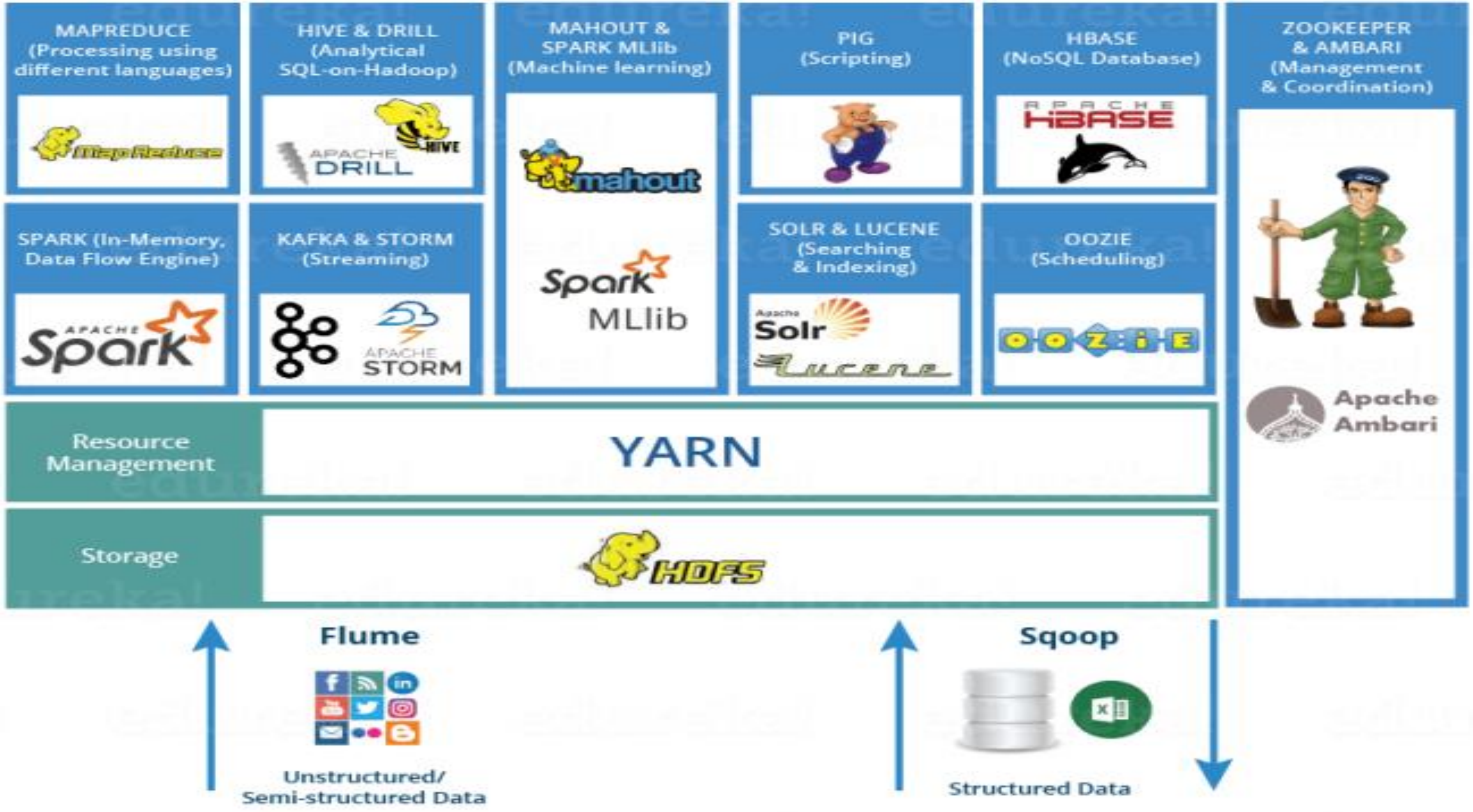
HADOOP ECOSYSTEM

Hadoop Ecosystem is neither a programming language nor a service, it is a platform or framework which solves big data problems. You can consider it as a suite which encompasses a number of services (ingesting, storing, analyzing and maintaining) inside it.

Below are the Hadoop components, that together form a Hadoop ecosystem.


MAPREDUCE (Processing using different languages)
MapReduce
HIVE & DRILL (Analytical SQL-on-Hadoop)
APACHE DRILL
HIVE
MAHOUT & SPARK MLlib (Machine learning)
mahout
Spark MLlib
PIG (Scripting)
HBASE (NoSQL Database)
APACHE HBASE
ZOOKEEPER & AMBARI (Management & Coordination)
Apache Ambari
SPARK (In-Memory, Data Flow Engine)
APACHE Spark
KAFKA & STORM (Streaming)
APACHE STORM
SOLR & LUCENE (Searching & Indexing)
Apache Solr
Lucene
OOZIE (Scheduling)
OOZIE
Resource Management
YARN
Storage
HDFS
Flume
Unstructured/ Semi-structured Data
Sqoop
Structured Data

Difference between Hadoop vs Hadoop Ecosystem

Hadoop is a programming framework used in the world of big data to solve significant big data challenges such as storing and processing.

The Hadoop ecosystem consists of tools and frameworks that can integrate with Hadoop. There are a lot of tools that come under the Hadoop ecosystem and each of them has its own functionalities.

Some of the tools are:

- **MapReduce and YARN**
- **Apache Spark -> In-memory Data Processing**
- **Sqoop and Flume** for data collection and ingestion
- **Hive and Pig** for query-based processing
- **HBase and Mongo DB** for NoSQL database
- **Mahout and Spark MLlib** for machine learning algorithms
- **Zookeeper** for managing cluster
- **Oozie** for job scheduling

HDFS -> Hadoop Distributed File System

YARN -> Yet Another Resource Negotiator

MapReduce -> Data processing using programming

Apache Spark -> In-memory Data Processing

Zookeeper -> Managing Cluster

Sqoop -> Data Ingesting Services

Introduction to Languages

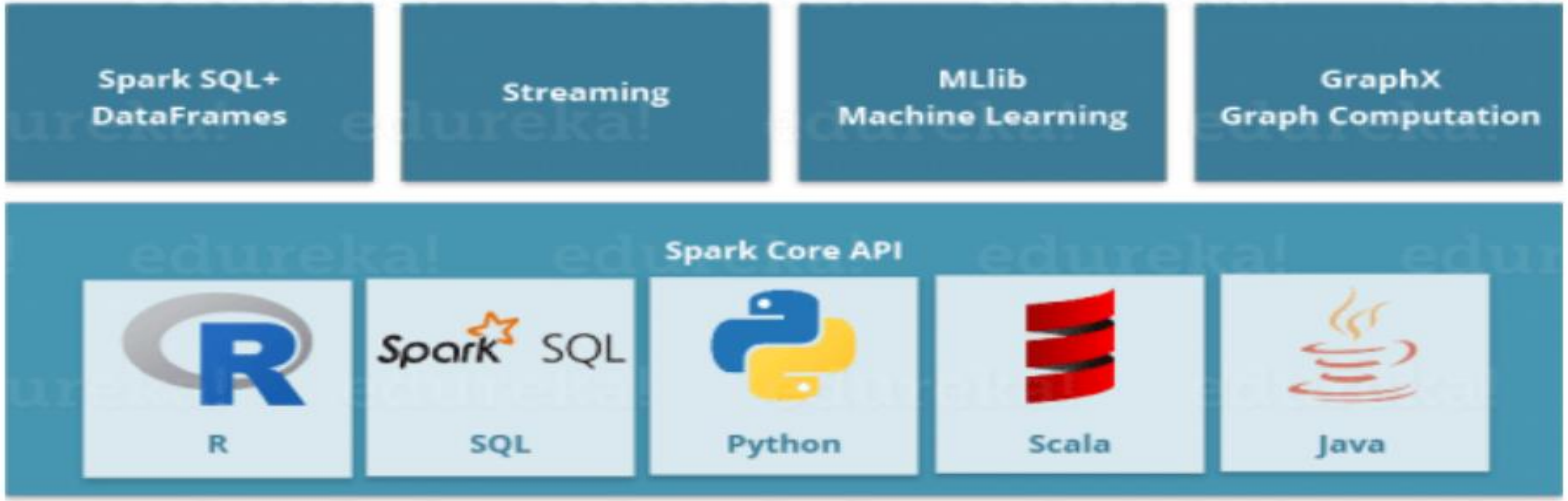
PIG, HIVE-> Data Processing Services using Query (SQL-like)

