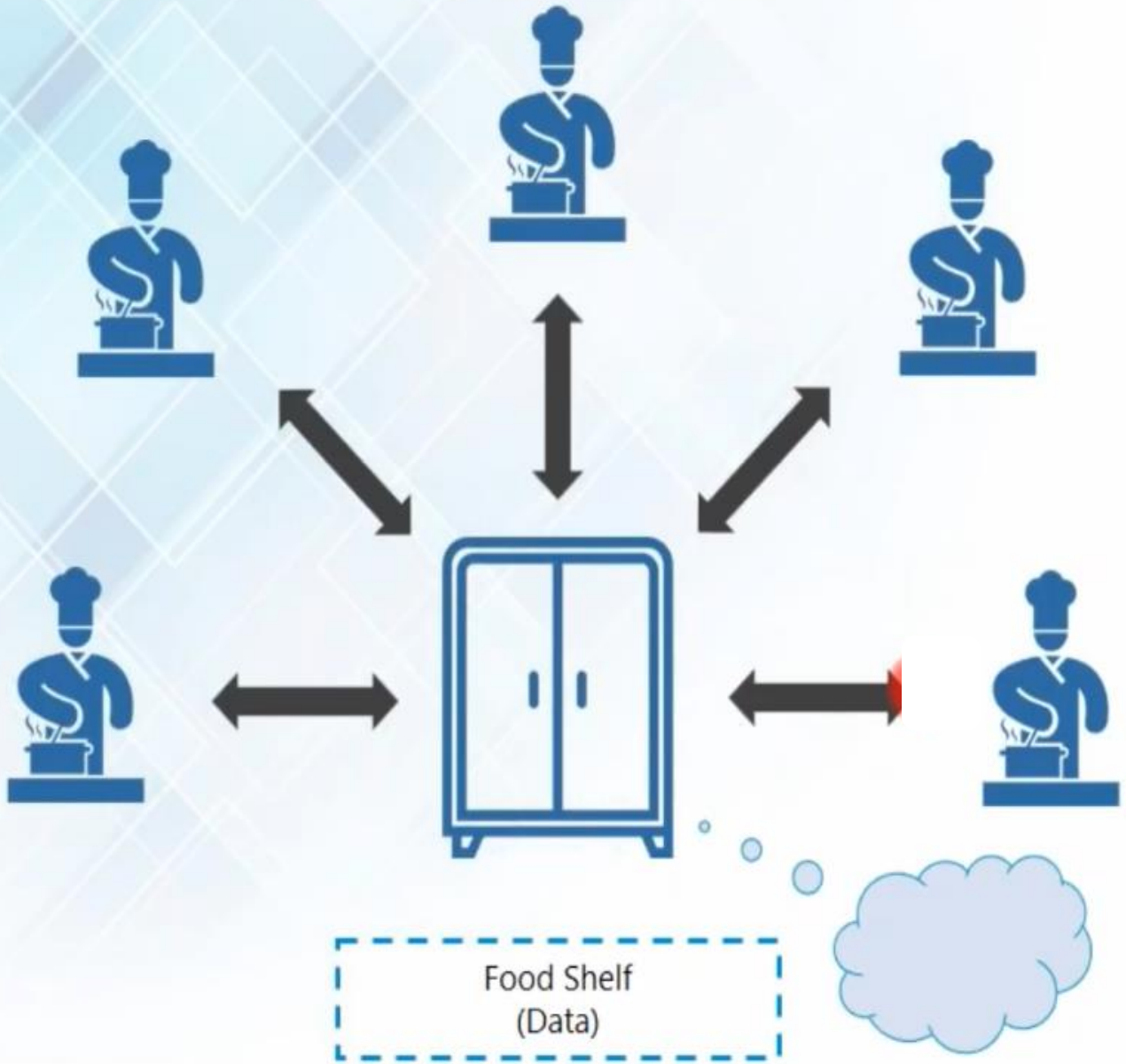
RDBMS

Issue 1: Too Many Orders Per Hour

Solution: Hiring Multiple Cook

Need of an Effective Solution

edureka!



Scenario:

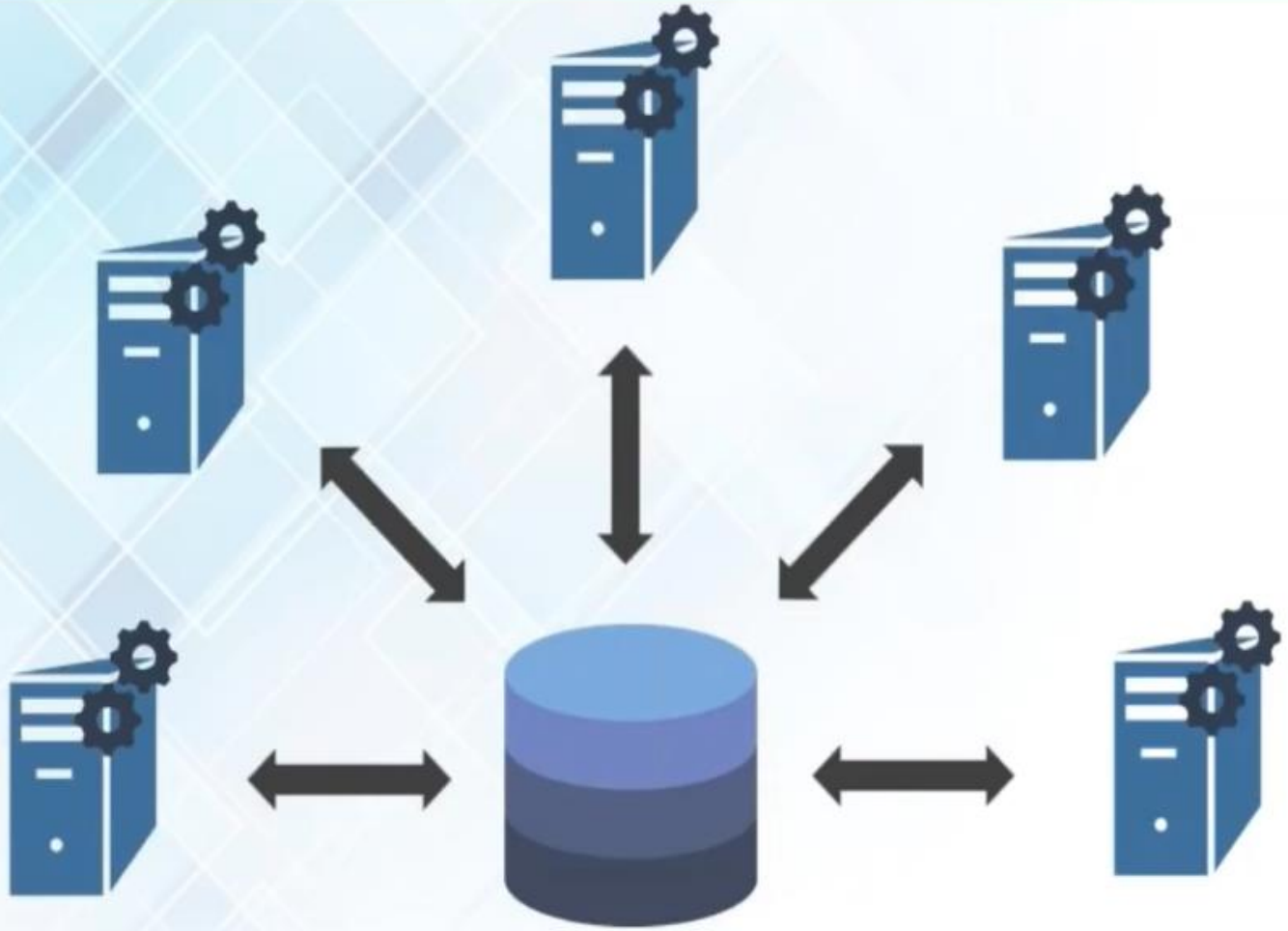
Multiple Cook cooking food

Issue:

Food Shelf becomes the BOTTLENECK

Need of an Effective Solution

edureka!



Data Warehouse

Scenario:

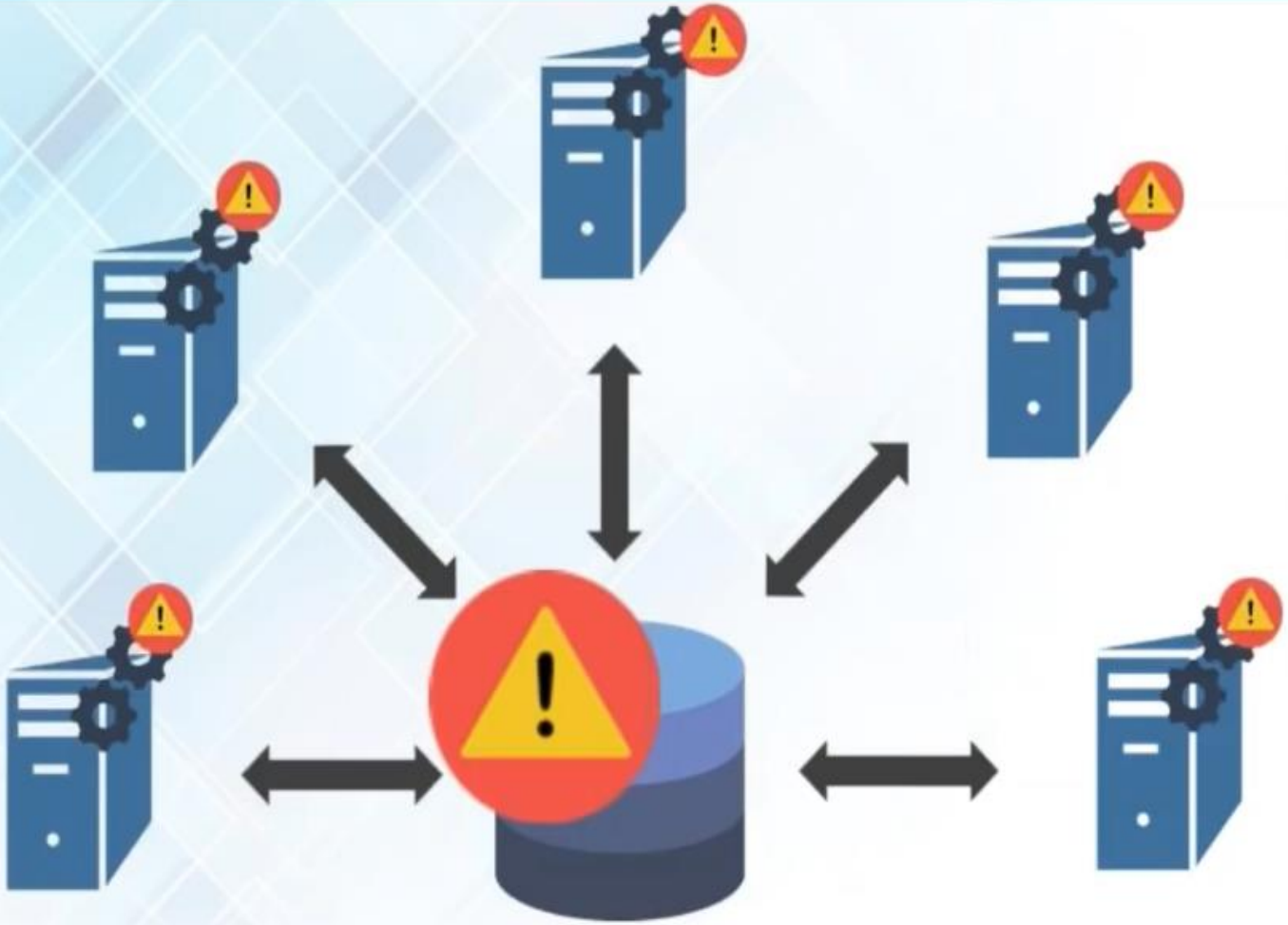
Multiple Processing Unit for data processing

Issue:

Bringing data to processing generated lots of Network overhead

Need of an Effective Solution

edureka!



