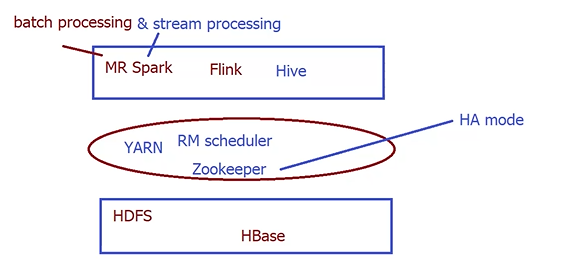
HBase

-Flink and spark both batch and stream processing

-yarn managing resources

-zookeeper coordination and high availability mode



A white box with black text

AI-generated content may be incorrect.

-column-based differs from row-based

A close-up of a computer screen

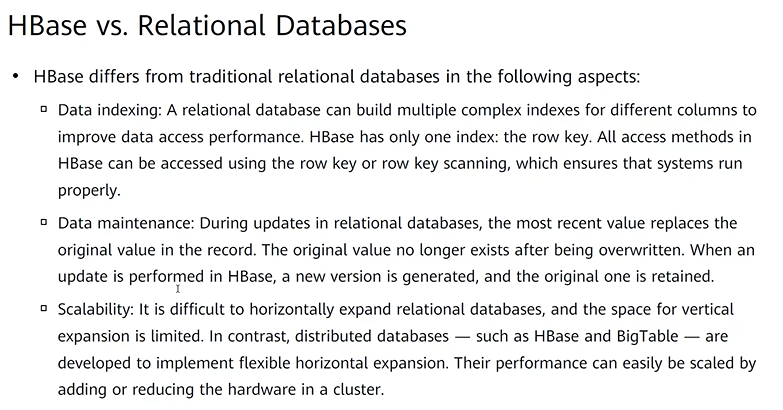
AI-generated content may be incorrect.

-I am working in a sharing environment and want the tools to have high reliability working 24/7…

-to access the data in real-time is not one of the properties of the hdfs, not just accessing but editing it in real-time

-the Hbase itself in its architecture is based on hdfs, but the hdfs itself cannot support the real-time read and write operations

-HBase can not work without zookeeper, it is a main component, as it provides the high availability mode

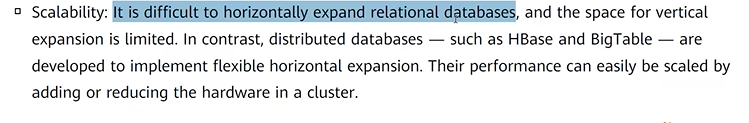


-RDB = Relational Database

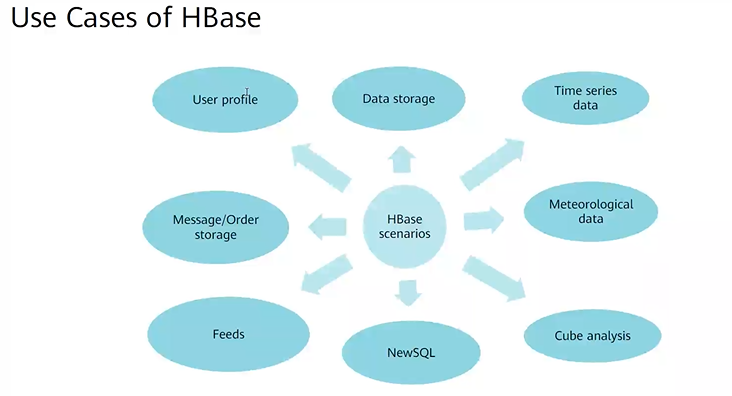
-Data indexing:can design complex index which helps retrieve the data fast, which improves the performance

-row key interval = multiple ones, can read the values corresponding to all of those row keys

-the new and the old version of the data can be differentiated upon using the time stamp



-HBase and BigTable have their performance improve just by adding more nodes, expansion can be scaled easily instead of depending on physical resources

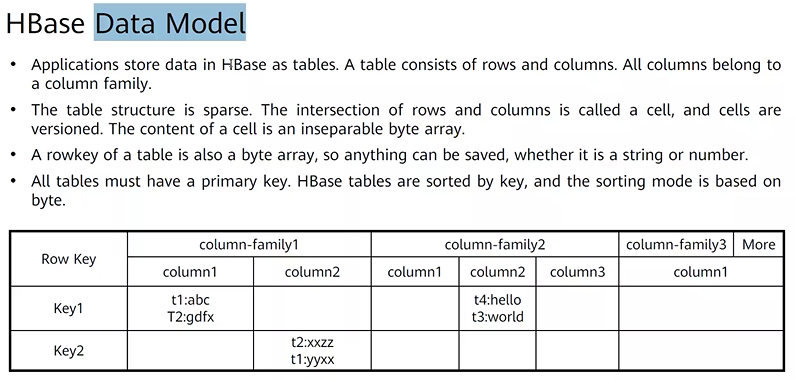


-the data is not always structured, it is various:structured/semi-structured/…

-the data is very massive

-the atomicity is not looked upon in the hbase: there is no all-or-nothing principle here

