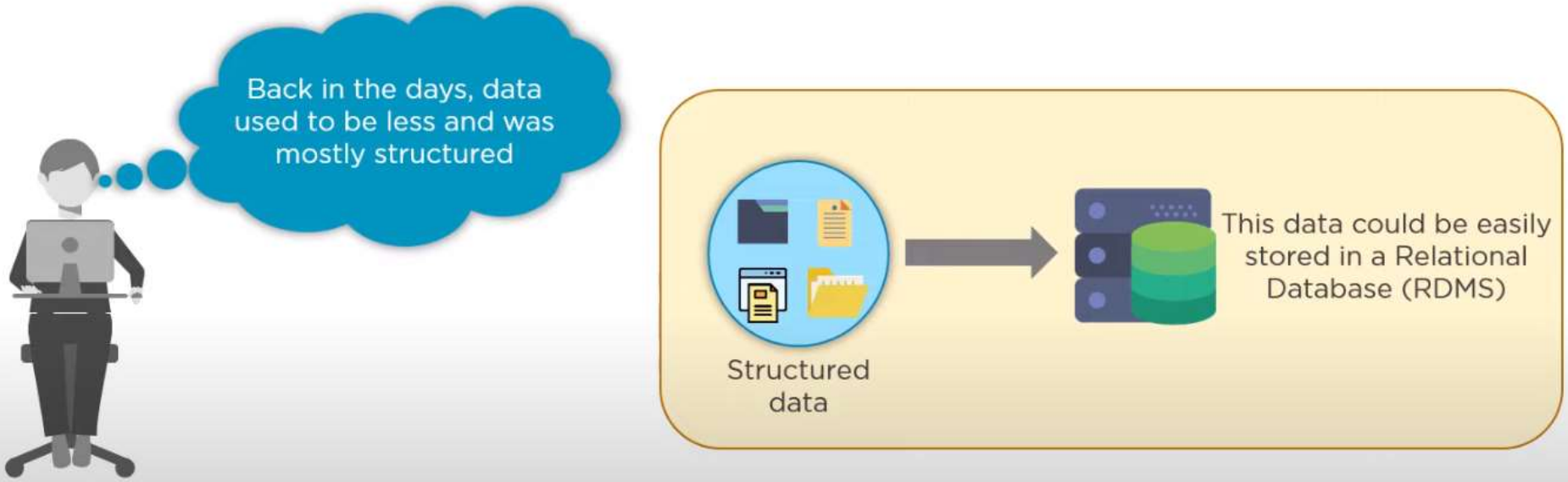
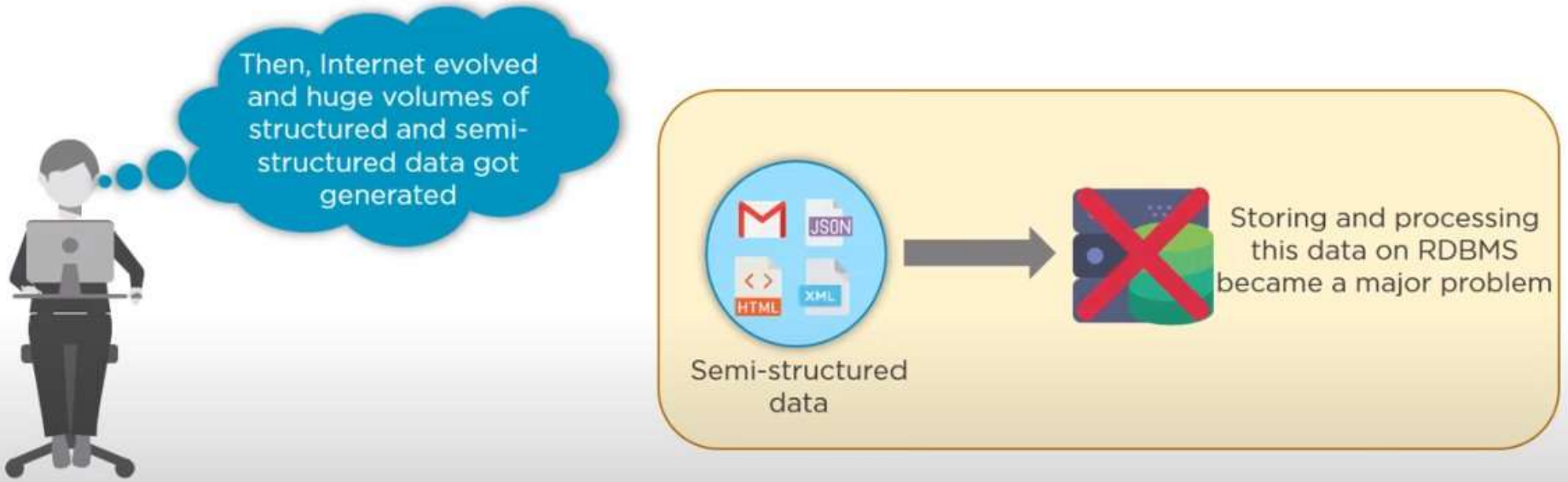
Introduction to HBase

