UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

BIG DATA
Analytics & Management

Lecture 4 (02/13, 02/15): HBase

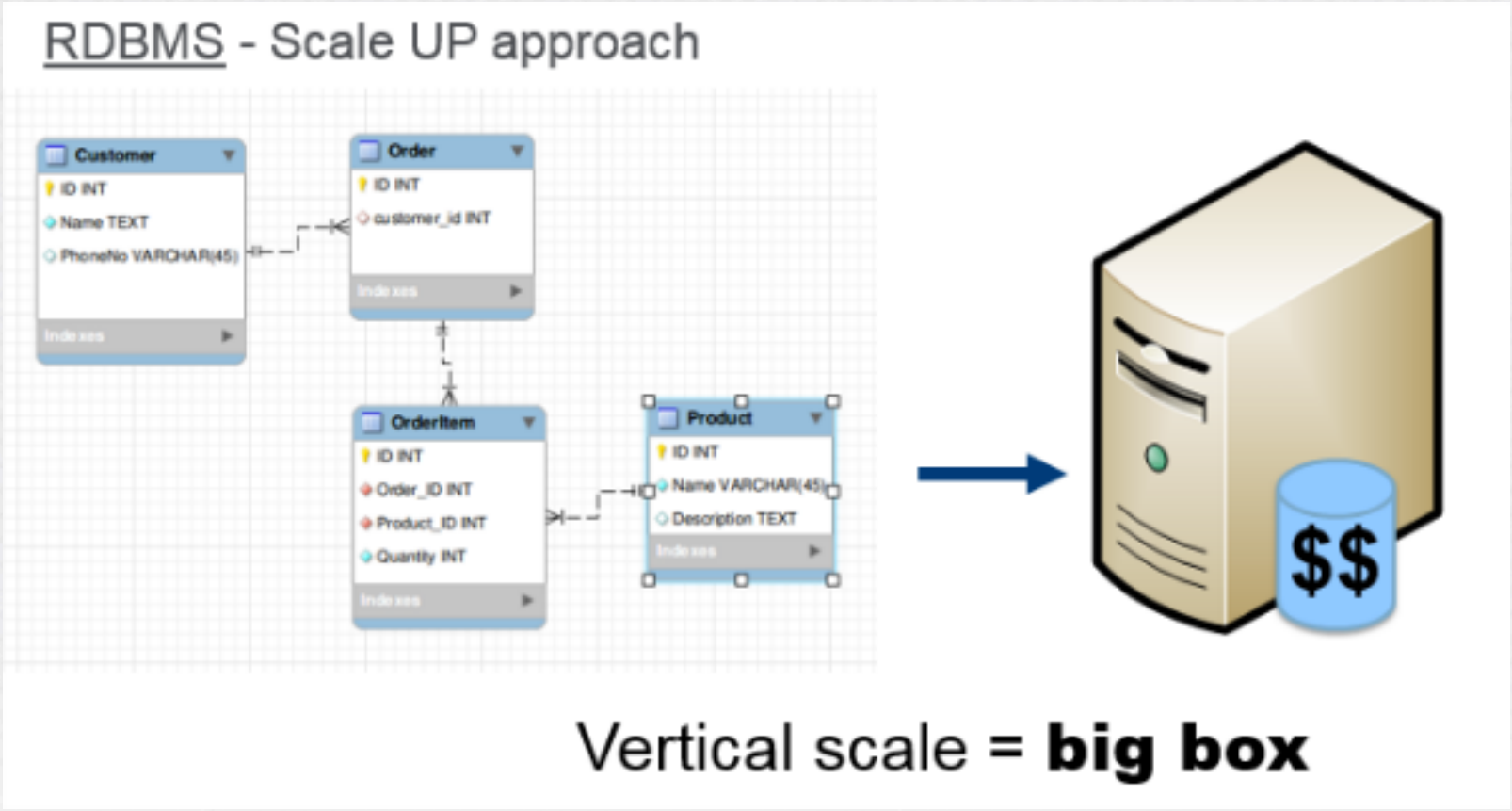
Decisions, Operations & Information Technologies
Robert H. Smith School of Business
Spring, 2017

Relational database

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

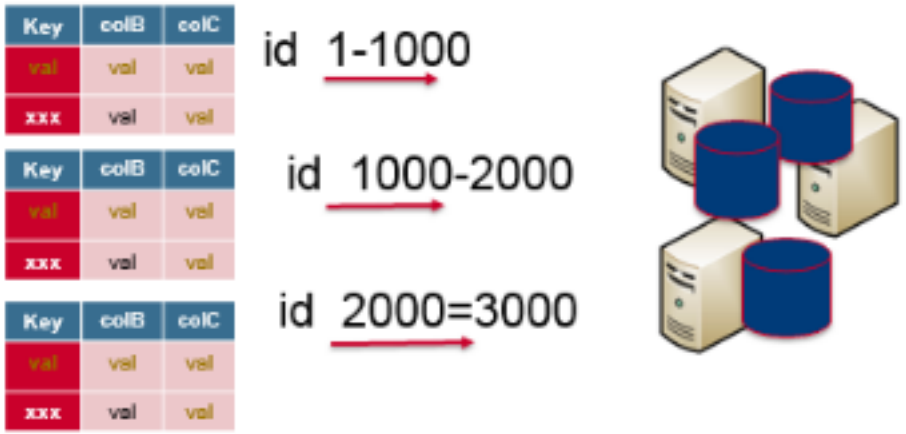
- Relational databases have provided a standard persistence model
- SQL has become a de-facto standard model of data manipulation (SQL)
- Relational databases manage concurrency for transactions
- Relational database have lots of tools

Scale up (vertically)



Scale up (horizontally)

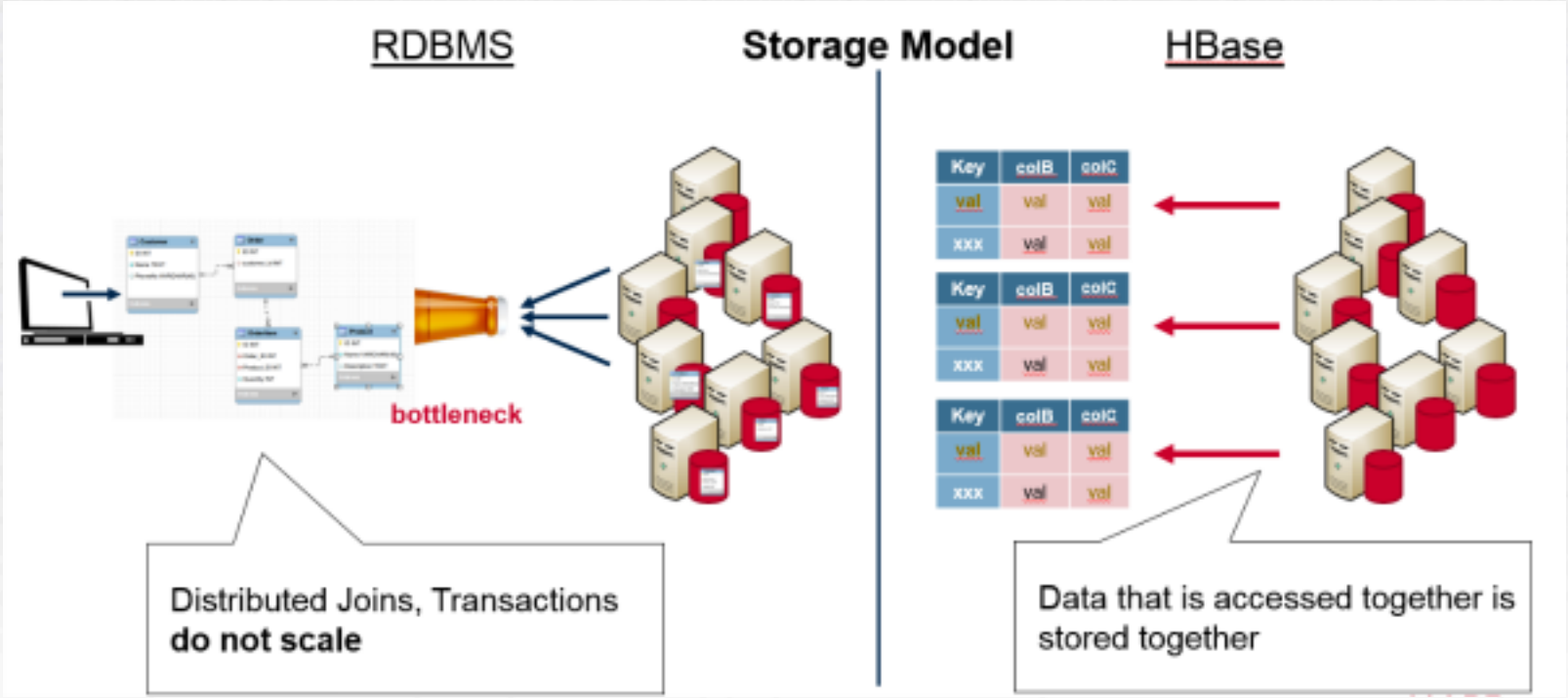
Horizontal scale: split table by rows into partitions across a cluster



Limitations of a relational model

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

- **Database normalization eliminates redundant data, which makes storage efficient**
- **It causes joins for queries, in order to bring the data back together again.**



```
mysql> SELECT * FROM suppliers;
+-------------+---------------+-------------------+------------------+
| supplier_id | supplier_name | supplier_address  | supplier_contact |
+-------------+---------------+-------------------+------------------+
|           1 | Microsoft     | 1 Microsoft Way   | Bill Gates       |
|           2 | Apple, Inc.   | 1 Infinite Loop   | Steve Jobs       |
|           3 | EasyTech      | 100 Beltway Drive | John Williams    |
|           4 | WildTech      | 100 Hard Drive    | Alan Wilkes      |
+-------------+---------------+-------------------+------------------+
4 rows in set (0.00 sec)
```

```
SELECT * FROM product;
+-----------+----------------------------+-----------------------------+-------------+
| prod_code | prod_name                  | prod_desc                   | supplier_id |
+-----------+----------------------------+-----------------------------+-------------+
|         1 | CD-RW Model 4543           | CD Writer                   |           3 |
|         2 | EasyTech Mouse 7632        | Cordless Mouse              |           3 |
|         3 | WildTech 250Gb 1700        | SATA Disk Drive             |           4 |
|         4 | Microsoft 10-20 Keyboard   | Ergonomic Keyboard          |           1 |
|         5 | Apple iPhone 8Gb           | Smart Phone                 |           2 |
+-----------+----------------------------+-----------------------------+-------------+
```

```
SELECT prod_name, supplier_name, supplier_address FROM product, suppliers WHERE (product.supplier_id = suppliers.supplier_id);
```

```
+--------------------------+---------------+-------------------+
| prod_name                | supplier_name | supplier_address  |
+--------------------------+---------------+-------------------+
| Microsoft 10-20 Keyboard | Microsoft     | 1 Microsoft Way   |
| Apple iPhone 8Gb         | Apple, Inc.   | 1 Infinite Loop   |
| CD-RW Model 4543         | EasyTech      | 100 Beltway Drive |
| EasyTech Mouse 7632      | EasyTech      | 100 Beltway Drive |
| WildTech 250Gb 1700      | WildTech      | 100 Hard Drive    |
+--------------------------+---------------+-------------------+
```

HBase

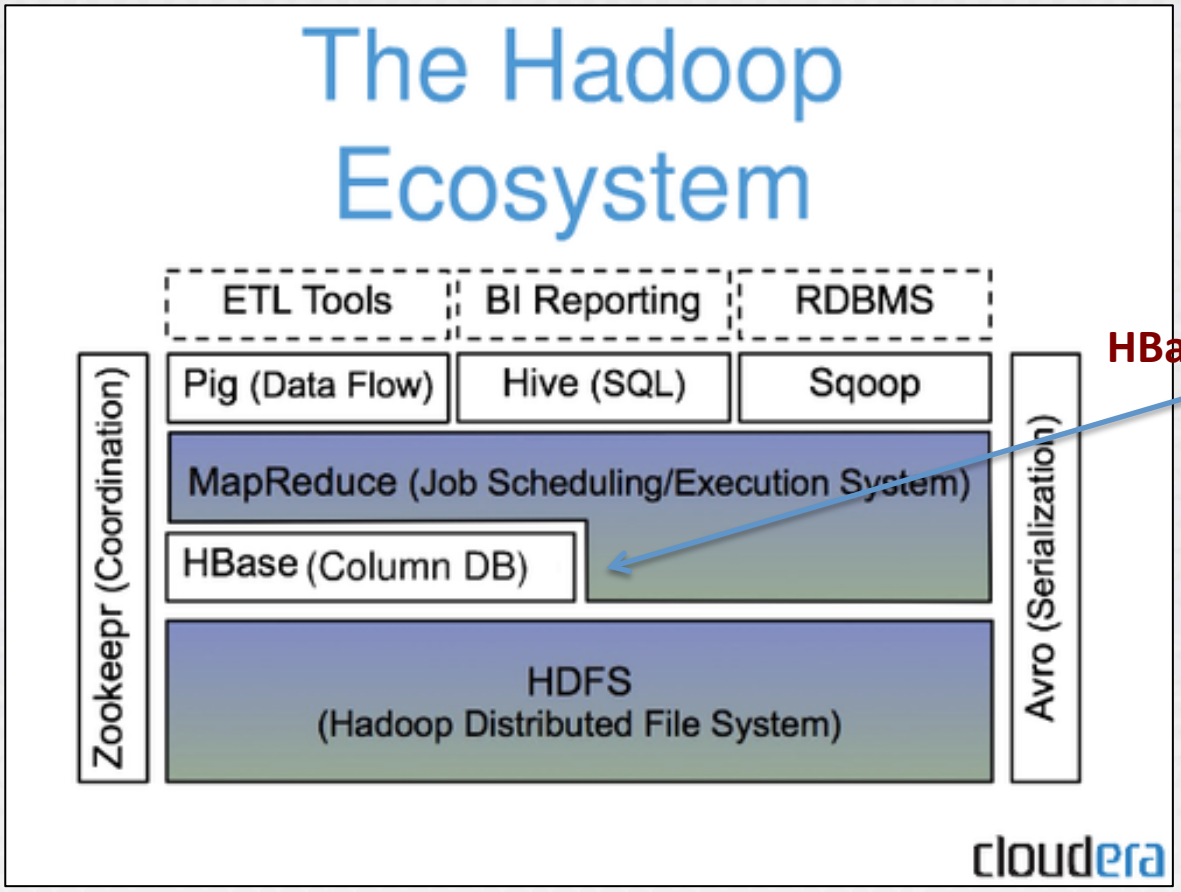
	supplier				product		
rid	id	name	contact	address	code	name	description
1	1	microsoft	Bill Gates	1 microsoft way	4	Microsoft 10-20 Keyboard	Ergonomic Keyboard
2							
3							
4							
5							

