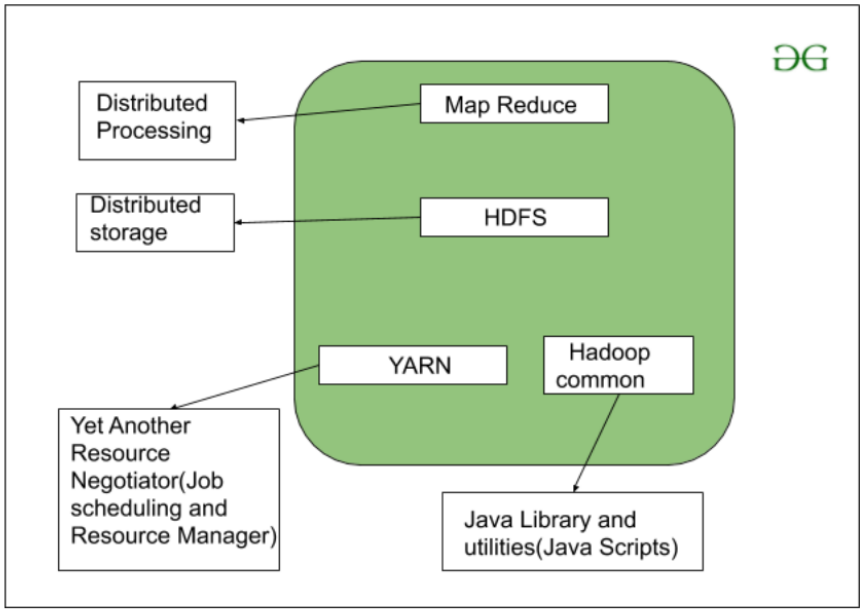
# HADOOP

Open Source framework  that allows distributed processing of large data-sets across the cluster of commodity hardware
Uses FIFO Scheduling

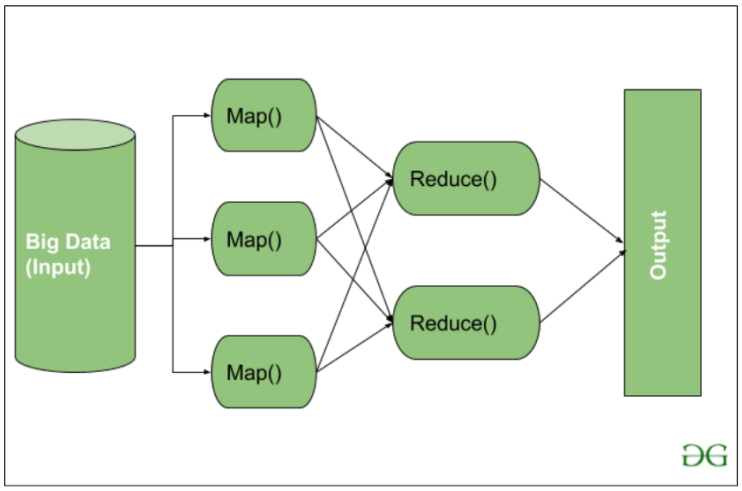



Features of Hadoop:

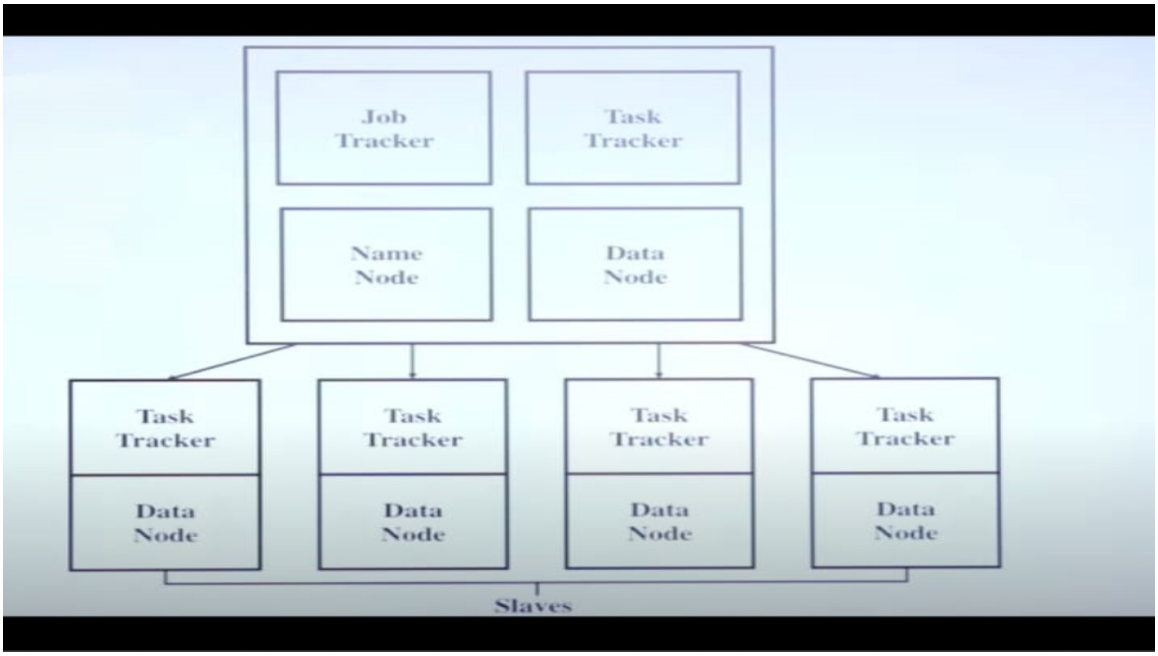
- Fault tolerance : re executing failed tasks
- Ease of use : simplifies development and management of applications
- Economic :  cost effective as it is run on commodity hardware
- Security : robust security measures
- Scalability : can handle petabytes of data

