HBase

RDBMS	HBASE
- Requires SQL	NoSQL
- Fixed schema	No fixed schema
- Row oriented	Column Oriented
- Not Scalable	Scalable
- Static	Dynamic
- Slower retrival of data	Faster retrival
- It follows ACID prop.	It follows CAP (consistency, Availability, Partition - tolerance) theorem
- Handles structured data	Handles structured, semi-str. & unstruc. data
- Cannot handle sparse data	Can handle sparse data.
- Latency is present	low latency

* Limitations of hadoop
- Performs batch processing, sequential manner
- Needed to search entire dataset.
- Processing a huge dataset results into huge data set, which is also processed sequentially

FEB 2022 M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S
• 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 • • • • • •

HBase.

→ Open-source, horizontally scalable.
→ Distributed column-oriented db build on top of hadoop fs.
→ Provides random real-time read/write access to data in hadoop fs.

HDFS	HBase
- For storing files	Storing in HDFS & processing.
- does not support fast individual record lookups	Provides fast lookup for larger tables.
- High latency batch processing	Low latency access to single rows (Random access)
Sequential access of data	Uses hash tables & provides random access, & stores data in indexed HDFS for faster lookups.

• It is column-oriented db. and table are sorted by row. The table schema only ~~provides~~ defines column families, which are key-value pairs.

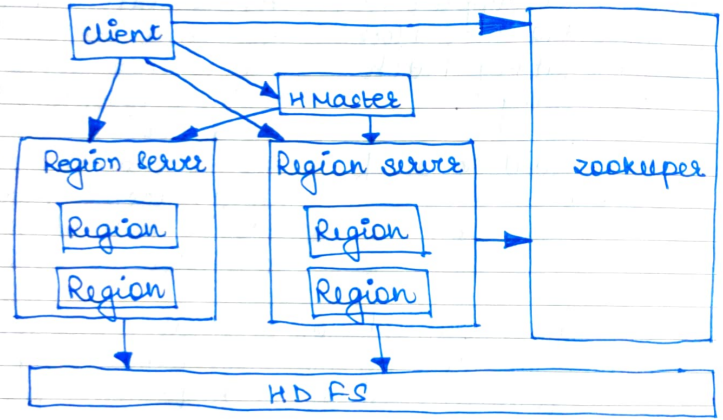
- Table is a collection of rows
- Row is a collection of column families
- Column family is a collection of columns
- Column is a collection of key-value pairs.

M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S MAR 2022
• 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 • • •

MAR APR MAY JUN

- Features of HBase
  - Linearly scalable.
  - Automatic failure support.
  - Provides consistent reads & writes.
  - Integrates with Hadoop, both as source & destin.
  - Has easy JAVA API for client.
  - Provides data replication across clusters.

★ Architecture



1. HMaster :-
   - → Master server of HBase
   - → Process in which regions are assigned to region server as well as DDL operations.
     - Table (create Table, remove, enable disable)
     - Column family (add column, modify column)
     - Region (move, assign)

   - → Manages region server instances present in cluster.
   - → Controlling load balancing, failover etc.
   - → Acts as an interface for DDL operations.

2. Zookeeper :-
   - → Maintains server state, coordination service.
   - → Provides services like maintaining configuration information, naming, providing distributed synchronization, server failure notification etc.
   - → Clients communicate with region servers via zookeeper

3. Region Server
   - → HBase Tables are divided horizontally by row key range into Regions.
   - → Regions are the basic building elements of HBase cluster that consists of distribution of tables & are comprised of column families.
   - → Runs on HDFS DN.
   - → Regions are responsible for handling, managing, executing as well as reads & writes HBase operations on that set of region.
   - → Default size of region → 256MB.

   - → HMaster → Region server
     - Hosting & managing regions
     - Splitting regions automatically
     - Handling read & Writes requests.
     - communication with client directly

• Region → building element of HBase consist of distributed of tables & comprised of column families. contains multiple store, one for each column family. Has Memstore, Hfile.

05 SATURDAY FEBRUARY 2022
DAY 036-329 WEEK 06

★ MetaTable. (stored on Region Servers)
→ It holds the location of the regions in HBase cluster.
→ It keeps list of all Regions in the system
→ Structure of Mata table:
1. key: region start key, region id.
2. Values: Region Server.
→ Handled by zookeeper.
- It is like a binary tree.

★ HBase Write Steps
① First step: Is to write the data to the write-ahead log, while the client issues a put request:
→ To the end of WAL file, all the edits are appended which is stored on disk.
→ In case a server crashes, WAL is used to recover not-yet-persisted data

② As soon as data is written to WAL, it is stored in Memstore. (256 mb written stored in → Hfiles in chunks 100 mb)

★ Region Server Components

1. WAL (Write-ahead log)
→ stores new data to permanent storage.
→ Also used for recovery.

2. Block Cache
→ It is the read cache.

FEB 2022 M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S S
• 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 • • • • • • •

06 SUNDAY FEBRUARY 2022
DAY 037-328 WEEK 06

read
→ stores frequent data in memory & removes least read data when full.

3. Memstore
→ Write Cache, stores new data which has not been written to disk.
→ sorts before writting.
→ One Memstore per column family

4. Hfiles
→ These files stores the rows as sorted KeyValue on disk.

★ Compaction
→ Combining of Hfiles to reduce the storage & reduce the no. of disk seeks needed for read.
1. Minor: Picks smaller Hfiles & combines them to bigger Hfile. It performs merge sort.
2. Major: In a bigger Hfile the same column families are placed together.
→ Scheduled during low peak load timings to avoid congestion.

