参考：<https://portal.huaweicloud.com/blogs/349b7bb5a7d611e69023286ed488c65c>

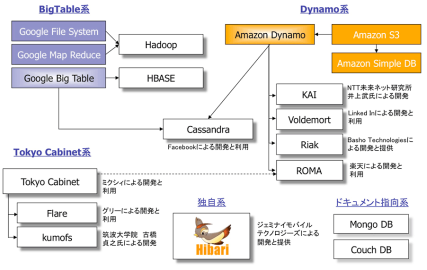
**Hbase（一） 简介**

【摘要】 Hbase简介...

**Hbase简介**

**NoSQL 数据库族**

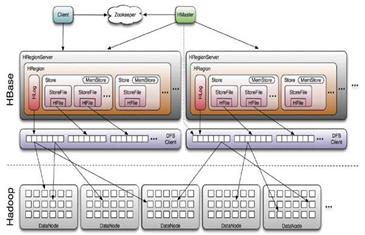
ü Key-Value 存储: Amazon Dynamo, Voldemort, BDB, Kyoto Cabinet ，memcache  
ü Key 结构化数据存储：Redis   
ü 类BigTable 存储: Google BigTable, Apache HBase, Apache Cassandra   
ü 文档数据库: MongoDB, CouchDB   
ü 全文索引: Apache Lucene, Apache Solr   
ü 图数据库: neo4j, HyperGraphDB, FlockDB



**BigTable**

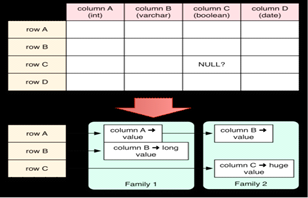
Hbase概念源自Google BigTable http://static.googleusercontent.com/media/research.google.com/zh-CN//archive/bigtable-osdi06.pdf

**架构概览**



**列式数据库**

**逻辑视图**



Hbase以列族作为逻辑单位，物理上，也是以列族为单位存在HFile中。

**空值：**

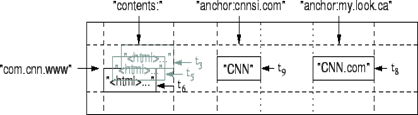
不会被存储，所以不会有消耗。

**常用存取模式为：**

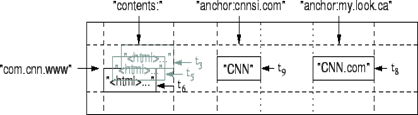
(Table, RowKey, Family, Column, Timestamp) → Value

Me:由此也可以称HBase为kv数据库，key由表、行键、列族、列名、版本来确定  
用java来描述就类似为：   
SortedMap<sortedmap>>>

**加上时间维度之后，逻辑视图也可以是：**



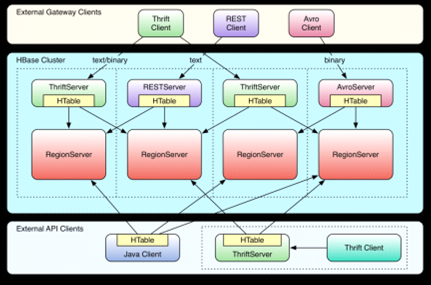
**物理视图**



Hbase负载均衡的单元是Region，就是以row key 排序的连续存储空间。类似Oracle的范围分区。Region不断增长，到阈值后会自动分区，一分为二。   
物理上，Region 由 Region Server加载，每个Server加载多个Region

**客户端**

RPC协议支持：Protocol Buffer、Thrift、Avro   
REST客户端支持，适合请求少但数据量大的场景。高吞吐量的场景中，适合二进制协议



REST客户端：http://wiki.apache.org/hadoop/Hbase/Stargate   
Thrift客户端：http://wiki.apache.org/hadoop/Hbase/ThriftApi   
Avro客户端：https://hbase.apache.org/0.94/apidocs/org/apache/hadoop/hbase/avro/package-summary.html   
HBql：Hbase基础上的SQL语法 http://www.hbql.com/   
Hbase-DSL：格式化查询辅助 https://github.com/altamiracorp/hbase-dsl   
JPA/JPO：使用DataNucleus在Hbase封装 http://www.datanucleus.org/   
PyHbase：与Avro网关服务通信 https://github.com/hammer/pyhbase   
AsyncHBase：异步、非阻塞、线程安全 https://github.com/StumbleUponArchive/asynchbase   
Hbase Shell：最常用的API，使用JRuby格式   
Java API：Java原生API   
MapReduce：MapReduce Java API 和 Clojure的 Hbase-Runner项目   
Hive：HiveQL支持 https://cwiki.apache.org/confluence/display/Hive/HBaseIntegration   
Pig：Pig Latin支持   
Cascading：MapReduce的替代API   
Web UI：Hbase的 Web UI可以查看状态，也可以修改少量的状态