Scenario:

Multiple Processing Unit for data processing

Issue:

Bringing data to processing generated lots of Network overhead

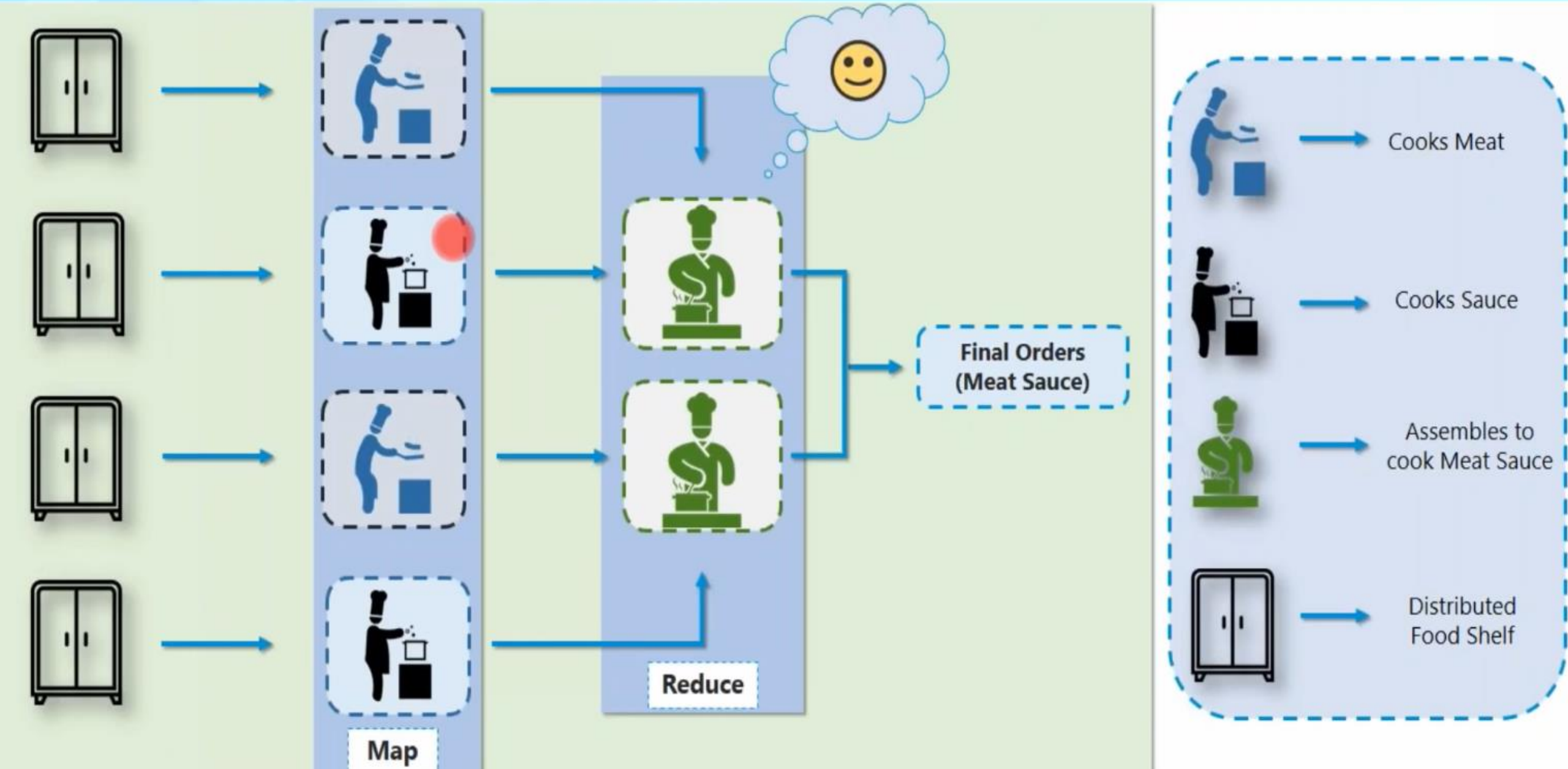
Data Warehouse

Issue 2: Food Shelf becomes the Bottleneck

Solution: Distributed and Parallel Approach

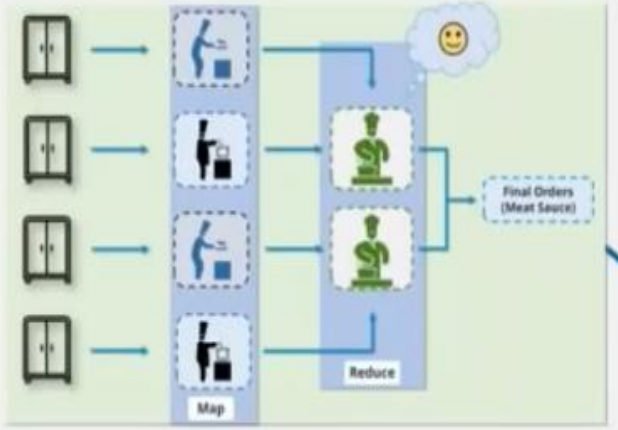
Effective Solution

edureka!



Need of a Framework

edureka!



Do we have a framework that works like that ?

Hadoop Distributions

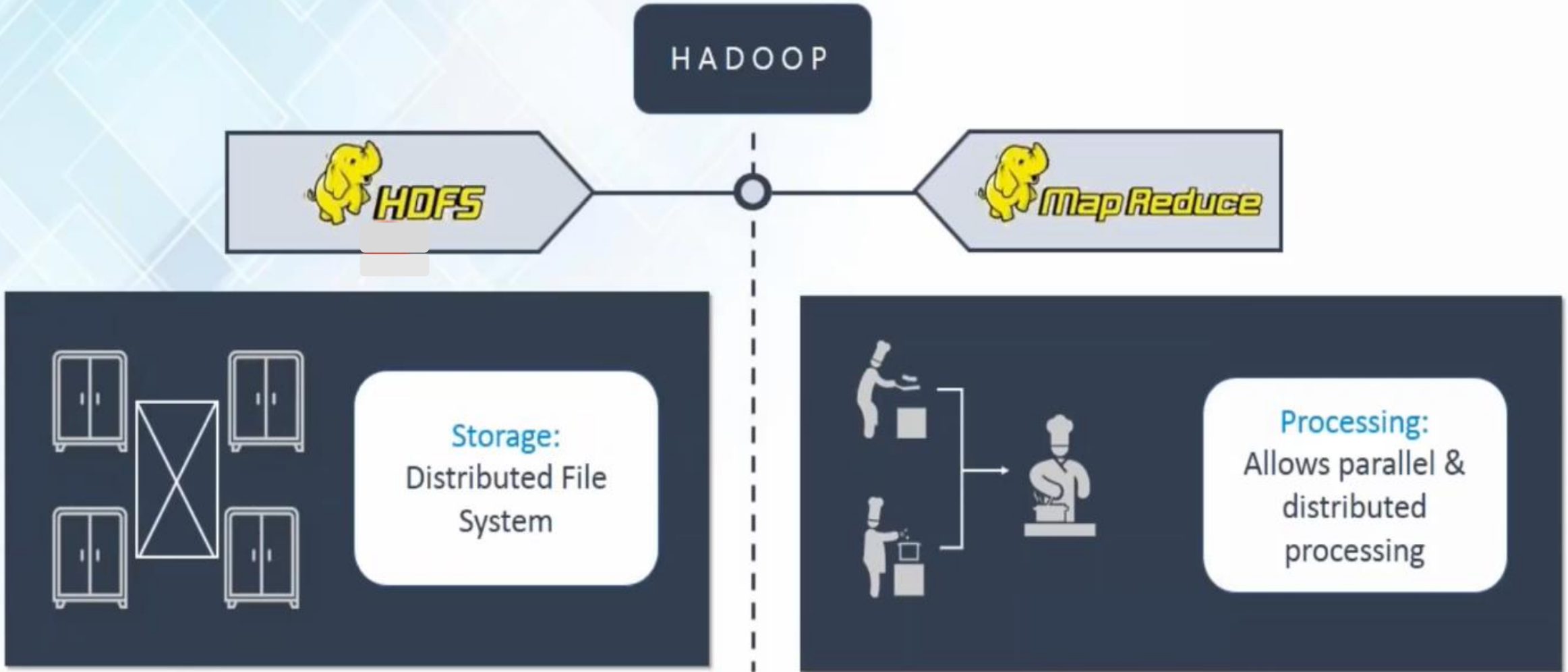
Open Source	Commercial	Cloud-base
Apache Hadoop	Cloudera	AWS
	Hortonworks	Windows Azure
	MapR	

Apache Hadoop: Framework to Process Big Data

Apache Hadoop: Framework to Process Big Data

edureka!

Hadoop is a framework that allows us to store and process large data sets in parallel and distributed fashion



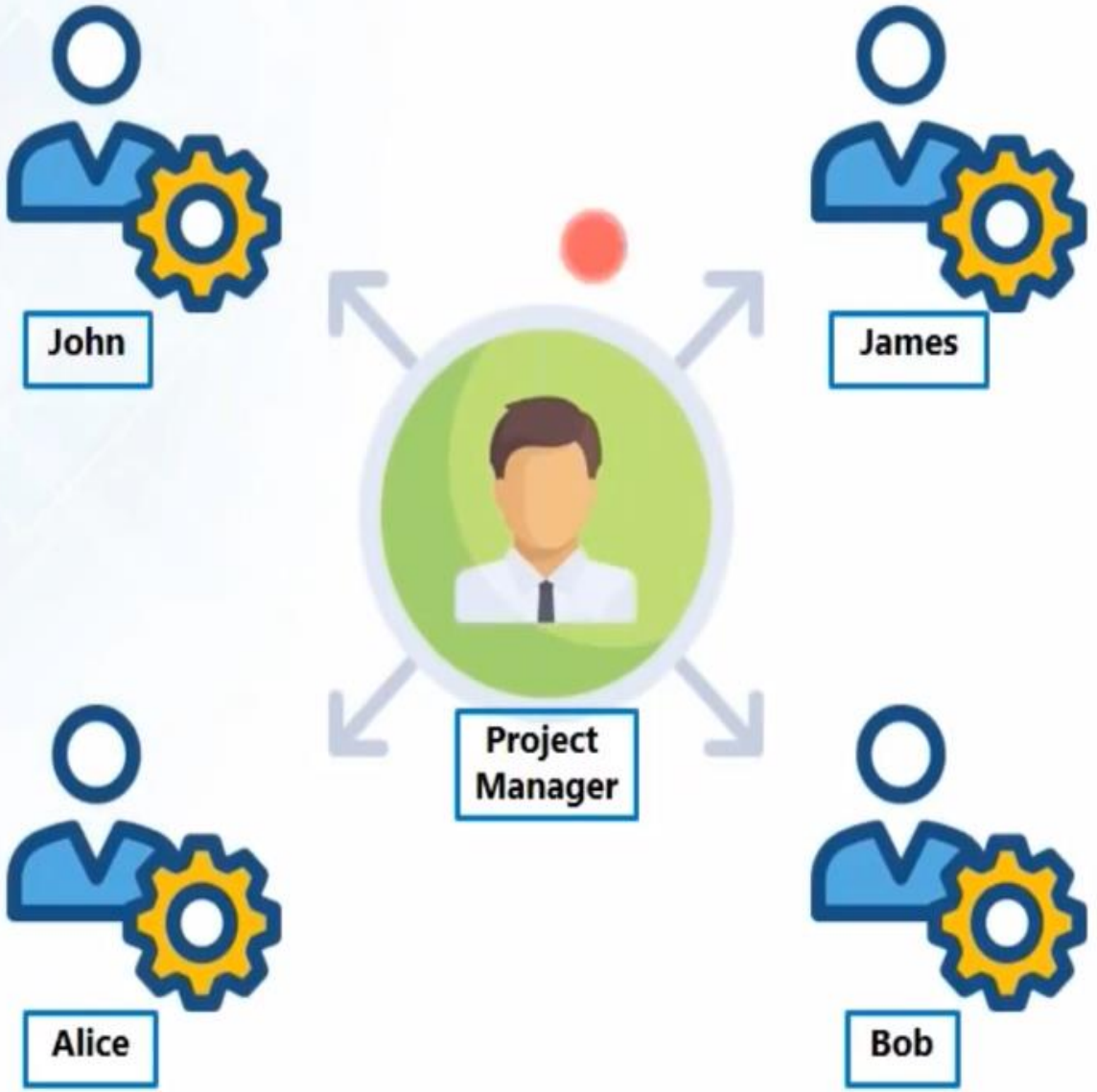
Hadoop: Master/Slave Architecture

Hadoop: Master/Slave Architecture

edureka!

