upGrad

NoSQL

Databases and Apache HBase

Course: Data Engineering- I

Lecture On: NoSQL Databases & Apache HBase

Session 3

How HBase works?

Segment 2

HBase Architecture

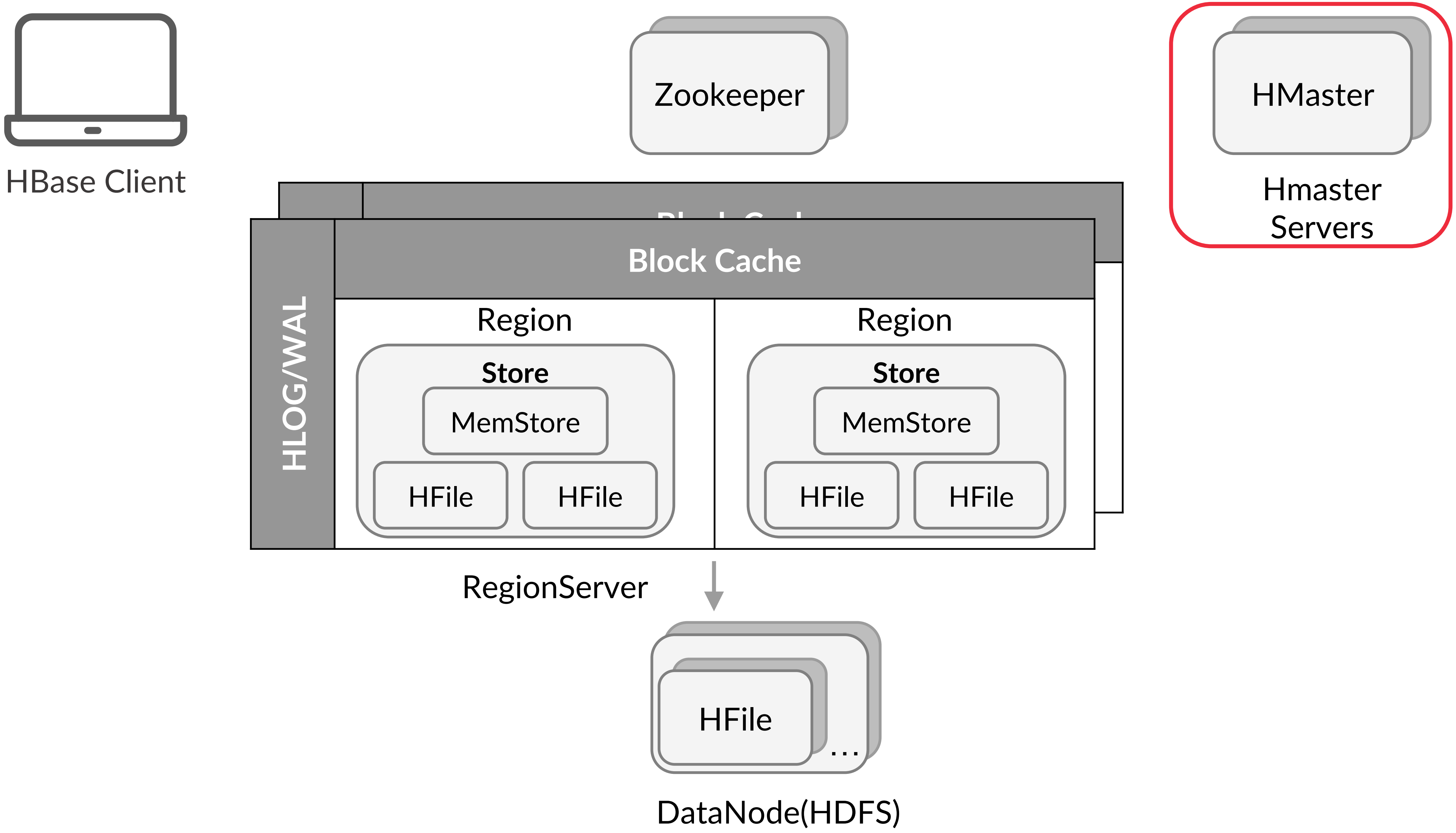
LEARNING OBJECTIVES

What are the components of HBase architecture?

What functions do each component perform?

How do components interact with each other?

HBASE ARCHITECTURE



HBASE ARCHITECTURE: HMASTER

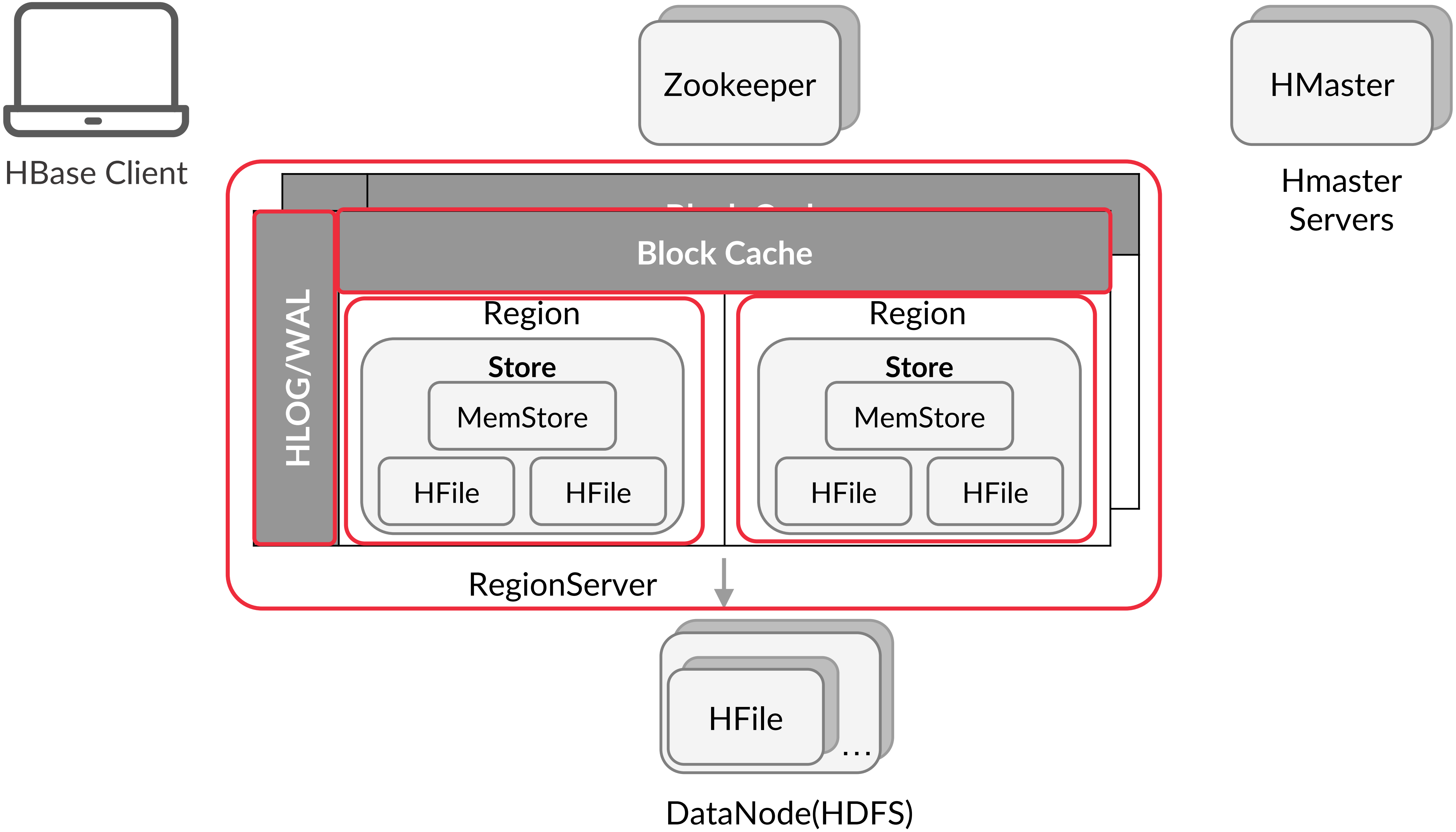
- Acts as the master server
- Manages multiple region servers
- An HBase cluster may have one or more master nodes
- Only one HMaster is active at a given time

HBASE ARCHITECTURE: HMASTER

This active master node is responsible for:

- Performing the admin functionalities
- Coordinating with region server
    - **Load Balancing:** When there is a high load on any region server, HMaster unloads the busy server and reassigns this load to less occupied servers.
    - **Recovery:** HMaster also handles region server failures by reassigning the regions or load of the failed region server to another functioning region server.

