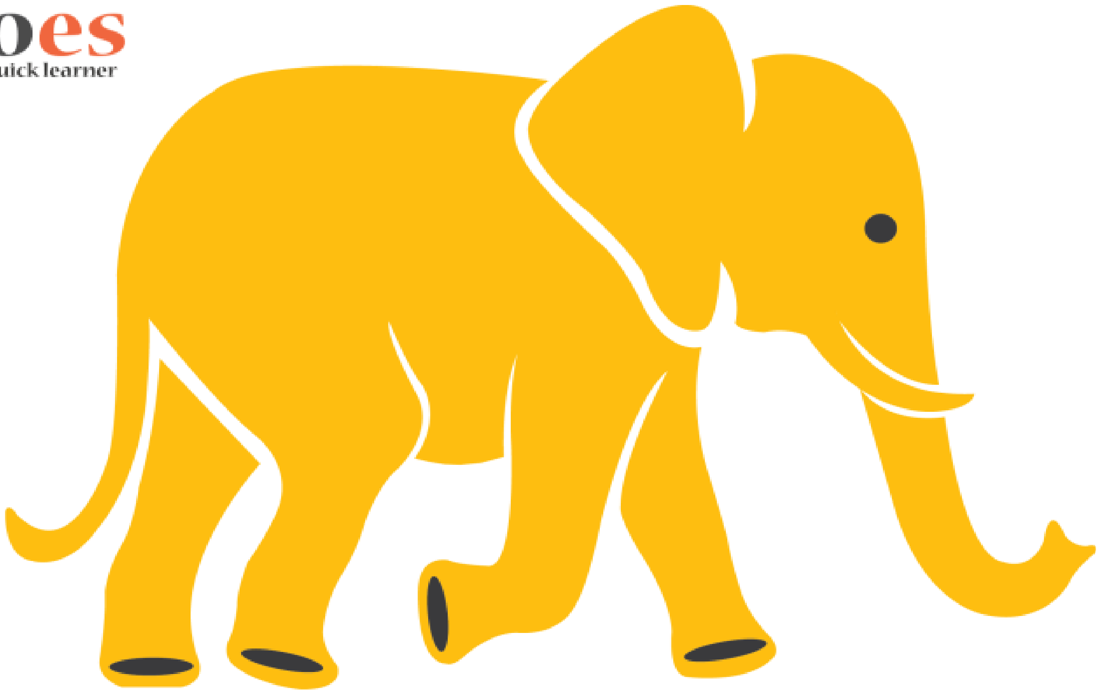


Module – 3
Introduction to Map Reduce Concepts

© kratoes

Session Objectives

This Session helps you to understand :

- Different Hadoop modes.
- Configuration files.
- Difference between blocks and input splits.
- Data distribution between traditional manner and mapreduce.
- MapReduce use cases.
- Concept of Combiner , Partitioner , Sort and Suffle.

© kratoes

Hadoop Modes

Hadoop can be run in 3 different modes.

- Standalone Mode
- Pseudo Distributed Mode(Single Node Cluster)
- Fully distributed mode (or multiple node cluster)

© kratoes

Hadoop Modes

Standalone Mode

- Default mode of Hadoop
- No Hadoop daemons, entire process runs in a single JVM
- Local file system is used for input and output
- Used for debugging purpose
- Suitable for running Hadoop programs during initial installation and Hadoop software testing
- No Custom Configuration is required in 3 hadoop (mapred-site.xml,core-site.xml, hdfs-site.xml) files.
- Standalone mode is much faster than Pseudo-distributed mode.

© kratoes

Hadoop Modes

Pseudo Distributed Mode (Single Node Cluster)

- Configuration is required in given 3 files for this mode.
- All Hadoop daemons are running in a single machine.
- Here one node will be used as Master Node / Data Node / Job Tracker /Task Tracker
- Used for Real Code to test in HDFS.

