SNS COLLEGE OF TECHNOLOGY

COIMBATORE

AN AUTONOMOUS INSTITUTION

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A' Grade

Approved by AICTE New Delhi & affiliated to the Anna University, Chennai

DEPARTMENT OF MCA

Course Name : 16CAT702 - BIG DATA ANALYTICS

Class : II Year / III Semester

Unit II - HADOOP

Topic IV – Hadoop Ecosystem

Fundamental Tools and Frameworks

- HDFS
- YARN (Yet Another Resource Negotiator)
- MapReduce
- Sqoop
- Flume

Hadoop framework - Fundamental Tools and Frameworks

- Hadoop is a framework that manages big data storage by means of parallel and distributed processing.
- Hadoop is comprised of various tools and frameworks that are dedicated to different sections of data management, like storing, processing, and analyzing.
- The Hadoop ecosystem covers Hadoop itself and various other related big data tools.

Hadoop Ecosystem

- Hadoop ecosystem and its various fundamental tools

HDFS

•In the traditional approach, all data was stored in a single central database. With the rise of big data, a single database was not enough to handle the task.

•The solution was to use a distributed approach to store the massive volume of information.

•Data was divided up and allocated to many individual databases.

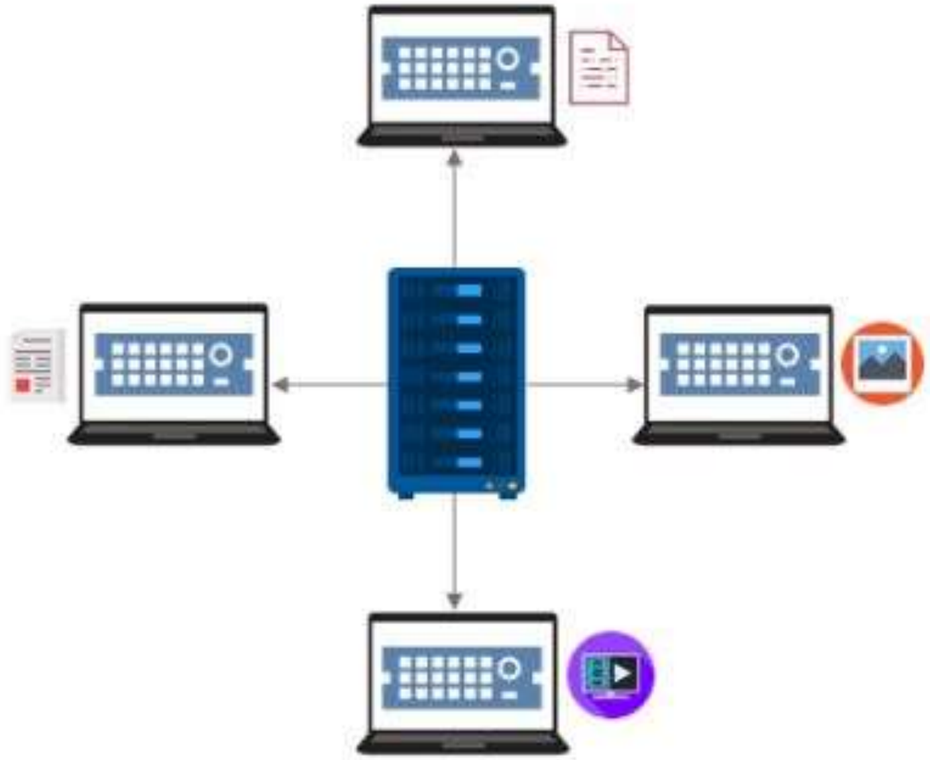
•HDFS is a specially designed file system for storing huge datasets in commodity hardware, storing information in different formats on various machines.

HDFS

There are two components in HDFS:

NameNode - NameNode is the master daemon. There is only one active NameNode. It manages the DataNodes and stores all the metadata.

DataNode - DataNode is the slave daemon. There can be multiple DataNodes. It stores the actual data.

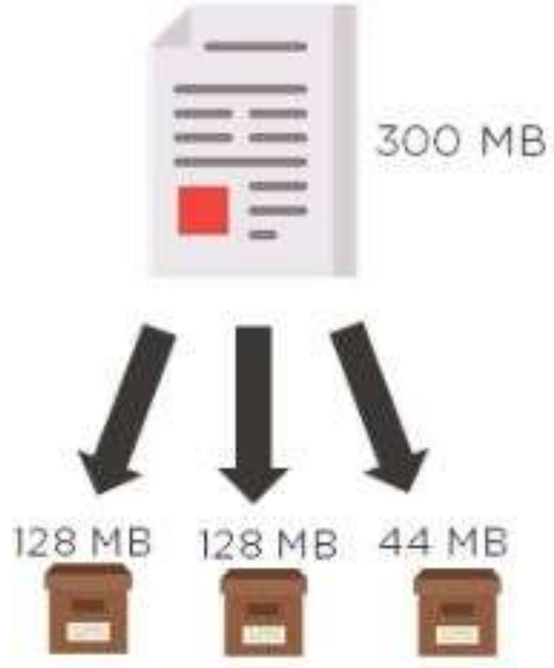


HDFS

The storage system has certain specifications.

HDFS splits the data into multiple blocks, defaulting to a maximum of 128 MB.

The default block size can be changed depending on the processing speed and the data distribution.



HDFS

As seen from the above image, we have 300 MB of data. This is broken down into 128 MB, 128 MB, and 44 MB.

The final block handles the remaining needed storage space, so it doesn't have to be sized at 128 MB.

This is how data gets stored in a distributed manner in HDFS.