**MapReduce**

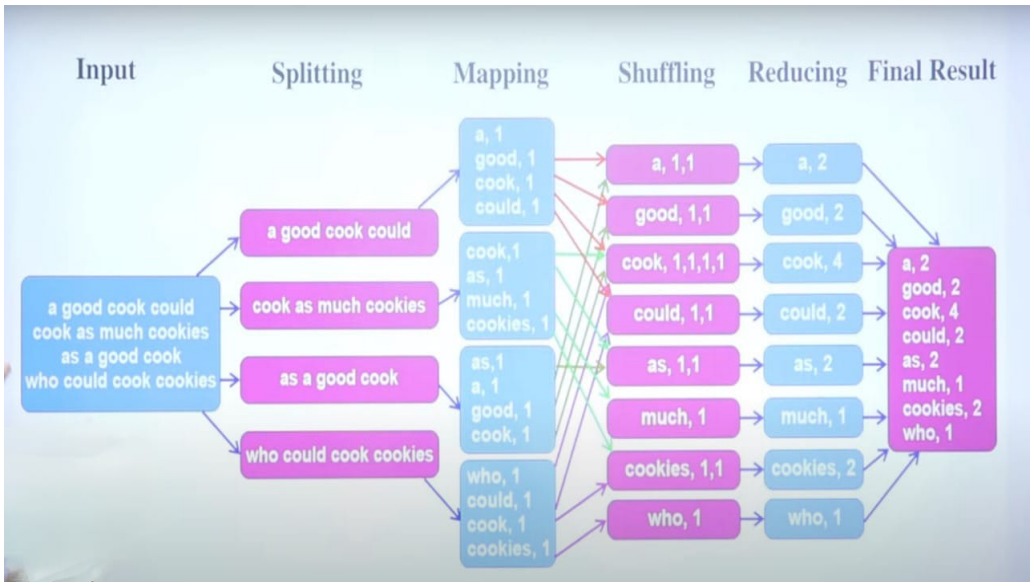
- The major feature of MapReduce is to perform the distributed processing in parallel in a Hadoop cluster which Makes Hadoop working so fast
- When you are dealing with Big Data, serial processing is no more of any use
- MapReduce has mainly 2 tasks which are divided phase-wise:
- In the first phase, Map is utilized and in next phase Reduce is utilized.
- Two essential daemons - Job tracker and task tracker




The Input is provided to the Map() function then it's output is used as an input to the Reduce function and after that, we receive our final output
The Input provided to the map is a set of Data.
The Map() function here breaks this Data into Tuples that are nothing but a key-value pair.
These key-value pairs are now sent as input to the Reduce().
The Reduce() function then combines this key-value pair based on its Key and value and perform some operation like sorting, summation type job, etc. which is then sent to the final Output Node