Hadoop Modes

Fully distributed mode (or multiple node cluster)

- This is a Production Phase
- Hadoop daemons run on a cluster of machines
- Data are used and distributed across many nodes.
- Different Nodes will be used as Master Node / Data Node / Job Tracker / Task Tracker

© kratoes

Self Understanding

Which of the following is not true for Hadoop Standalone Mode ?

1. Default mode of Hadoop
2. Used for debugging purpose
3. Hadoop daemons are running in a single machine
4. Standalone mode is much faster than other modes

© kratoes

Self Understanding

Which of the following is not true for Hadoop Standalone Mode ?

1. Default mode of Hadoop
2. Used for debugging purpose
3. Hadoop daemons are running in a single machine
4. Standalone mode is much faster than other modes

© kratoes

MapReduce

- MapReduce is a software framework/programming model introduced by Google.
- It is a sub-project of the Apache Hadoop Project.
- MapReduce processes and generates large data sets on clusters of computers.
- The framework takes care of scheduling tasks, monitoring them and re-executing any failed tasks.
- MapReduce framework beneficial because library routines can be used to create parallel programs without any worries about infra-cluster communication, task monitoring or failure handling processes.
- MapReduce runs on a large cluster of commodity machines and is highly scalable.
- It has several forms of implementation provided by multiple programming languages, like Java, C# and C++.
- The main advantage of the MapReduce framework is its fault tolerance.

© kratoes

Different phases of MapReduce

In Map Reduce, data is processed in Key-Value pairs

Map stage :

- The map or mapper’s job is to process the input data.
- Generally the input data is in the form of file or directory .
- The input file is passed to the mapper function line by line.
- The mapper processes the data and creates several small chunks of data.

© kratoes

Different phases of MapReduce

Shuffle and Sort

- Shuffle and Sorts intermediate data from all mappers. Data is sorted by Key.
- This stage happens when all the Map tasks are completed and before Reduce tasks start.

© kratoes

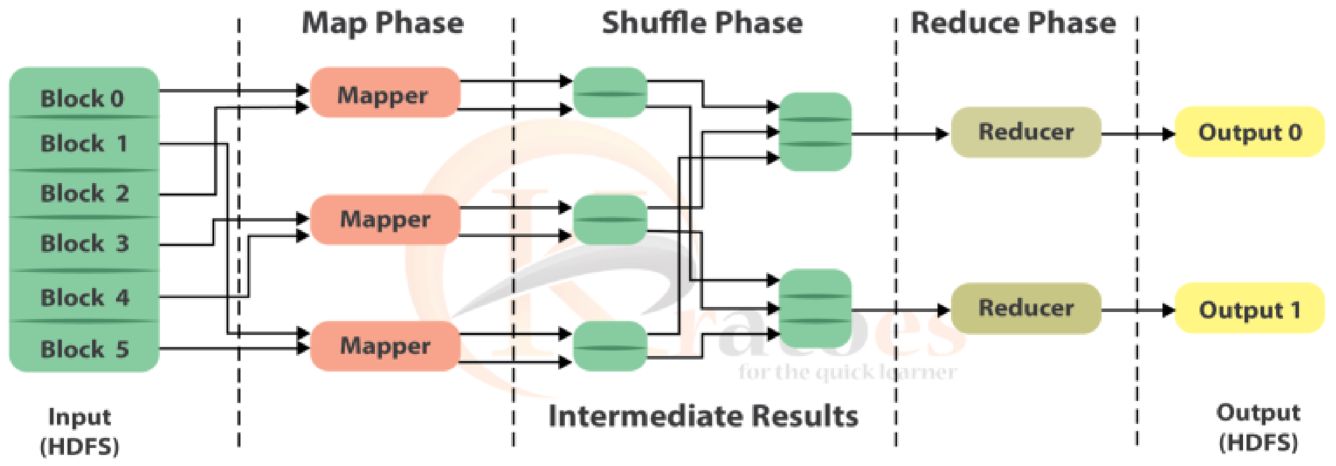
Different phases of MapReduce

Reduce stage

- The Reducer's job is to process the data that comes from the mapper.
- After processing, it produces a new set of output, which will be stored in the HDFS.

