HBase

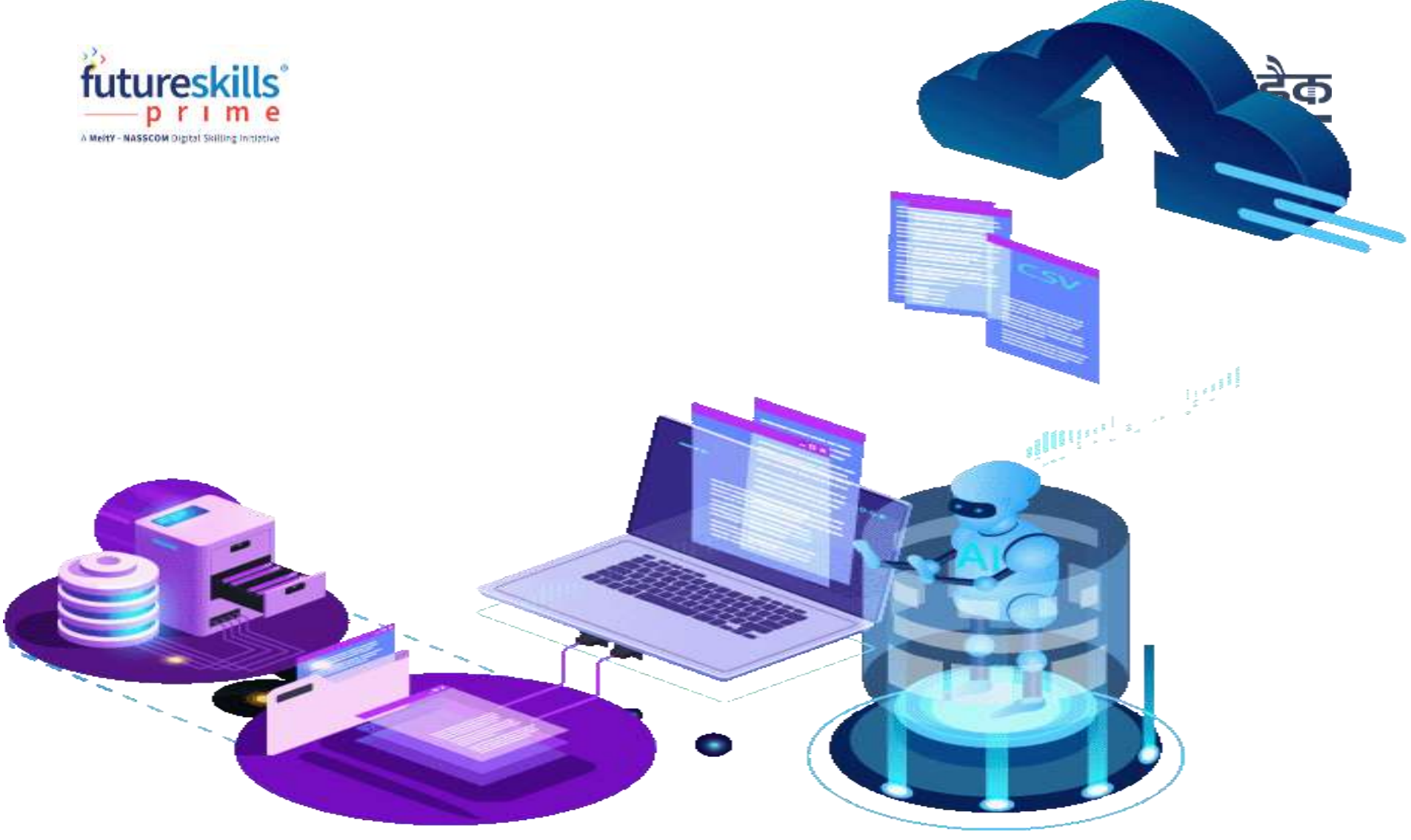
futureskills prime
A MeitY - NASSCOM Digital Skilling Initiative

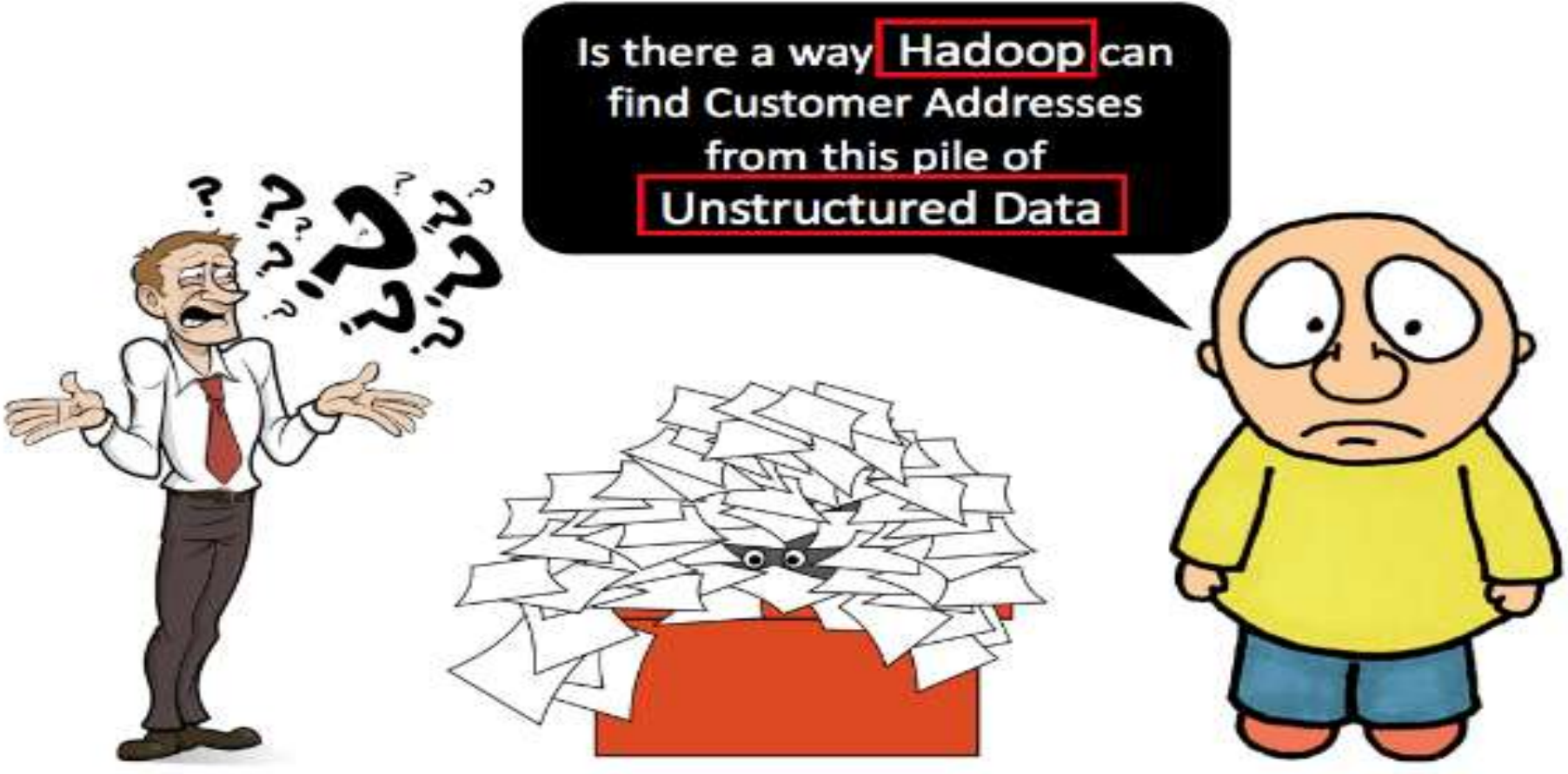
Topics at a glance

- Hadoop and its data access limitations.
- Why Hbase?
- Hbase and its importance in Hadoop frame work.
- History and Architecture of Hbase.
- Hbase components and their responsibilities.
- Hbase data storage model.
- Advantages and disadvantages of Hbase.
- Conclusion of the session

futureskills prime

A MeitY - NASSCOM Digital Skilling Initiative




Is there a way Hadoop can find Customer Addresses from this pile of Unstructured Data

Why Hbase???