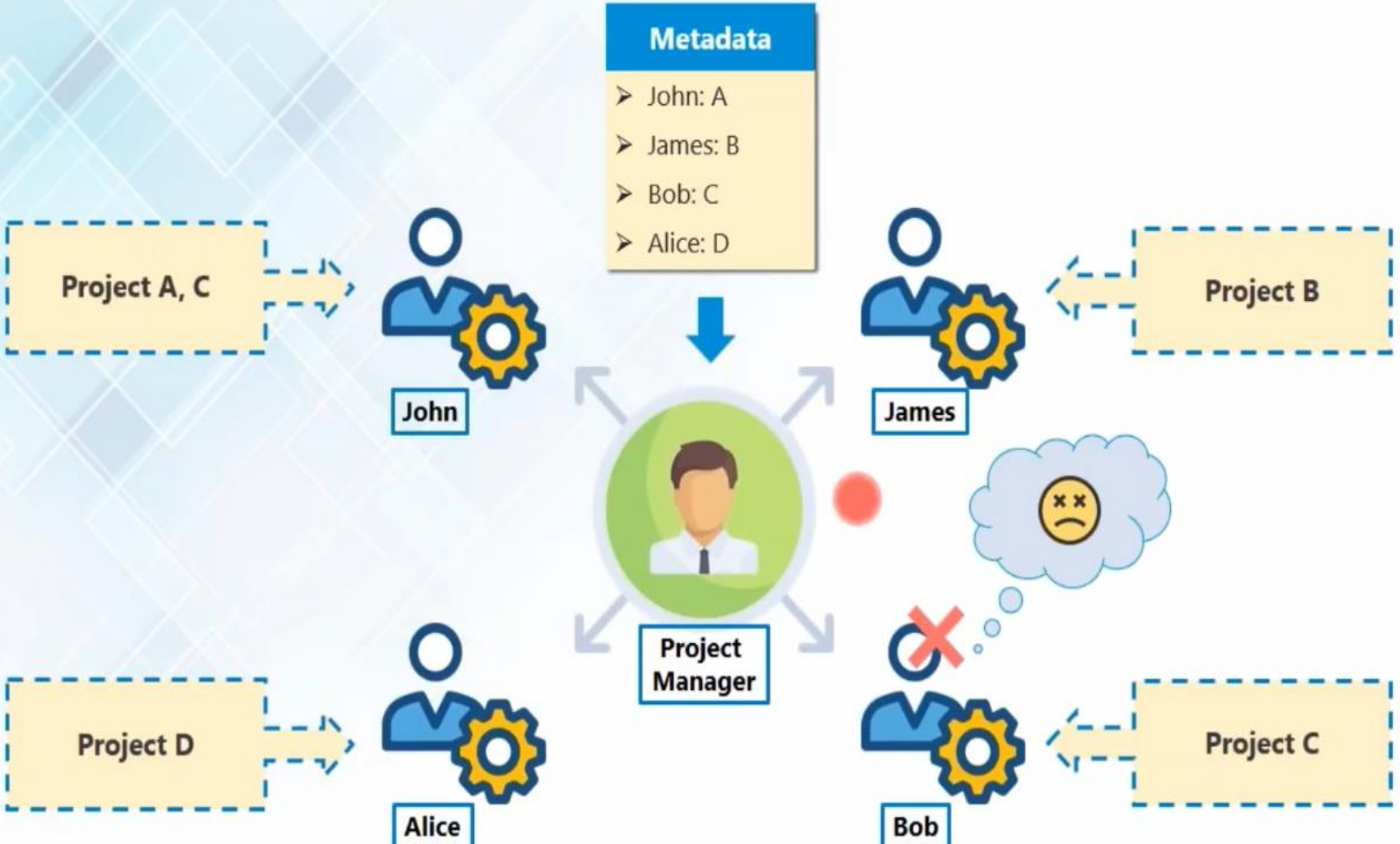
Scenario:

A project Manager managing a team of four employees. He assigns project to each of them and tracks the progress



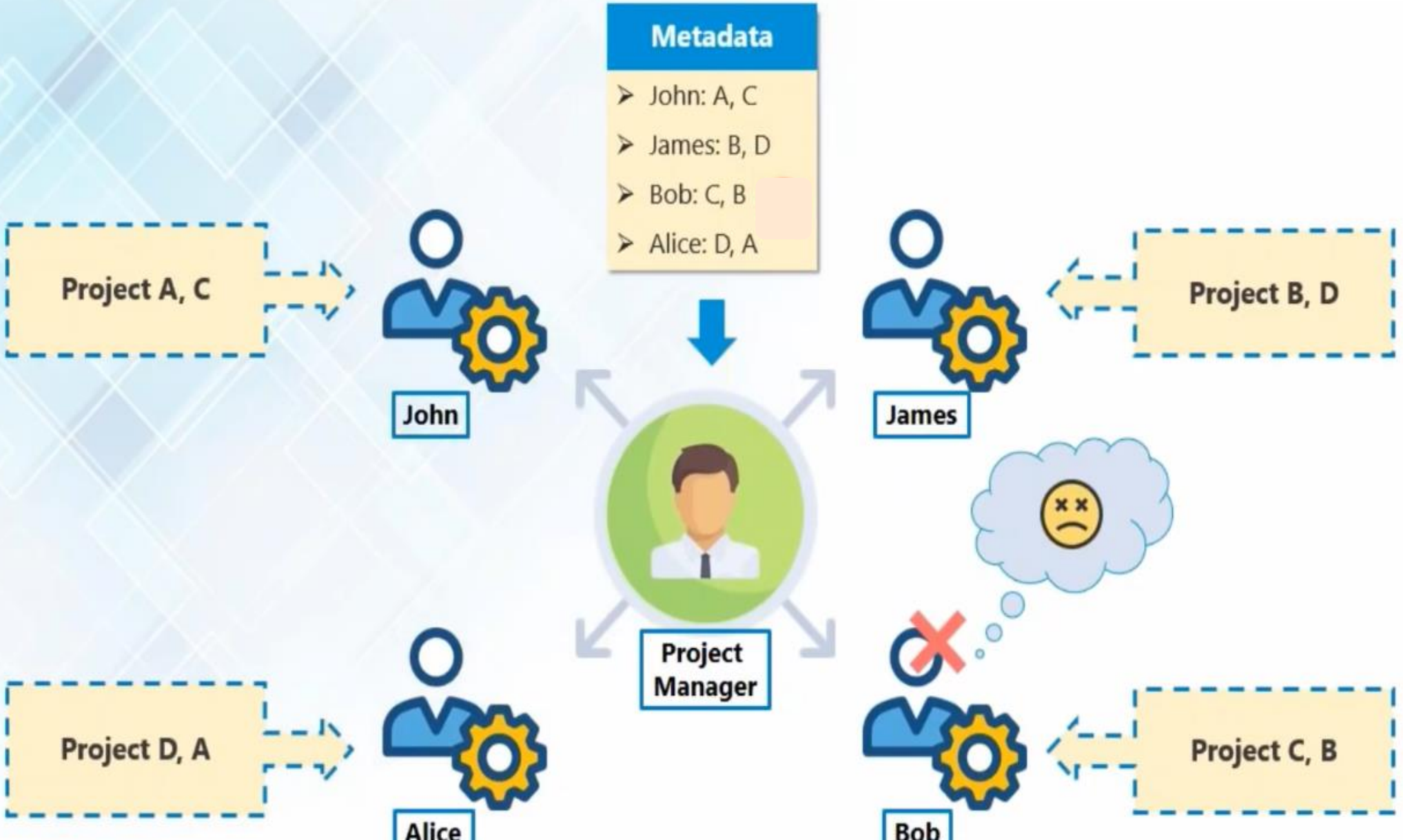
Hadoop: Master/Slave Architecture

edureka!



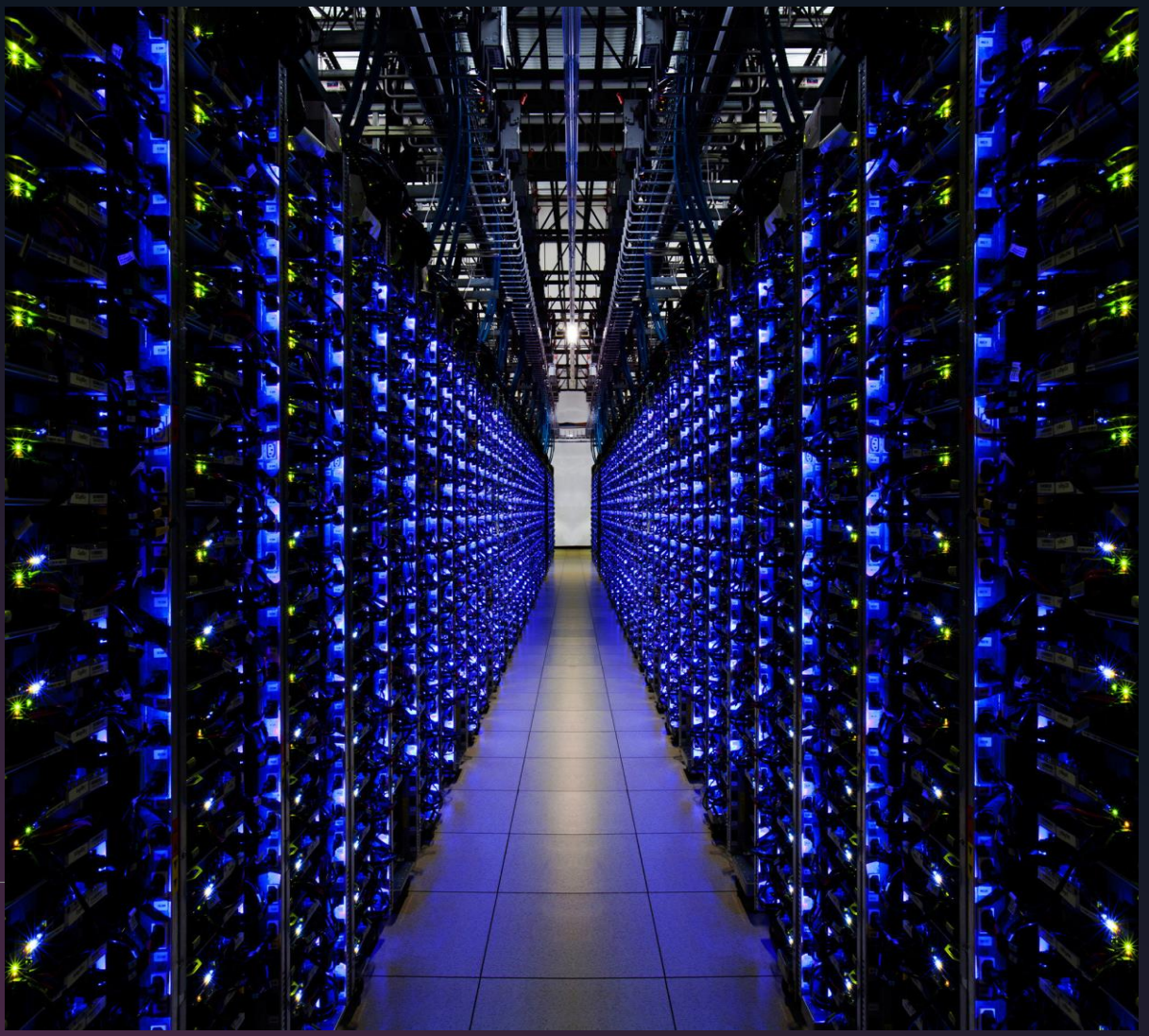
Hadoop: Master/Slave Architecture

edureka!



Cluster/Node

Cluster



A Cluster in a collection of multiple nodes which communicates with each other to perform set of operation at high available rates. Each node is single entity machine or server .

