Hbase在酷狗的实践应用

王 劲

2015.6

酷狗音乐

目录

Hbase系统架构

Hbase实践应用

概述

HBase是Apache Hadoop的数据库，能够对大型数据提供随机、实时的读写访问。HBase的目标是存储并处理大型的数据。HBase是一个开源的，分布式的，多版本的，面向列的存储模型。它存储的是松散型数据。

Hbase表的特点

1 大：一个表可以有上亿行，上百万列

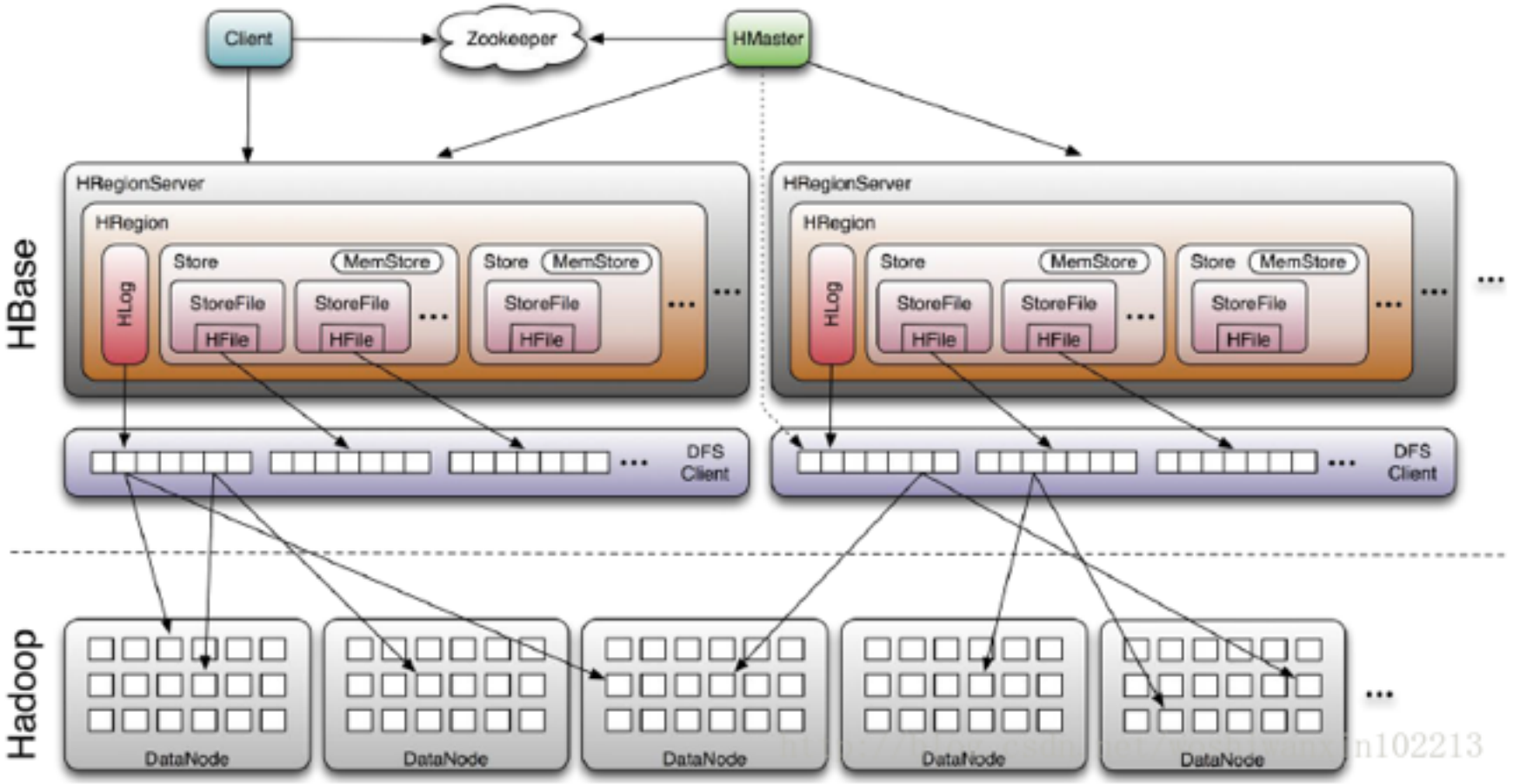
2 面向列:面向列(族)的存储和权限控制，列(族)独立检索。

3 稀疏:对于为空(null)的列，并不占用存储空间，因此，表可以设计的非常稀疏。

逻辑视图

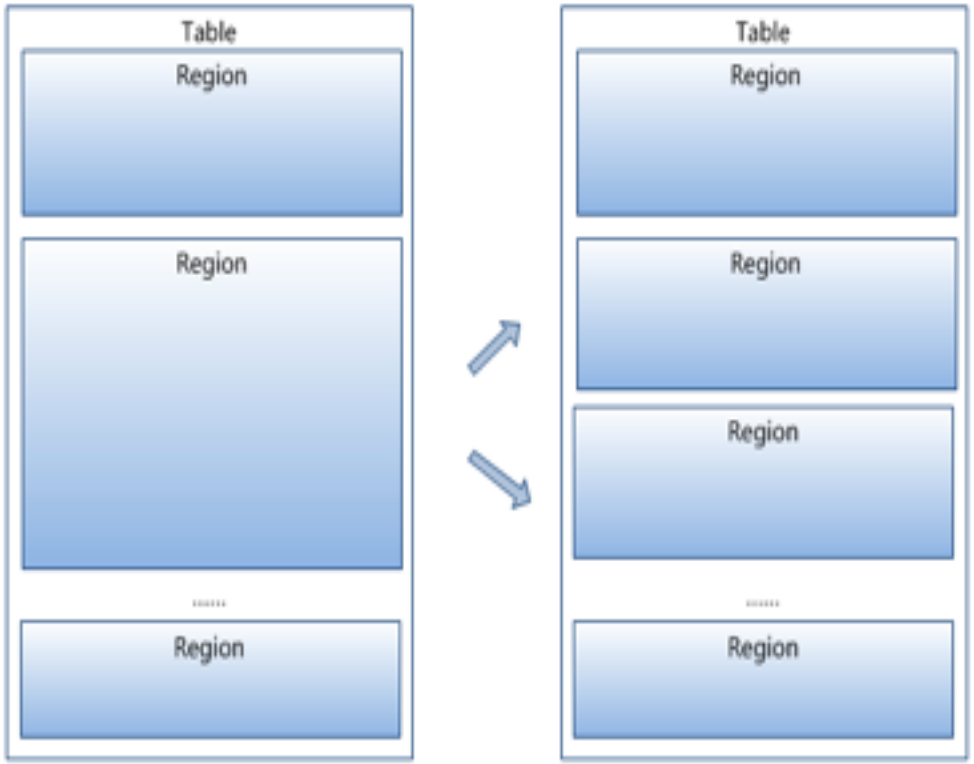
Row Key	Timestamp	Column Family	
		URI	Parser
r1	t3	url=http://www.taobao.com	title=天天特价
	t2	host=taobao.com	
	t1		
r2	t5	url=http://www.alibaba.com	content=每天...
	t4	host=alibaba.com	