© kratoes

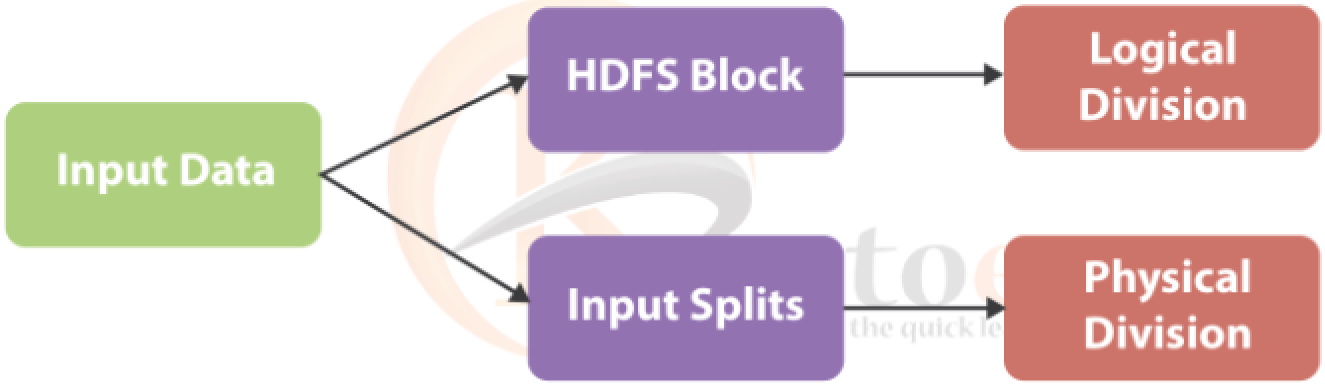
Different phases of MapReduce



© kratoes

Splits Vs. Blocks

- Block is the physical representation of data. Split is the logical representation of data present in Block.
- Block and split size can be changed in properties.
- Map reads data from Block through splits i.e. split act as a broker between
- Block and Mapper



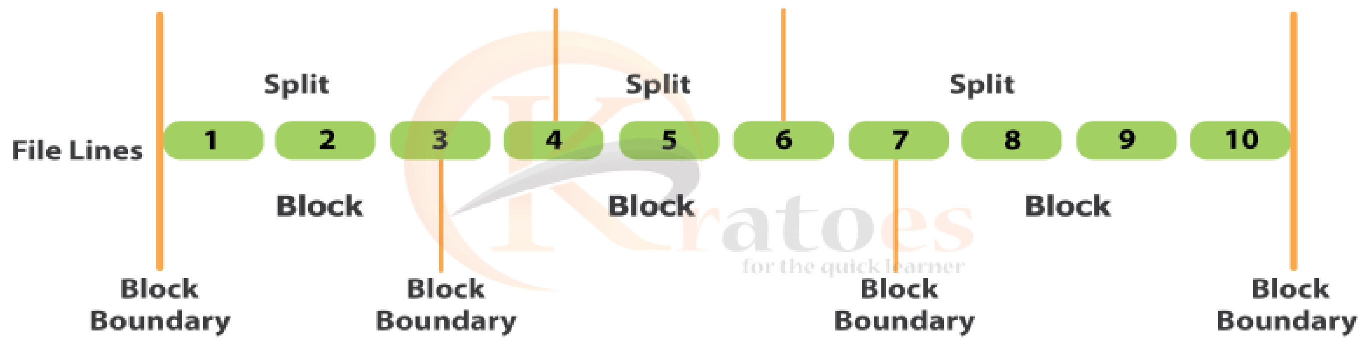
© kratoes

Input Split

- Input files contains multiple data records which are processed one at a time by Mappers.
- HDFS Framework splits the input file into multiple blocks (128MB) and store in data nodes. HDFS has no awareness of the content or type of these files. It is most likely that logical data records span multiple blocks when blocks are created.
- MapReduce framework calculates input splits, logical representation of data
  stored in file blocks, by identifying where the first whole record in a block begins and where t
  he last whole record in the block ends.

