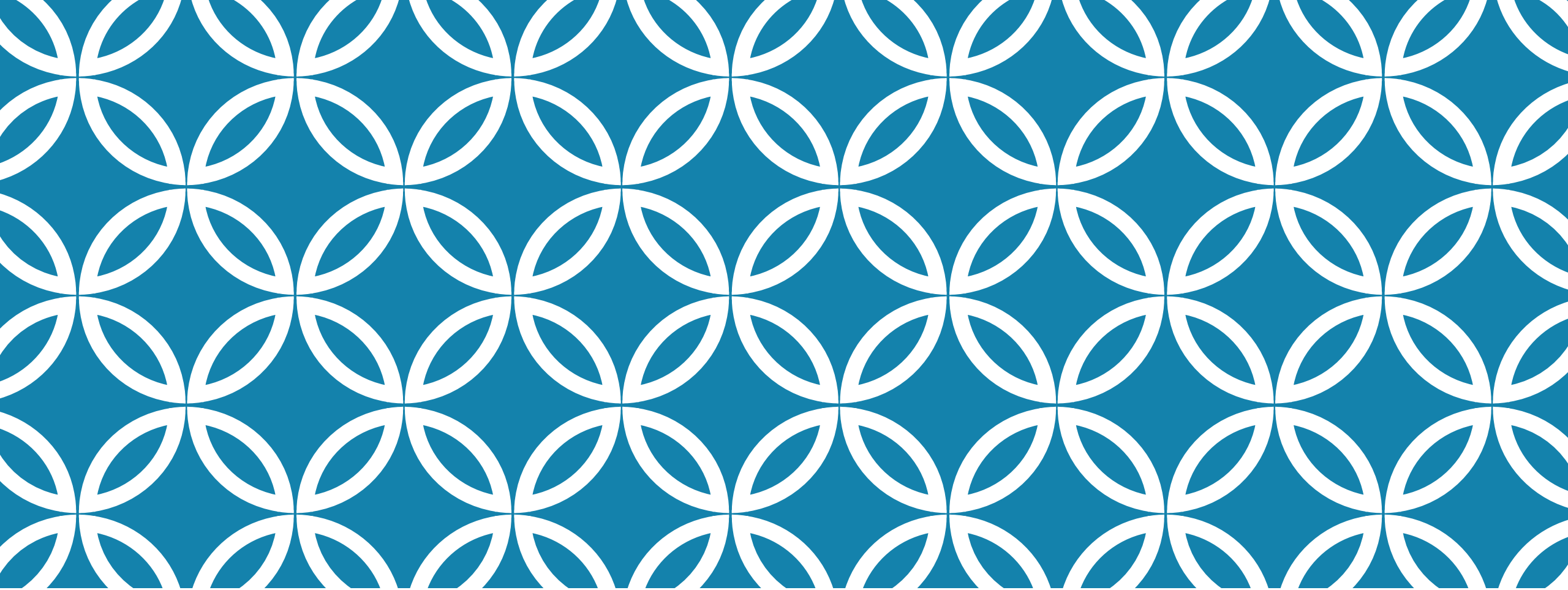


HADOOP MAP-REDUCE

Dr. Salahuddin

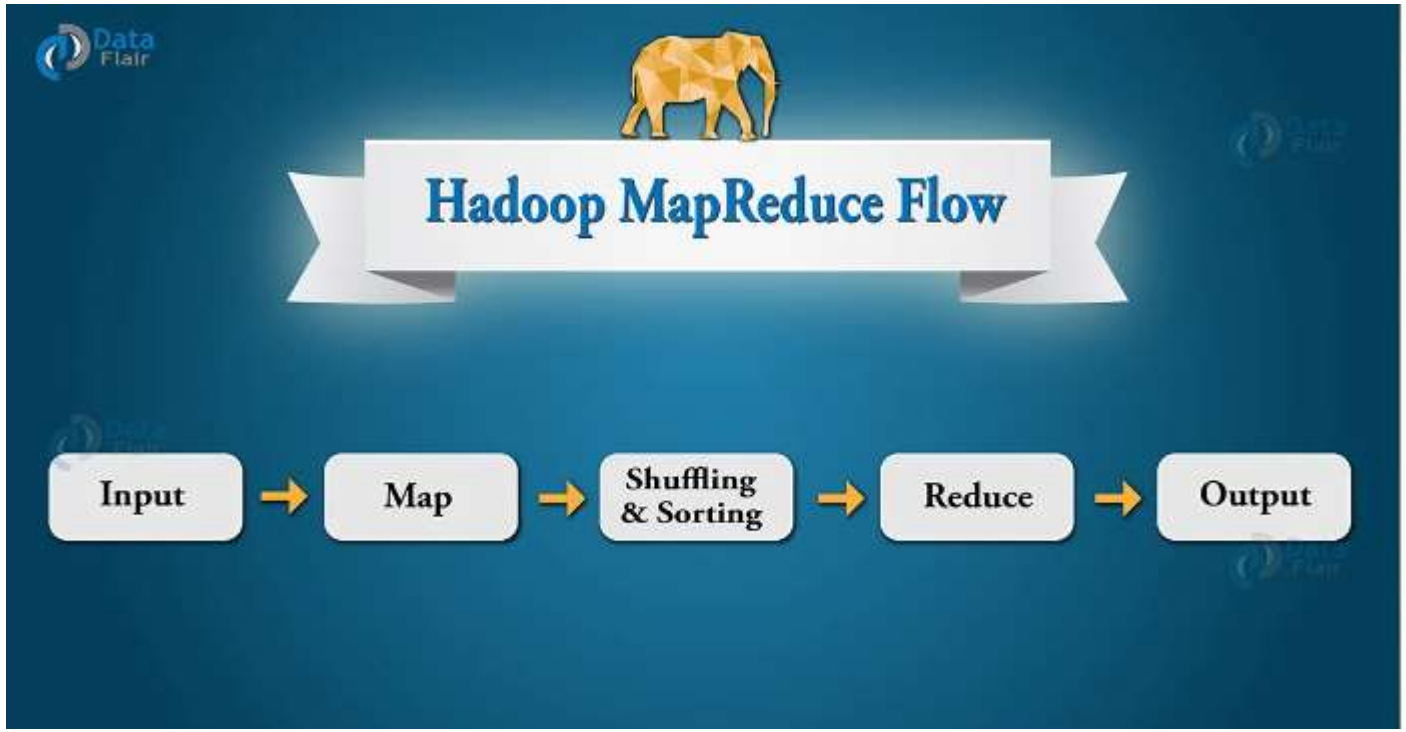
MAP-REDUCE

- **Hadoop** MapReduce processes a huge amount of data in parallel by dividing the job into a set of independent tasks (sub-job). In Hadoop, MapReduce works by breaking the processing into phases: Map and Reduce.

- Map-Reduce is not similar to the other regular processing framework like Hibernate, JDK, .NET, etc. All these previous frameworks are designed to use with a traditional system where the data is stored at a single location like Network File System, Oracle database, etc.