HADOOP CORE COMPONENTS

HDFS

MapReduce

Storage:
Distributed File System

Processing:
Allows parallel & distributed processing

HDFS Core Components:

01

NameNode

02

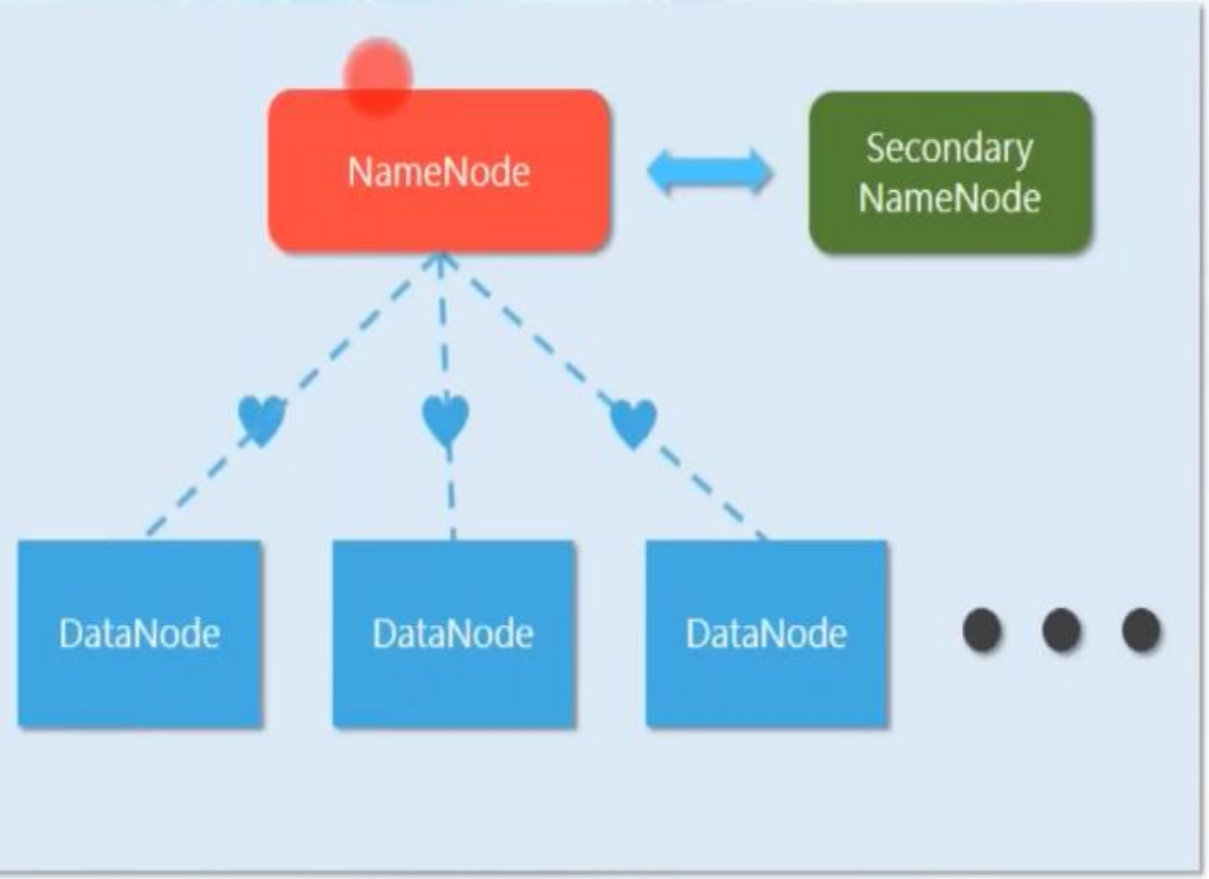
DataNode

03

Secondary NameNode

NameNode & DataNode

edureka!



NameNode:

- Maintains and Manages DataNodes
- Records metadata i.e. information about data blocks e.g. location of blocks stored, the size of the files, permissions, hierarchy, etc.
- Receives heartbeat and block report from all the DataNodes

DataNode:

- Slave daemons
- Stores actual data
- Serves read and write requests from the clients

HDFS Core Components:

01

NameNode

02

DataNode

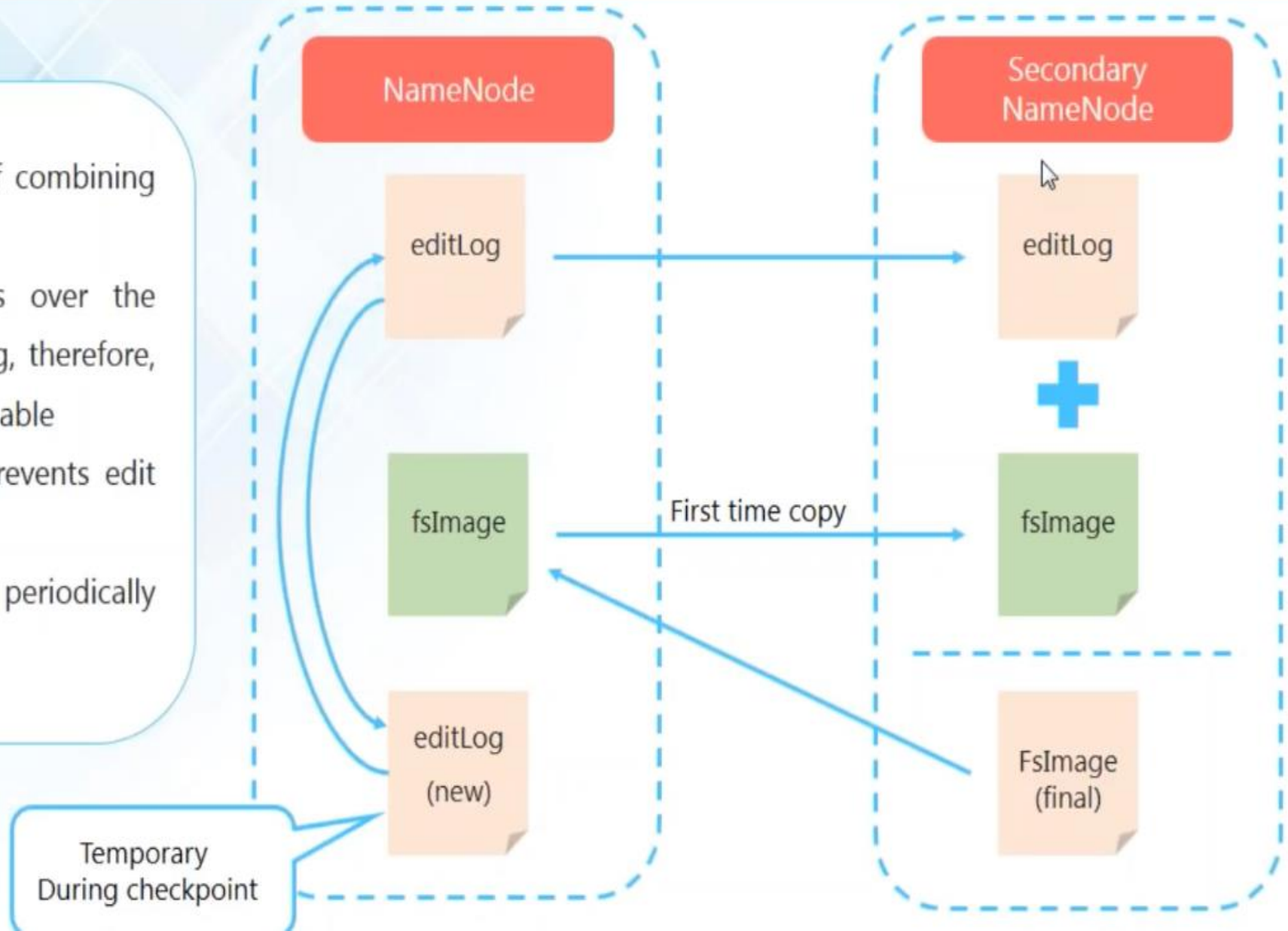
03

Secondary NameNode

Secondary NameNode & Checkpointing

edureka!

- Checkpointing is a process of combining edit logs with FsImage
- Secondary NameNode takes over the responsibility of checkpointing, therefore, making NameNode more available
- Allows faster Failover as it prevents edit logs from getting too huge
- Checkpointing happens periodically (default: 1 hour)

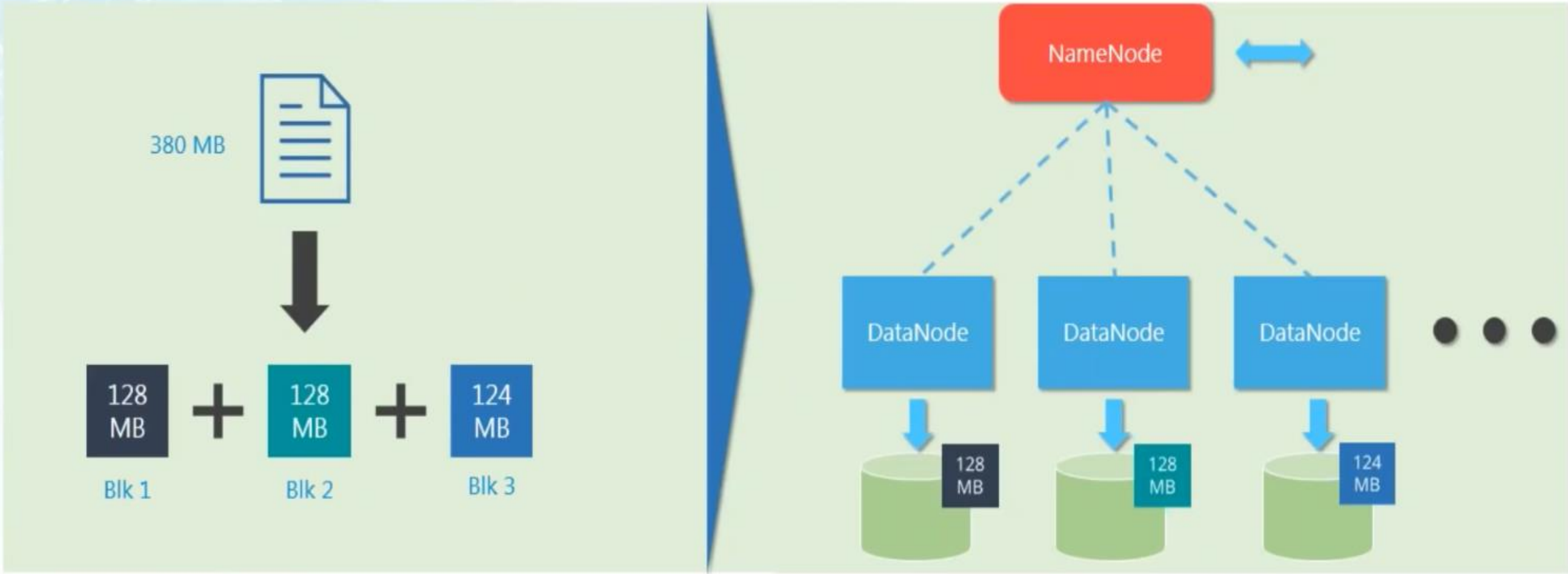


How the data is actually stored in DataNodes?
HDFS Data Blocks

HDFS Data Blocks

edureka!

- Each file is stored on HDFS as blocks
- The default size of each block is 128 MB in Apache Hadoop 2.x (64 MB in Apache Hadoop 1.x)



Advantage of HDFS

- Abstraction
- Scalable
- Parallel

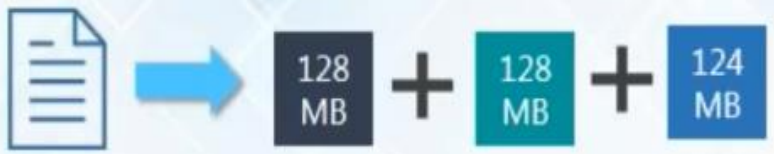
Fault Tolerance: How Hadoop cope up with DataNode Failure?

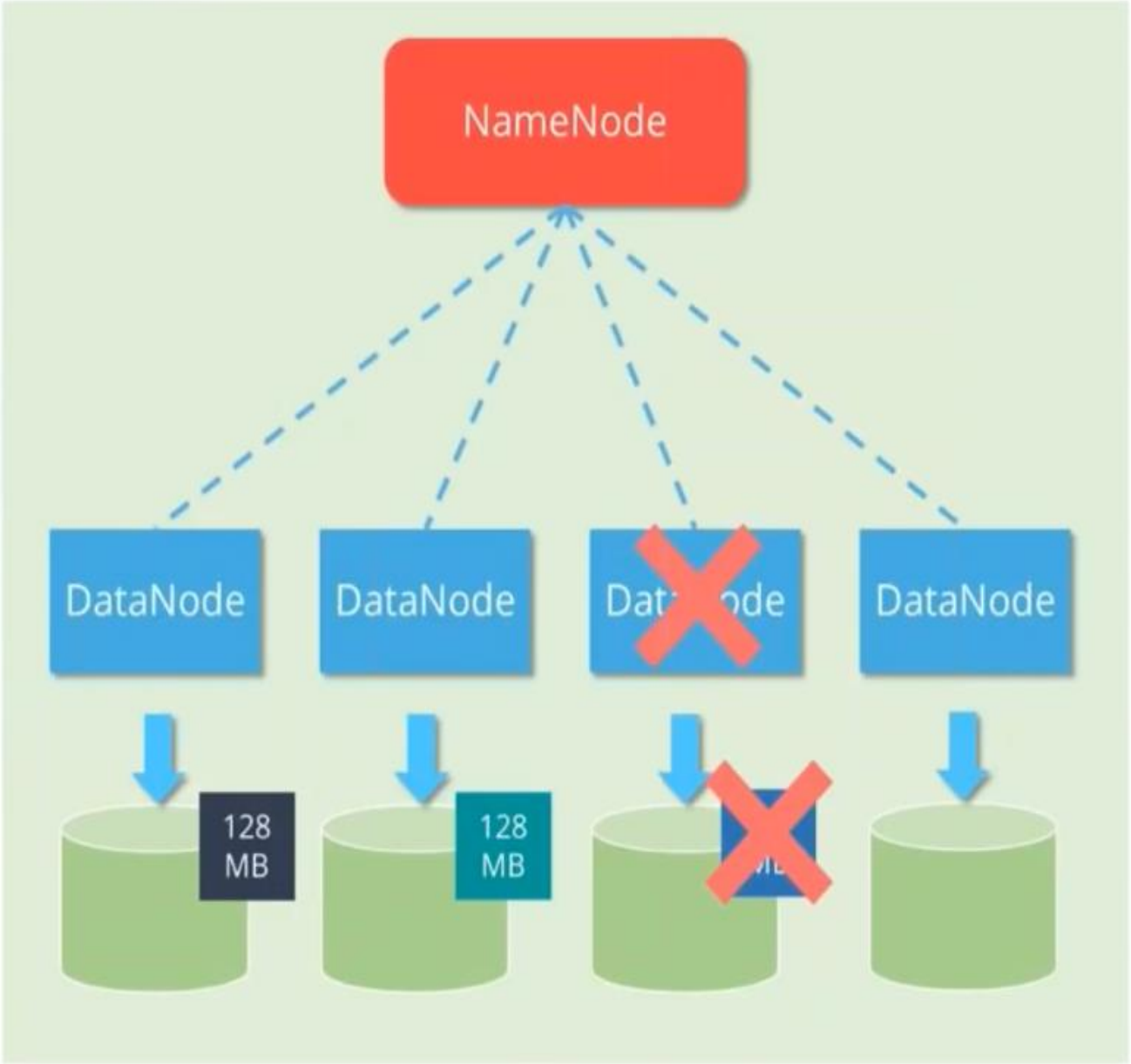
Fault Tolerance

edureka!