HBase: overview

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

- HBase is a distributed column-oriented data store built on top of HDFS
- HBase is an Apache open source project whose goal is to provide storage for the Hadoop Distributed Computing
- Data is logically organized into tables, rows and columns

Part of hadoop's ecosystem



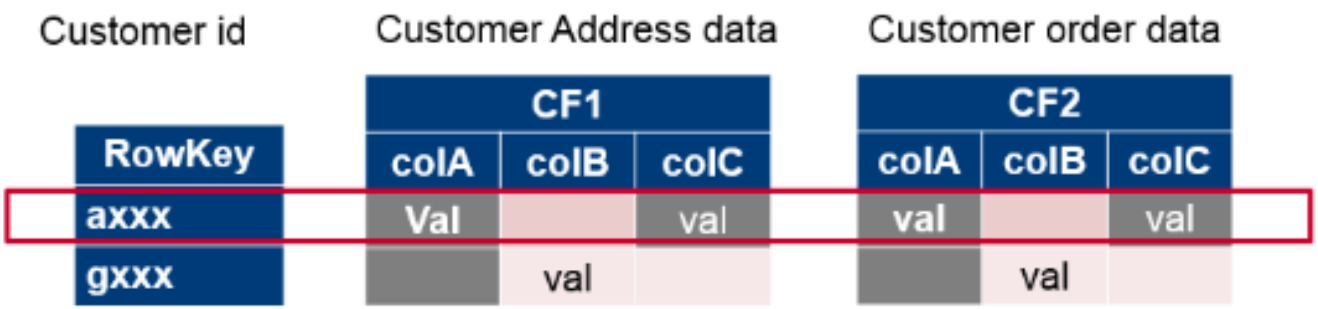
HBase is built on top of HDFS

HBase files are internally stored in HDFS

HBase Data Model

HBase

- **HBase is referred to as a column family-oriented data store.**
- **It's also row-oriented.**
  - ❑ **Each row is indexed by a key that you can use for lookup**



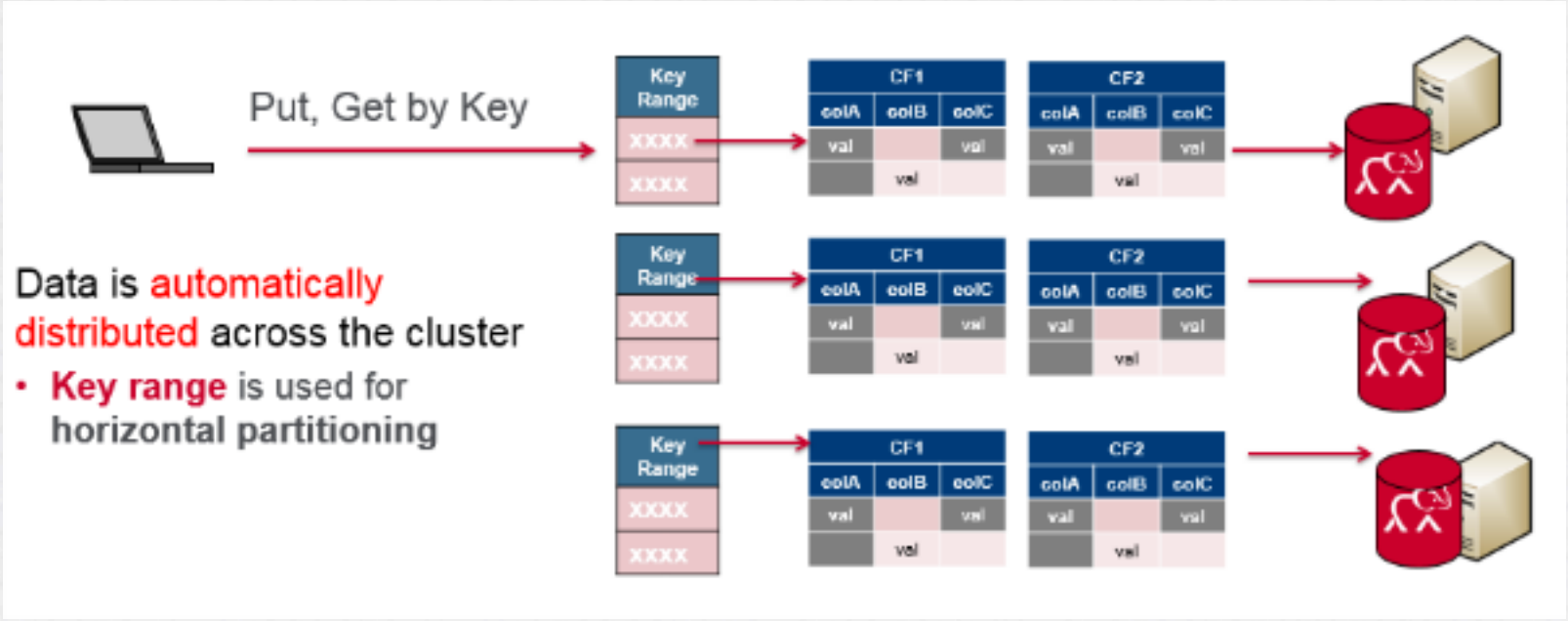
Data is accessed and stored together:

- RowKey is the primary index
- Column Families group similar data by row key

HBase

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

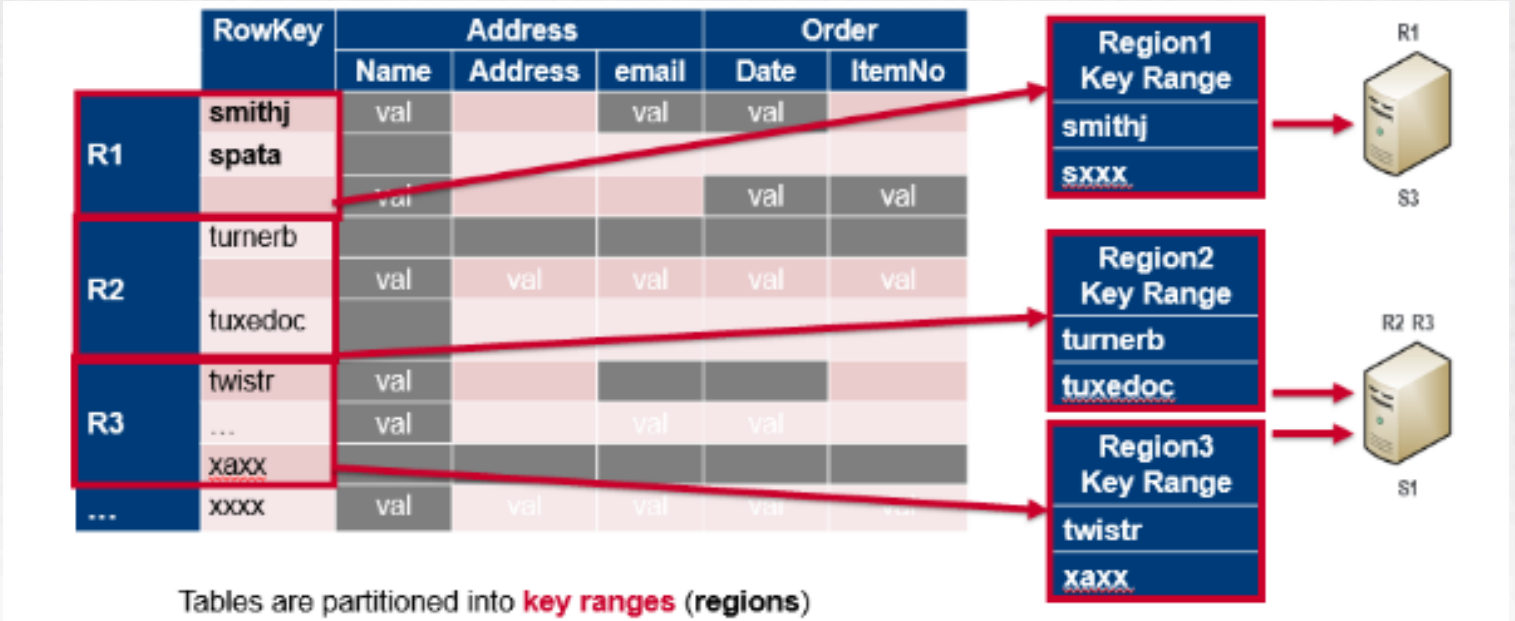
- **HBase is a distributed database**



"Tables" are partitioned into regions

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

- "Tables" are divided into sequences of rows, by key range, called regions
- These regions are then assigned to the data nodes in the cluster called "RegionServers"
- This is done automatically and is how HBase was designed for horizontal sharding



Tables are partitioned into key ranges (regions)
Region= served by nodes (Region Servers)
Regions are spread across cluster

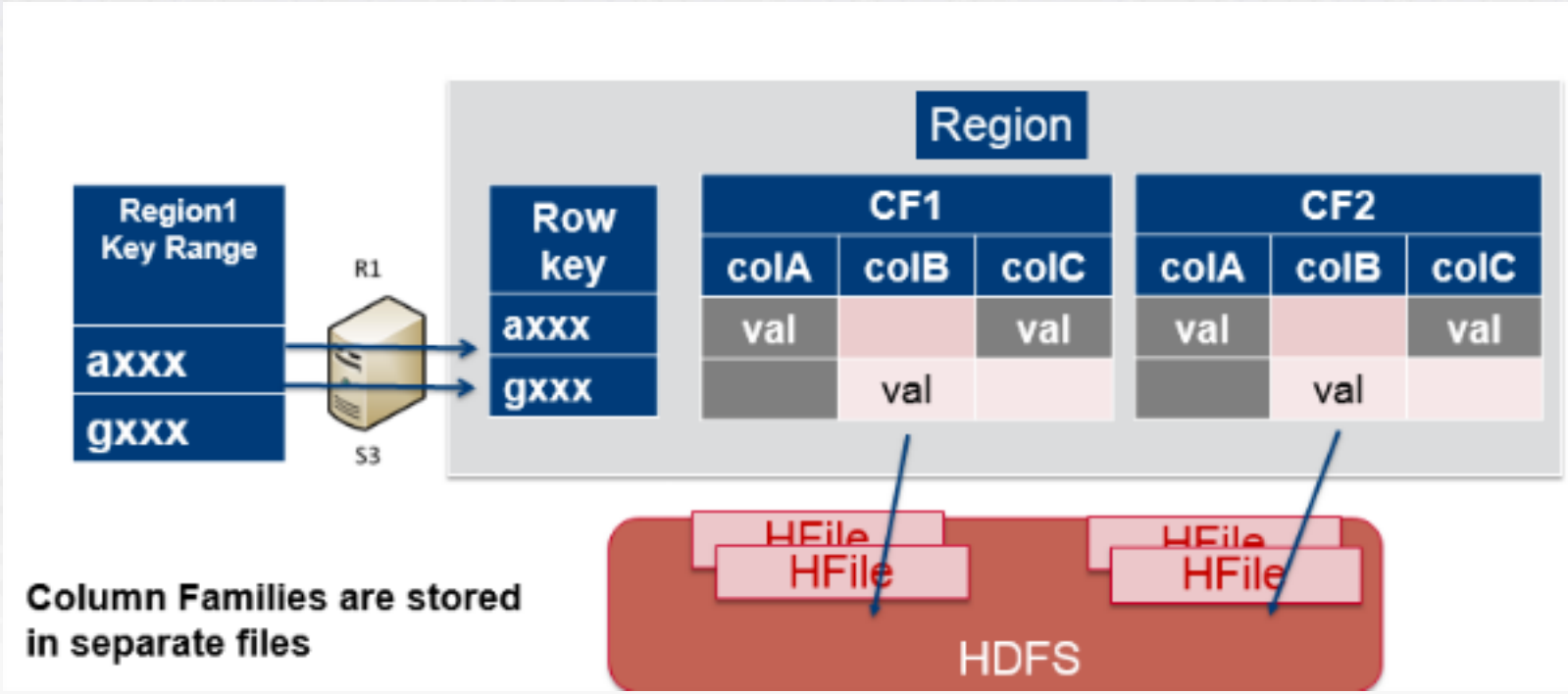
Row

- A row consists of a row key and one or more columns with values
- Records in HBase are stored in sorted order, according to rowkey (related rows are near each other)