HBASE ARCHITECTURE: REGION SERVERS



HBASE ARCHITECTURE: REGION SERVERS

- It is the Slave server.
- There are multiple region servers in a cluster.
- Each region server contains:
  - **WAL** (Write Ahead Log), which stores new or updated data that has not been written to HDFS. The region server stores the WAL file in HDFS.

HBASE ARCHITECTURE: REGION SERVERS

- It is the Slave server.
- There are multiple region servers in a cluster.
- Each region server contains:
  - **WAL** (Write Ahead Log), which stores new or updated data that has not been written to HDFS. The region server stores the WAL file in HDFS.
  - **Block Cache,** which stores frequently read data from HDFS.

HBASE ARCHITECTURE: REGION SERVERS

- It is the Slave server.
- There are multiple region servers in a cluster.
- Each region server contains:
  - **WAL** (Write Ahead Log), which stores new or updated data that has not been written to HDFS. The region server stores the WAL file in HDFS.
  - **Block Cache,** which stores frequently read data from HDFS.
  - **Region**, which contains a Store per column family. The Store consists of a Memstore.

HBASE TABLE

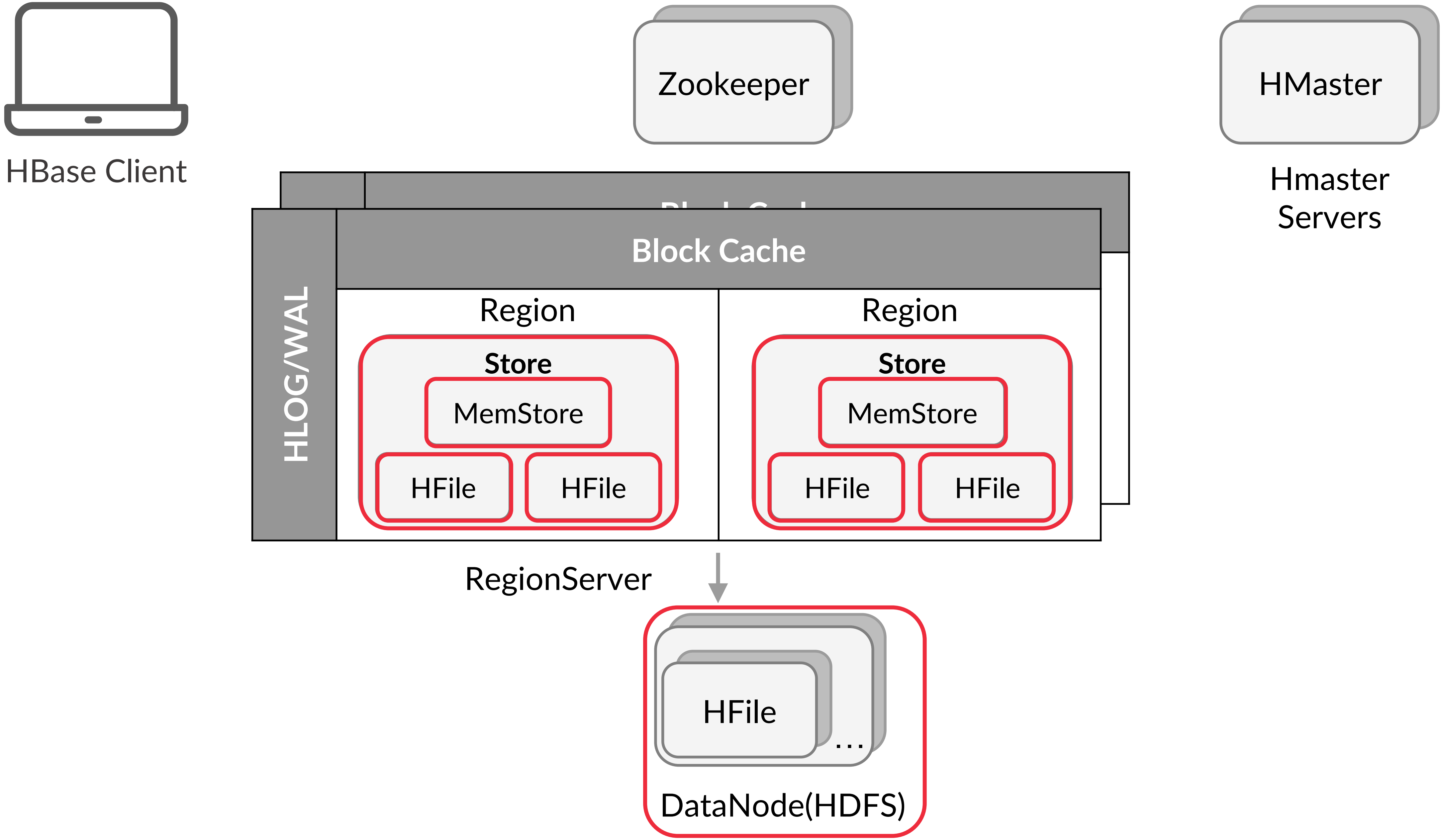
Lexicographic Order

Row key	A		B	
	Col-1	Col-2	Col-1	Col-2
Row1	2	4	6	4
Row11	1	6	5	8
Row2	0	9	3	5
	5	1	0	8
Row30	6	2	4	0
Row4	2	5	7	9
Row6	6	3	1	0

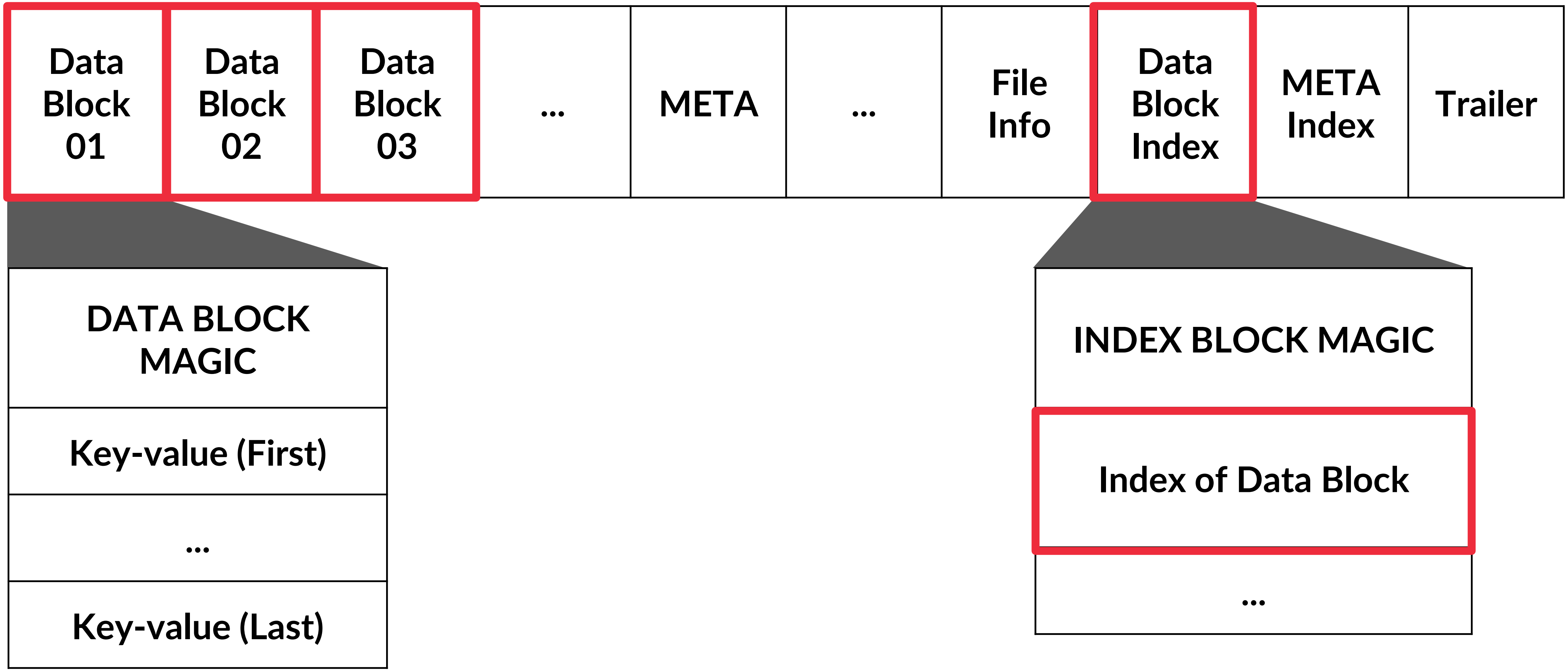
REGIONS

	Row key	A: Col-1	A: Col-2	B: Col-1	B: Col-2
Region1	Row1	2	4	6	4
	Row11	1	6	5	8
Region2	Row2	0	9	3	5
		5	1	0	8
	Row30	6	2	4	0
Region3	Row4	2	5	7	9
	Row6	6	3	1	0

HBASE ARCHITECTURE



HFILE STRUCTURE



Data Block 01 | Data Block 02 | Data Block 03 | ... | META | ... | File Info | Data Block Index | META Index | Trailer

DATA BLOCK MAGIC

Key-value (First)

...