Scenario:

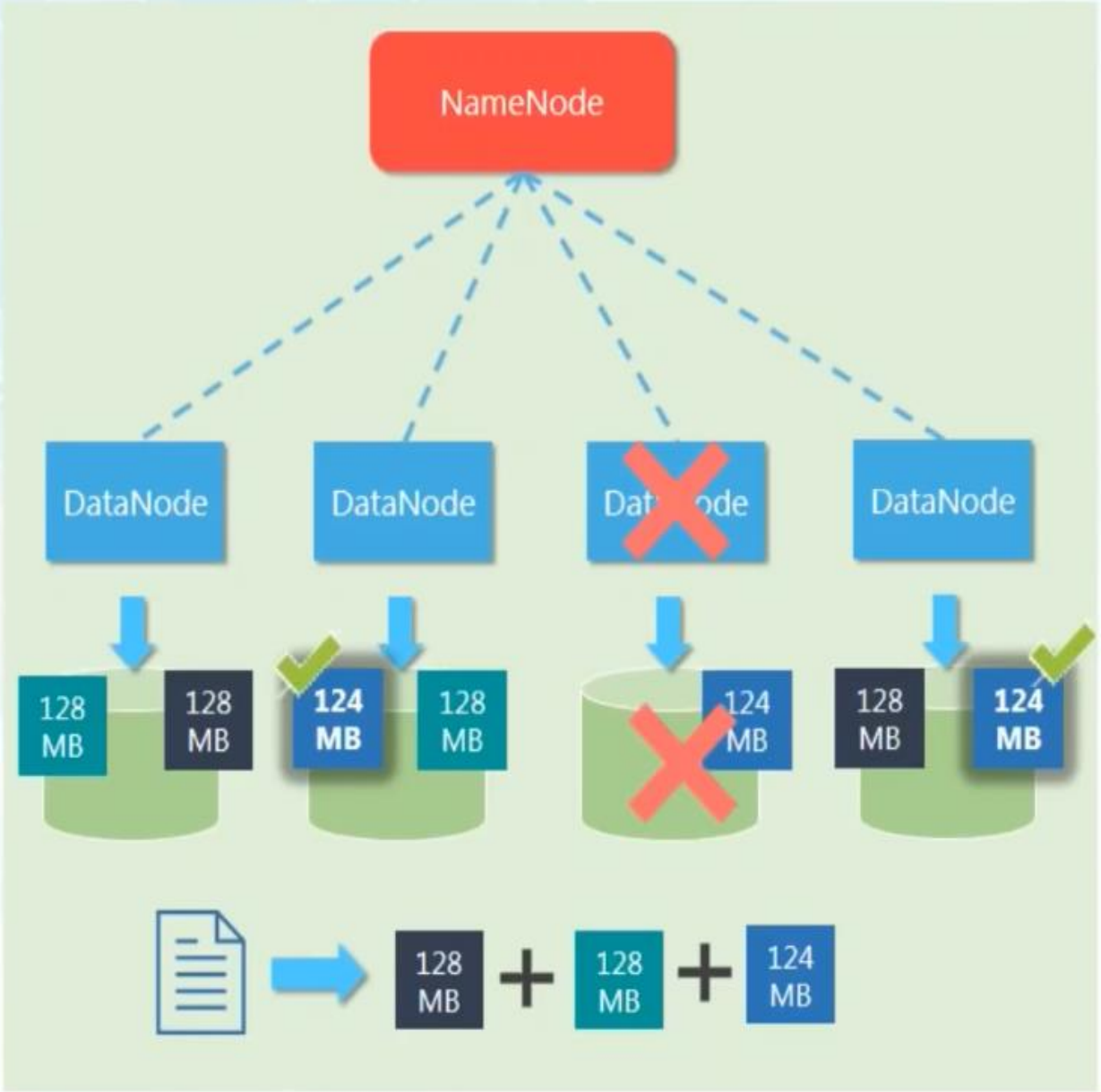
One of the DataNodes crashed containing the data blocks





Solution: Replication Factor

Fault Tolerance: Replication Factor



Solution:

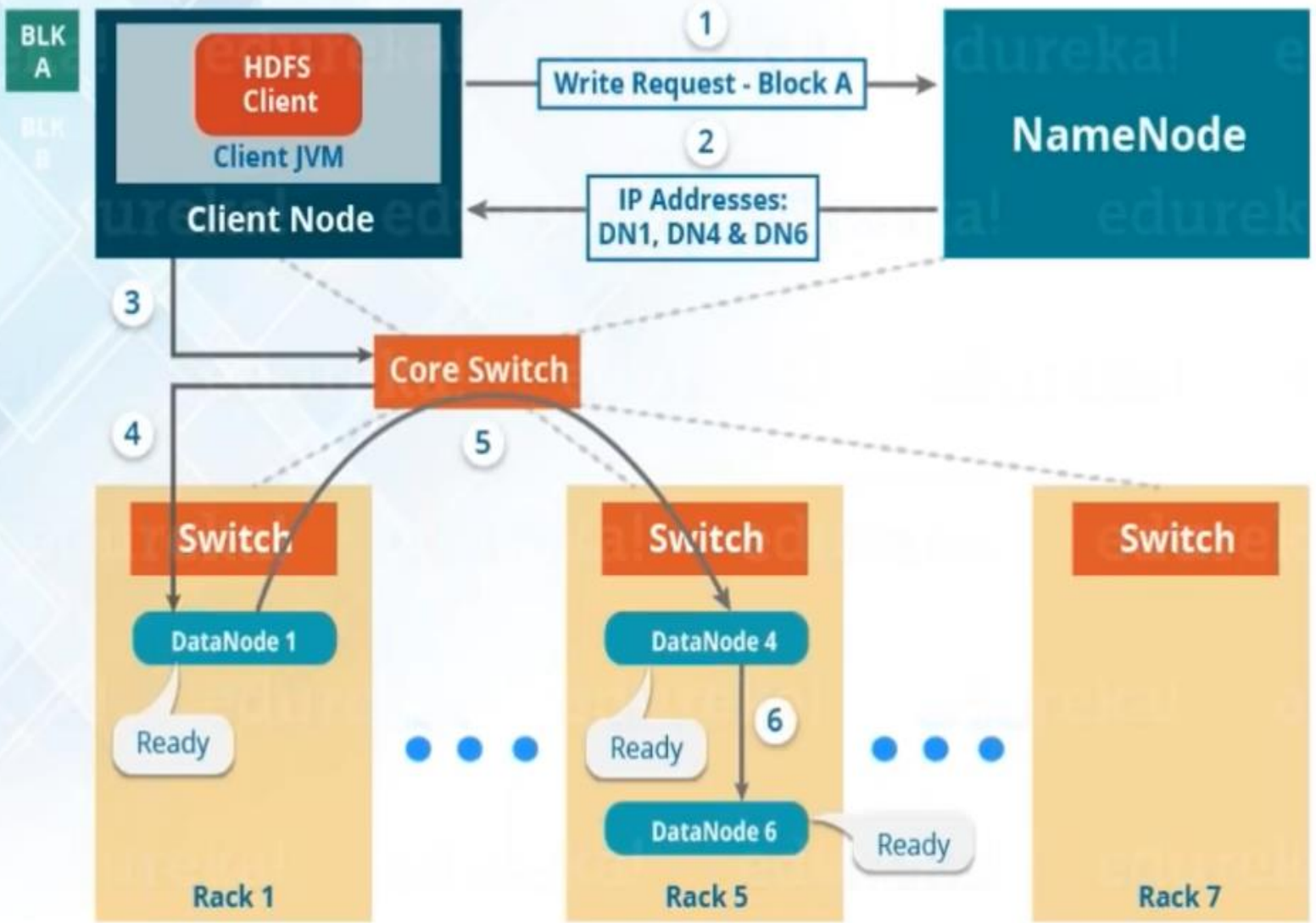
Each data blocks are replicated (thrice by default) and are distributed across different DataNodes

HDFS Write Mechanism

HDFS Write Mechanism – Pipeline Setup

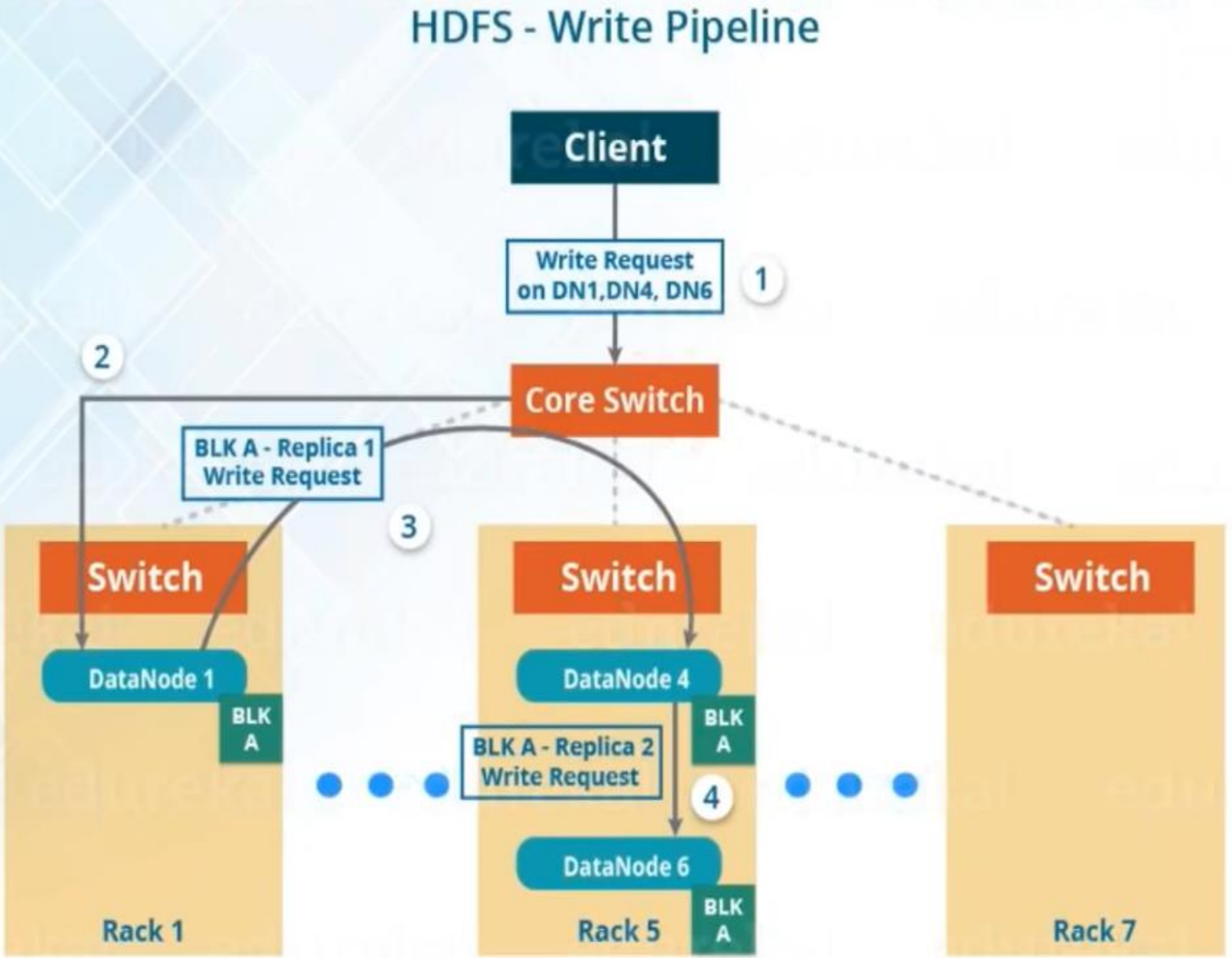
edureka!