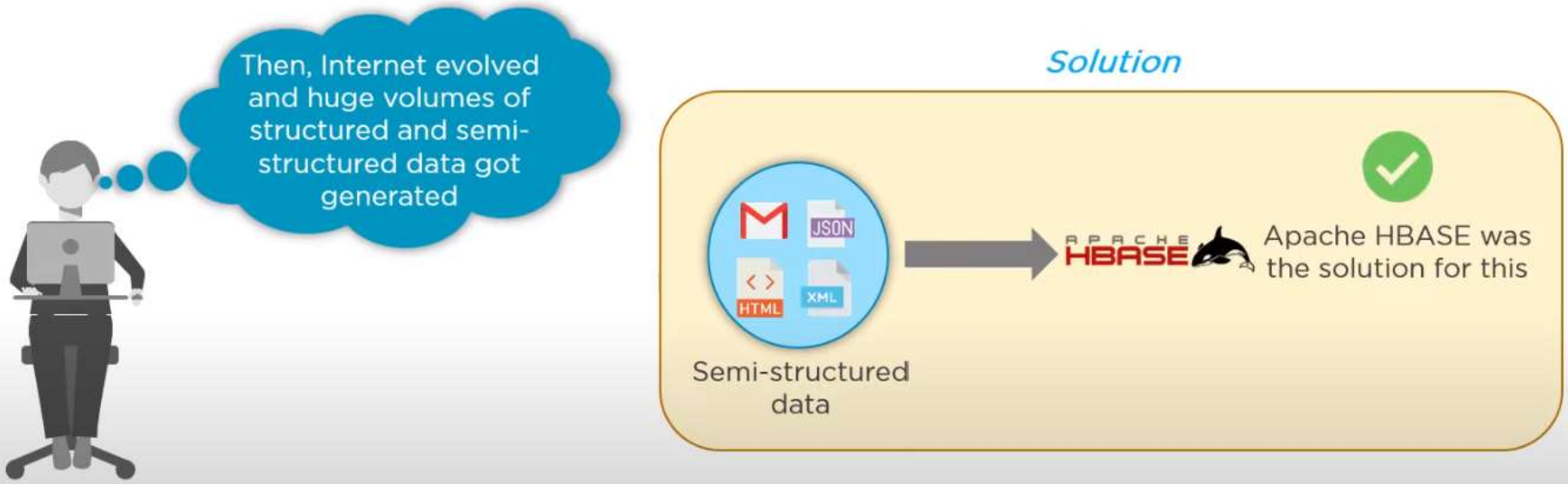
Introduction to HBase



Introduction to HBase

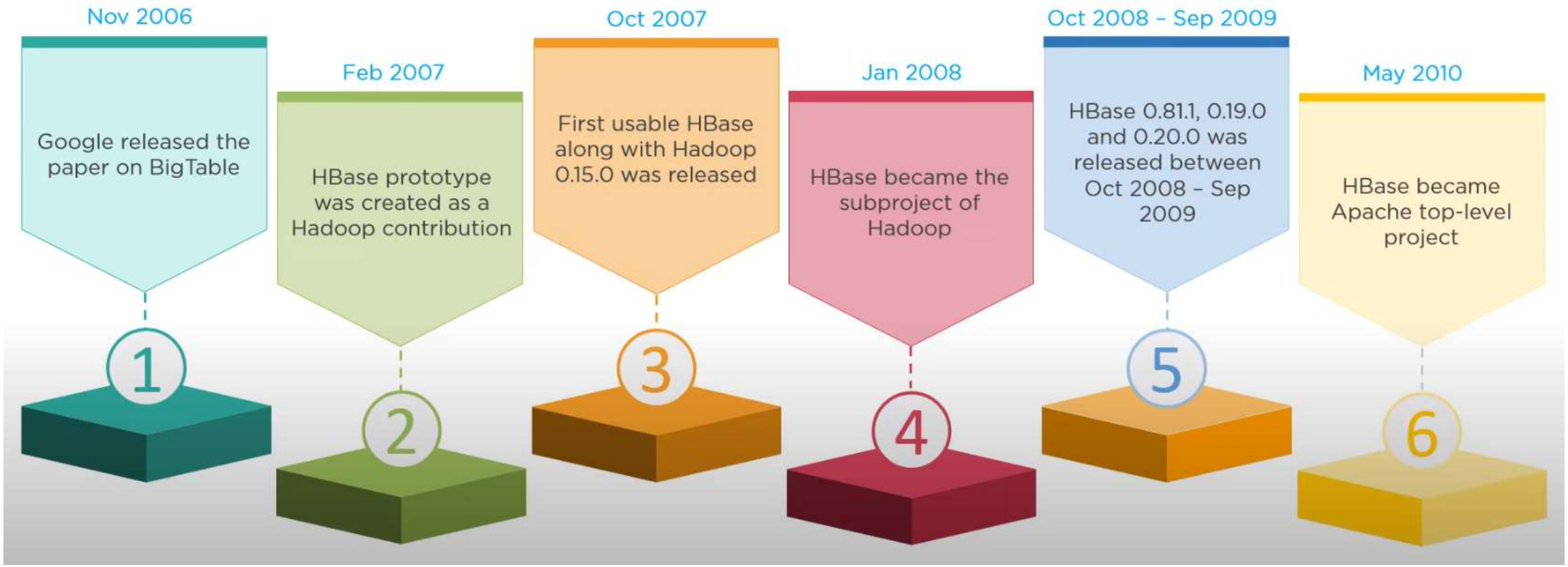


Introduction to HBase



HBase History

HBase History



What is HBase?

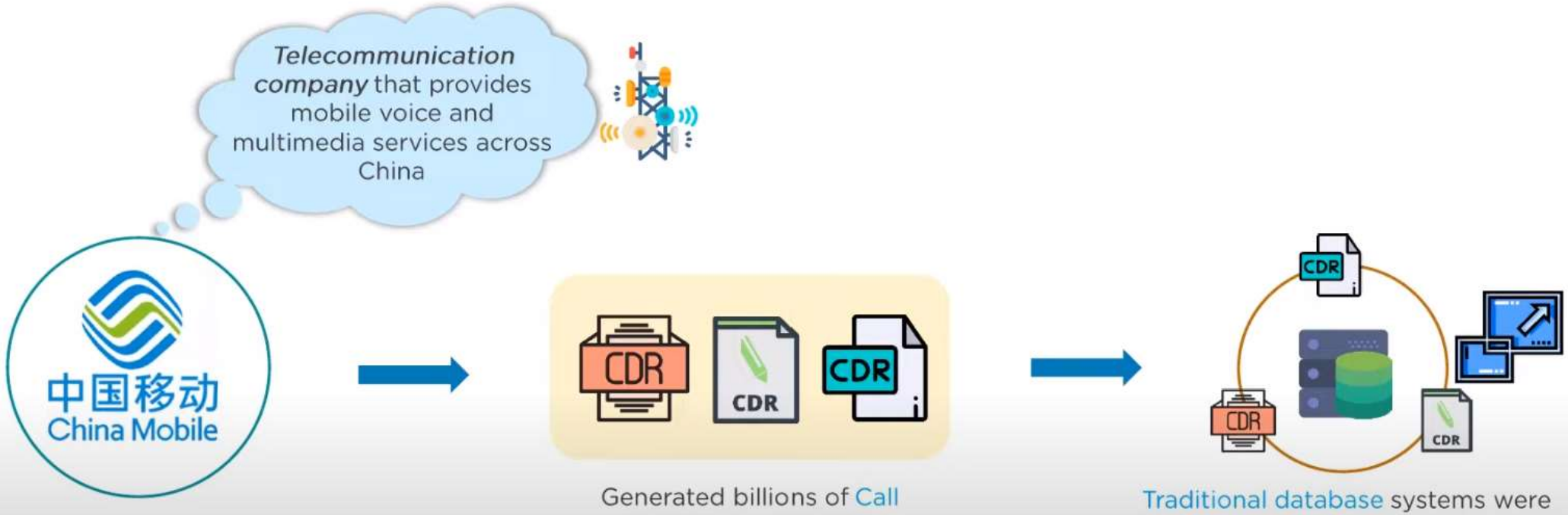
HBase is a column oriented database management system derived from Google's NoSQL database BigTable that runs on top of HDFS

APACHE HBASE

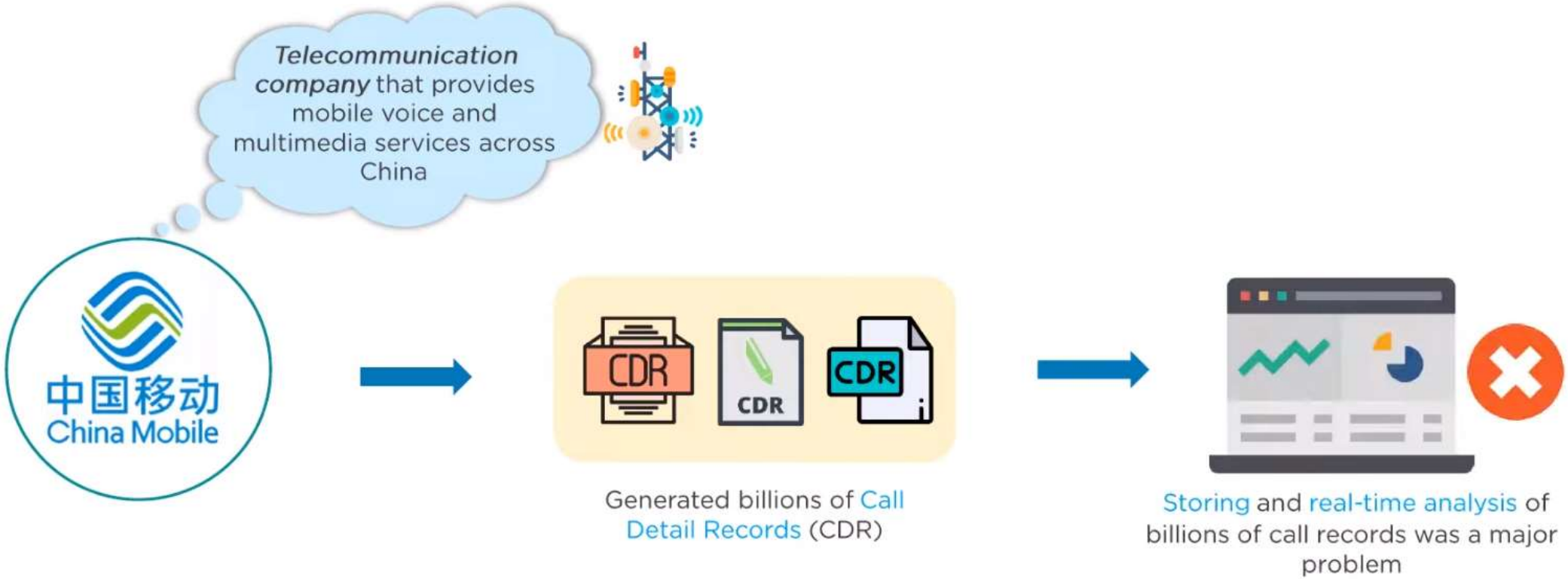
1. Open source project that is horizontally scalable
2. NoSQL database written in JAVA which performs faster querying
3. Well suited for sparse data sets (can contain missing or NA values)