RowKey	Address			Order				...
	street	city	state	Date	ItemNo	Ship Address	Cost	
smithj	val		val	val			val	
spata								
sxxxx	val			val	val	val		
...								
turnerb	val	val	val	val	val	val	val	
...								
	val							
twistr	val		val	val			val	
...								
zaxx	val	val	val	val	val	val		
zxxx	val						val	

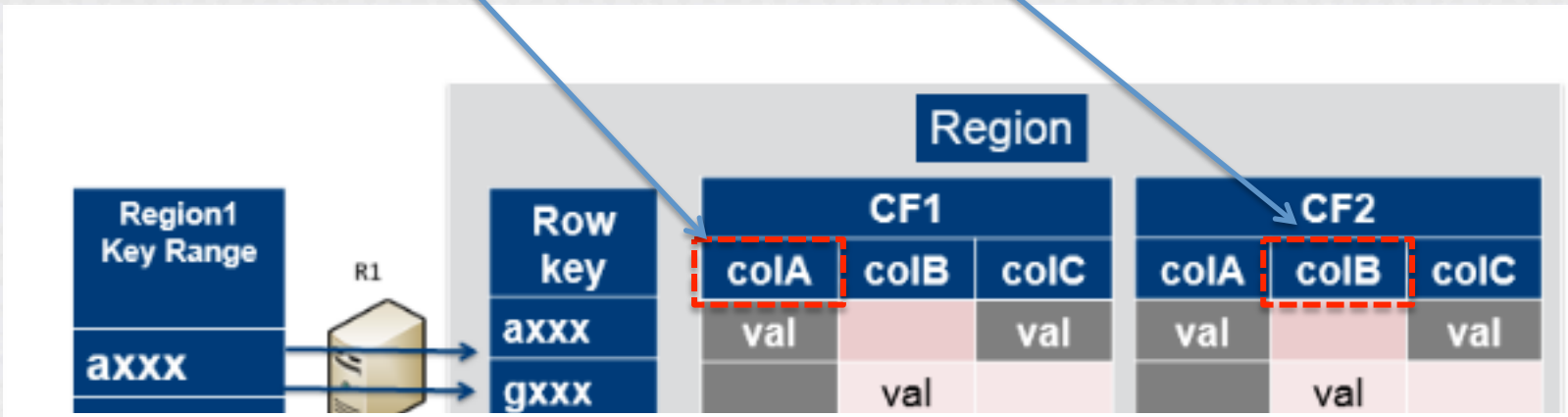
Column families

- **Defined by the user and fixed at table creation**
- **Mapped to storage files**
- **Stored in separate files, which can also be accessed separately**



Column

- **A column in HBase consists of a column family and a column qualifier, which are delimited by a : (colon) character**
  - **E.g., CF1:colA, CF1:colB, CF2:colB**

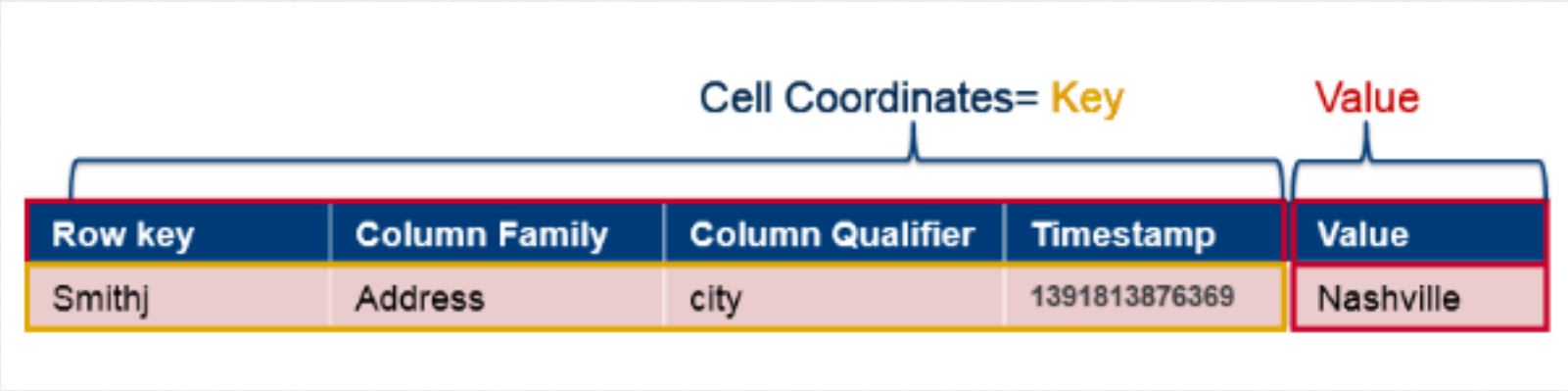


Cell and timestamp

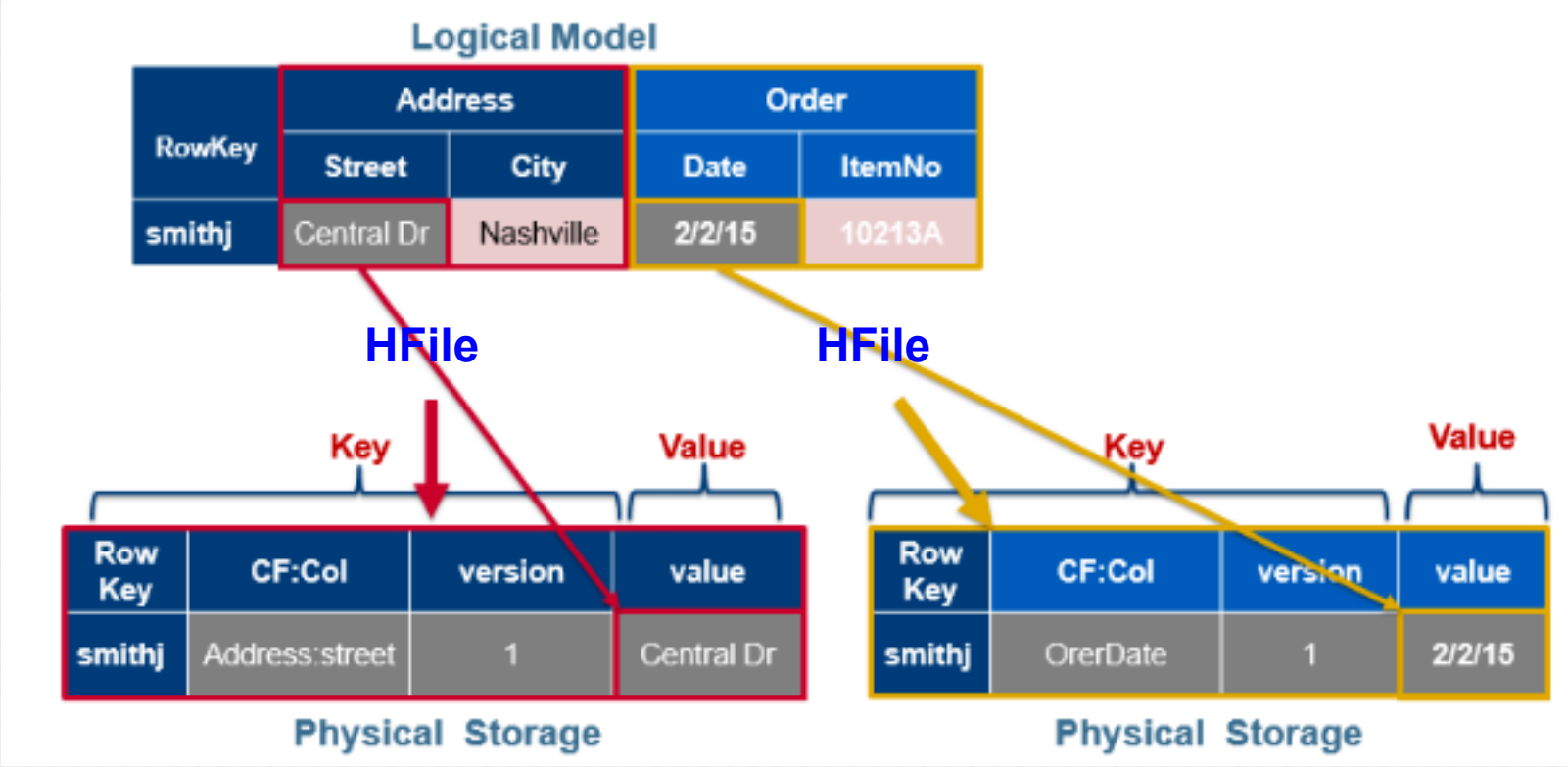
UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

- **A cell is a combination of row, column family, and column qualifier, and contains a value and a timestamp (representing the value's version)**
- **A timestamp is written alongside each value, and is the identifier for a given version of a value.**
  - **By default, the timestamp represents the time on the RegionServer when the data was written**
  - **We can specify a different timestamp value when we put data into the cell**

Data in HBase - logical view



Data in HBase – physical view



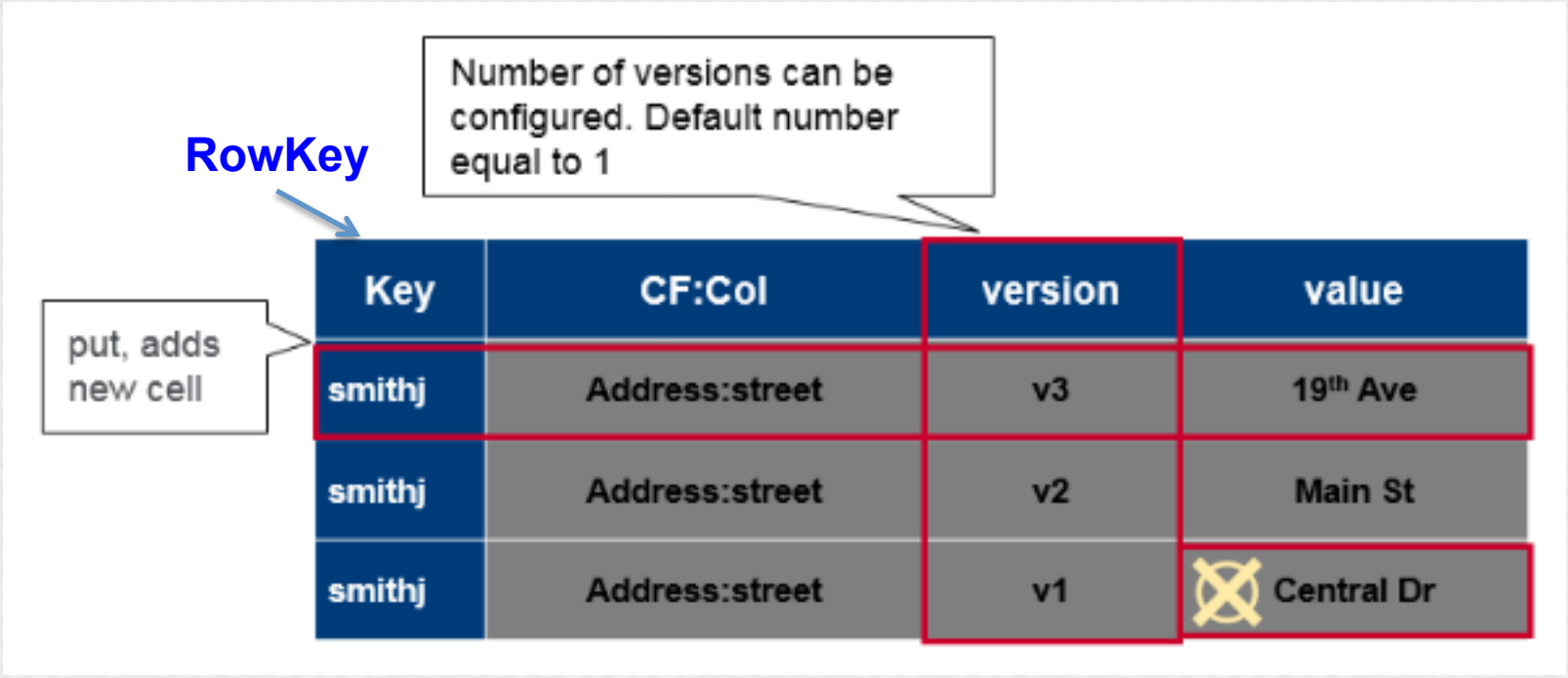
Example

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

Row Key	Time Stamp	ColumnFamily `contents`	ColumnFamily `anchor`	ColumnFamily `people`
"com.cnn.www"	t9		anchor:cnnsi.com = "CNN"	
"com.cnn.www"	t8		anchor:my.look.ca = "CNN.com"	
"com.cnn.www"	t6	contents:html = "<html>…"		
"com.cnn.www"	t5	contents:html = "<html>…"		
"com.cnn.www"	t3	contents:html = "<html>…"		

```
{
  "com.cnn.www": {
    contents: {
      t6: contents:html: "<html>..."
      t5: contents:html: "<html>..."
      t3: contents:html: "<html>..."
    }
    anchor: {
      t9: anchor:cnnsi.com = "CNN"
      t8: anchor:my.look.ca = "CNN.com"
    }
    people: {}
  }
  "com.example.www": {
    contents: {
      t5: contents:html: "<html>..."
    }
    anchor: {}
    people: {
      t5: people:author: "John Doe"
    }
  }
}
```