Setting up HDFS - Write Pipeline



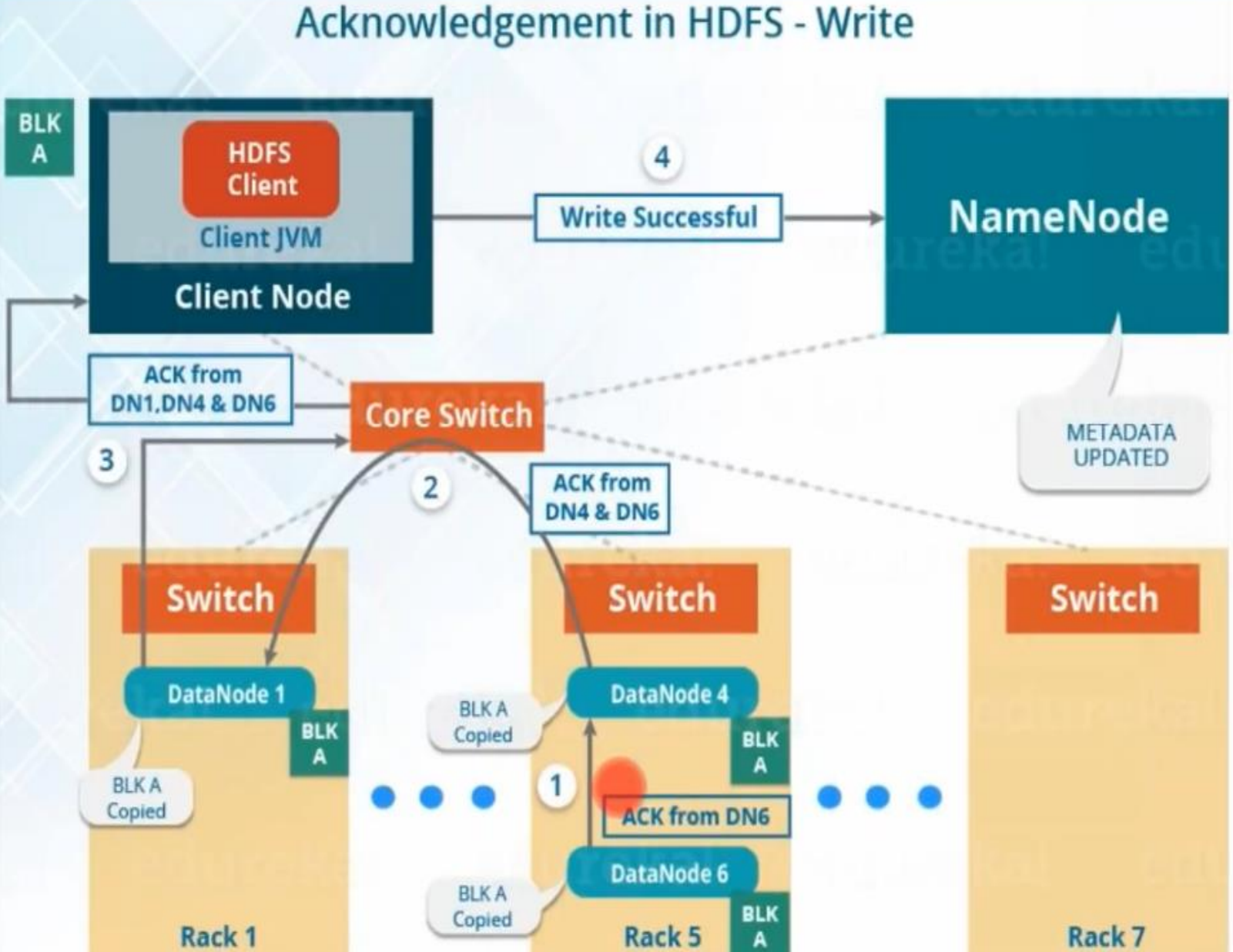
HDFS Write Mechanism – Writing a Block

edureka!



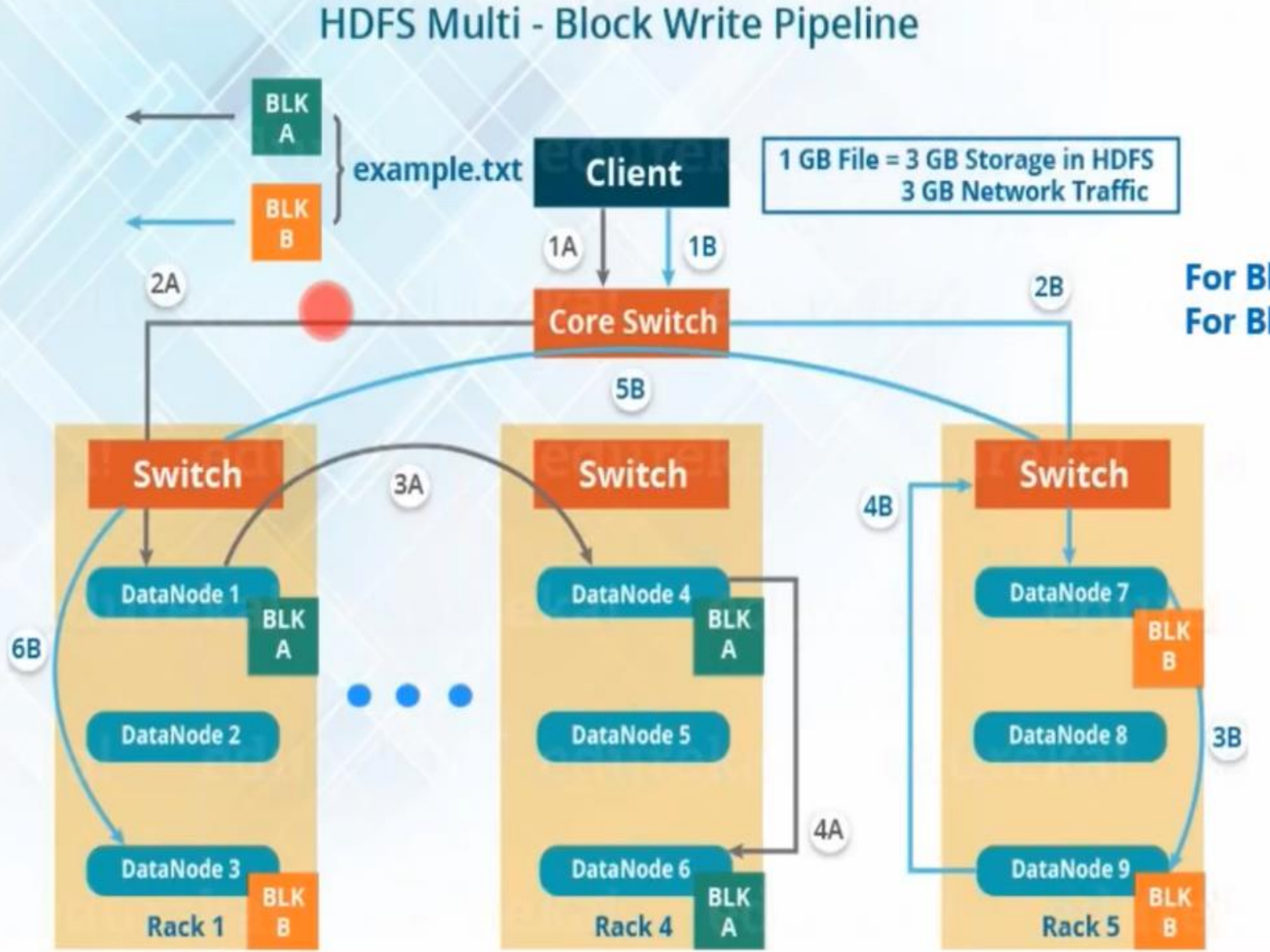
HDFS Write Mechanism - Acknowledgement

edureka!



HDFS Multi-Block Write Mechanism

edureka!

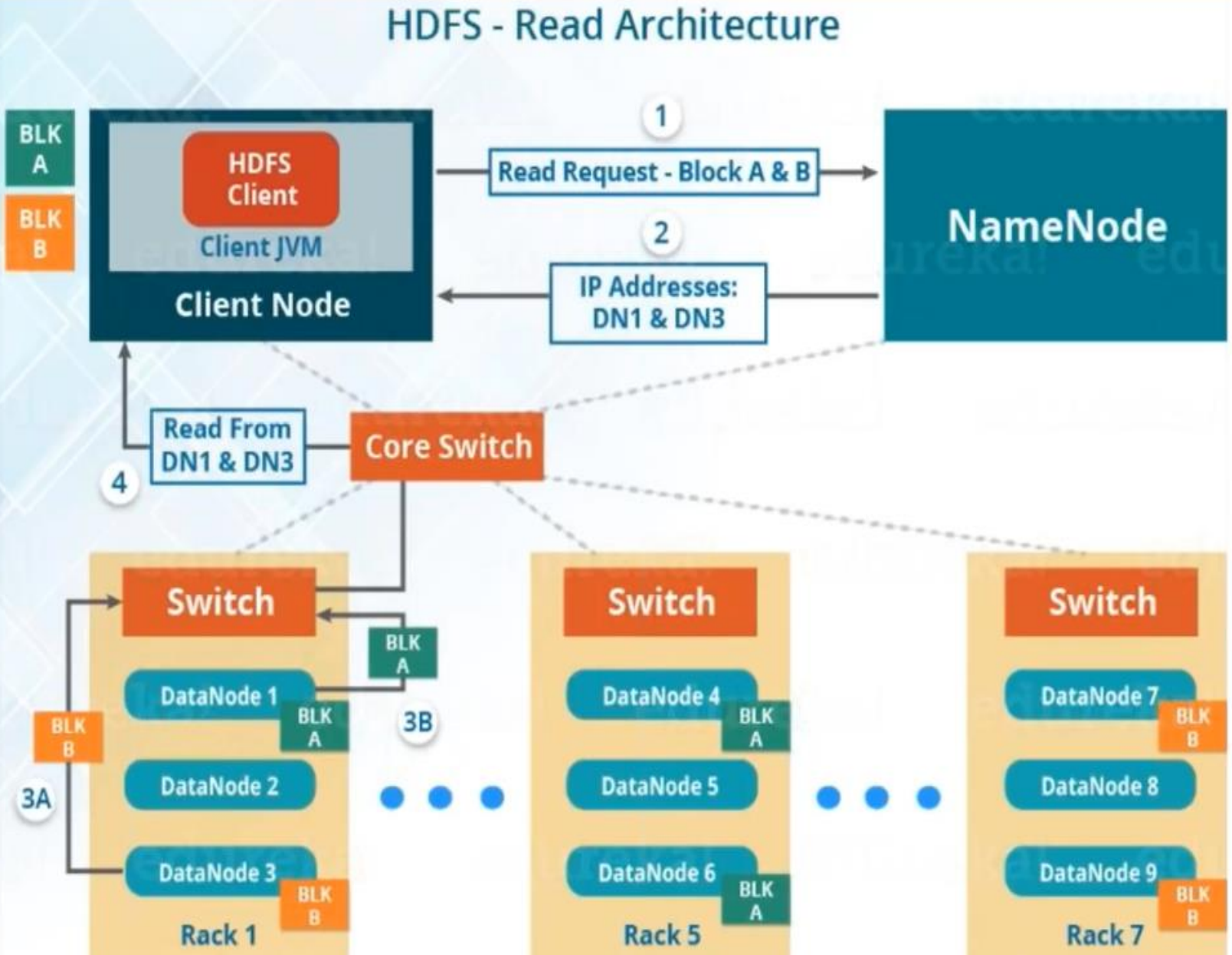


For Block A: 1A -> 2A -> 3A -> 4A
For Block B: 1B -> 2B -> 3B -> 4B -> 5B -> 6B

HDFS Read Mechanism

HDFS Read Mechanism

edureka!