物理存储


Client
Zookeeper
HMaster
HBase
HRegionServer
HRegion
HLog
Store
MemStore
StoreFile
HFile
DFS Client
Hadoop
DataNode

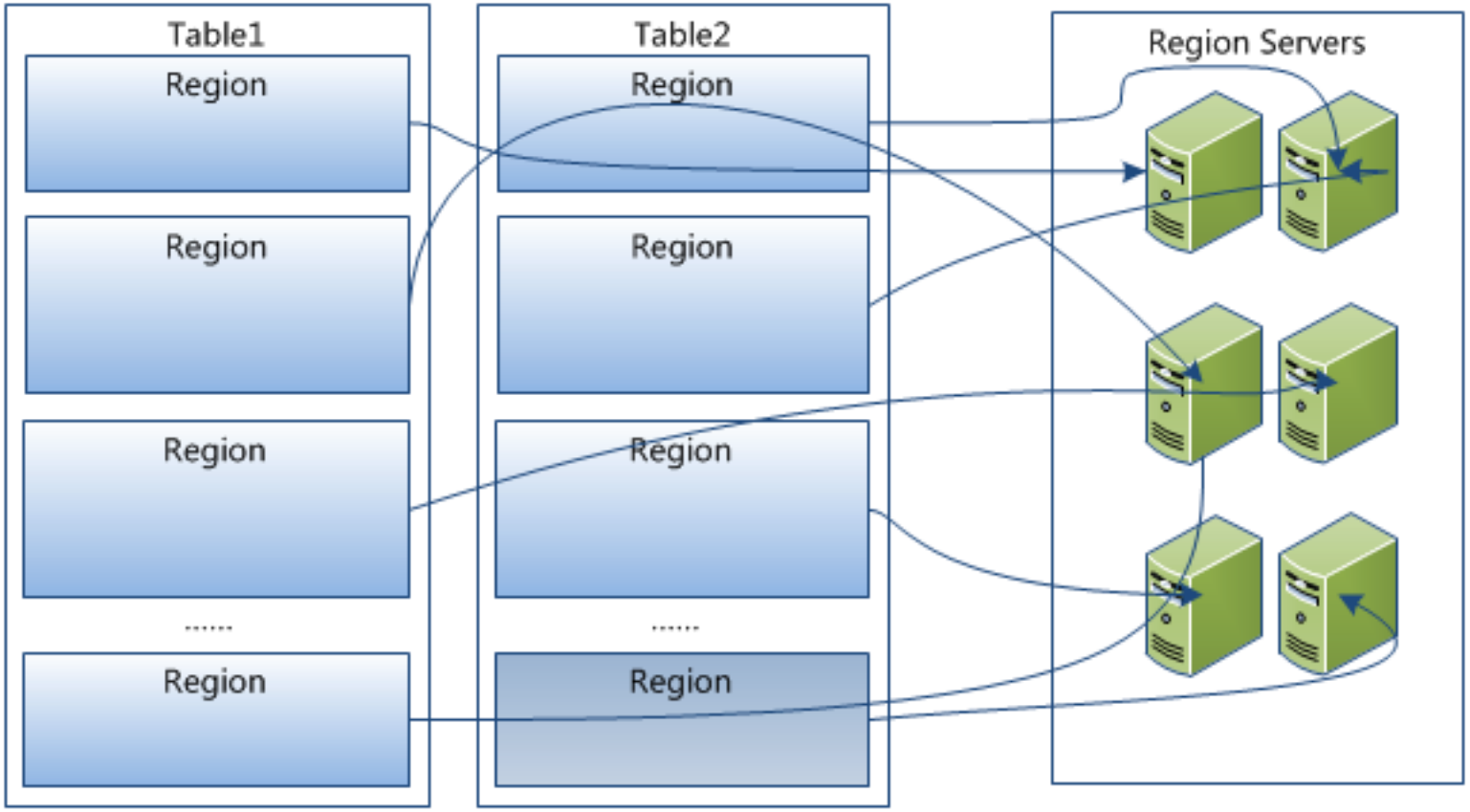
HRegion

图一