© kratoes

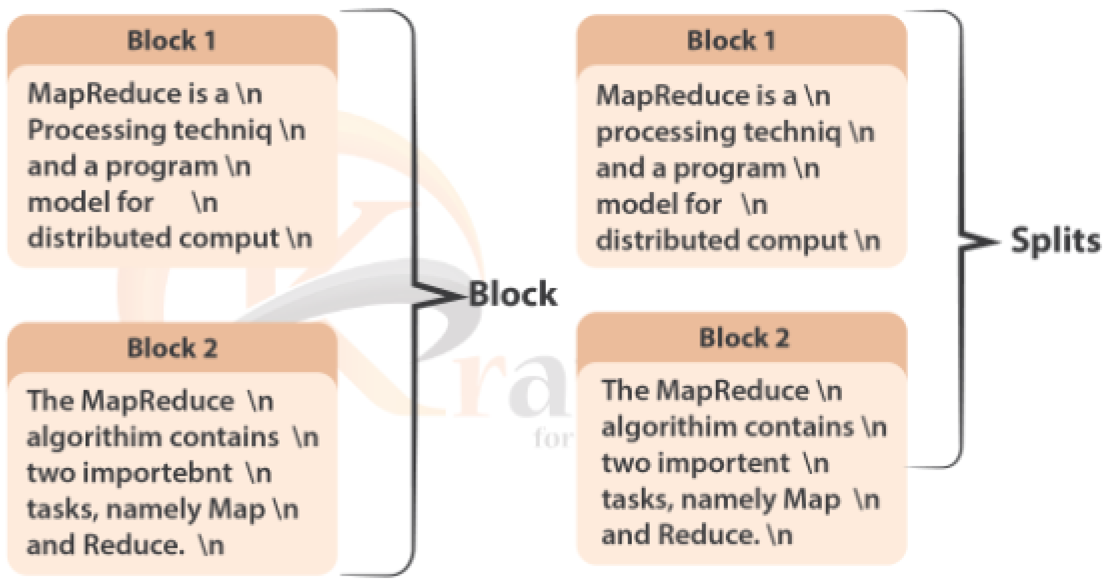
Input Split



© kratoes

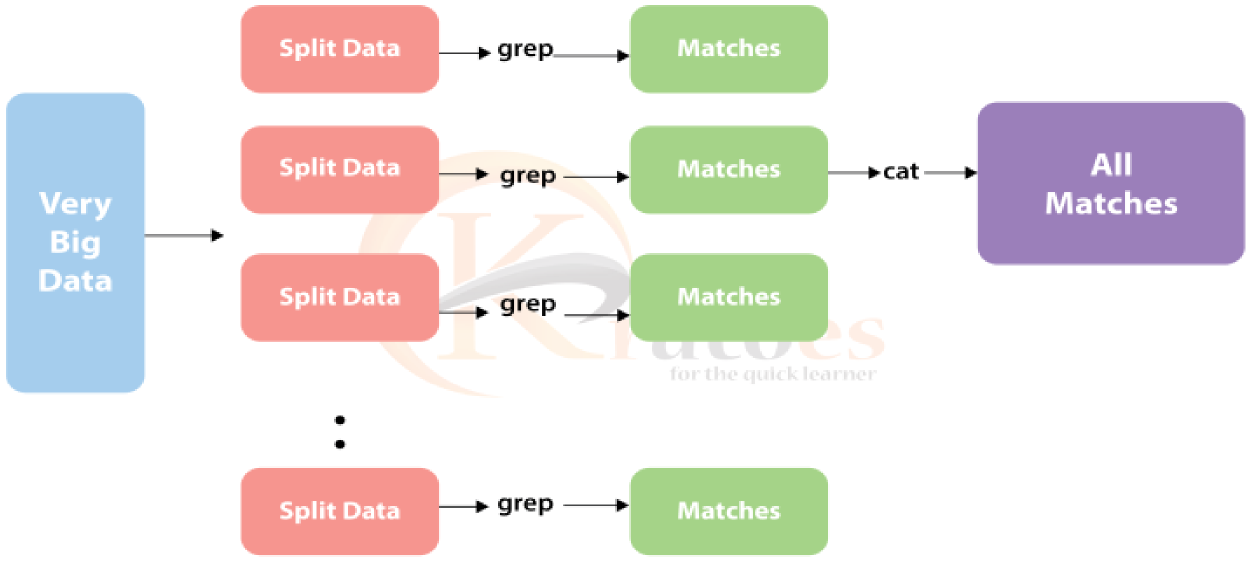
Blocks vs Slits

1TB file -Block Size is 64 MB
Number of blocks => 1TB/64 MB => 64MB, 64MB, 64MB, 64MB, 64MB, 64MB, 64MB, ……



© kratoes

Block vs Splits(Cont'd)



© kratoes

Key features of MapReduce

- Automatic parallelization and distribution of tasks
- Build in fault tolerance – In case a slave node dies, Resource Manager transfers the job to other available slave nodes where the block replica resides.
- MapReduce abstracts out/ hides all the housekeeping and distribution complexities from the developer. Developers concentrate on writing Map and Reduce functions.
- Taking processing to data.
- Processing data in parallel.

© kratoes

Self Understanding

Which of the below property gets configured on hadoop-env.sh ?

1. Replication factor.
2. Directory names to store hdfs files.
3. Host and port where MapReduce task runs.
4. Java Environment variables.

© kratoes

Self Understanding

Which of the below property gets configured on hadoop-env.sh ?

1. Replication factor.
2. Directory names to store hdfs files.
3. Host and port where MapReduce task runs.
4. Java Environment variables.

© kratoes

Self Understanding

In the secondary namenode the amount of memory needed is

1. Similar to that of primary node
2. Should be at least half of the primary node
3. Must be double of that of primary node
4. Depends only on the number of data nodes it is going to handle