NOSQL Databases:

- Cassandra,
- Mongo,
- Cloudera,
- CouchDB,
- HBase -> NoSQL Database

- Apache Spark's streaming APIs allow for real-time data ingestion, while Hadoop MapReduce can store and process the data within the architecture. Spark can then be used to perform real-time stream processing or batch processing on the data stored in Hadoop.

- Apache Spark is a framework for real time data analytics in a distributed computing environment.
- The Spark is written in Scala and was originally developed at the University of California, Berkeley.
- It executes in-memory computations to increase speed of data processing over MapReduce.
- It is 100x faster than Hadoop for large scale data processing by exploiting in-memory computations and other optimizations. Therefore, it requires high processing power than Map-Reduce



- As you can see, Spark comes packed with high-level libraries, including support for R, SQL, Python, Scala, Java etc. These standard libraries increase the seamless integrations in complex workflow. Over this, it also allows various sets of services to integrate with it like MLlib, GraphX, SQL + Data Frames, Streaming services etc. to increase its capabilities.

- Apache Spark best fits for real time processing, whereas Hadoop was designed to store unstructured data and execute batch processing over it. When we combine, Apache Spark's ability, i.e. high processing speed, advance analytics and multiple integration support with Hadoop's low cost operation on commodity hardware, it gives the best results. ★★★

- That is the reason why, Spark and Hadoop are used together by many companies for processing and analyzing their Big Data stored in HDFS

There are five main components of Apache Spark:

- **Apache Spark Core:** It is responsible for functions like scheduling, input and output operations, task dispatching, etc.
- **Spark SQL:** This is used to gather information about structured data and how the data is processed.
- **Spark Streaming:** This component enables the processing of live data streams.
- **Machine Learning Library:** The goal of this component is scalability and to make machine learning more accessible.
- **GraphX:** This has a set of APIs that are used for facilitating graph analytics tasks.

Advantage of Spark:

1.Perfect for interactive processing, iterative processing and event steam processing
2.Flexible and powerful
3.Supports for sophisticated analytics
4.Executes batch processing jobs faster than MapReduce
5.Run on Hadoop alongside other tools in the Hadoop ecosystem

Disadvantage of Spark:

1.Consumes a lot of memory
2. Issues with small file
3.Less number of algorithms
4. Higher latency compared to Apache fling

Latency is the time it takes for data to pass from one point on a network to another.

Basis	Hadoop	Spark
Processing Speed & Performance	Hadoop's MapReduce model reads and writes from a disk, thus slowing down the processing speed.	Spark reduces the number of read/write cycles to disk and stores intermediate data in memory, hence faster-processing speed.
Usage	Hadoop is designed to handle batch processing efficiently.	Spark is designed to handle real-time data efficiently.
Latency	Hadoop is a high latency computing framework, which does not have an interactive mode.	Spark is a low latency computing and can process data interactively.
Data	With Hadoop MapReduce, a developer can only process data in batch mode only.	Spark can process real-time data, from real-time events like Twitter, and Facebook.
Cost	Hadoop is a cheaper option available while comparing it in terms of cost	Spark requires a lot of RAM to run in-memory, thus increasing the cluster and hence cost.
Algorithm Used	The PageRank algorithm is used in Hadoop.	Graph computation library called GraphX is used by Spark.
Fault Tolerance	Hadoop is a highly fault-tolerant system where Fault-tolerance achieved by replicating blocks of data. If a node goes down, the data can be found on another node	Fault-tolerance achieved by storing chain of transformations If data is lost, the chain of transformations can be recomputed on the original data
Security	Hadoop supports LDAP, ACLs, SLAs, etc and hence it is extremely secure.	Spark is not secure, it relies on the integration with Hadoop to achieve the necessary security level.
Machine Learning	Data fragments in Hadoop can be too large and can create bottlenecks. Thus, it is slower than Spark.	Spark is much faster as it uses MLib for computations and has in-memory processing.
Scalability	Hadoop is easily scalable by adding nodes and disk for storage. It supports tens of thousands of nodes.	It is quite difficult to scale as it relies on RAM for computations. It supports thousands of nodes in a cluster.
Language support	It uses Java or Python for MapReduce apps.	It uses Java, R, Scala, Python, or Spark SQL for the APIs.
User-friendliness	It is more difficult to use.	It is more user-friendly.
Resource Management	YARN is the most common option for resource management.	It has built-in tools for resource management.

APACHE ZOOKEEPER

- **Apache Zookeeper** is an open source distributed coordination service that helps to manage a large set of hosts. Management and coordination in a distributed environment is tricky. Zookeeper automates this process and allows developers to focus on building software features rather than worry about it’s distributed nature.

- Zookeeper helps you to maintain configuration information, naming, group services for distributed applications. It implements different protocols on the cluster so that the application should not implement on their own. It provides a single coherent view of multiple machines.

- Apache Zookeeper is the coordinator of any Hadoop job which includes a combination of various services in a Hadoop Ecosystem.
- Apache Zookeeper coordinates with various services in a distributed environment.

- Before Zookeeper, it was very difficult and time consuming to coordinate between different services in Hadoop Ecosystem. The services earlier had many problems with interactions like common configuration while synchronizing data. Even if the services are configured, changes in the configurations of the services make it complex and difficult to handle. The grouping and naming was also a time-consuming factor.

- Due to the above problems, Zookeeper was introduced. It saves a lot of time by performing synchronization, configuration maintenance, grouping and naming.

- Although it’s a simple service, it can be used to build powerful solutions.

Why Apache Zookeeper?