YARN (Yet Another Resource Negotiator)

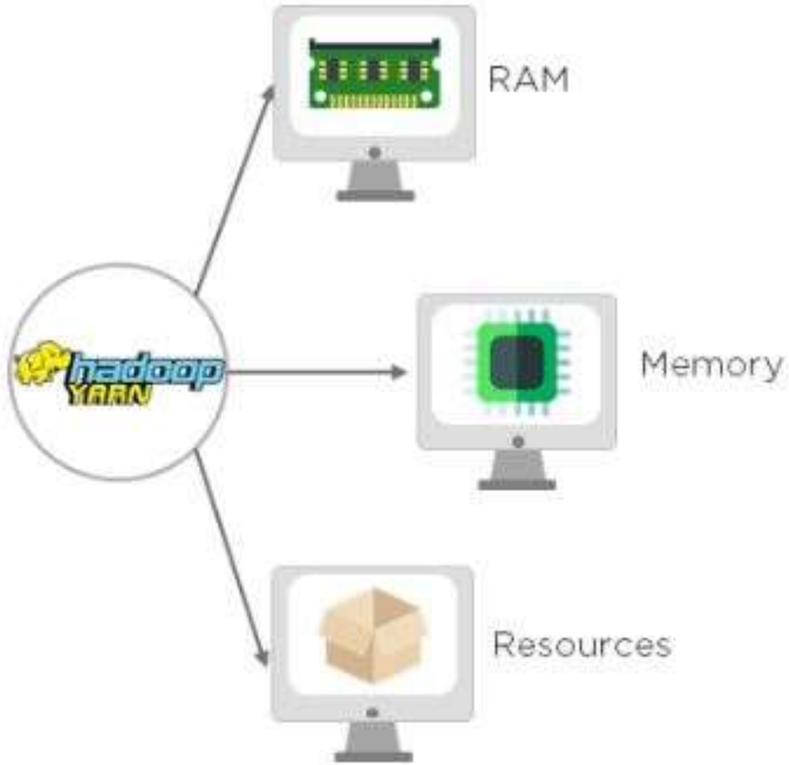
YARN is an acronym for Yet Another Resource Negotiator. It handles the cluster of nodes and acts as Hadoop's resource management unit. YARN allocates RAM, memory, and other resources to different applications.

YARN (Yet Another Resource Negotiator)

YARN has two components :

ResourceManager (Master) - This is the master daemon. It manages the assignment of resources such as CPU, memory, and network bandwidth.

NodeManager (Slave) - This is the slave daemon, and it reports the resource usage to the Resource Manager.

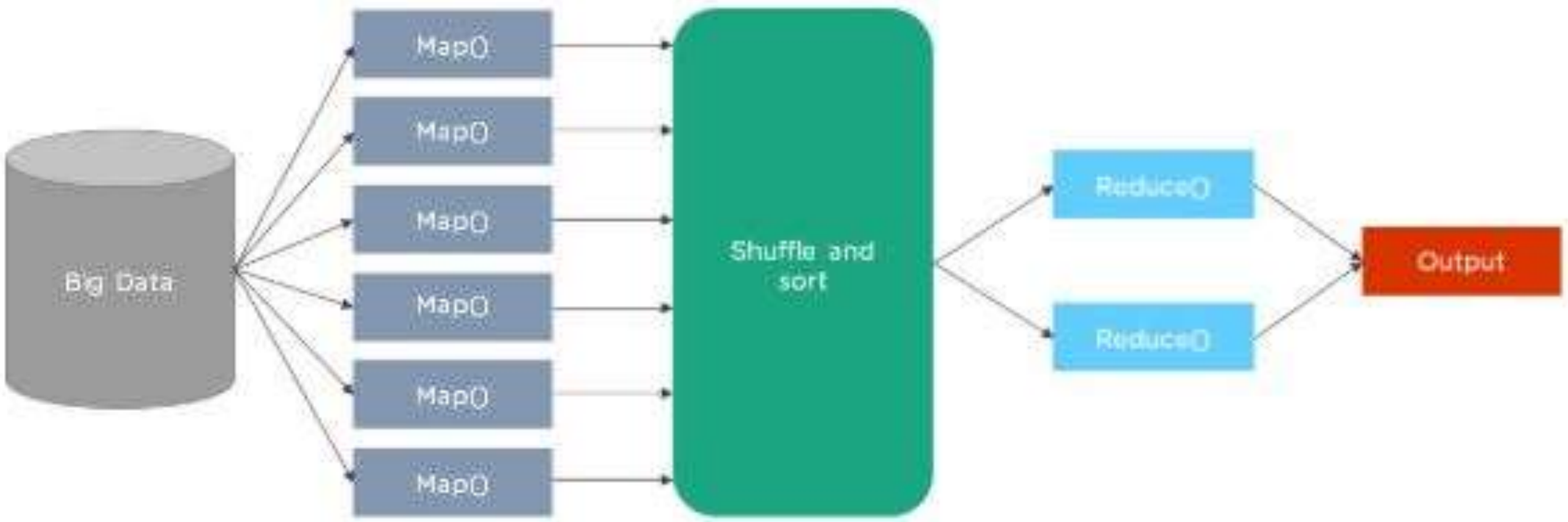


MapReduce

- Hadoop data processing is built on MapReduce, which processes large volumes of data in a parallelly distributed manner.

MapReduce

- With the help of the figure below, we can understand how MapReduce works:



MapReduce

- Big data that needs to be processed, with the intent of eventually arriving at an output. So in the beginning, input data is divided up to form the input splits.
- The first phase is the Map phase, where data in each split is passed to produce output values. In the shuffle and sort phase, the mapping phase's output is taken and grouped into blocks of similar data.
- Finally, the output values from the shuffling phase are aggregated. It then returns a single output value.

Sqoop

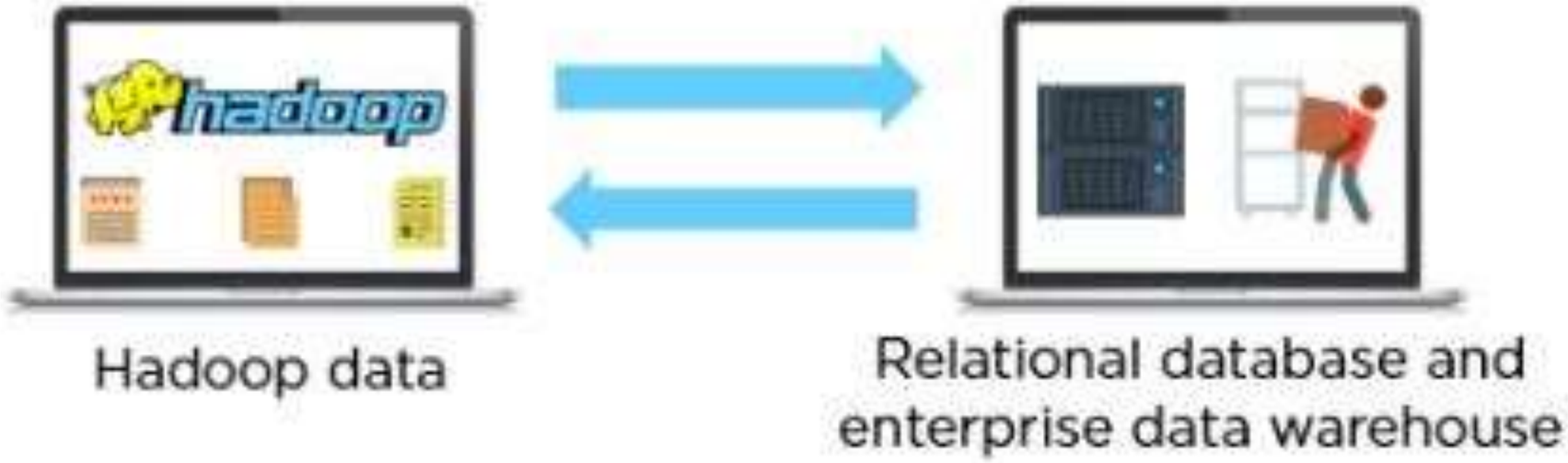
- Let us now see the data collection and ingestion tools, starting with Sqoop.
- Sqoop is used to transfer data between Hadoop and external datastores such as relational databases and enterprise data warehouses. It imports data from external datastores into HDFS, Hive, and HBase.

```
sqoop eval --connect jdbc:mysql://localhost/db --username root --
password cloudera --query "sql statement"


Example:
sqoop eval --connect jdbc:mysql://localhost/employee --username
root --password cloudera --query "select * from employee_table"
```