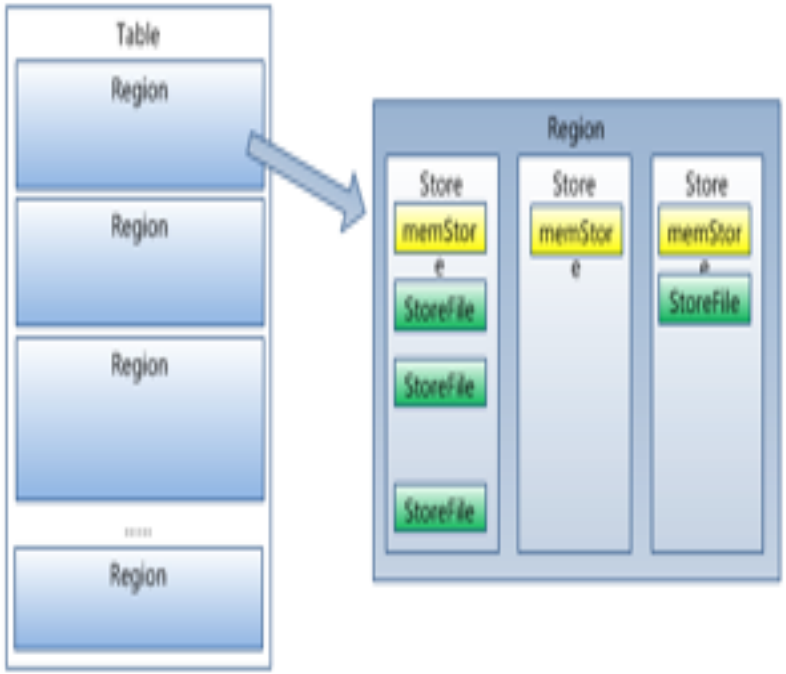


图二

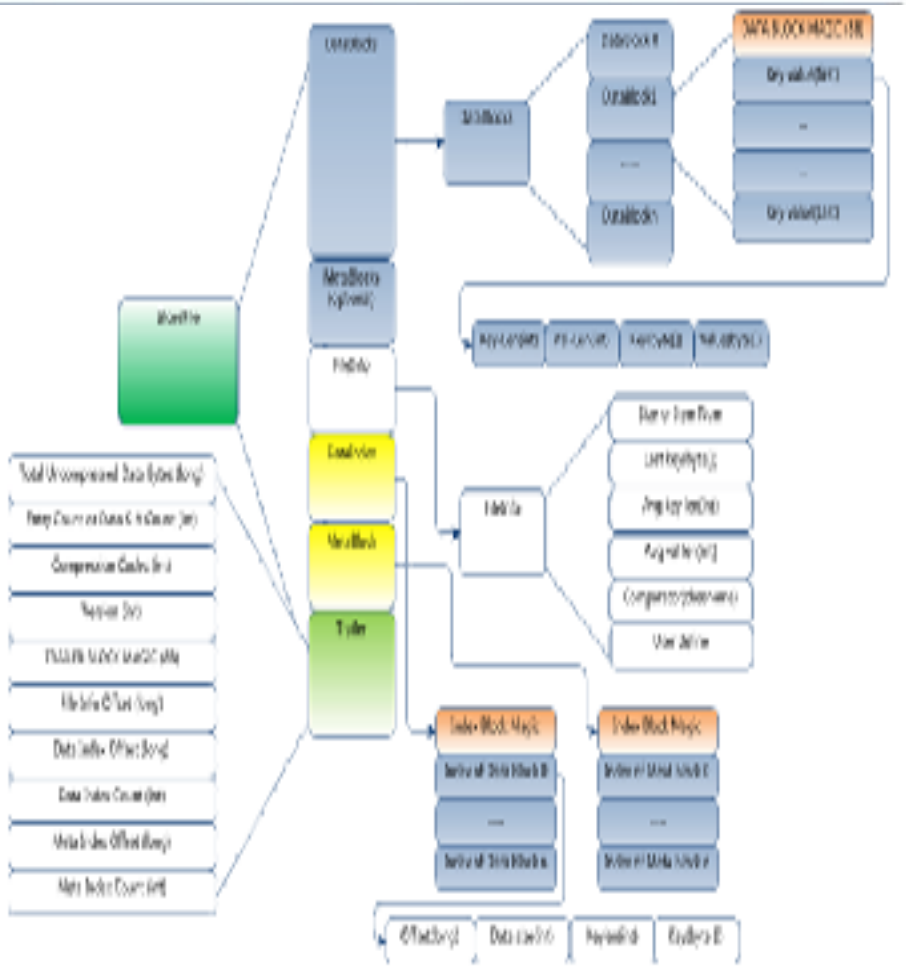
HRegion



HFile

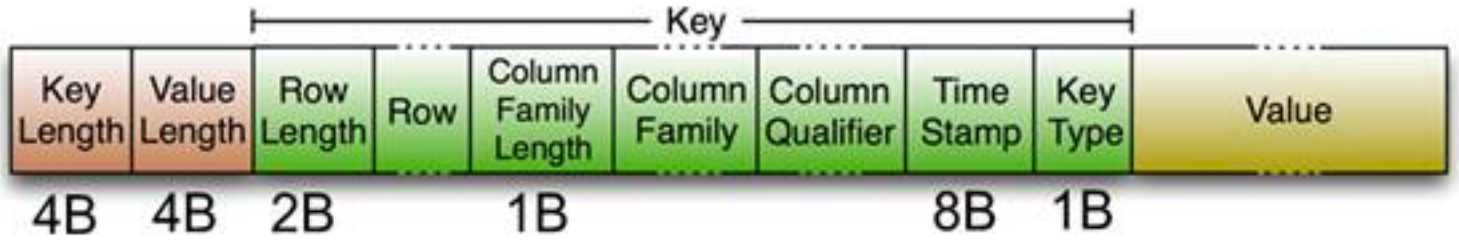


图一

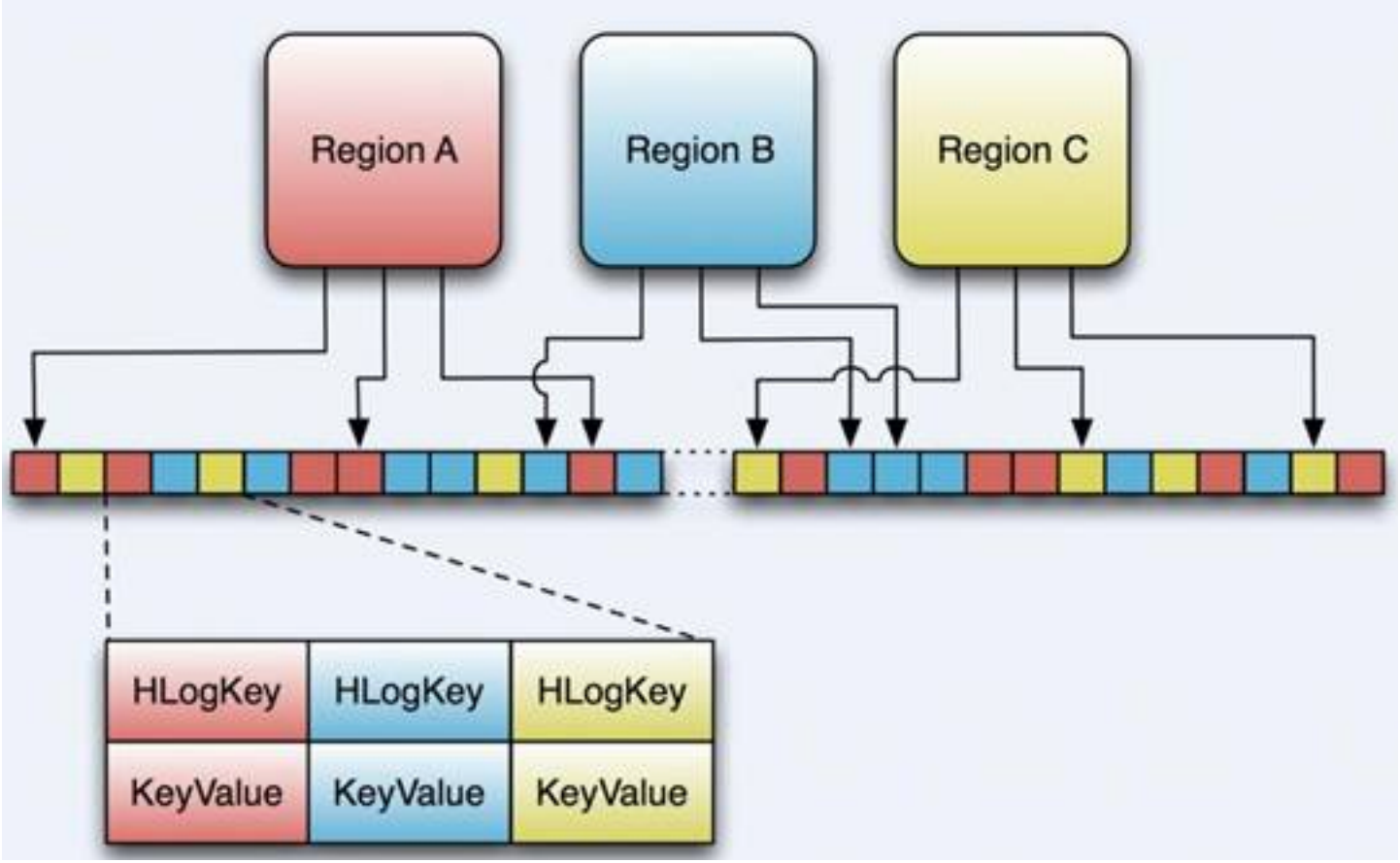