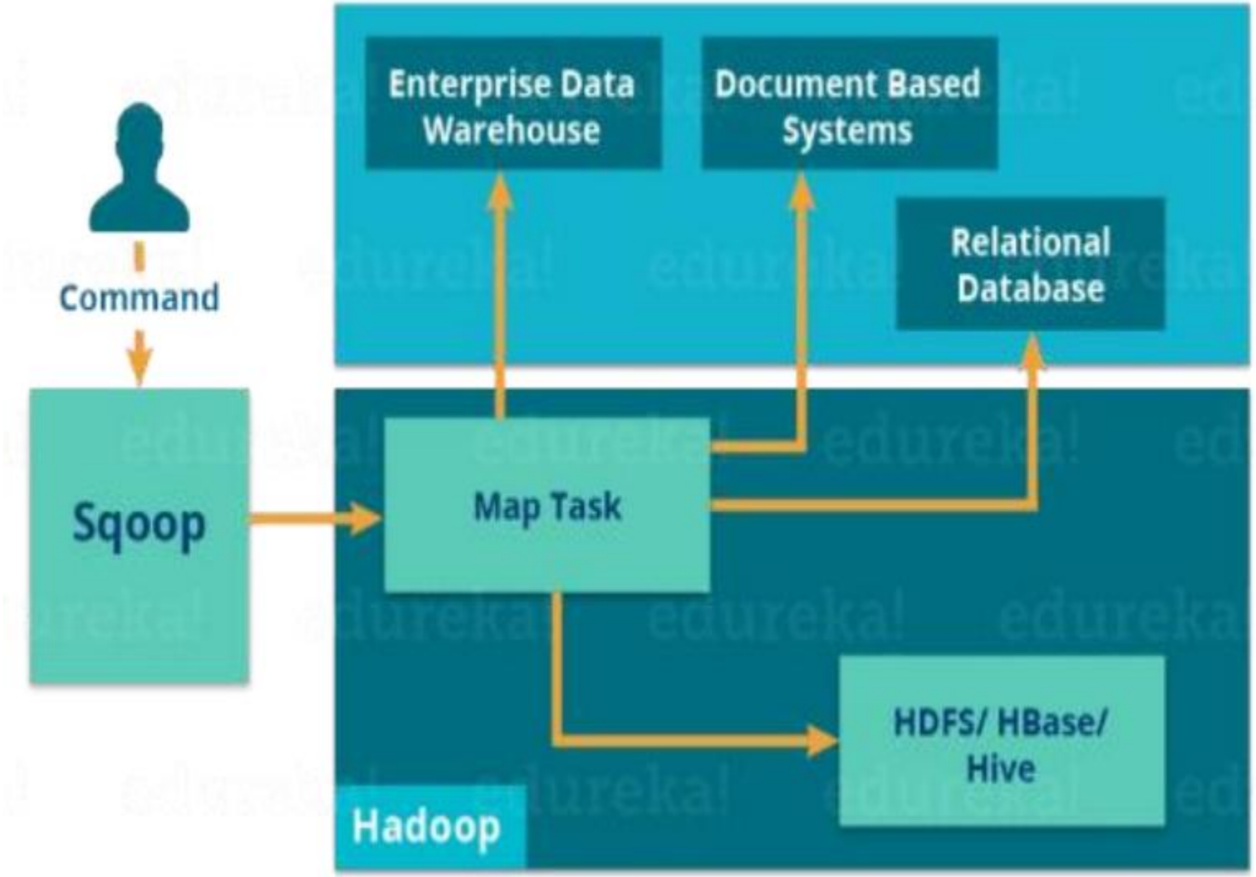
Here, are important reasons behind the popularity of the Zookeeper:

- It allows for mutual exclusion and cooperation between server processes.
- It ensures that your application runs consistently.
- The transaction process is never completed partially. It is either given the status of Success or failure. The distributed state can be held up, but it’s never wrong
- Irrespective of the server that it connects to a client will be able to see the same view of the service.
- Helps you to encode the data as per the specific set of rules
- It helps to maintain a standard hierarchical namespace similar to files and directories
- Computers, which run as a single system which can be locally or geographically connected
- It allows to Join/leave node in a cluster and node status at the real time
- You can increase performance by deploying more machines
- It allows you to elect a node as a leader for better coordination
- ZooKeeper works fast with workloads where reads to the data are more common than writes

APACHE SQOOP h → db

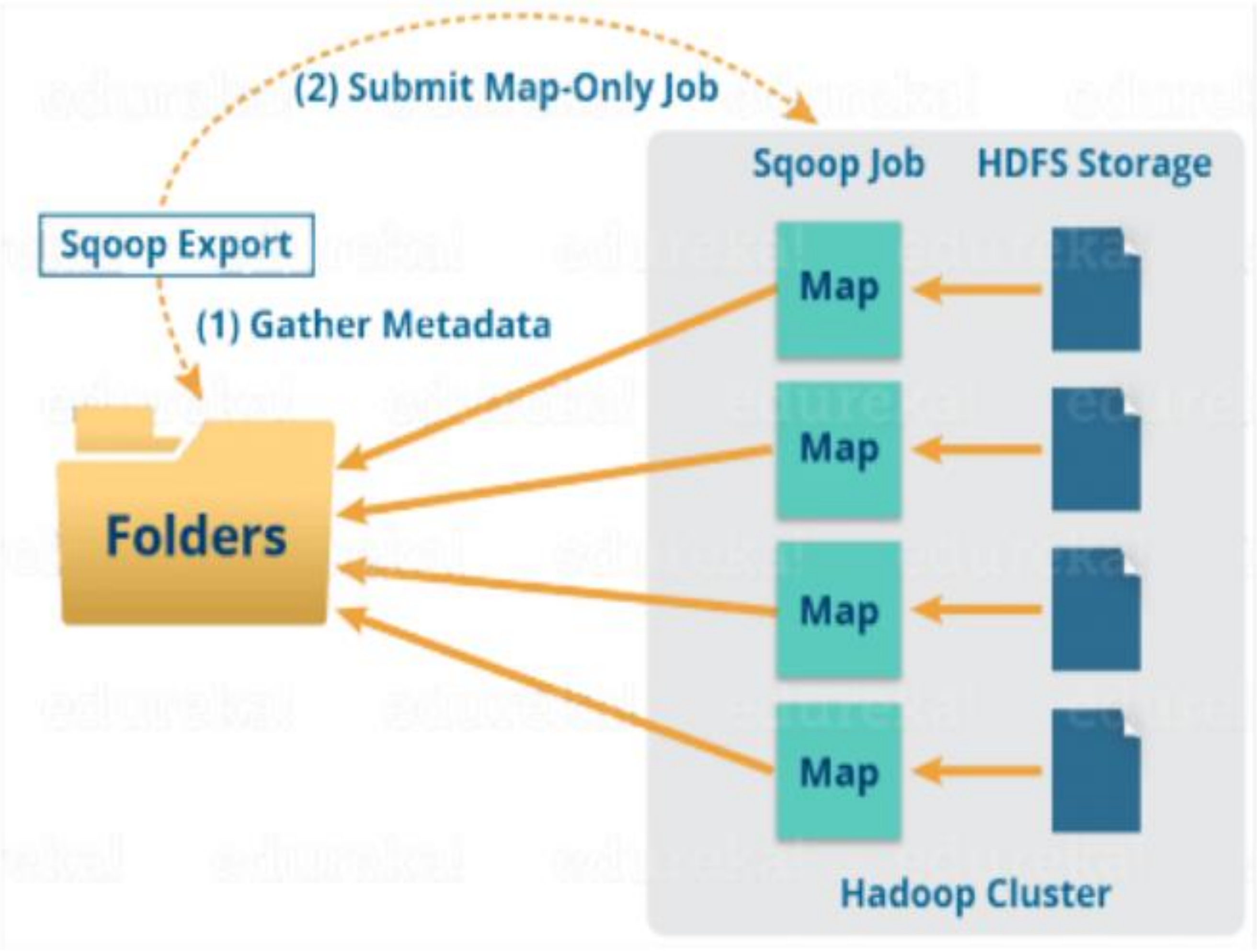
- Apache Sqoop(TM) is a tool designed for efficiently transferring bulk data between Apache Hadoop and structured datastores such as relational databases.

