MDS501
Unit 5: Introduction to Big Data

Dipesh Koirala

Outline

- Introduction to big data and the challenges of handling big data
- Commonly used tools for big data: The map-reduce programming paradigm. Hadoop, HDFS, (py)Spark, Hive.
- Data warehousing and data lake architecture
- Real-time analytics with Apache Kafka

Introduction to Big Data

- Big data is a dataset that *is so huge and complicated* that no typical data management technologies can effectively store or process it.
- Big data is similar to regular data, except it is much larger.
- Big data is a field dedicated to the storage, processing and analysis of large collections of data.
- Big Data simply means datasets containing a large amount of diverse data

Introduction to Big Data

- **A lot of Data is Collected** (from TB to PB to ZB to….)
- Every click on the internet, every bank transaction, every video we watch on YouTube, every email we send, every like on our Instagram post makes up data for tech companies.

- With such a massive amount of data being collected, it only makes sense for **companies to use this data** to understand their customers and their behavior better.
- Approximately 402.74 million terabytes of data are created daily, which is equivalent to about 147 zettabytes per year.

Characteristics of Big Data

Volume:

- *Magnitude of data.* Refers to the vast amounts of data generated every second.
- If we take all the data generated in the world between the beginning of time and 2008, the same amount of data will soon be generated every minute.

Velocity:

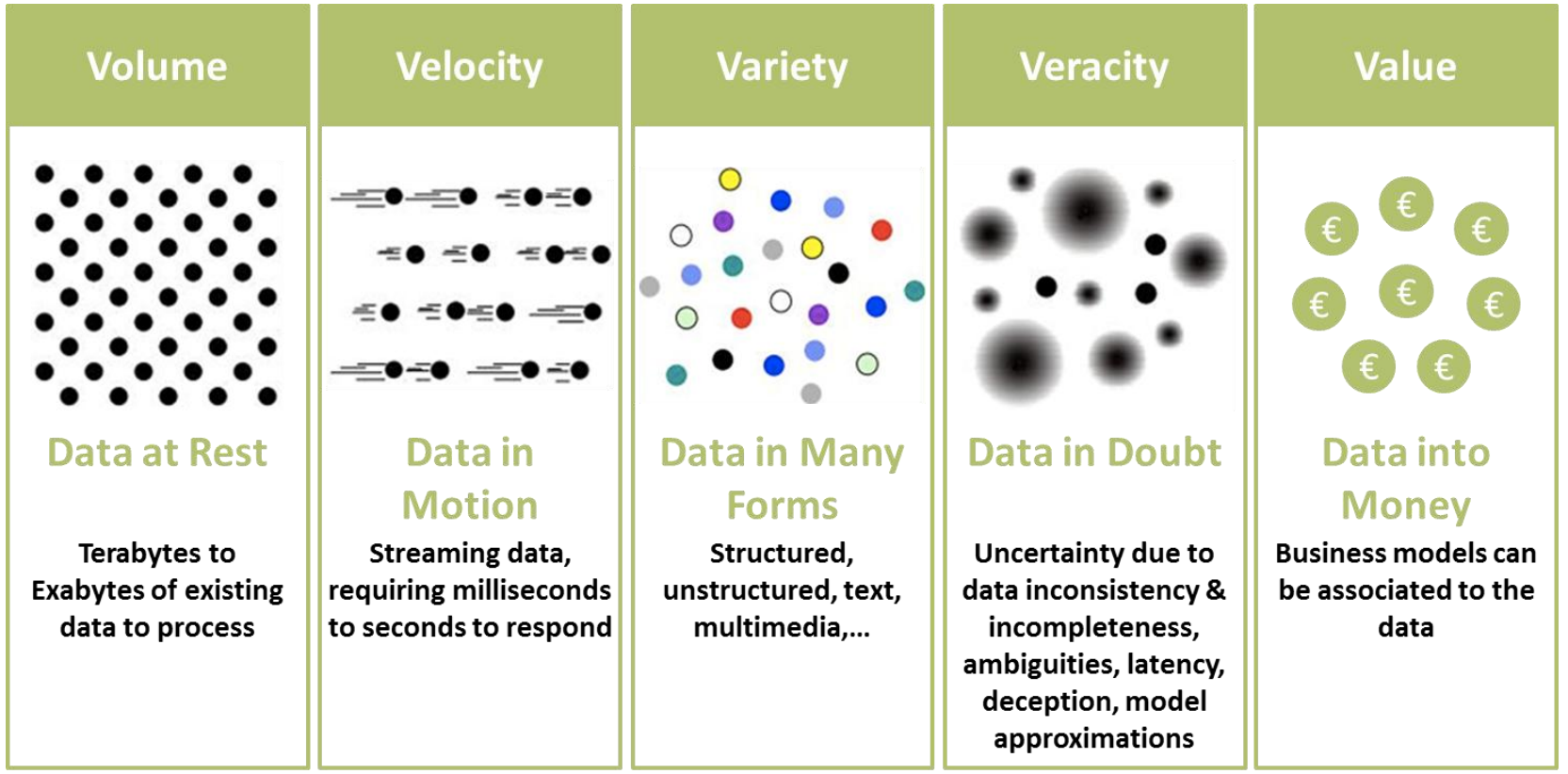
- Refers *to the rate or speed at which new data is generated* and the speed at which data moves around. Data comes in at a high rate from networks, social media, mobile phones, and other sources

Variety:

- Refers to different forms of data. E.g., Text, images, logs, social media content, audio, videos.

Characteristics of Big Data

- 5V's



Adapted by a post of Michael Walker on 28 November 2012

Characteristics of Big Data

Veracity:

- refers to the *quality of data sets and how trustworthy they are*. If they aren't fixed through data cleansing processes, poor data quality can lead to incorrect decisions or misinterpreted trends.

