An Introduction to Big Data Concepts and Terminology

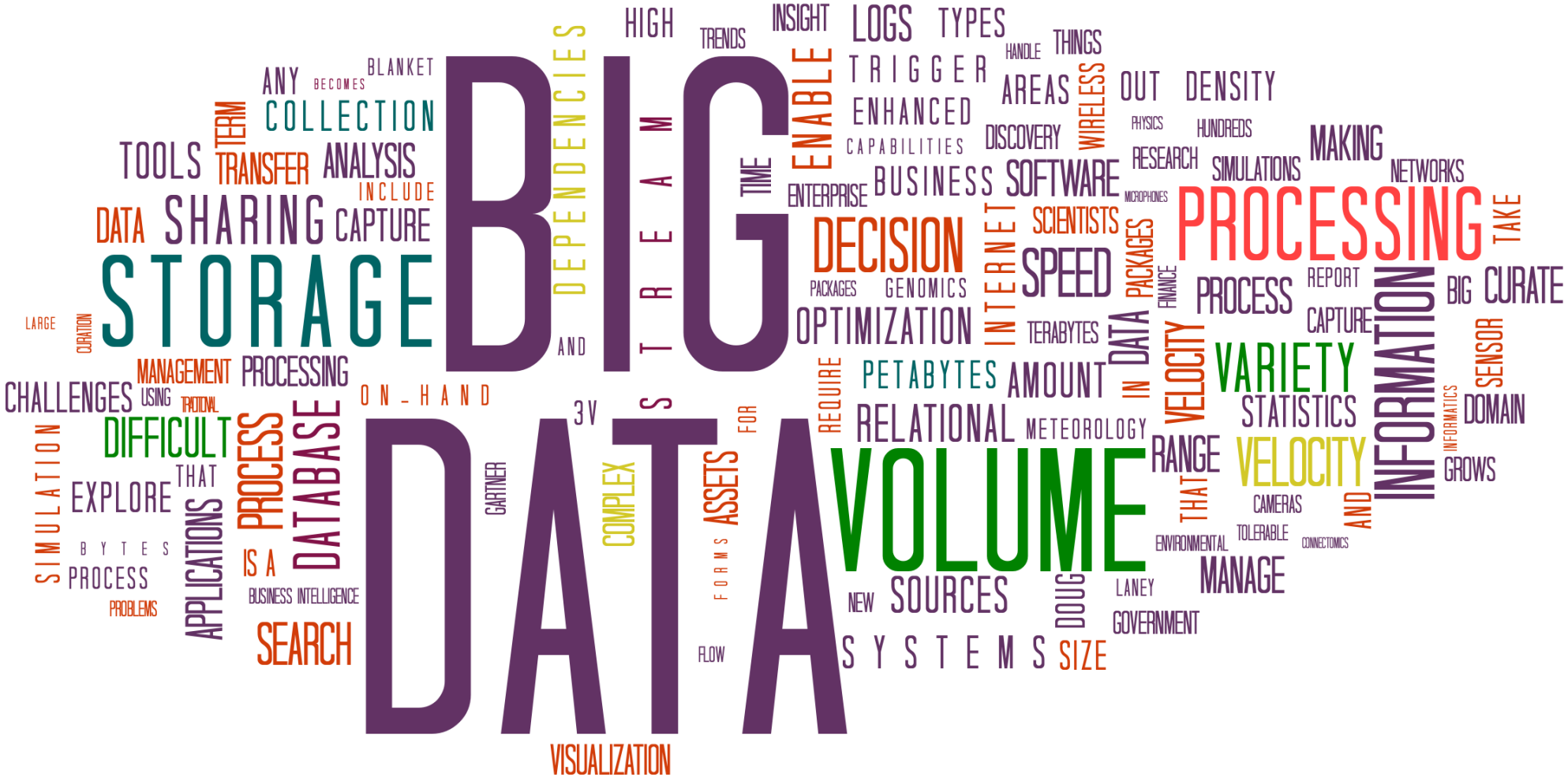
Mahmoud Parsian
Ph.D. in Computer Science

What's Big Data

Big Data is a blanket term for the non-traditional strategies and technologies needed to

- Gather
- Organize
- Process, Analyze, and
- Gather insights from LARGE Data Sets:
  - Data Sets of billions of elements
  - Peta & Tera bytes of data

What's Big Data: Many Related Technologies



What's Big Data – working with Big Data

While the problem of working with data that exceeds the computing power or storage of a single computer is not new, the pervasiveness, scale, and value of this type of computing has greatly expanded in recent years.