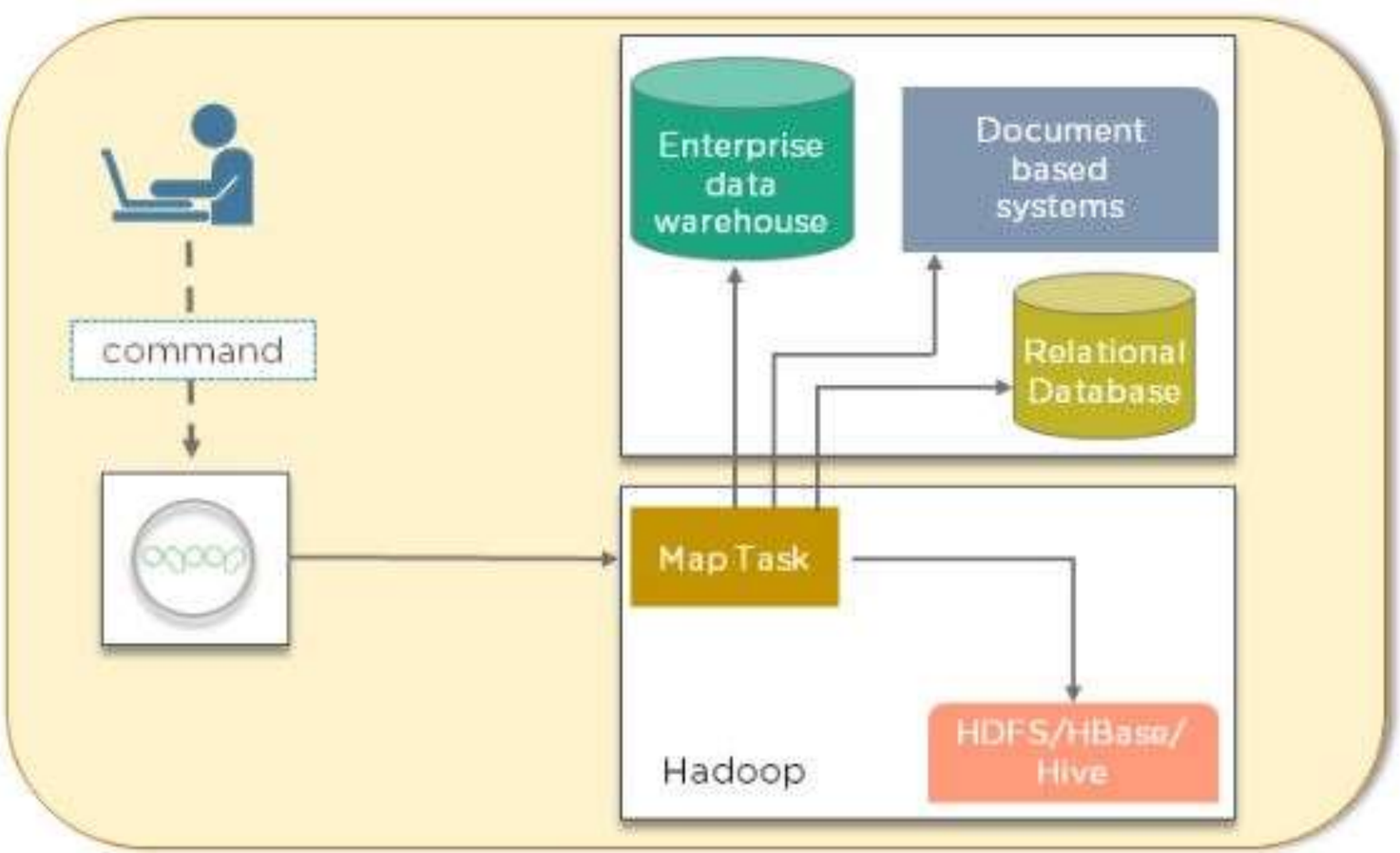
Sqoop

- As seen below, the client machine gathers code, which will then be sent to Sqoop. The Sqoop then goes to the Task Manager, which in turn connects to the enterprise data warehouse, documents based systems, and RDBMS. It can map those tasks into Hadoop.