Companies using HBase

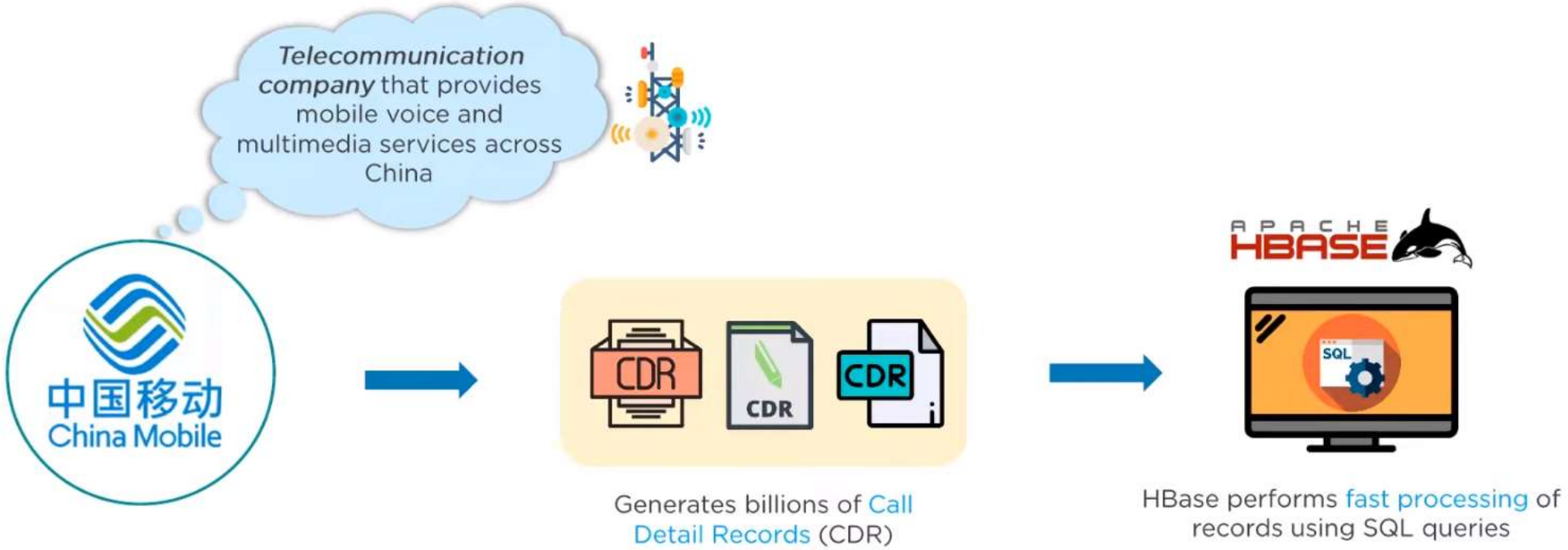
HBase Use Case



HBase Use Case



HBase Use Case



Applications of HBase

Medical

HBase is used for storing genome sequences

Storing disease history of people or an area

E-Commerce

HBase is used for storing logs about customer search history

Performs analytics and target advertisement for better business insights

Sports

HBase stores match details and history of each match

Uses this data for better prediction

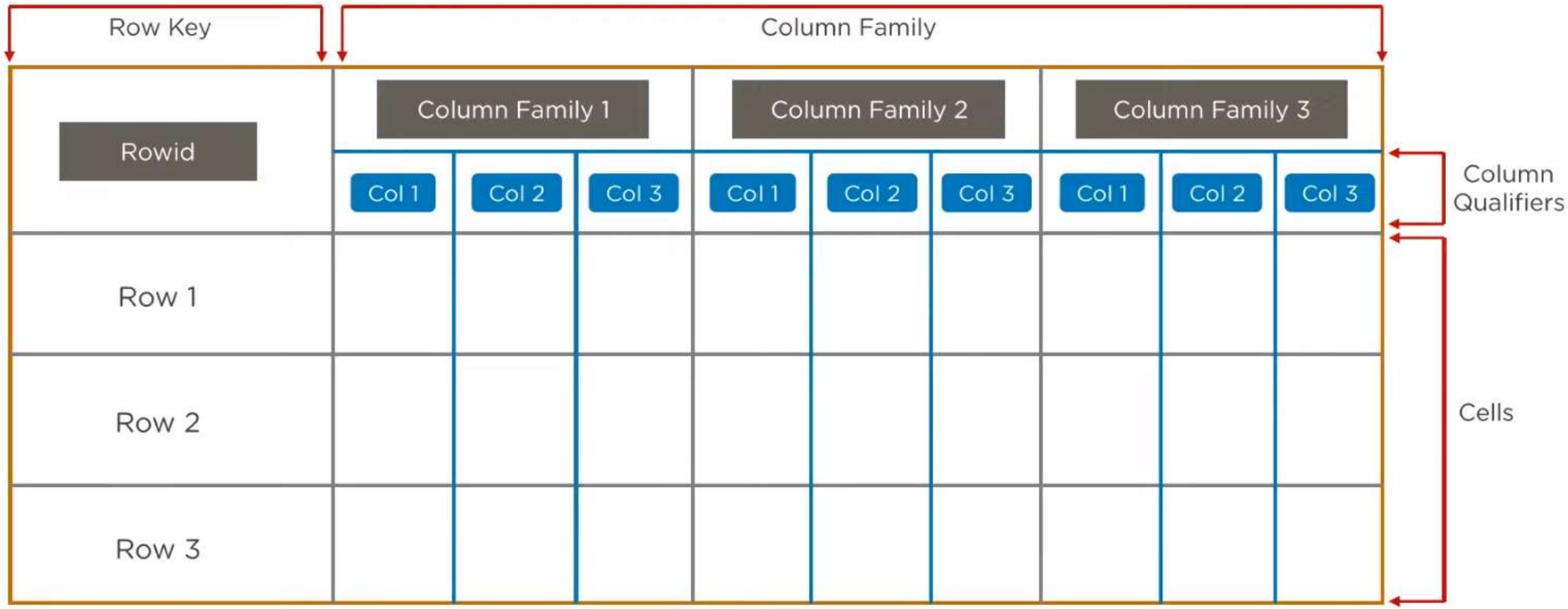
HBase vs RDBMS

HBase	RDBMS
Does not have a fixed schema (schema-less). Defines only column families	Has a fixed schema which describes the structure of the tables
Works well with structured and semi-structured data	Works well with structured data
It can have de-normalized data (can contain missing or NA values)	RDBMS can store only normalized data
Built for wide tables that can be scaled horizontally	Built for thin tables that is hard to scale

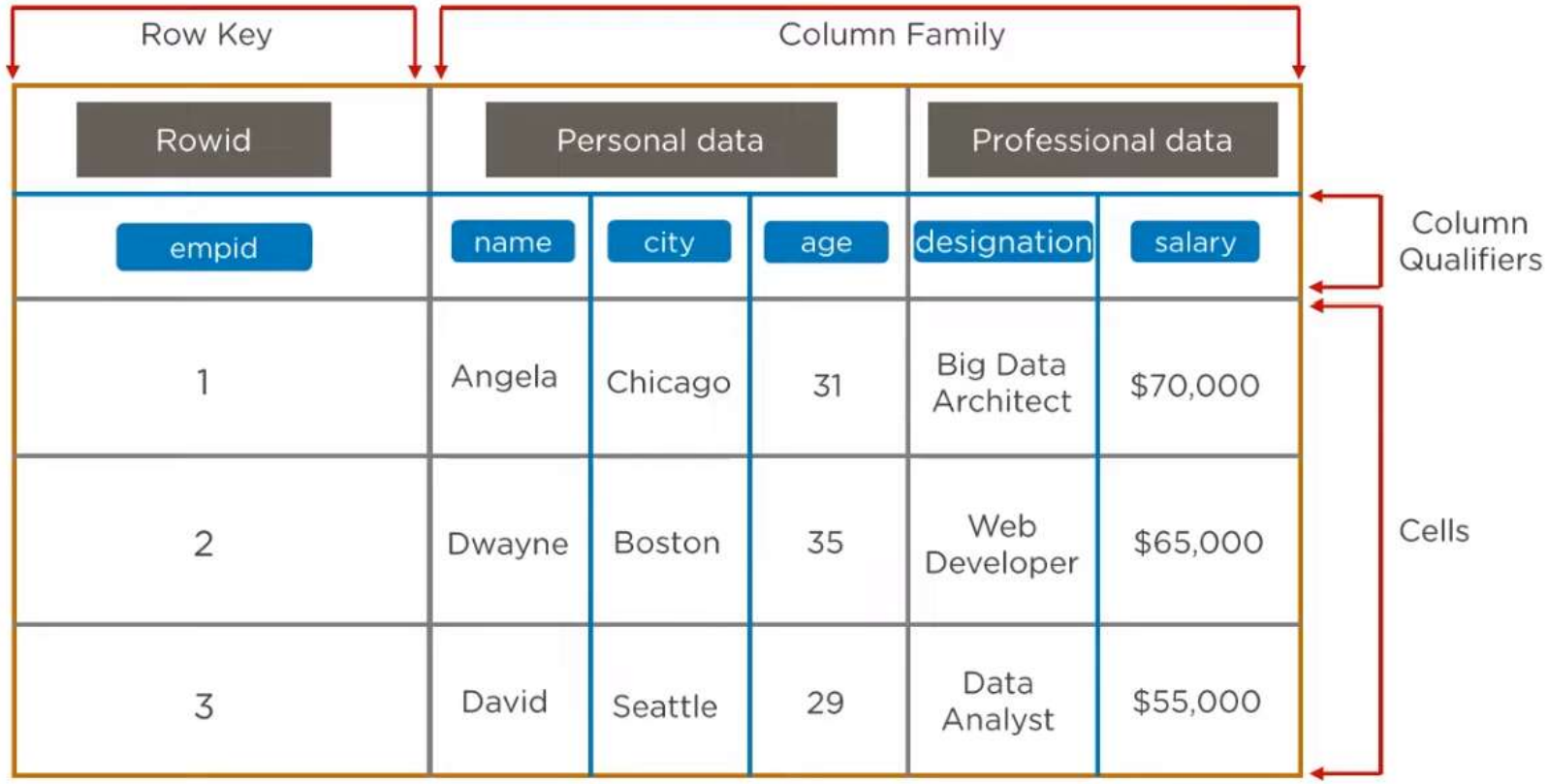
Features of HBase

Scalable	Automatic failure support	Consistent read and write	JAVA API for client access	Block cache and bloom filters
				
