BigTable 读后感

1. 概念

简单来讲，BigTable是一个分布式的结构化数据存储系统，可以可靠地处理PB级别的数据，并且能够部署到上千台机器上。在Google公司得到了广泛应用。

Bigtable不支持完整的关系数据模型，而是让用户自己去定义表，数据都是没有格式的。数据的下标是行和列的名字，名字可以是任意的字符串。Bigtable将存储的数据都视为字符串，但是 Bigtable 本身不去解析这些字符串，客户程序通常会在把各种结构化或者半结构化的数据串行化到这些字符串里。

1. 数据模型

Bigtable是一个稀疏的、分布式的、持久化存储的多维度排序Map，Map的索引是行关键字(row key)、列关键字(column families)以及时间戳(timestamps)；Map中的每个 value 都是一个未经解析的 byte 数组。

表中的行关键字可以是任意的字符串，对同一行关键字的读或写都是原子的。

列关键字组成的集合叫做“列族“，列族是访问控制的基本单位。存放在同一列族下的所有数据通常都属于同一个类型。列关键字的命名如下： 列族：限定词。访问控制、磁盘和内存的使用统计都是在列族层面进行的。

表的每一个数据项都可以包含同一份数据的不同版本；不同版本的数据通过时间戳来索引，Bigtable 可以给时间戳赋值，用来表示精确到毫秒的“实时”时间；用户程序也可以给时间戳赋值。

1. 接口函数

Bigtable 提供了建立和删除表以及列族的 API 函数。Bigtable 还提供了修改集群、表和列族的元数据的API，比如修改访问权限。

1. BigTable的组成

Bigtable 包括了三个主要的组件:链接到客户程序中的库、一个 Master 服务器和多个 Tablet 服务器。针对系统工作负载的变化情况，BigTable 可以动态的向集群中添加(或者删除)Tablet服务器。

一个 BigTable 集群存储了很多表，每个表包含了一个 Tablet 的集合，而每个 Tablet 包含了某个范围内的行的所有相关数据。初始状态下，一个表只有一个 Tablet。随着表中数据的增长，它被自动分割成多个 Tablet 。使用一个三层的、类似Ｂ+树[10]的结构存储 Tablet 的位置信息。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 服务器通过轮询 Tablet 服务器文件锁的状态来检测何时 Tablet 服务器不再为 Tablet 提供服务。Tablet 的持久化状态信息保存在 GFS 上。

当对 Tablet 服务器进行写操作时，Tablet 服务器首先要检查这个操作格式是否正确、操作发起者是否有执行这个操作的权限。权限验证的方法是 通过从一个 Chubby 文件里读取出来的具有写权限的操作者列表来进行验证，成功的修改操作会记录在提交日志里。

1. 缩减

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 memtable 会被转换成 SSTable，然后写入 GFS。每一次 Minor Compaction 都会创建一个新的 SSTable，合并所有的 SSTable 并生成一个新的 SSTable 的 Merging Compaction 过程叫作 Major Compaction。

1. 优化
   1. 局部性群组

客户程序可以将多个列族组合成一个局部性群族，此外，可以以局部性群组为单位设定一些有用的调试参数，提高 METADATA 表中具有位置相关性的列族的访问速度。

* 1. Commit 日志的实现

为了避免多次读取日志文件，我们首先把日志按照关键字(table，row name，log sequence number)排序。排序之后，对同一个 Tablet 的修改操作的日志记录就连续存放在了一起，因此，我们只要一次磁盘 Seek 操作、之后顺序读取就可以了。为了并行排序，我们先将日志分割成 64MB 的段，之后在不同的 Tablet 服务器对 段进行并行排序。这个排序工作由 Master 服务器来协同处理，并且在一个 Tablet 服务器表明自己需要从 Commit 日志文件恢复 Tablet 时开始执行。

为了确保在 GFS 负载高峰时修改操作还能顺利进行，每个 Tablet 服务器实际上有两个日志写入线程，每个线程都写自己的日志文件，并且在任何时刻，只有一个线程是工作的。

* 1. 利用不变的SSTable对系统进行简化

我们对内存表采用 COW(Copy-on-write) 机制，这样就允许读写操作并行执行。SSTable 的不变性使得分割 Tablet 的操作非常快捷。

1. 性能评估
   1. 单个Tablet服务器的性能

* 随机读的性能比其他操作慢一个或多个数量级。
* 内存中的随机读操作速度快很多。
* 随机和序列写操作的性能比随机读要好些。
* 序列读的性能好于随机读
* 扫描的性能更高
  1. 性能提升

随着Tablet服务器的增多，系统的整体吞吐量有了很大增长，但是，这个提升并不是线性的，是由于多台服务器间的负载不均衡造成的，结果导致随着服务器数量的增加，每台服务器上的吞吐量急剧下降。

1. 在Google上的应用
   1. 谷歌分析

谷歌分析使用两个表，一个是Row Click，每一行存放了一个最终用户的会话。行的名字是一个包含 Web 站点名字以及用户会话创建时间的元组。这种模式保证了对同一个 Web 站点的访问会话是顺序的，会话按时间顺序存储。Summary 表(大约有 20TB 的数据)包含了关于每个 Web 站点的、各种类型的预定义汇总信息。

* 1. 谷歌地球

这个系统使用一个表存储预处理数据，使用另外一组表存储用户数据。Imagery 表的每一行都代表了一个单独的地理区域。行都有名称，以确保毗邻的区域存储在了一起。Imagery 表中有一个列族用来记录每个区域的数据源，由于每个地理区域都是由很少的几张图片构成的，因此这个列族是非常稀疏的。这个服务系统使用一个表来索引 GFS 中的数据。

* 1. 个性化查询

个性化查询使用 Bigtable 存储每个用户的数据。每个用户都有一个唯一的用户 id，每个用户 id 和一个列名绑定。一个单独的列族被用来存储各种类型的行为 (比如，有个列族可能是用来存储所有的 Web 查询的)。每个数据项都被用作 Bigtable 的时间戳，记录了相应的用户行为发生的时间。个性化查询使用以 Bigtable 为存储的 MapReduce 任务生成用户的数据图表。这些用户数据图表用来个性化当前的查询结果。