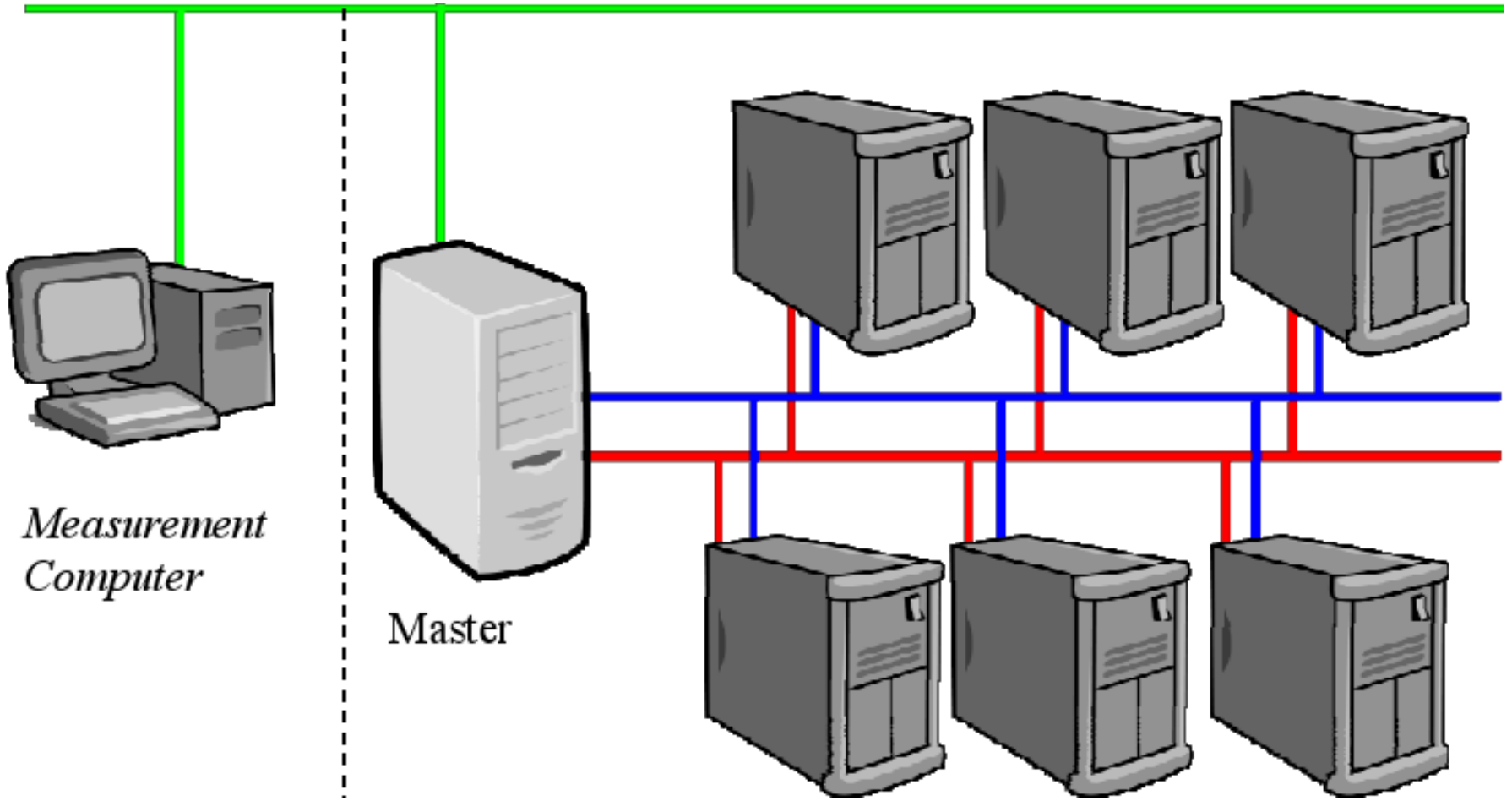
Big Data → Cluster Computing

Terabytes → Scale

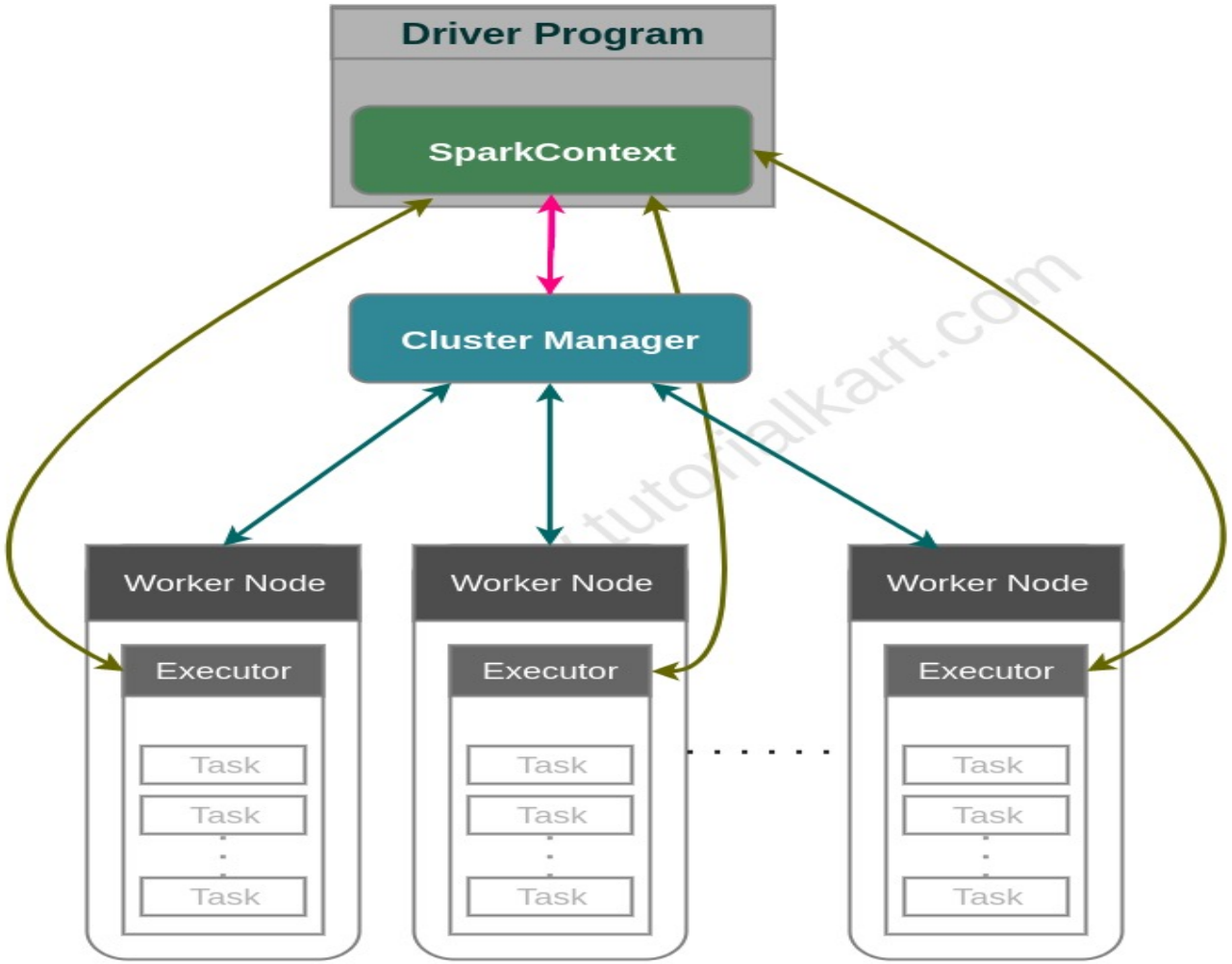
Petabytes → Use 100's or 1000's of computer servers

Replication → Handle Fault Tolerant

What's Big Data → Cluster Computing



Spark Cluster Manager



What's Big Data?

- Question: What is the definition of "big data"
- Answer: It is is difficult to nail down because projects, vendors, practitioners, and business professionals use it quite differently.
- There is no single answer!
- There are many answers!

What's Big Data?

- Generally speaking, **big data** is:
  - Large datasets (can not fit in a single server)
  - Cluster computing (network of 10's, 100's, 1000's of connected computers)
  - Distributed File System (handle Petabytes and Terabytes of data)
- The category of computing strategies and technologies that are used to handle large datasets
  - Cluster Computing
  - Apache Hadoop
  - Apache Spark

The category of computing strategies?