Key-value (Last)

INDEX BLOCK MAGIC

Index of Data Block

...

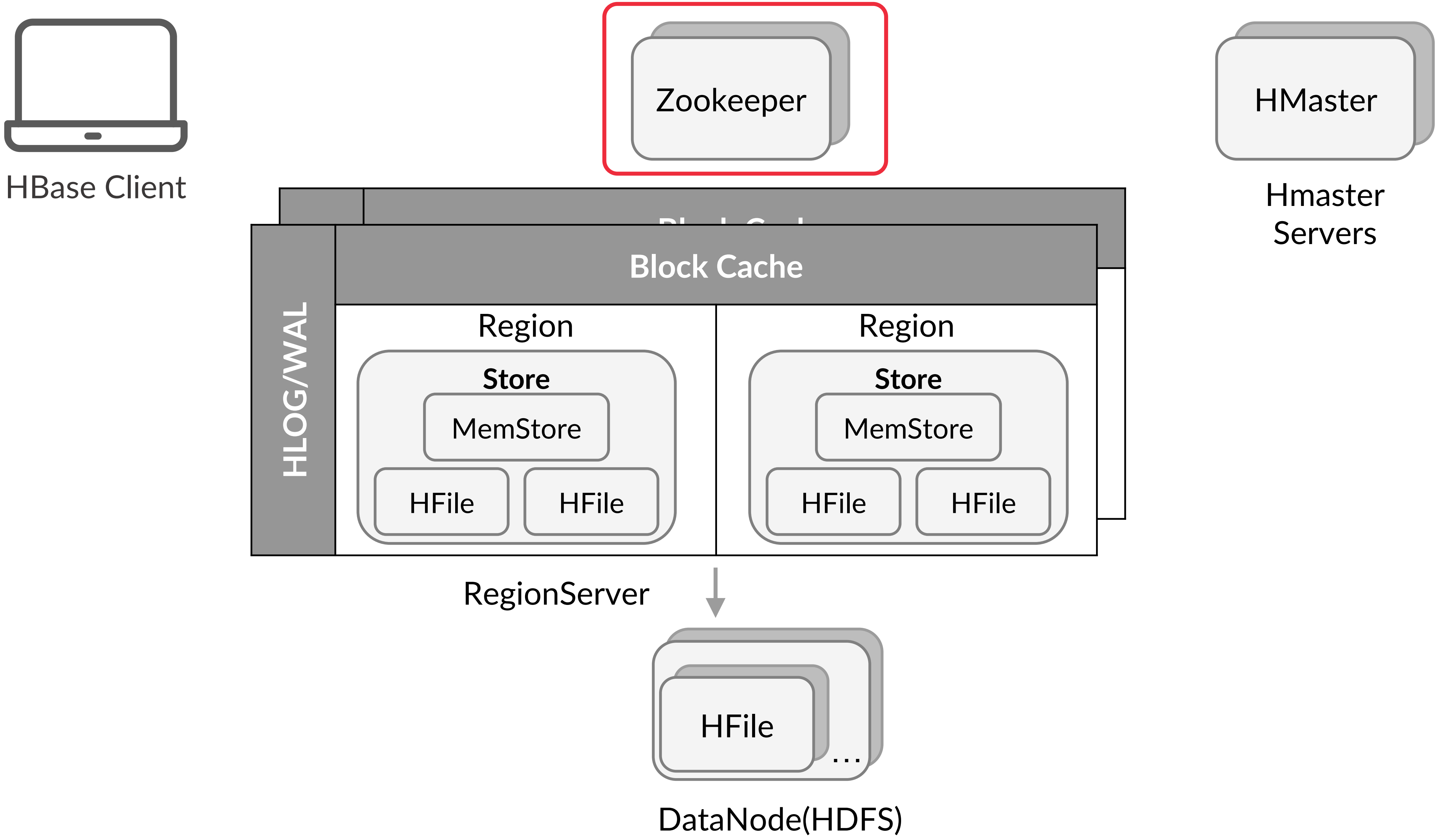
B+ TREES

- These are data structures used to store large amounts of data which cannot be stored in the main memory.
- The Internal nodes are stored in the main memory.
- The Leaf nodes are stored in the secondary memory.

HBASE ARCHITECTURE: REGION SERVERS

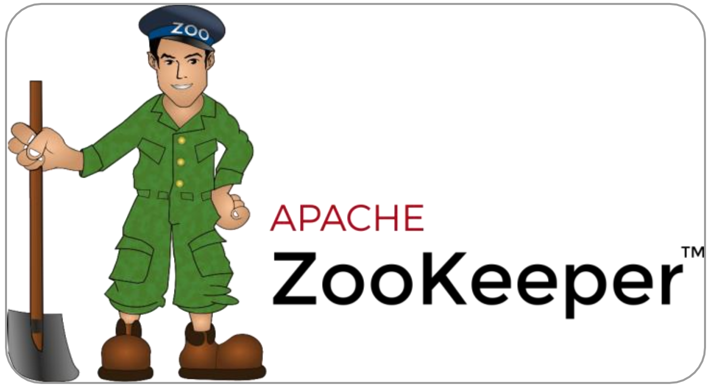
- It is the Slave server.
- It manages regions. There are multiple region servers in a cluster.
- Each region server contains:
  - **WAL** (Write Ahead Log), which stores new or updated data that has not been written to HDFS. The region server stores the WAL file in HDFS.
  - **Block Cache**, which stores frequently read data from HDFS.
  - **Region,** which contains one Store per column family. The Store consists of a Memstore.
- A client talks directly with the region server to perform read/write operations.
- This region server assigns the request to a specific region.