Data can be scaled across various nodes as it is stored in HDFS	Write Ahead Log across clusters which provides automatic support against failure	HBase provides consistent read and write of data	Provides easy to use JAVA API for clients	Supports block cache and bloom filters for high volume query optimization

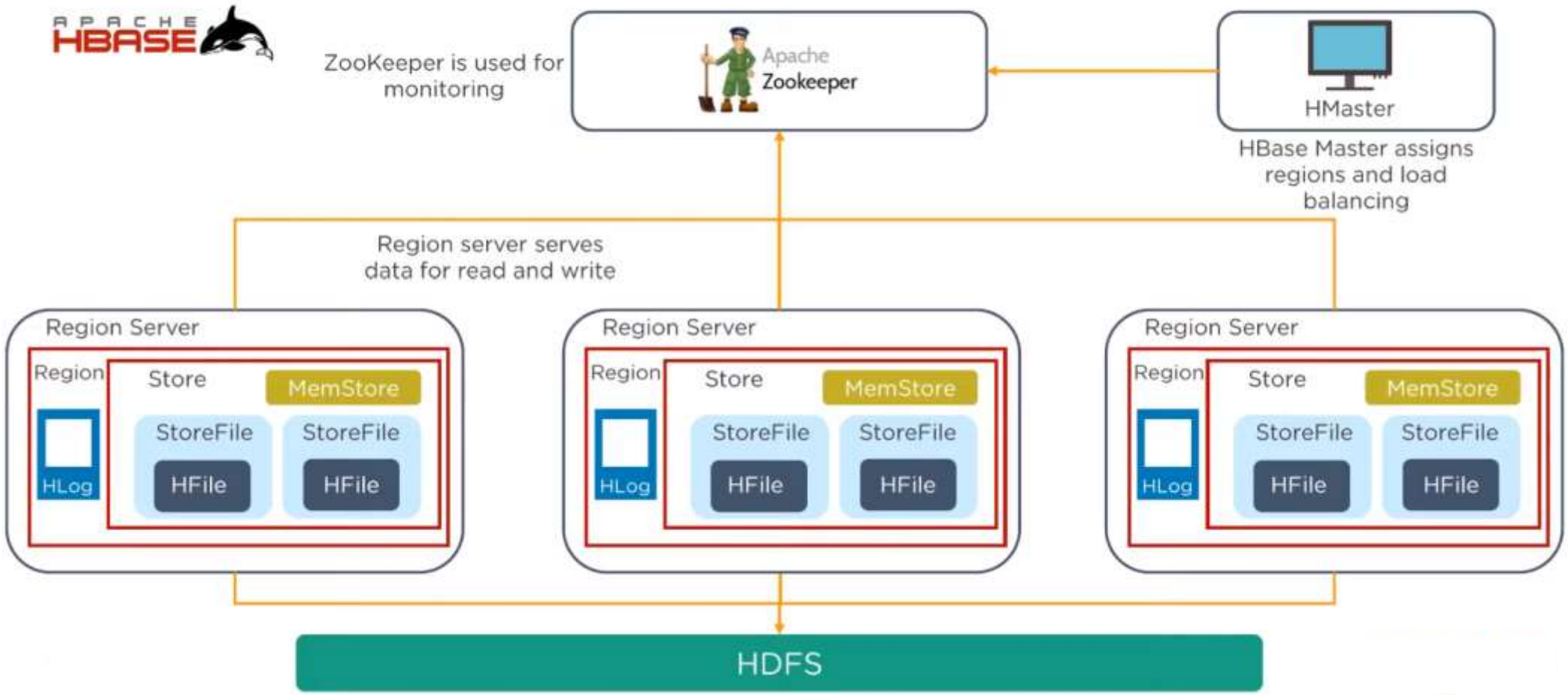
HBase column oriented storage



HBase column oriented storage



HBase Architectural Components



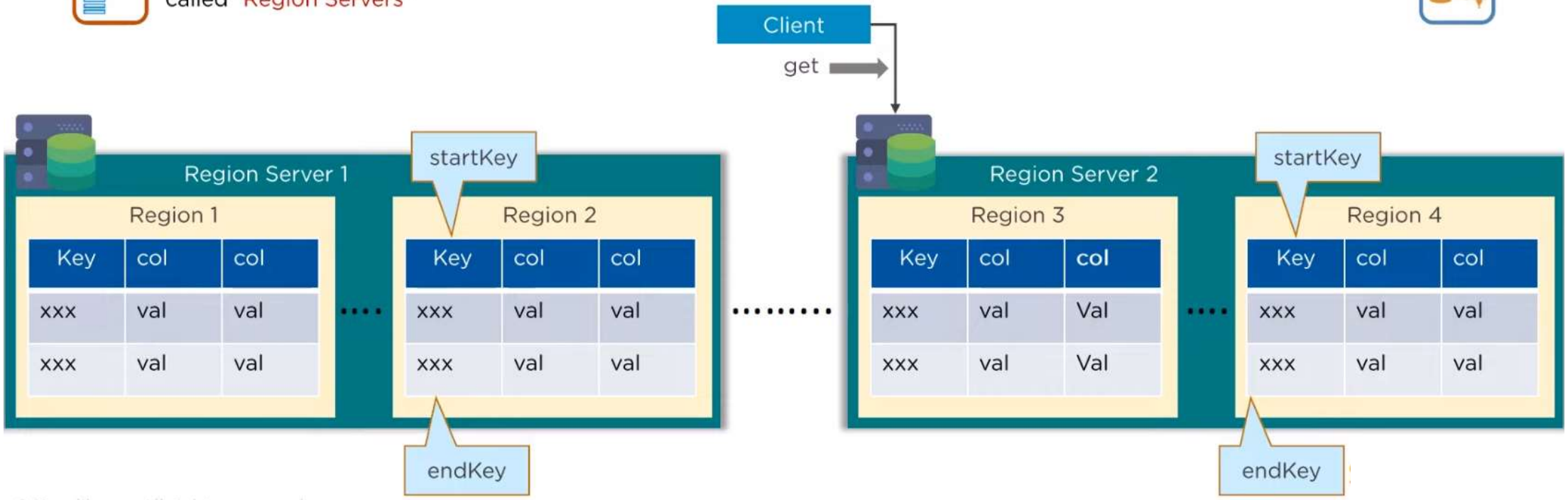
HBase Architectural Components - Regions

HBase tables are divided horizontally by row key range into "Regions"

A region contains all rows in the table between the region's start key and end key

Regions are assigned to the nodes in the cluster, called "Region Servers"

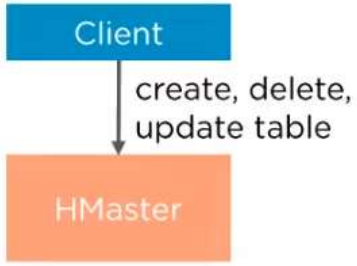
These servers serve data for read and write