- MapReduce is a model/paradigm
  - Partition data into smaller chunks, and
  - Parallelize transformations on chunked data
- Apache Hadoop
- Apache Spark
- Apache Tez
- Amazon Athena
- Google BiqQuery
- Snowflake

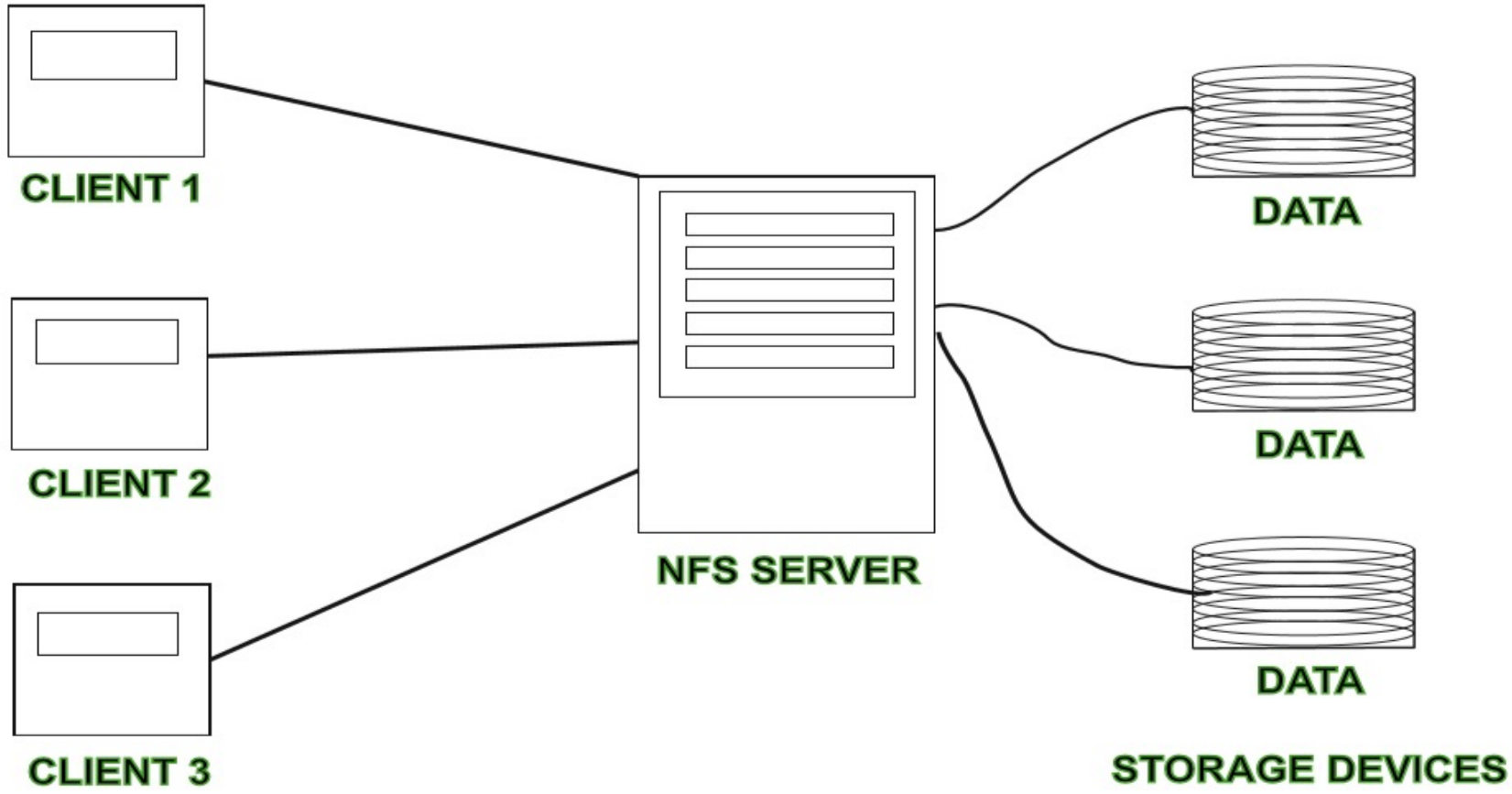
What's Big Data? Large datasets

- Large Dataset means a dataset too large to reasonably process or store with traditional tooling or on a single computer.
  - Indexing billions of documents for search engine
  - Finding cancer patterns in 1 million DNA Samples
  - Twitter data
  - Facebook messages
- This means that the common scale of big datasets is constantly shifting and may vary significantly from organization to organization.

Large datasets → Distributed File System

- NFS = Network File Server
- Consider NFS with 100 file servers
- Each file server can store: 200 TB
- Therefore: total space = 100 x 200 TB = 20,000 TB
  = 20,000 TB with replication of 1
  = 10,000 TB with replication of 2
  = 5,000 TB with replication of 4
  → Replication of data costs $money

Large datasets → Distributed File System



Where do we store Large datasets?

- Amazon S3
- Distributed File Systems:
  - Hadoop Distributed File Systems (HDFS)
  - Google File System
  - NetWare
  - NFS (Network File System)

Why Are Big Data Systems Different?

- 4 V's of Big Data:
  - Volume of Big Data (size of data)
  - Velocity of Big Data (speed of data creation)
  - Variety of Big Data (structured, unstructured)
  - Veracity of Big Data (validity of data)

Volume of Big Data

- The volume of data refers to the size of the data sets that need to be analyzed and processed, which are now frequently larger than
  - Terabytes (1 TB = 1,024 GB = 1,048,576 MB)
  - Petabytes (1 PB = 1,024 TB = 1,048,576 GB)
- The sheer volume of the data requires distinct and different processing technologies than traditional storage and processing capabilities. In other words, this means that the data sets in Big Data are too large to process with a regular laptop or desktop processor.
  - Cluster Computing
  - Distributed File System

Volume of Big Data: Example, Credit Card

Credit Card Transactions:

1. An example of a high-volume data set would be all credit card transactions on a day within Europe.

2. There were **369 billion** purchase transactions for goods and services worldwide in 2018

Volume of Big Data: Example, DNA Samples

Processing 1000,000 DNA samples,

- One sample has 5000,000 records
- All samples : 1000,000 x 5000,000

  = 5,000,000,000,000

  = 5 Trillion data points

Velocity of Big Data

- Velocity refers to the **speed with which data is generated**.
- High velocity data is generated with such a pace that it requires distinct (distributed) processing techniques.
- An example of a data that is generated with high velocity
  - Text messages
  - Twitter messages
  - Facebook posts

Variety of Big Data

- Variety makes Big Data really big.
- Big Data comes from a great variety of sources and generally is one out of three types of data:
  - Structured (table of rows and named columns)
  - Semi structured (JSON, XML)
  - Unstructured (Text files, log files)
- The variety in data types frequently requires distinct processing capabilities and specialist algorithms. An example of high variety data sets would be the audio and video files that are generated at various locations in a city.