With the evolution of the internet web application scope was increased -

- Huge volumes of structured and semi-structured data started getting generated.
- Semi-structured data (emails, JSON, XML, and .csv files and exe files)
- Loads of semi-structured data was created across the globe.
- So storing and processing of this data became a major challenge.

Hadoop and its limitations

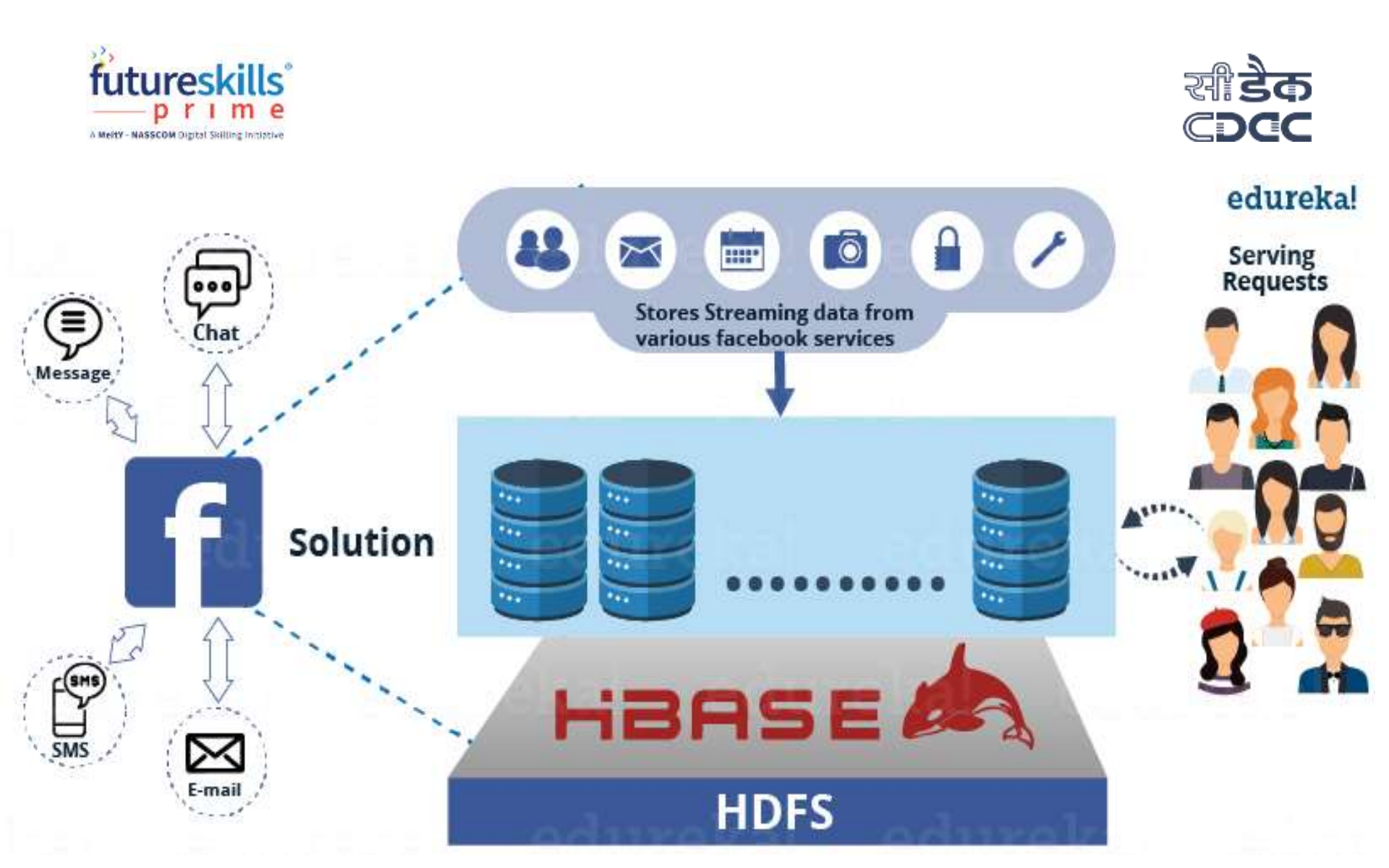
- Hadoop can perform only batch processing, and data will be accessed only in a sequential manner.
- So if needed to access any data randomly then need a new access methodology.

✓New program tool added in Hadoop framework to provide the random access to user.

- HBase
- Cassandra,
- CouchDB,
- Dynamo and MongoDB

HBase

- HBase is an open-source NoSQL database and Part of the Hadoop framework.
- Similar to Google's big table. Initially, it was Google Big Table, afterward; it was renamed as HBase .
- Hbase is primarily written in **Java** and needed for **real-time Big Data** applications.
- HBase is a distributed column-oriented **non-relational** database management system that runs on top of Hadoop Distributed File System (HDFS).
- 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.
- It uses log storage with the Write-Ahead Logs (WAL).
- It supports fast random access and heavy writing competency.


futureskills prime
A MeitY - NASSCOM Digital Skilling Initiative
सी-डैक CDAC
edureka!
Serving Requests
Stores Streaming data from various facebook services
Chat
Message
Solution
SMS
SMS
E-mail
HBASE
HDFS

How is HBase different from other NoSQL models

- HBase stores data in the form of key/value pairs in a columnar model. In this model, all the columns are grouped together as Column families.
- HBase on top of Hadoop will increase the throughput and performance of distributed cluster set up.
- Provides faster random reads and writes operations.

