HBase

Background

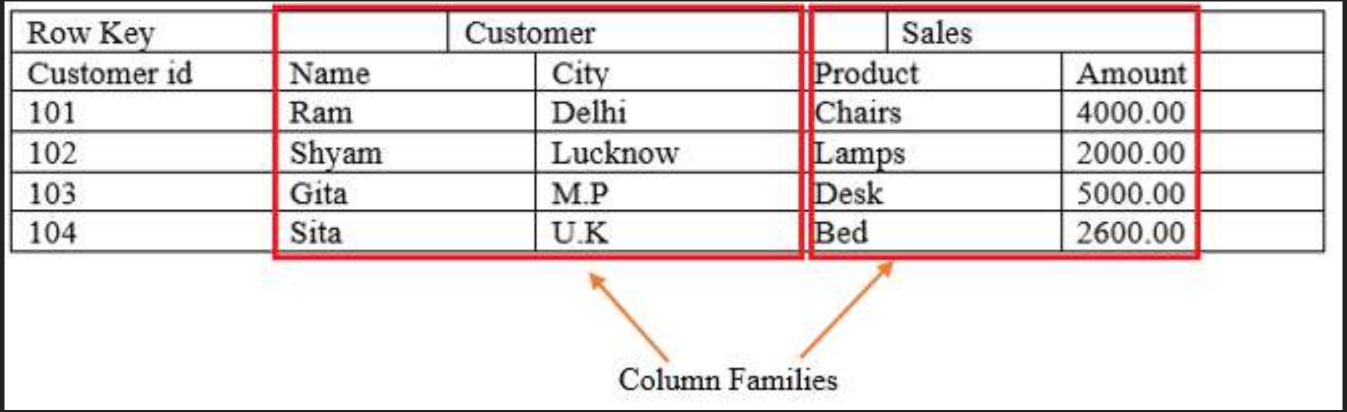
HBase is a wide column storage database designed for either transactions or analytics processing, and it is built on top of HDFS. In this video, we will compare how it performs for certain operations with the other popular wide column data store, Cassandra.

HDFS provides high latency writes and reads (since they all come directly from disk), and only allows for sequential writes as opposed to in the middle of a file - we will see how using an LSM tree based architecture HBase can mitigate this issue and provide low latency writes and reads on random accesses.

Data Model

Wide column format:

- Each row has a single row key, and any columns that they want (or column families which consist of multiple values)
- The row key can be comprised of multiple parts which allows the sorting of rows within a table