Versioned data



Notes on data model

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

- HBase schema consists of several ***Tables***
- Each table consists of a set of ***Column Families***
  - **Columns** are **NOT** part of the schema

- HBase has ***Dynamic Columns***
  - Because column names are encoded inside the cells
  - Different cells can have different columns

"Roles" column family has different columns in different cells

RowKey	Value
cutting	Info:{ 'height': '9ft', 'state': 'MD' } Roles:{ 'ASF': 'Director', 'Hadoop': 'Founder' }
tlipcon	Info:{ 'height': '5ft7', 'state': 'CA'} Roles:{ 'Hadoop': 'Committer', 'Hadoop': 'Founder', 'Hive': 'Contributor' }

Notes on data model (Cont'd)

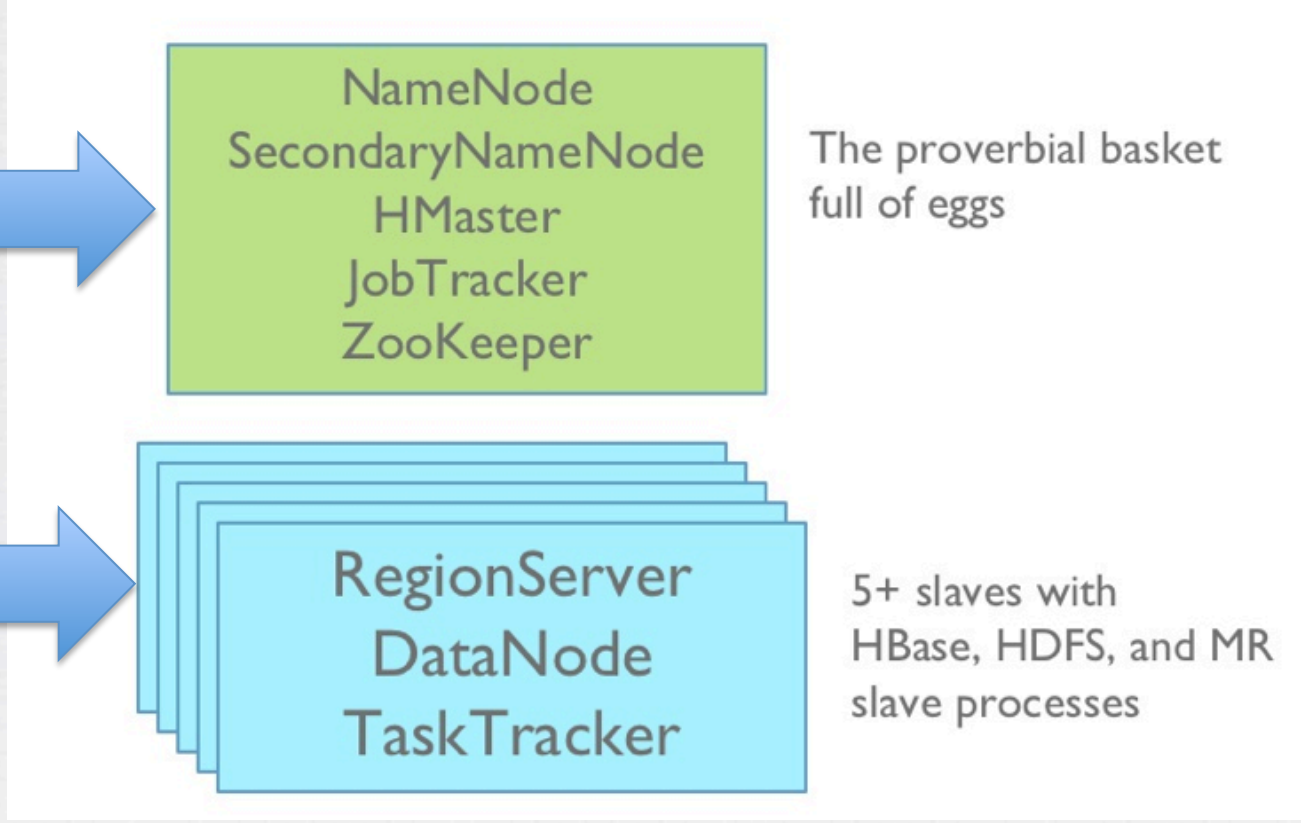
- The ***version number*** can be user-supplied
  - ❑ Even does not have to be inserted in increasing order
  - ❑ Version number are unique within each key
- Table can be very sparse
  - ❑ Many cells are empty
- ***Keys*** are indexed as the primary key

Has two columns
[cnnsi.com & my.look.ca]

Row Key	Time Stamp	ColumnFamily `contents`	ColumnFamily `anchor`
"com.cnn.www"	t9		`anchor:cnnsi.com` = "CNN"
"com.cnn.www"	t8		`anchor:my.look.ca` = "CNN.com"
"com.cnn.www"	t6	`contents:html` = "<html>..."	
"com.cnn.www"	t5	`contents:html` = "<html>..."	
"com.cnn.www"	t3	`contents:html` = "<html>..."	

HBase Architecture

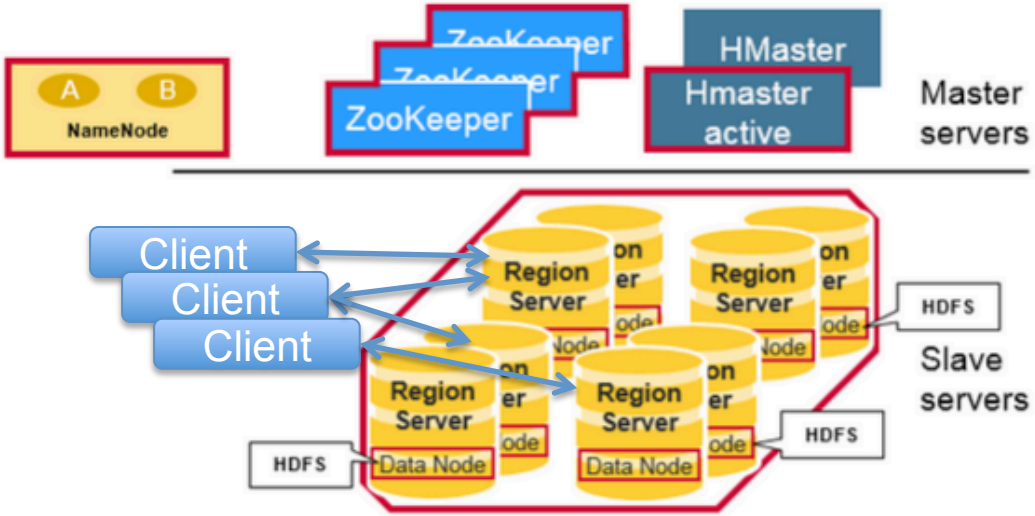
Master-slave relationship



Master-slave relationship

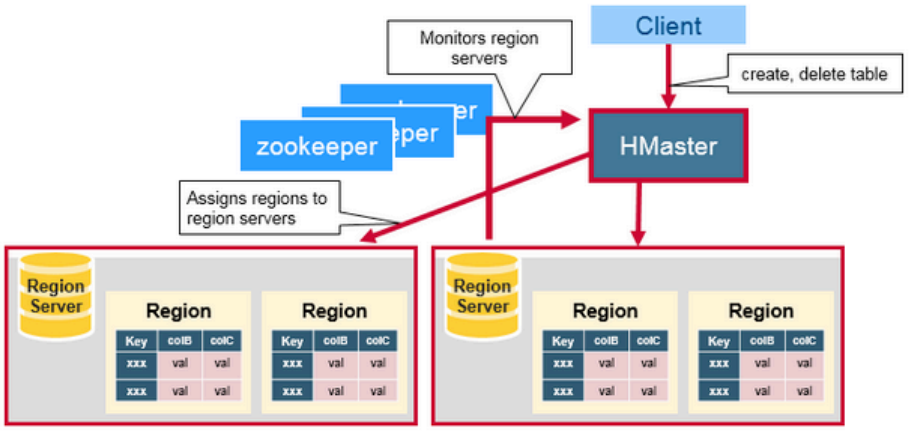
UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

- The HMaster
  - One active master (and other sleep masters)
- The HRegionServer
  - Many region servers
  - Each region server has many regions
- The HBase client



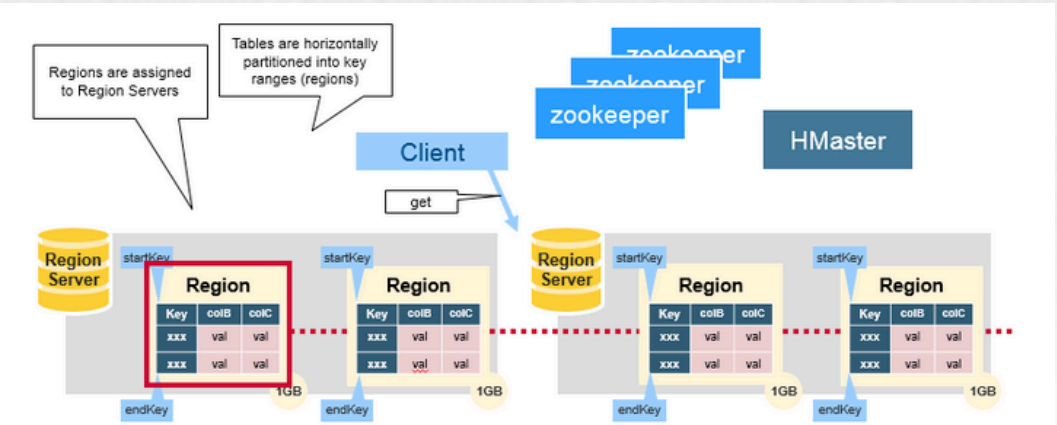
HMaster

- A master is responsible for:
  - ❑ Coordinating the region servers
    - Assigning regions on startup, re-assigning regions for recovery or load balancing
    - Monitoring all RegionServer instances in the cluster (listens for notifications from ZooKeeper)
  - ❑ Admin functions
    - Interface for creating, deleting, updating tables



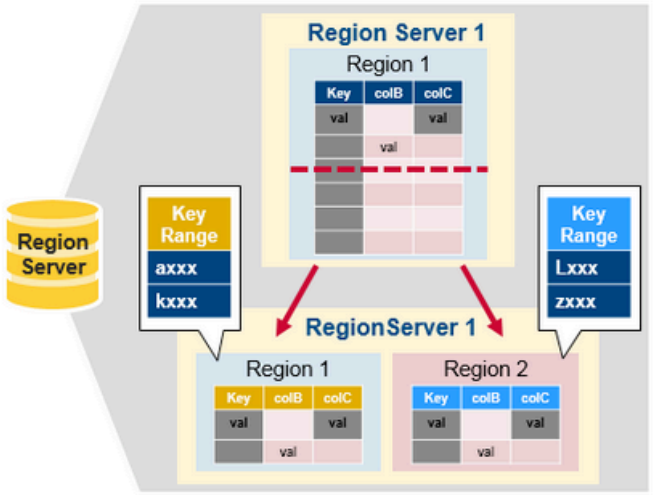
Region

- 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(inclusive) and end key(exclusive)
- Regions are assigned to the nodes in the cluster, called "RegionServers"
- Each region is 1GB in size (default)
- A region server can serve about 1,000 regions