HBase	RDBMS
• HBase works well with structured and semi-structured data	• RDBMS has a fixed schema which describes the structure of the tables
• Hbase does not have a fixed schema. Here, only column families are defined	• Works well with structured data
• It can have denormalized data	• RDBMS can store only normalized data
• It is built for wide tables that can be scaled horizontally	• It is built for thin tables that is hard to scale

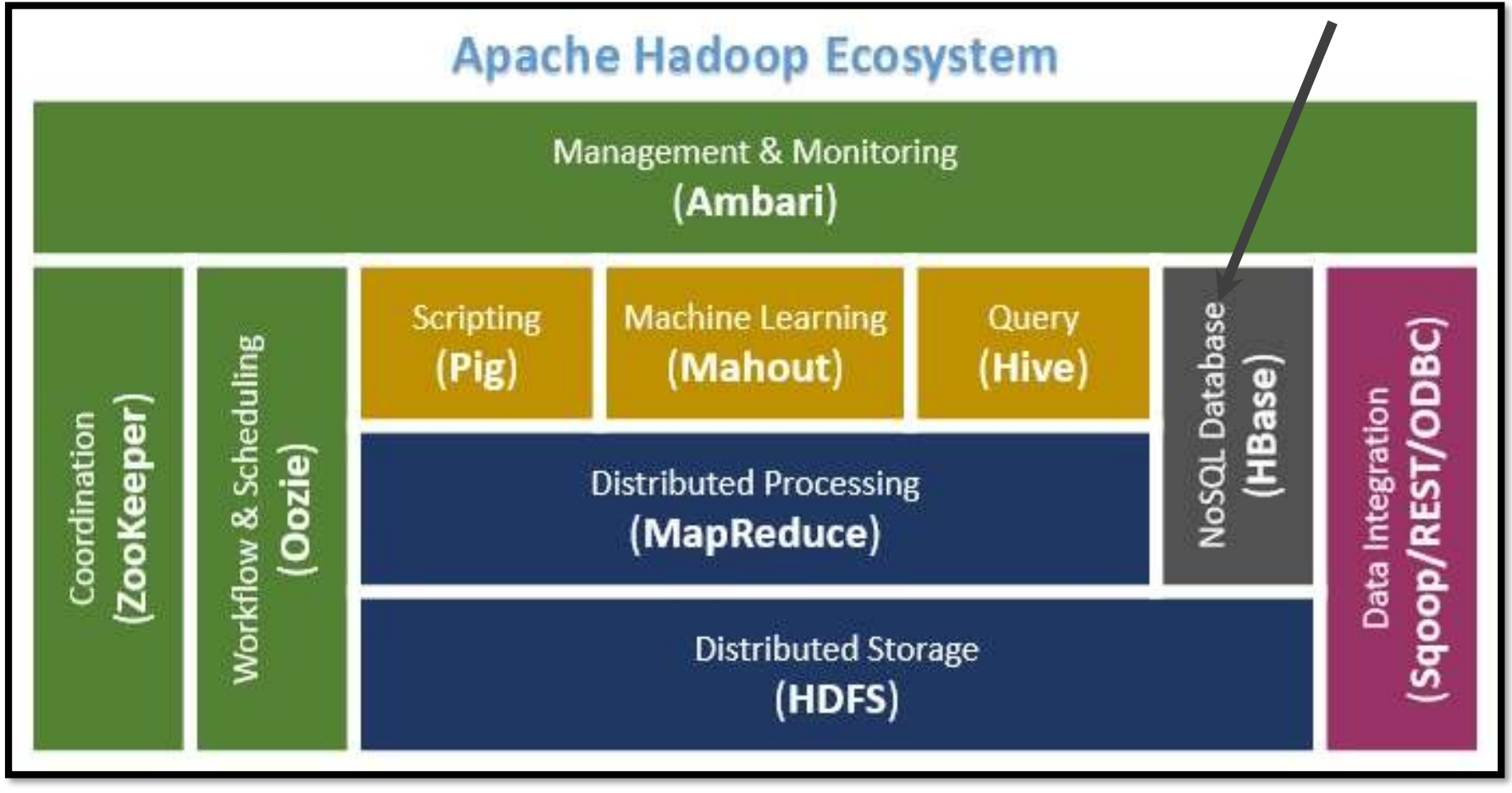
Features of Hbase

- **Horizontally scalable:** Can add any number of columns anytime.
- **Automatic Failover:** Allows a system administrator to automatically switch data handling to a standby system in the event of system compromise/failure.
- **Integrations with Map/Reduce framework:** Al the commands and java codes internally implement Map/ Reduce to do the task and it is built over Hadoop Distributed File System.
- It doesn't enforce relationships within your data.
- It is designed to run on a cluster of computers, built using commodity hardware.
- HBase is built for low latency operations

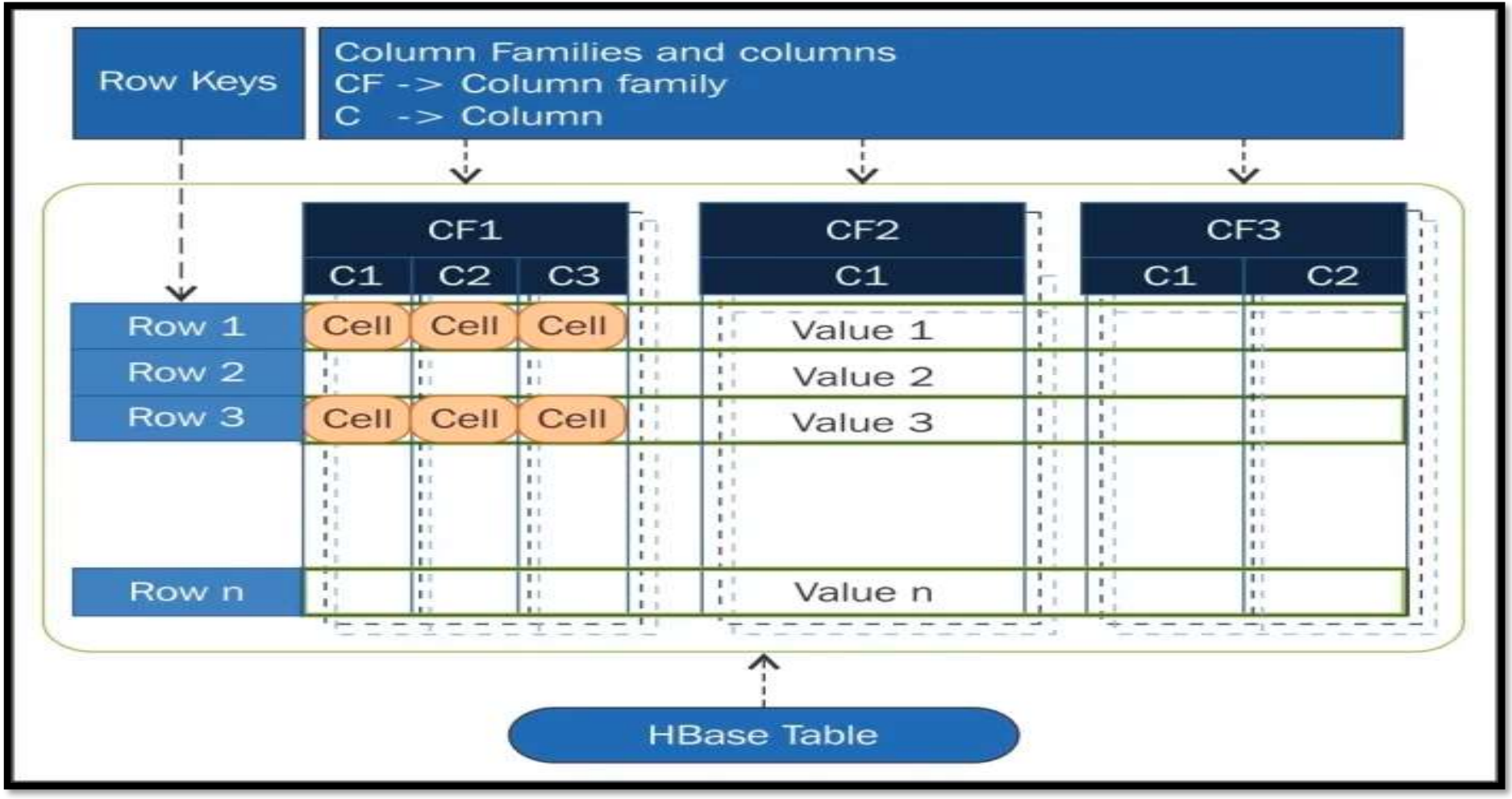
History of HBase

- In Nov 2006, Google released the paper on BigTable.
- Feb 2007, Initial HBase prototype was created as a Hadoop contribution.
- Oct 2007, The first usable HBase along with Hadoop 0.15.0 was released.
- Jan 2008, HBase became the sub project of Hadoop.
- Oct 2008, HBase 0.18.1 was released.
- Jan 2009, HBase 0.19.0 was released.
- Sept 2009, HBase 0.20.0 was released.
- May 2010, HBase became Apache top-level project.