HBASE ARCHITECTURE: ZOOKEEPER

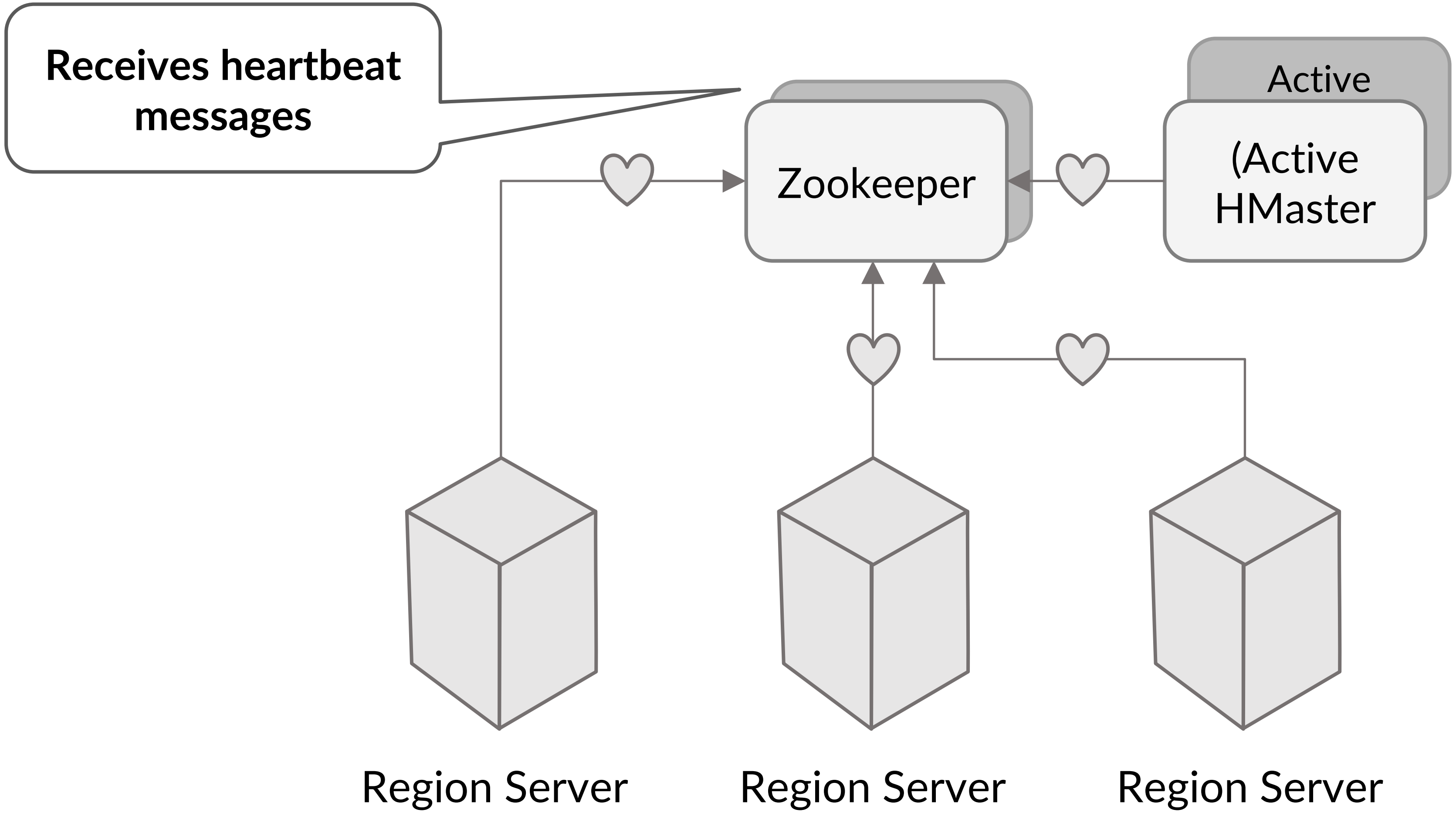


HBASE ARCHITECTURE: ZOOKEEPER

- It is a distributed open-source coordinating service for distributed applications.
- It maintains the configuration information of clusters and naming services of clusters, in addition to providing distributed synchronisation, group services, etc.
- HBase uses Zookeeper to maintain the live server state in the cluster.



ZOOKEEPER IN HBASE CLUSTER



Receives heartbeat messages

Zookeeper

Active

(Active HMaster

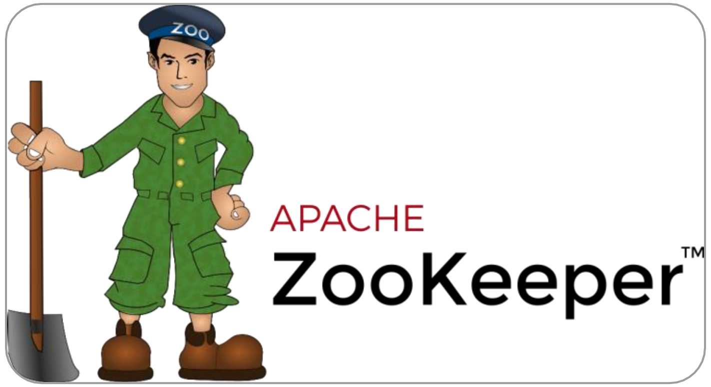
Region Server

Region Server

Region Server

HBASE ARCHITECTURE: ZOOKEEPER

- It is a distributed open-source coordinating service for distributed applications.
- It maintains the configuration information of clusters and naming services of clusters in addition to providing distributed synchronisation, group services, etc.
- HBase uses Zookeeper to maintain the live server state in the cluster.
- It provides server failure notifications.
- It stores the location of the META table.



KEY TAKEAWAYS

- HBase architecture is based on the concept of master-slave architecture.
- There are three servers: HMaster, Region Servers and Zookeeper.
- HMaster is the master server and manages the region servers.
- A region server acts as the slave server and manages multiple regions.
- A region has multiple stores, a Memstore and HFiles where actual data is present.
- Zookeeper is responsible for the communication between HMaster and region servers.

Thank You!

Segment 3

Read Operation in HBase

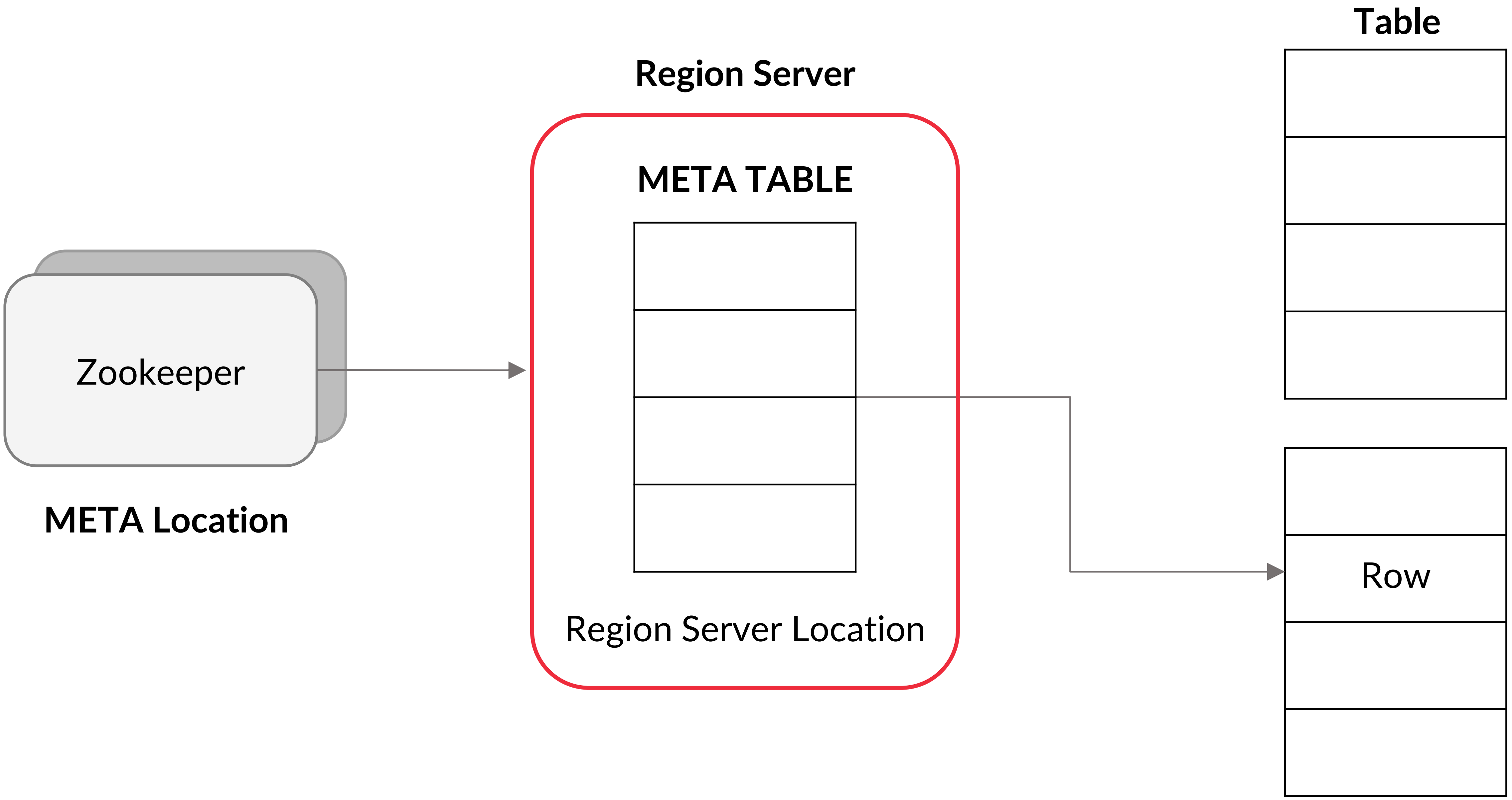
LEARNING OBJECTIVES

What is Meta Table and its structure?

How is the location of region server fetched by client?

How can a client perform read operation in HBase?