Veracity of Big Data

- Quality of Data is Very Important
- Veracity refers to the **quality of the data** that is being analyzed.
- If data quality is not good, then ERROR in decision making
- **High veracity data** has many records that are valuable to analyze and that contribute in a meaningful way to the overall results.
- Low veracity data, on the other hand, contains a high percentage of meaningless data. The non-valuable in these data sets is referred to as noise.
- An example of a high veracity data set would be data from a medical experiment or trial.

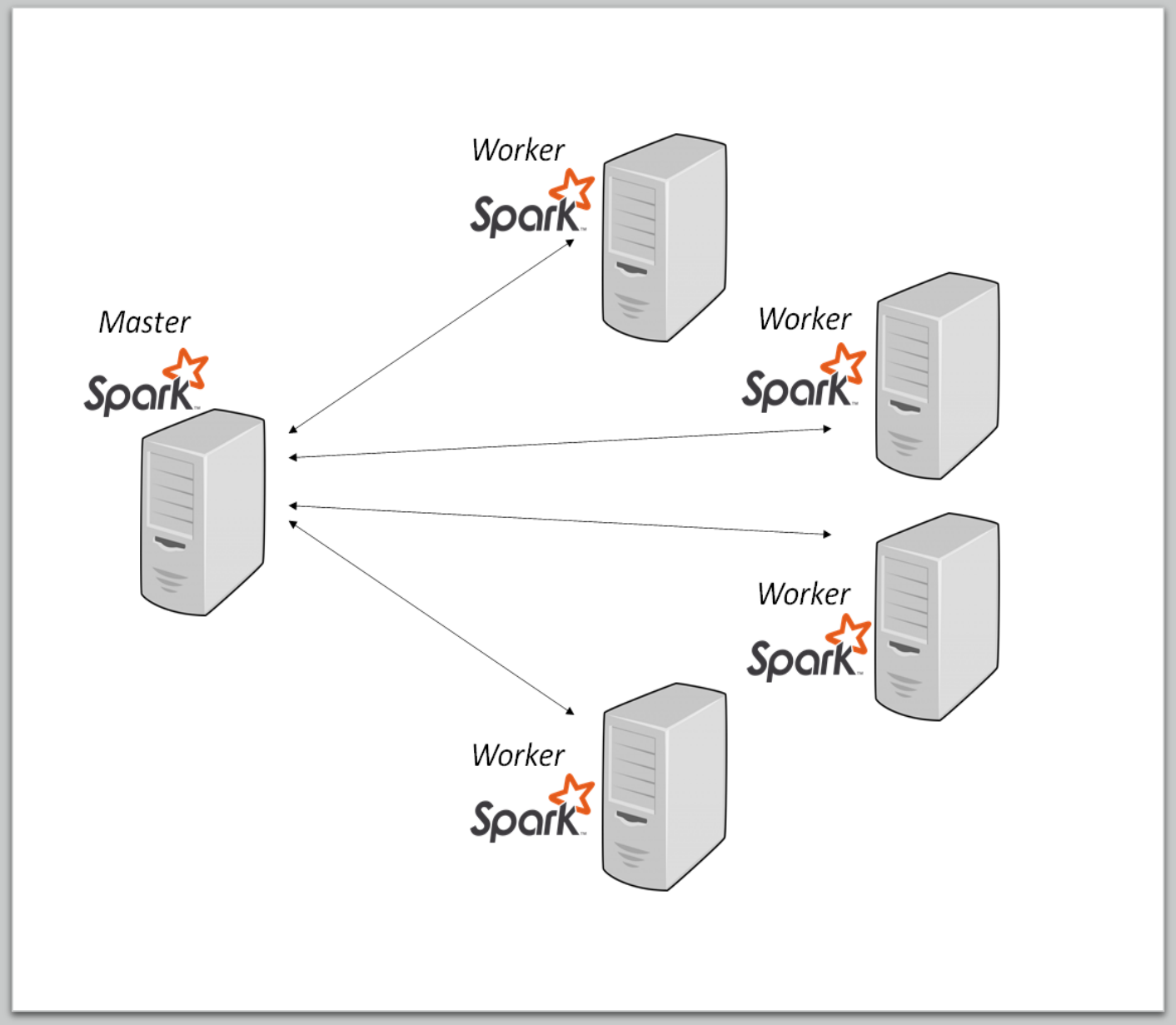
What Does a Big Data Life Cycle Look Like?

The general categories of activities involved with big data processing are:

1. Creating data from many data sources
2. Ingesting data into the system
3. Persisting the data in storage
4. Computing and Analyzing data
5. Visualizing & Presenting the results

Clustered Computing

Because of the volume of big data, individual computers are often inadequate for handling the data at most stages. To better address the high storage and computational needs of big data, computer clusters are a better fit.



Clustered Computing Benefits

- Resource Pooling: Combining the available storage space to hold data is a clear benefit, but CPU and memory pooling is also extremely important. Processing large datasets requires large amounts of all three of these resources.
- High Availability: Clusters can provide varying levels of fault tolerance and availability guarantees to prevent hardware or software failures from affecting access to data and processing. This becomes increasingly important as we continue to emphasize the importance of real-time analytics.
- Scalability: Clusters make it easy to scale horizontally by adding additional machines to the group. This means the system can react to changes in resource requirements without expanding the physical resources on a machine.

Ingesting Data into the System

- Data ingestion is the process of taking raw data and adding it to the system.
- The complexity of this operation depends heavily on the format and quality of the data sources and how far the data is from the desired state prior to processing.
- One way that data can be added to a big data system are dedicated ingestion tools. Technologies like **Apache Sqoop** can take existing data from relational databases and add it to a big data system.
- **Examples:**
  - Apache Flume
  - Apache Chukwa
  - Apache Kafka
  - Gobblin

Persisting the Data in Storage

- The ingestion processes typically hand the data off to the components that manage storage, so that it can be reliably persisted to disk. While this seems like it would be a simple operation, the volume of incoming data, the requirements for availability, and the distributed computing layer make more complex storage systems necessary.

- Persisting Solutions:
  - Amazon S3
  - Hadoop's HDFS
  - Google File System
  - NFS

Computing and Analyzing Data

- Once the data is available, the system can begin processing the data to surface actual information. The computation layer is perhaps the most diverse part of the system as the requirements and best approach can vary significantly depending on what type of insights desired. Data is often processed repeatedly, either iteratively by a single tool or by using a number of tools to surface different types of insights.
- Batch processing is one method of computing over a large dataset. The process involves breaking work up into smaller pieces, scheduling each piece on an individual machine, reshuffling the data based on the intermediate results, and then calculating and assembling the final result. These steps are often referred to individually as splitting, mapping, shuffling, reducing, and assembling, or collectively as a distributed map reduce algorithm. This is the strategy used by **Apache Hadoop's MapReduce**. Batch processing is most useful when dealing with very large datasets that require quite a bit of computation.
- Real-time processing: While batch processing is a good fit for certain types of data and computation, other workloads require more **real-time processing**. Real-time processing demands that information be processed and made ready immediately and requires the system to react as new information becomes available. One way of achieving this is **stream processing**, which operates on a continuous stream of data composed of individual items. Another common characteristic of real-time processors is in-memory computing, which works with representations of the data in the cluster's memory to avoid having to write back to disk.
- Apache Storm, Apache Flink, and Apache Spark provide different ways of achieving real-time or near real-time processing. There are trade-offs with each of these technologies, which can affect which approach is best for any individual problem. In general, real-time processing is best suited for analyzing smaller chunks of data that are changing or being added to the system rapidly.

