《BigTable：结构化数据的分布式存储系统》阅读整理

Bigtable论文主要描述了BigTable的数据模型、设计思想和具体实现。

Bigtable兼具了广泛的可使用性、可拓展性、高性能和高可用性。Bigtable在实现策略上与传统数据库有很多相似的地方，但是Bigtable不支持完整的关系型数据模型；相反，Bigtable提供的是一种非常简单的数据模型，以及为客户端提供接口，可以灵活地操控数据层和格式，可以控制数据在物理层的存储方式。对于Bigtable来说，存储的数据都是未解释的字符串，bigtable对数据的行键和列名建立索引。最后，通过调整Bigtable的表结构参数，客户端还可以控制数据是存在内存还是硬盘上。

BigTable和数据库比较类似，它的实现参照了很多数据库的实现策略（并行数据库，内存数据库），但它提供了一套完全不同的接口；BigTable不支持关系型模型，即不支持schema，它提供的模型允许用户动态控制数据的分布和格式。

Bigtable是一个稀疏的、分布式的、持久化的多维有序Map。这张Map针对行键、列名和时间戳建立了索引；Map中的每个值都是一串未解释的字节数组。BigTable视一切数据为字符串，用户可以把结构化或半结构化的数据串行化到字符串里，进行存储。

Bigtable的数据模型，行键：任意小于64KB的字符串。Bigtable内部的行键会按照字典序排列，对于任何单个的行键操作都是原子性的。Tablet：片，是Bigtable分配和负载均衡的单元单位，动态管理。列族：一定数量的列的集合，是访问控制的单元单位。一般同一列族内的列都是同种类型，一个table里的列族数量一般期望是尽量少，最多不超过几百个，但一个table里可以有很多的列。列键： 一般从属于某个列族。时间戳：一个64位整数，每一行不同的时间戳对应不同的行版本。时间戳一般由客户端自动生成，也可以由用户编程中指定数字。版本：Bigtable引入了版本管理（semi），由于每一个时间点行操作是原子的，所以默认由时间戳代表该行的不同的版本。但我称Bigtable的版本管理是semi-version-control，因为实际上，由于性能和容量的原因决定了它不像svn或git那样会保存之前所有修改的状态。一般可以由用户选择要保留最近的n个版本。其余的会被Bigtable的GC机制回收掉。

Bigtable的实现主要有三个模块组成：一个链接给所有客户端的库，一个master server，和若干个tablet server。master节点的作用是：分配tablet到各个tablet server，发现新增或者被移除的tablet server，均衡tablet server的负载，回收GFS上的文件，管理表结构更改（包括表或者列族的创建）

tablet server的职责是，管理一个tablet集合。tablet server直接处理读写请求，当tablet数据过多，需要发起tablet分裂操作。tablet server可以动态地添加或者移除，以适应集群负载。

与许多单master分布式存储系统相似，客户端数据不经过master，而是直接向tablet server发起请求。由于连tablet的位置信息也不需要通过master节点，master节点的负载是很少的。

Bigtable使用一个三层结构的B+树来存储tablet位置信息。第一层是一个Chubby文件，存储了root tablet的位置，而root tablet存储了METADATA表所有分片的位置，METADATA表中则存储了用户表所有分片的位置。

客户端中缓存了tablet的位置信息，当客户端发现tablet位置信息错误，它会递归地查找B+树，以取得新的位置信息。当客户端首次建立链接时，至多需要三个网络往返时间取得tablet位置信息；但如果是发现METADATA表信息错误，至多需要六次网络往返。为了节省时间，METADATA表存储在内存中；为了进一步节省时间，客户端每次回读取超过一个数据分片，当它读取METADATA表的信息时，这称为预读取。

BigTable使用其分布式文件系统GFS实现数据存储，使用其运算架构MapReduce模型来完成海量数据的处理，使用Chubby进行协同服务管理。

BigTable可以处理海量数据，分布式并发处理的效率很高，便于数据结构扩展（系统可以动态扩展），适合由廉价设备组成的系统（有较高的硬件容错能力），但不适合频繁的读/写操作。

Google的Bigtable等相关技术极大地普及了云计算中非常核心的分布式技术，例如Hadoop这一开源产品便克隆了该技术。

软件1801 王汉功

18301020