META TABLE



Table

Region Server

META TABLE

Zookeeper

META Location

Region Server Location

Row

META TABLE

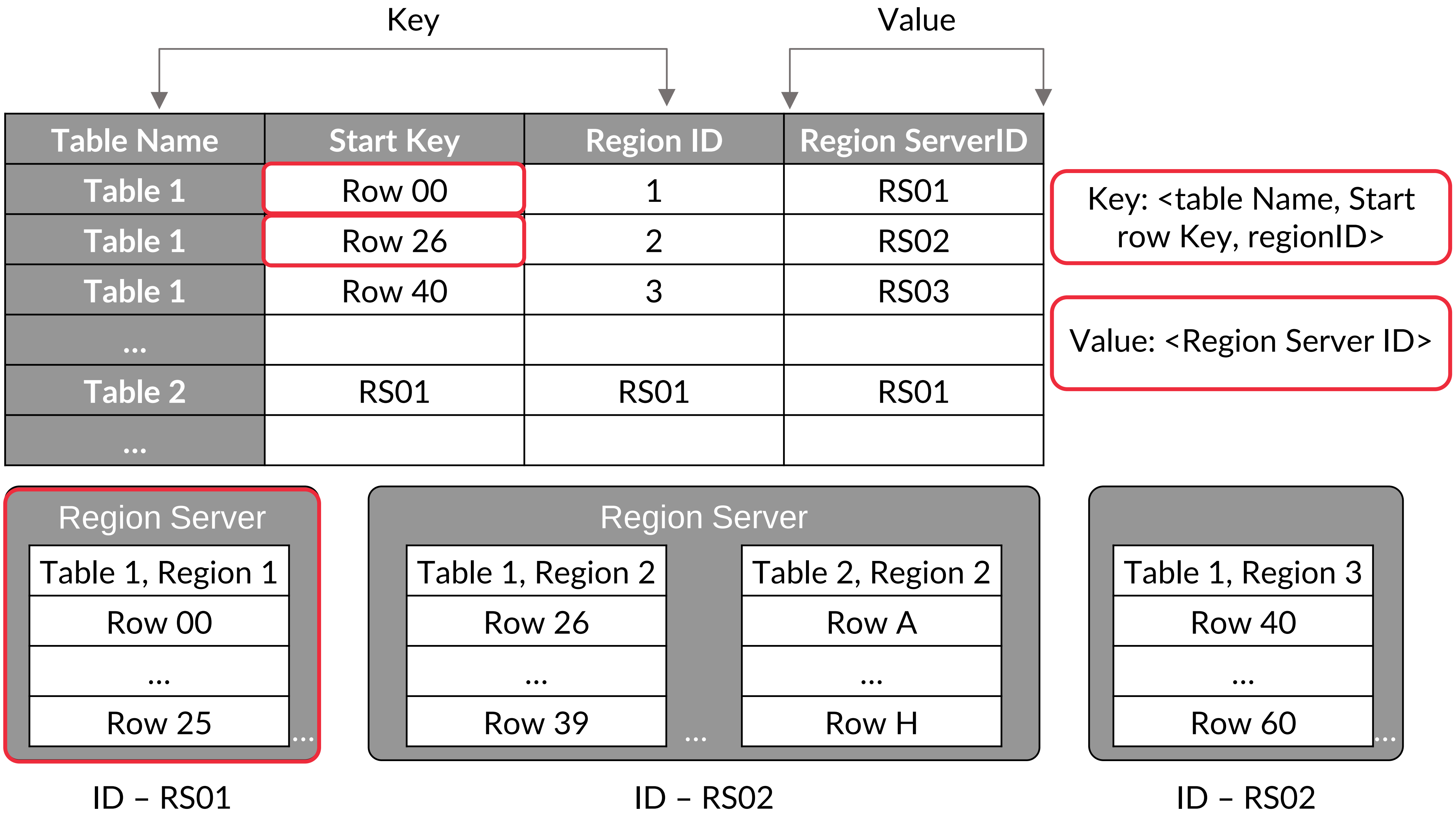


Table Name	Start Key	Region ID	Region ServerID
Table 1	Row 00	1	RS01
Table 1	Row 26	2	RS02
Table 1	Row 40	3	RS03
...			
Table 2	RS01	RS01	RS01
...			

Key: <table Name, Start row Key, regionID>

Value: <Region Server ID>

Region Server

Table 1, Region 1
Row 00
...
Row 25

...

ID – RS01

Region Server

Table 1, Region 2
Row 26
...
Row 39

...