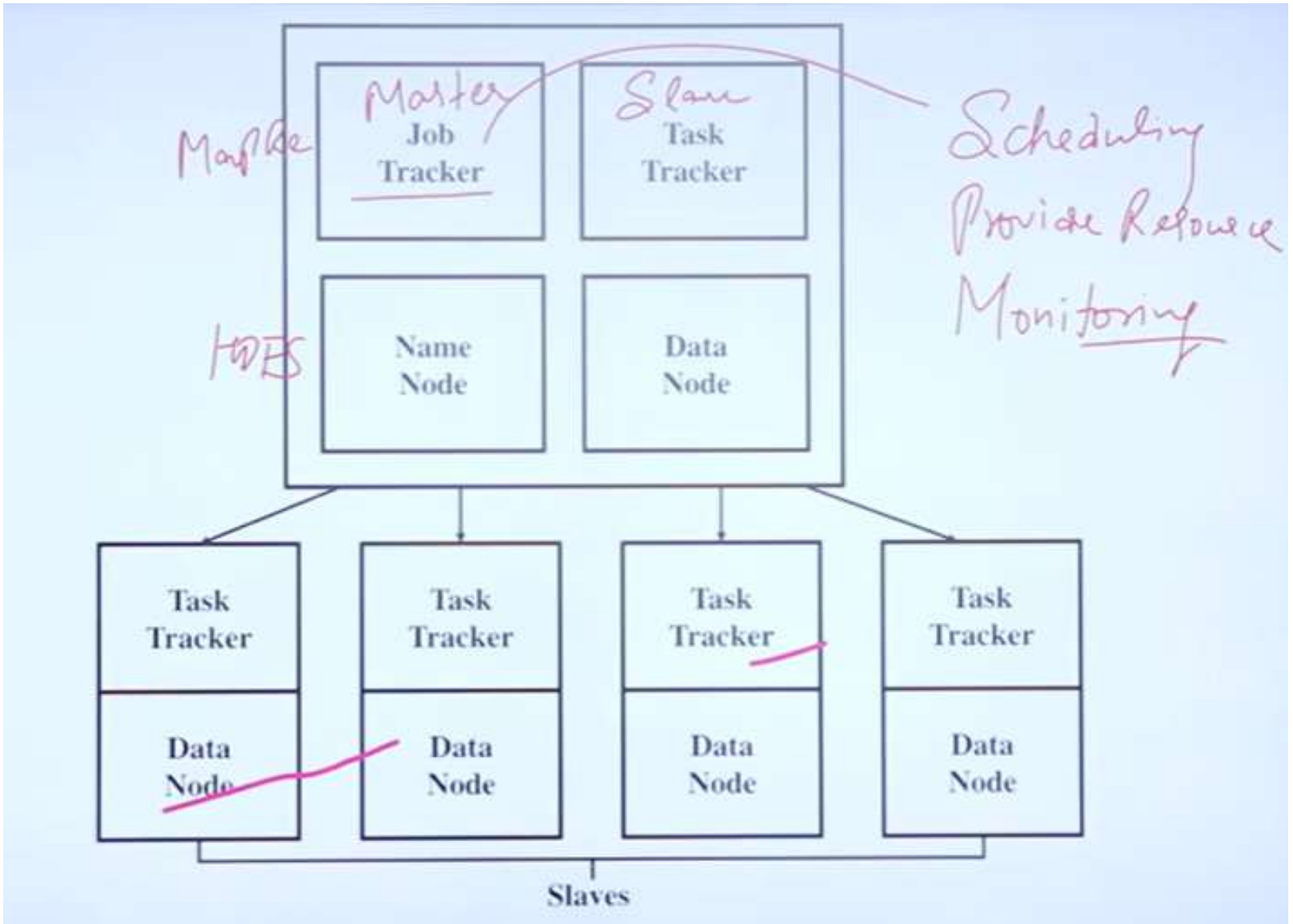
- But when we are processing big data the data is located on multiple commodity machines with the help of HDFS.
- So when the data is stored on multiple nodes we need a processing framework where it can copy the program to the location where the data is present, Means it copies the program to all the machines where the data is present.
- Here the Map-Reduce came into the picture for processing the data on Hadoop over a distributed system. Hadoop has a major drawback of cross-switch network traffic which is due to the massive volume of data. Map-Reduce comes with a feature called **Data-Locality**. Data Locality is the potential to move the computations closer to the actual data location on the machines.