图二

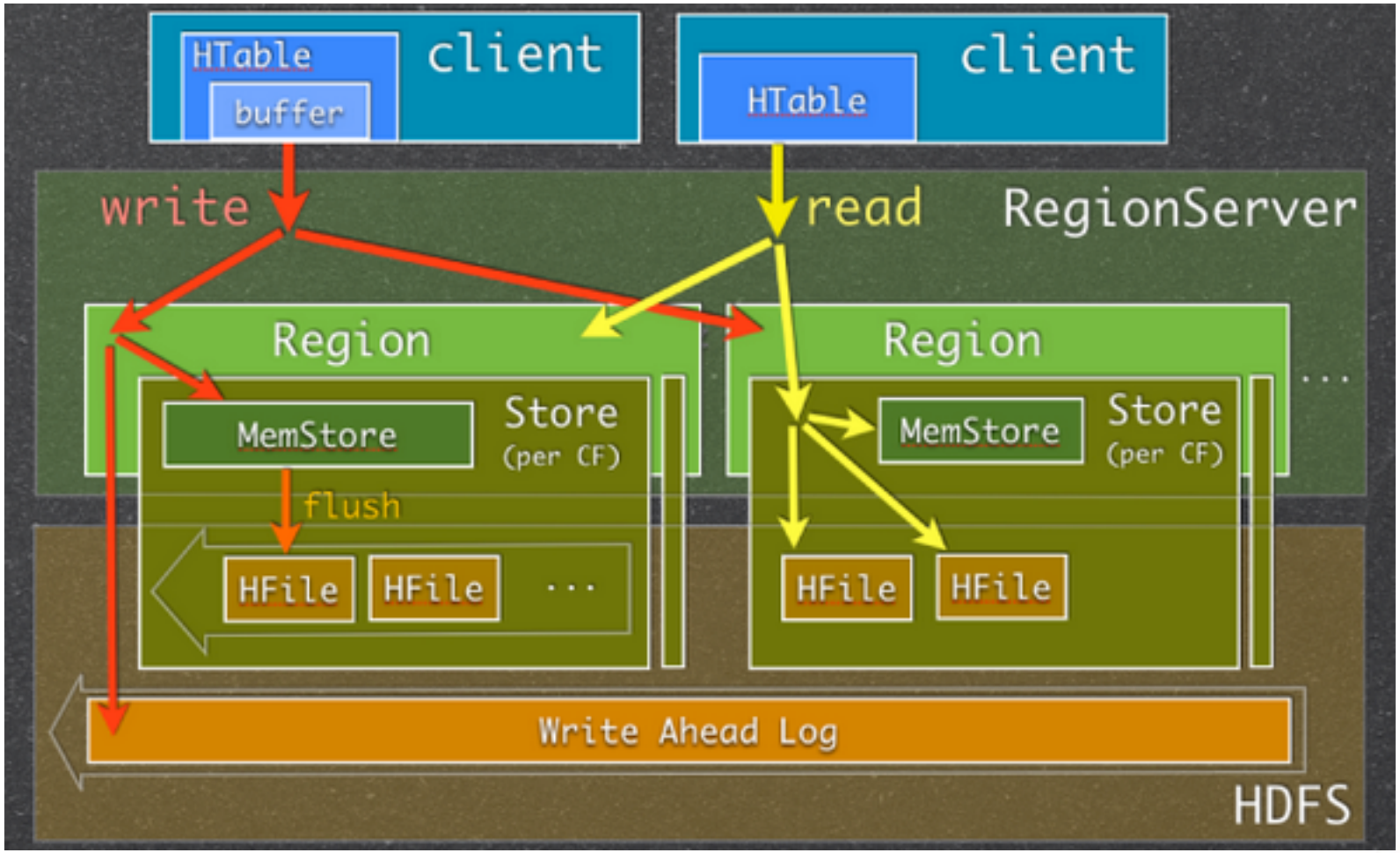
KeyValue



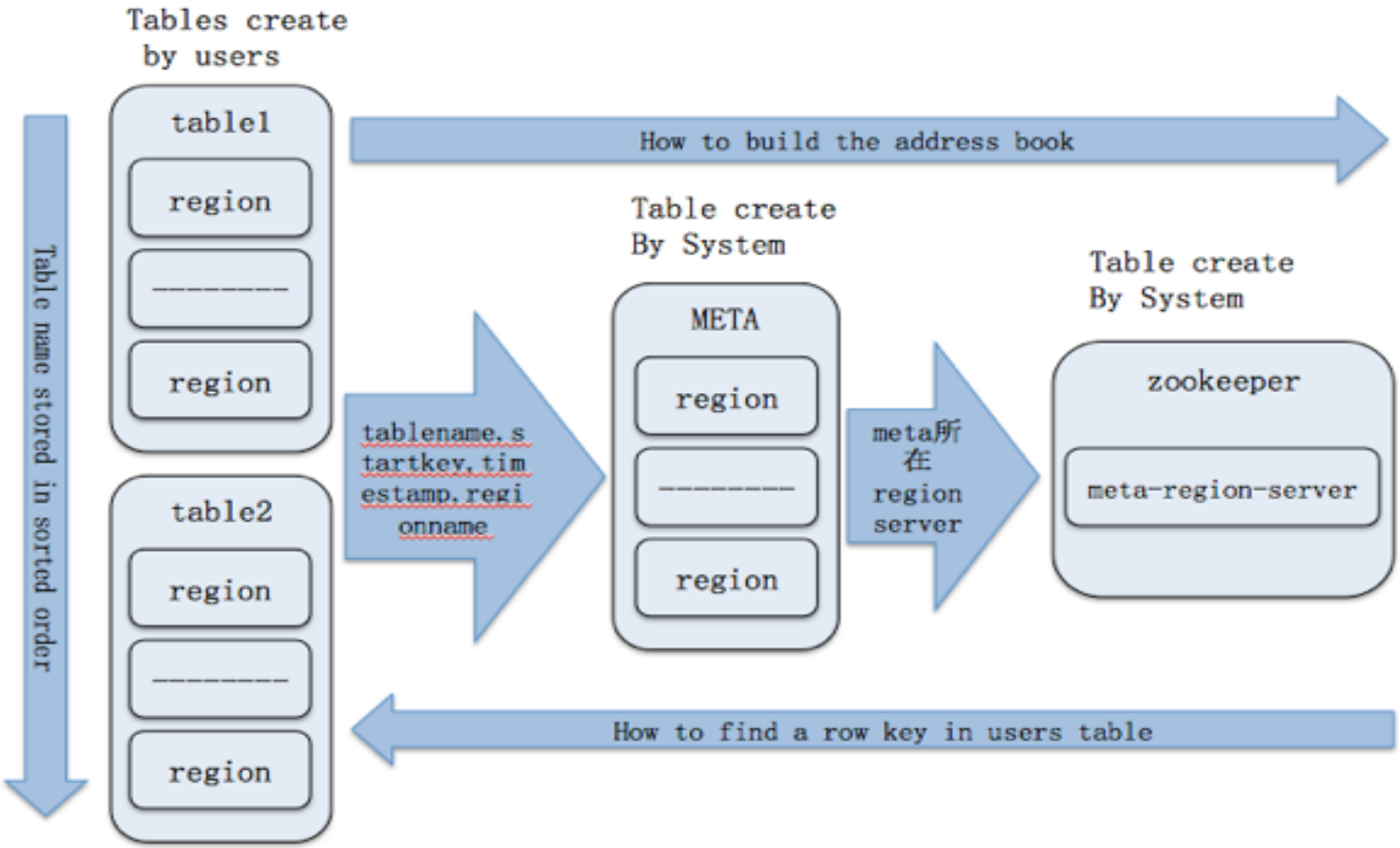
HLog File



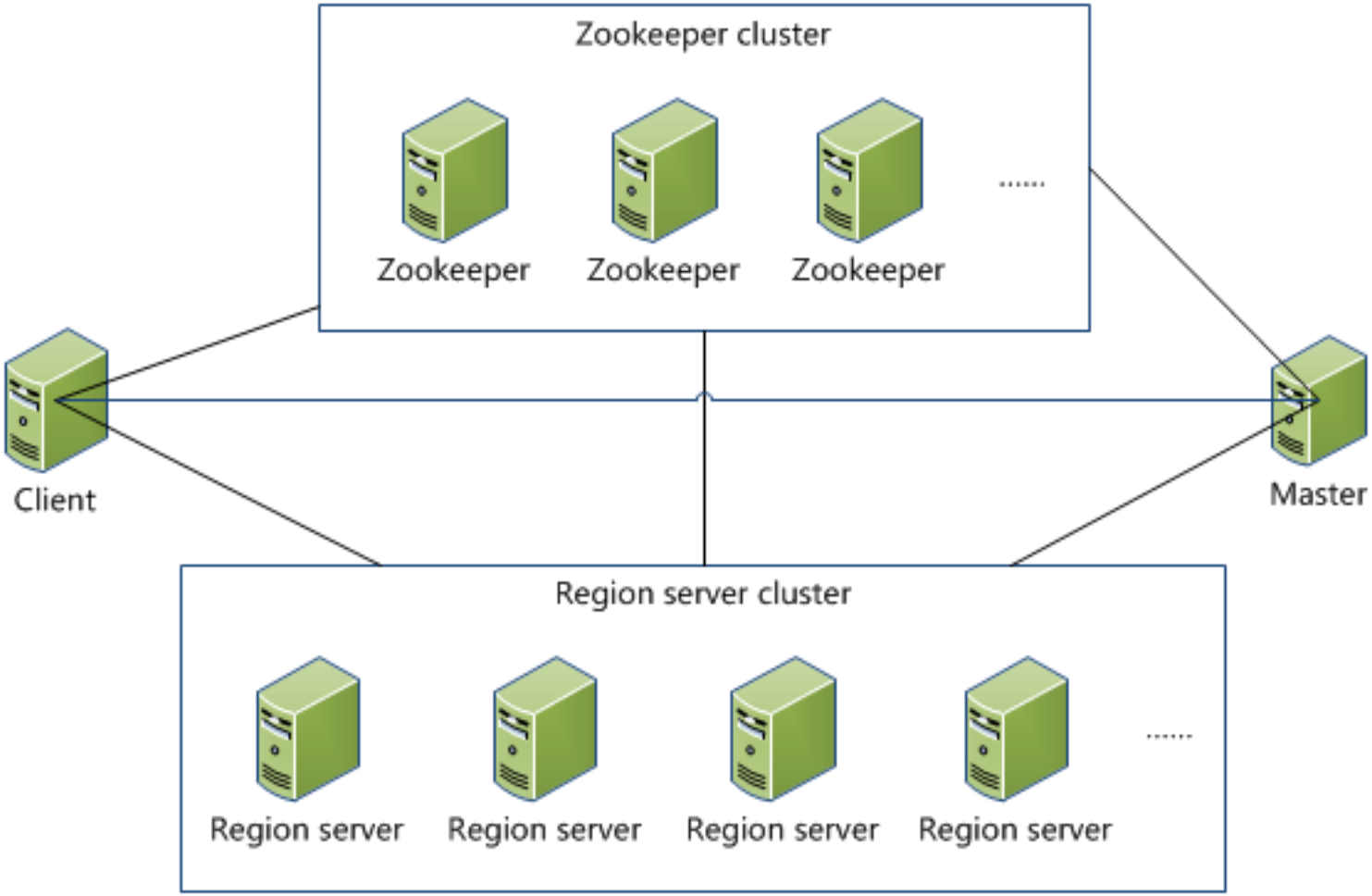
读写过程


HTable
buffer
client
HTable
client
write
read
RegionServer
Region
Region
MemStore
Store
(per CF)
MemStore
Store
(per CF)
flush
HFile
HFile
...
HFile
HFile
Write Ahead Log
HDFS