Table 2, Region 2
Row A
...
Row H

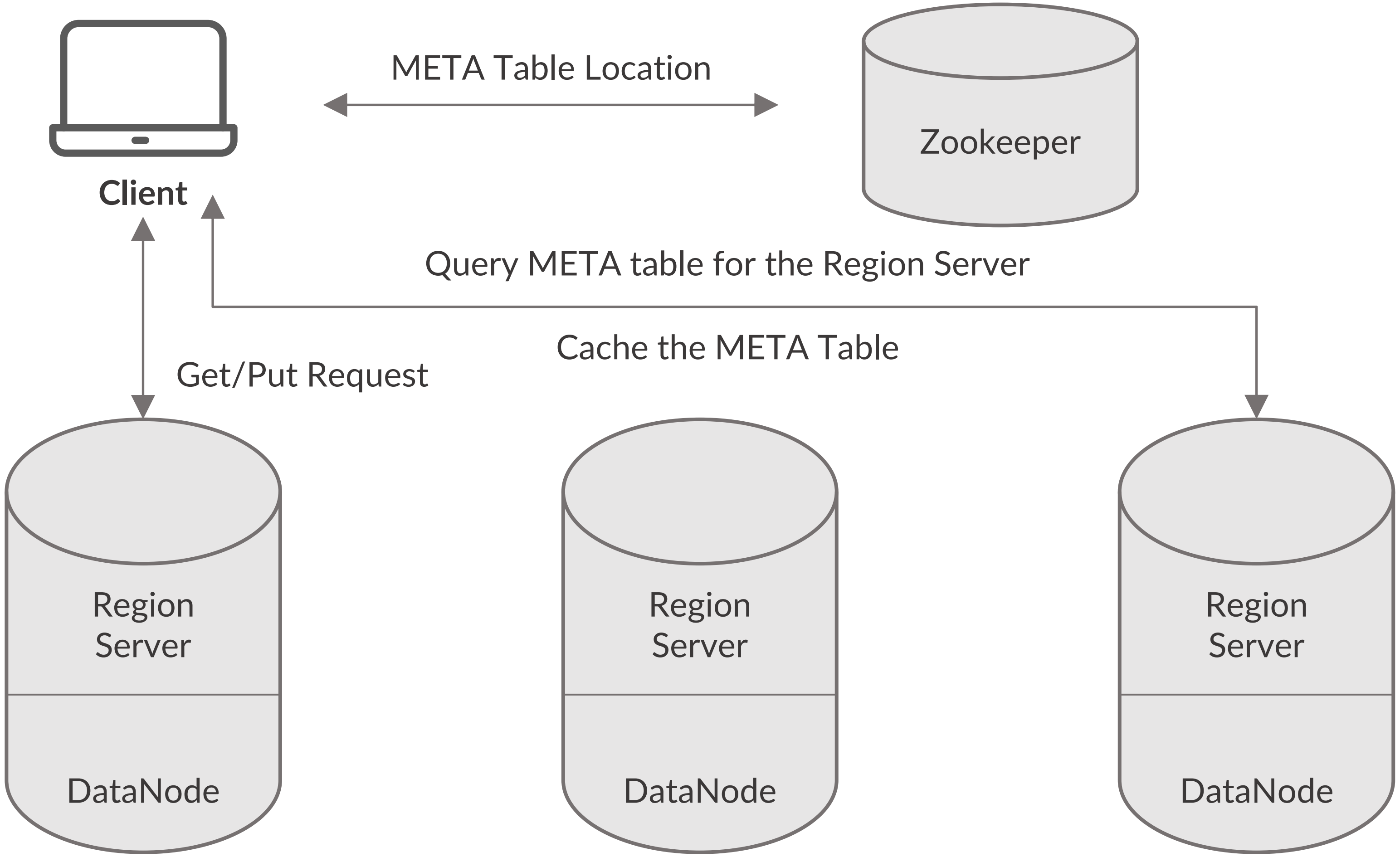
ID – RS02

Table 1, Region 3
Row 40
...
Row 60

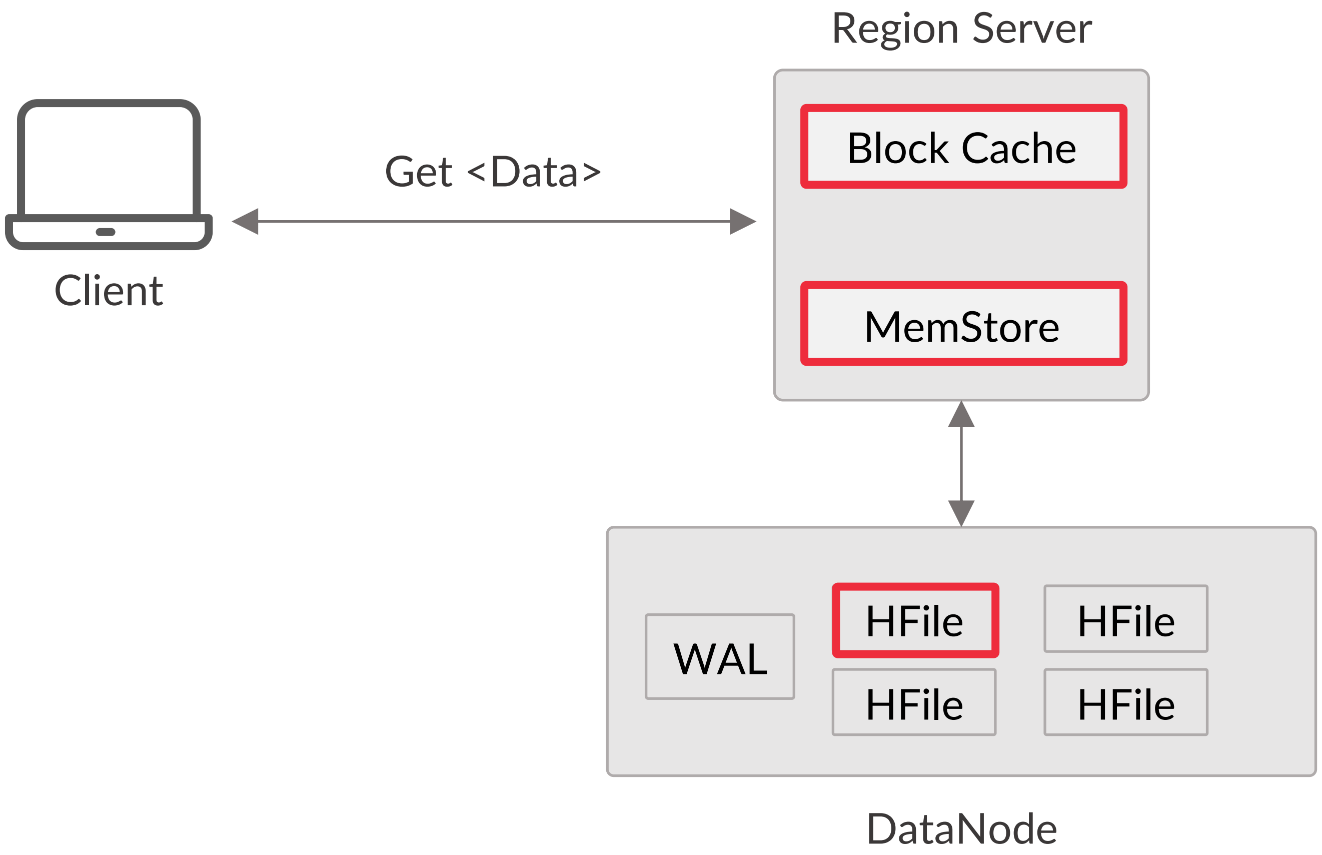
...

ID – RS02

FETCHING THE REGION SERVER



HBASE READ OPERATION



HBASE READ OPERATION

- The region server first checks its **Block Cache,** which stores the recently accessed data.

HBASE READ OPERATION

- The region server first checks its **Block Cache,** which stores the recently accessed data.
- If the data is not available in the Block Cache, it checks the required data in the in-memory store, i.e., **MemStore.**

HBASE READ OPERATION

- The region server first checks its **Block Cache,** which stores the recently accessed data.
- If the data is not available in the Block Cache, it checks the required data in the in-memory store, i.e., **MemStore.**
- If the MemStore does not contain that particular key-value, the region server uses **Bloom Filters** to find the HFile(StoreFile), which contains that particular key-value pair.

BLOOM FILTERS

- It is designed to check whether an element is present in a set or not.
- It has a probabilistic data structure.
- It has a bit vector-based data structure.

HBASE READ OPERATION

- The region server first checks its **Block Cache** which stores the recently accessed data.
- If the data is not available in the Block Cache, it checks the required data in the in-memory store, i.e., **MemStore.**
- If the MemStore does not contain that particular key-value, the region server uses **Bloom Filters** to find the HFile(StoreFile) which contains that particular key-value pair.
- Once the **HFile** is identified, data is read in an efficient manner and returned to the client (if present).

KEY TAKEAWAYS

- Meta Table stores the location of region servers.
- The location of Meta Table is stored in Zookeeper.
- The client first fetches the location of region server where the region having the required data is present.
- For reading the data:
  - First, the block cache is checked, and if the data is not found in the block cache, the Memstore is checked for the data.
  - If it is not found in the Memstore, then the HFile, where the data is present, is searched using the bloom filters.
  - The data is read from the HFile.

Thank You!

Segment 4

Write Operation in HBase

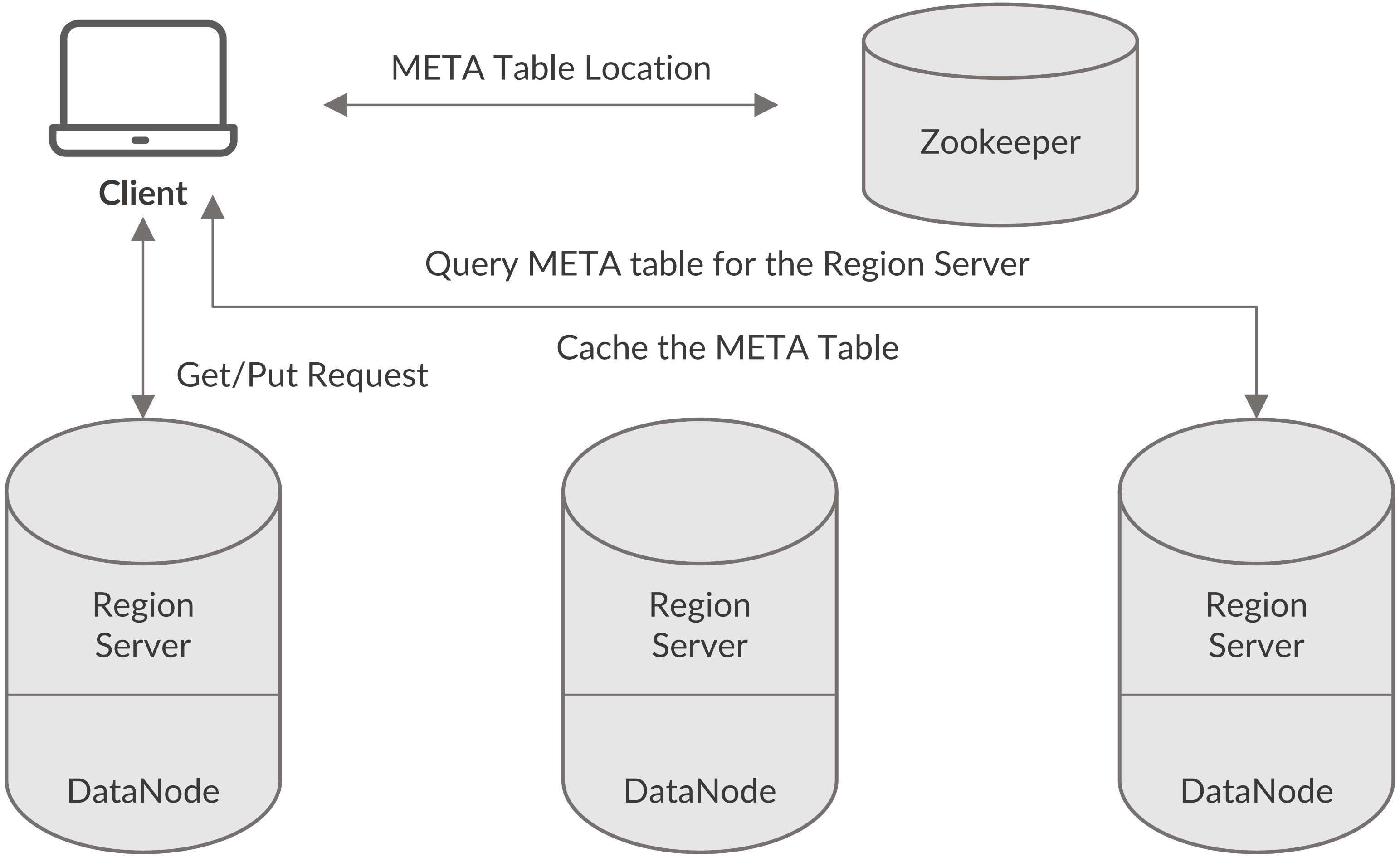
LEARNING OBJECTIVES

How is write operation performed in HBase?

