**Big Table——分布式结构化数据存储系统**

**一、基本特点**

数据没有Schema,用户自己定义 Schema

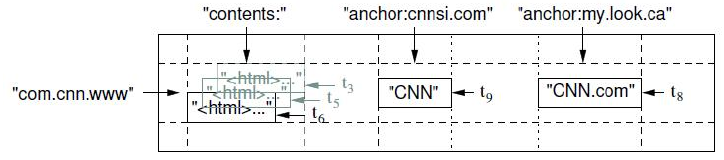
**二、数据模型**

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= 行关键字+列关键字+时间戳

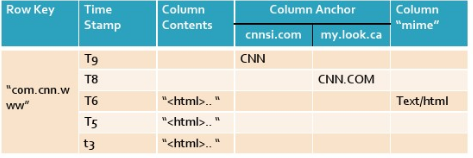
（row:string, column:string, time:int64）->string

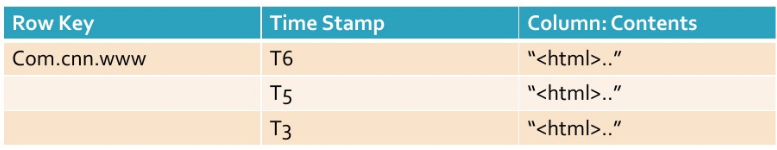
**三、数据模型举例**

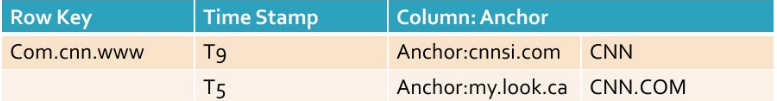


用URL（反向存储）作为行关键字

**四、物理视图**









**五、逻辑模型**

**数据模型——行**

每一行可以参与动态分区，一个分区叫做一个”Tablet”，Tablet是数据分布和负载均衡调整的最小单位。（SSTable是列存储的单位）

逻辑上的“表”被划分为若干子表（Tablet）

每个Tablet由多个SSTable文件组成

SSTable文件存储在GFS之上

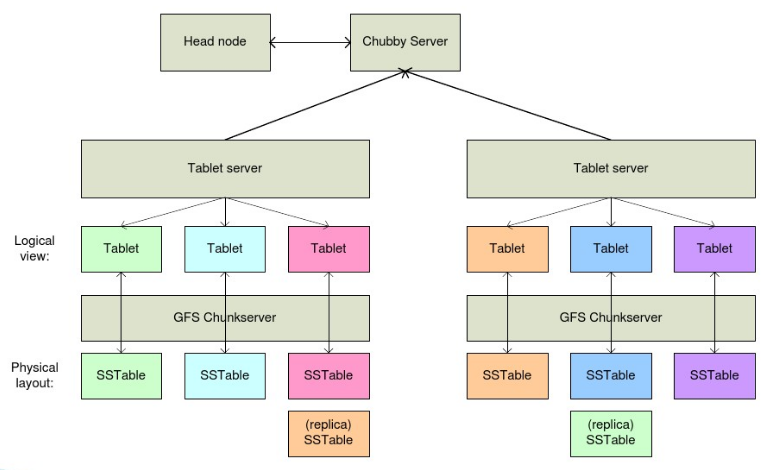
每个Tablet存储了table的一部分行

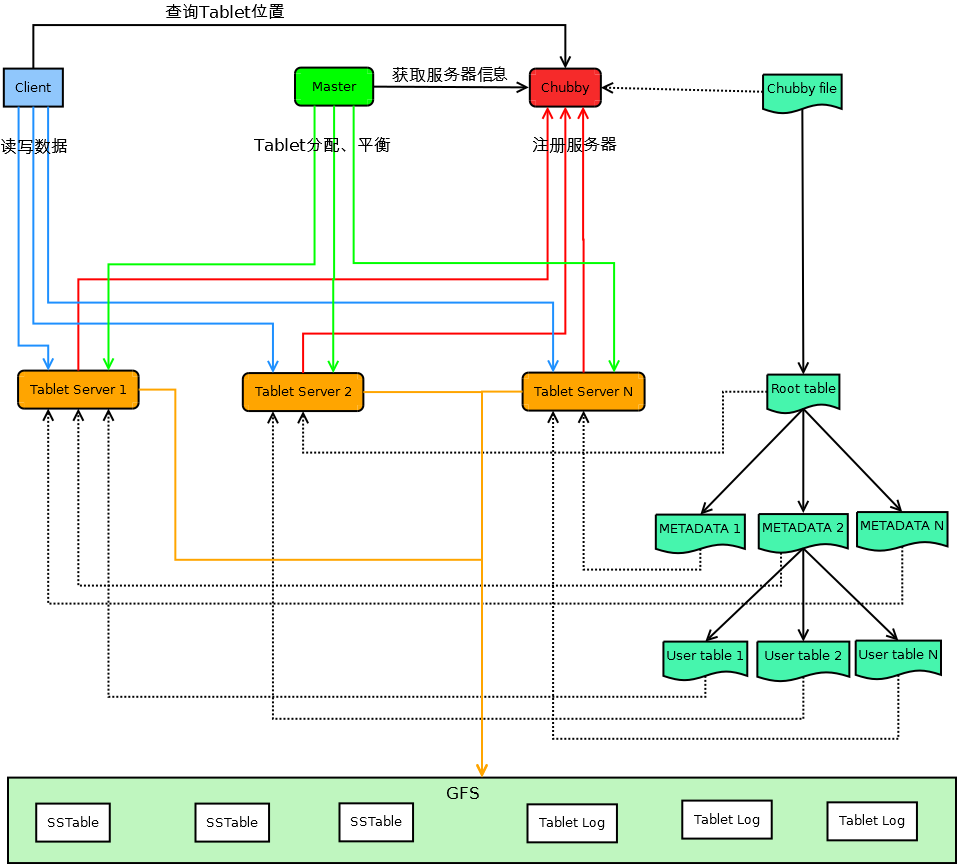
元数据：起始行键盘、终止行键

如果子表体积超过阀值，则进行分割

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。

**六、体系结构**





**七、事务处理**

**BigTable支持单行上的事务处理，用户可以对存储在一个行关键字下的数据进行原子性的读-更新-写操作，BigTable并不支持通用的跨行事务处理。**

**八、BigTable构件**

Bigtable建立在其它几个Google基础构件上：