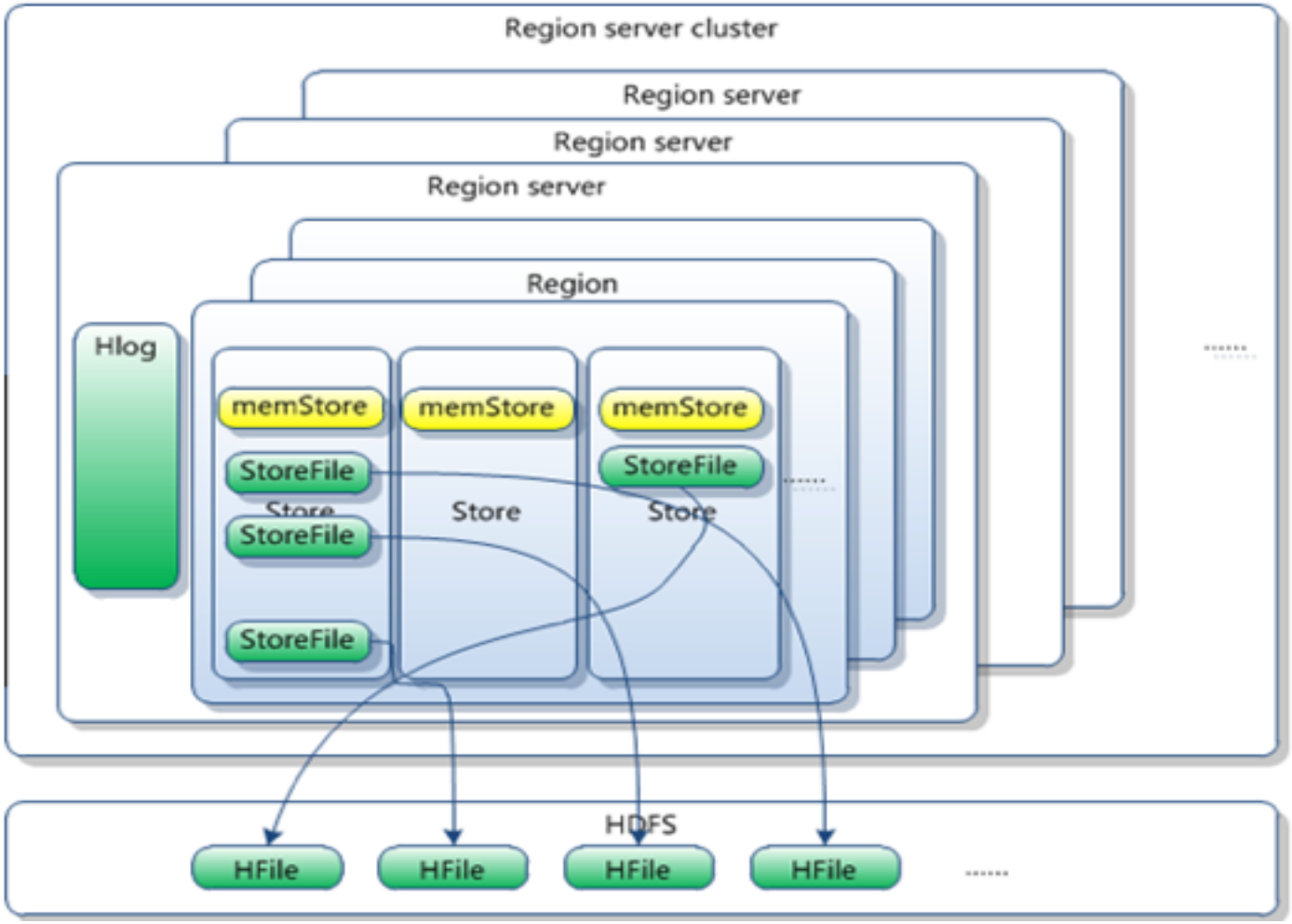
Regin定位



部署架构


Zookeeper cluster
Zookeeper
Zookeeper
Zookeeper
.....
Client
Master
Region server cluster
Region server
Region server
Region server
Region server
.....

逻辑架构


Region server cluster
Region server
Region server
Region server
Region
Hlog
memStore
memStore
memStore
StoreFile
StoreFile
StoreFile
StoreFile
Store
Store
Store
HDFS
HFile
HFile
HFile
HFile