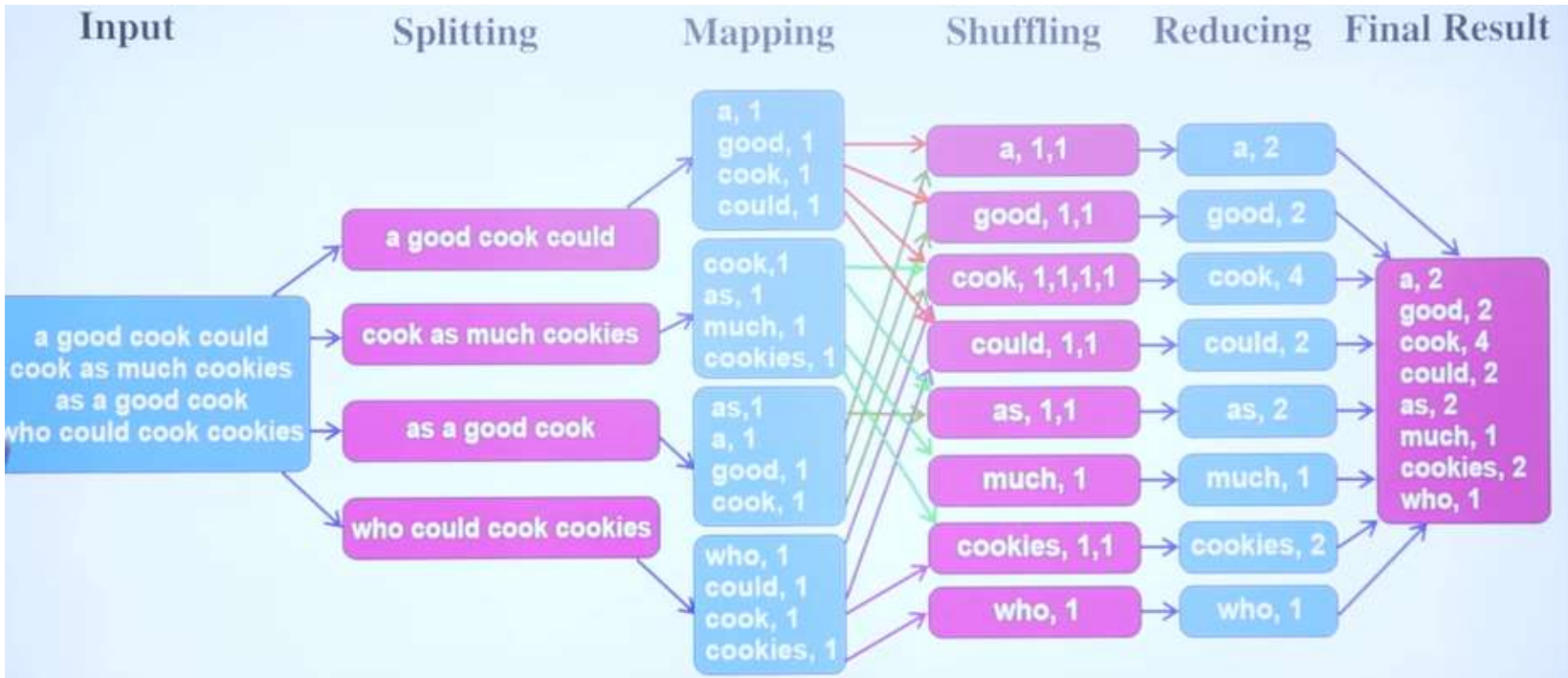
- Since Hadoop is designed to work on commodity hardware it uses Map-Reduce as it is widely acceptable which provides an easy way to process data over multiple nodes.
- Map-Reduce is not the only framework for parallel processing. Nowadays Spark is also a popular framework used for distributed computing like Map-Reduce.



LET'S UNDERSTAND DATA-FLOW IN MAP-REDUCE

Map Reduce is a terminology that comes with **Map Phase** and **Reducer Phase**. The map is used for Transformation while the Reducer is used for aggregation kind of operation. The terminology for Map and Reduce is derived from some functional programming languages like Lisp, Scala, etc. The Map-Reduce processing framework program comes with 3 main components i.e. our **Driver code**, **Mapper**(For Transformation), and **Reducer**(For Aggregation).


Job Tracker
Task Tracker
Name Node
Data Node
Task Tracker
Data Node
Task Tracker
Data Node
Task Tracker
Data Node
Task Tracker
Data Node
Slaves


Input
Splitting
Mapping
Shuffling
Reducing
Final Result
a good cook could
cook as much cookies
as a good cook
who could cook cookies
a good cook could
cook as much cookies
as a good cook
who could cook cookies
a, 1
good, 1
cook, 1
could, 1
cook,1
as, 1
much, 1
cookies, 1
as,1
a, 1
good, 1
cook, 1
who, 1
could, 1
cook, 1
cookies, 1
a, 1,1
good, 1,1
cook, 1,1,1,1
could, 1,1
as, 1,1
much, 1
cookies, 1,1
who, 1
a, 2
good, 2
cook, 4
could, 2
as, 2
much, 1
cookies, 2
who, 1
a, 2
good, 2
cook, 4
could, 2
as, 2
much, 1
cookies, 2
who, 1

LET'S UNDERSTAND DATA-FLOW IN MAP-REDUCE

- Let's take an **example** where you have a file of 10TB in size to process on Hadoop. The 10TB of data is first distributed across multiple nodes on Hadoop with HDFS. Now we have to process it for that we have a Map-Reduce framework. So to process this data with Map-Reduce we have a Driver code which is called **Job**.
- We need to initiate the Driver code to utilize the advantages of this Map-Reduce Framework.
- There are also **Mapper** and **Reducer** classes provided by this framework which are predefined and modified by the developers as per the organizations requirement.