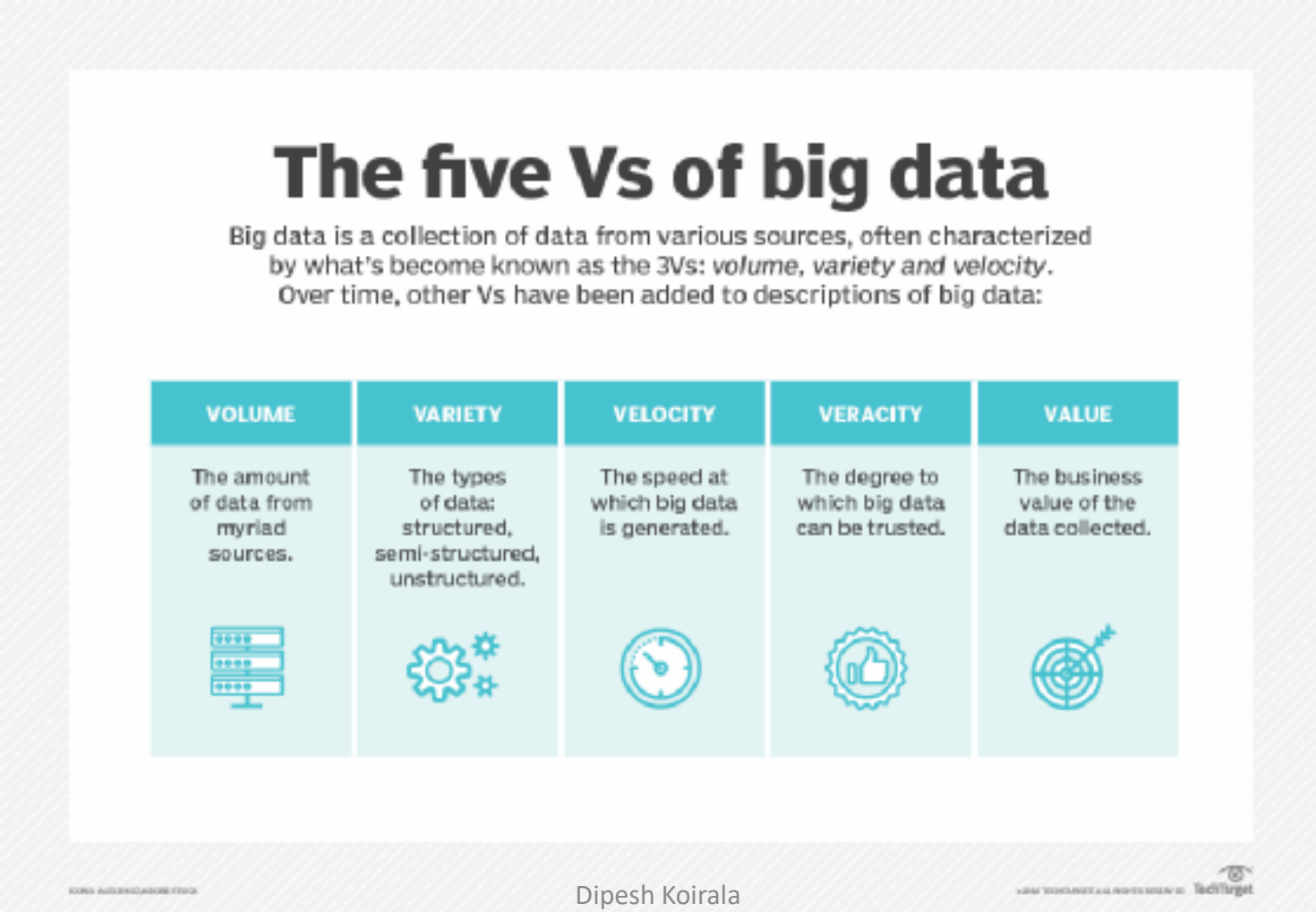
Value:

- Refers to usefulness of the data. Not all the data that's collected *has real business value or benefits*. Organizations must identify the relevant parts of data that deliver value.

- **Veracity** focus on ensuring data is accurate and reliable and **Value** focuses on Extracting meaningful insights and benefits i.e., *unlock the full potential*

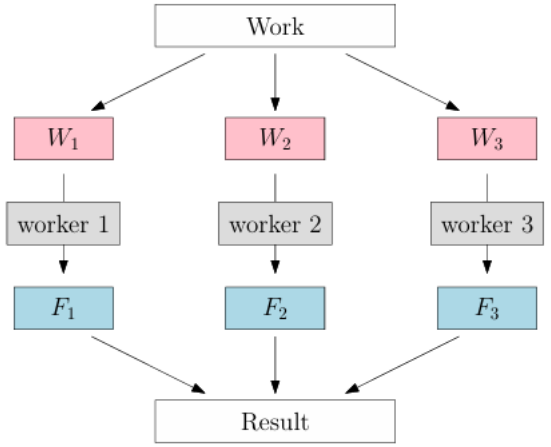
Characteristics of Big Data

- 5V's



Map-Reduce Programming Paradigm

- MapReduce is a programming model developed by Google in 2004, for processing large datasets in a distributed manner.
- It lead to the development of distributed execution framework.
- There are a number of implementations of this model, including Google's approach, programmed in C++, and Apache's Hadoop implementation, programmed in Java.



Map-Reduce Programming Paradigm

The name "MapReduce" refers to the 2 tasks or functions:

Map() and Reduce()

- that the model performs to help “chunk” a large data processing task into many smaller tasks that can run faster in parallel.