-high throughput exists in the Hbase, it is stored in several regions serving multiple users at the same time



-every group of columns are linked by a column family

-by grouping into families the performance improves retrieving the needed info once

-in traditional, there can’t be null or missing values like that

-a cell is a inseparable byte array:to allow any type of data that I want without predescription

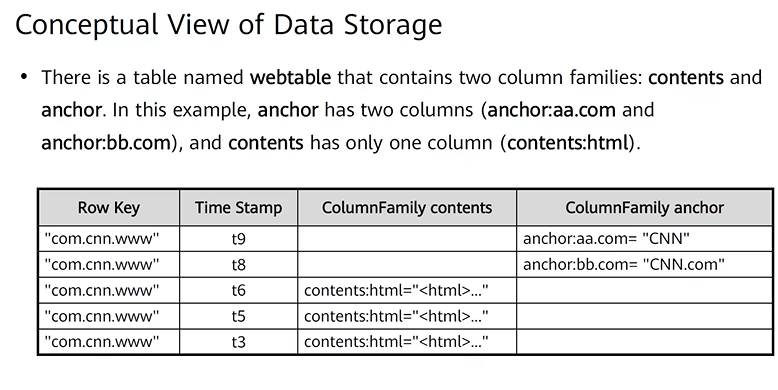
-the rowkeys are also byte arrays for the same reason

-Big table gets windowed by horizontal lines

-the windows get distributed to region servers

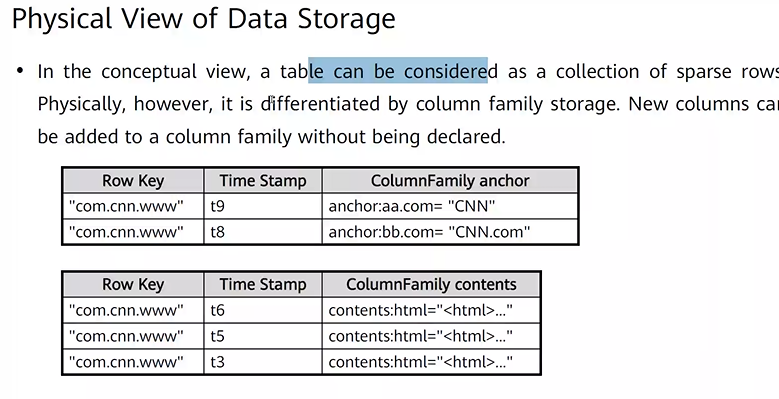


-column qualifier: basically the name given to the column



-conceptual=something temporary something visual that I can see while running

-so basically this is the versions of the given row at different time points, but this isn’t the real case but something that you can visualize as a user or so



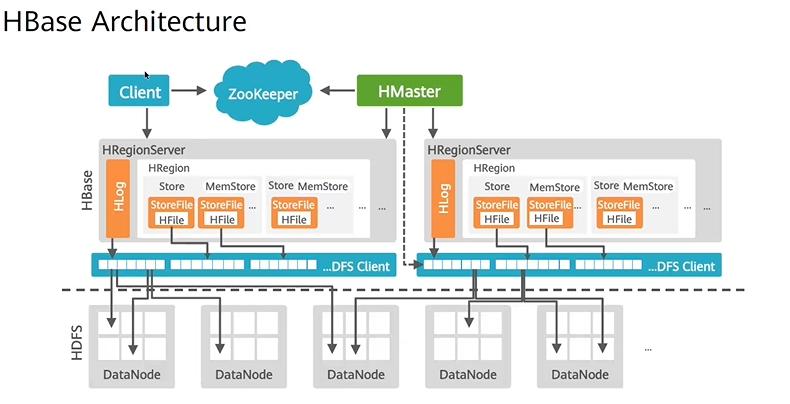
-no needed preparation, if I want to add a new column family I just do it

A screenshot of a computer

AI-generated content may be incorrect.