BRIEF WORKING OF MAPPER

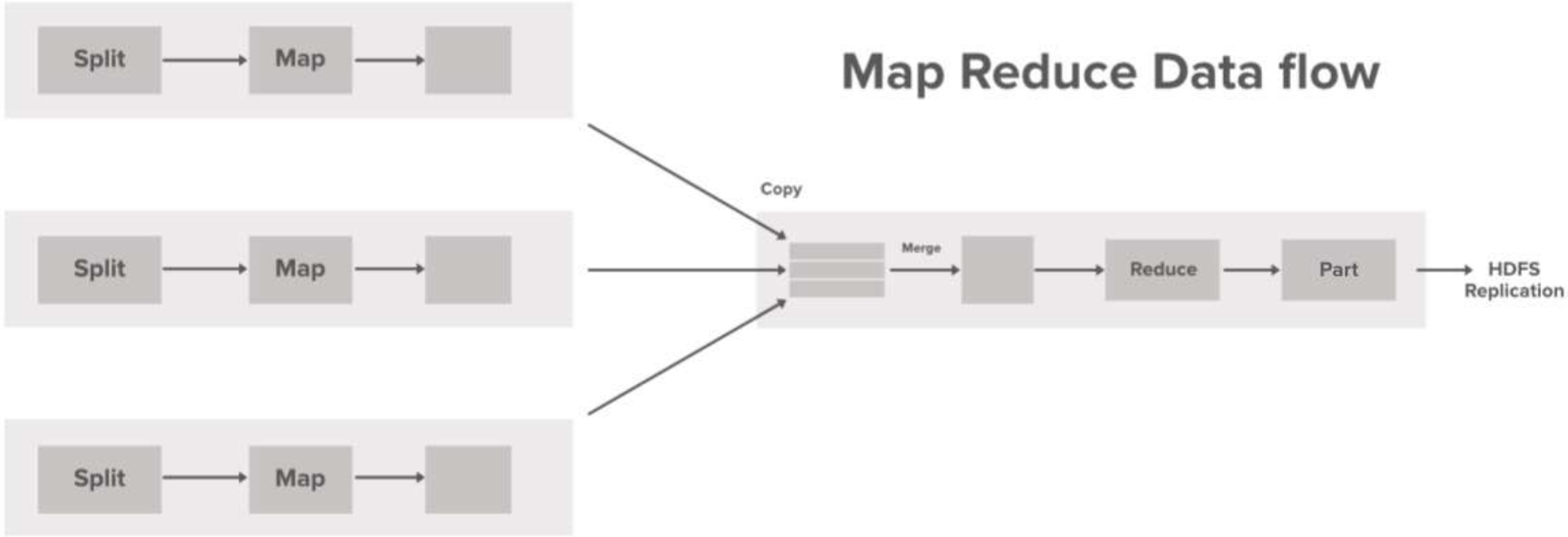
Mapper is the initial line of code that initially interacts with the input dataset. suppose, If we have 100 Data-Blocks of the dataset we are analyzing then, in that case, there will be 100 Mapper program or process that runs in parallel on machines(nodes) and produce there own output known as intermediate output which is then stored on Local Disk, not on HDFS.

The output of the mapper act as input for Reducer which performs some sorting and aggregation operation on data and produces the final output.