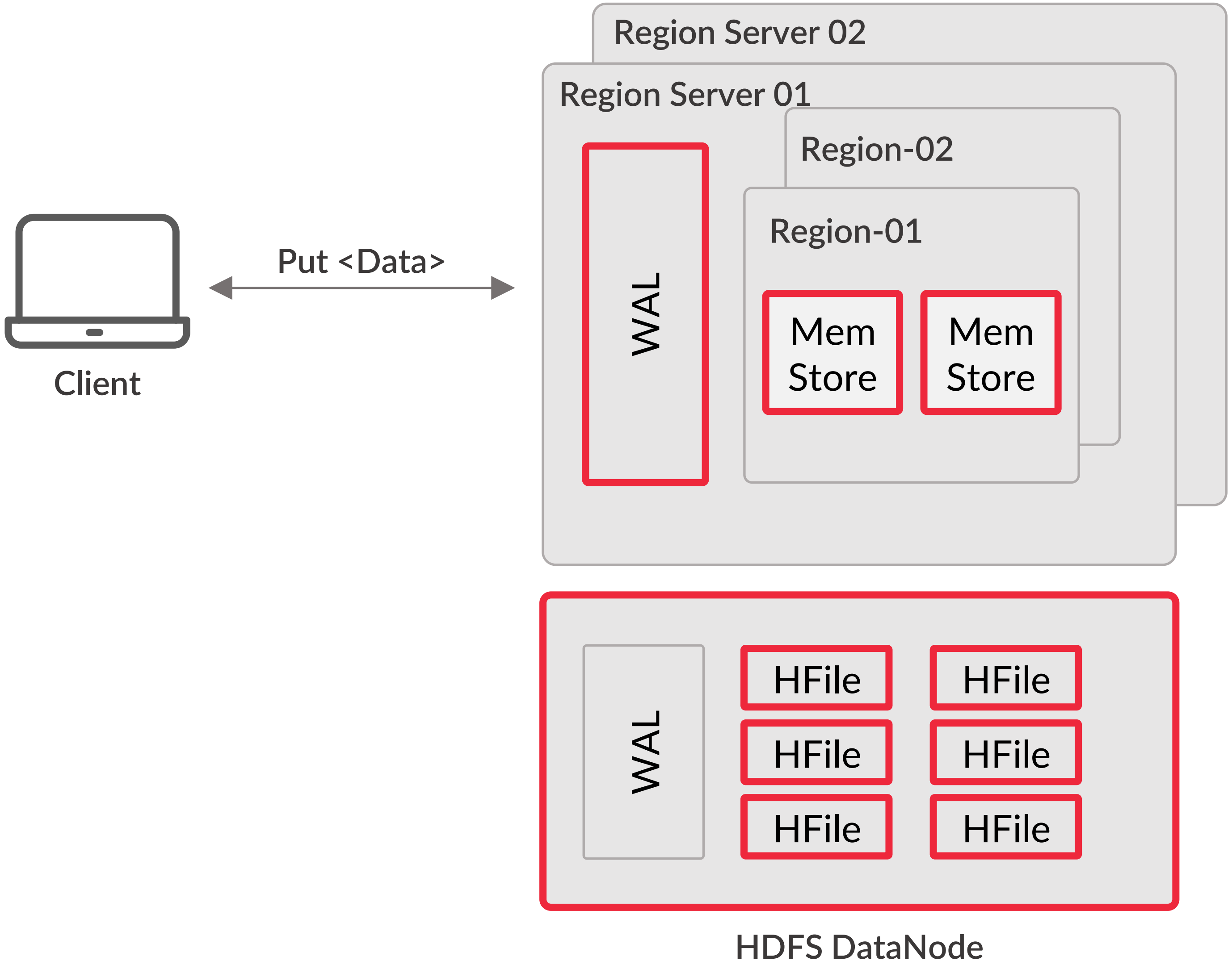
What are compactions?

What are the steps in delete operation?

FETCHING THE REGION SERVER



HBASE WRITE OPERATION



HBASE WRITE OPERATION

- The data first needs to be written to the **WAL.**

HBASE WRITE OPERATION

- The data first needs to be written to the **WAL.**
- After that, it is placed in the **MemStore** of a region and the key-value pairs.

HBASE WRITE OPERATION

- The data first needs to be written to the **WAL.**
- After that, it is placed in the **MemStore** of a region and the key-value pairs.
- When the MemStore becomes full, its contents are flushed to HDFS(DataNode) to form a new HFile.

HBASE WRITE OPERATION

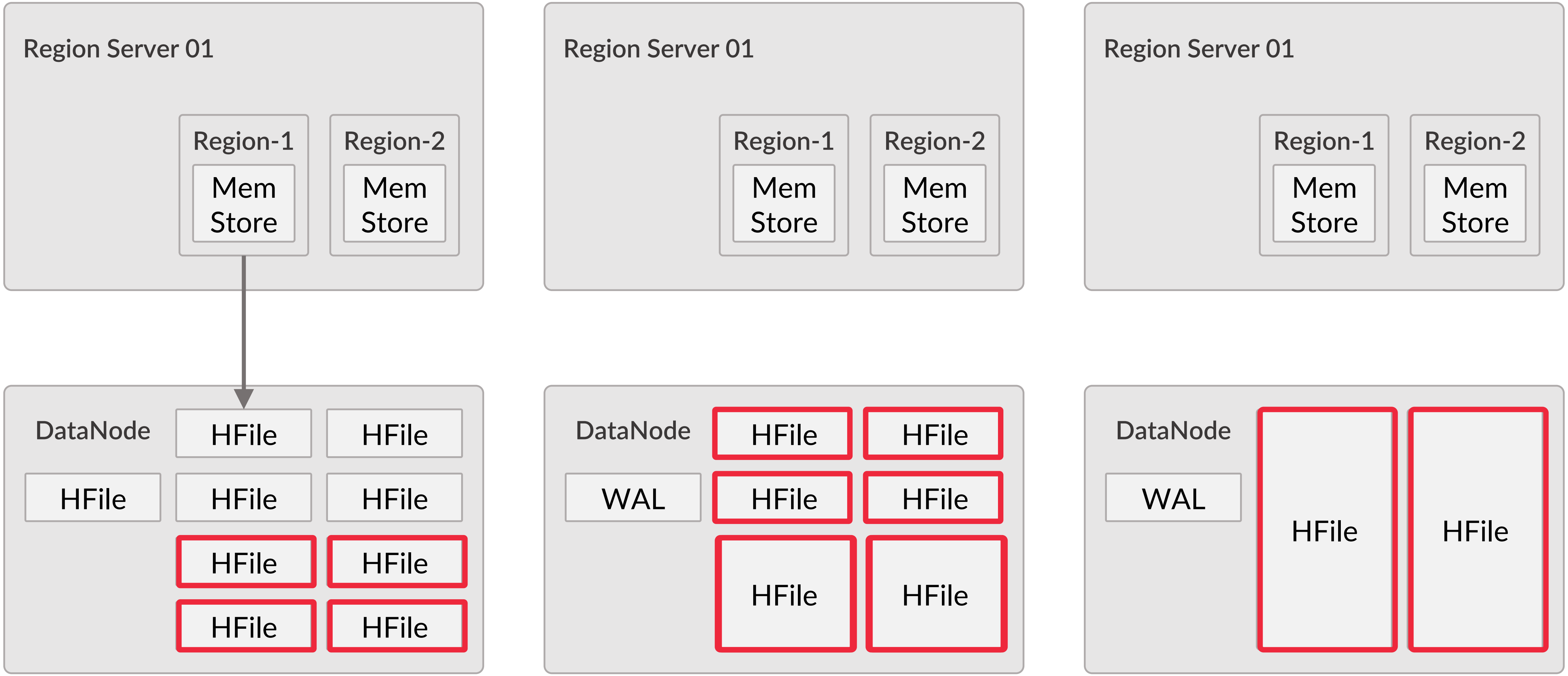
- The data first needs to be written to the **WAL.**
- After that, it is placed in the **MemStore** of a region and the key-value pairs.
- When the MemStore becomes full, its contents are flushed to HDFS(DataNode) to form a new HFile.
- Finally, an acknowledgement is sent back to the client.

COMPACTION

- Heavy write operations lead to two challenges:
  - Low efficiency of read operations
  - Dirty data
- Compaction is the process of combining the smaller HFiles into larger HFiles.
- Two types:
  - **Minor Compactions:** Smaller HFiles are combined to form large HFiles.

COMPACTION

Minor and Major



Minor Compaction

Major Compaction

COMPACTION

- Heavy write operations lead to two challenges:
  - Low efficiency of read operations
  - Dirty data
- Compaction is the process of combining the smaller HFiles into larger Hfiles.
- Two types:
  - **Minor Compactions:** Smaller HFiles are combined to form large HFiles.
  - **Major Compactions:** Store files of a region are combined into a single Store File. It removes all the deleted and expired data from the HFiles.