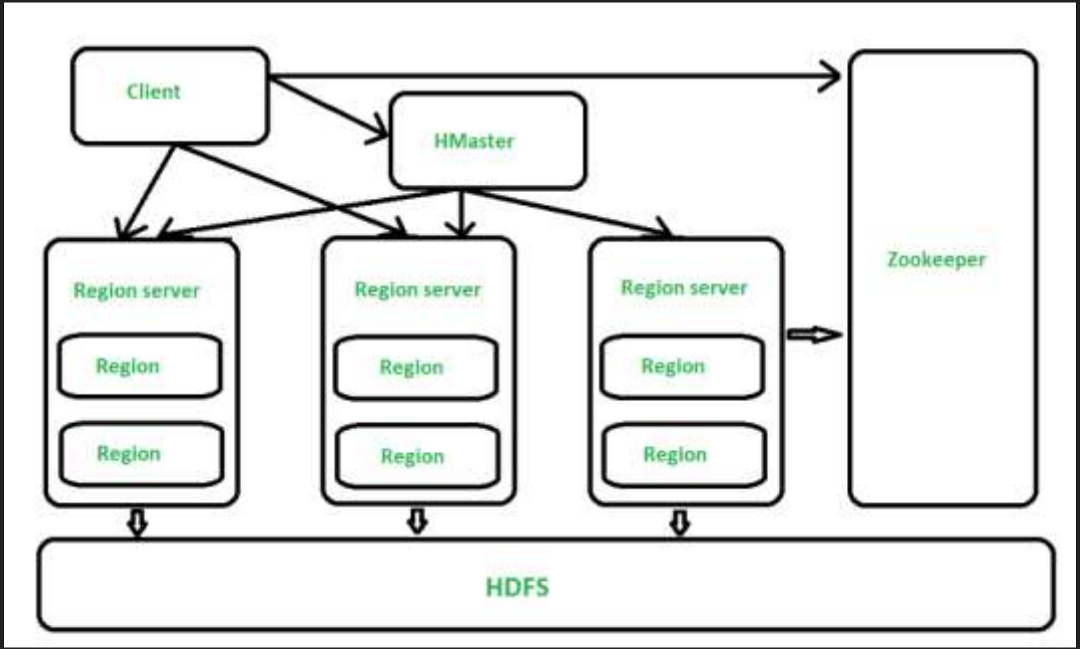


Row Key	Customer		Sales	
Customer id	Name	City	Product	Amount
101	Ram	Delhi	Chairs	4000.00
102	Shyam	Lucknow	Lamps	2000.00
103	Gita	M.P	Desk	5000.00
104	Sita	U.K	Bed	2600.00

Column Families

Architectural Overview

- Master server
- Region server
- Replication



Master Server

Many parallels to HDFS NameNode:

- Stores all file metadata, as well as the locations of the chunks of files
  - Keep in mind that this is effectively the partitioning schema of HBase
  - Range based partitioning that can be split if too big, based on row key
- For both reads and writes, client first reaches out to master server in order to figure out the location of the file that it will be reading to and writing from
- Can be used in conjunction with ZooKeeper to replicate its write ahead log and thus support a backup master node, increases fault tolerance

Region Server

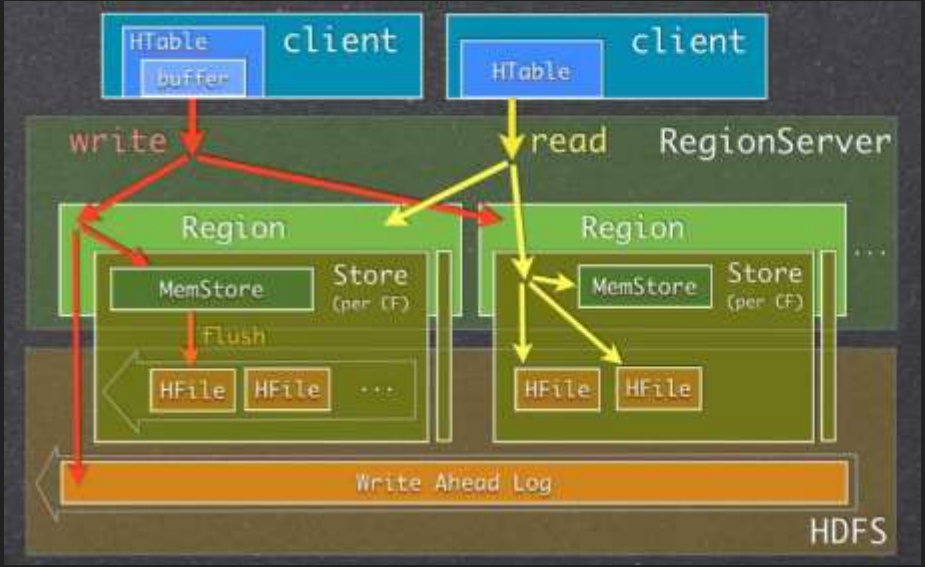
Run as a separate thread on HDFS datanode servers:

- Occasionally send heartbeats to ZooKeeper so master knows they are alive
- Uses an LSM tree style in memory data structure where writes are first sent (in addition to a write ahead log for persistence)
- Once the LSM tree becomes too large, writes are flushed to sorted HFiles
  - Basically the same as an SSTable
- SSTables stored in column oriented format
  - Good for high read throughput and analytics over a range of the row key

Replication

Recall: Region nodes, occasionally will write new HFiles (similar to SSTables).

Since the region node is (usually) running on an HDFS replication node, it uses the HDFS replication pipeline to synchronously replicate the HFile to other nodes.



Analytics Perspective

- Column oriented storage (all elements of a column family stored in one file) make for very high read throughput of one column
- Being built on HDFS allows for good integration with both MapReduce and dataflow engines (Spark, Tez) for high throughput analytics
    - Use batch processes to perform joins! Joins are not supported in HBase

Conclusion

While HBase looks very similar to Cassandra on a surface level, they serve very different use cases!

Cassandra:

- Real time transactions processing database, super fast writes

HBase:

- Great as a data lake for running huge batch/streaming processes on, also capable of fast reads and writes
- If you want to support more structured data with advanced queries, opt for a SQL based data warehouse