HBase existence in the Hadoop Ecosystem



HBase Table – To store data



HBase: Keys and Column Families

Each record is divided into ***Column Families***

Each row has a ***Key***

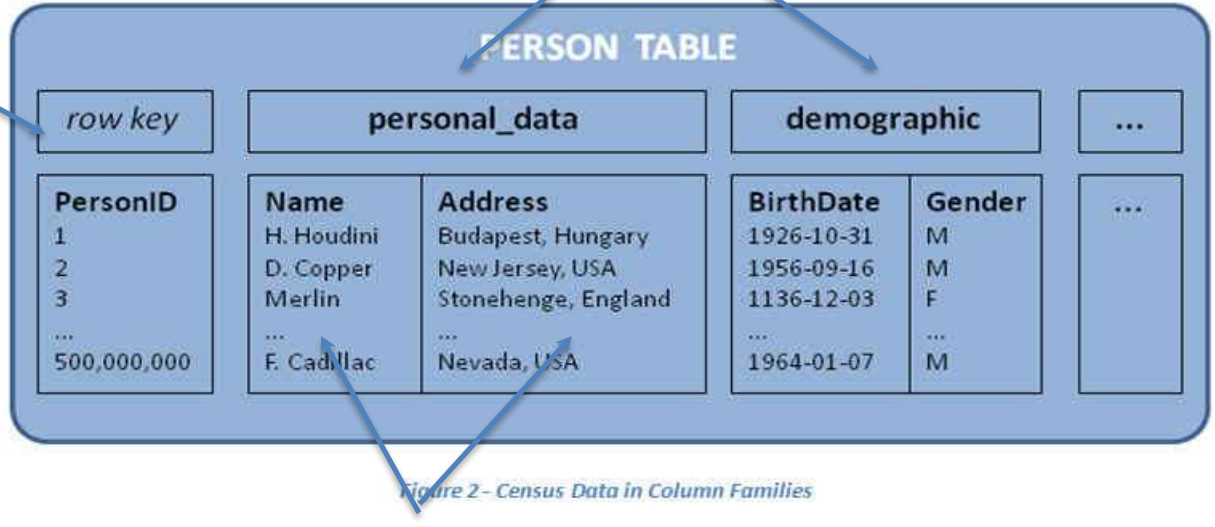


Figure 2 - Census Data in Column Families

Each column family consists of one or more ***Columns***

Example- Storage Mechanism in HBase

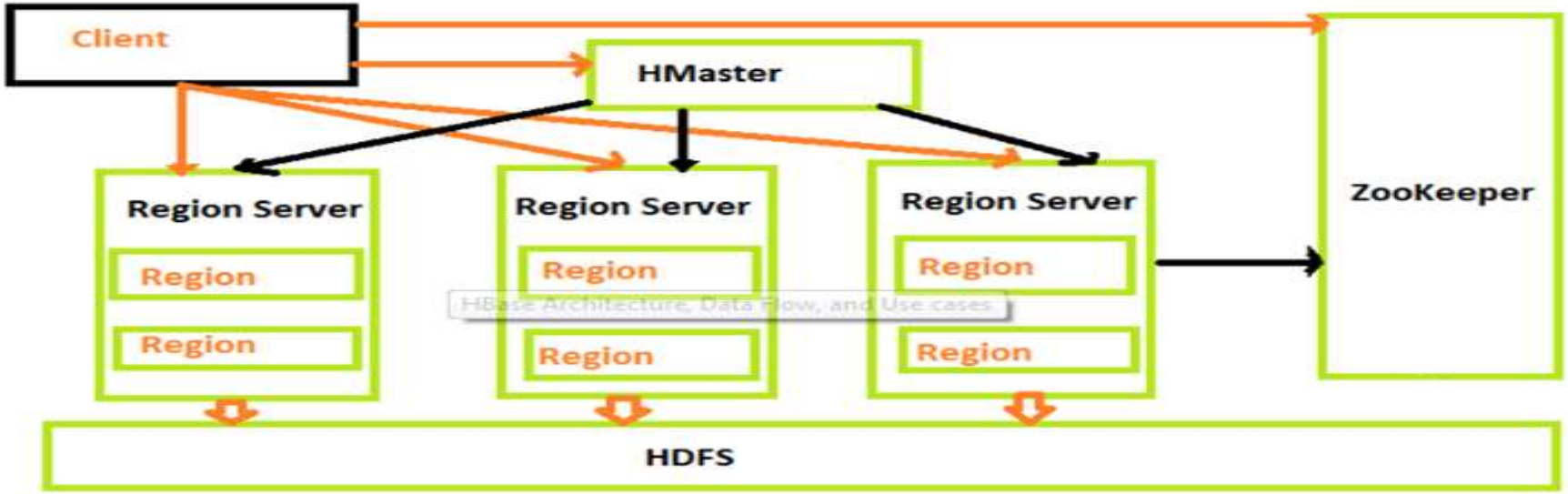
- HBase is a column-oriented database.
- Data is store in form of table.

Row Key | Column Family

Row Key	Customers		Products		
Customer ID	Customer Name	City & Country	Product Name	Price	Column Qualifiers
1	Sam Smith	California, US	Mike	$500	Cell
2	Arijit Singh	Goa, India	Speakers	$1000	
3	Ellie Goulding	London, UK	Headphones	$800	
4	Wiz Khalifa	North Dakota, US	Guitar	$2500	