- Map(): takes the input data and process to generate intermediate key/value pairs
- Reduce(): takes the output from Map(), merge all intermediate values associated with the same key and produce final set of key value pairs

How MapReduce Works

i. Input: A MapReduce application accepts input data, which can include structured or unstructured data.

ii. Splitting: The input data is split into smaller blocks. These blocks are distributed to mappers

iii. Mapping: the map function processes the data it receives, converting the data into key/value pairs.

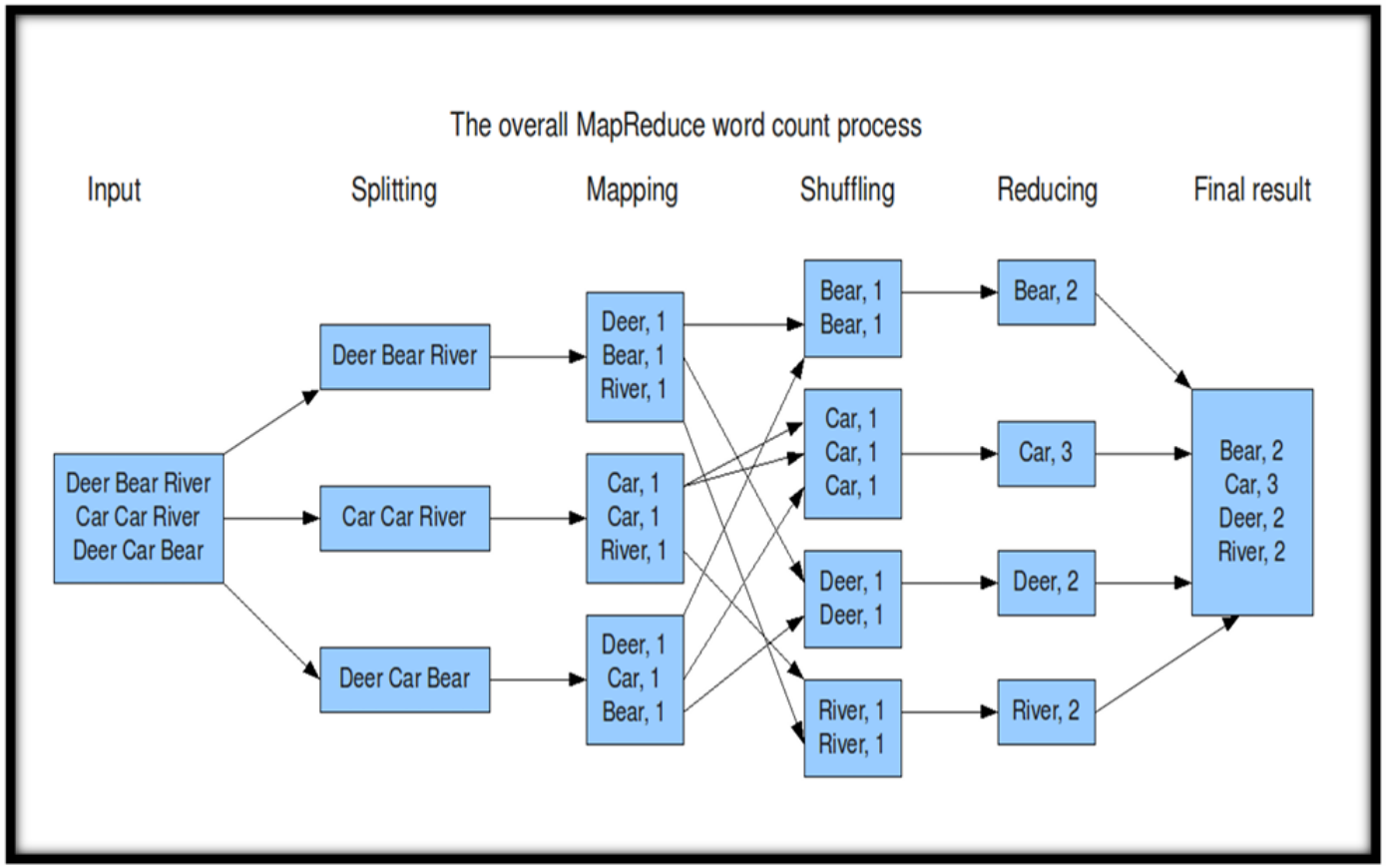
iv. Shuffling: sorts the map outputs and assigns all key/value pairs with the same "key" (topic) to the same reducer.

v. Reducing: Reduce functions process the key/value pairs that the mappers emit. This can involve merging, tabulating or performing other operations on the data, depending on the kind of processing required.

vi. Result

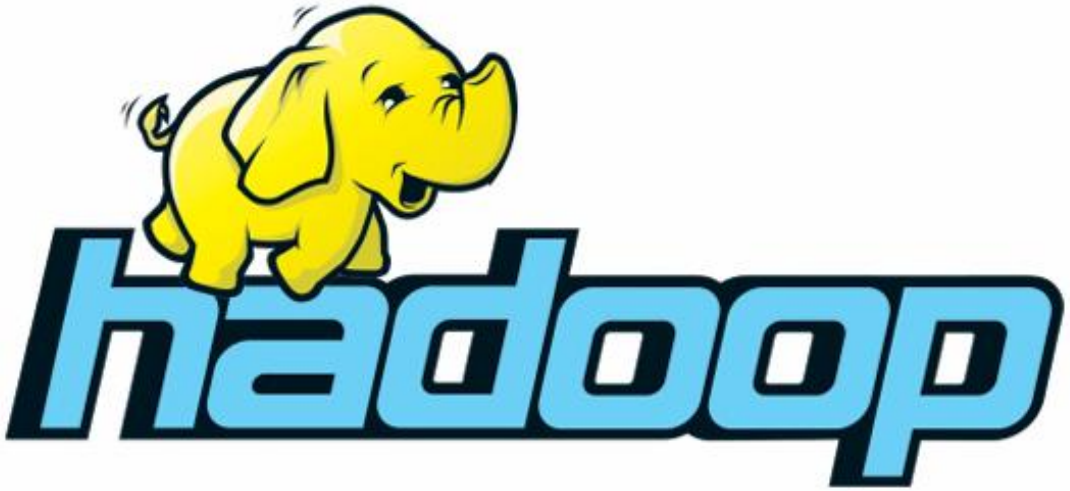
How MapReduce Works

- Solving Word Count Problem through Map-Reduce Paradigm



Hadoop

- Open source software framework designed *for storage and processing of large scale data on clusters* of commodity hardware