HADOOP CORE COMPONENTS

HDFS

Storage:
Distributed File System

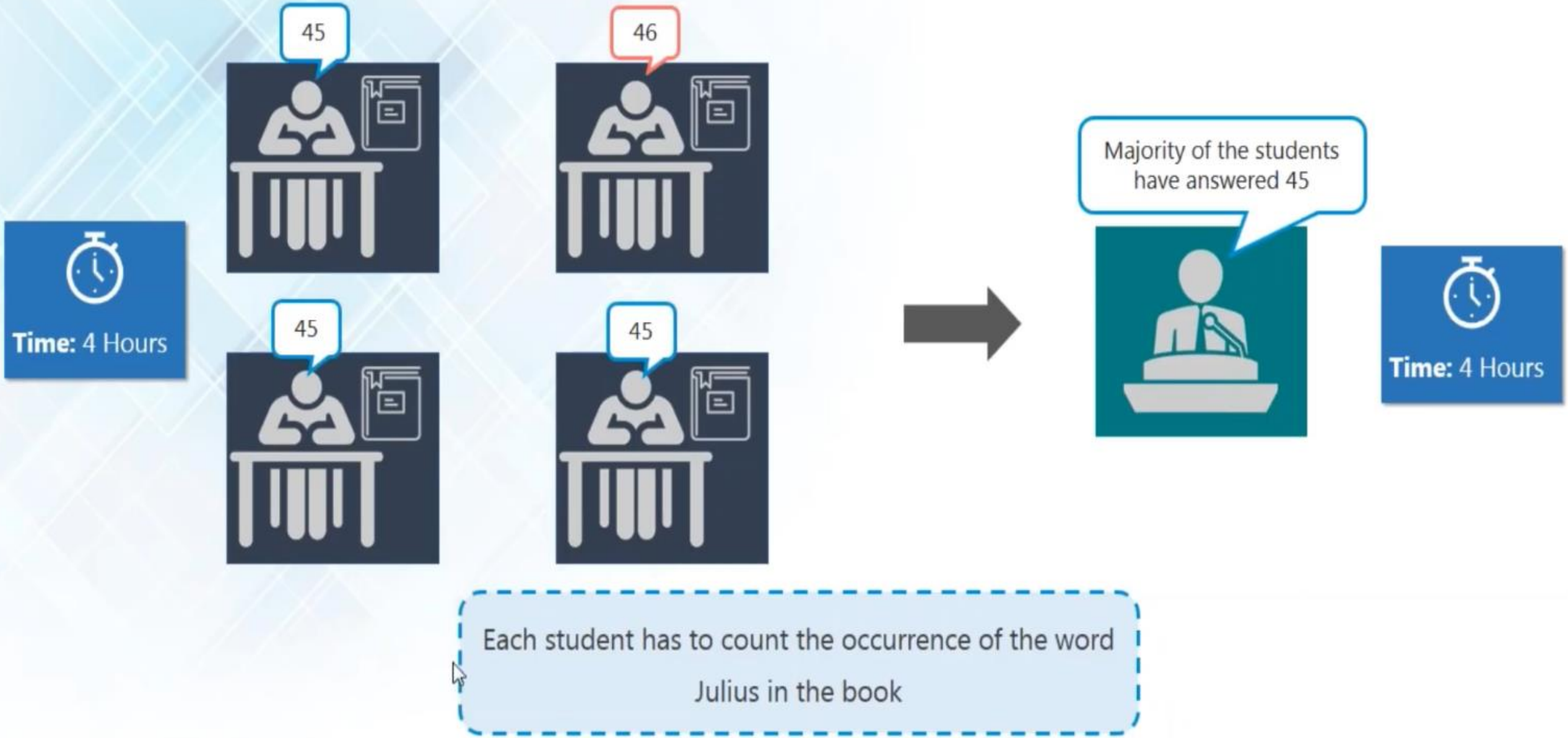
MapReduce

Processing:
Allows parallel & distributed processing

Let us understand
MapReduce with a story

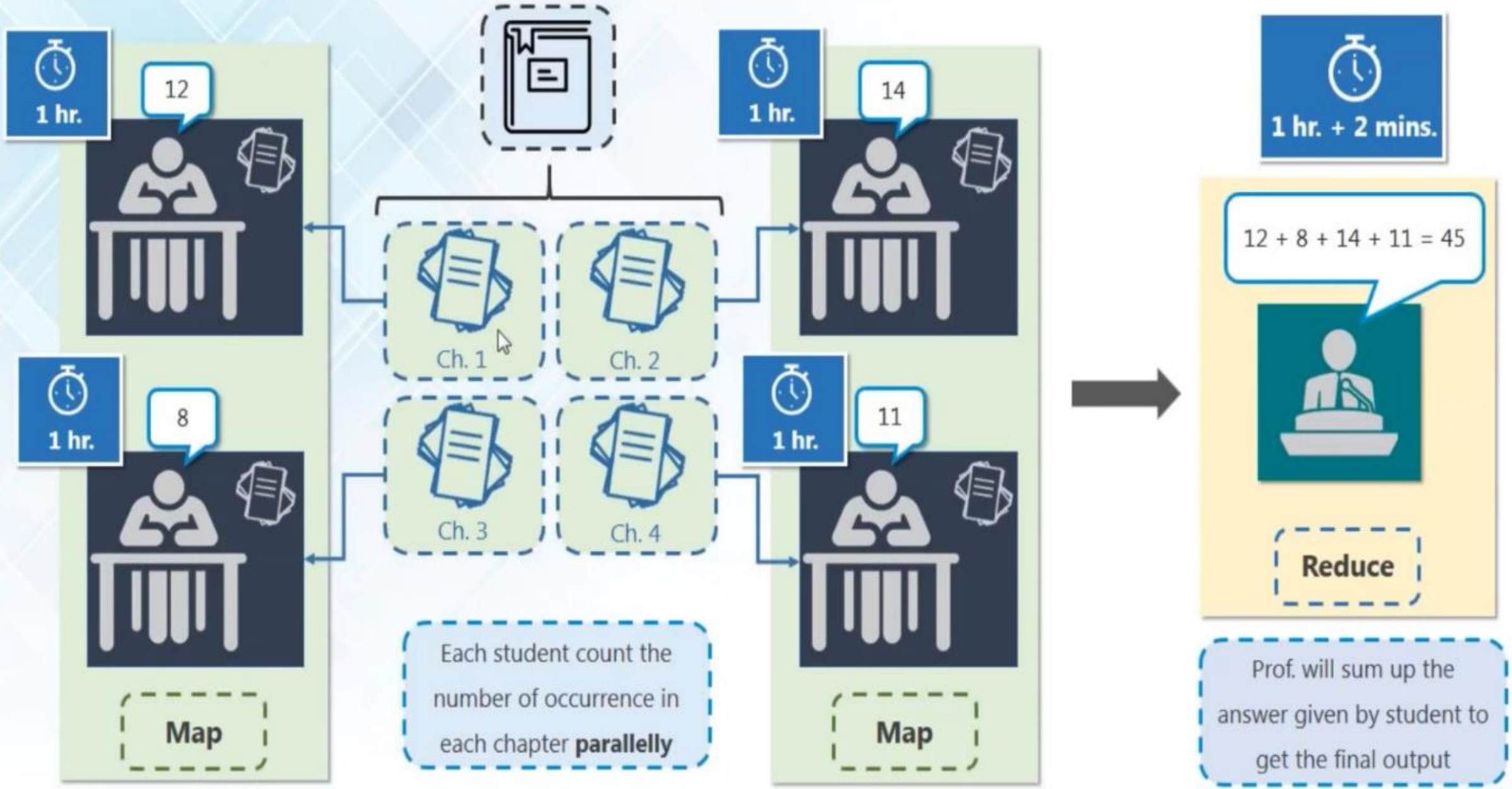
Story of MapReduce

edureka!



Story of MapReduce

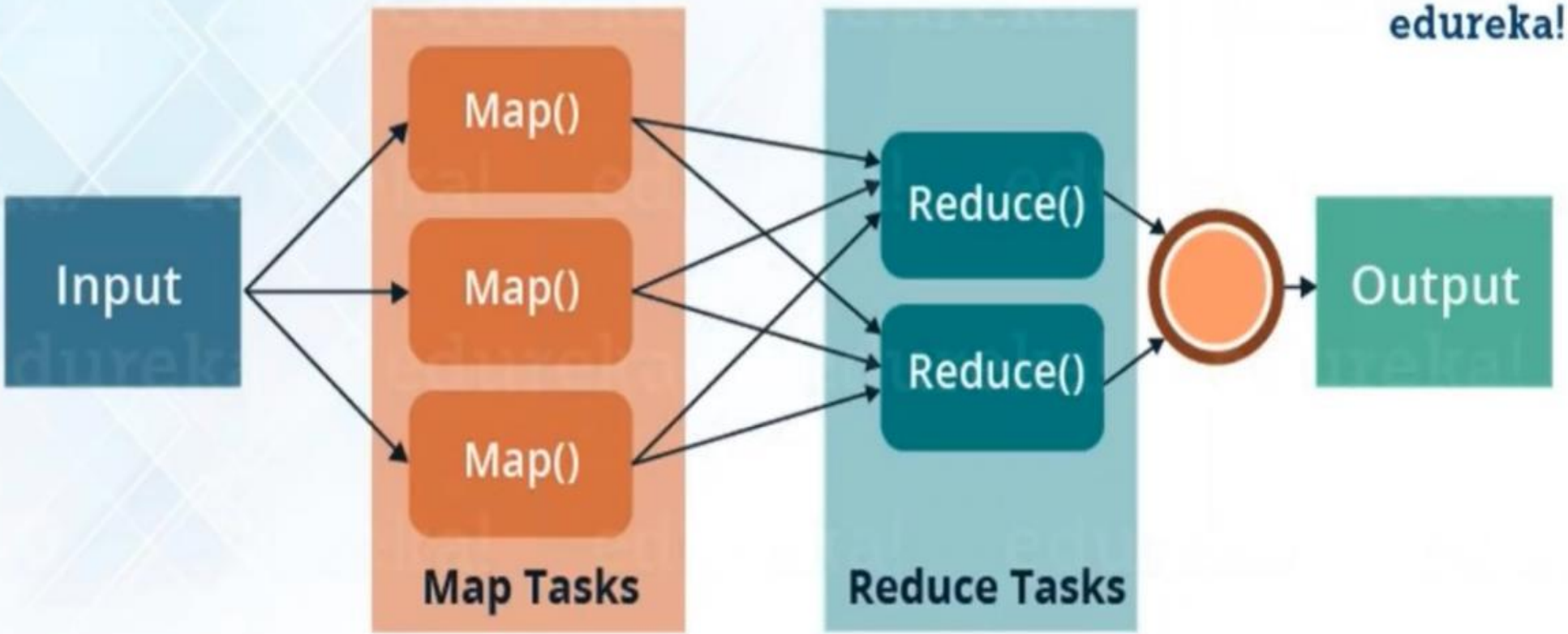
edureka!



What is MapReduce?