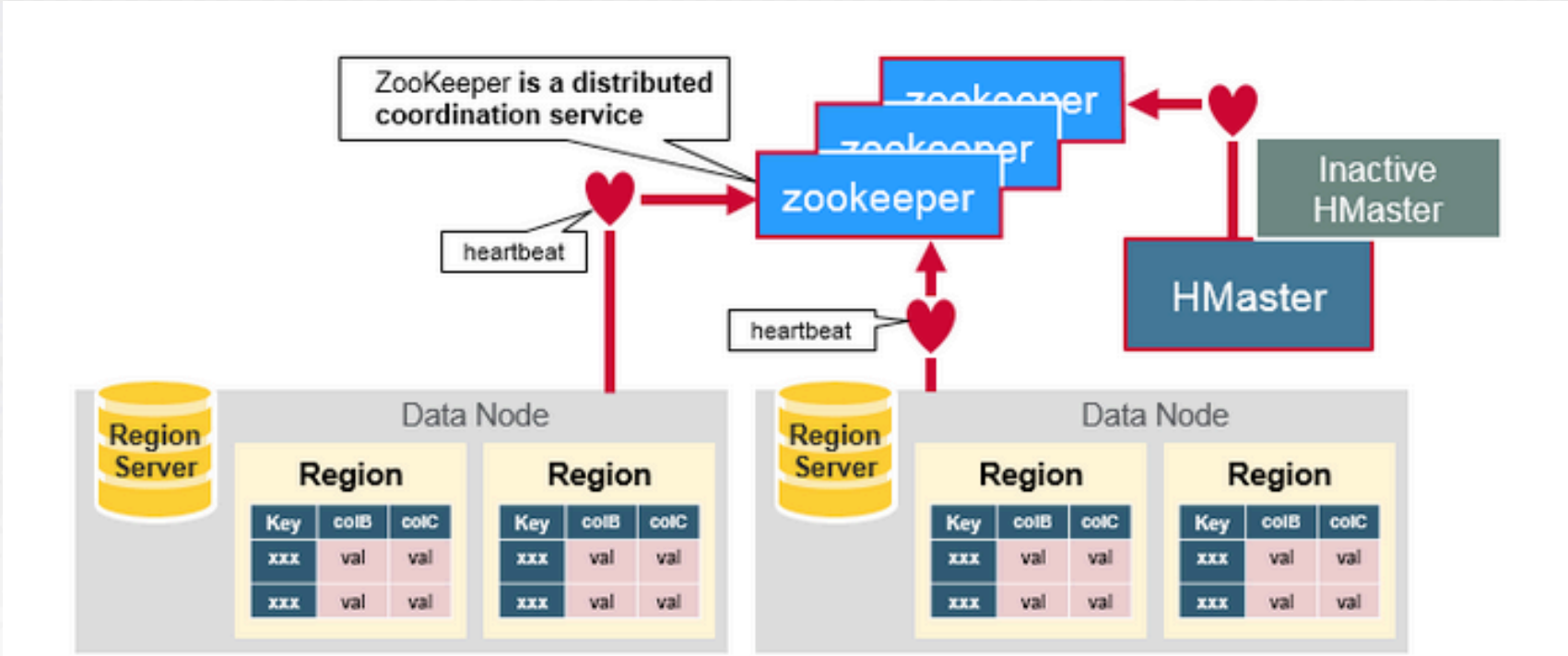
Region split

- Initially there is one region per table
- When a region grows too large, it splits into two child regions
- Both child regions, representing one-half of the original region, are opened in parallel on the same Region server, and then the split is reported to the HMaster
- For load balancing reasons, the HMaster may schedule for new regions to be moved off to other servers



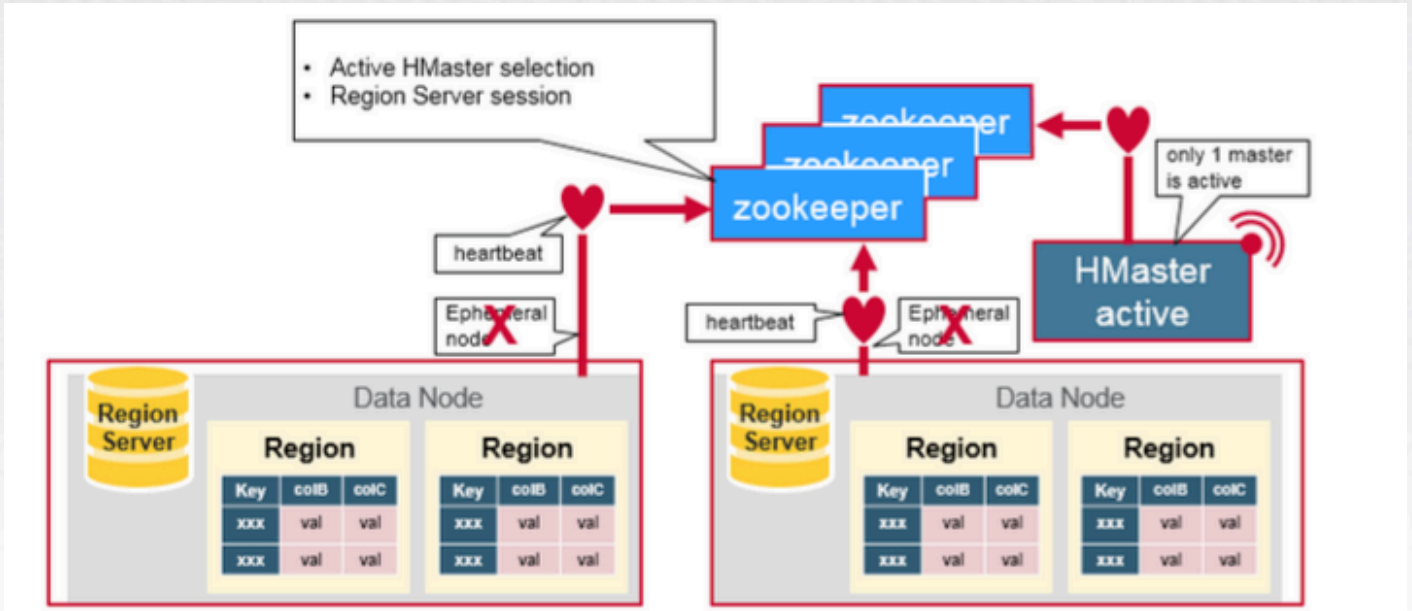
ZooKeeper: the coordinator

- HBase uses ZooKeeper as a distributed coordination service to maintain server state in the cluster
- Zookeeper maintains which servers are alive and available, and provides server failure notification



How the components work together

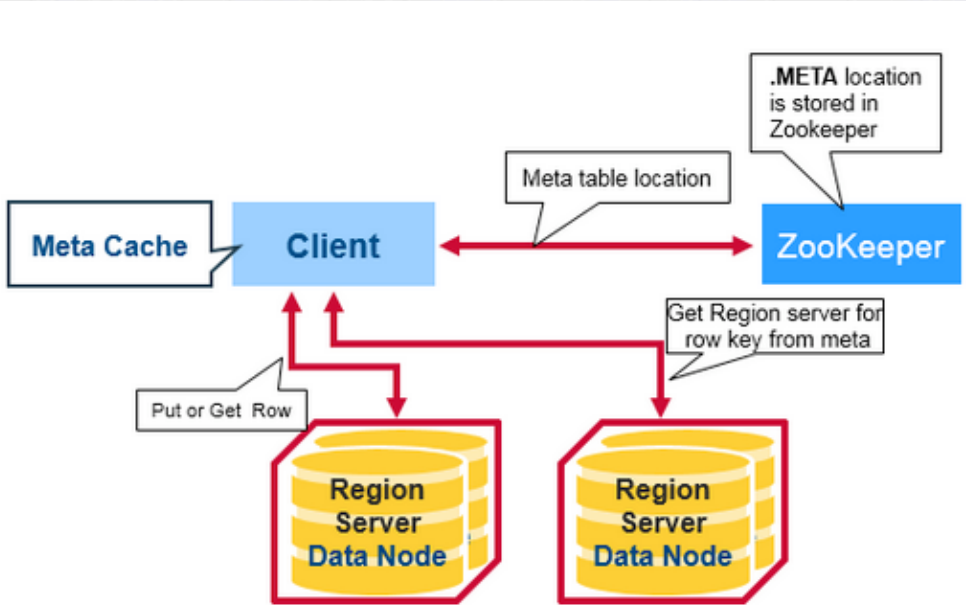
- Zookeeper is used to coordinate shared state information for members of distributed systems
- Region servers and the active HMaster connect with a session to ZooKeeper
- The ZooKeeper maintains ephemeral nodes for active sessions via heartbeats



First time read or write

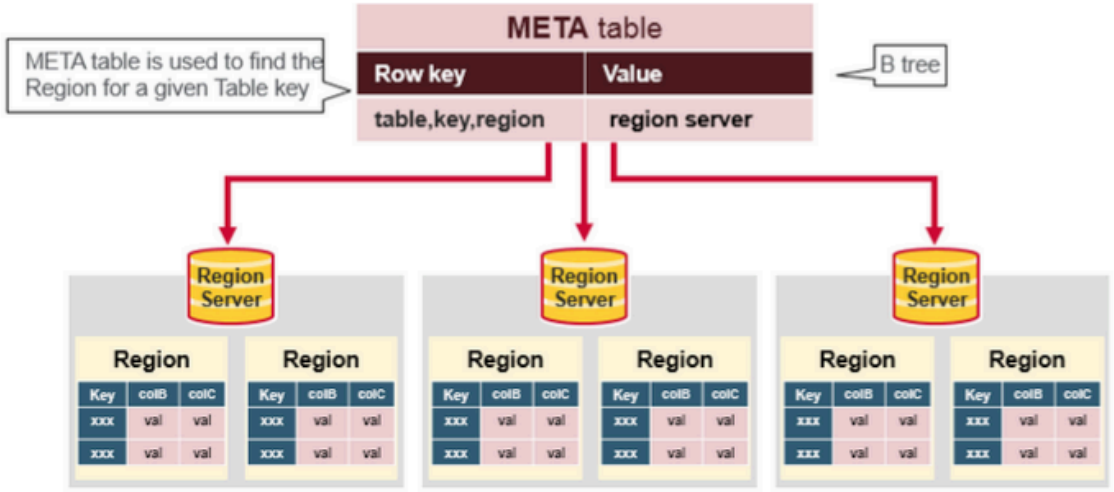
- A special HBase Catalog table called the META table, which holds the location of the regions in the cluster
- ZooKeeper stores the location of the META table

1. The client gets the Region server that hosts the META table from ZooKeeper
2. The client will query the .META. 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
3. It will get the Row from the corresponding Region Server



META table

- It is an HBase table that keeps a list of all regions in the system
- The .META. table is like a *b*-tree.
- The .META. table structure is as follows:
  - ❑ Key: region start key, region id
  - ❑ Value: RegionServer



For future reads…

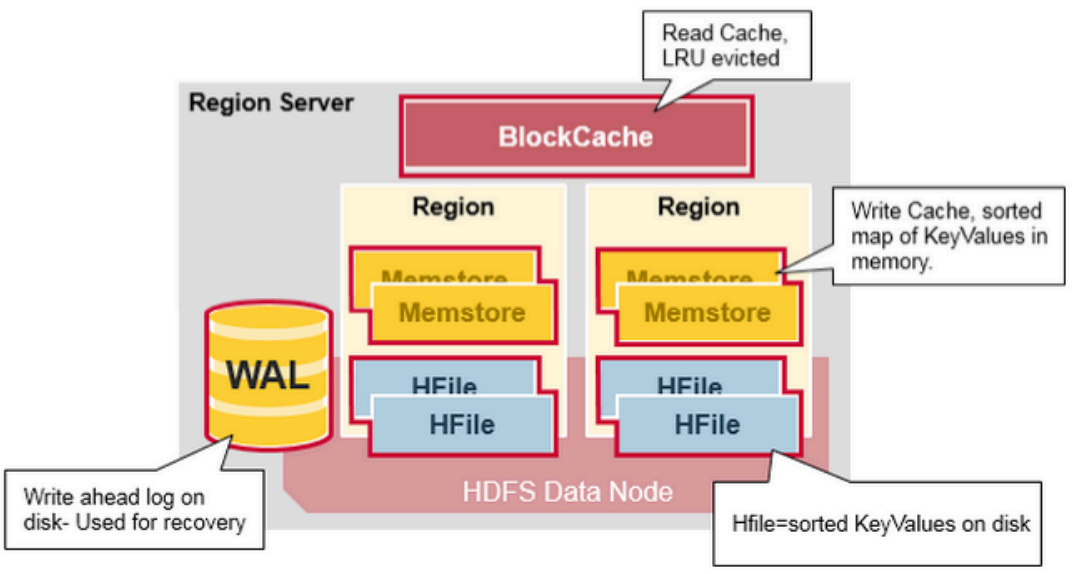
UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

- For future reads, the client uses the cache to retrieve the META location and previously read row keys
- Over time, it does not need to query the META table, unless there is a miss because a region has moved; then it will re-query and update the cache

Region Server components

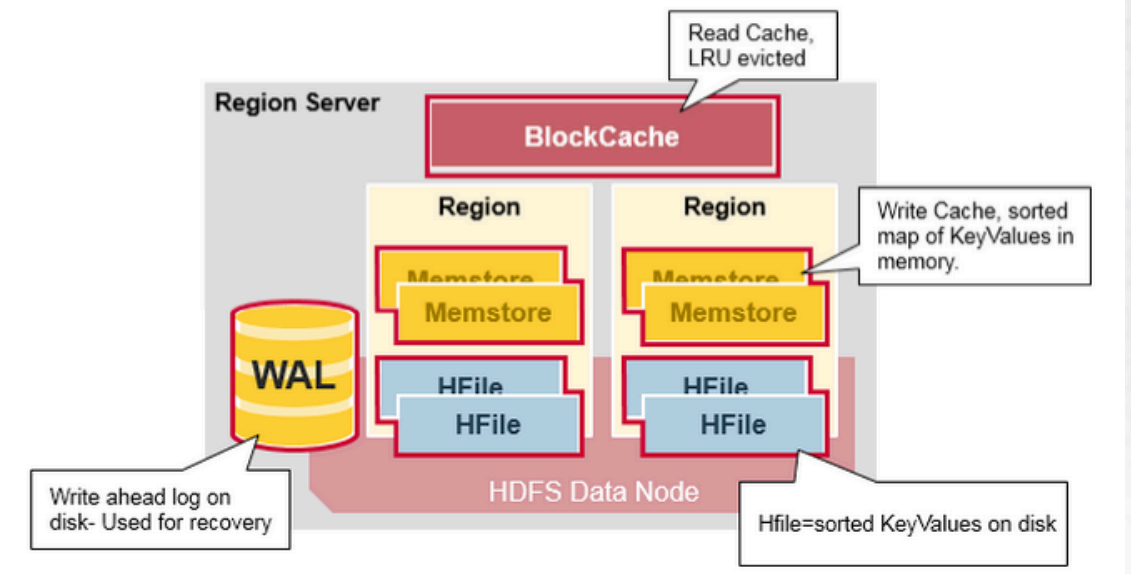
UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