Hadoop

- **Created by Doug Cutting** and Mike Carafella in 2005.

- Based on work done by Google in the early 2000s
  - "The Google File System" in 2003
  - "MapReduce: Simplified Data Processing on Large Clusters" in 2004

- Cutting named the program after his son's toy elephant.

Hadoop

Components

i. Hadoop Common: a set of shared programming libraries used by the other modules

ii. Hadoop Distributed File System (HDFS): a Java-based file system to store data across multiple machines

iii. MapReduce framework: a programming model to process large sets of data in parallel

iv. YARN (Yet Another Resource Negotiator): handles the management and scheduling of resource requests in a distributed environment

Hadoop

- The *software framework* that supports HDFS, MapReduce and other related entities is called the **project Hadoop or simply Hadoop.**
- This is open source and distributed by Apache.

HDFS

- Responsible *for storing data on the cluster*
- Provides redundant storage for massive amounts of data
- Data files are split into blocks and distributed across the nodes in the cluster
- Each block is replicated multiple times

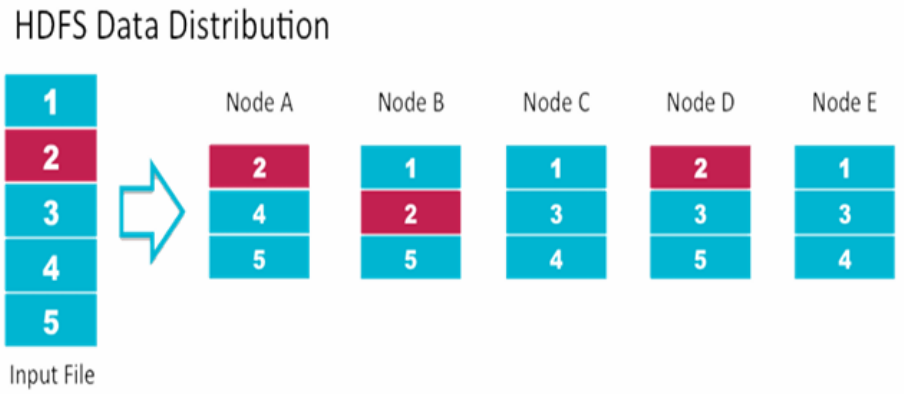
Challenges of Handling Big Data

HDFS

- is a java based file system that provides *scalable, fault tolerance, reliable and cost efficient data storage* for Big data.