HBASE DELETE OPERATION

- This is a special type of Update operation.
- The deleted value is marked with a **Tombstone Marker**.
- When the deleted data is accessed, NULL is returned.
- HFiles are immutable.
- Deleted data is removed when Major Compaction is done.
- Three types of Tombstone Markers:
  - Version Delete Marker
  - Column Delete Marker
  - Family Delete Marker

KEY TAKEAWAYS

- For write operation:
  - First the data is written to write ahead log (WAL).
  - The data from WAL is moved to Memstore. Once the Memstore is full, the data is flushed into HDFS creating a new HFile.
- Heavy write operations lead to dirty data and reduction in read efficiency.
- Compaction is a technique in which smaller HFiles are combined to form larger HFiles.
- There are two types of compactions: major and minor.
- The delete operation is a special type of update operation.

Thank You!

Segment 5

HBase Schema Design

LEARNING OBJECTIVES

How can the HBase schema be designed optimally?

How can filters be applied while reading data?

SCHEMA DESIGN

- Consider-
  - n- number of rows
  - e- number of key-value pairs in HFile
  - b- number of blocks
  - c- number of columns

SCHEMA DESIGN

- If data is found in **Block Cache,** Complexity- **O(1)**
- If data is found in **MemStore,** Complexity- **O(log(e))**
- If not found in MemStore-
  - To find the required **block** in HFile, Complexity- **O(log(b))**
  - If data found in the same block, Complexity- **O(b/e)**
  - If not found in the same block, Complexity- **O(c)**
- If c increases, complexity increases.
- Rows should be equally distributed across region servers.
- Filtering on the basis of column, timestamp and version.
- Denormalise data.

Segment 6

HBase Use Cases

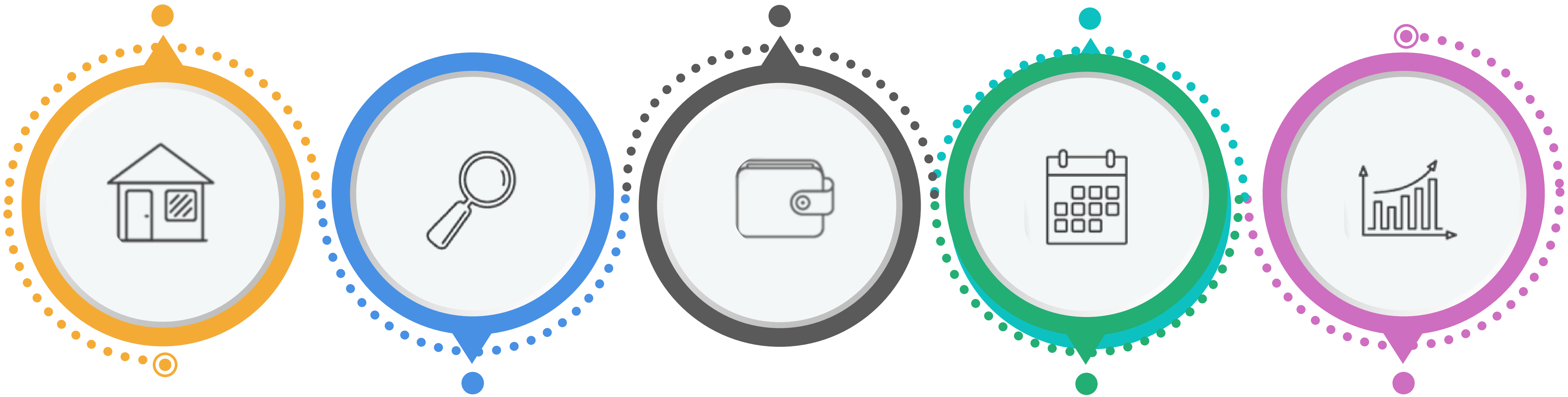
LEARNING OBJECTIVES

Which industries are using HBase?

What purposes does HBase serve in different industries?

Which companies are using HBase?

FIELDS OF USE



Medical

E-Commerce

Banking

Telecom

Sports

Finance

HBASE IN THE INDUSTRY

Bloomberg
FLIPBOARD
photobucket
HORTONWORKS
OPOWER
Cerner
Optimizely
Dropbox
jive
an Aurea company
appnexus
YAHOO!
FINRA
splice MACHINE
mi
intel
f
IBM
TREND MICRO
explorys
FLURRY
HUAWEI
GROUPON
淘宝网
Taobao.com
airbnb
salesforce
AOL
box
KLOUT
Pinterest
OCLC
cloudera
CyberAgent
ebay
HubSpot
Google
Limelight NETWORKS
Microsoft
NGDATA
rocketfuel

HBASE IN THE INDUSTRY

- The use cases that HBase serves in the industry are very diverse, for example:
  - Yahoo uses HBase for content personalisation and web cache for searching.
  - E-commerce giant eBay uses HBase for optimising eBay's search engine performance.
  - Pinterest uses HBase for relevant search.
  - Bloomberg uses HBase for time series data storage.
  - Facebook used to employ HBase for their messaging application.

KEY TAKEAWAYS

- HBase serves industries like e-commerce, sports, medical, etc.
- Companies like Yahoo, eBay, Airbnb and Intel are using HBase for various purposes.

Thank You!

Segment 7

HBase Advantages and Disadvantages

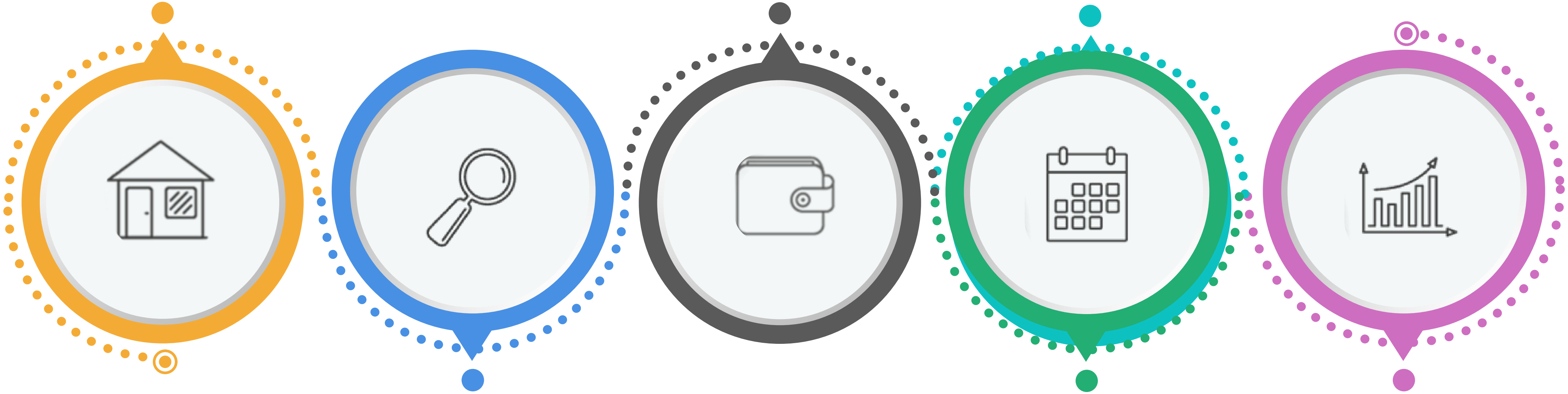
LEARNING OBJECTIVES

What are the advantages of using HBase?

What are the disadvantages of using HBase?

What other NoSQL Databases can overcome the HBase disadvantages?

HBASE: ADVANTAGES



Distributed Databases

Read/Write Consistency

Schema-Less

Query Optimisation

Less Reading and Processing Time

Horizontal Scalability

HBASE: DISADVANTAGES

No Built-in Authentication

Does Not Support SQL Functions Like joins, groupby, etc.

Resource Intensive

Single Sort Key

HBASE: DISADVANTAGES

- Single point of failure
- No query language available
- Key-value pair

- The system is always available.
  Expressive query language
- Cassandra Query Language (CQL)

SESSION SUMMARY

- HBase architecture has three servers- HMaster, Region Servers and Zookeeper.
- HBase architecture is a master-slave architecture, with HMaster being the master and Region server being the slave.
- Zookeeper is responsible for the communication between HMaster and region server.
- When smaller HFiles are combined to form larger HFiles, it is called compaction.
- Equal distribution of rows can improve the schema design.
- HBase is used in industries like medical, e-commerce, sports etc.

Thank You!