© kratoes

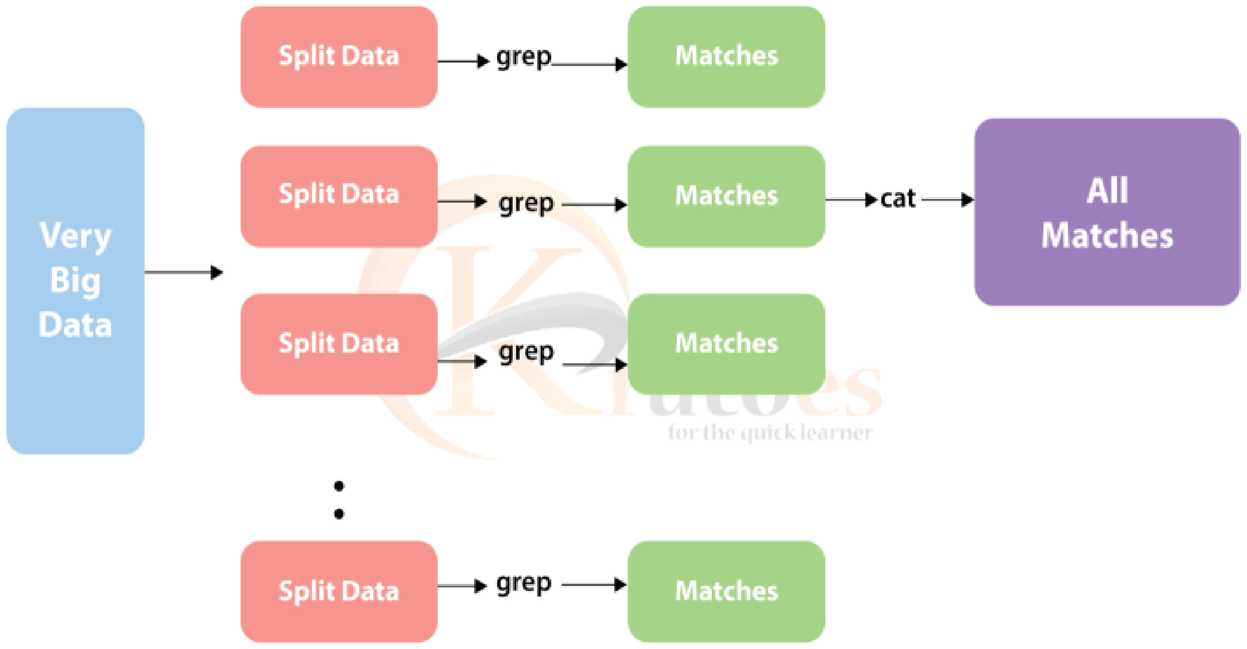
Self Understanding

In the secondary namenode the amount of memory needed is

1. Similar to that of primary node
2. Should be at least half of the primary node
3. Must be double of that of primary node
4. Depends only on the number of data nodes it is going to handle