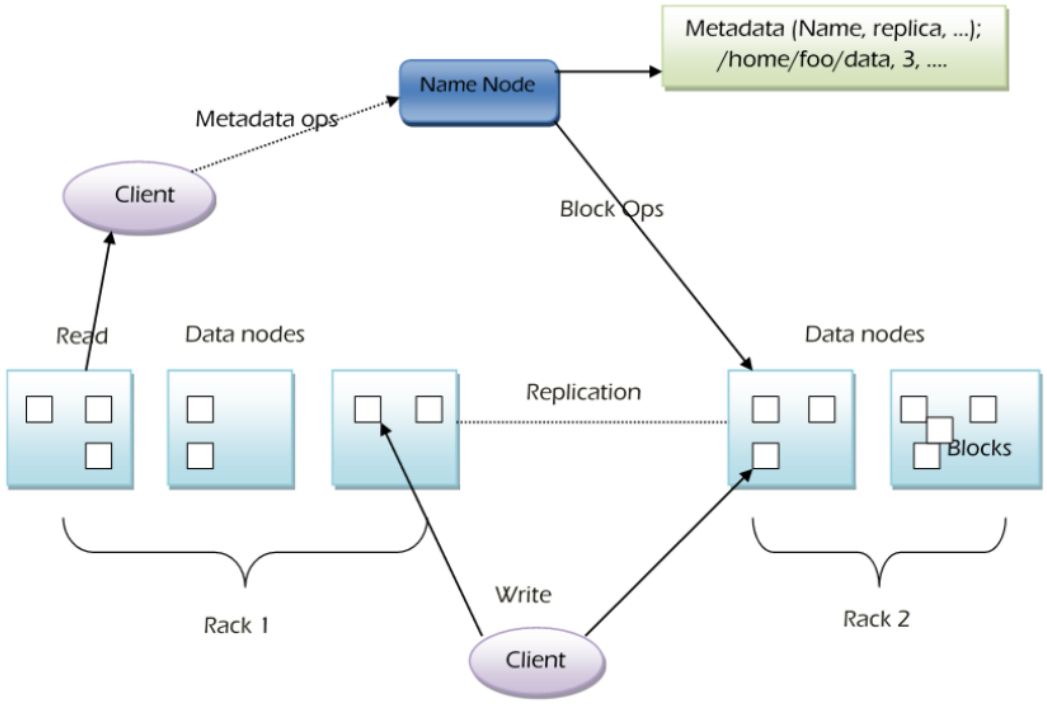



**WORD COUNT FOR MAP REDUCE**




1. Input :
    a. Input data is loaded into the system
2. Splitting:
    a. The input data is split into manageable blocks or chunks of data
    b. Hadoop each block size is 128mb
    c. Each block is individually processed by map tasks
3. Mapping:
    a. Each map task processes a block of data at a time to produce key value pairs
    b. For example in the word count application word becomes the key and its value is the number
    c. Eg: a : 1, good : 1
4. Shuffling:
    a. Once all map tasks have processed their respective blocks of data and output , the framework collects and shuffles these outputs. This phase involves redistributing the data based on keys
    b. All values associated with the same key are brought to the same reducer
    c. Eg: a : 1,1 ;good : 1,1
5. Reducing:
    a. Each reducer deals with one key and its corresponding value
    b. The reducer sums the values, effectively counting the total number of occurrences for each word
    c. Eg: a:2, good:2
6. Final result:
    a. The output from all the reducers are collected and written back to HDFS providing a consolidated output

## HDFS ARCHITECTURE




- Hadoop Distributed File System follows the master-slave architecture.
- Each cluster comprises a single master node and multiple slave nodes.
- The files get divided into one or more blocks, and each block is stored on different slave machines
- The master node(NameNode) stores and manages the file system namespace, that is information about blocks of files
- The slave nodes(DataNode) store data blocks of files.

1. NameNode:
    - The Name Node is the centerpiece of an HDFS file system
    - Name node does not store any of these files data itself.
    - The Name Node is responsible for maintaining the meta information of the Hadoop file system.
    - The Name Node is a Single Point of Failure for the HDFS Cluster
    - When the Name Node goes down, the file system goes offline
    - The Name Node will update two important permanent 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.

2. Secondary NameNode:
    - When NameNode runs out of disk space, a secondary NameNode is activated to perform a checkpoint.
    - 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 as it 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.

3. Data Node:
    - A Data Node stores data in the Hadoop File System and is the place to hold the data
    - In the beginning, Data Node connects to the Name Node and establishes the service
    - Then Data Node responds to requests from the Name Node for file system operations
    - Client applications can talk directly to a Data Node by using the location of the data provided by Name Node

4. CheckPoint Node:
    - The Checkpoint node is a node that periodically creates checkpoints of the namespace.
    - It downloads Fsimage and edits from the Active Namenode. Then it merges them locally, and at last, it uploads the new image back to the active NameNode.

5. Task Tracker:
    - A Task Tracker is a node in the cluster that accepts tasks like Map, Reduce and Shuffle operations from a Job Tracker.

- Task tracker is responsible for instantiating & monitoring individual map and reduces work
- Task Tracker is also known as s/w daemon for Hadoop architecture.
- Every Task Tracker is configured with a set of slots. That tells how many tasks it can accept
- responsible for executing the tasks assigned by the job tracker

6. Job Tracker
    - The Job Tracker is the service within Hadoop and farms out Map Reduce tasks to specific nodes in the cluster
    - Job tracker will reside on top of the Name Node
    - Job Tracker manages the map reduce tasks and distributes individual tasks to machine running the task tracker
    - Client applications submit jobs to the Job tracker
    - The Job Tracker talks to the Name Node to determine the data location
    - The Job Tracker submits the work to the chosen Task Tracker nodes.
    - When the work is completed, the Job Tracker updates its status.
    - The Job Tracker is a point of failure for the Hadoop Map Reduce service. If it goes down, all running jobs are halted