Visualizing the Results

Visualizing data is one of the most useful ways to spot trends and make sense of a large number of data points.

- Final Results may be saved in:
  - ElasticSearch
  - Relational Databases
  - Snowflake

- Visualization tools:
  - Jupyter Notebook
  - Apache Zeppelin

Big Data Glossary

- **Big data**: Big data is an umbrella term for datasets that cannot reasonably be handled by traditional computers or tools due to their volume, velocity, and variety. This term is also typically applied to technologies and strategies to work with this type of data.
- **Batch processing**: Batch processing is a computing strategy that involves processing data in large sets. This is typically ideal for non-time sensitive work that operates on very large sets of data. The process is started and at a later time, the results are returned by the system.
- **Cluster computing**: Clustered computing is the practice of pooling the resources of multiple machines and managing their collective capabilities to complete tasks. Computer clusters require a cluster management layer which handles communication between the individual nodes and coordinates work assignment.

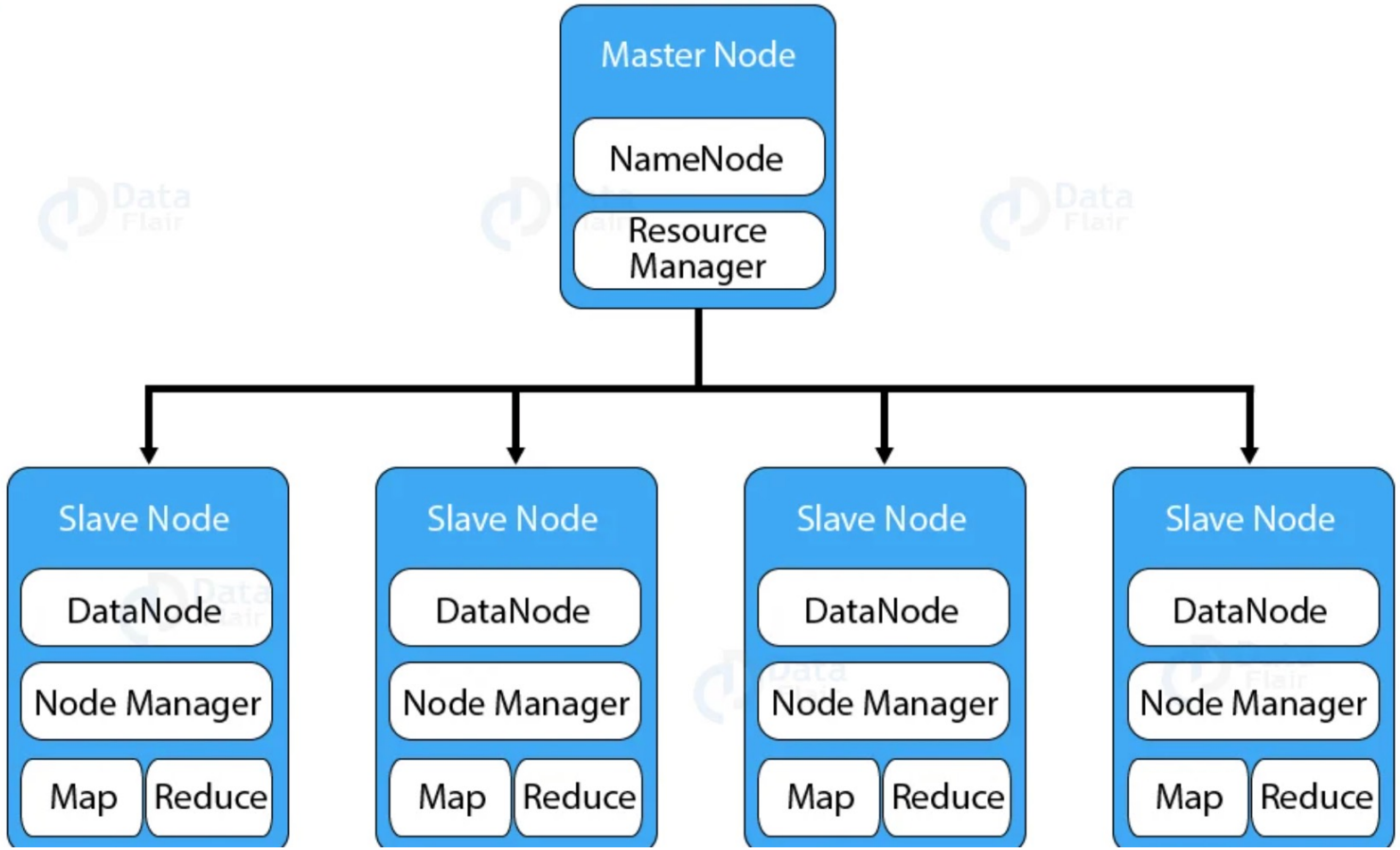
Big Data Glossary

- **Data lake**: Data lake is a term for a large repository of collected data in a relatively raw state. This is frequently used to refer to the data collected in a big data system which might be unstructured and frequently changing. This differs in spirit to data warehouses (defined below).
- **Data mining**: Data mining is a broad term for the practice of trying to find patterns in large sets of data. It is the process of trying to refine a mass of data into a more understandable and cohesive set of information.
- **Data warehouse**: Data warehouses are large, ordered repositories of data that can be used for analysis and reporting. In contrast to a *data lake*, a data warehouse is composed of data that has been cleaned, integrated with other sources, and is generally well-ordered. Data warehouses are often spoken about in relation to big data, but typically are components of more conventional systems.
- **ETL**: ETL stands for extract, transform, and load. It refers to the process of taking raw data and preparing it for the system's use. This is traditionally a process associated with data warehouses, but characteristics of this process are also found in the ingestion pipelines of big data systems.