- While Sqoop can import as well as export structured data from RDBMS or Enterprise data warehouses to HDFS or vice versa.



- When we submit Sqoop command, our main task gets divided into sub tasks which is handled by individual Map Task internally. Map Task is the sub task, which imports part of data to the Hadoop Ecosystem. Collectively, all Map tasks imports the whole data. Export also works in a similar manner.

When we submit our Job, it is mapped into Map Tasks which brings the chunk of data from HDFS. These chunks are exported to a structured data destination. Combining all these exported chunks of data, we receive the whole data at the destination, which in most of the cases is an RDBMS (MYSQL/Oracle/SQL Server).



Introduction to Languages and Databases:

Apache PIG

- Pig is a scripting platform that runs on Hadoop clusters designed to process and analyze large datasets. Pig is extensible, self-optimizing, and easily programmed.
- Programmers can use Pig to write data transformations without knowing Java. Pig uses both structured and unstructured data as input to perform analytics and uses HDFS to store the results.

Components of Pig

There are two major components of the Pig:

1. Pig Latin script language and 2. A runtime engine

Pig Latin script language

- The Pig Latin script is a procedural data flow language. It contains syntax and commands that can be applied to implement business logic. Examples of Pig Latin are LOAD and STORE.

A runtime engine

- The runtime engine is a compiler that produces sequences of MapReduce programs. It uses HDFS to store and retrieve data. It is also used to interact with the Hadoop system (HDFS and MapReduce).
- The runtime engine parses, validates, and compiles the script operations into a sequence of MapReduce jobs. **10 line of pig latin = approx. 200 lines of Map-Reduce Java code**

How Pig works?

In PIG, first the load command, loads the data. Then we perform various functions on it like grouping, filtering, joining, sorting, etc. At last, either you can dump the data on the screen or you can store the result back in HDFS.

- The compiler internally converts pig latin to MapReduce. It produces a sequential set of MapReduce jobs, and that's an abstraction (which works like black box).
- PIG was initially developed by Yahoo.
- It gives you a platform for building data flow for ETL (Extract, Transform and Load), processing and analyzing huge data sets.

Features of Pig

https://data-flair.training/blogs/apache-pig-features/

Developers and analysts like to use Pig as it offers many features. Some of the features are as follows:

- Provision for step-by-step procedural control and the ability to operate directly over files
- Schemas that, though optional, can be assigned dynamically
- Support to User Defined Functions, and to various data types

How Pig Works and Stages of Pig Operations

Pig operations can be explained in the following three stages:

Stage 1: Load data and write Pig script

In this stage, data is loaded and Pig script is written.

```
A = LOAD ‘myfile’ AS (x, y, z);
B = FILTER A by x > 0;`
C = GROUP B BY x;
D = FOREACH A GENERATE x,COUNT(B);
STORE D INTO ‘output’
```

i. Rich set of operators
One of the major advantages is, in order to perform several operations, there is a huge set of operators offered by Apache Pig, such as join, sort, filer, etc.

ii. Ease of programming
Basically, for SQL Programmer, Pig Latin is a boon. It is as similar to SQL. Hence, if you are good at SQL it is easy to write a Pig script.

iii. Optimization opportunities
Also, it’s a benefit working here because in Apache Pig the tasks optimize their execution automatically. Hence, as a result, programmers only need to focus on the semantics of the language.

iv. Extensibility
Extensibility is one of the most interesting features it has. It means users can develop their own functions to read, process, and write data, using the existing operators.

Optional Schema
However, the schema is optional, in Apache Pig. Hence, without designing a schema we can store data. So, values are stored as $01, $02 etc.

Stage 2: Pig Operations

In the second stage, the Pig execution engine Parses and checks the script. If it passes the script optimized and a logical and physical plan is generated for execution.
The job is submitted to Hadoop as a job defined as a MapReduce Task. Pig Monitors the status of the job using Hadoop API and reports to the client.

Stage 3: Execution of the plan

In the final stage, results are dumped on the section or stored in HDFS depending on the user command

APACHE HIVE:

- Apache Hive is a data ware house system for Hadoop that runs SQL like queries called HQL (Hive query language) which gets internally converted to map reduce jobs. Hive was developed by Facebook. It supports Data definition Language, Data Manipulation Language and user defined functions.
- Basically, HIVE is a data warehousing component which performs reading, writing and managing large data sets in a distributed environment using SQL-like interface.

HIVE + SQL = HQL

- The query language of Hive is called Hive Query Language(HQL), which is very similar like SQL.
- It has 2 basic components: Hive Command Line and JDBC/ODBC driver.
- The Hive Command line interface is used to execute HQL commands.
- While, Java Database Connectivity (JDBC) and Object Database Connectivity (ODBC) is used to establish connection from data storage.

- Secondly, Hive is highly scalable. As, it can serve both the purposes, i.e. large data set processing (i.e. Batch query processing) and real time processing (i.e. Interactive query processing).
- It supports all primitive data types of SQL.
- You can use predefined functions, or write tailored user defined functions (UDF) also to accomplish your specific needs.

Features of Hive

- Hive is fast and scalable ✓
- It provides SQL-like queries (i.e., HQL) that are implicitly transformed to MapReduce or Spark jobs. ✓
- It is capable of analyzing large datasets stored in HDFS. ✓
- It allows different storage types such as plain text, RCFile, and HBase.
- It uses indexing to accelerate queries.
- It can operate on compressed data stored in the Hadoop ecosystem.
- It supports user-defined functions (UDFs) where user can provide its functionality.

Limitations of Hive

- Hive is not capable of handling real-time data.
- It is not designed for online transaction processing.
- Hive queries contain high latency.

Support ✓ real-tym
handle ✗ 〃 〃

Differences between Hive and Pig

Hive	Pig
• Hive is commonly used by Data Analysts.	• Pig is commonly used by programmers.
• It follows SQL-like queries.	• It follows the data-flow language.
• It can handle structured data.	• It can handle semi-structured data.
• It works on server-side of HDFS cluster.	• It works on client-side of HDFS cluster.
• Hive is slower than Pig.	• Pig is comparatively faster than Hive.

NoSQL Databases

NoSQL databases (aka "not only SQL") are non-tabular databases and store data differently than relational tables. NoSQL databases come in a variety of types based on their data model.