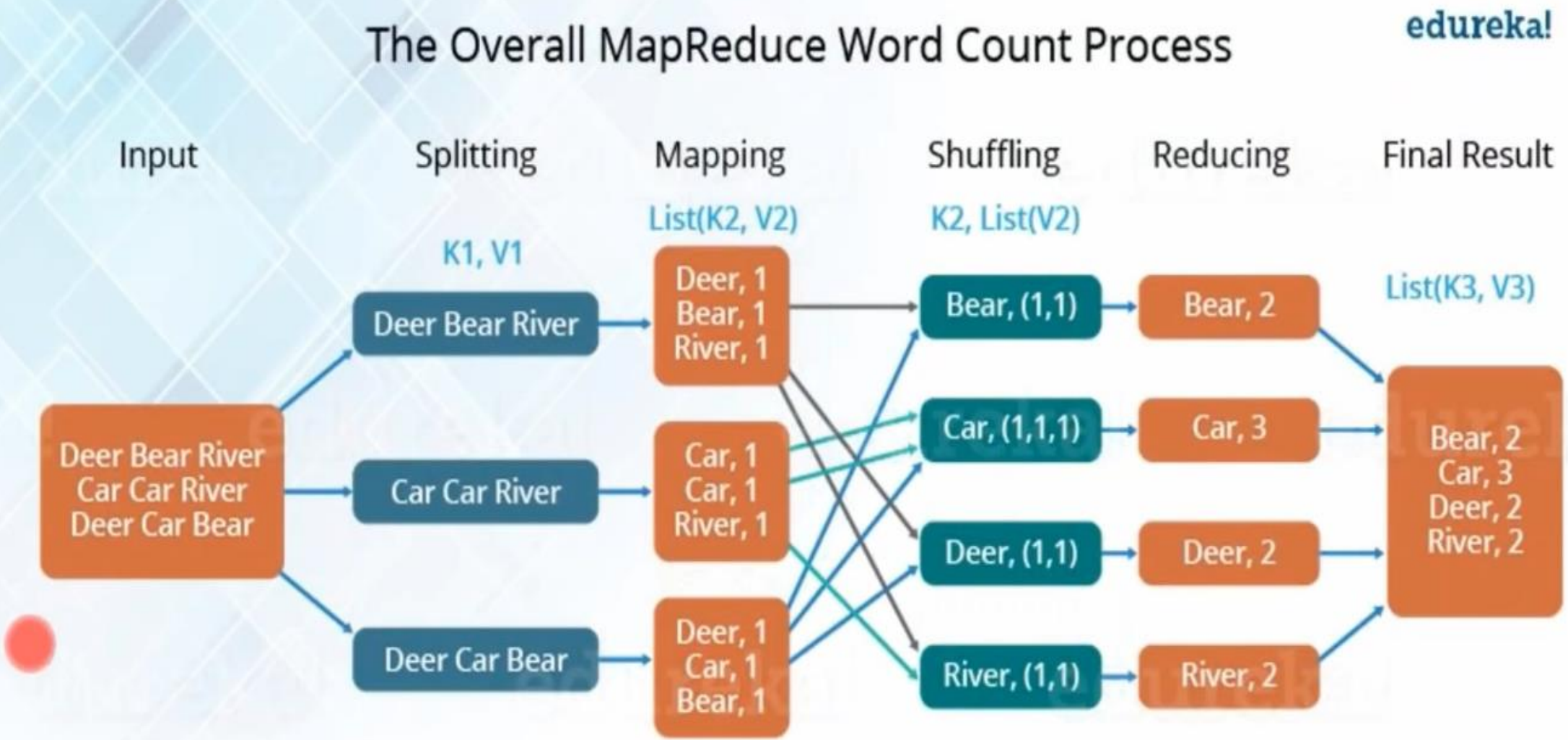
edureka!

MapReduce is a programming framework that allows us to perform distributed and parallel processing on large data sets in a distributed environment



MapReduce Word Count Program

edureka!



MapReduce
Word Count Program

MapReduce Word Count Program

edureka!

Three Major Parts of MapReduce Program:

1

Mapper Code:

You write the mapper logic over here i.e. how map task will process the data to produce the key-value pair to be aggregated

2

Reducer Code:

You write reducer logic here which combines the intermediate key-value pair generated by Mapper to give the final aggregated output

3

Driver Code

You specify all the job configurations over here like job name, Input path, output path, etc.

Mapper Code

edureka!

Byte Offset Type

Mapper Value Input Type

Mapper Key Output Type

Mapper Value Output Type

```
public static class Map extends Mapper<LongWritable,Text,Text,IntWritable> {

public void map(LongWritable key, Text value, Context context) throws IOException,InterruptedException {

String line = value.toString();
StringTokenizer tokenizer = new StringTokenizer(line);
while (tokenizer.hasMoreTokens()) {
value.set(tokenizer.nextToken());
context.write(value, new IntWritable(1));
}
```

Mapper Input:

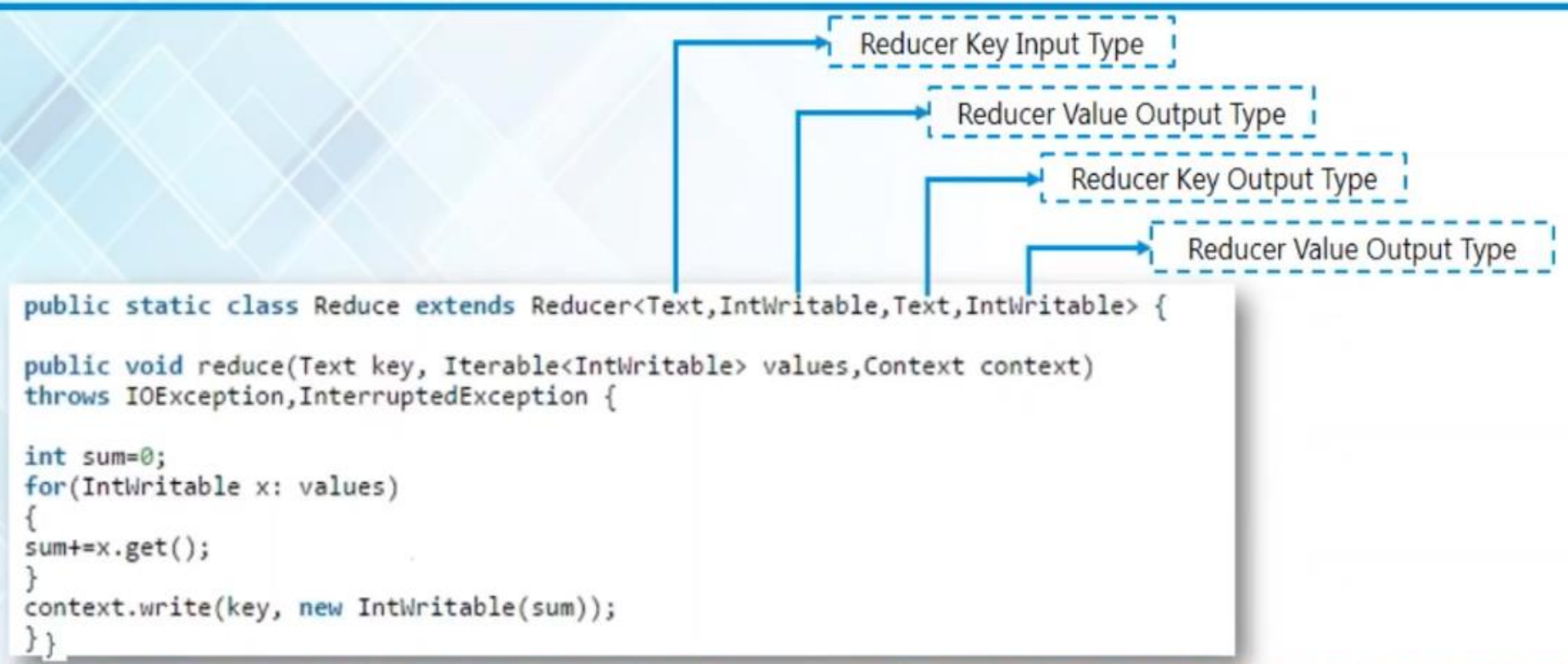
- The key is nothing but the offset of each line in the text file: *LongWritable*
- The value is each individual: *Text*

Mapper Output:

- The key is the tokenized words: Text
- We have the hardcoded value in our case which is 1: IntWritable
- Example – Dear 1, Bear 1, etc.

Reducer Code

edureka!



Reducer Input:

- Keys are unique words which have been generated after the sorting and shuffling phase: Text
- The value is a list of integers corresponding to each key: IntWritable
- Example: Bear, [1, 1], etc.

Reducer Output:

- The key is all the unique words present in the input text file: Text
- The value is the number of occurrences of each of the unique words: IntWritable
- Example: Bear, 2; Car, 3, etc. .

Driver Code

edureka!

In the driver class, we set the configuration of our MapReduce job to run in Hadoop

```
Configuration conf= new Configuration();
Job job = new Job(conf,"My Word Count Program");
job.setJarByClass(WordCount.class);
job.setMapperClass(Map.class);
job.setReducerClass(Reduce.class);
job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);
job.setInputFormatClass(TextInputFormat.class);
job.setOutputFormatClass(TextOutputFormat.class);
Path outputPath = new Path(args[1]);

//Configuring the input/output path from the filesystem into the job
FileInputFormat.addInputPath(job, new Path(args[0]));
FileOutputFormat.setOutputPath(job, new Path(args[1]));
```

- Specify the name of the job , the data type of input/output of the mapper and reducer
- Specify the names of the mapper and reducer classes.
- Path of the input and output folder
- The method setInputFormatClass () is used for specifying the unit of work for mapper
- Main() method is the entry point for the driver

Hadoop Ecosystem

oozie
(Work flow)

HCatalog
Table & schema Management

Pig
(Scripting)

Hive
(Sql Query)

mahout
(Machine Learning)

Drill
(Interactive Analysis)

AVRO
(JSON)

Thrift
(Cross Language Service)

APACHE HBASE
HBASE
(Columnar Store)

Sqoop
(Data Collection)

Zookeeper
(Coordination)

Ambari
Apache Ambari
(Management & Monitoring)

Mapreduce
(Data Processing)

Flume
(Data Collection)

Yarn
(Cluster Resource Management)

HDFS
(Hadoop Distributed File system)

Pig

- ETL library for Hadoop
- Generates MapReduce jobs
- Developed at Yahoo!
- Used the Pig Latin language
- Good for processing on all data

HBase

- Wide-column NoSQL database
- Create tables over HDFS data
- Managing the metastore database

APACHE HBASE

Hive

- SQL-like query language that generates MapReduce code
- Developed at Facebook
- Batch, not interactive
- Good for processing on some part of data
- Used with HBase

Zookeeper

- Centralized service for Hadoop configuration information
- Where data synchronization matters
- Distributed in-memory computation
- Example : Advertise serving in online game

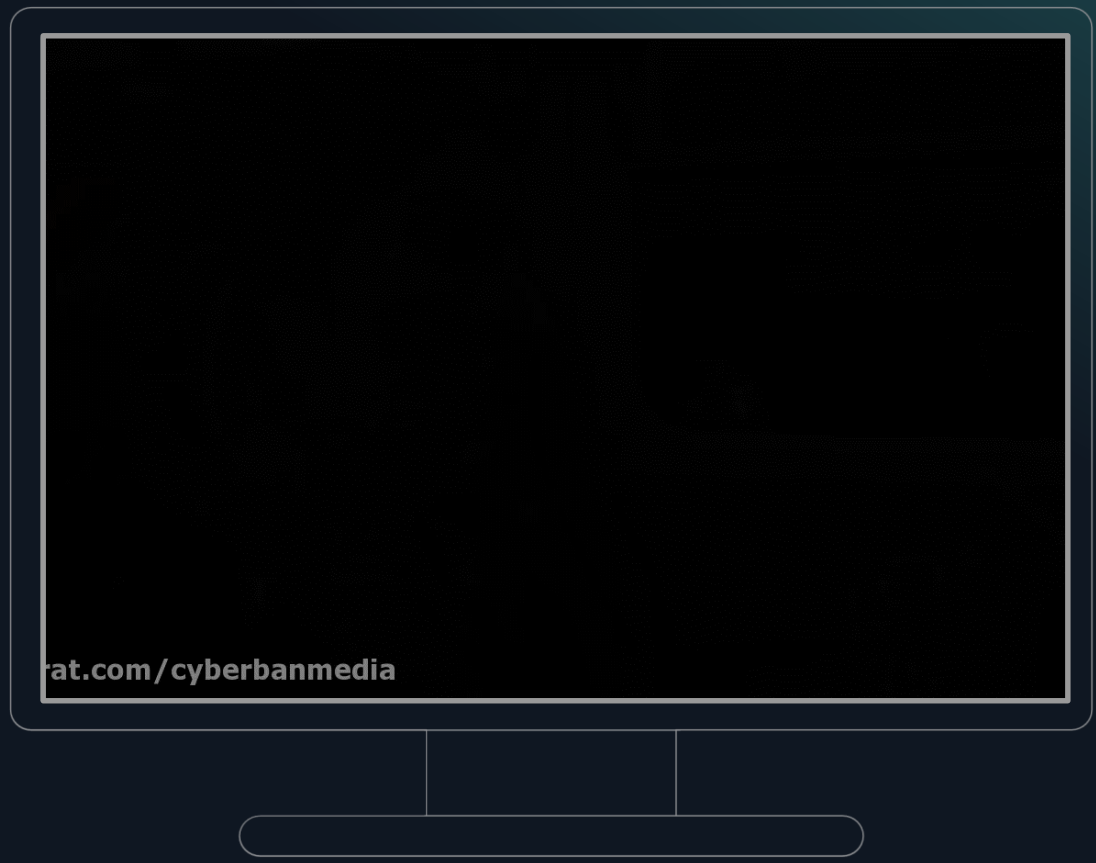
Flume

- Library for working with log data

Mahout

- Library for common machine learning algorithms
- Many data-mining algorithms :
  - Recommendation (Spotify)
  - Classification(spam ID)
  - Clustering(Google News)

At the end...




Data is at the heart of everything we do
DELUSION IS IN YOUR DNA
PRIME DECISION