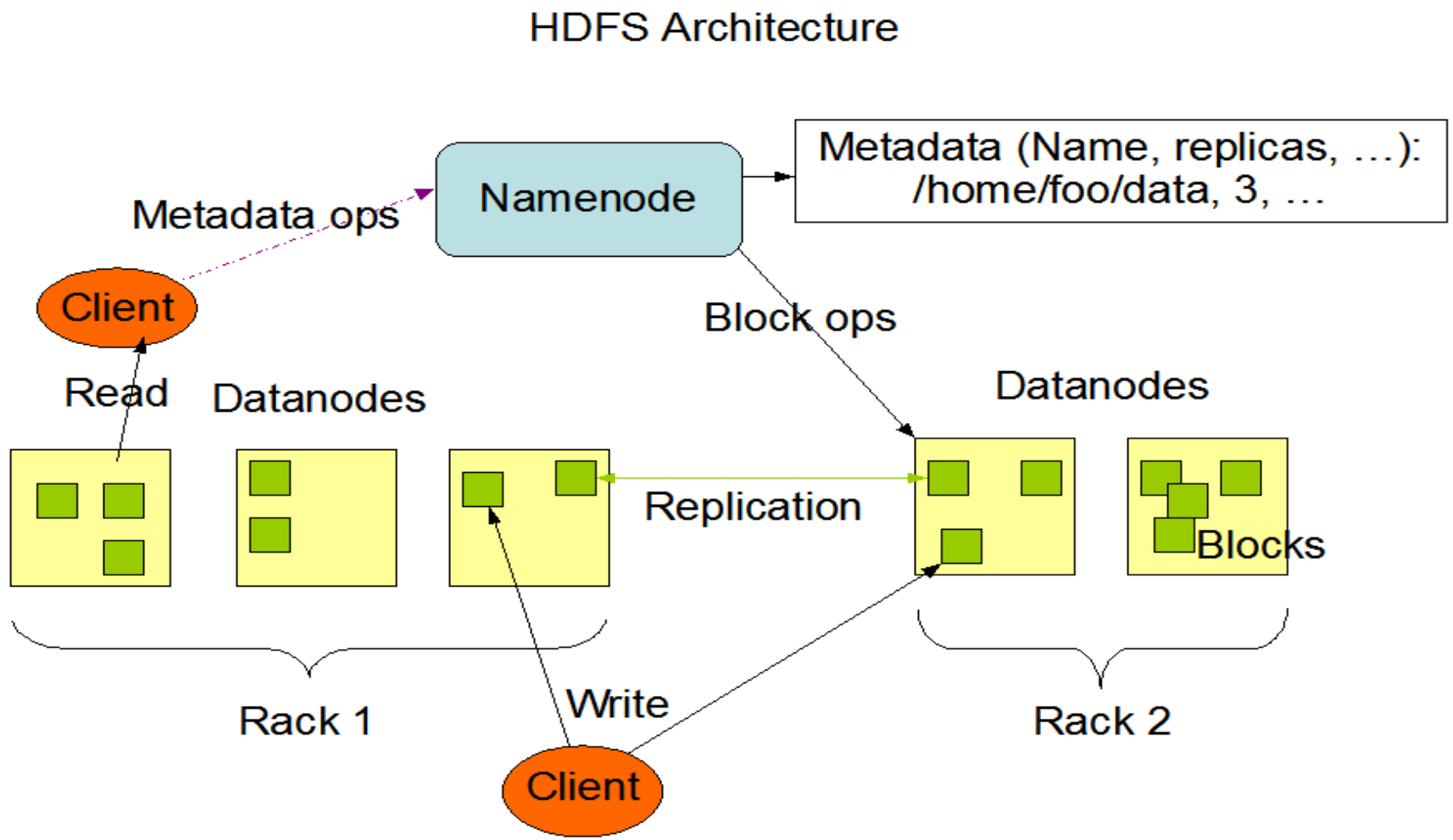
Big Data Glossary: Apache Hadoop

- Hadoop can process big data using MapReduce paradigm
- **Hadoop** is a framework that is used to efficiently store and process large datasets ranging in size from gigabytes to petabytes of data.
- Hadoop consists of a distributed file system called HDFS, with a cluster management and resource scheduler on top called YARN (Yet Another Resource Negotiator).
- Batch processing capabilities are provided by the MapReduce computation engine.

Big Data Glossary: Hadoop Architecture

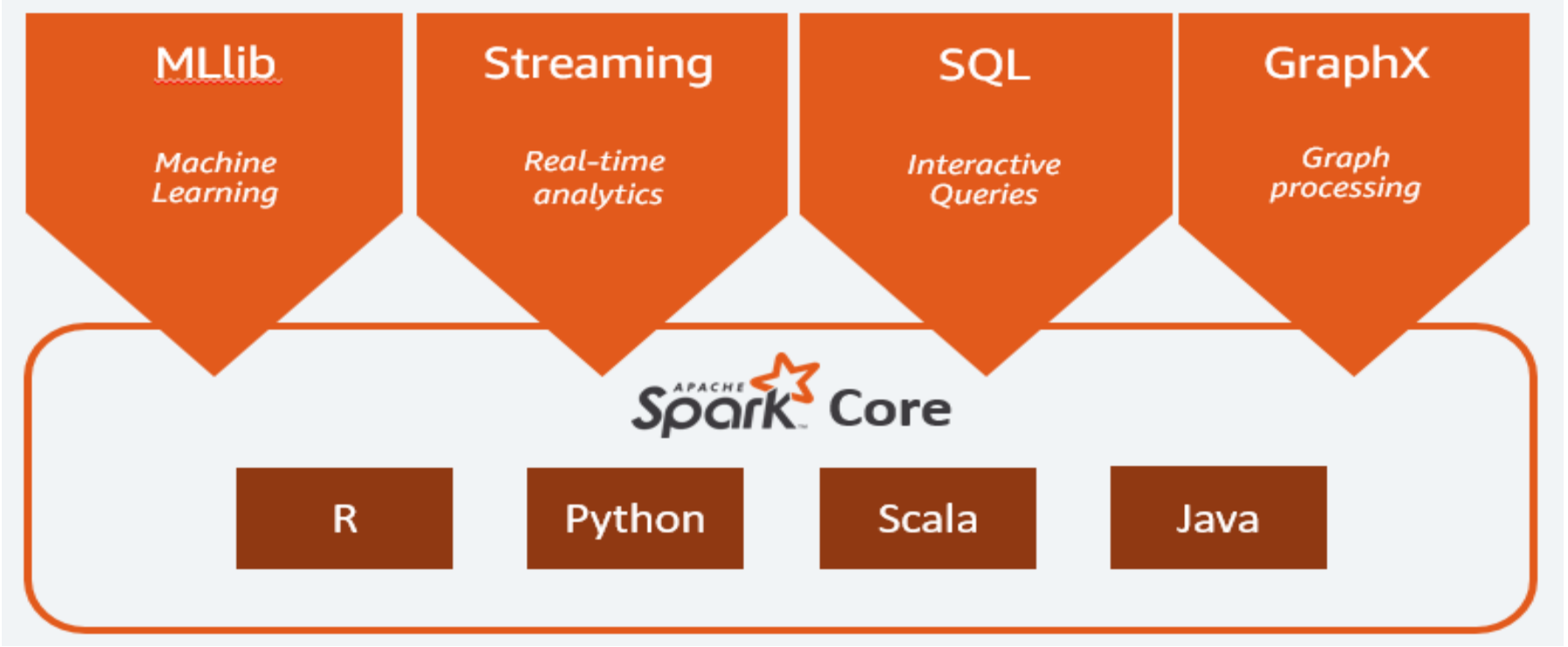


Big Data Glossary: Hadoop HDFS Architecture



Big Data Glossary

- **Spark**: Unified engine for large-scale data analytics
- **Spark** is a multi-language engine for executing data engineering, data science, and machine learning on single-node machines or clusters.