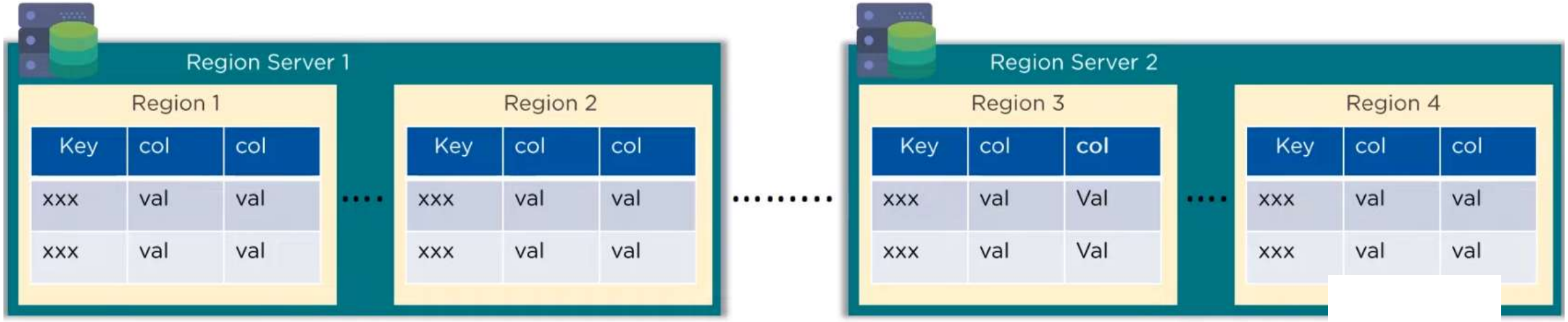


HBase Architectural Components - HMaster

Region assignment, Data Definition Language operation (create, delete) are handled by HMaster

Assigning and re-assigning regions for recovery or load balancing and monitoring all servers

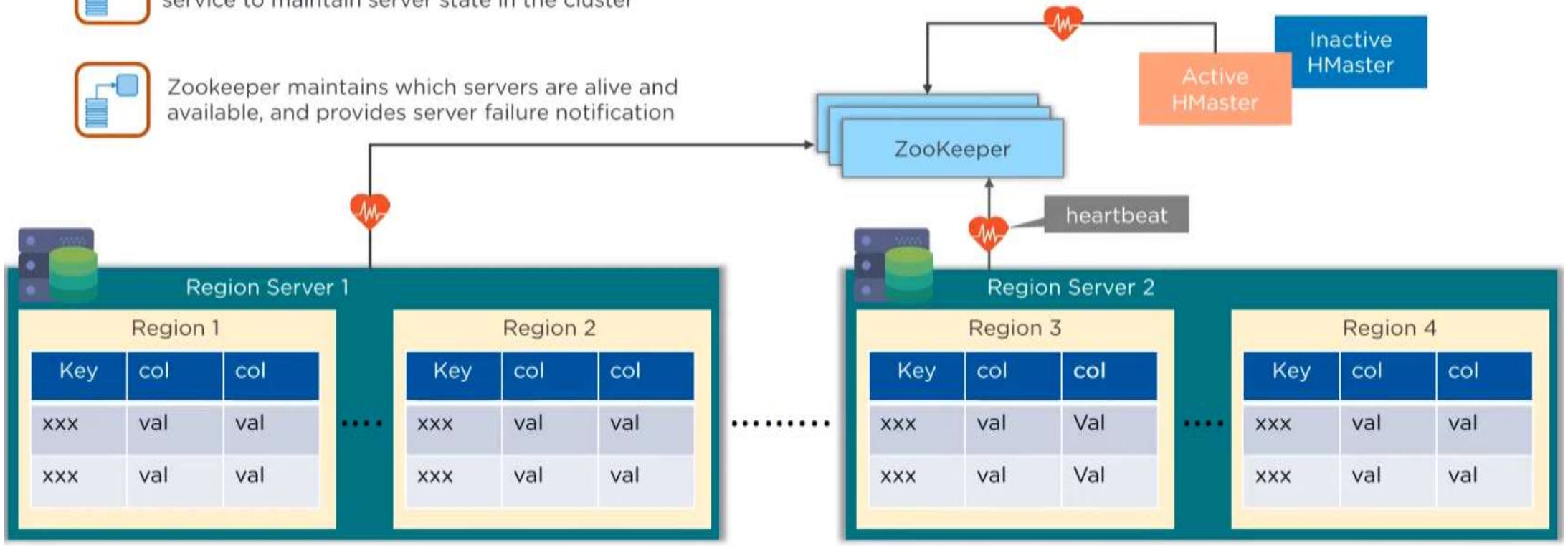




HBase Architectural Components - ZooKeeper

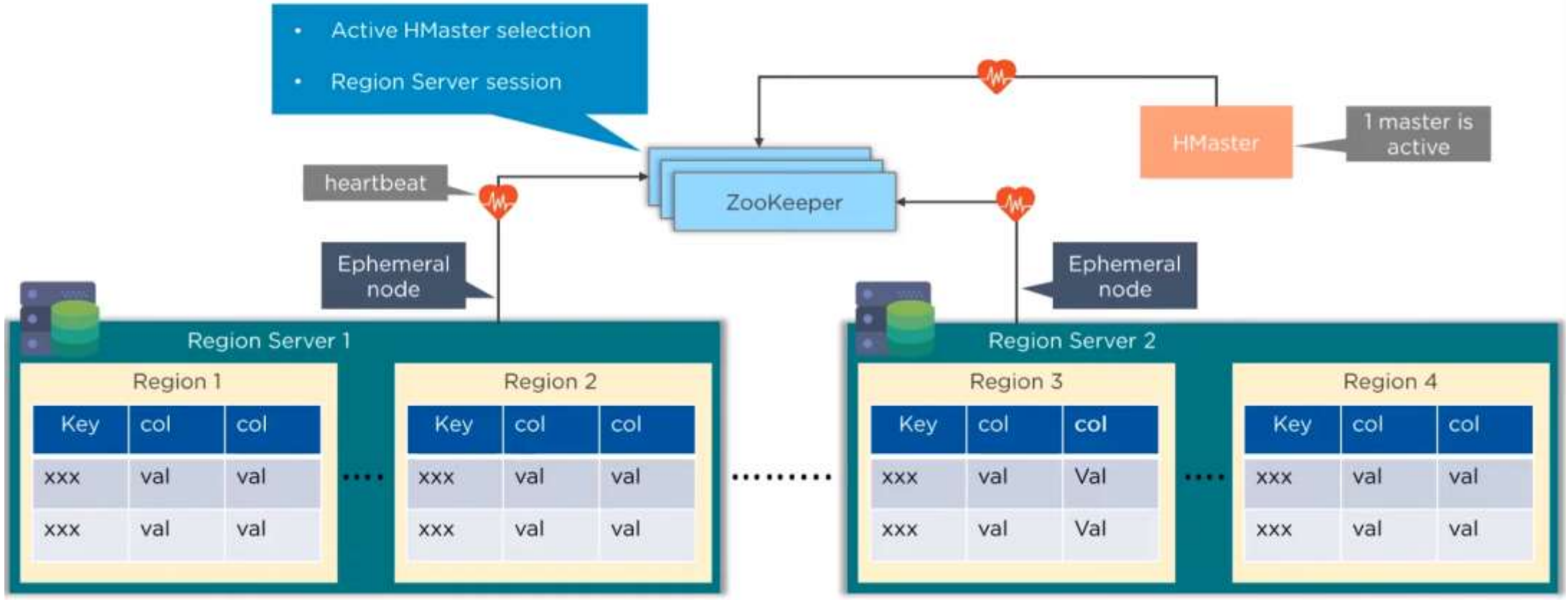
ZooKeeper is a distributed coordination service to maintain server state in the cluster

Zookeeper maintains which servers are alive and available, and provides server failure notification



Region servers send their status to ZooKeeper indicating they are ready for read and write operation

How the components work together?



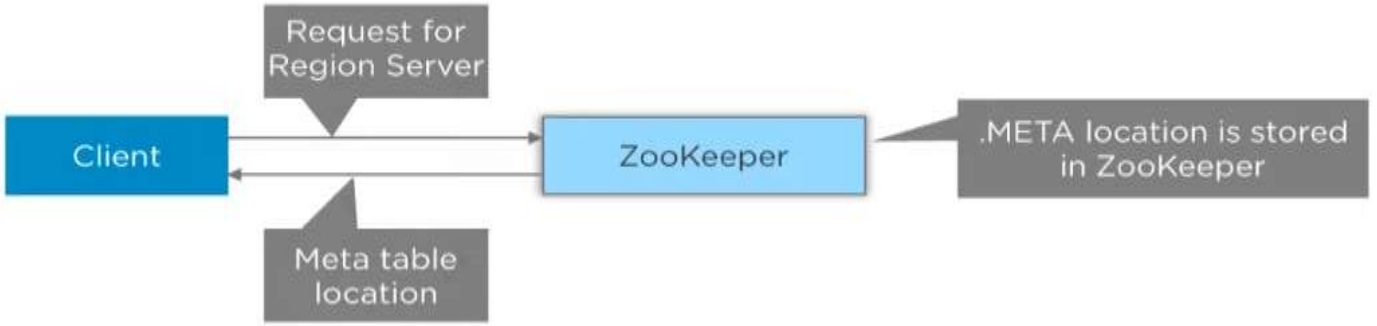
ZooKeeper maintains ephemeral nodes for active sessions via heartbeats to indicate that region servers are up and running

