**MAP REDUCE**

* Split input files (e.g., by HDFS blocks)
* Move code to data
* Operate on key / value pairs
* Mappers filter and transform input data
* Sort & Shuffle provides order to mapper data
* Reducers aggregate the mapper output (post Sort & Shuffle)
* Stores Reducer output in HDFS

**MAP REDUCE STEPS**

* Split input data in independent chunks is already available via HDFS
* Job needs to be scheduled to carry out required process
* Schedule tasks on nodes where data is already present
* Map Phase - Transformation Phase
  + input Data | output list <key, value> pairs
* Sort & Shuffle - Group & Order Phase
  + input list of <key, value> pairs | output sorted & grouped list of <key, value> pairs
* Reduce Phase - Aggregation Phase
  + sorted & grouped list of <key, value> pairs | output aggregated <key, value>
* Manager required for scheduling jobs, collating results & updating status
* Process and storage nodes are same. i.e., MR Tasks and HDFS Data Node run on same machine

**MR-V1 PROCESS**

* A Client invokes a Map Reduce, from a Calling Node (maybe a Data Node or an Extra Node in the cluster)
* An instance of Job Tracker is created in the memory of the Calling Node
* The Job Tracker queries the Name Node and finds the Data Nodes (location of the data to be used)
* Job Tracker then launches Task Trackers in the memory of all the Data Nodes as above to run the jobs
* Job Tracker gives the code to Task Tracker to run as a Task
* Task Tracker is responsible for creating the tasks & running the tasks
* In effect the Mapper of the Job is found here
* Once the Task is completed, the result from the Tasks is sent back to the Job Tracker
* Job Tracker also keeps a track of progress by each Task Tracker
* The Job Tracker also receives the results from each Task Tracker and aggregates the results
* In effect the Reducer of the Job is found here

**MR-V2 PROCESS**

* A Client invokes a Map Reduce, from a Calling Node (maybe a Data Node or an Extra Node in the cluster)
* An instance of Resource Manager is created in the memory of the Calling Node
* The Resource Manager then launches containers with appropriate resources (memory) with App Node Manager in memory of the Calling Node
* Along with this Application Master is invoked. Application Master is “pause” mode till all containers
* With Task Node Manager (as below) are created
* The Resource Manager queries the Name Node and finds the Data Nodes (location of the data used)
* The Resource Manager then launches containers with appropriate resources (memory) with Task Node Manager in all the Data Nodes as above to run the jobs
* Application Master gives the code to Task Node Manager to run as a Task
* Task Node Manager is responsible for creating & running tasks. In effect the Mapper of the Job is here
* Once the Task is completed, the result from the Tasks is sent back to the Application Master
* Application Master also keeps a track of progress by each Task Node Manager
* The Application Master also receives the results from each Task Node Manager and aggregates the results
* In effect the Reducer of the Job is found here
* Thus, from previous version, Job Tracker has been replaced by Resource Manager & Application Master
* From previous version, Task Tracker has been replaced by Task Node Managers

**MAP REDUCE FAILURE RECOVERY**

***MRv1***

* Task Failure new task is started by the Task Tracker
* Task Tracker Failure new Task Tracker is started by the Job Tracker
* Job Tracker Failure no recovery; single point of failure

***MRv2***

* Task Failure new task is started by Task Node Manager
* Task Node Manager Failure new container with Task Node Manager is created by Resource Manager this Task Node Manager is given the code and started by Application Master
* Application Master Failure New Application Master is started by App Node Manager
* App Node Manager Failure new container with App Node Manager is created by Resource Manager this App Node Manager invokes the Application Master
* Resource Manager Failure new resource manager with saved state is started