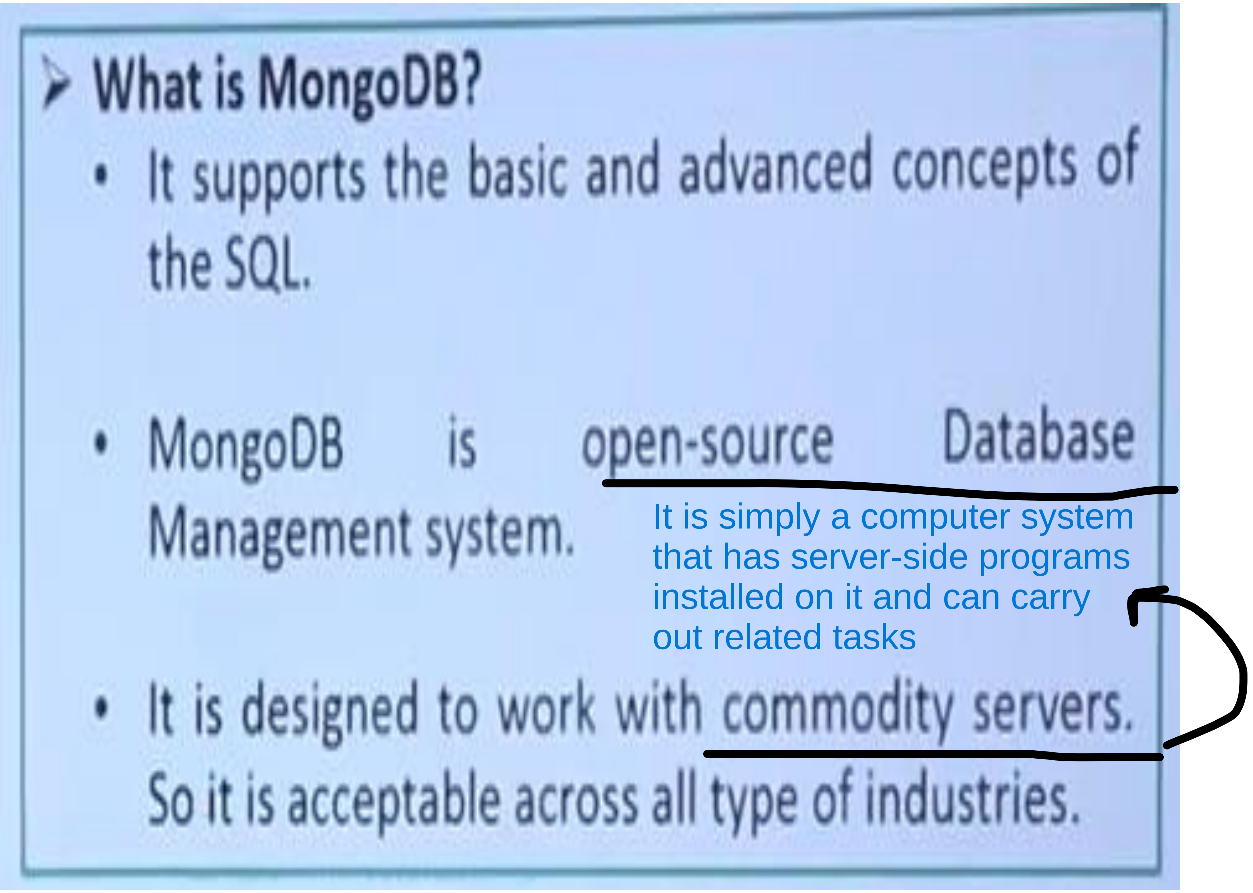
Types of NoSQL databases

Over time, four major types of NoSQL databases emerged: document databases, key-value databases, wide-column stores, and graph databases.

- **Document databases** store data in documents similar to JSON (JavaScript Object Notation) objects. Each document contains pairs of fields and values. The values can typically be a variety of types including things like strings, numbers, booleans, arrays, or objects. **Ex: MongoDB**
- **Key-value databases** are a simpler type of database where each item contains keys and values. **Ex: DynamoDB**
- **Wide-column stores** store data in tables, rows, and dynamic columns. **Ex :Apache Cassandra,**
- **Graph databases** store data in nodes and edges. Nodes typically store information about people, places, and things, while edges store information about the relationships between the nodes. **Ex: Neo4J**

SQL	NoSQL
Relational Database Management System (RDBMS)	Non-relational or distributed database system.
These databases have fixed or static or predefined schema	They have dynamic schema
These databases are not suited for hierarchical data storage.	These databases are best suited for hierarchical data storage.
These databases are best suited for complex queries	These databases are not so good for complex queries
Vertically Scalable	Horizontally scalable
Follows ACID property A---Atomicity C---Consistency I---isolation D----Durability	Follows CAP(consistency, availability, partition tolerance)
Examples: MySQL, PostgreSQL, Oracle, MS-SQL Server, etc	**Examples:** MongoDB, GraphQL, HBase, Neo4j , Cassandra, etc

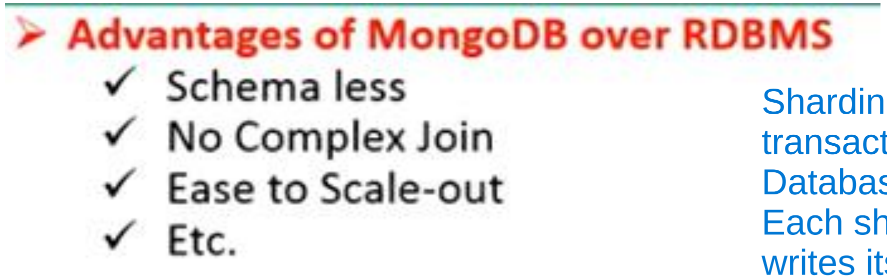
MongoDB, the most popular NoSQL database, is an open-source document-oriented database. It means that MongoDB isn't based on the table-like relational database structure but provides an altogether different mechanism for storage and retrieval of data. This format of storage is called BSON(Binary JavaScript Object Notation) (similar to JSON(JavaScript Object Notation) format).



➢ What is MongoDB?

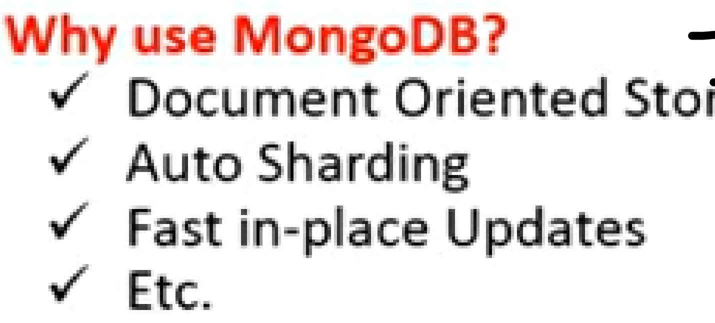
- It supports the basic and advanced concepts of the SQL.
- MongoDB is open-source Database Management system.
- It is designed to work with commodity servers. So it is acceptable across all type of industries.

It is simply a computer system that has server-side programs installed on it and can carry out related tasks



➢ Advantages of MongoDB over RDBMS

✓ Schema less
✓ No Complex Join
✓ Ease to Scale-out
✓ Etc.

