美柚

HBase分布式列数据库

郭厦同
alien.xt.xm@gmail.com

美柚

目录

1 HBase简介

- Hadoop Distributed File System (HDFS™): A distributed file system that provides high-throughput access to application data.

- Hadoop YARN: A framework for job scheduling and cluster resource management.

- Hadoop MapReduce: A YARN-based system for parallel processing of large data sets.

面向行存储

- 数据按行存储
- 没有索引的查询使用大量I/O
- 建立索引需要耗费大量时间和资源
- 严格的ACID事务导致可用性和伸缩性下降

面向列存储

- 数据按列存储
- 数据即是索引
- 只访问查询的列（降低I/O）
- 一列一个线索处理（并发）
- 数据类型一致（压缩高效）

Apache HBase™ is the Hadoop database, a distributed, scalable, big data store.

- User表(id,name,age,gender)

- Bigtable: 3列,无所不包,key/value查询

HBase特性

- 支持海量数据
- 拥有良好扩展性
- 高性能读写
- 满足强一致性要求
- Schema灵活多变
- 面向列存储
- 数据多版本
- Java/Shell/Http Rest/Thrift

2 HBase数据模型

表(Table)

一个HBase的表由很多行构成

主键(RowKey)

- 表中行的唯一标识，作为检索的主键
- 长度不超过64kb的任意字符串，按字典排序

查询的三种方式

- 通过Rowkey查询
- 通过Roekey的范围查询
- 全表扫描

列族(ColumnFamily)

- 定义表结构的时候指定
- 包含一个或多个列
- 数据按列族存储
- 列族最好别指定太多
- 查询时跨列族查询情况少

列(ColumnQualifier)

- 表示为<cf>:<限定符|列>
- 可动态指定

时间戳(TimeStamp)

- 对应每次数据操作的时间，可自动生成，也可手动指定

版本回收

- 每个数据单元，保存指定个数最新版本
- 保存指定时间长度的版本（如7天）

单元格(Cell/Value)

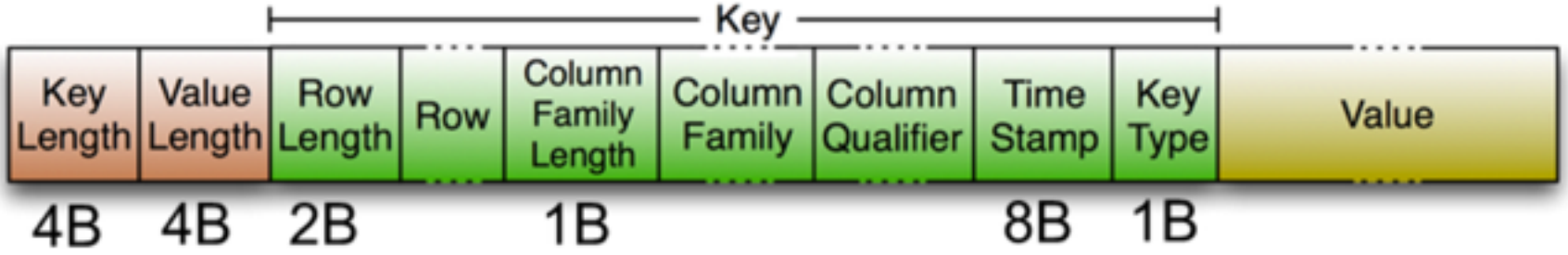
- 单元格由(rowkey + cf:c + timestamp)唯一决定
- 以字节数组形式存储，需要应用程序自己转换