Data Replication

- Default data replication is 3-fold



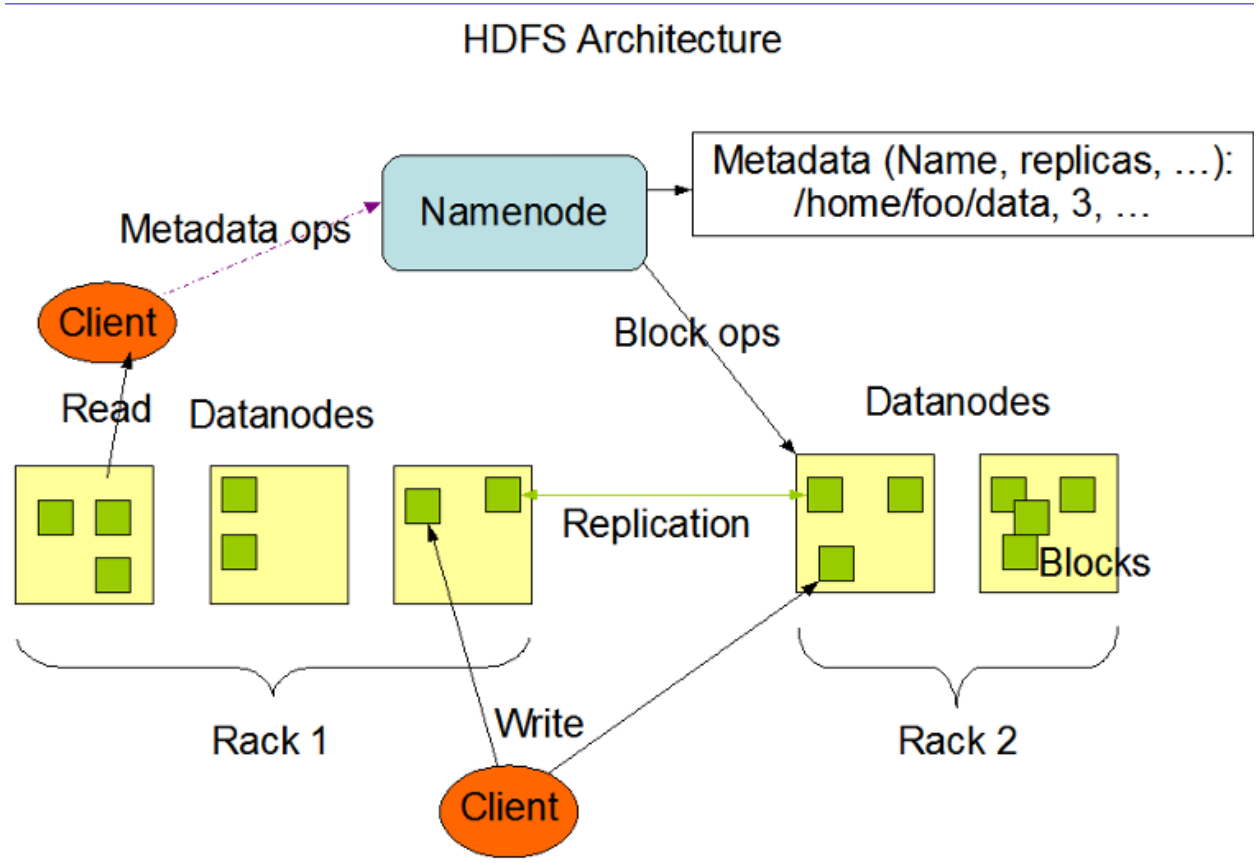
Hadoop

HDFS

- HDFS cluster is composed of a NameNode and various DataNodes
- NameNode
  - a server which holds all the metadata regarding the stored files
  - manages incoming file system operations
  - maps data blocks (parts of files) to DataNodes
- DataNode
  - handles file read and write requests
  - create, delete and replicate data blocks amongst their disk drives
  - continuously loop, asking the NameNode for instructions.
- Note: size of 1 data block is typically 128 megabytes

Hadoop

HDFS


HDFS Architecture
Namenode
Metadata (Name, replicas, …):
/home/foo/data, 3, …
Metadata ops
Client
Block ops
Read
Datanodes
Datanodes
Replication
Blocks
Rack 1
Write
Rack 2
Client

Hadoop

Yet Another Resource Negotiator (YARN)

- distributes a MapReduce program across different nodes and takes care of coordination

- Three important services
  - ResourceManager: a global YARN service that receives and runs applications (e.g., a MapReduce job) on the cluster
  - JobHistoryServer: keeps a log of all finished jobs
  - NodeManager: responsible to oversee resource consumption on a node

Hadoop

Advantages:

- Open Source
- Scalable
- Fault-Tolerant
- Cost-Effective

Topics

- (py)Spark, Hive.
- Data warehousing and data lake architecture
- Real-time analytics with Apache Kafka

Spark

- Apache Spark is a fast, open source, large-scale data-processing engine often used for machine learning (ML) and artificial intelligence (AI) applications.
- Spark was developed to address shortcomings in MapReduce as it can be slow and inefficient.

MapReduce requires

- replication: maintaining multiple copies of data in different locations
- coordinating access to resources used by more than one program
- intense I/O: input/output of disk storage

Spark

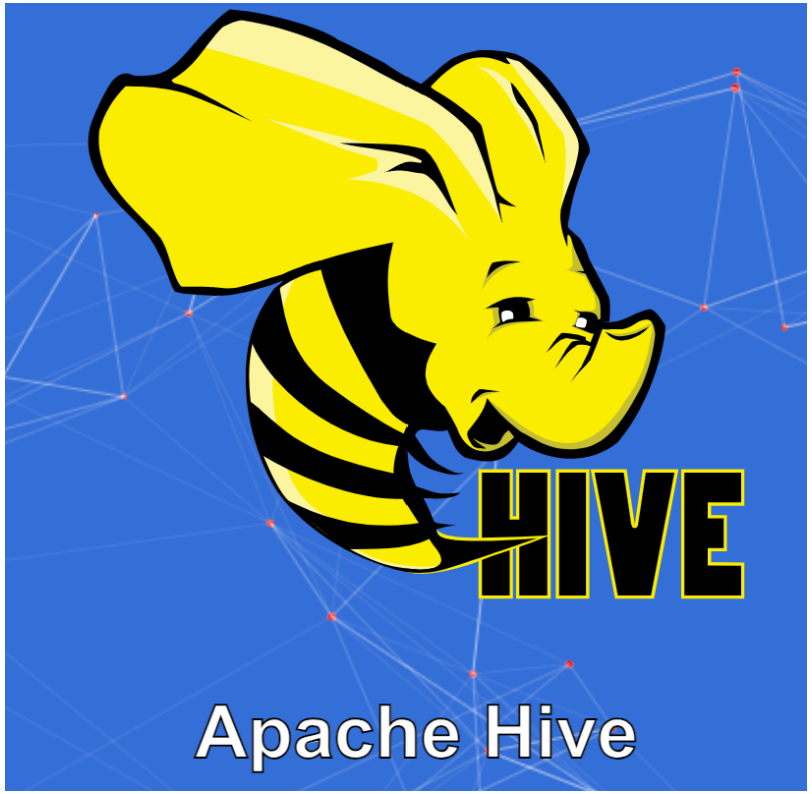
- Spark specifically reduces unnecessary processing.
- Whereas MapReduce writes intermediate data to disk, Spark uses RDDs to cache and compute data in memory.
- The result is that Spark's analytics engine can process data 10–100 times faster than MapReduce.

Spark

- Apache Spark is written in Scala programming language.
- PySpark has been released in order to support the collaboration of Apache Spark and Python
- it actually is a Python API for Spark.
- With PySpark, Python and SQL-like commands can be written to manipulate and analyze data in a distributed processing environment.

Hive

- Apache Hive is a data warehousing and SQL-like query language for Hadoop.



- Hive was created to allow non-programmers familiar with SQL to work with petabytes of data, using a SQL-like interface called HiveQL.

- Developed by Facebook, it is now a part of the Apache Software Foundation and used by numerous organizations for big data processing.