- BlockCache: is the read cache. It stores frequently read data in memory. Least Recently Used data is evicted when full
- **WAL** (write ahead log): used to store new data that hasn't yet been persisted to permanent storage; it is used for recovery in the case of failure



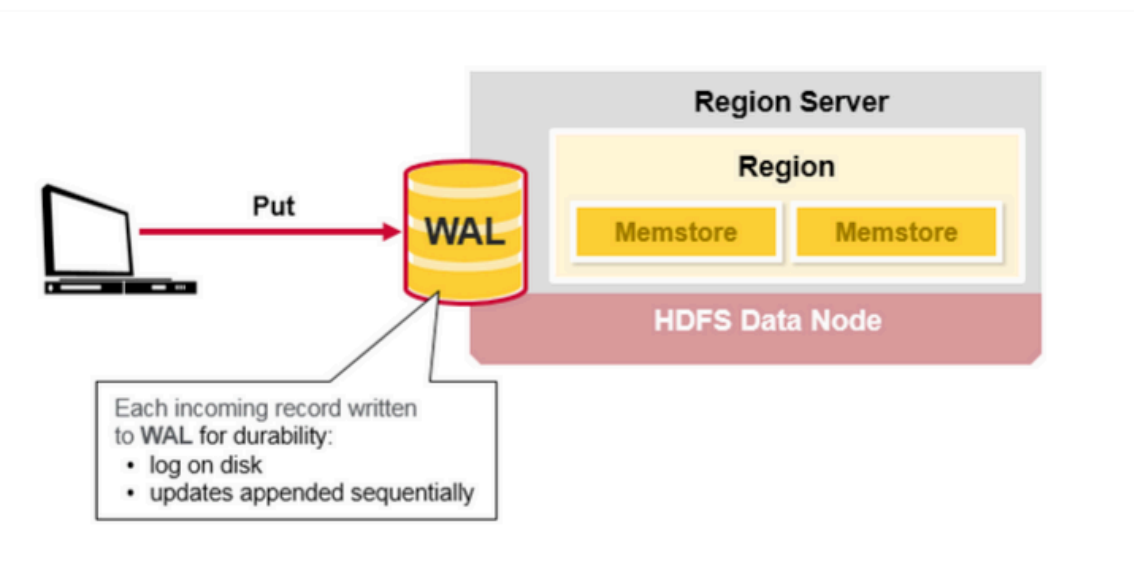
Region Server components

- **MemStore**: is the write cache. It stores new data which has not yet been written to disk. It is sorted before writing to disk
  - There is one MemStore per column family per region
- **HFile**: stores the rows as sorted KeyValue on disk



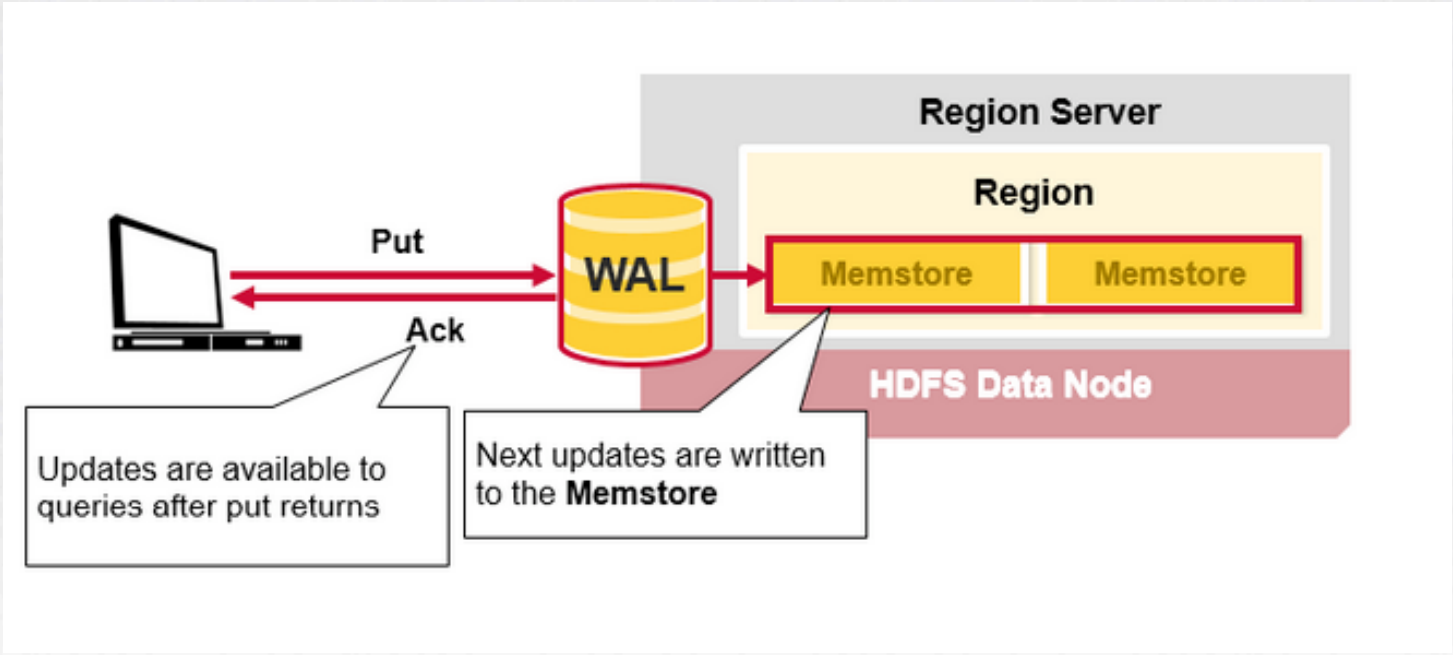
HBase write step (1)

- When the client issues a Put request, the first step is to write the data to WAL:
  - ❑ Edits are appended to the end of WAL file that is on disk
  - ❑ WAL is used to recover not-yet-persisted in case a server failure



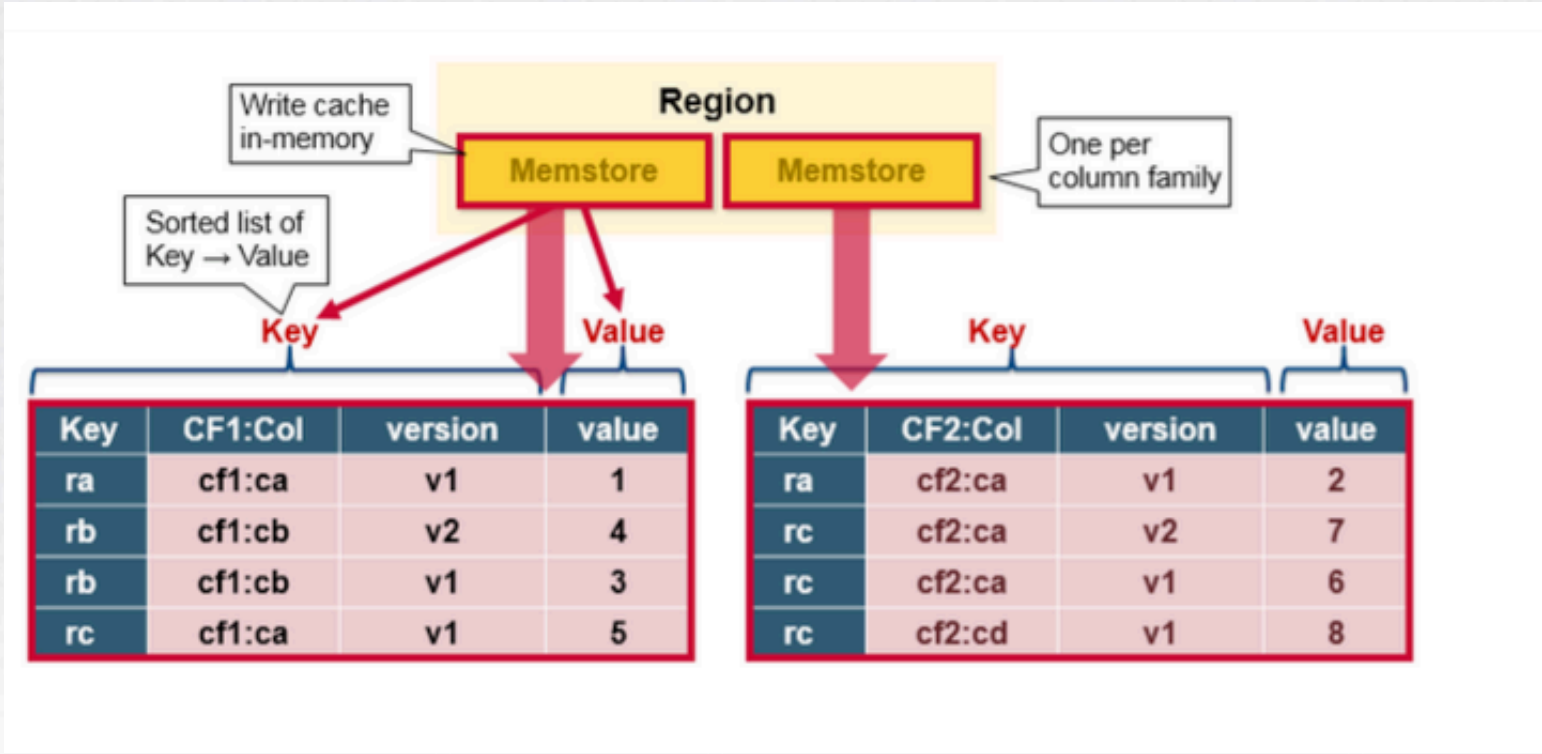
HBase write step (2)

- Once the data is written to WAL, it is placed in the MemStore. Then the put request acknowledge returns to the client



MemStore

- The MemStore stores updates in memory as sorted KeyValues, the same as it would be stored in an HFile.

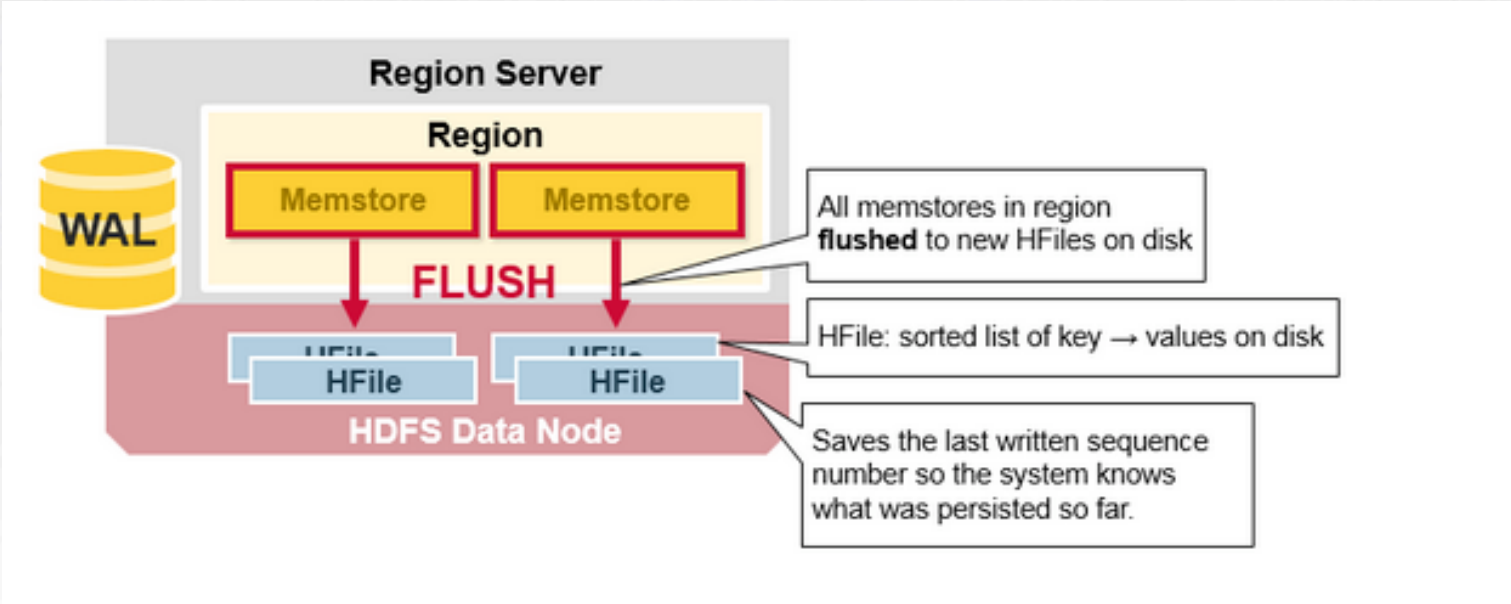


Key	CF1:Col	version	value
ra	cf1:ca	v1	1
rb	cf1:cb	v2	4
rb	cf1:cb	v1	3
rc	cf1:ca	v1	5