Hbase应用场景

★ 大数据量存储，大数据量高并发操作

★ 需要对数据随机读写操作

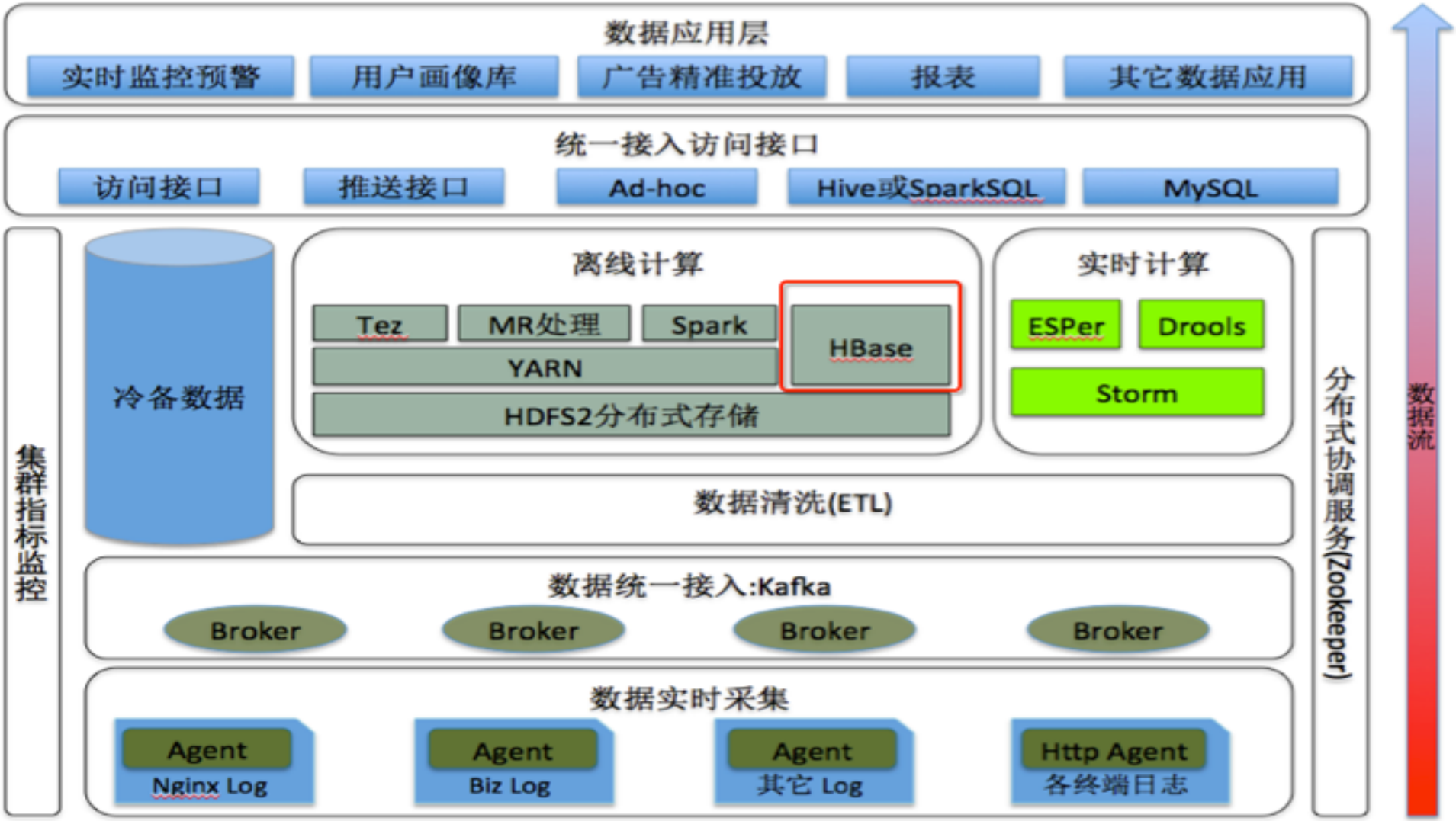
★ 读写访问均是非常简单的操作

目录

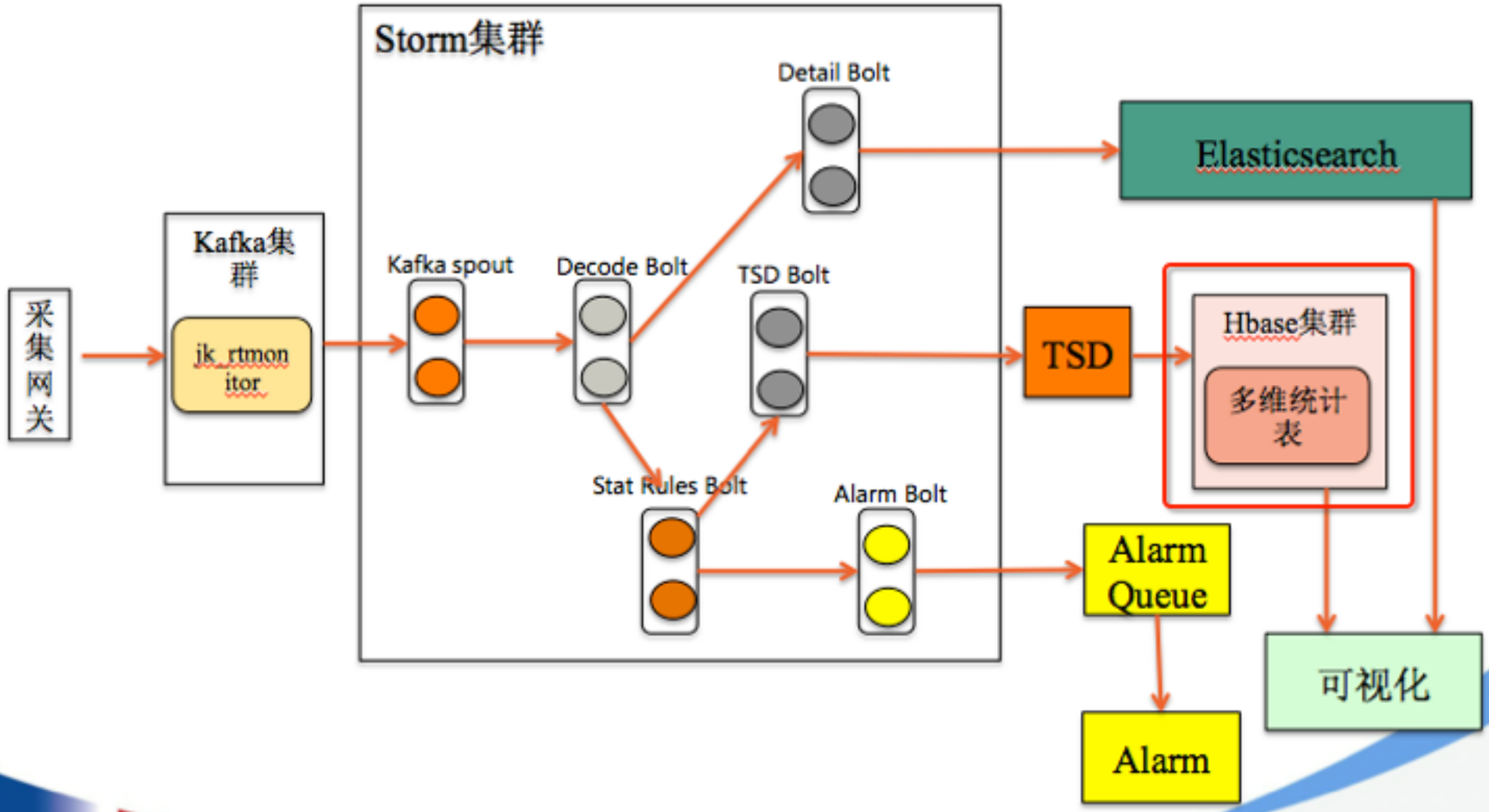
Hbase系统架构

Hbase实践应用

Hbase应用



Hbase应用



系统参数优化

1、Memory性能调整

控制swappness=0 参数，尽量减少应用的内存被交换到交换分区中，默认是60

2、I/O性能调整

a、修改I/O调度算法。

Linux已知的I/O调试算法有4种:

deadline - Deadline I/O scheduler

as - Anticipatory I/O scheduler

cfq - Complete Fair Queuing scheduler

noop - Noop I/O scheduler

b、文件系统调整(ext4、ext3)

c、文件系统Mount时可加入选项noatime、nodiratime。

#vi /etc/fstab (编辑该文件)

/dev/sda1 / ext4 defaults,noatime 0 0

d、调整块设备的READAHEAD，调大RA值。

blockdev --setra 1024 /dev/sda（设置预读取缓存）

3、CPU调整

修改最大用户进程数(max user processes)

系统参数优化

4、网络调整

net.core.somaxconn=65535
net.core.somaxconn是Linux中的一个kernel参数，表示socket监 听（listen）的backlog上限。
net.core.netdev_max_backlog = 8192
每个网络接口接收数据包的速率比内核处理这些包的速率快时，允许送到队列的数据包的最大数目
net.ipv4.tcp_syncookies = 1
表示开启SYN Cookies。当出现SYN等待队列溢出时，启用cookies来处理，可防范少量SYN攻击，默认为0，表
闭；
net.ipv4.tcp_tw_reuse = 1
表示开启重用。允许将TIME-WAIT sockets重新用于新的TCP连接，默认为0，表示关闭；
net.ipv4.tcp_tw_recycle = 1
表示开启TCP连接中TIME-WAIT sockets的快速回收，默认为0，表示关闭。
net.ipv4.tcp_fin_timeout = 15
表示如果套接字由本端要求关闭，这个参数决定了它保持在FIN-WAIT-2状态的时间。
net.ipv4.tcp_keepalive_time = 1800
表示当keepalive起用的时候，TCP发送keepalive消息的频度。缺省是2小时，改为20分钟。
net.ipv4.ip_local_port_range = 1024 65000
表示用于向外连接的端口范围。缺省情况下很小：32768到61000，改为1024到65000。
net.ipv4.tcp_max_syn_backlog = 65535
表示SYN队列的长度，默认为1024，加大队列长度为8192，可以容纳更多等待连接的网络连接数。
net.ipv4.tcp_max_tw_buckets = 5000
表示系统同时保持TIME_WAIT套接字的最大数量。