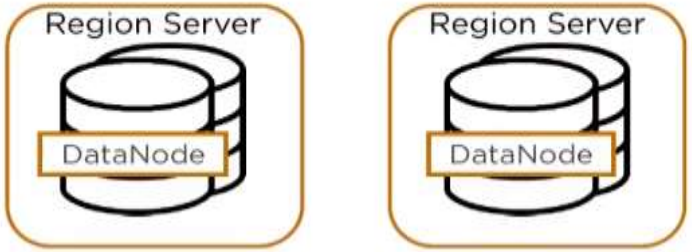
HBase Read or Write

There is a special HBase Catalog table called the META table, which holds the location of the regions in the cluster

Here is what happens the first time a client reads or writes data to HBase

The client gets the Region Server that hosts the META table from ZooKeeper

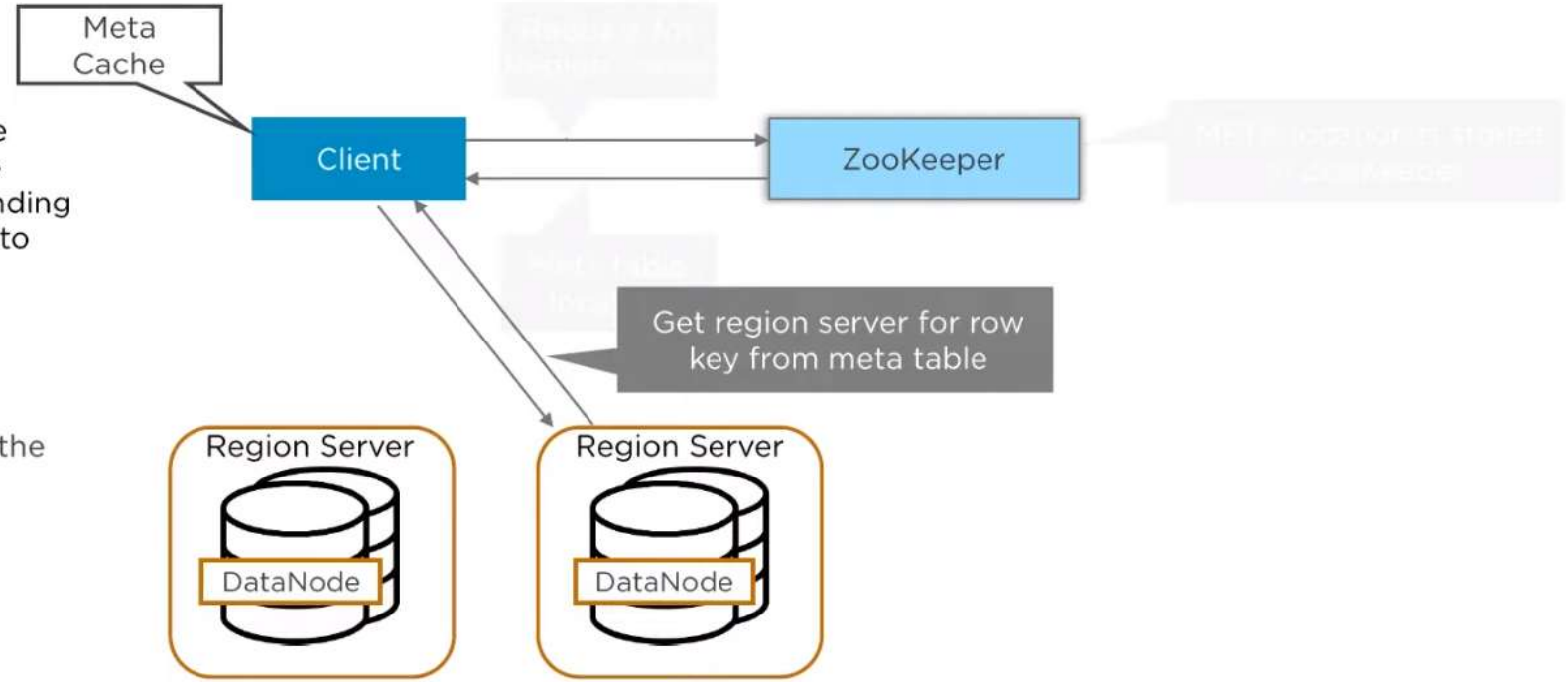




HBase Read or Write

There is a special HBase Catalog table called the META table, which holds the location of the regions in the cluster

Here is what happens the first time a client reads or writes data to HBase



The client will query the .META server to get the region server corresponding to the row key it wants to access

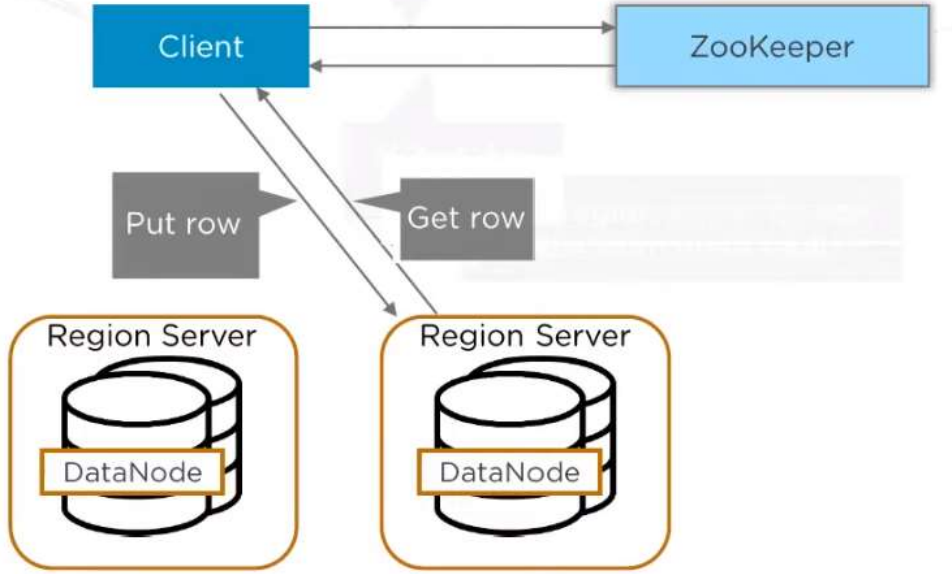
The client caches this information along with the META table location

HBase Read or Write

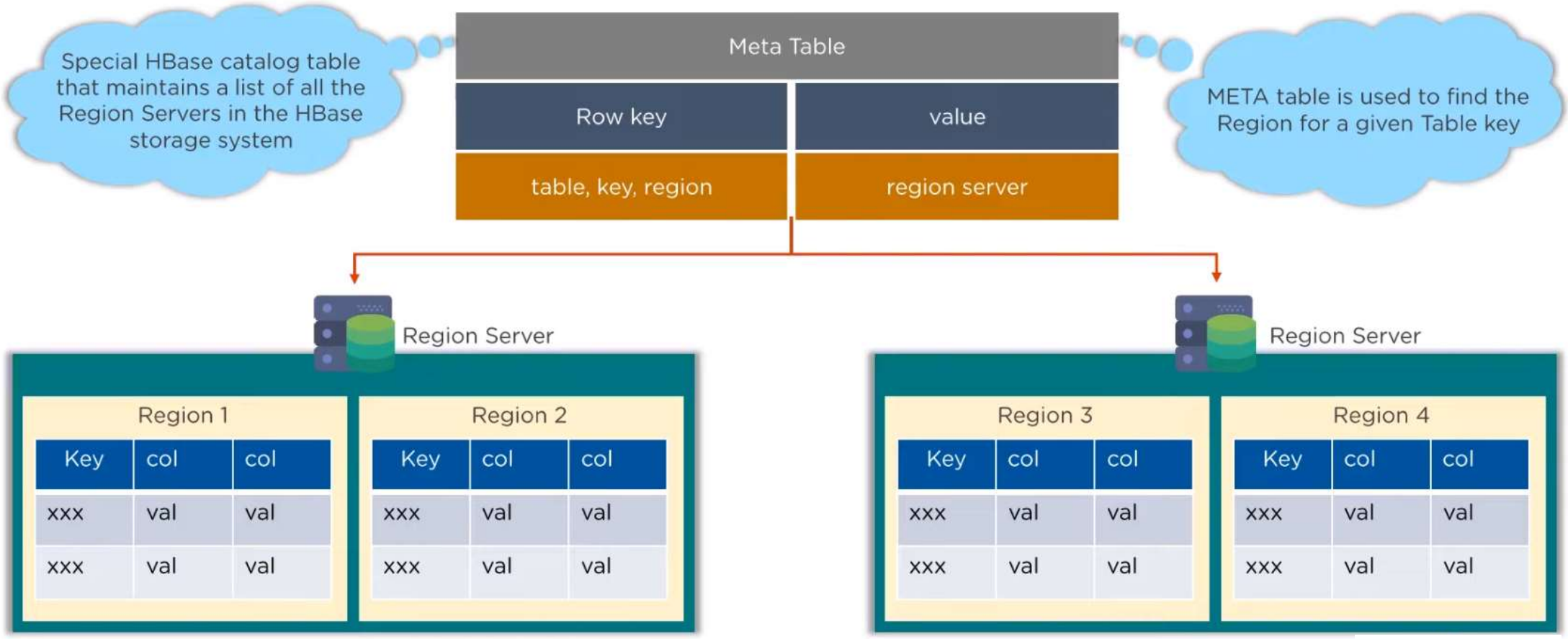
There is a special HBase Catalog table called the META table, which holds the location of the regions in the cluster