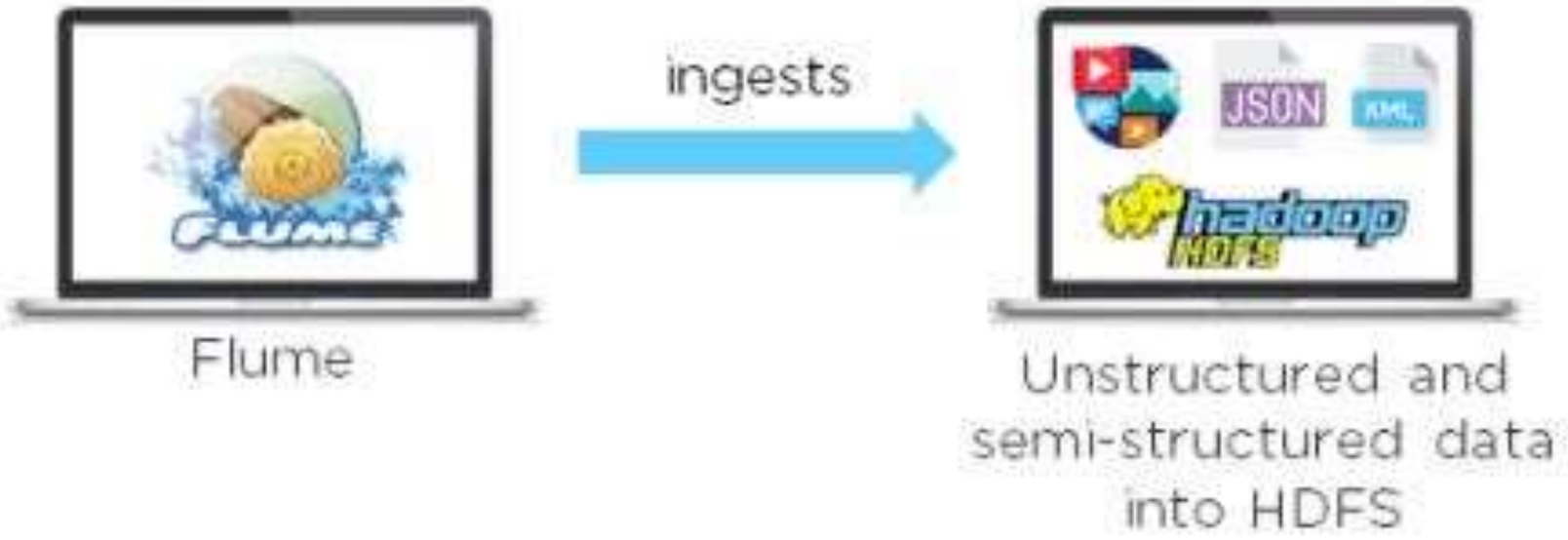
Flume

- Flume


Enterprise data warehouse
Document based systems
Relational Database
command
Map Task
Hadoop
HDFS/HBase/ Hive

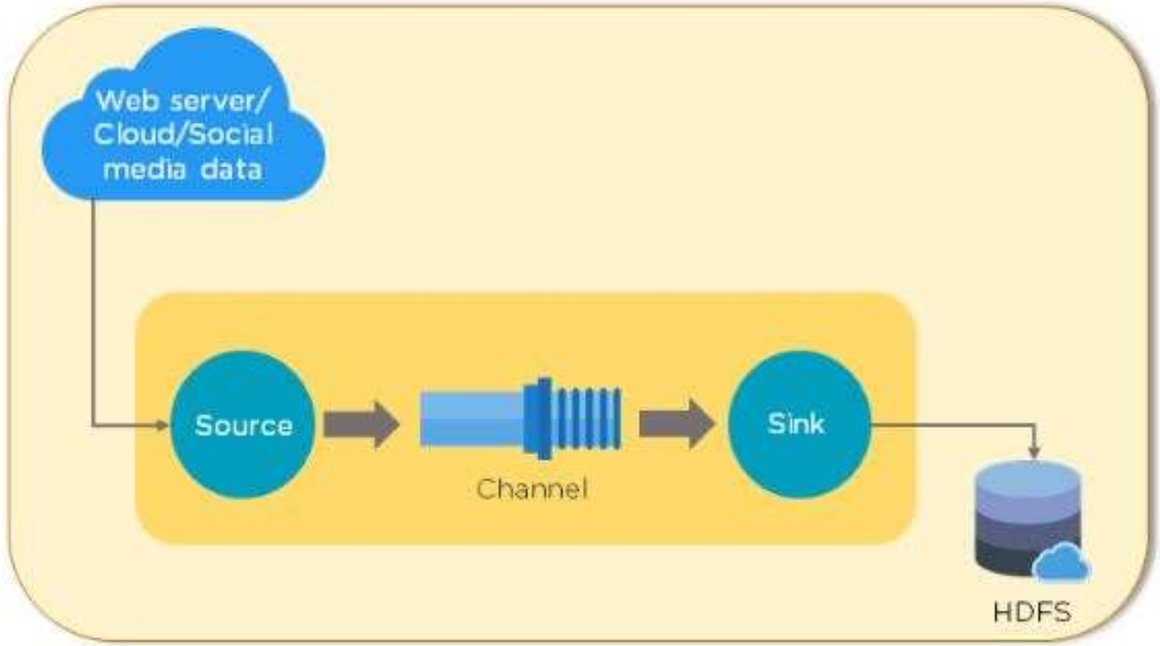
Flume

- Flume is another data collection and ingestion tool, a distributed service for collecting, aggregating, and moving large amounts of log data. It ingests online streaming data from social media, logs files, web server into HDFS.



Flume

- As you can see below, data is taken from various sources, depending on your organization's needs. It then goes through the source, channel, and sink. The sink feature ensures that everything is in sync with the requirements. Finally, the data is dumped into HDFS.



Pig

•Let us now have a look at Hadoop's scripting languages and query languages.

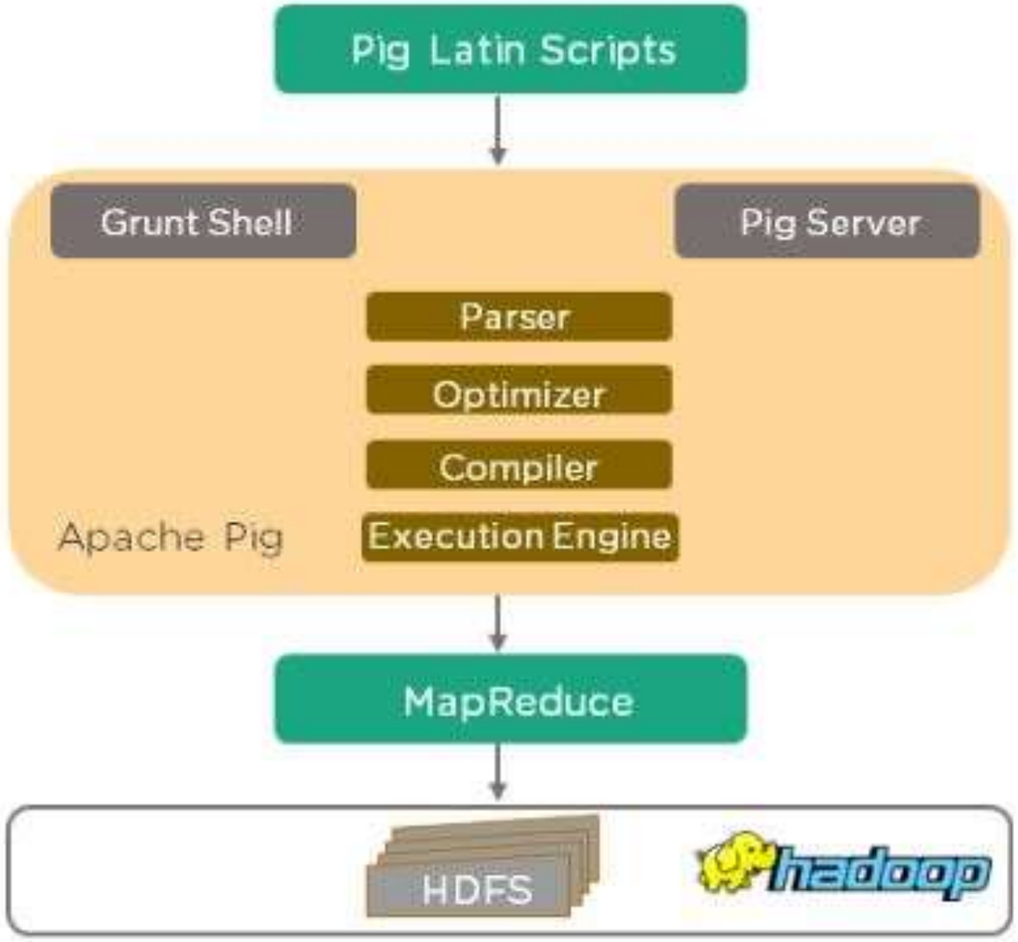
It consists of:

Pig Latin - This is the language for scripting

Pig Latin Compiler - This converts Pig Latin code into executable code

Pig also provides Extract, Transfer, and Load (ETL), and a platform for building data flow. Did you know that ten lines of Pig Latin script equals approximately 200 lines of MapReduce job? Pig uses simple, time-efficient steps to analyze datasets. Let's take a closer look at Pig's architecture.

Pig


Pig Latin Scripts
Grunt Shell
Pig Server
Parser
Optimizer
Compiler
Execution Engine
Apache Pig
MapReduce
HDFS
hadoop

Pig

- Programmers write scripts in Pig Latin to analyze data using Pig. Grunt Shell is Pig's interactive shell, used to execute all Pig scripts.
- If the Pig script is written in a script file, the Pig Server executes it. The parser checks the syntax of the Pig script, after which the output will be a DAG (Directed Acyclic Graph).
- The DAG (logical plan) is passed to the logical optimizer. The compiler converts the DAG into MapReduce jobs.
- The MapReduce jobs are then run by the Execution Engine.
- The results are displayed using the "DUMP" statement and stored in HDFS using the "STORE" statement.

Hive

- Hive uses SQL (Structured Query Language) to facilitate the reading, writing, and management of large datasets residing in distributed storage. The hive was developed with a vision of incorporating the concepts of tables and columns with SQL since users were comfortable with writing queries in SQL.

Hive

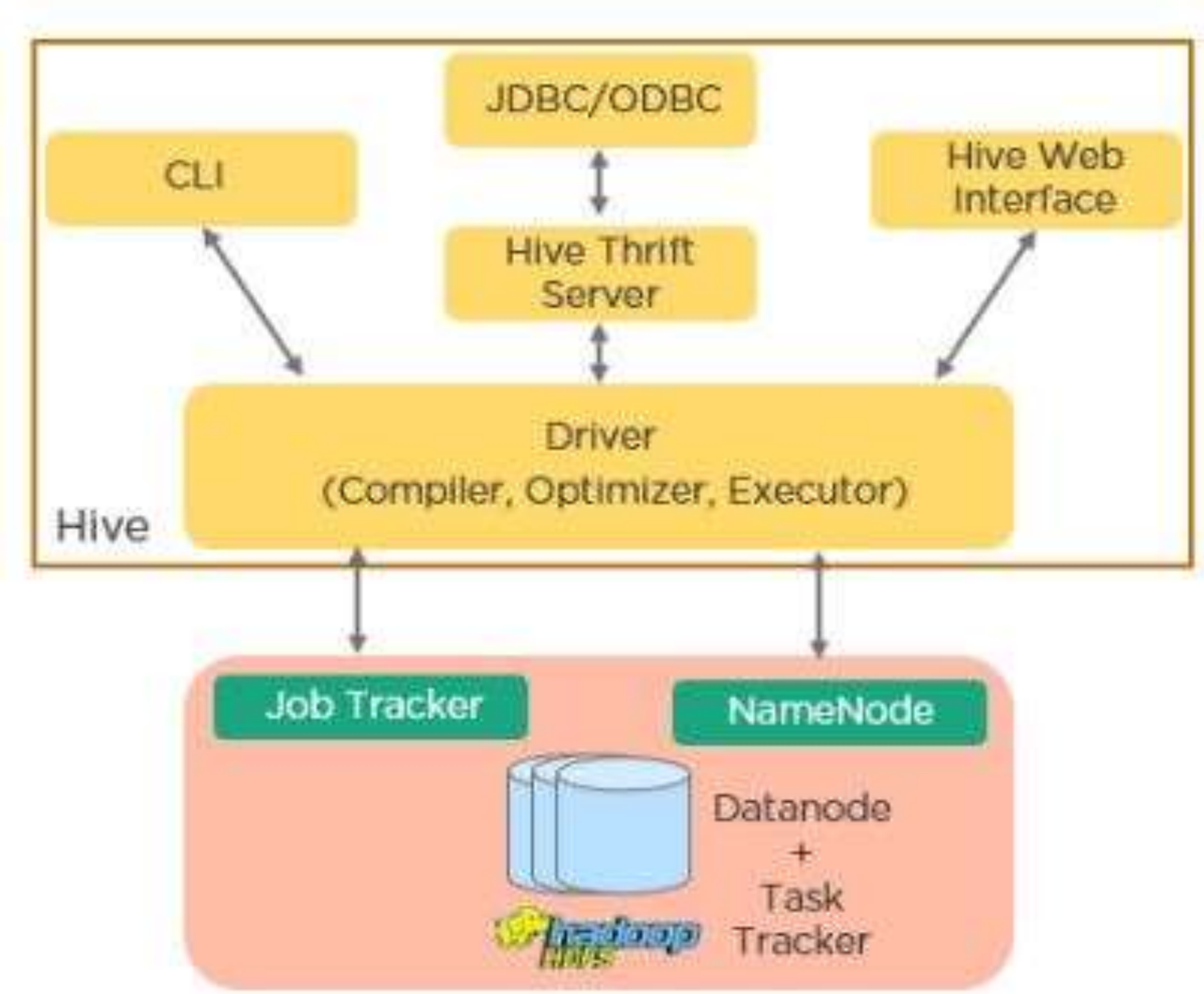
Apache Hive has two major components:

- Hive Command Line
- JDBC/ ODBC driver

The Java Database Connectivity (JDBC) application is connected through JDBC Driver, and the Open Database Connectivity (ODBC) application is connected through ODBC Driver. Commands are executed directly in CLI. Hive driver is responsible for all the queries submitted, performing the three steps of compilation, optimization, and execution internally. It then uses the MapReduce framework to process queries.