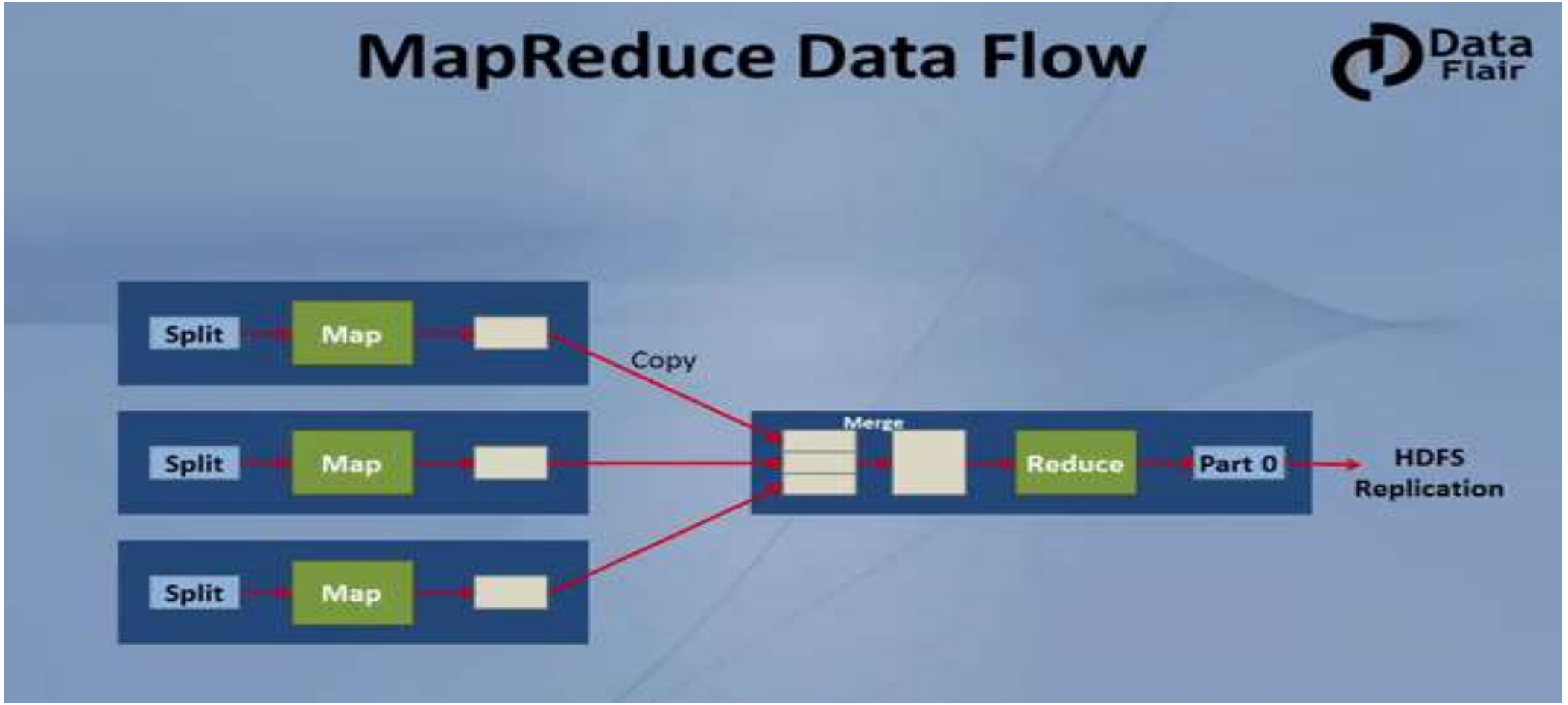
BRIEF WORKING OF REDUCER

Reducer is the second part of the Map-Reduce programming model. The Mapper produces the output in the form of key-value pairs which works as input for the Reducer. But before sending this intermediate key-value pairs directly to the Reducer some process will be done which shuffle and sort the key-value pairs according to its key values. The output generated by the Reducer will be the final output which is then stored on HDFS(Hadoop Distributed File System). Reducer mainly performs some computation operation like addition, filtration, and aggregation.

HOW HADOOP MAPREDUCE WORKS?



HOW HADOOP MAPREDUCE WORKS?



MAPREDUCE INTERNALS

MapReduce is the combination of two different processing idioms called **Map** and **Reduce**, where we can specify our custom business logic. The map is the first phase of processing, where we specify all the complex logic/business rules/costly code. On the other hand, Reduce is the second phase of processing, where we specify light-weight processing. For example, aggregation/summation

STEPS OF DATA-FLOW:

Step 1: One block is processed by one **mapper** at a time. In the mapper, a developer can specify his own business logic as per the requirements. In this manner, Map runs on all the nodes of the cluster and process the data blocks in parallel.

Step 2: Output of Mapper also known as intermediate output is written to the local disk. An output of mapper is not stored on HDFS as this is temporary data and **writing on HDFS** will create unnecessary many copies.

Step 3: Output of mapper is shuffled to **reducer** node (which is a normal slave node but reduce phase will run here hence called as reducer node). The shuffling/copying is a physical movement of data which is done over the network.

STEPS OF DATA-FLOW:

Step 4: Once all the mappers are finished and their output is shuffled on reducer nodes then this intermediate output is merged & sorted. Which is then provided as input to reduce phase.

Step 5: Reduce is the second phase of processing where the user can specify his own custom business logic as per the requirements. An input to a reducer is provided from all the mappers. An output of reducer is the final output, which is written on HDFS.