**Bigtable读后感**

BigTable是谷歌在2006年的一篇论文中提出来的，为后来HBase的出现提供了理论基础。Bigtable是一个稀疏的、分布式的、持久化存储的多维排序Map。Map的索引是行关键字、列关键字以及时间戳；Map中的每个value都是一个未经解析的byte数组。

Bigtable中的行关键字可以是任意的字符串，并且每行的读写操作都是原子的；Bigtable中的行关键字是按照字典顺序排序存储的，表中的行都可以进行动态分区，每个分区叫tablet，tablet是数据分布和负载均衡的最小单位。由于行键是按照字典序存储的，所以查询时以行关键字作为条件查询速度毫秒级。

列关键字组成的集合叫做“列族”，列族时访问控制的基本单位。列关键字的命名语法为：列族：限定词。每行数据都会有用来当作版本号的时间戳，可以系统自动赋值，也可以用户自己指定。最新的数据行排在最前面。还可以利用时间戳来进行垃圾收集。

Bigtable使用Google的分布式文件系统GFS存储日志文件和数据文件。BigTable内部存储数据的文件是Google SSTable格式的。SSTable是一个持久化的、排序的、不可更改的Map<key,value>结构，其值都是任意的byte串，因此使用key查询速度很快。Big Table还依赖一个高可用的、序列化的分布式锁服务组件——Chubby。BigTable使用Chubby完成以下几个任务：

1. 确保在任何时间内最多只有一个活动的Master副本；
2. 存储BigTable数据的自引导指令的位置
3. 查找Tablet服务器，以及在Tablet服务器失效时进行善后；
4. 存储BigTable模式信息
5. 存储访问控制列表。BigTable包括了三个主要的组件：链接到客户程序的库、一个Mater服务器和多个Tablet。针对系统工作负载的变化情况，BigTable可以动态的向集群添加或者删除Tablet服务器。

Master服务器主要为Tablet服务器分配Tablets、检测新加入的或者过期失效的Tablet服务器、对Tablet服务器进行负载均衡、以及对保存在GFS上的文件进行垃圾收集。除此之外，还处理模式的相关修改操作，例如建立表和列族。每个Tablet服务器都管理一个Tablet的集合，每个Tablet的服务器负责处理它所加载的Tablet的读写操作，以及在Tablets过大时，对其进行分割。客户端读取的数据都不经过Master服务器；客户程序直接和Tablet服务器通信进行读写操作。在任何一个时刻，一个Tablet只能分配给一个Tablet服务器。Master服务器记录了当前有那些活跃的Tablet服务器、那些Tablet分配给了那些Tablet服务器、那些Tablet还没有被分配。

BigTable使用Chubby跟踪记录Tablet服务器的状态。当一个Tablet服务器启动时，它在Chubby的一个指定目录下建立一个有唯一性名字的文件，并且获取该文件的独占锁。Master服务器实时监控着这目录，因此Master服务能够知道有新的Tablet服务器加入了。只要文件存在Tablet服务器就会试图重新获得对该文件的独占锁，如果文件不存在了，那么Tablet服务器就不能在提供服务了。

1. Master 服务器从Chubby 获取一个唯一的Master 锁，用来阻止创建其它的Master 服务器实例；
2. Master 服务器扫描Chubby 的服务器文件锁存储目录，获取当前正在运行的服务器列表；
3. Master 服务器和所有的正在运行的Tablet 表服务器通信，获取每个Tablet 服务器上Tablet 的分配信息；
4. Master 服务器扫描METADATA 表获取所有的Tablet 的集合。在扫描的过程中，当Master 服务器发现了一个还没有分配的Tablet，Master 服务器就将这个Tablet 加入未分配的Tablet 集合等待合适的时机分配。

Tablet的持久化状态信息保存在GFS上。更新操作提交到REDO日志中。这些更新操作中，最近提交的那些放在一个排序的缓存中，我们称这个缓存为memtable；较早更新存放在一系列的SSTable中。随着写操作的执行，memtable的大小不断增加。当memtable的尺寸到达一个门限值的时候，这个memtable就会被冻结，然后创建一个新的memtable；被冻结住的memtable会被转换成SSTable，然后写入GFS。客户程序可以将多个列族组合成一个局部性群族。对Tablet中的每个局部性群族都生成一个单独的SSTable。将同城不会一起访问的列族分割成不同的局部性群族可以提高读取操作的效率客户程序可以控制一个局部性群族的SSTable是否需要压缩，一般使用两遍的、可定制的压缩

为了提高读操作的性能，Tablet服务器使用二级缓存的策略，一级用来缓存Tablet服务器通过SSTable接口的Key-Value对；Block是二级缓存，用来缓存从GFS读取的SSTable的Block。

整个BigTable设计符合大部分大数据程序的需求，打破了关系型数据库的结构化存储，能够部署在成千上万台服务器上，可以存储PB级数据，对整个互联网行业的快速发展提供了坚实的理论基础与成功案例。