-to read in row-based, I need to go through full row to get what I want which decreases the performance needing additional time

-if It is semi-structured I need to go to column-based deliberately



-client takes the read/write(demand site)

-Hmaster is the management node(master node) for the active-standby mode/ high availability mode

-zookeeper provides the high availability mode

-memstore is where the caching occurs

-if failure occurs to Hregionserver it migrates, it is migrated by hmaster and the decision is determined through hlog

-i want the load to be balanced on the servers no full load on one server, this is also the responsibility of the hmaster

-hmaster is responsible for the creation of the stored data

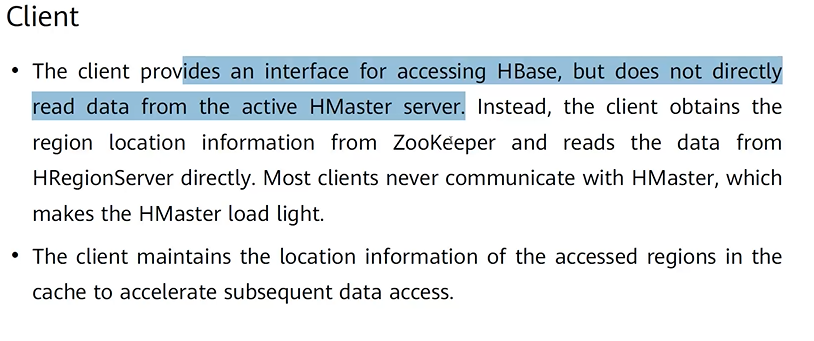
-zookeeper is connected to the client due to it having the routing information(the metadata)

-client has cache memory that keeps the temporary needed info like the address

-Hbase has a process called flush, which is getting rid of the data

-as long as the data is in Hbase, it maintains its order, but when transferred from it this doesn’t happen anymore

-if it is read/write operation the data is cached in the memstore, the data goes to the hdfs and then to the datanodes, it was ordered first in the rowkeys but now that it went to the datanodes it goes to the available places making it become not sorted anymore



-now Hmaster doesn’t get involved in the read/write operations of the client making the load light on it



-sometimes I need to make a split, sometimes the region reaches the threshold and becomes too large

-the zookeeper monitors all of the nodes

A white background with black text

AI-generated content may be incorrect.

A close-up of a list

AI-generated content may be incorrect.

-storeFile doesn’t belong to the hbase but belongs to the hdfs, the format is hformat

A screenshot of a computer

AI-generated content may be incorrect.

-the region has a certain threshold, if reached the region must be splitted into whatever number based on the keys given

-the splitting doesn’t mean the dividing the data or distribution of it, but the data itself is transferred to be in two regions instead of one, but not actually divided

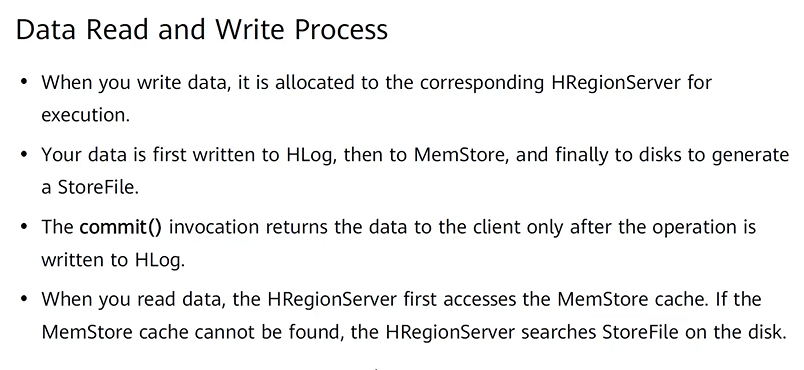
A close-up of a book

AI-generated content may be incorrect.

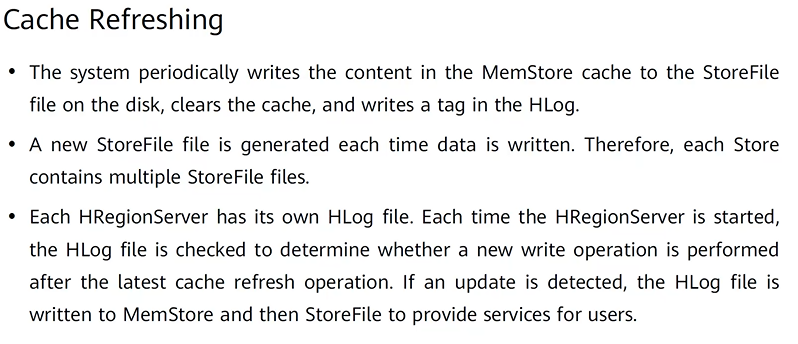
-I can not actually understand the purpose of splitting if it works this way, what did it add exactly

A screenshot of a computer

AI-generated content may be incorrect.