Here is what happens the first time a client reads or writes data to HBase

It will get the Row from the corresponding Region Server

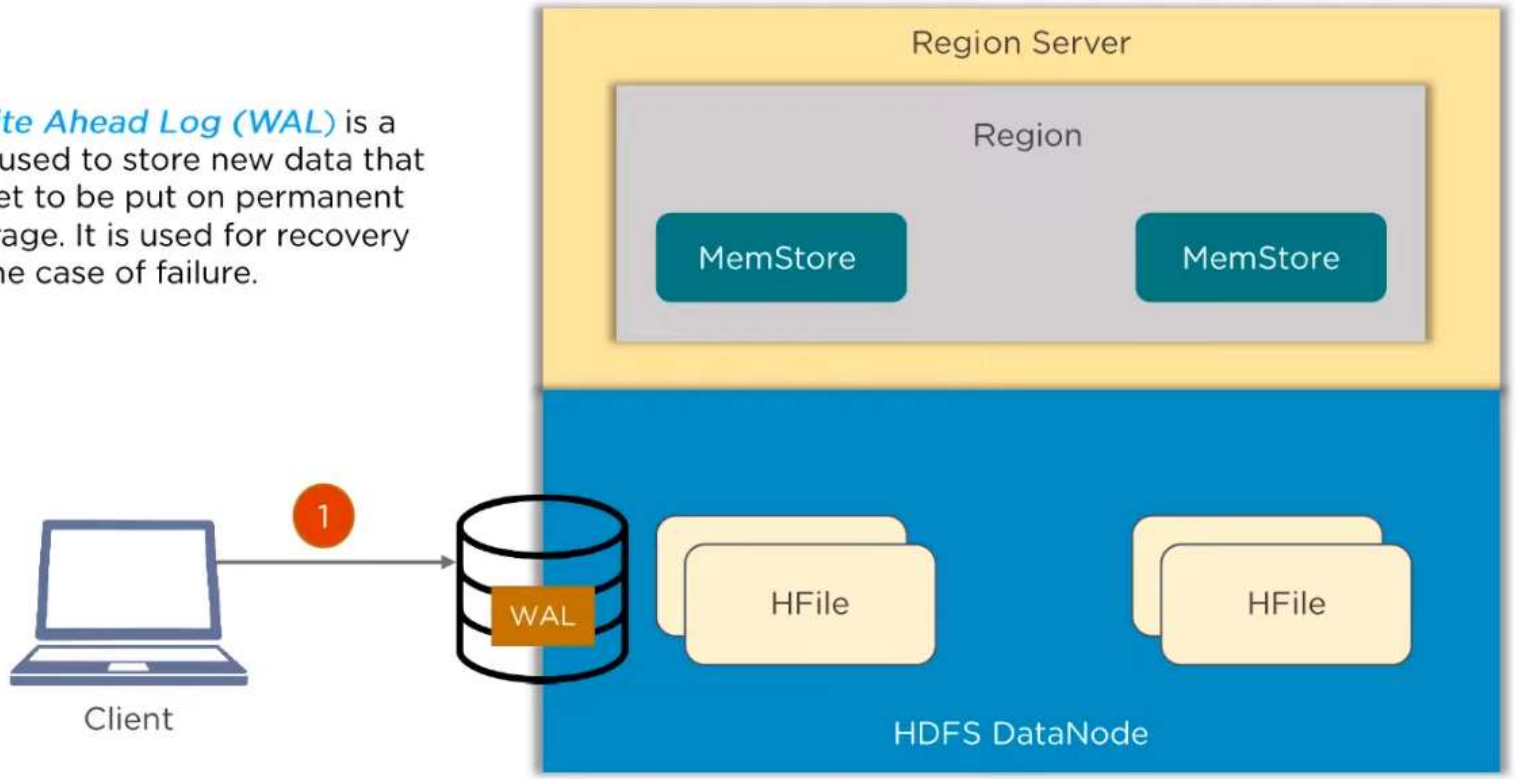


HBase Meta Table



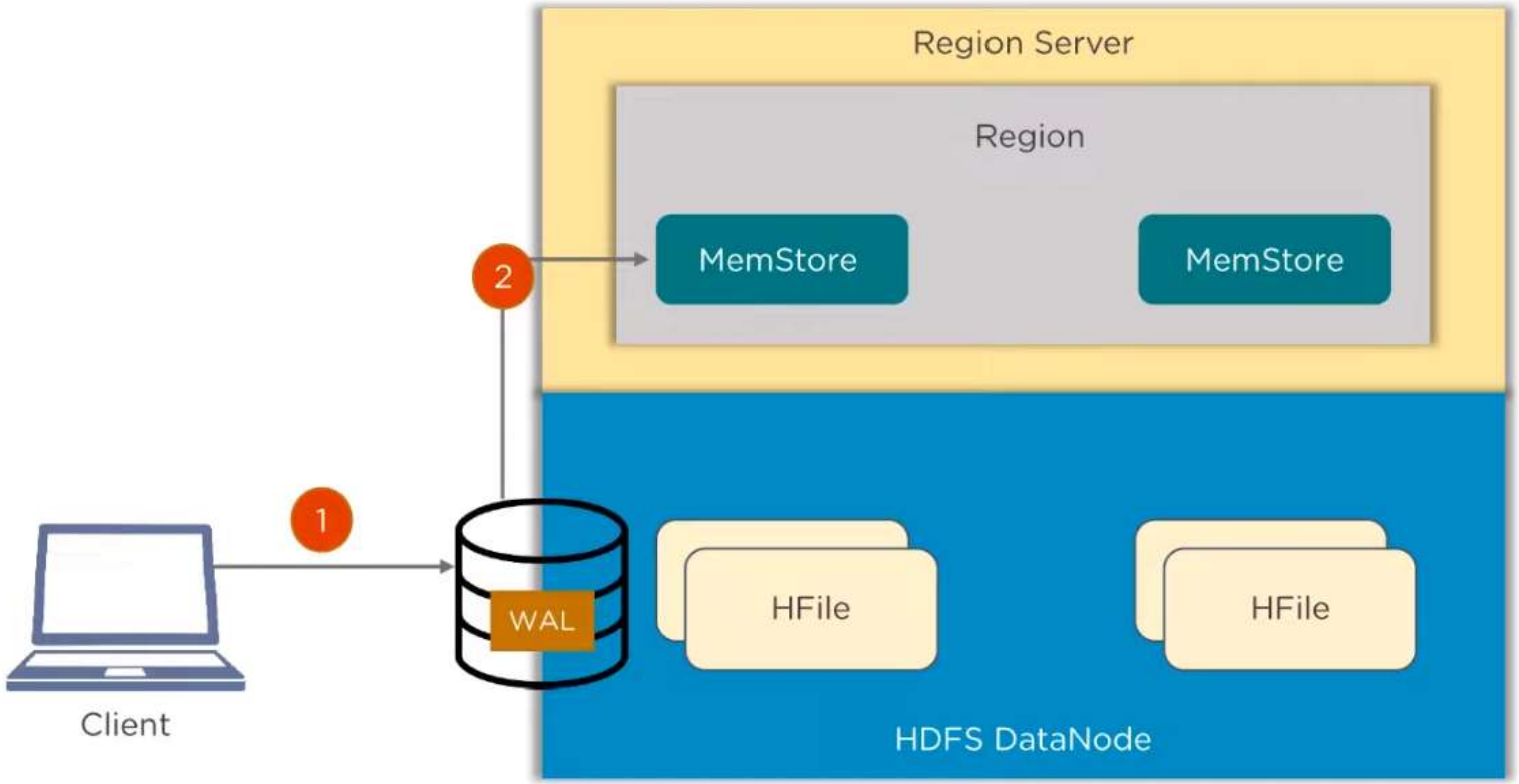
HBase Write Mechanism

Write Ahead Log (WAL) is a file used to store new data that is yet to be put on permanent storage. It is used for recovery is the case of failure.