逻辑结构

美柚

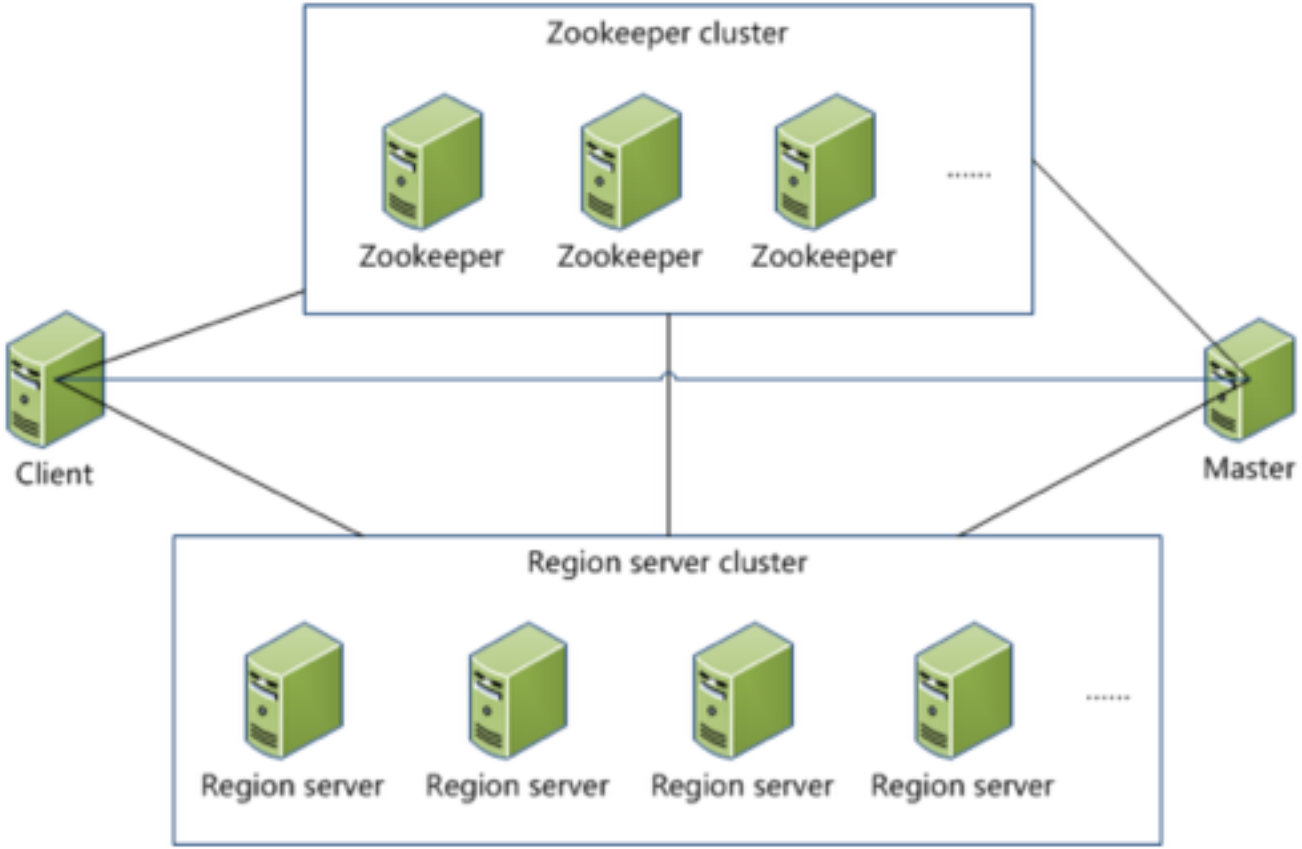
行键	时间戳	列族(contents)	列族(anchor)	列族(mine)
“com.cnn.www”	t9		anchor:cnnsi.com= “CNN”	
	t8		anchor:my.look.ca= “CNN.com”	
	t6	contents:html= “<html>…”		mine.type= “text/html”
	t5	contents:html= “<html>…”		
	t3	contents:html= “<html>…”		

Cell存储


Key
Key Length
Value Length
Row Length
Row
Column Family Length
Column Family
Column Qualifier
Time Stamp
Key Type
Value
4B
4B
2B
1B
8B
1B

3 HBase原理与架构

物理架构


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

HRegion & HRegionServer

- 表在行方向上，通过Rowkey的范围将表水平切割
- 每个Region纪录为Startkey和Endkey
- 自动切分Region
- 每个Region里的Rowkey都是排序的
- 存放和管理本地HRegion
- 读写HDFS，管理Table中的数据。
- Client直接通过HRegionServer读写数据