Figure: HBase Table

HBase Architecture

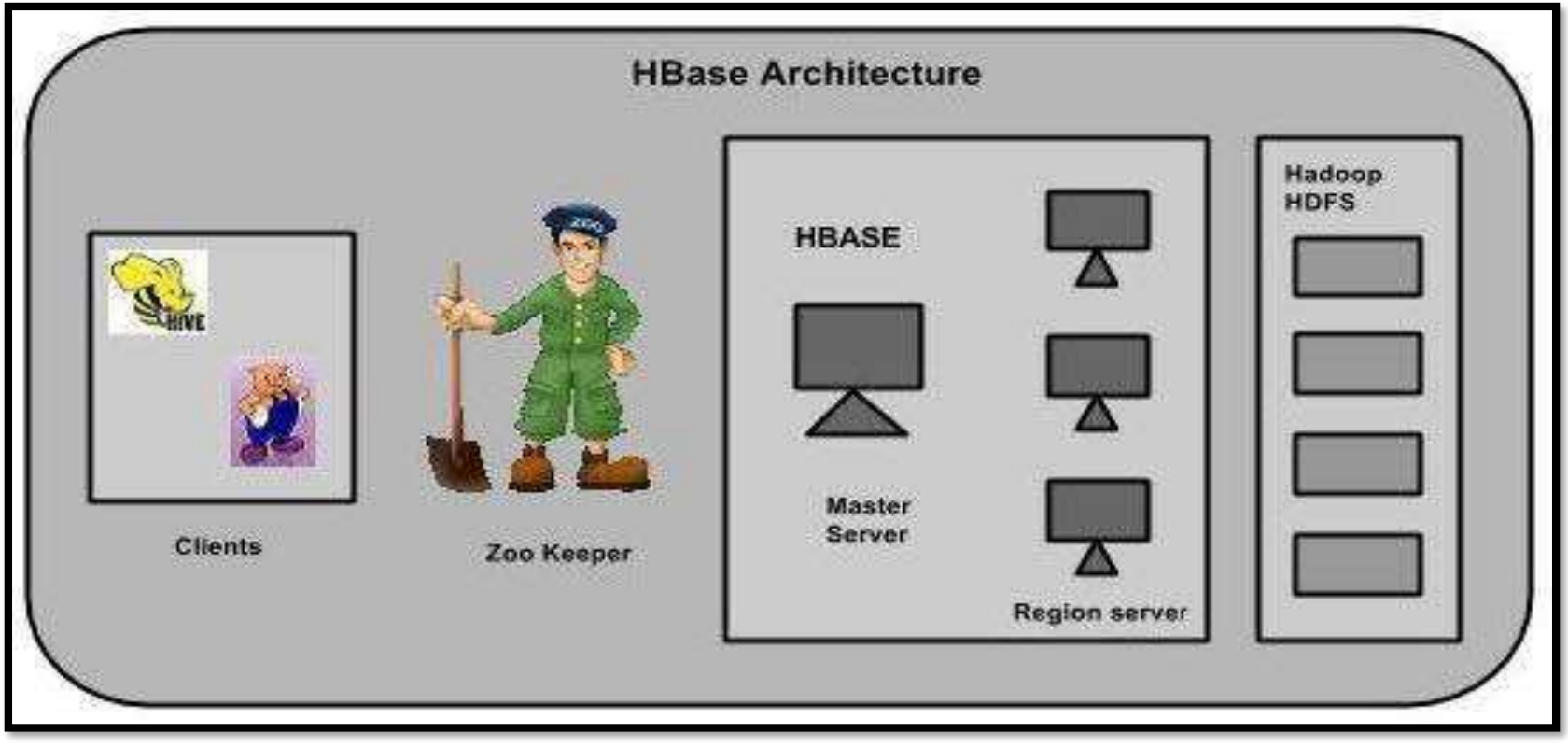


- Apache Zookeeper monitors the system.
- HBase Master assigns regions and load balancing.
- The Region server serves data to read and write.
- The Region Server is all the different computers in the Hadoop cluster.
- It consists of Region, HLog, Store, Memory Store, and different files.
- All this is a part of the HDFS storage system.

HBase - Components

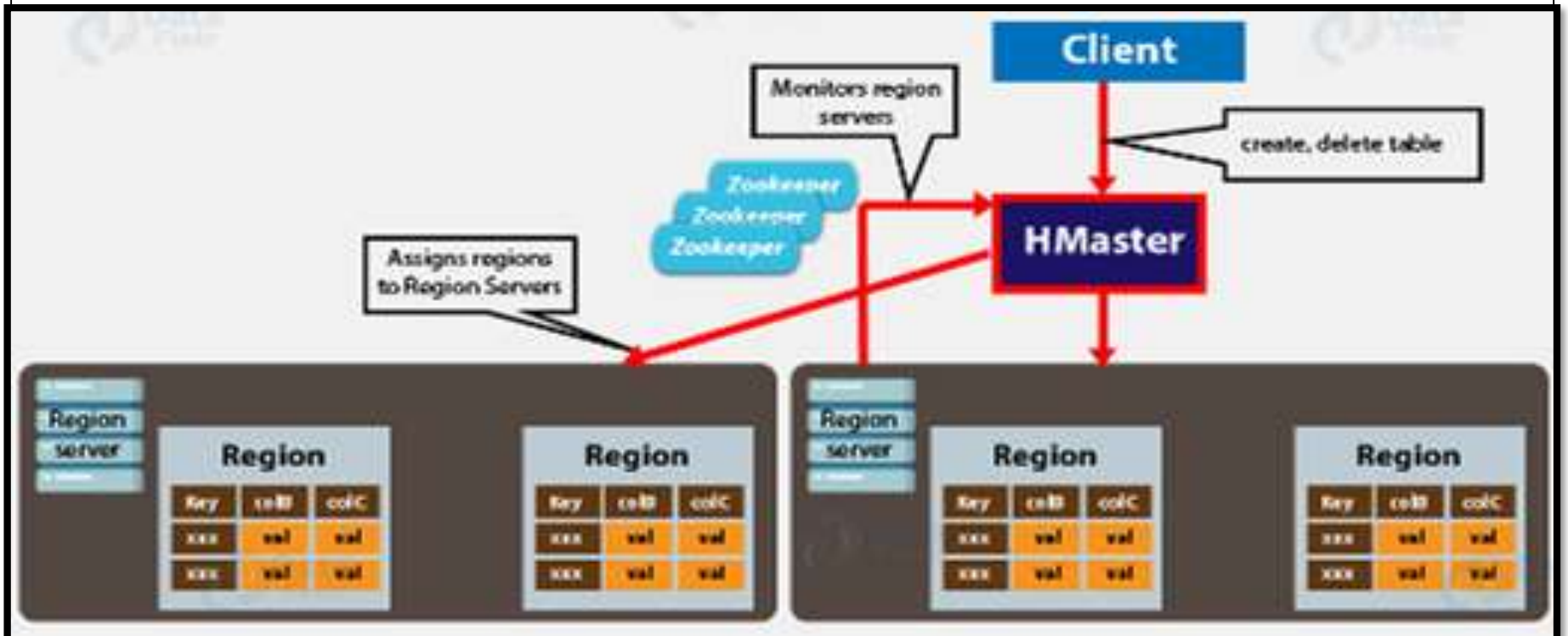
HBase has three major components -

1. Master servers. **2. Region server** **3. Zookeeper**



1. HMaster

- **HMaster** in HBase is the implementation of a Master server in HBase.
- It acts as a monitoring agent to monitor all Region Server instances present in the cluster and acts as an interface for all the metadata changes.
- In a distributed cluster environment, Master runs on **Name Node**.



Responsibilities of a HMaster in HBase architecture

a. Coordinating the region servers as following -

Assigns Regions on startup.

Recovery and load balancing.