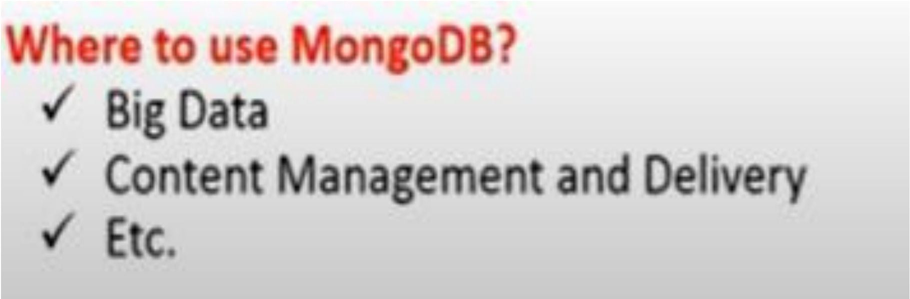


Why use MongoDB?

✓ Document Oriented Storage
✓ Auto Sharding
✓ Fast in-place Updates
✓ Etc.

Sharding reduces the transaction cost of the Database;
Each shard reads and writes its own data
Sharding means dividing a larger part into smaller parts.

These shards are not only smaller, but also faster and hence easily manageable.



Where to use MongoDB?

✓ Big Data
✓ Content Management and Delivery
✓ Etc.

Features of MongoDB:

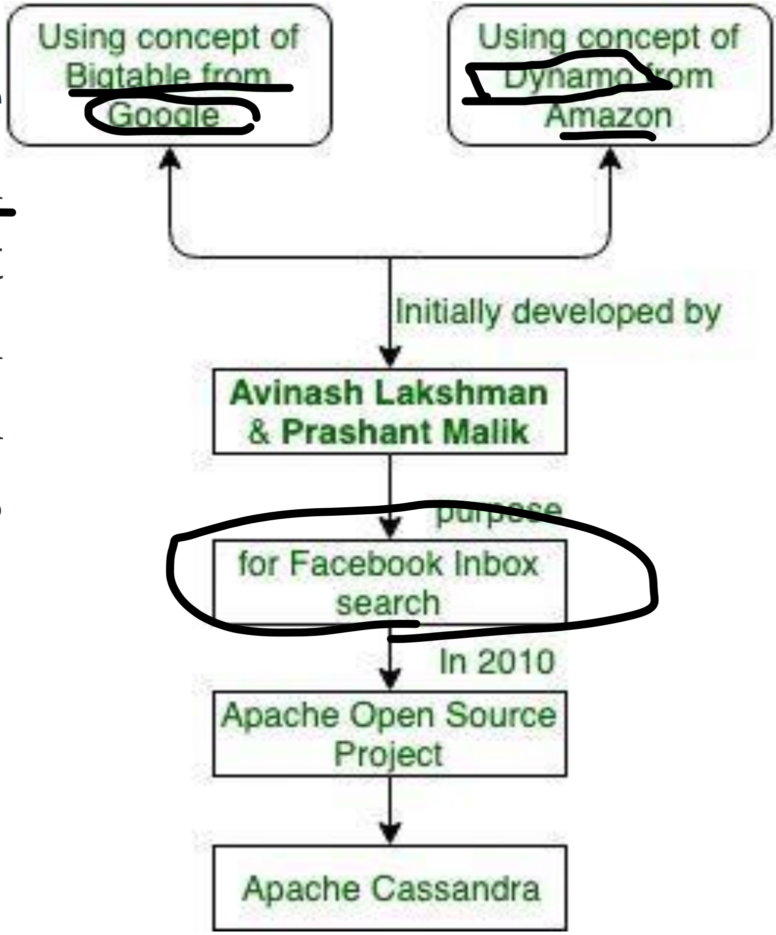
- **Document Oriented:** MongoDB stores the main subject in the minimal number of documents and not by breaking it up into multiple relational structures like RDBMS. For example, it stores all the information of a computer in a single document called Computer and not in distinct relational structures like CPU, RAM, Hard disk, etc.
- **Indexing:** Without indexing, a database would have to scan every document of a collection to select those that match the query which would be inefficient. So, for efficient searching Indexing is a must and MongoDB uses it to process huge volumes of data in very less time.
- **Scalability**: MongoDB scales horizontally using sharing (partitioning data across various servers). Data is partitioned into data chunks using the shard key, and these data chunks are evenly distributed across shards that reside across many physical servers. Also, new machines can be added to a running database.

- **Replication and High Availability**: MongoDB increases the data availability with multiple copies of data on different servers. By providing redundancy, it protects the database from hardware failures. If one server goes down, the data can be retrieved easily from other active servers which also had the data stored on them.
- **Aggregation**: Aggregation operations process data records and return the computed results. It is similar to the GROUPBY clause in SQL. A few aggregation expressions are sum, avg, min, max, etc

Introduction to Apache Cassandra

Cassandra is a distributed database management system which is open source with wide column store, NoSQL database to handle large amount of data across many commodity servers which provides high availability with no single point of failure. It is written in Java and developed by Apache Software Foundation

Avinash Lakshman & **Prashant Malik** initially developed the Cassandra at Facebook to power the Facebook inbox search feature. Facebook released Cassandra as an open source project on Google code in July 2008. In March 2009 it became an Apache Incubator project and in February 2010 it becomes a top-level project. Due to its outstanding technical features Cassandra becomes so popular



Using concept of Bigtable from Google

Using concept of Dynamo from Amazon

Initially developed by

Avinash Lakshman & Prashant Malik

purpose

for Facebook Inbox search

In 2010

Apache Open Source Project

Apache Cassandra

Apache Cassandra is used to manage very large amounts of structure data spread out across the world. It provides highly available service with no single point of failure.

- It is scalable, fault-tolerant, and consistent.
- It is column-oriented database.
- Its distributed design is based on Amazon’s Dynamo and its data model on Google’s Big table.
- It is Created at Facebook and it differs sharply from relational database management systems.

- Cassandra implements a Dynamo-style replication model with no single point of failure but its add a more powerful “column family” data model. Cassandra is being used by some of the biggest companies such as Facebook, Twitter, Cisco, Rackspace, eBay, Netflix, and more.
- Cassandra has peer-to-peer distributed system across its nodes, and data is distributed among all the nodes of the cluster.
- All the nodes of Cassandra in a cluster play the same role. Each node is independent, at the same time interconnected to other nodes. Each node in a cluster can accept read and write requests, regardless of where the data is actually located in the cluster. When a node goes down, read/write request can be served from other nodes in the network.

Features of Cassandra:

Cassandra has become popular because of its technical features. There are some of the features of Cassandra:

1.Easy data distribution –

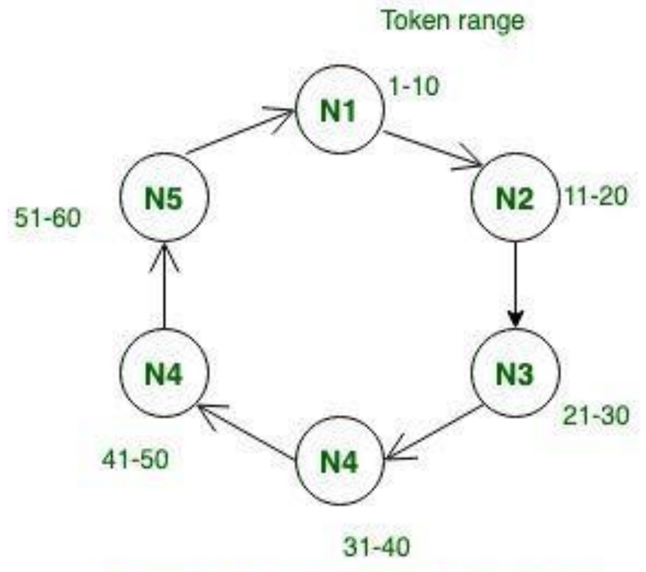
It provides the flexibility to distribute data where you need by replicating data across multiple data centers.

for example:

If there are 5 node let say N1, N2, N3, N4, N5 and by using partitioning algorithm we will decide the token range and distribute data accordingly. Each node have specific token range in which data will be distribute. let's have a look on diagram for better understanding.