HMaster

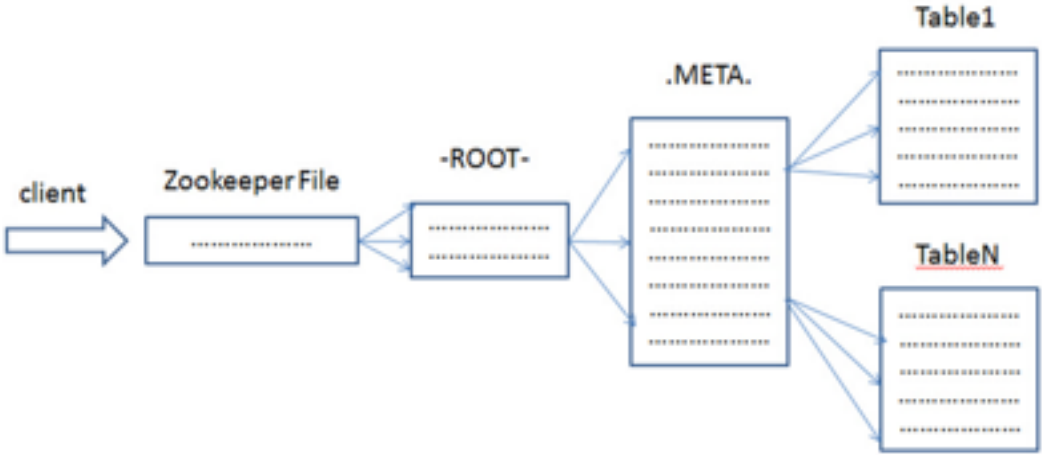
- 实现DDL操作,ns,table,cf修改
- 管理元数据，存储在HDFS上
- 权限控制（ACL）
- 管理HRegionServer，实现负载均衡
- 管理和分配HRegion，HRegion split和HRegionServer宕机时分配或迁移到别的HRegionServer

ZooKeeper(协调系统)

- 存放整个HBase集群的元数据以及集群的状态信息
- 实现HMaster主从节点的failover

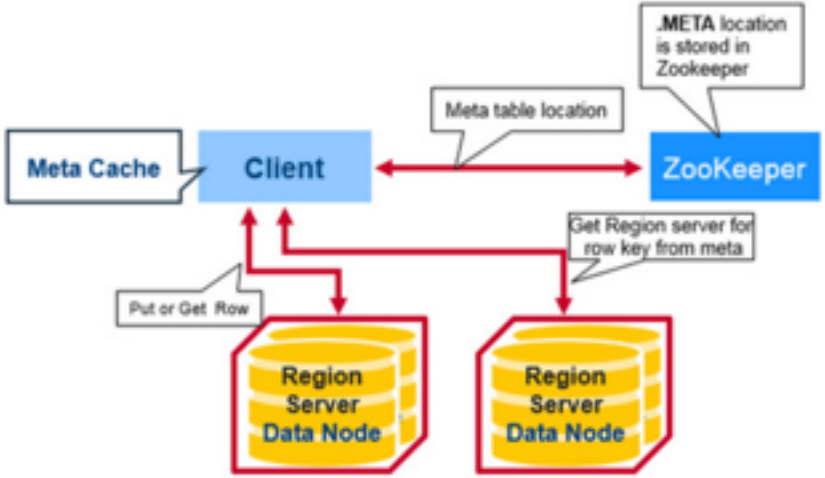
-ROOT- & .META.(0.96前)

- .META.纪录用户表的Region信息，可以有多个Region
- -ROOT-纪录了.META.的Region信息，只有一个Region，存放在Zookeeper