Monitors all RegionServer instances in the HBase Cluster.

b. Admin functions

When a client wants to change any schema or change in Metadata operations, HMaster takes responsibility for these operations as follow-

- Table (create able, removeTable, enable, disable)
- ColumnFamily (add Column, modify Column)
- Region (move, assign)

2. Regions & Regions Server

➢**Table** is **split** According to **rowkey** Scope **horizontal** to several **region. (Start to end key)**

➢After split rows are called region and these Regions are assigned to certain nodes in the cluster for management is called **Region Server.**

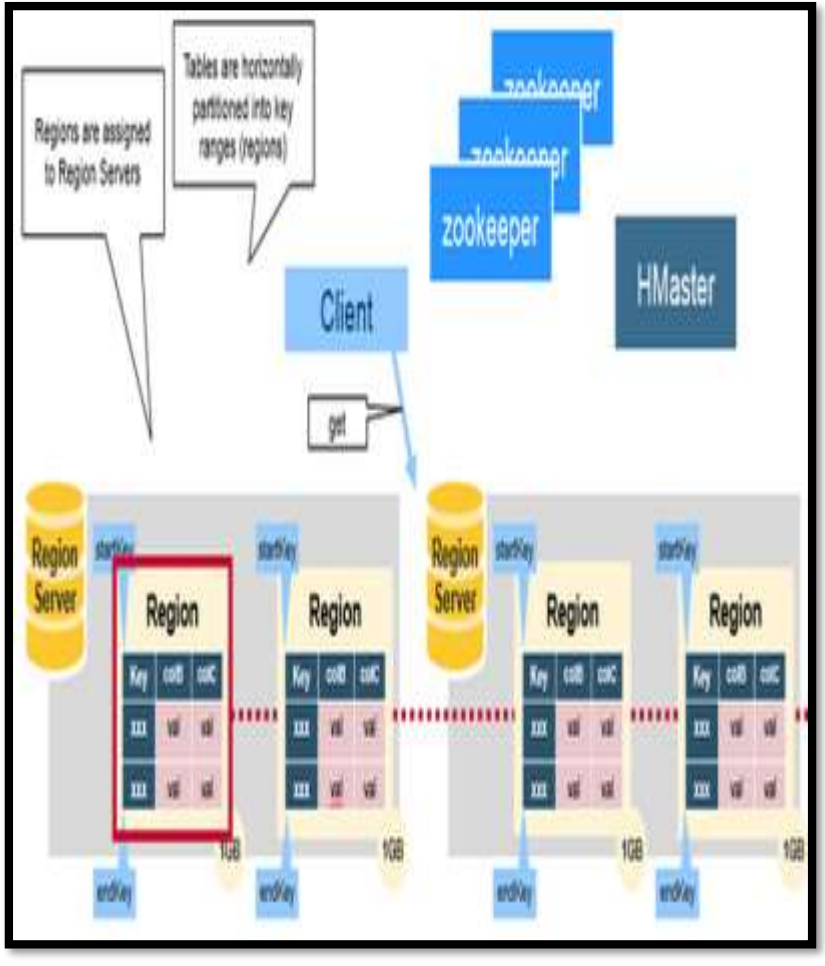
• They are responsible for processing data read and write requests.

• Each Region Server can manage about 1000 regions.

➢HRegion Server is the Region Server implementation.

➢Responsible for serving and managing regions or data that is present in a distributed cluster.

➢Region servers run on Data Nodes present in the Hadoop cluster.



How Regions splits

Sr.No	Personal Details		Education_ Details		Job details	
Emp_ id	Name	Age	Graduate	Percent age	Company Name	Designation
1	Amit	25	Bsc	76	HCL	Project Lead
2	Sumit	30	BTech	80	TCS	Project Manager
3	Varsha	35	MTech	75	Wipro	Project Engineer

HDFS

Region Server

1	Amit	25	Bsc	76	HCL	Project Lead
2	Sumit	30	BTech	80	TCS	Project Manager
3	Varsha	35	MTech	75	Wipro	Project Engineer

HBase Regions & Regions Server ..

➢When HBase Region Server receives writes and read requests from the client, it assigns the request to a specific region, where the actual column family resides.
➢Client can directly contact with HRegion servers, there is no need of HMaster mandatory permission to the client regarding communication with HRegion servers.
➢The client requires HMaster help when operations related to metadata and schema changes are required.

❖HMaster can get into contact with multiple HRegion servers and performs the following functions.

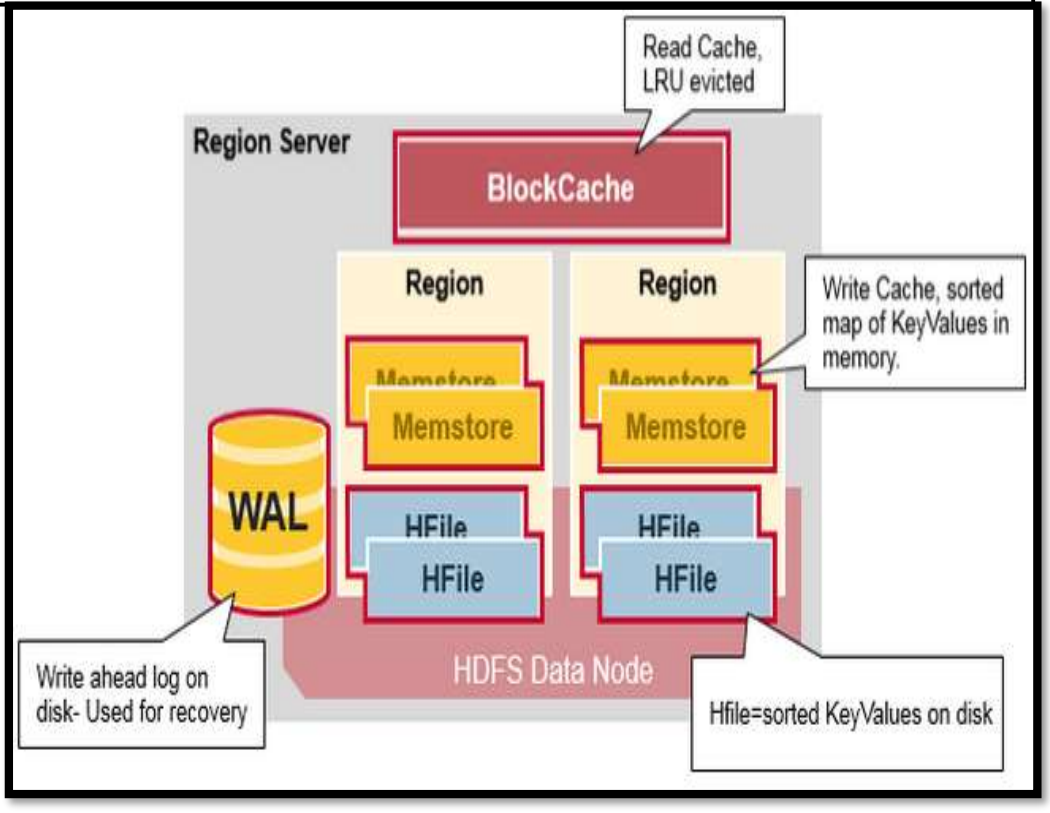
- Hosting and managing regions
- Splitting regions automatically
- Handling read and writes requests
- Communicating with the client directly

Region Server

- Region Server runs on HDFS DataNode and Responsible for processing data read and write requests.
- If Client required any data then he will directly interacts with Region Server.
- Regions are tables that are split up and spread across the region servers.