M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S S MAR
• 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 • • • 2022

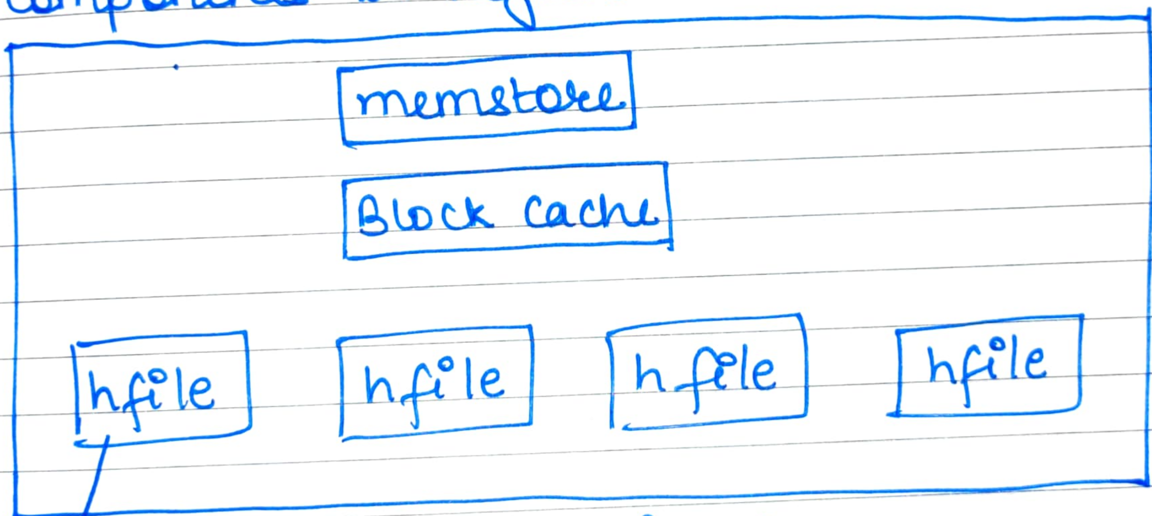
MAR APR MAY JUN

HBASE Architecture

- Zookeeper → Hmaster → Region Server → Regions → Column Family store.

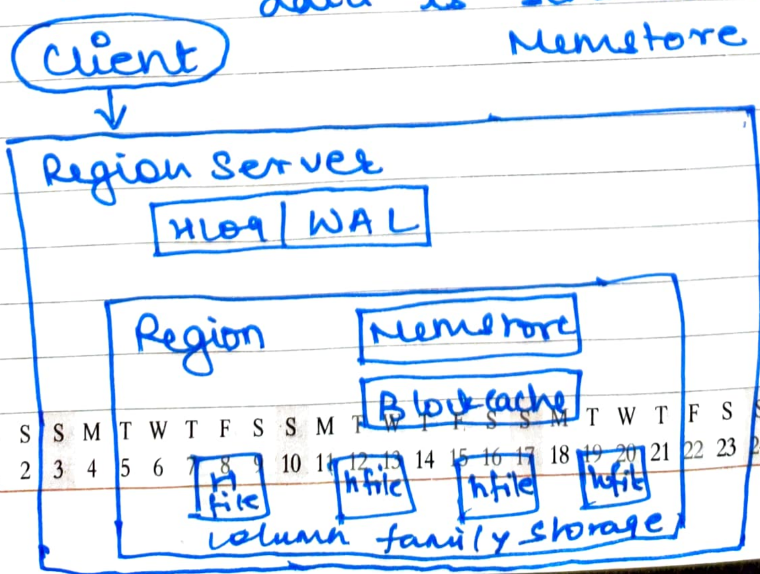
- Region
  → Actual data is splitted in regions
  → size (default) of region → 256 mb. (for optimal performance).

→ components in Region



storing column families

Write operation



→ data is sent to WAL & then to Memstore → after memory is full of memstore it flushes data into Hfiles.

Before writing data on disk, it is written in buffer which is memstore (assume 100 mb)

→ Read Operation

8

client → Metatable on Region Servers &
→ It gives location of region where
9 table is stored → memstore, Block
cache, hfiles are scanned for data. →
10 given to client.

11

12

1

2

3

* Commands

→ status – Provides the status of HBase, no. of servers
→ version – Provides the version of HBase
→ table_help – Provides help for table-reference cmds.
→ whoami – Provides information abt the user.

* Data definition language
→ create, list, disable, is_disabled, enable, is_enabled, describe, alter, exists, drop, drop_all,

* DML
→ put – puts a cell value at a specified column in a specified row in a particular table.
→ get – Fetch contents of row or a cell.
→ delete – delete a cell value in a table
→ deleteall – delete all the cells in a given row.
→ scan, count, truncate,

eg.

Row key	Data
cutting	info: {'height': '9ft', 'state': 'CA'} roles: {'ASF': 'Director', 'Hadoop': 'Founder'}
tlipcon	info: {'height': '5ft7', 'state': 'CA'} roles: {'Hadoop': 'committer'@ts=2010, 'Hadoop': 'PMC'@ts=2011, 'Hive': 'contributor'}

creates skp. HBase tables of column families.

info Column Family

Row key	Column key	Timestamp	Cell value
cutting	info: height	1273516197868	9ft
cutting	info: state	1043871824184	CA
tlipcon	info: height	1273878447049	5ft 7
tlipcon	info: state	1273616297446	CA

Roles Column Family

Row key	Column key	Timestamp	Cell value
cutting	roles: ASF	1273878230 22	Director
cutting	roles: Hadoop	1183746289103	Founder
tlipcon	roles: Hadoop	. . .	PMC
tlipcon	roles: Hadoop	. . . .	committer
tlipcon	roles: Hive	. . . .	contributor

Sorted on disk by Row key, column key, desc. timestamp.