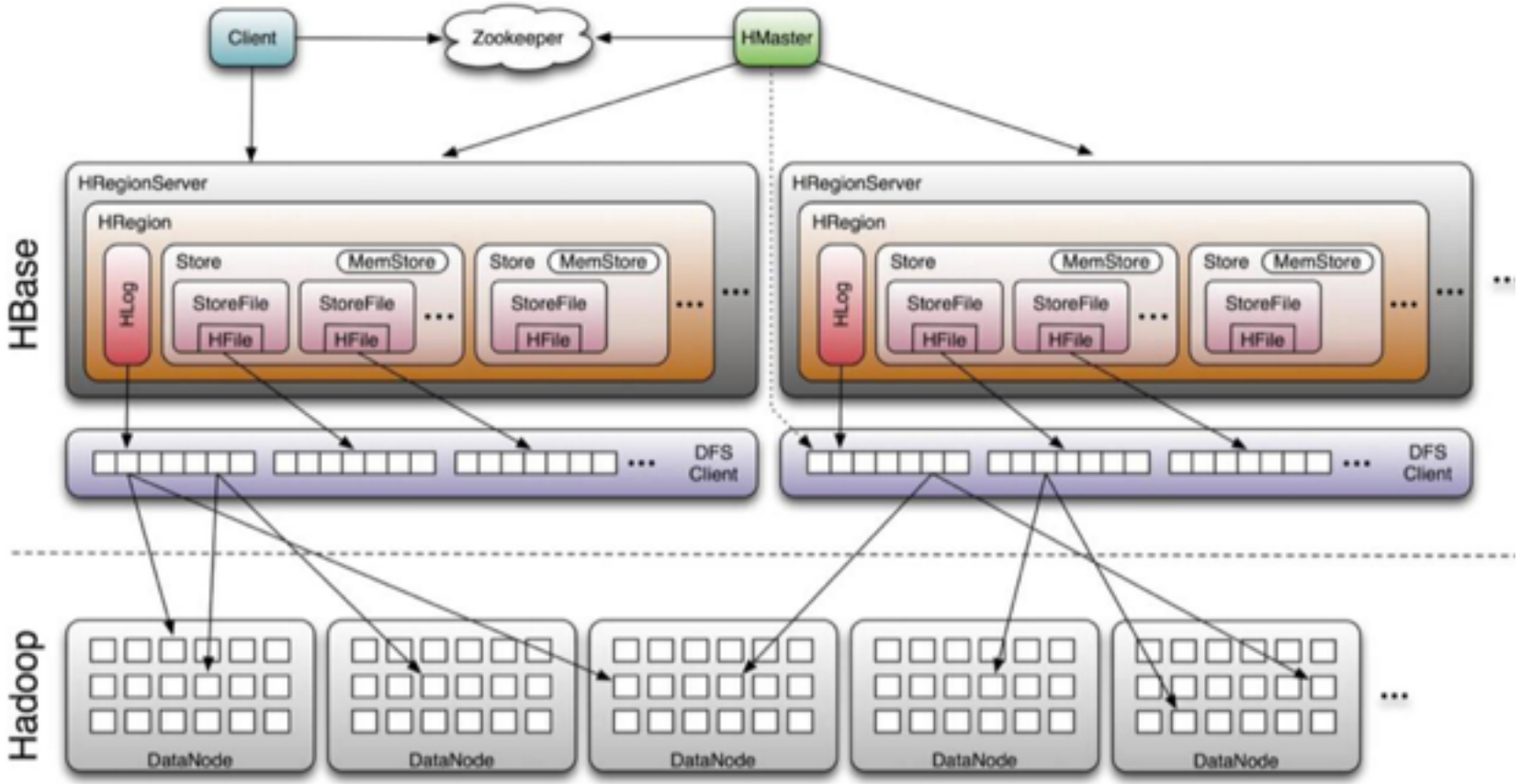


hbase:meta(0.96后)