**事务支持**

Hbase支持单行事务，对单个行键下的存储数据支持原子的“读-改-写”序列   
Hbase单元格也可以当做计数器使用，并支持原子更新

**WAL与持久化**

Hbase 每次更新数据时，先将数据写入WAL(write-ahead log)中，然后写入内存中的memstore。   
Memstore中的数据已经按照行健排序，超过最大值Flush到HFile中也是按照这个顺序存贮的。WAL是Hbase保证一致性和故障恢复的手段。   
Hbase存储架构来源自LSM树。HFile通常保存在HDFS中，写入就不可修改。这个机制会刷出很多小文件，所以通过minor compaction合并同列族的多个文件为一个大文件。这是一个多路归并的过程，因为已经是排序好的，所以非常快，IO消耗也较低。Major compaction会顺序重写数据，过程中与minor不同的是，会扫描出删除的“墓碑标记”，把标记删除的数据真正删除。

**事务支持**

**CPU**

Master dual quad-core CPUs, 2.0-2.5 GHz   
Slave dual quad-core CPUs, 2.0-2.5 GHz

**RAM**

NameNode计算方式：每100TB数据或每100万个文件大约占用堆1GB左右   
SecondaryNameNode：同NameNode   
DataNode：一般1G够用   
Hbase Master：轻量级负载，一般4G够用   
Hbase Region Server：内存负载重，一般设为12G，最大不超过16G，不然FGC时间会长，可能会使Master误判RS假死   
ZooKeeper：一般1G

**Disk**

Master一般采用RAID10   
Salve一般采用JBOD（Just a Bunch Of Disks，磁盘簇），有米上RAID更好   
SATA磁盘就可以满足，SAS更好

**Network**

按数据规模估算，一般的选择是千兆网卡，并保证交换机有足够的吞吐能力。   
一般数据量\*2配置网络，因为网络通常达不到理论标准

**安装**

ü 下载HBASE：http://www.apache.org/dyn/closer.cgi/hbase/   
ü 安装JAVA，并设置JAVA\_HOME   
ü 安装Hadoop，选择HDFS作为底层文件系统，全部jar要配置到Hbase的java.library.path   
ü 配置SSH和域名等辅助配置（可选）   
ü 调低/etc/sysctl.conf 的vm.swappiness配置，降低SWAP使用优先级(可选)   
ü NTP同步时间，服务器间不能有太大的时间偏差   
ü 文件句柄设置，评估每个列族至少1个文件，一般按5个评估。也可以压力测试后，查看一个region server的句柄数lsof -p | wc –l; 调整ulimit /etc/security/limits.conf 的nofile、/etc/sysctl.conf的fs.file-max   
ü 编辑hadoop的hdfs-site.xml，修改dfs.datanode.max.xcievers为4096，设置DataNode可处理的文件上限数   
ü 编辑conf/hbase-site.xml，至少配置hbase.rootdir（HDFS根地址）和hbase.zookeeper.property.dataDir、hbase.cluster.distributed、hbase.zookeeper.quorum   
ü 编辑conf/hbase-env.sh，配置java堆等选项   
ü 编辑conf/ regionservers 每行一个RS主机名（基于域名配置）   
ü 编辑 conf/log4j.properties配置日志级别   
ü 启动bin/start-hbase.sh   
ü 交互bin/hbase shell，进入shell后输入status命令可以查看server运行状态，或使用Web UI，默认端口master 60010、RS 60030   
ü 关闭bin/stop-hbase.sh

**支持的文件系统**

一般默认的文件系统为HDFS，但是不仅仅支持HDFS。   
通过设置URI，通过URI标识符中的scheme，即第一个冒号前的部分，标识了使用的磁盘。   
本地文件：file:///   
HDFS：hdfs://:/   
S3：s3://（基于块） s3n://（原生模式，不超过5G）

**单机和分布式运行模式**

单机模式：是默认模式，使用本地文件系统，ZK和HBASE运行在一个JVM中   
伪分布式：一个主机，多个进程的模拟分布式，基于HDFS文件系统   
分布式：conf/hbase-site.xml的属性hbase.cluster.distributed为true，基于HDFS，可选内部或外部的ZK，配置RS；保持各个server的配置同步，可以人工、脚本方式，基于云的部署可以考虑Apache Whirr、Puppet或Chef

作者 | 林钰鑫

转载请注明出处：华为云博客 https://portal.hwclouds.com/blogs