Hive

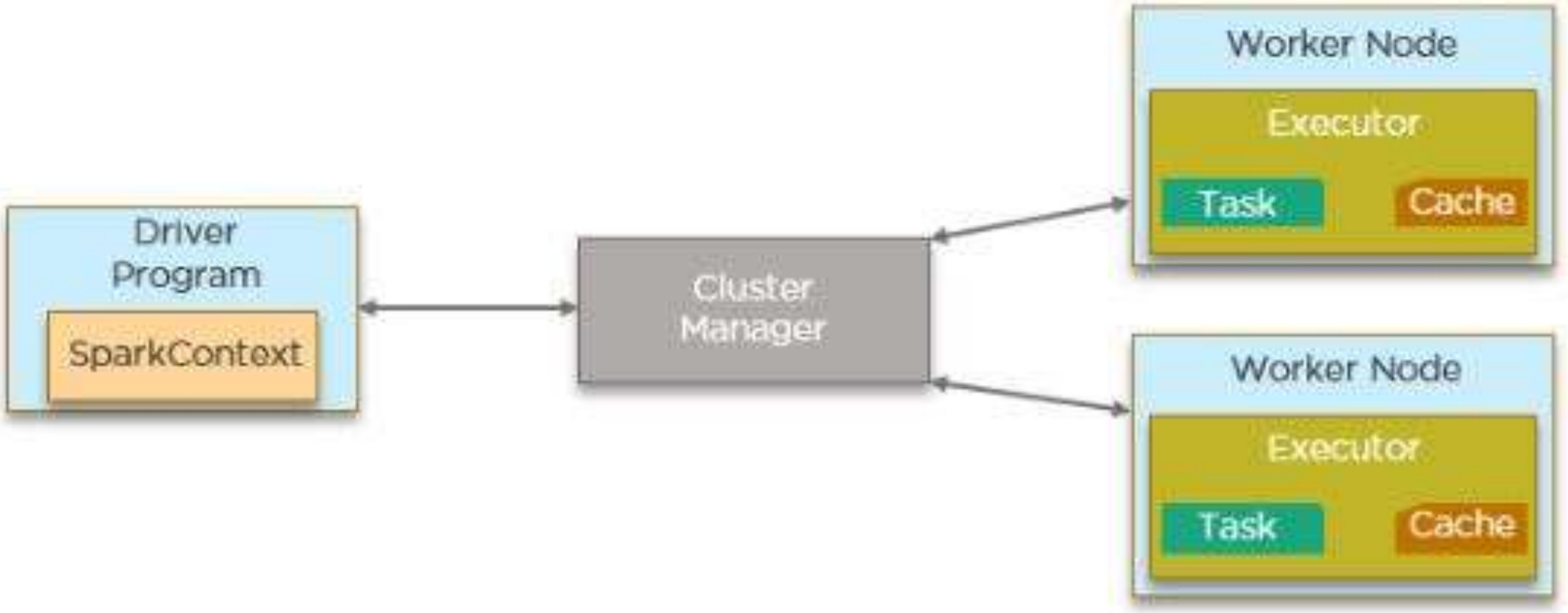
Hive's architecture



Spark

Spark is a huge framework in and of itself, an open-source distributed computing engine for processing and analyzing vast volumes of real-time data. It runs 100 times faster than MapReduce. Spark provides an in-memory computation of data, used to process and analyze real-time streaming data such as stock market and banking data, among other things.

Spark


Worker Node
Executor
Task
Cache
Driver
Program
SparkContext
Cluster
Manager
Worker Node
Executor
Task
Cache

Spark

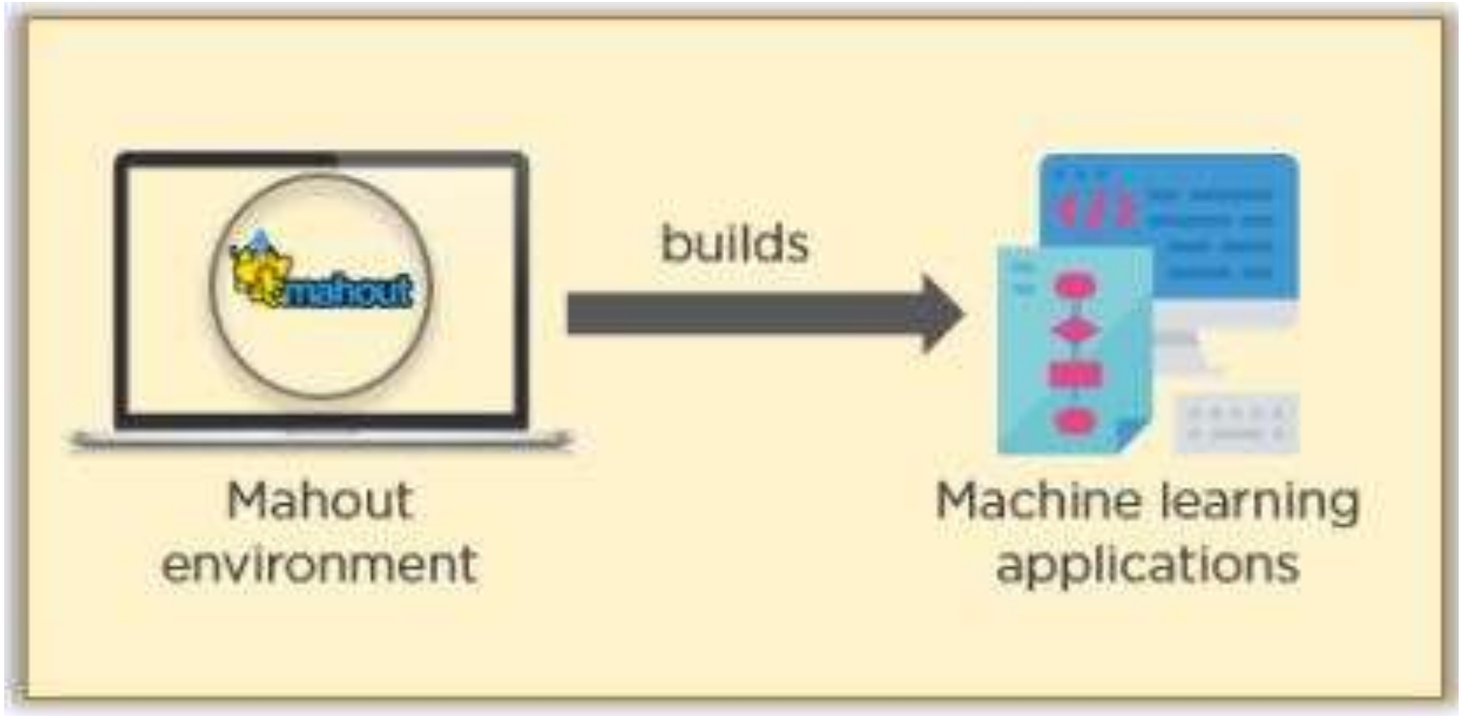
As seen from the above image, the MasterNode has a driver program. The Spark code behaves as a driver program and creates a SparkContext, which is a gateway to all of the Spark functionalities. Spark applications run as independent sets of processes on a cluster. The driver program and Spark context take care of the job execution within the cluster. A job is split into multiple tasks that are distributed over the worker node. When an RDD is created in the Spark context, it can be distributed across various nodes. Worker nodes are slaves that run different tasks. The Executor is responsible for the execution of these tasks. Worker nodes execute the tasks assigned by the Cluster Manager and return the results to the SparkContext.