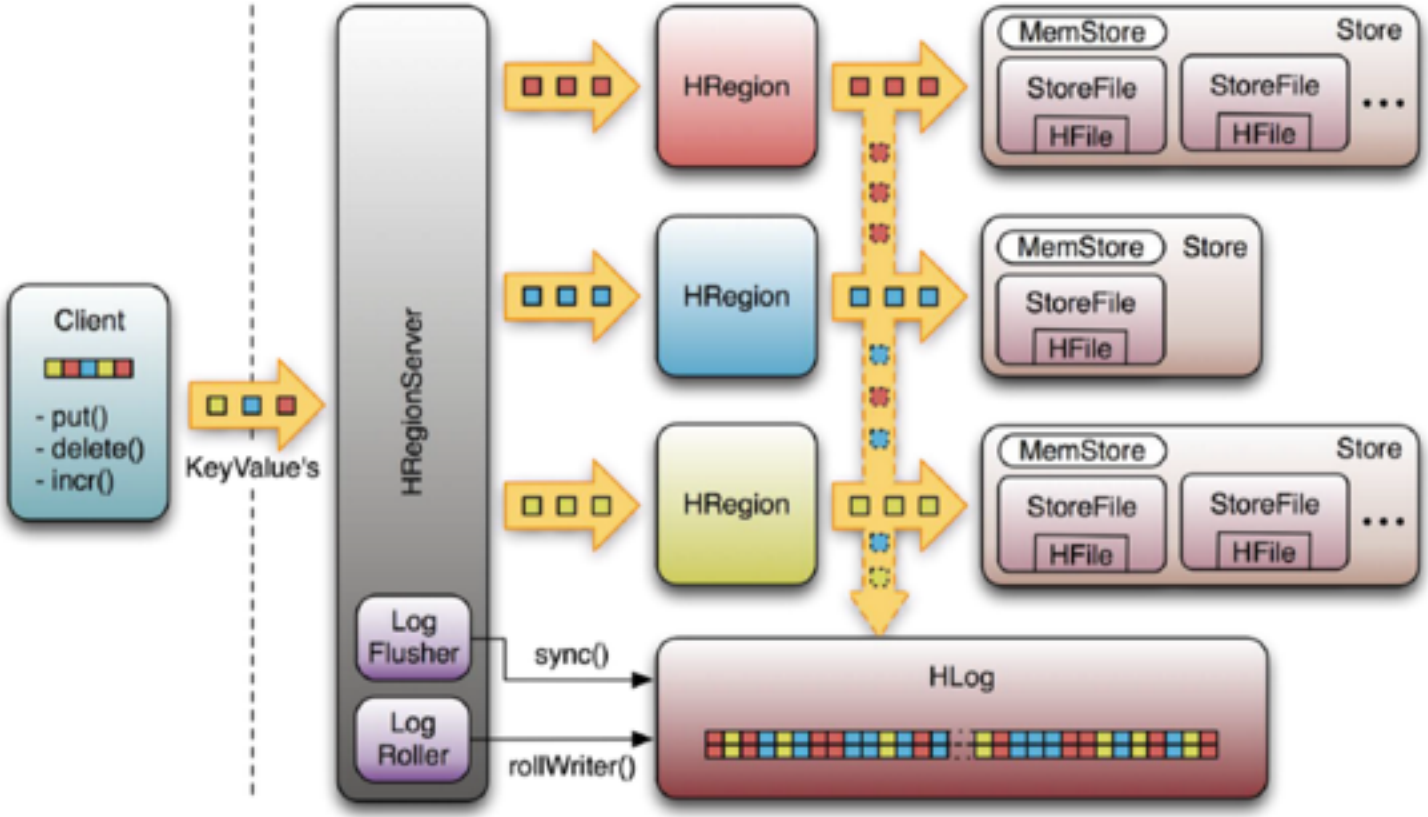
- hbase:meta纪录着用户表的Region信息，存储于Zookeeper(/hbase/meta-region-server)，缓存hbase:meta位置
- HRegionServer中查询用户Table对应请求的RowKey所在的HRegionServer，缓存该位置信息



物理架构


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

物理架构


Client
- put()
- delete()
- incr()
KeyValue's
HRegionServer
Log Flusher
Log Roller
sync()
rollWriter()
HRegion
HRegion
HRegion
MemStore
Store
StoreFile
HFile
StoreFile
HFile
...
MemStore
Store
StoreFile
HFile
MemStore
Store
StoreFile
HFile
StoreFile
HFile
...
HLog

4 HBase实践

5 HBase模型及表设计

什么时候用HBase?

- 查询模式确立不轻易改变
- 高速插入，大量读取
- 海量的数据
- 简单操作(如key/value)

用户浏览行为记录

- 时间版本查询，可以快速获取最近n条
- 近期数据放在Memstore，减少I/O
- 分布式，负载均衡

列族的设计

- 列族最好不要超过3个，每个列族都保存在对应的HFile里，当flush和compaction时，会产生大量I/O
- 将经常查询和不经常查询的数据放到不同的列簇
- 列族和列名不要太长，因为每个Cell都保存着

RowKey的设计

- 越短越好，最好别超过16个字节
  - 每个Cell都存储着Rowkey，过长会影响存储效率，以及占用空间
  - MemStore缓存部分数据，如果过长，将导致内存利用率低，检索效率低

- 散列原则
  - 高位作为散列字段，时间字段不要放高位

- 唯一性原则

6 HBase优化

查询优化

- Bloomfilter
- 二级索引
- 表压缩
- 预先分区
- Hive on HBase

Thank you

alien.xt.xm@gmail.com

厦门美柚信息科技有限公司

http://github.com/alienxt