Hive

Scenario

- As data is stored in the Apache Hadoop Distributed File System (HDFS) wherein data is organized and structured,

- Apache Hive helps in processing this data and analyzing it producing data-driven patterns and trends.

Hive: It is a platform used to develop SQL type scripts to do MapReduce operations.

Hadoop

Features:

- It is designed for OLAP.
- It provides SQL type language for querying called HiveQL or HQL.
- It is familiar, fast, scalable, and extensible.

Data Warehouse and Data Lake

Structured, Semi-structured and Unstructured Data

Unstructured data

The university has 5600 students.
John's ID is number 1, he is 18 years old and already holds a B.Sc. degree.
David's ID is number 2, he is 31 years old and holds a Ph.D. degree. Robert's ID is number 3, he is 51 years old and also holds the same degree as David, a Ph.D. degree.

Semi-structured data

```
<University>
 <Student ID="1">
  <Name>John</Name>
  <Age>18</Age>
  <Degree>B.Sc.</Degree>
 </Student>
 <Student ID="2">
  <Name>David</Name>
  <Age>31</Age>
  <Degree>Ph.D. </Degree>
 </Student>
....
</University>
```

Structured data

ID	Name	Age	Degree
1	John	18	B.Sc.
2	David	31	Ph.D.
3	Robert	51	Ph.D.
4	Rick	26	M.Sc.
5	Michael	19	B.Sc.

Data Warehouse and Data Lake

Data Warehouse

- is a *centralized repository designed for* ***analytical processing*** of large volumes of structured historical data.
- It aggregates data from various sources and enables complex queries and reports.
- Optimized for **analytical operations** (OLAP - Online Analytical Processing).

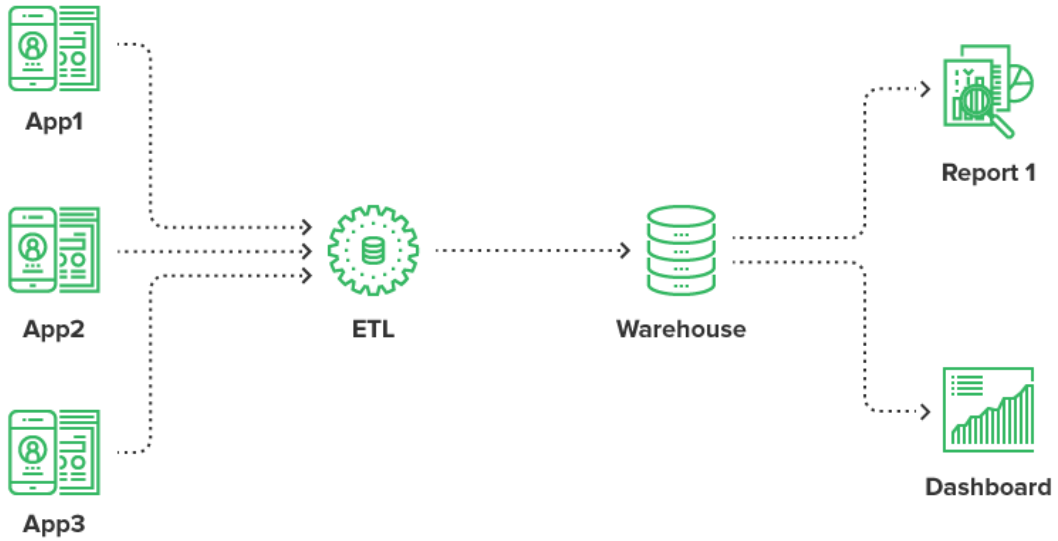
Data Warehouse and Data Lake

Data Warehouse

- Used for *historical analysis, reporting, and decision-making.*

- Data is typically cleaned, transformed, and loaded (ETL process).
- Has predefined schema

- Common use cases: Business Intelligence (BI), analytics, reporting dashboards.
- Technologies: **Amazon Redshift, Google BigQuery, Snowflake, Microsoft Azure Synapse.**



Data Warehouse and Data Lake

Data Lake

- is a vast storage repository that can store ***structured, semi-structured, and unstructured data*** *in its raw format.*

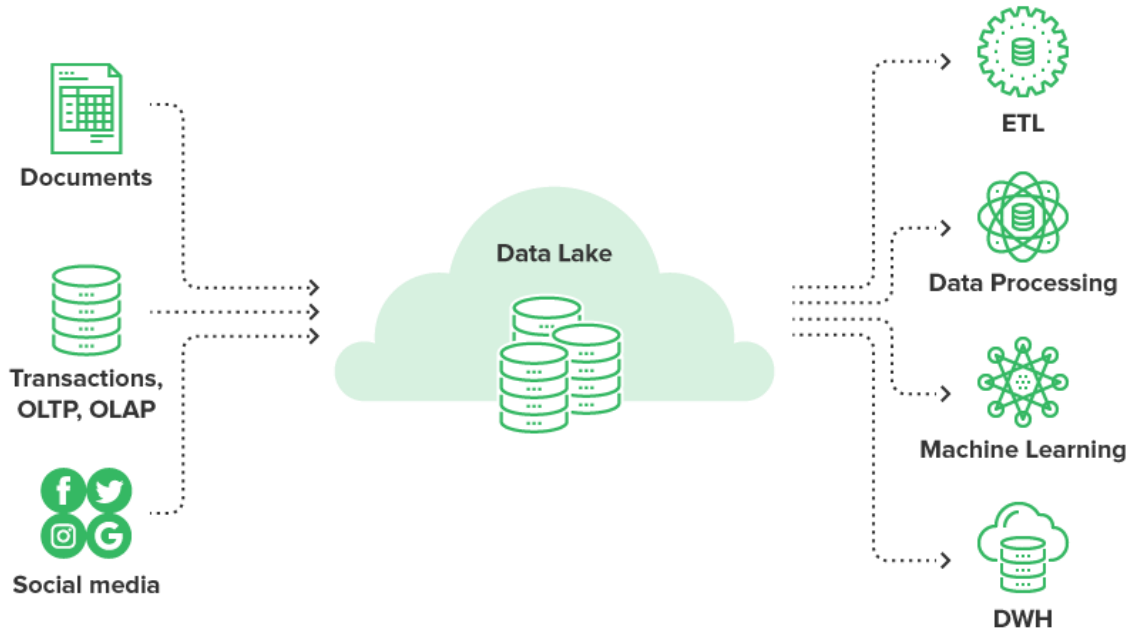
- It is designed for big data and advanced analytics.

