HBase系统结构怎么理解?

厉梦如 2019-05-19 发表

问题描述

Q：HBase系统结构怎么理解?

解决方法

A：

HBase系统架构:由Client、ZooKeeper、HMaster、HRegionServer、HRegion、HStore、HLog、HDFS等部件组成。

l Client

n 使用HBase RPC机制与HMaster和HRegionServer进行通信;

n Client与HMaster进行通信进行管理类操作;

n Client与HRegionServer进行数据读写类操作。

l ZooKeeper

n Zookeeper Quorum存储.meta.表地址、HMaster地址;

n HRegionServer把自己以Ephedral方式注册到Zookeeper中，HMaster随时感知各个HRegionServer的健康状况;

n Zookeeper避免HMaster单点问题。

l HMaster

n HMaster没有单点问题，HBase中可以启动多个HMaster，通过Zookeeper的Master Election机制保证总有一个Master在运行

n 主要负责Table和Region的管理工作:

n 管理用户对表的增删改查操作;

n 管理HRegionServer的负载均衡，调整Region分布;

n Region Split后，负责新Region的分布;

n 在HRegionServer停机后，负责失效HRegionServer上Region迁移。

l HRegionServer

n HBase中最核心的模块，主要负责响应用户I/O请求，向HDFS文件系统中读写数据。

n HRegionServer管理一些列HRegion对象;

n 每个HRegion对应Table中一个Region，HRegion由多个HStore组成;

n 每个HStore对应Table中一个Column Family的存储;

n Column Family就是一个集中的存储单元，故将具有相同IO特性的Column放在一个Column Family会更高效。

l Region

n 当Table随着记录数不断增加而变大后，会逐渐分裂成多份splits，成为regions，一个region由[startkey,endkey)表示，不同的region会被Master分配给相应的RegionServer进行管理。

l HStroe

n HBase存储的核心。由MemStore和StoreFile组成，MemStore是Sorted Memory Buffer。

l HLog

n 在分布式系统环境中，无法避免系统出错或者宕机，一旦HRegionServer意外退出，MemStore中的内存数据就会丢失，为防止数据丢失，引入了HLog。

n 每次用户操作写入Memstore的同时，也会写一份数据到HLog文件，HLog文件定期会滚动出新，并删除旧的文件(已持久化到StoreFile中的数据)。当HRegionServer意外终止后，HMaster会通过Zookeeper感知，HMaster首先处理遗留的HLog文件，将不同region的log数据拆分，分别放到相应region目录下，然后再将失效的region重新分配，领取到这些region的HRegionServer在Load Region的过程中，会发现有历史HLog需要处理，因此会Replay HLog中的数据到MemStore中，然后flush到StoreFiles，完成数据恢复。

l HDFS

n HDFS是一个分布式文件系统。它通过将一个大的文件划分成一个个固定大小的Block来实现分布式存储。每一个Block的默认大小为128MB。每一个Block都存在多个备份，并且被部署在不同的数据节点上，来保障数据的安全。目前，HBase的所有底层数据都以文件的形式交由HDFS来存储。HBase一侧本身不固化保存数据信息。

n RegionServer和DataNode一般会放在相同的Server上实现数据的本地化。