© kratoes

Traditional Solution



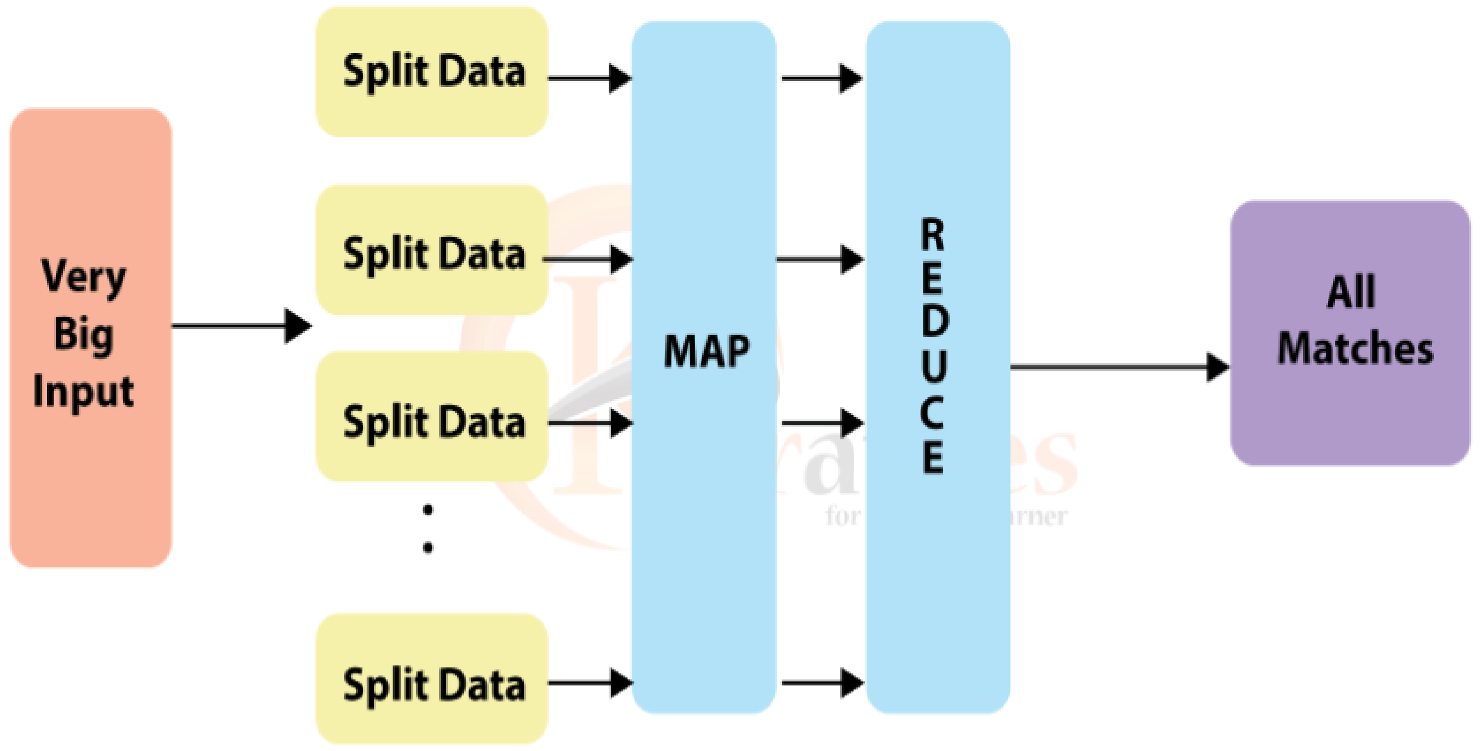
© kratoes

Problem with traditional solution

- RDBMS finds it challenging to handle such huge data volumes.
- To address this, RDBMS added more CPUs or more memory to the database management system to scale up vertically.
- RDBMs turn out to be very expensive to handle and store big data.
- The majority of the data comes in a semi-structured or unstructured format but RDBMs designed and structured to accommodate structured data only .
- The inability of relational databases to handle “big data” led to the emergence of new technologies.