****

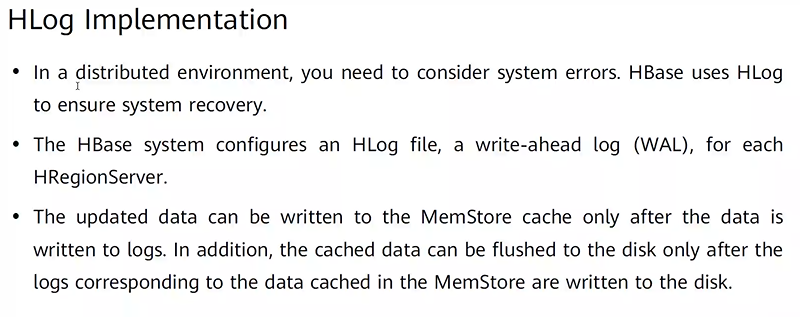
-the data is first written in Hlog which is written, to know later whether the data was written correctly or not, before continuing anything, and monitoring it further , when it goes to a new regionserver the logs are checked to see how to continue further



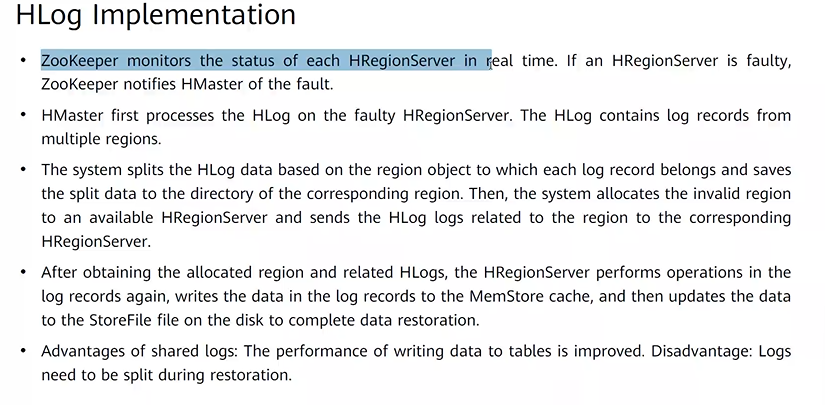
-if he found that there are instructions in the hlog that hasn’t been completed, then it must be completed first

A screenshot of a computer

AI-generated content may be incorrect.

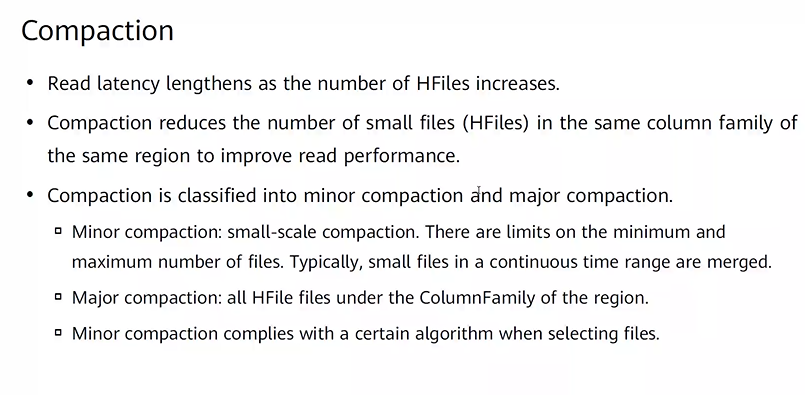


-the order:hlog/memstore/disk must happen in this order, If something wrong happens the process must be started from over



-hmaster takes from the hlog what is important considering each region

-the logs are split for me to give everyone what it needs



-for low latency splitting is needed if the number of hfiles becomes very large

A diagram of a diagram

AI-generated content may be incorrect.

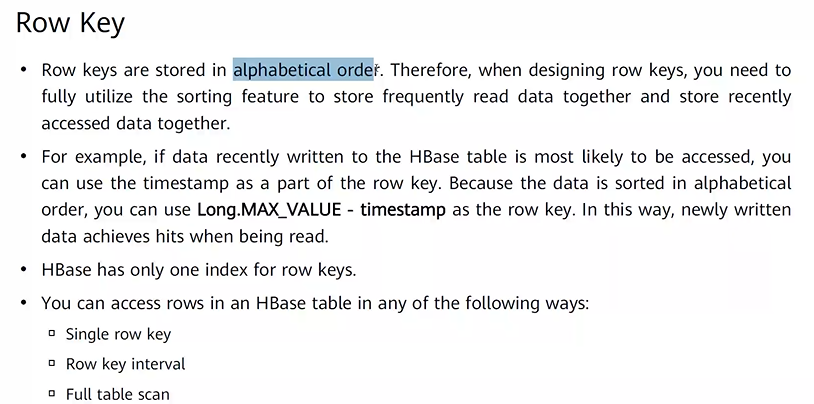
A screenshot of a computer

AI-generated content may be incorrect.

A white background with black text

AI-generated content may be incorrect.

-it is not actually a filter, but rather a method of checking



-the timestamp is used as a method to make access faster

A white background with black text

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.