Mahout

Hadoop Machine Learning and its different permutations

Mahout

Mahout is used to create scalable and distributed machine learning algorithms such as clustering, linear regression, classification, and so on. It has a library that contains built-in algorithms for collaborative filtering, classification, and clustering.



Ambari

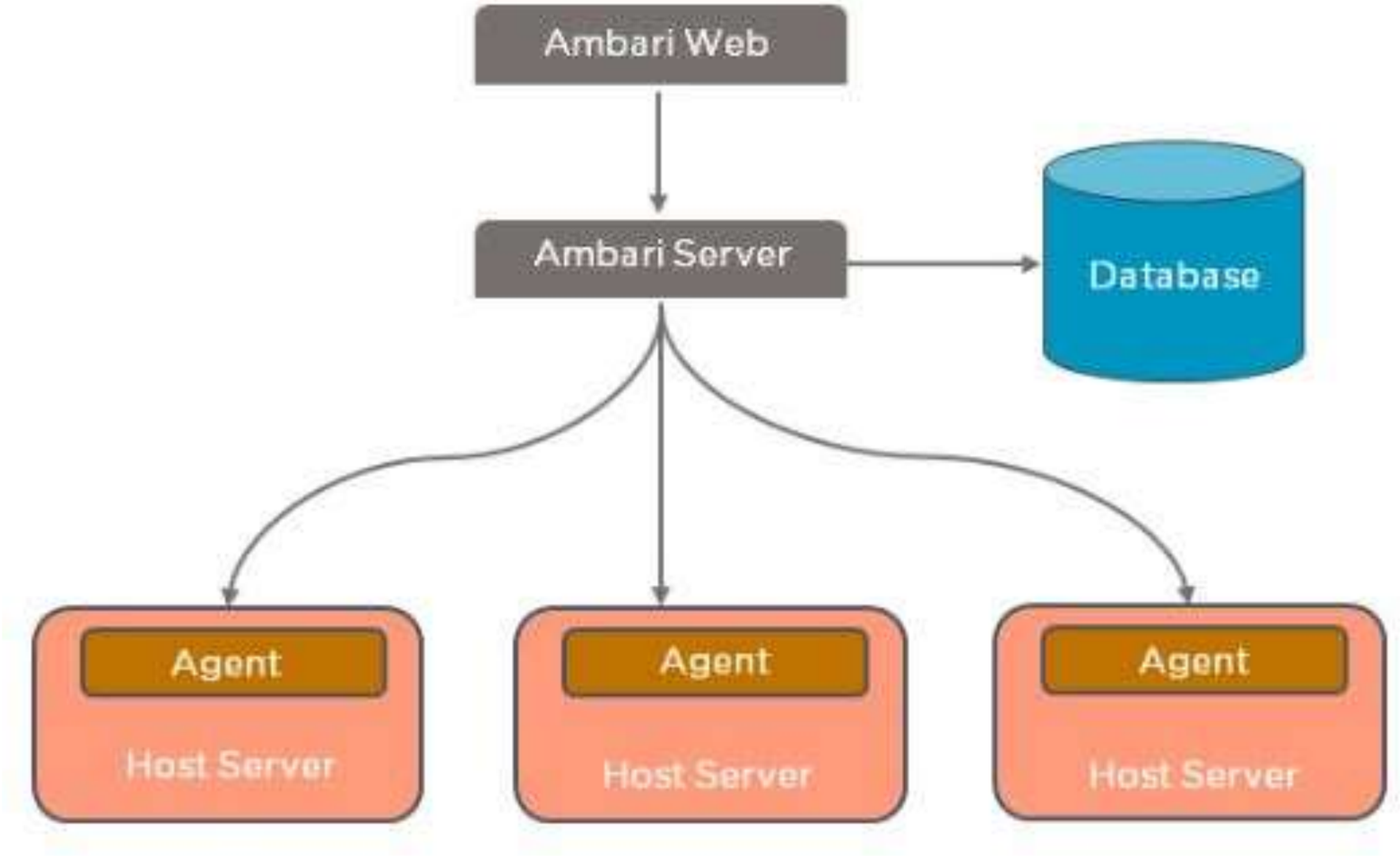
Apache Ambari. It is an open-source tool responsible for keeping track of running applications and their statuses. Ambari manages, monitors, and provisions Hadoop clusters. Also, it also provides a central management service to start, stop, and configure Hadoop services.

Ambari

the Ambari web, which is your interface, is connected to the Ambari server. Apache Ambari follows a master/slave architecture. The master node is accountable for keeping track of the state of the infrastructure. For doing this, the master node uses a database server that can be configured during the setup time. Most of the time, the Ambari server is located on the MasterNode, and is connected to the database. This is where agents look into the host server. Agents run on all the nodes that you want to manage under Ambari. This program occasionally sends heartbeats to the master node to show its aliveness. By using Ambari Agent, the Ambari Server is able to execute many tasks.