Big Data Glossary: In-memory computing

- In-memory computing is a strategy that involves moving the working datasets entirely within a cluster's collective memory.
- Intermediate calculations are not written to disk and are instead held in memory.
- This gives in-memory computing systems like Apache Spark a huge advantage in speed over I/O bound systems like Hadoop's MapReduce.
- Memory access is 100+ times faster than Disk access

Big Data Glossary: Machine Learning

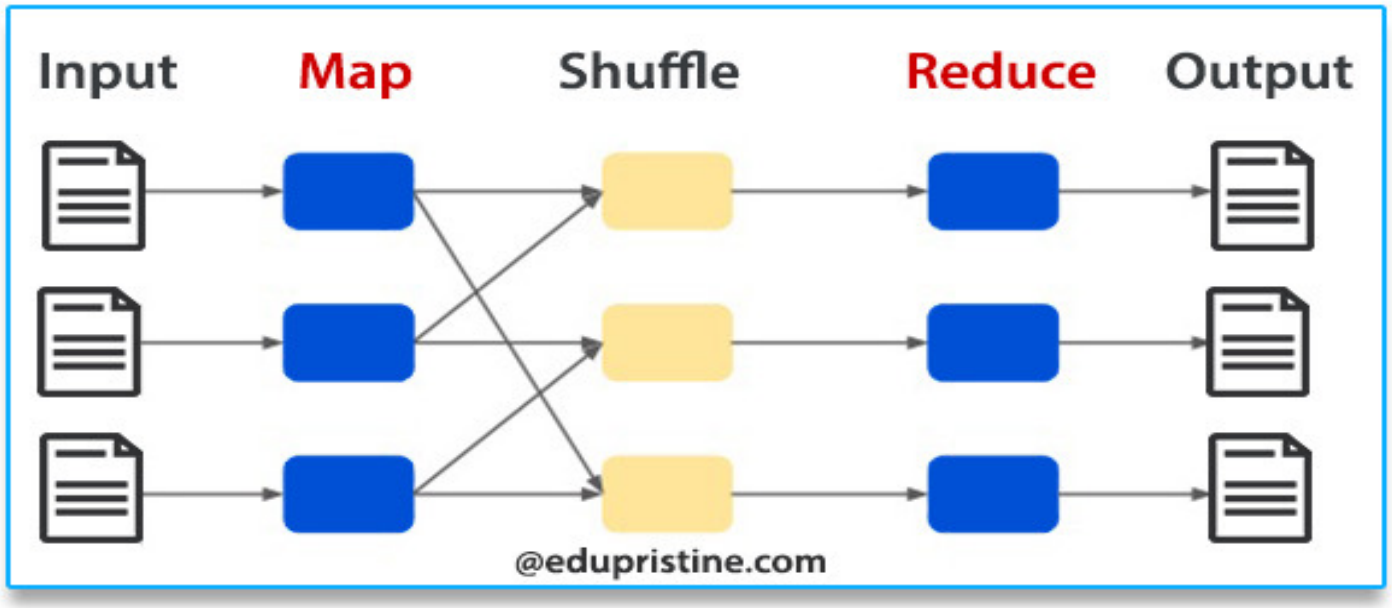
- Machine learning is the study and practice of designing systems that can learn, adjust, and improve based on the data fed to them.
- This typically involves implementation of predictive and statistical algorithms that can continually zero in on “correct” behavior and insights as more data flows through the system.

Big Data Glossary

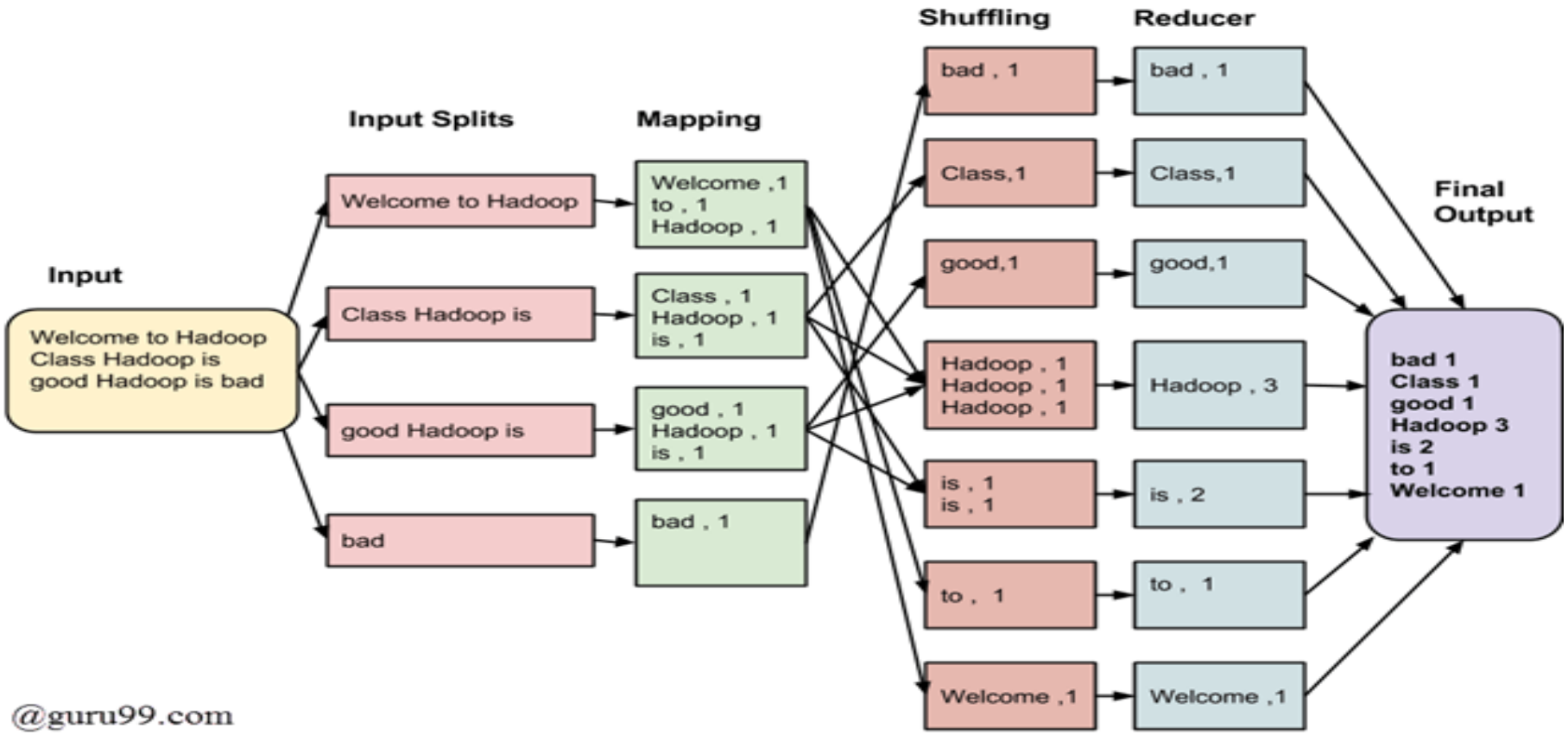
- **MapReduce (big data algorithm)**:
- MapReduce is an algorithm for scheduling work on a computing cluster.
- The process involves splitting the problem set up (mapping it to different nodes) and computing over them to produce intermediate results, shuffling the results to align like sets, and then reducing the results by outputting a single value for each set.
- `map(input) → {(key1, value1), (key2, value2), …}`
- `reduce(key, [V1, V2, …]) →`