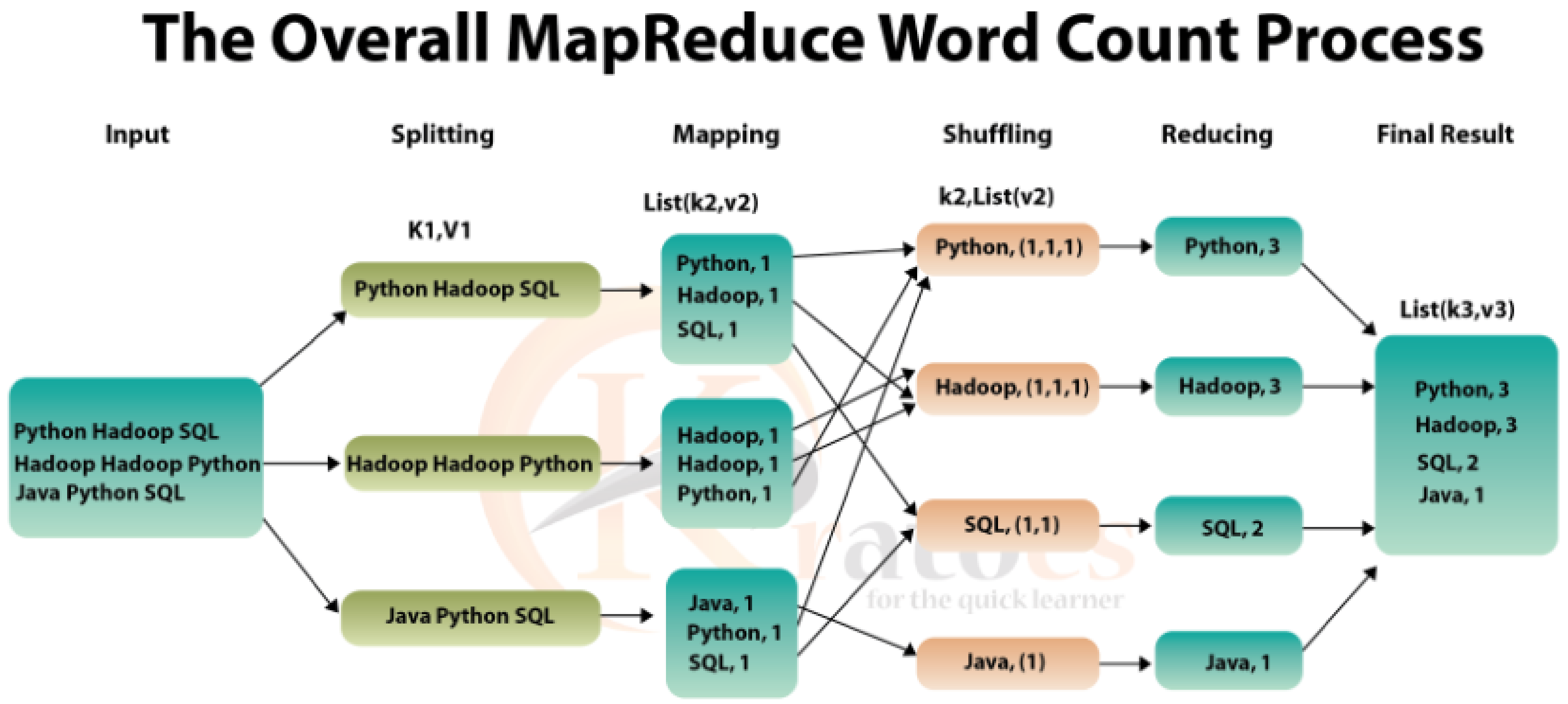
© kratoes

Map Reduce Solution



© kratoes

MapReduce Paradigm

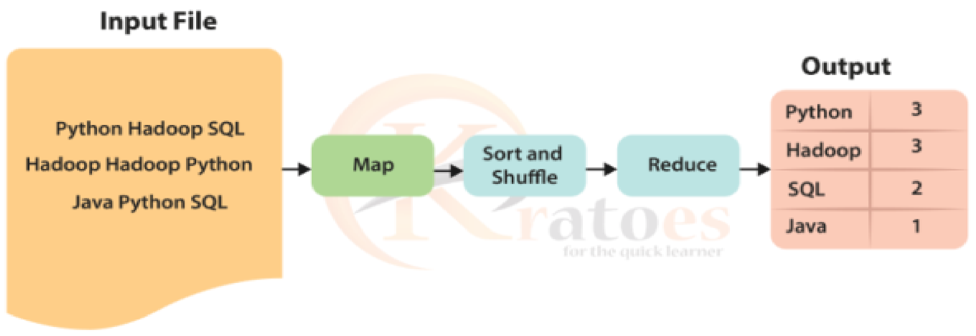


© kratoes

Example of Word Count Program in MapReduce

- Assume Input file contains below lines of data:
  - Dear Bear River
  - Car Car River
  - Deer Car Bear

- Word Count MapReduce program output:
  - Dear, 1
  - Bear, 2
  - River, 2
  - Car, 3



© kratoes

MapReduce – Execution flow

- Automatic parallelization and distribution of tasks
- Build in fault tolerance – In case a slave node dies, Resource Manager transfers the job to other available slave nodes where the block replica resides.
- MapReduce abstracts out/ hides all the housekeeping and distribution complexities from the developer. Developers concentrate on writing Map and Reduce functions.
- Taking processing to data
- Processing data in parallel

MapReduce Use Cases

List of use cases where Map Reduce is usually used

Case1 :NetApp collects over 600,000 data transactions weekly, consisting of unstructured logs and system diagnostic information.
Solution: A Cloudera Hadoop system captures the data and allows parallel processing of data.

Case 2 :A health IT company instituted a policy of saving seven years of historical claims and remit data, but its in-house database systems had trouble meeting the data retention requirement while processing millions of claims every day
Solution: A Hadoop system allows archiving seven years' claims and remit data.

© kratoes

MapReduce Use Cases

Case 1 : Collecting lots (billions) of data points from sensors / machines attached to the patients. This data was periodically purged before because storing this large volume of data on expensive storage was cost-prohibition.
Solution: Continuously streaming data from sensors/machines is collected and stored in HDFS. HDFS provides scalable data storage at reasonable cost.

Case 2 : Storing billions of mobile call records and providing real time access to the call records and billing information to customers.
Solution: HBase is used to store billions of rows of call record details. 30TB of data is added monthly.

© kratoes

Self Understanding

Mapper is a mandatory part of any MapReduce program.

1. True.
2. False.

© kratoes

Self Understanding

Mapper is a mandatory part of any MapReduce program.

1. True.
2. False.

© kratoes

Summary

- Different modes of Hadoop are Standalone Mode, Pseudo Distributed Mode and fully distributed mode.
- Configuration files to set-up the cluster.
- Difference between Blocks and Input Splits .
- Distributed computing in traditional manner Vs. MapReduce.
- MapReduce use cases.
- Understanding the concept of combiners, partitioners and shuffle & sort.

© kratoes

Kratoes
for the quick learner

Thank You!

© kratoes