Key	CF2:Col	version	value
ra	cf2:ca	v1	2
rc	cf2:ca	v2	7
rc	cf2:ca	v1	6
rc	cf2:cd	v1	8

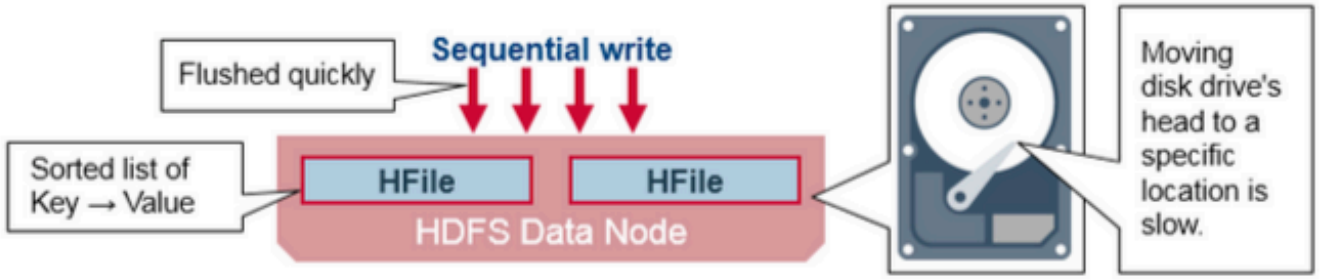
Region flush

- When MemStore accumulates enough data, the entire sorted set is sequentially written to a new HFile in HDFS
- This is one reason why there is a limit to the number of column families in HBase



HFile

- HFile contains sorted key/values
- Sequential write
  - It is fast, as it avoids moving the disk drive head



Key Value

Key	CF1:Col	version	value
ra	cf1:ca	v1	1
rb	cf1:cb	v2	4
rb	cf1:cb	v1	3
rc	cf1:ca	v1	5

Key Value

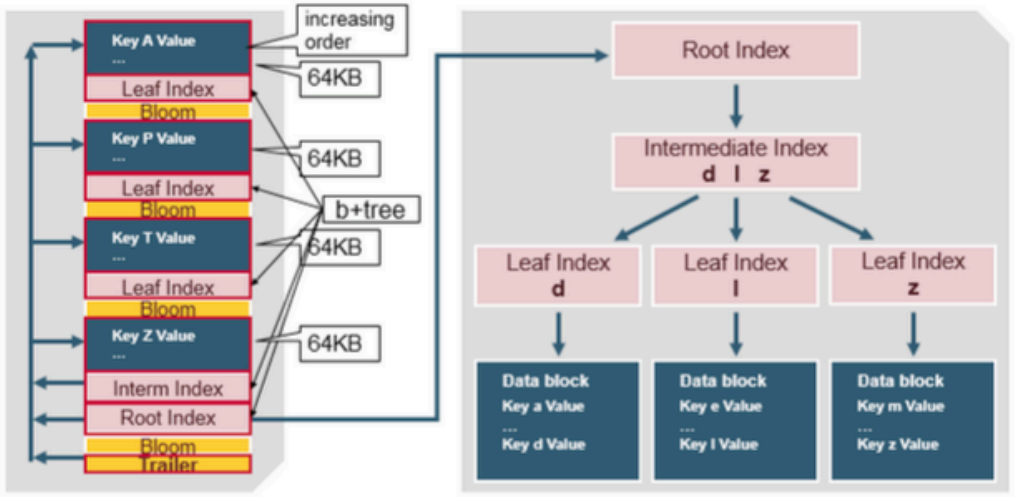
Key	CF2:Col	version	value
ra	cf2:ca	v1	2
rc	cf2:ca	v2	7
rc	cf2:ca	v1	6
rc	cf2:cd	v1	8

HFile structure

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

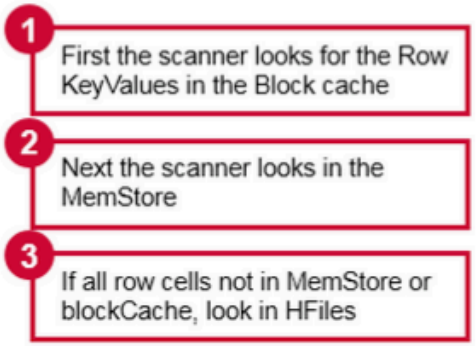
- HFile contains a muti-layered index which allows Hbase to seek to the data without having to read the entire file.
- The multi-level index is like a *b*+ tree

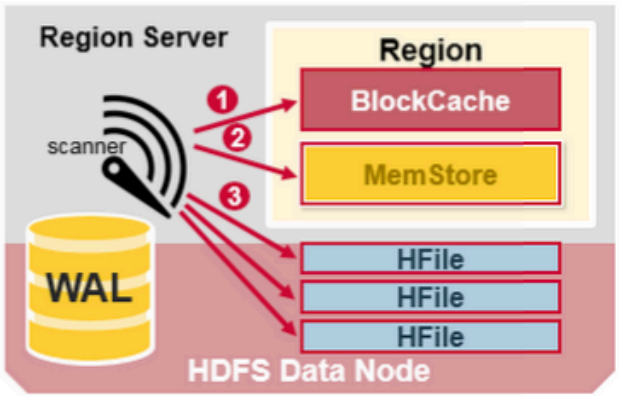
- Key/Value pairs are stored in increasing order
- Indexes point by row key to the key/value data in 64KB "blocks"
- Each block has its own leaf-index
- The last key of each block is put in the intermediate index
- The root index points to the intermediate index



HBase read merge

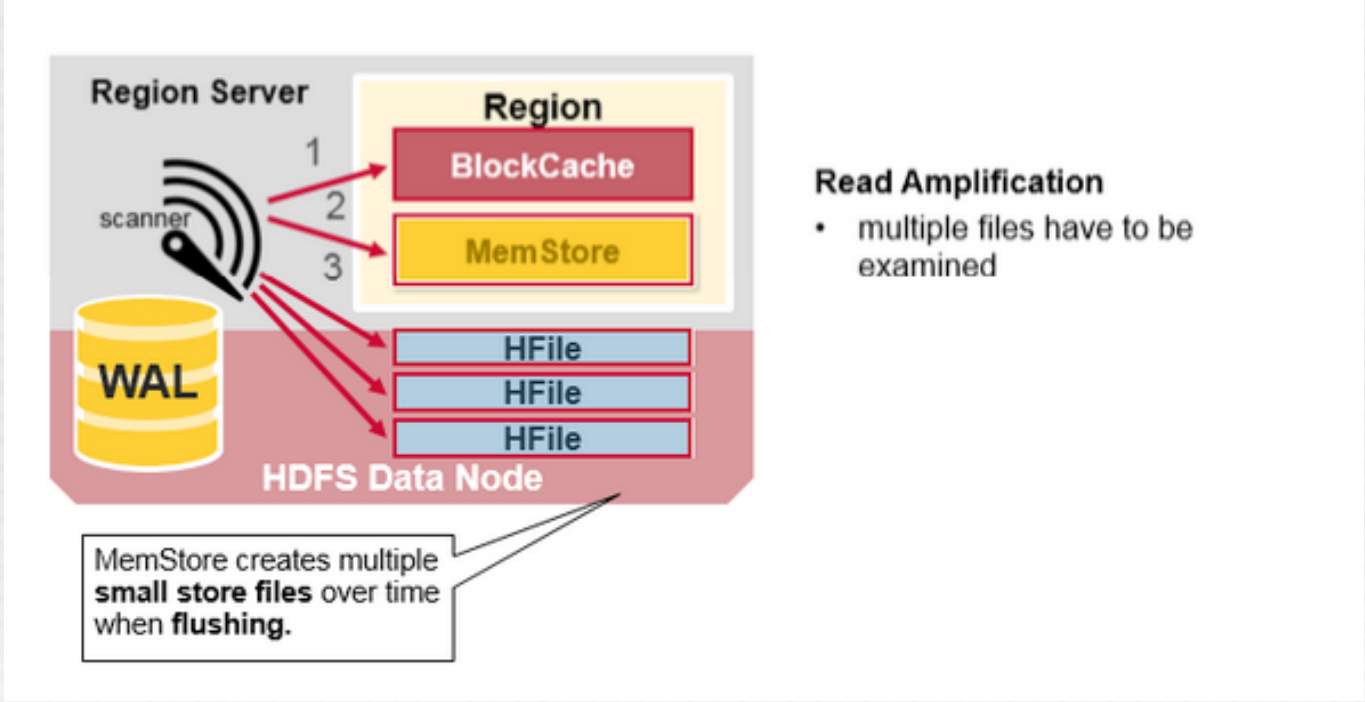
- KeyValue cells corresponding to one row can be in multiple places
  - ❑ HFile, MemStore, BlockCache. How does the system get the cells to return?
    1. First, the scanner looks for the Row cells in the Block cache - the read cache. Recently Read Key Values are cached here, and Least Recently Used are evicted when memory is needed.
    2. Next, the scanner looks in the MemStore, the write cache in memory containing the most recent writes.
    3. If the scanner does not find all of the row cells in the MemStore and Block Cache, then HBase will load HFiles into memory, which may contain the target row cells.





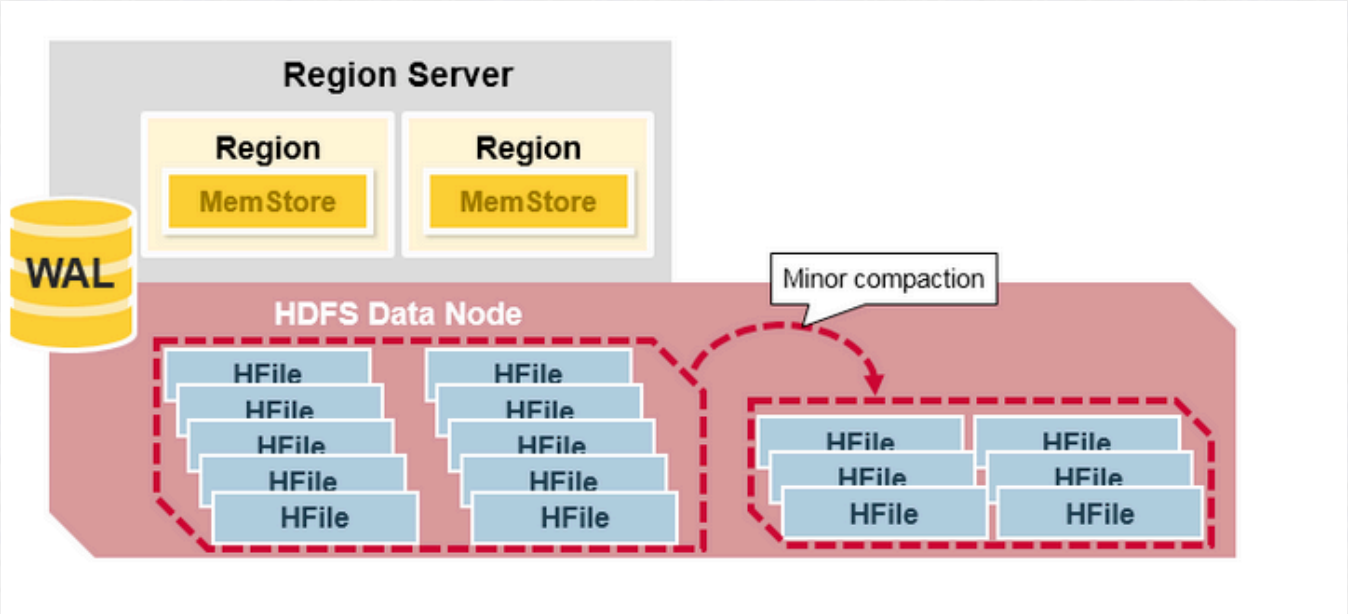
Read amplification

- There may be many HFiles per MemStore, which means for a read, multiple files may have to be examined, which can affect the performance



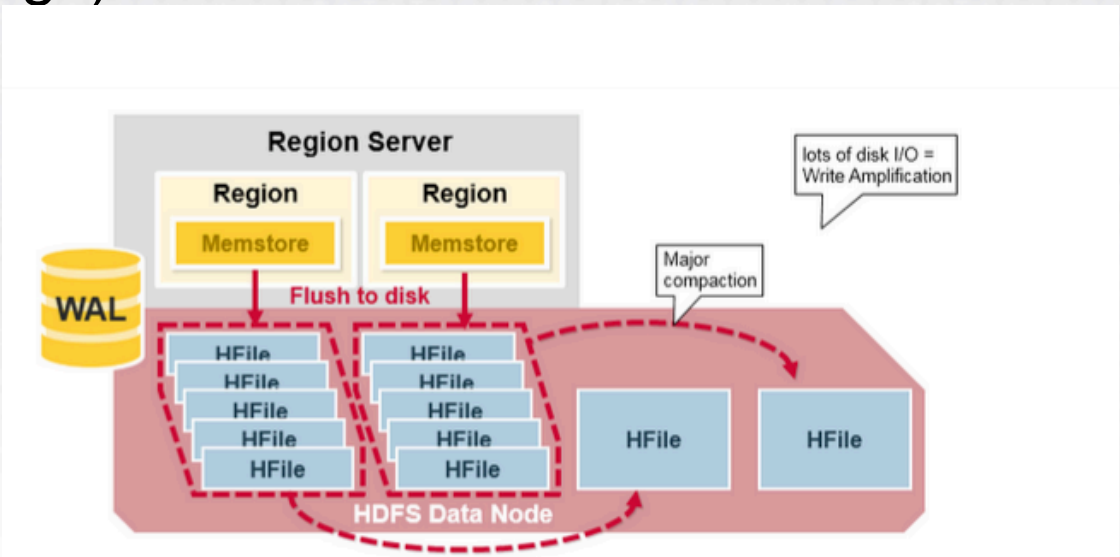
Minor compaction

- Hbase automatically pick some smaller HFiles and rewrite them into fewer bigger HFiles