`{(K1, final_value), (K2, final_value2), …}`

Big Data Glossary: MapReduce Model



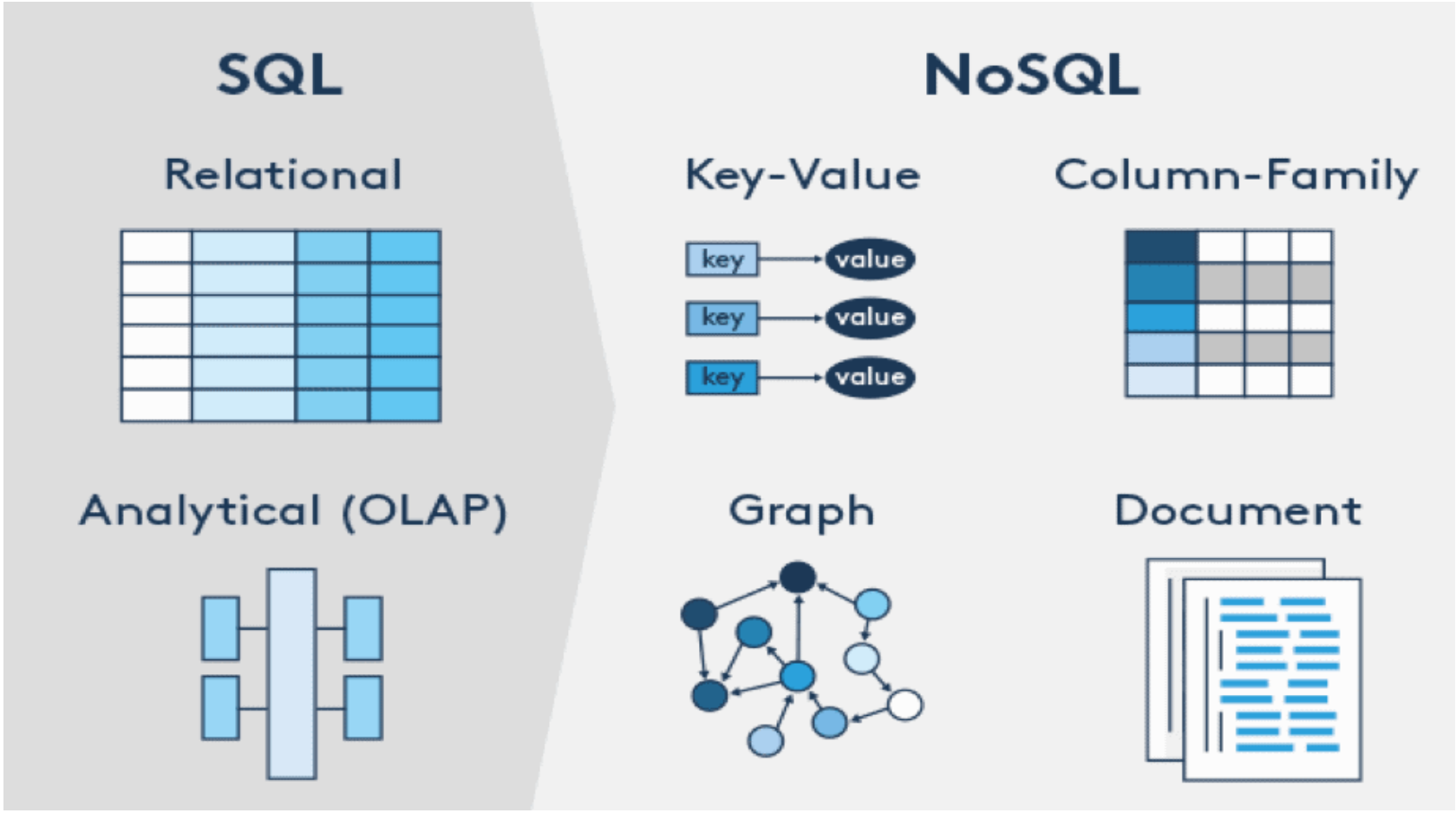
Big Data Glossary: MapReduce Example



Big Data Glossary: NoSQL

- NoSQL is a broad term referring to databases designed outside of the traditional relational model.
- NoSQL databases have different trade-offs compared to relational databases, but are often well-suited for big data systems due to their flexibility and frequent distributed-first architecture.
- NoSQL Examples:
  - Spark SQL
  - Amazon Athena
  - Google BigQuery
  - Snowflake

Big Data Glossary: NoSQL



Big Data Glossary: Stream processing

- Stream processing is the practice of computing over individual data items as they move through a system.
- This allows for real-time analysis of the data being fed to the system and is useful for time-sensitive operations using high velocity metrics.
- Stream processing examples:
  - Text Messaging
  - Engine data
  - Twitter data
  - Facebook Data
  - Credit Card Transactions data
  - DNA Sequencing Machines

Big Data Glossary: Stream processing