2. Flexible data storage

Cassandra accommodates all possible data formats including: structured, semi-structured, and unstructured. It can dynamically accommodate changes to your data structures accordingly to your need.



Example : Ring structure with token range.

3. Elastic scalability –

Cassandra is highly scalable and allows to add more hardware to accommodate more customers and more data as per requirement.

4. Fast writes

Cassandra was designed to run on cheap commodity hardware. Cassandra performs blazingly fast writes and can store hundreds of terabytes of data, without sacrificing the read efficiency.

5. Always on Architecture

Cassandra has no single point of failure and it is continuously available for business-critical applications that can’t afford a failure.

6.Fast linear-scale performance

Cassandra is linearly scalable therefore it increases your throughput as you increase the number of nodes in the cluster. It maintains a quick response time.

7.**Transaction support**

Cassandra supports properties like Atomicity, Consistency, Isolation, and Durability (ACID) properties of transactions.

CouchDB

CouchDB is an open-source NoSQL database that focuses on ease of use. It is developed by Apache. It is fully compatible with the web. CouchDB uses JSON to store data, JavaScript as its query language to transform the documents, using MapReduce, and HTTP for an API.

Why CouchDB

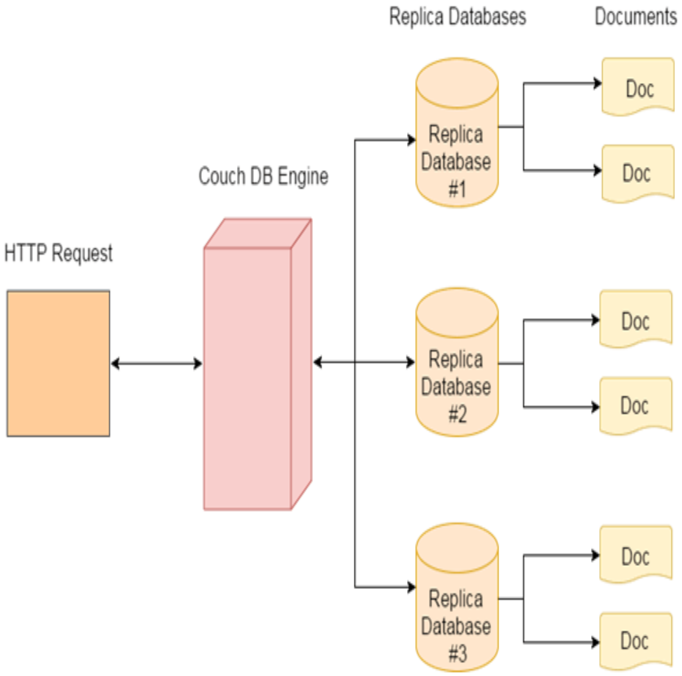
Apache CouchDB is one of the latest breeds of databases. CouchDB has a schema-free document model which is a better fit for common applications. It is very powerful to query your data.

What is the main reason behind using CouchDB?

- CouchDB is easy to use. There is one word to describe CouchDB "Relax". It is also the byline of the CouchDB official logo.
- CouchDB has an HTTP-based REST API, which makes communication with the database very easy.
- CouchDB has a simple structure of HTTP resources and methods (GET, PUT, DELETE) that are easy to understand and use.
- In CouchDB, data is stored in a flexible document-based structure so, there is no need to worry about the structure of the data.
- CouchDB facilitates users with powerful data mapping, which allows querying, combining, and filtering the information.
- CouchDB provides easy-to-use replication, using which you can copy, share, and synchronize the data between databases and machines.

CouchDB Data Model

- Database is the outermost data structure/container in CouchDB.
- Each database is a collection of independent documents.
- Each document is responsible for maintaining its own data and self-contained schema.
- Document metadata contains revision information, which makes it possible to merge the differences occurred while the databases were disconnected.
- CouchDB implements multi version concurrency control, to avoid the need to lock the database field during writes.



Features of CouchDB

Following is a list of most attractive features of CouchDB:

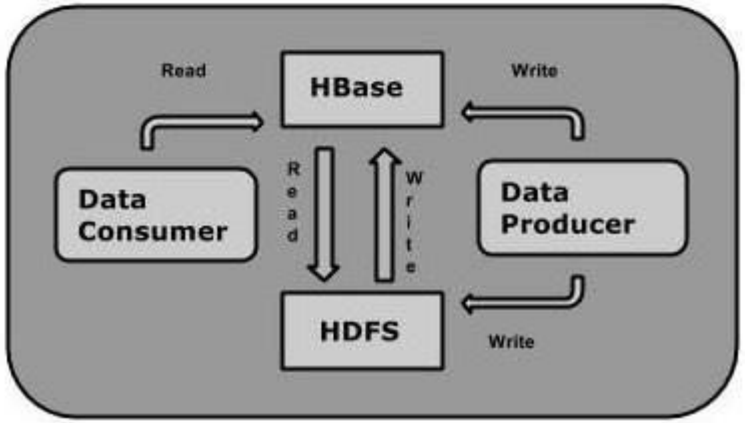
- **Document Storage:** CouchDB is a NoSQL database which follows document storage. Documents are the primary unit of data where each field is uniquely named and contains values of various data types such as text, number, Boolean, lists, etc.Documents don't have a set limit to text size or element count.
- **Browser Based GUI:** CouchDB provides an interface Futon which facilitates a browser based GUI to handle your data, permission and configuration.

- **Replication:** CouchDB provides the simplest form of replication. There is no other database is so simple to replicate.
- **ACID Properties:** The CouchDB file layout follows all the features of ACID properties. Once the data is entered in to the disc, it will not be overwritten. Document updates (add, edit, delete) follow Atomicity, i.e., they will be saved completely or not saved at all. The database will not have any partially saved or edited documents. Almost all of these update are serialized and any number of clients can read a document without waiting and without being interrupted.
- **JSONP for Free:** If you update your config to allow_jsonp = true then your database is accessible cross domain for GET requests.
- **Authentication and Session Support:** CouchDB facilitates you to keep authentication open via a session cookie like web application.
- **Security:** CouchDB also provides database-level security. The permissions per database are separated into readers and admins. Readers can both read and write to the database.
- **Validation:** You can validate the inserted data into the database by combining with authentication to ensure the creator of the document is the one who is logged in.
- **Map/Reduce List and Show:** The main reason behind the popularity of MongoDB and CouchDB is map/reduce system.

APACHE HBASE

What is HBase?