Components of Region server :

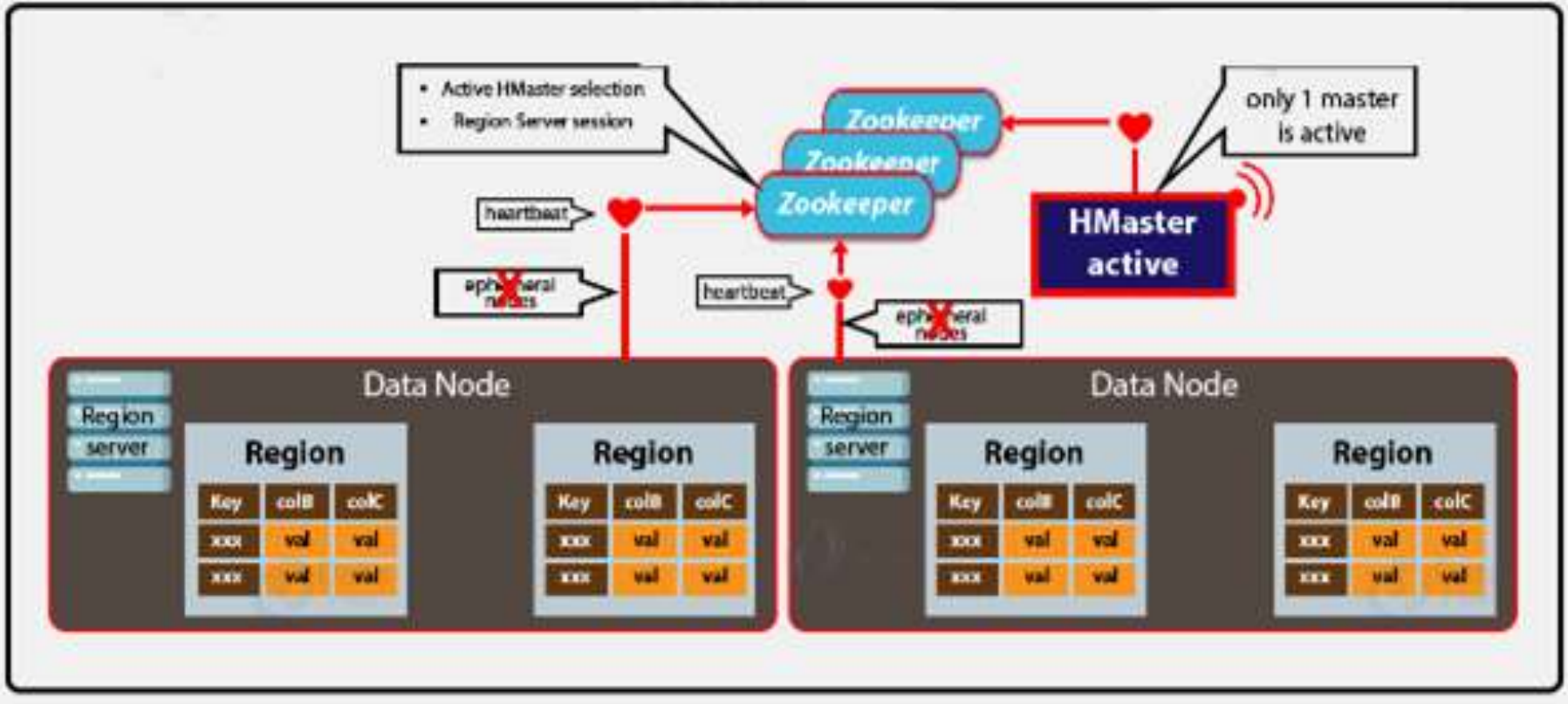
- **WAL (Write Ahead Log)** is a file on a distributed file system for **Storing new data**
- **Block Cache -** This is the read cache. **Memory The** most frequently accessed data is stored in the LRU (Least Current Used) cache.
- **MemStore -** This is the write cache, in **Memory**.
- **Hfile -**Store HBase data on hard disk (HDFS).



3.Hbase- Zookeeper

- Hbase use Zookeeper to coordinate shared state information for members of distributed systems.
- Active HMaster and Region servers, connects with a session to Zookeeper.
- For active sessions ZooKeeper maintains ephemeral nodes by using heartbeats.
- Zookeeper maintains which servers are healthily available and notifies them when the server fails.
- **Ephemeral nodes** mean znodes which exist as long as the session which created the znode is active and then znode is deleted when the session ends.
- Zookeeper uses a consistency protocol to ensure the consistency of the distributed state.
- Each Region Server in HBase Architecture produces an ephemeral node. Further, to discover available region servers and HMaster shall monitors these nodes.
- Active HMaster sends heartbeats to Zookeeper.

How the Components Work Together



Working Process of Zookeeper

1. **Active HMaster**
2. **Inactive Hmaster**

❖ HBase META Table

- META Table is a special HBase Catalog Table. Basically, it holds the location of the regions in the HBase Cluster.
- It keeps a list of all Regions in the system.
- Structure of the .META. table is as follows:
- **Key:** region start key, region id
- **Values:** RegionServer

Hbase Table parameters

- **Tables**: Data is stored in a table format in Hbase.
- **Row Key**: Row keys are used to search records which make searches fast.
- **Column Families**: Various columns are combined in a column family. These column families are stored together which makes the searching process faster because data belonging to same column family can be accessed together in a single seek.
- **Column Qualifiers**: Each column's name is known as its column qualifier.
- **Cell**: Data is stored in cells.
- **Timestamp**: Timestamp is a combination of date and time. Whenever data is stored, it is stored with its timestamp.