1）使用Google的分布式文件系统(GFS) 存储日志文件和数据文件；

2）依赖高可用的、序列化的分布式锁服务组件Chubby。

**数据文件**

BigTable内部存储数据的文件：Google SSTable。

SSTable是一个持久化的、排序的、不可更爱的Map结构。从内部看SSTable是 一系列的数据块，通常每个块的大小是64KB。

数据块的定位：使用块索引（通常存储在SSTable末尾）

**Chubby**

分布式锁服务组件，一个Chubby服务包括了5个活动的副本，其中一个副本被选为Master，并处理请求。

Chubby服务可用的前提：

1）大多数副本都正常运行

2）彼此之间能够互相通信

Bigtable使用Chubby完成以下的几个任务：

1）确保在任何给定的时间内最多只有一个活动的Master副本；

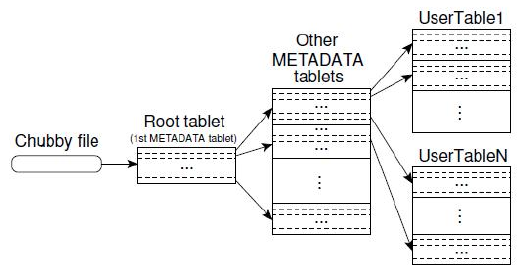
2）查找Tablet服务器，以及在Tablet服务器失效时进行善后；

3）存储BigTable数据的自引导指令的位置；

4）存储BigTable的模式信息（每张表的列族信息）；

5）存储访问控制列表。

**Tablet存放位置**

****

一个存储在Chubby中的文件，包含Root Tablet的位置信息（也有说这是第一层）

第一层：Root Tablet,包含了一个特殊的METADATA表，记录METADATA里所有的TABLET的位置信息

第二层：METADATA表，其中的每个TABLET包含了一个用户TABLET的集合。

第三层：用户表Table

Root Tablet:

1实际上是METADATA的第一个Tablet

2永远不会被分割

METADATA表还存储次级信息(secondary information)，主要包括每个Tablet的事件日志（例如一个服务器何时开始为该Tablet提供服务）。

↓

有助于排查错误和性能分析

**Tablet的客户端缓存**