- Supports advanced analytics, including machine learning and data mining.
- Schema on read

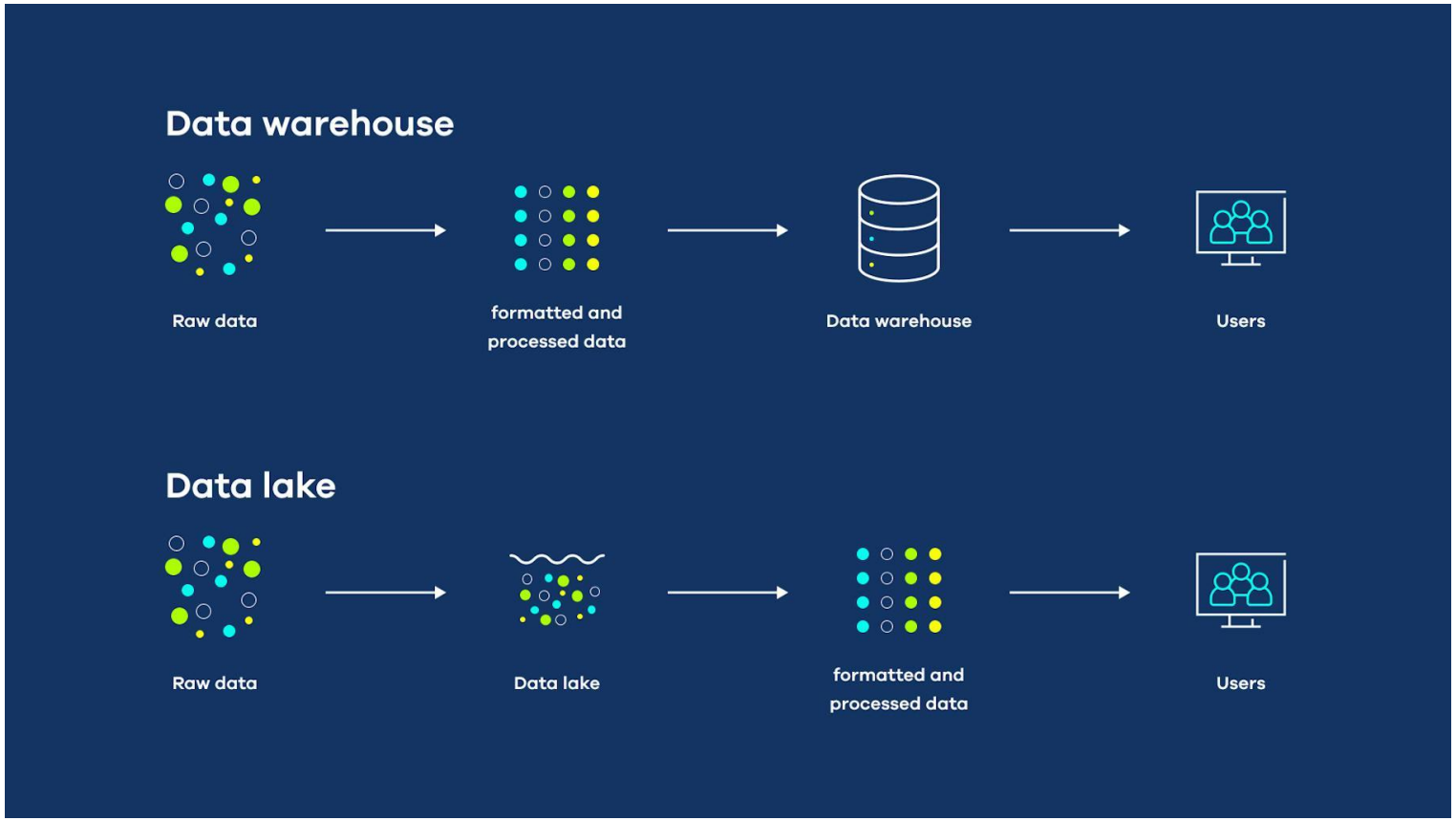
Data Warehouse and Data Lake

Data Lake

- Stores **all types of data:** structured (tables), semi-structured (JSON, XML), and unstructured (images, audio, video).
- Data is ingested in its native/raw format and processed later (ELT process).
- Technologies: **Apache Hadoop, AWS S3, Azure Data Lake, Google Cloud Storage**