- HBase is a distributed column-oriented database built on top of the Hadoop file system. It is an open-source project and is horizontally scalable.
- HBase is a data model that is similar to Google’s big table designed to provide quick random access to huge amounts of structured data. It leverages the fault tolerance provided by the Hadoop File System (HDFS).
- It is a part of the Hadoop ecosystem that provides random real-time read/write access to data in the Hadoop File System.
- One can store the data in HDFS either directly or through HBase. Data consumer reads/accesses the data in HDFS randomly using HBase. HBase sits on top of the Hadoop File System and provides read and write access.



HDFS	HBase
HDFS is a distributed file system suitable for storing large files.	HBase is a database built on top of the HDFS.
HDFS does not support fast individual record lookups.	HBase provides fast lookups for larger tables.
It provides high latency batch processing;	It provides low latency access to single rows from billions of records (Random access).
It provides only sequential access of data.	HBase internally uses Hash tables and provides random access, and it stores the data in indexed HDFS files for faster lookups.

Storage Mechanism in HBase

HBase is a **column-oriented database** and the tables in it are sorted by row. The table schema defines only column families, which are the key value pairs. A table have multiple column families and each column family can have any number of columns. Subsequent column values are stored contiguously on the disk. Each cell value of the table has a timestamp. In short, in an HBase:

- Table is a collection of rows.
- Row is a collection of column families.
- Column family is a collection of columns.
- Column is a collection of key value pairs.

Features of HBase

- HBase is linearly scalable.
- It has automatic failure support.
- It provides consistent read and writes.
- It integrates with Hadoop, both as a source and a destination.
- It has easy java API for client.
- It provides data replication across clusters.

Where to Use HBase

- Apache HBase is used to have random, real-time read/write access to Big Data.
- It hosts very large tables on top of clusters of commodity hardware.
- Apache HBase is a non-relational database modeled after Google's Bigtable. Bigtable acts up on Google File System, likewise Apache HBase works on top of Hadoop and HDFS.

Applications of HBase

- It is used whenever there is a need to write heavy applications.
- HBase is used whenever we need to provide fast random access to available data.
- Companies such as Facebook, Twitter, Yahoo, and Adobe use HBase internally.

Cloudera Distribution Hadoop(CDH) Overview

CDH is the most complete, tested, and popular distribution of Apache Hadoop and related projects. CDH delivers the core elements of Hadoop – scalable storage and distributed computing – along with a Web-based user interface and vital enterprise capabilities. CDH is Apache-licensed open source and isthe only Hadoop solution to offer unified batch processing, interactive SQL and interactive search, and role-based access controls.

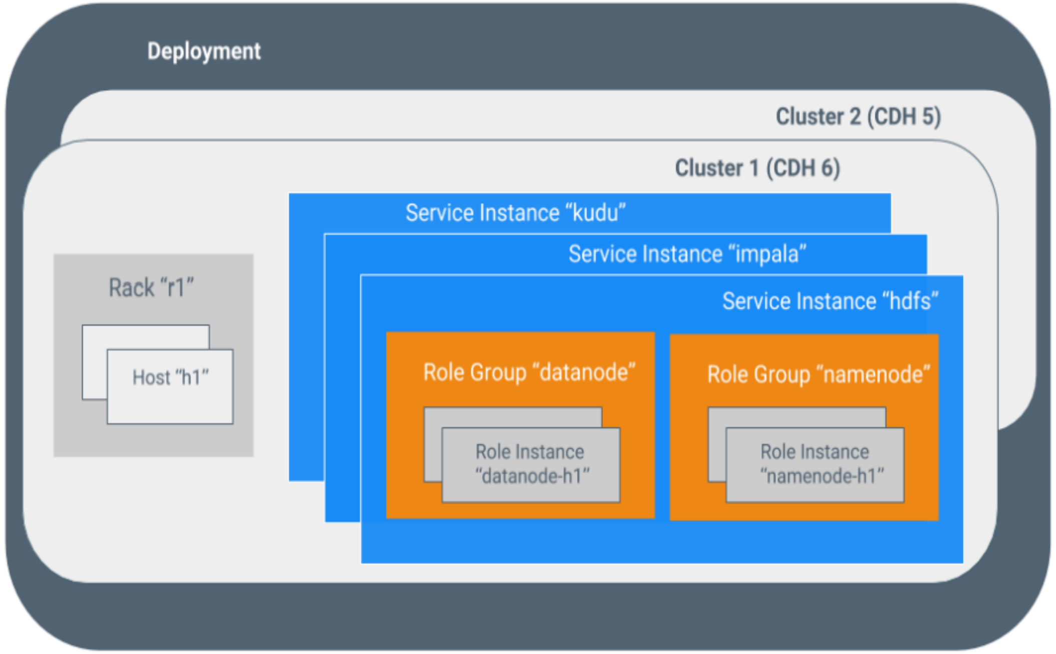
CDH provides:

- **Flexibility**—Store any type of data and manipulate it with a variety of different computation frameworksincluding batch processing, interactive SQL, free text search, machine learning and statistical computation.
- **Integration**—Get up and running quickly on a complete Hadoop platform that works with a broad range of hardware and software solutions.
- **Security**—Process and control sensitive data.
- **Scalability**—Enable a broad range of applications and scale and extend them to suit your requirements.
- **High availability**—Perform mission-critical business tasks with confidence.
- **Compatibility**—Leverage your existing IT infrastructure and investment.

Cloudera provides documentation for CDH as a whole, whether your CDH cluster is managed by Cloudera Manager or not. In addition, you may find it useful to refer to documentation for the individual components included in CDH. Where possible, these links point to the main documentation for a project, in the Cloudera release archive. This ensures that you are looking at the correct documentation for the version of a project included in CDH.

- **Apache Avro**
- **Apache Flume**
- **Apache HBase**
- **Hue**
- **Apache Oozie**
- **Apache Pig**
- **Apache Solr**
- **Apache Sqoop**
- **Apache Crunch**
- **Apache Hadoop**
- **Apache Hive**
- **Kite**
- **Apache Parquet**
- **Apache Sentry**
- **Apache Spark**
- **Apache ZooKeeper**

Cloudera Manager is an end-to-end application for managing CDH clusters. Cloudera Manager sets the standard for enterprise deployment by delivering granular visibility into and control over every part of the CDH cluster—empowering operators to improve performance, enhance quality of service, increase compliance and reduce administrative costs. With Cloudera Manager, you can easily deploy and centrally operate the complete CDH stack and other managed services. The application automates the installation process, reducing deployment time from weeks to minutes; gives you a cluster-wide,real-time view of hosts and servicesrunning; provides a single, central console to enact configuration changes across your cluster; and incorporates a full range of reporting and diagnostic tools to help you optimize performance and utilization. This primer introduces the basic concepts, structure, and functions of Cloudera Manager



1. What Hadoop Eco System ? Explain Difference Between Hadoop and Hadoop Eco System?
2. Briefly Explain Apache Spark , Advantages and difference between Hadoop and Spark?
3. Explain Apache Zookeeper and Sqoop in detail?
4. What are scripting Languages supported by Hadoop/spark cluster? Explain indetail.
5. What is NoSql Database? Explain Difference between SQL and NoSql?
6. Explain Mango DB, Cassandra , CouchDB and Hbase in detail?
7. Briefly explain Cloudera Distributed Hadoop(CDH)?