1 When client issues a put request, it will write the data to the write-ahead log (WAL)

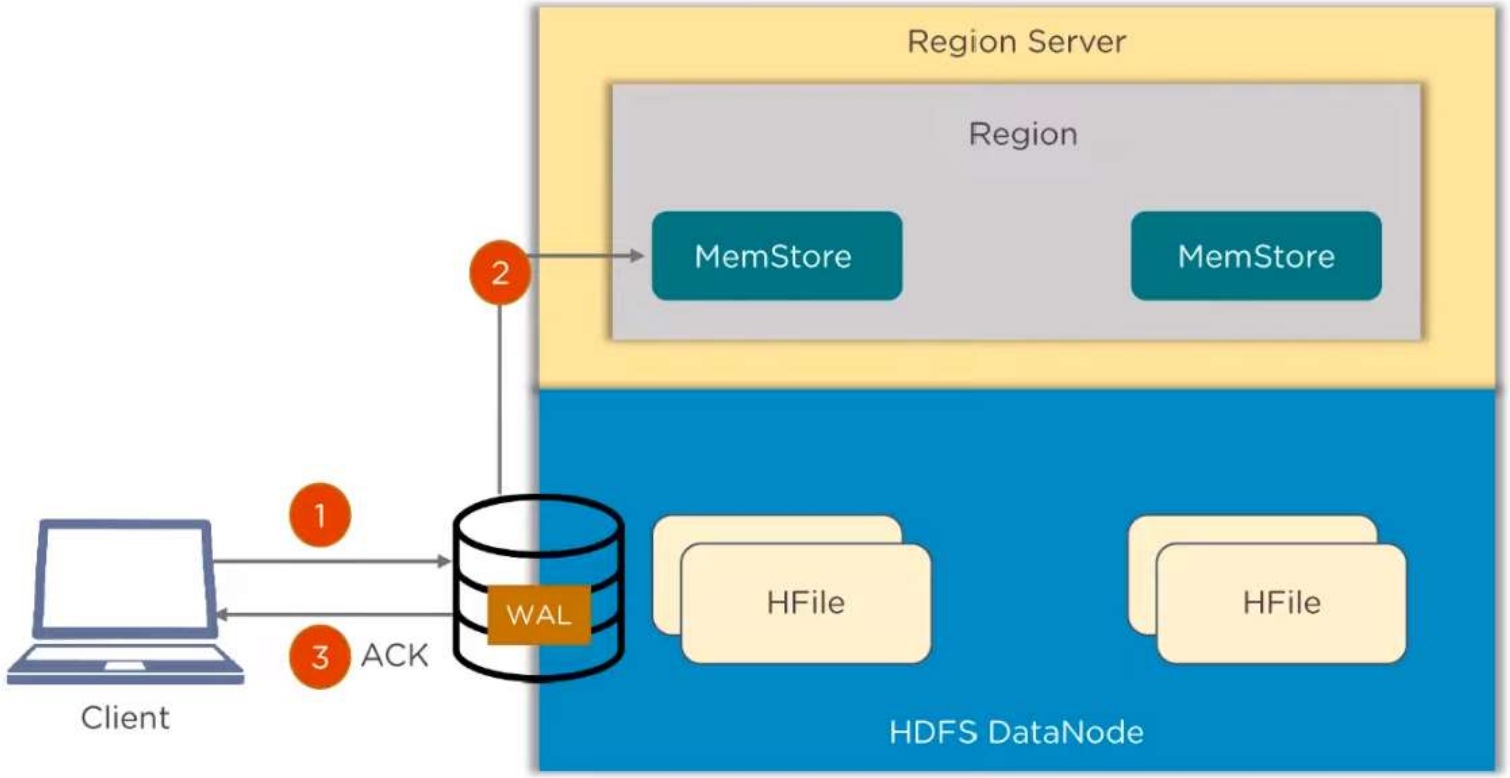
HBase Write Mechanism



MemStore is the write cache that stores new data that has not yet been written to disk. There is one MemStore per column family per region.

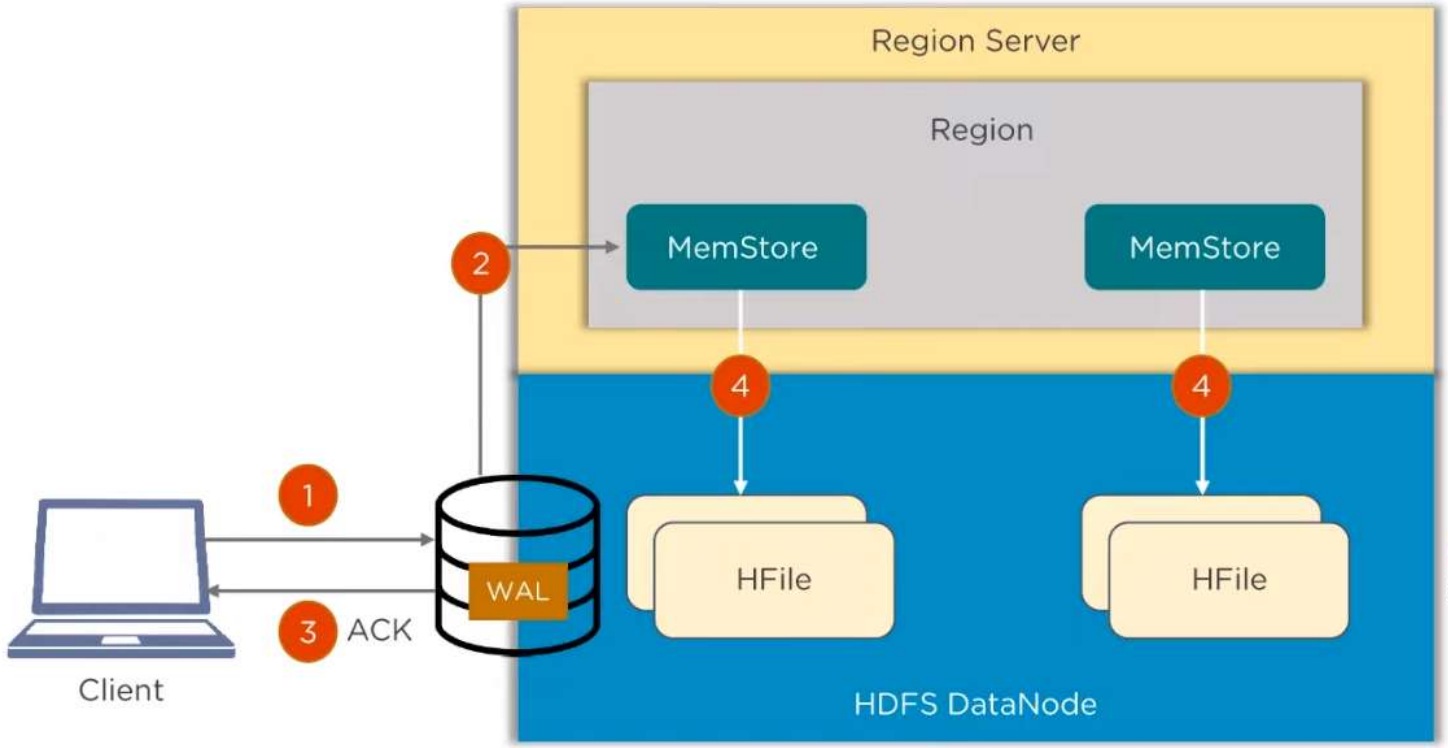
2 Once data is written to the WAL, it is then copied to the MemStore

HBase Write Mechanism



3 Once the data is placed in MemStore, the client then receives the acknowledgment

HBase Write Mechanism



Hfiles store the rows of data as sorted KeyValue on disk

4 When the MemStore reaches the threshold, it dumps or commits the data into a HFile

Commands	description
create	create a table in database
put	add a record in a table
get	retrieve a record from a table
Scan	retrieve a set of records from a table
delete, deleteall	delete a entire row or a column, or a cell from a table
alter	Alter a table (add or delete column family)
describe	Describe the named table
list	List all tables in database
disable/enable	Disable/enable the named table
drop	Drop the named table