Data Warehouse and Data Lake

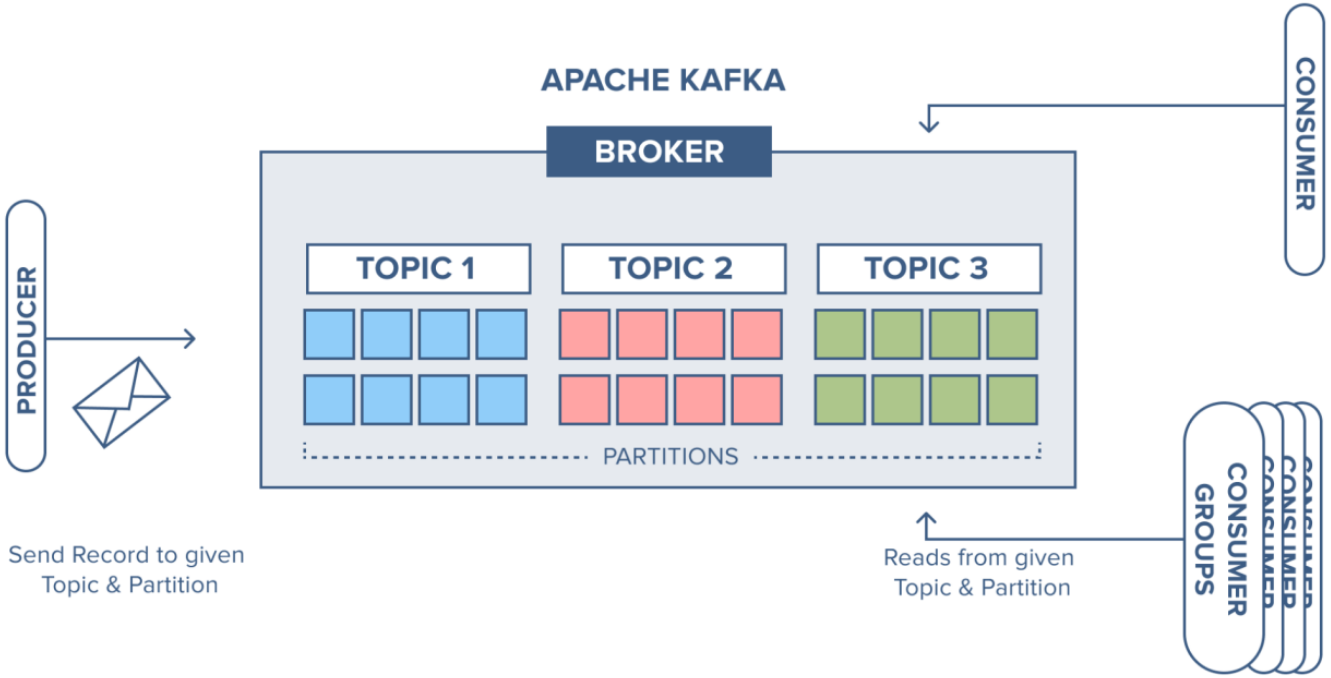


Real – time analytics with Kafka

- Real-time analytics is about analyzing data as soon as it's generated.

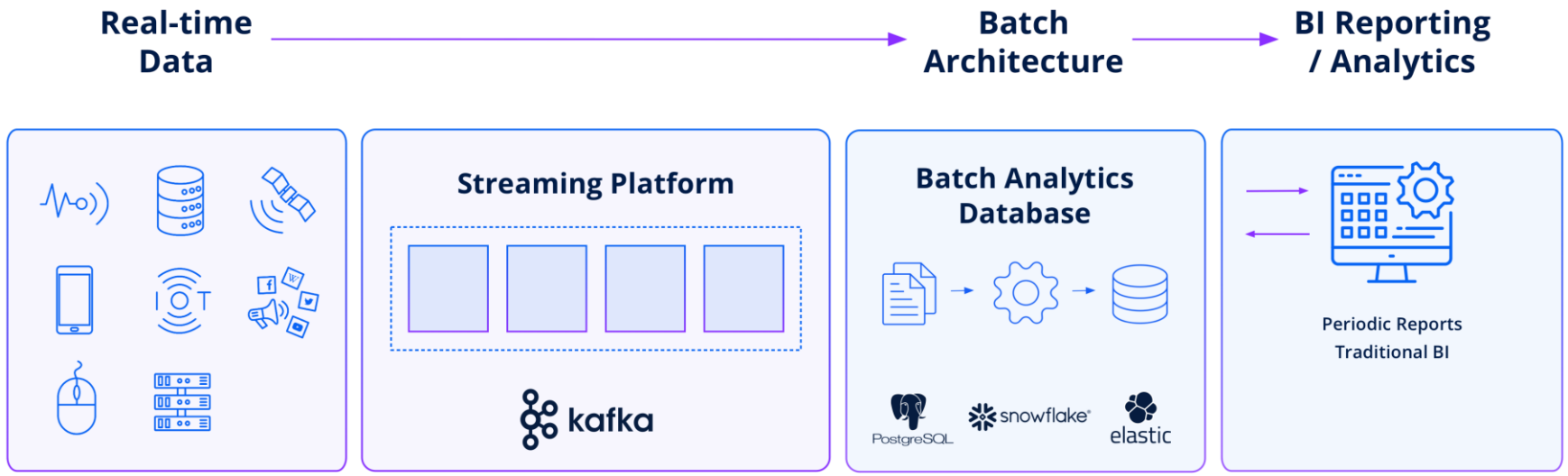
- Kafka is open-source software that provides a framework for storing, reading, and analyzing streaming data.
- Kafka is designed to be run in a “distributed” environment, which means that rather than sitting on one user’s computer

Real – time analytics with Kafka



Real - time analytics with Kafka

- Uber – Tracks millions of rides in real-time.
- Netflix – Processes user clicks for recommendations.
- Banking – Fraud detection by analyzing transactions instantly.
- PayPal: Uses Kafka to detect $1M+/day in fraud attempts.



References

https://www.databricks.com/glossary/mapreduce

https://www.ibm.com/think/topics/resilient-distributed-dataset

https://www.turing.com/resources/real-time-analytics-with-apache-kafka

End of Unit 5

Thank you