Hbase优化原则

1、随机读密集型：

优化方向在于有效利用缓存和索引，配置参数包括block块缓存、memstore大小、HFile数据块大小（比如64k，数据块越小，索引粒度越细）、启用Bloom filter等。

2、顺序读密集型：

顺序读一般是大规模扫描，所以缓存不能提升性能。配置参数包括HFile数据块大小（减少硬盘寻道次数）、Scanner- caching（每次扫描返回更多的数据，减少RPC调用次数，可以通过Scan.setCaching(int)控制到每次扫描）和关闭 BlockCache等。

3、写密集型：

提高写性能的秘诀在于减少MemStoreflush、数据合并和分割次数。配置参数包括MemStore大小、

RegionFile大小、分配给MemStore的堆大小、MemStore-Local 缓存和GC参数等。

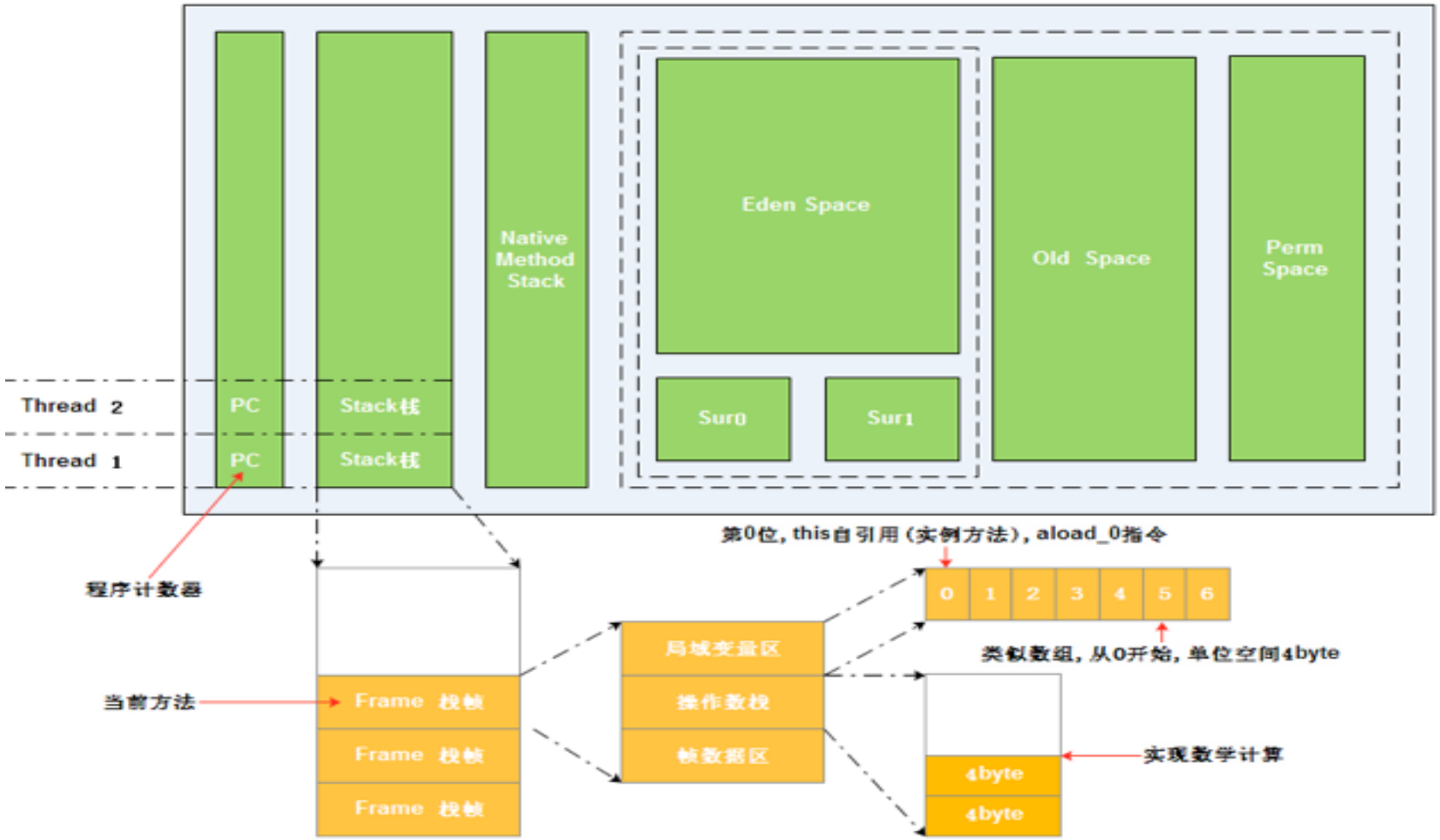
4、混合型：根据应用情况反复调整参数达到性能最优

5、其他影响性能的因素还包括数据压缩、RowKey设计、Major Compactions（建议在集群负载低的时候收工进行）和RegionServer处理RPC请求线程数等。

Hbase集群配置

hbase.hregion.memstore.flush.size	256M
hbase.hregion.memstore.block.multiplier	8
hbase.regionserver.global.memstore.upperLimit	0.4
hbase.regionserver.global.memstore.lowerLimit	0.35
hbase.regionserver.handler.count	50
hbase.hregion.max.filesize	100G
hbase.hstore.compactionThreshold	10
hbase.regionserver.region.split.policy	org.apache.hadoop.hbase.regionserver.ConstantSizeRegionSplitPolicy
hfile.block.cache.size	0.35
hbase.hstore.blockingStoreFiles	2100000000
hbase.hstore.compaction.max	20
hbase.hregion.majorcompaction	0
hbase.hregion.memstore.mslab.enabled	true

Hbase JVM优化



Hbase JVM 优化

-server 使用服务器模式
-Xmx20480m 最大分配20g内存
-Xms20480m 最小分配20g内存
-Xmn5000m 高吞吐server，增大年轻代内存

-XX:+UseParNewGC 年轻代采用并行收集策略，加快收集速度
-XX:MaxTenuringThreshold=20 20岁的对象才可以进入到旧生代，增加新生代minor collection的机率

-XX:+CMSParallelRemarkEnabled 表示并行remark
-XX:+UseConcMarkSweepGC 年老代采用CMS收集
-XX:+UseCMSCompactAtFullCollection 对CMS采用压缩策略
-XX:CMSFullGCsBeforeCompaction=15 15次full gc 做一次碎片整理
-XX:CMSInitiatingOccupancyFraction=70 堆区占用到70%的时候，开始进行CMS

-XX:+HeapDumpOnOutOfMemoryError
-XX:HeapDumpPath=/data2/hadoop/logs/hbase-log/oom.log
-XX:+PrintGCDetails 打印GC详细日志
-XX:+PrintGCTimeStamps 打印GC 时间
-XX:+PrintGCDateStamps 打印出gc的日期
-Xloggc:/data2/hadoop/logs/hbase-log/gc-hbase.log gc日志