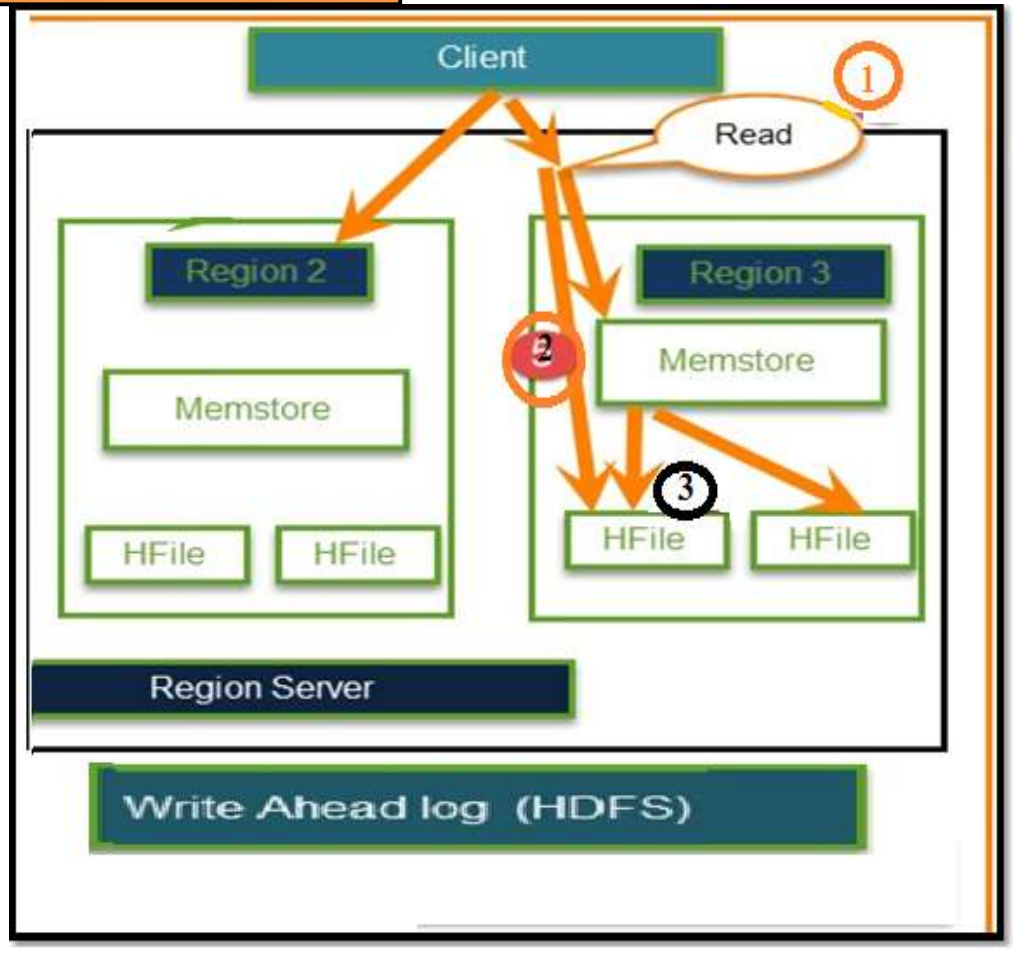
HDFS vs. HBase

HDFS	HBase
HDFS is a Java-based file system utilized for storing large data sets.	HBase is a Java based No-SQL database.
HDFS has a rigid architecture that does not allow changes. It doesn't facilitate dynamic storage.	HBase allows for dynamic changes and can be utilized for standalone applications.
HDFS is ideally suited for write-once and read-many times use cases	HBase is ideally suited for random write and read of data that is stored in HDFS.

HBase - Read

- A Read against HBase must be reconciled between the HFiles, MemStore & BLOCKCACHE.
- The Block Cache is designed to keep frequently accessed data from the HFiles in memory so as to avoid disk reads.
- Each column family has its own Block Cache.

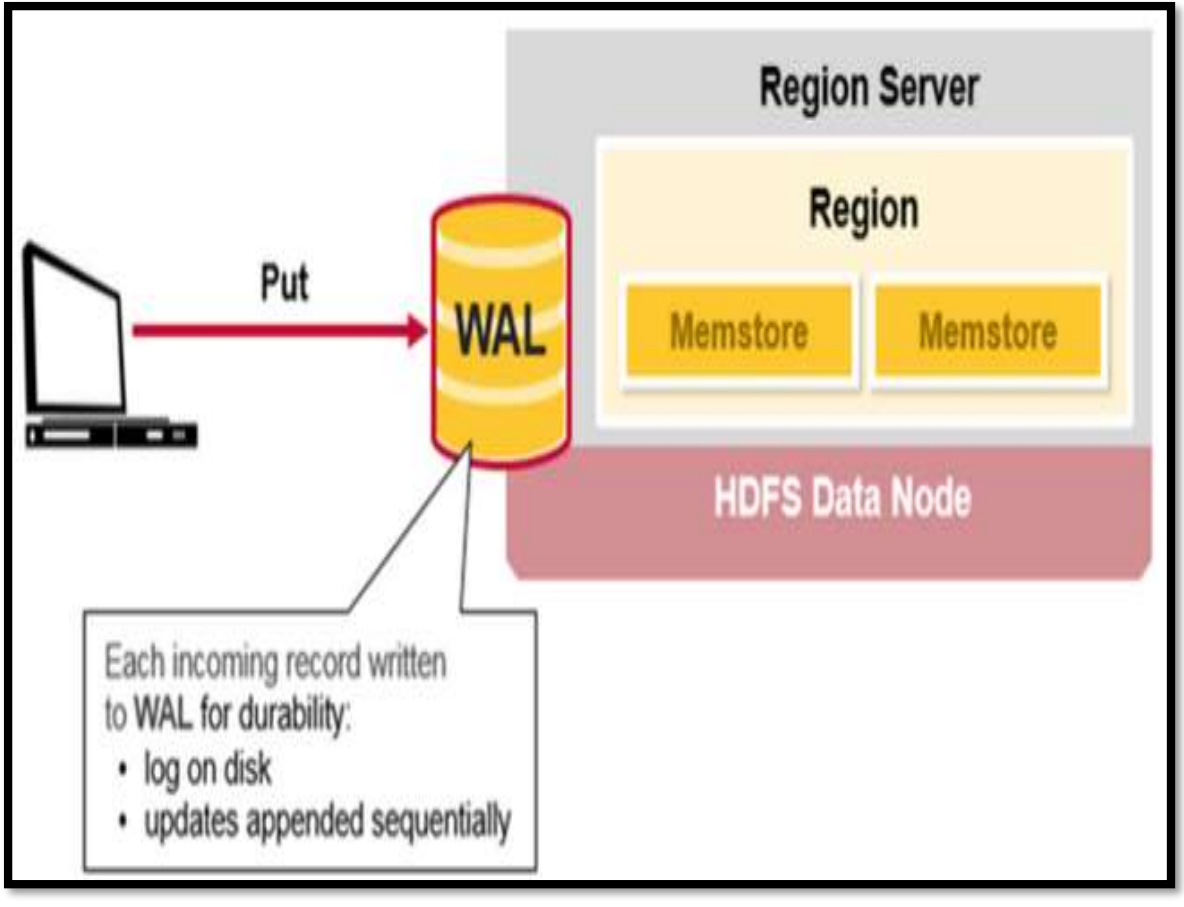
  Block: It is the smallest indexed unit of data and is the smallest unit of data that can be read from disk. default size 64KB.



Hbase - Write

When a write is made, by default, it goes into two places:

- write-ahead log (WAL), Hlog.
- in-memory write buffer, MemStore.



Advantages of HBase

- Hbase designed to store Denormalized Data.
- Hbase Supports Automatic Partitioning
- **Strong consistency model**– All readers will see same value, while a write returns.
- **Scales automatically**
  – While data grows too large, Regions splits automatically.
  – To spread and replicate data, it uses HDFS.
- **Built-in recovery** – It uses Write Ahead Log for recovery.
- **Integrated with Hadoop**
- Hbase is **schema-less, no data model has been defined.**
- Hbase has the ability to perform Random read and write operations.
- Hbase provides **data replication** across clusters for higher availability.
- Feature **random access** (internal hash table) to stores data in HDFS files for faster lookups/searching.

Disadvantages of HBase

- **Single point of failure -** If HMaster goes down, complete cluster will be fail and no work/task will be performed.
- Cannot perform functions like SQL and doesn't support SQL structure.
- Does not contain any query optimizer
- Does not support for **transaction**.
- **Business continuity reliability**
  – Write Ahead Log replay very slow.
  – Also, a slow complex crash recovery.
- **Joining** and **normalization** is very **difficult** to perform.
- Very difficult to store large **binary data**.

Real Time Example of HBase-Facebook

How Facebook use Hbase to store user data

User Account ID	Account Type (Personal/ Business)	Type of Contents Posted	Posted for (Public/ Private)	Time Stamp	Violating community standards	Last Login Activity Time of Account
Rahul3@fb	Personal	Image + Text	Public	20/08/2022 08.45.00.PM	No	05.30 PM
ABZ@fb	Business	Text +Video	Public	25/08/2022 06.45.00.PM	No	08.30 PM
Tarun@fb	Personal	Text +Video	Public	25/08/2022 06.45.00.PM	Yes	08.30 PM

User Account ID	Account Type (Personal/ Business)	Type of Contents Posted	Action Required	Account Suspended	Account Blocked	Remarks
Tarun@fb	Personal	Text +Video Is a hated contents	Yes	For a period 2 week etc.	Yes	Violating Community standard

Any Query??