Ambari

The Ambari web



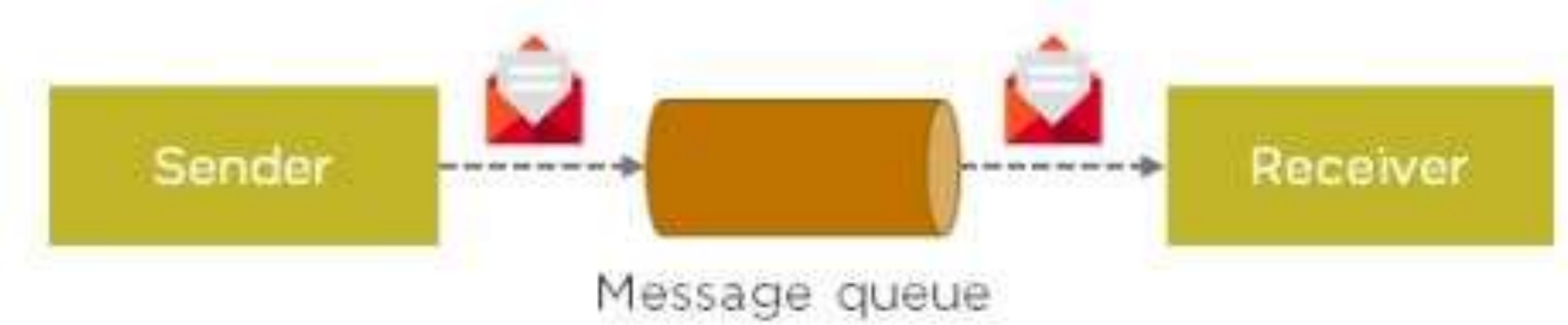
Kafka

We have two more data streaming services, Kafka and Apache Storm.

Kafka is a distributed streaming platform designed to store and process streams of records. It is written in Scala. It builds real-time streaming data pipelines that reliably get data between applications, and also builds real-time applications that transform data into streams.

Kafka

Kafka uses a messaging system for transferring data from one application to another. As seen below, we have the sender, the message queue, and the receiver involved in data transfer



Storm

The storm is an engine that processes real-time streaming data at a very high speed. It is written in Clojure. A storm can handle over 1 million jobs on a node in a fraction of a second. It is integrated with Hadoop to harness higher throughputs.

Ranger

let us take a look at the security frameworks in the Hadoop ecosystem.

Ranger is a framework designed to enable, monitor, and manage data security across the Hadoop platform. It provides centralized administration for managing all security-related tasks. Ranger standardizes authorization across all Hadoop components, and provides enhanced support for different authorization methods like role-based access control, and attributes based access control, to name a few.

Knox

Apache Knox is an application gateway used in conjunction with Hadoop deployments, interacting with REST APIs and UIs. The gateway delivers three types of user-facing services:

Proxying Services - This provides access to Hadoop via proxying the HTTP request

Authentication Services - This gives authentication for REST API access and WebSSO flow for user interfaces

Client Services - This provides client development either via scripting through DSL or using the Knox shell classes

Oozie

Let us now take a look at the workflow system, Oozie.

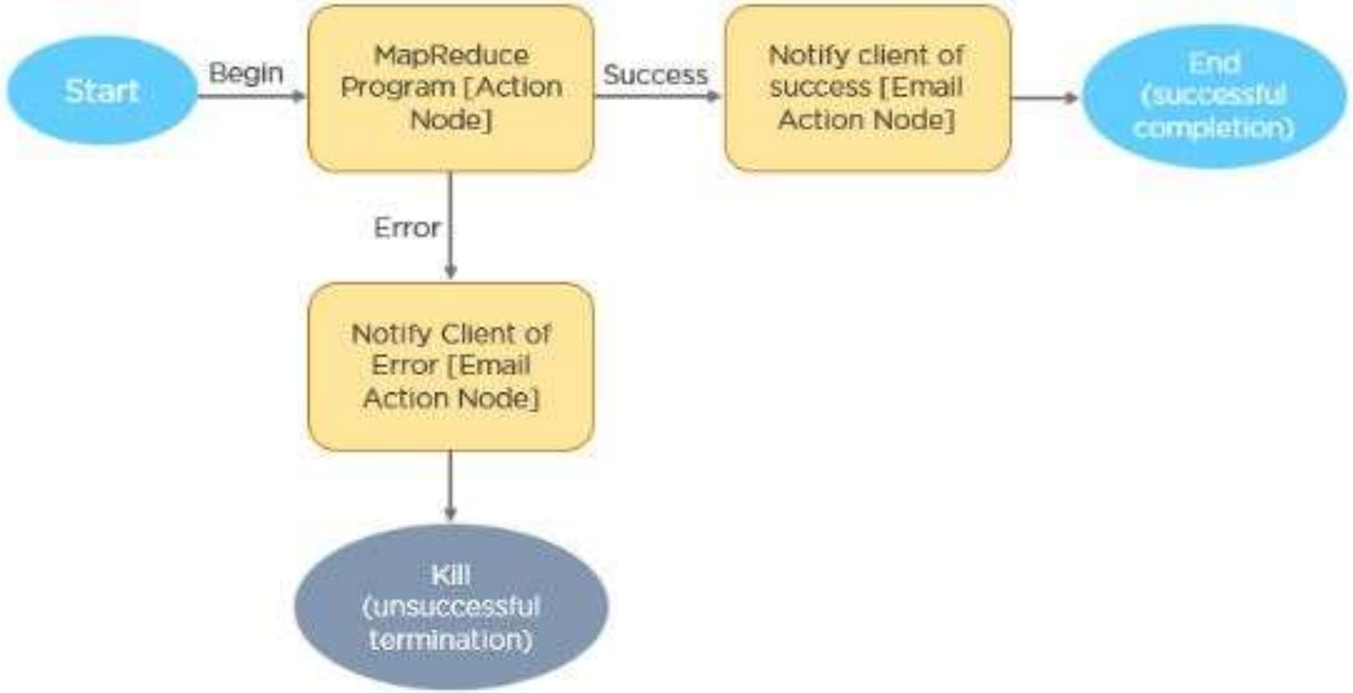
Oozie is a workflow scheduler system used to manage Hadoop jobs. It consists of two parts:

Workflow engine - This consists of Directed Acyclic Graphs (DAGs), which specify a sequence of actions to be executed

Coordinator engine - The engine is made up of workflow jobs triggered by time and data availability

Oozie

As seen from the flowchart below, the process begins with the MapReduce jobs. This action can either be successful, or it can end in an error. If it is successful, the client is notified by an email. If the action is unsuccessful, the client is similarly notified, and the action is terminated.



Reference

1. https://www.simplilearn.com/tutorials/hadoop-tutorial/hadoop-ecosystem

THANK YOU