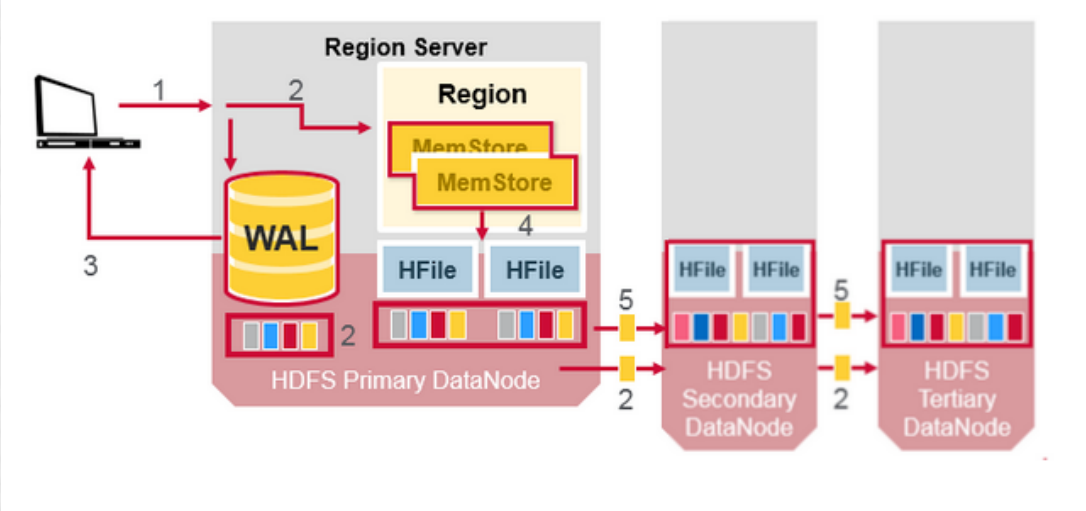


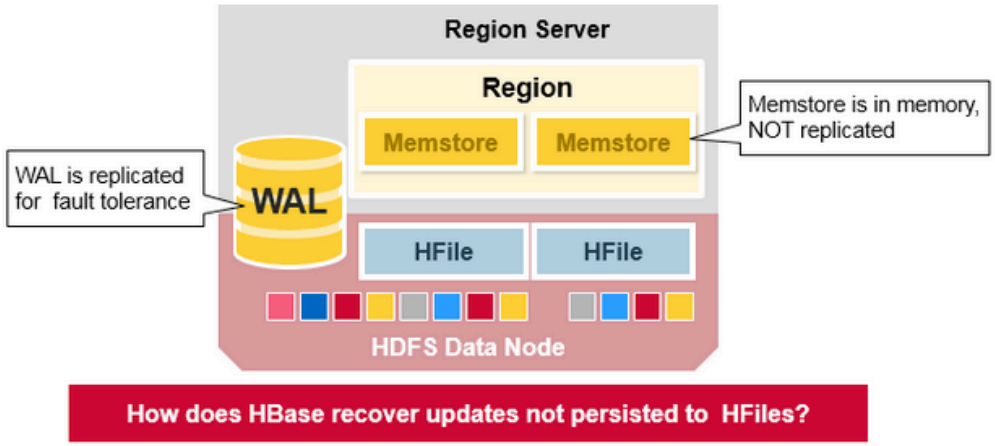
Major compaction

- Major compaction merges and rewrites all the HFiles in a region to one HFile per column family, and in the process, drops deleted or expired cells
- It improves read performance, but involves lots of disk I/O and network traffic
- It can be scheduled to run automatically (usually weekends or evenings)



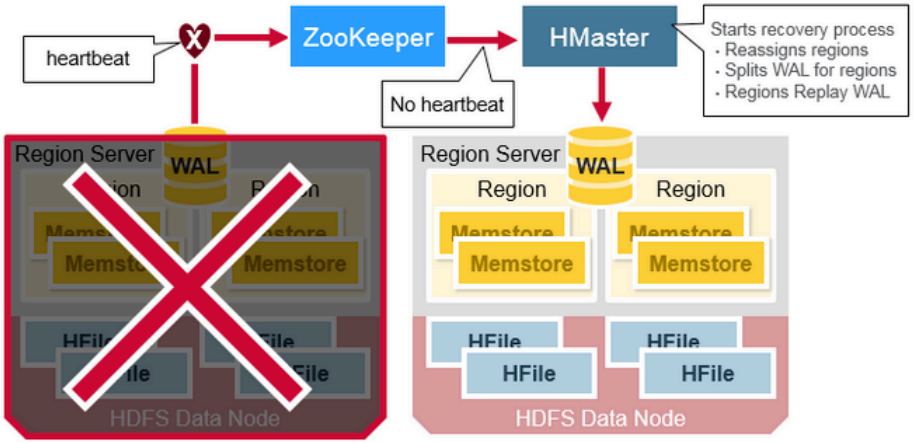
HDFS data replication





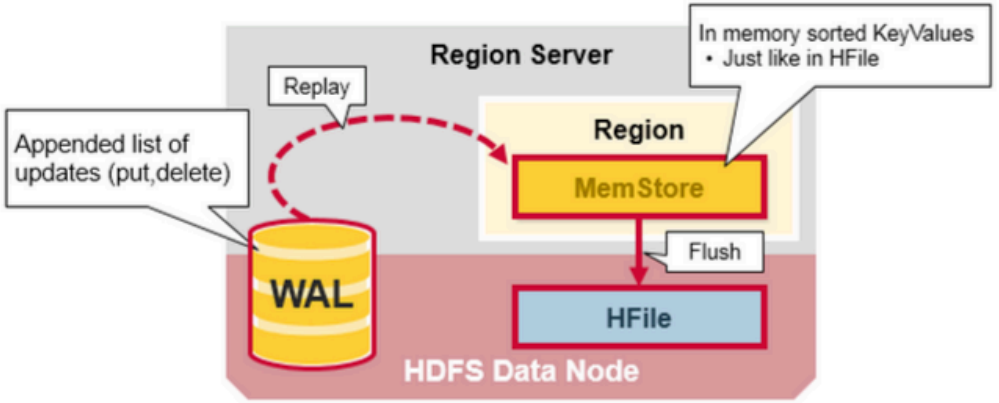
HBase crash recovery

- When the HMaster detects that a region server has crashed, the HMaster re-assigns the regions from the crashed server to active Region Servers
- The HMaster splits the WAL belonging to the crashed region server into separate files and stores these file in the new region servers' data nodes. Each Region Server then replays the WAL from the respective split WAL, to rebuild the MemStore for that region



WAL replay

- WAL files contain a list of edits, with one edit representing a single put or delete. Edits are written chronologically, so, for persistence, additions are appended to the end of the WAL file that is stored on disk
- Replaying a WAL is done by reading the WAL, adding and sorting the contained edits to the current MemStore. At the end, the MemStore is flush to write changes to an HFile



Install HBase (pseudo mode)

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

- Make sure you already install Hadoop (e.g., 1.0.3)

- Download Hbase (0.98)
  - ❑ http://apache.mirrors.hoobly.com/hbase/
  - ❑ Unzip it as hbase-0.98/
- Edit your .bashrc or .bash_profile to add HBASE_HOME and HBase_CONF_DIR
  - ❑ export HBASE_HOME=path_to_your_unzipped_hbase
  - ❑ export HBASE_CONF_DIR=$HBASE_HOME/conf
- DON'T forget to source it

conf/hbase-env.sh

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

hbase-0.98.16-hadoop — vim — 140×36

vim | bash

```
# Set environment variables here.

# This script sets variables multiple times over the course of starting an hbase process,
# so try to keep things idempotent unless you want to take an even deeper look
# into the startup scripts (bin/hbase, etc.)

# The java implementation to use.  Java 1.6 required.
 export JAVA_HOME=$(/usr/libexec/java_home)

# Extra Java CLASSPATH elements.  Optional.
export HBASE_CLASSPATH=/Users/DoubleJ/Software/hadoop-1.0.3/conf

# The maximum amount of heap to use, in MB. Default is 1000.
# export HBASE_HEAPSIZE=1000

# Extra Java runtime options.
# Below are what we set by default.  May only work with SUN JVM.
# For more on why as well as other possible settings,
# see http://wiki.apache.org/hadoop/PerformanceTuning
export HBASE_OPTS="-XX:+UseConcMarkSweepGC"

# Uncomment one of the below three options to enable java garbage collection logging for the server-side processes.

# This enables basic gc logging to the .out file.
# export SERVER_GC_OPTS="-verbose:gc -XX:+PrintGCDetails -XX:+PrintGCDateStamps"

# This enables basic gc logging to its own file.
# If FILE-PATH is not replaced, the log file(.gc) would still be generated in the HBASE_LOG_DIR .
# export SERVER_GC_OPTS="-verbose:gc -XX:+PrintGCDetails -XX:+PrintGCDateStamps -Xloggc:<FILE-PATH>"

# This enables basic GC logging to its own file with automatic log rolling. Only applies to jdk 1.6.0_34+ and 1.7.0_2+.
# If FILE-PATH is not replaced, the log file(.gc) would still be generated in the HBASE_LOG_DIR .
# export SERVER_GC_OPTS="-verbose:gc -XX:+PrintGCDetails -XX:+PrintGCDateStamps -Xloggc:<FILE-PATH> -XX:+UseGCLogFileRotation -XX:NumberOfGC
LogFiles=1 -XX:GCLogFileSize=512M"
```

conf/hbase-env.sh

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

hbase-0.98.16-hadoop — vim — 140×36

vim | bash

```
# Where log files are stored.  $HBASE_HOME/logs by default.
# export HBASE_LOG_DIR=${HBASE_HOME}/logs

# Enable remote JDWP debugging of major HBase processes. Meant for Core Developers
# export HBASE_MASTER_OPTS="$HBASE_MASTER_OPTS -Xdebug -Xrunjdwp:transport=dt_socket,server=y,suspend=n,address=8070"
# export HBASE_REGIONSERVER_OPTS="$HBASE_REGIONSERVER_OPTS -Xdebug -Xrunjdwp:transport=dt_socket,server=y,suspend=n,address=8071"
# export HBASE_THRIFT_OPTS="$HBASE_THRIFT_OPTS -Xdebug -Xrunjdwp:transport=dt_socket,server=y,suspend=n,address=8072"
# export HBASE_ZOOKEEPER_OPTS="$HBASE_ZOOKEEPER_OPTS -Xdebug -Xrunjdwp:transport=dt_socket,server=y,suspend=n,address=8073"

# A string representing this instance of hbase. $USER by default.
# export HBASE_IDENT_STRING=$USER

# The scheduling priority for daemon processes.  See 'man nice'.
# export HBASE_NICENESS=10

# The directory where pid files are stored. /tmp by default.
# export HBASE_PID_DIR=/var/hadoop/pids

# Seconds to sleep between slave commands.  Unset by default.  This
# can be useful in large clusters, where, e.g., slave rsyncs can
# otherwise arrive faster than the master can service them.
# export HBASE_SLAVE_SLEEP=0.1

# Tell HBase whether it should manage it's own instance of Zookeeper or not.
 export HBASE_MANAGES_ZK=true

# The default log rolling policy is RFA, where the log file is rolled as per the size defined for the
# RFA appender. Please refer to the log4j.properties file to see more details on this appender.
# In case one needs to do log rolling on a date change, one should set the environment property
# HBASE_ROOT_LOGGER to "<DESIRED_LOG LEVEL>,DRFA".
# For example:
# HBASE_ROOT_LOGGER=INFO,DRFA
# The reason for changing default to RFA is to avoid the boundary case of filling out disk space as
# DRFA doesn't put any cap on the log size. Please refer to HBase-5655 for more context.
```

conf/hbase-site.xml

UNIVERSITY OF MARYLAND
ROBERT H. SMITH SCHOOL OF BUSINESS

hbase-0.98.16-hadoop — vim — 140×36

vim | bash

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<configuration>
        <property>
                <name>hbase.cluster.distributed</name>
                <value>true</value>
        </property>
        <property>
                <name>hbase.rootdir</name>
                <value>hdfs://localhost:8020/hbase</value>
                <description>The directory shared by region servers</description>
        </property>

        <property>
                <name>hbase.zookeeper.quorum</name>
                <value>localhost</value>
        </property>

        <property>
                <name>hbase.zookeeper.property.clientPort</name>
                <value>2181</value>
        </property>

</configuration
~
~
```