分布式数据库数据一致性的原理、与技术实现方案

题目标签

学习时长：20分钟

题目难度：中等

知识点标签：分布式

题目描述

分布式数据库数据一致性的原理、与技术实现方案

1. 面试题分析

1. 根据题目要求我们可以知道：

    1、分布式系统

    2、CAP理论

3、Raft算法

2. 容易被忽略的坑

- 分析片面
- 没有深入

01背景

可用性（Availability）和一致性（Consistency）是分布式系统的基本问题，先有著名的CAP理论定义过分布式环境下二者不可兼得的关系，又有神秘的Paxos协议号称是史上最简单的分布式系统一致性算法并获得图灵奖，再有开源产品ZooKeeper实现的ZAB协议号称超越Paxos。

在大数据场景下，分布式数据库的数据一致性管理是其最重要的内核技术之一，也是保证分布式数据库满足数据库最基本的ACID特性中的“一致性”(Consistency)的保障，在分布式技术发展下，数据一致性的解决方法和技术也在不断的演进。

02分布式系统的挑战

一致性可理解为所有节点都能访问到最新版本的数据，这在单机场景下非常容易实现，使用共享内存和锁即可解决，但数据存储在单机会有两个限制：

1）单机不可 用系统整体将不可用；

2）系统吞吐量受限于单机的计算能力。

消除这两个限制的方法是用多机来存储数据的多个副本，负责更新的客户端会同时更新数据的多个副 本，于是问题就来了，多机之间的网络可能无法连接，当负责更新的客户端无法同时到连接多个机器时，如何能保证所有客户端都能读到最新版本的数据？

03CAP理论

CAP理论由加州大学伯克利分校的计算机教授Eric Brewer在2000年提出，其核心思想是任何基于网络的数据共享系统最多只能满足数据一致性(Consistency)、可用性 (Availability)和网络分区容忍(Partition Tolerance)三个特性中的两个，三个特性的定义如下：

1.数据一致性：等同于所有节点拥有数据的最新版本

2.可用性：数据具备高可用性

3.分区容忍：容忍网络出现分区，分区之间网络不可达

在大规模的分布式环境下，网络分区是必须容忍的现实，于是只能在可用性和一致性两者间做出选择，CAP理论似乎给分布式系统定义了一个悲观的结局，一时间 大家都按照CAP理论在对热门的分布式系统进行判定，譬如认为HBase是一个CP系统，Cassandra是AP系统，我个人认为这是不严谨的，理由是CAP理论是对分布式系统中一个数据无法同时达到可用性和一

致性的断言，而一个系统中往往存在很多类型的数据，部分数据（譬如银行账户中的余额）是需要强 一致性的，而另外一部分数据（譬如银行的总客户数）并不要求强一致性，所以拿CAP理论来划分整个系统是不严谨的， CAP理论带来的价值是指引我们在设计分布式系统时需要区分各种数据的特点，并仔细考虑在小概率的网络分区发生时究竟为该数据选择可用性还是一致性。

04分布式数据一致性

1.数据一致性是什么

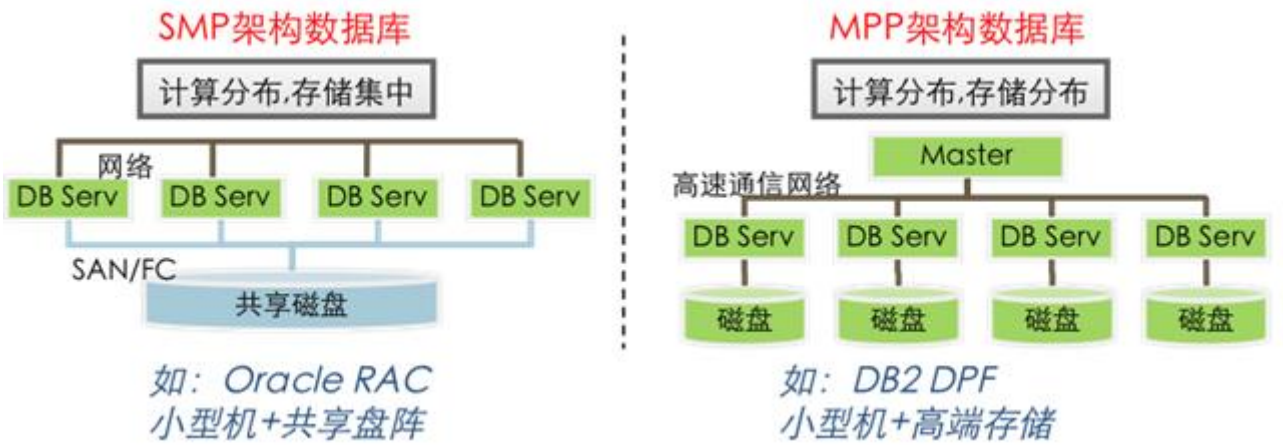
大部份使用传统关系型数据库的DBA在看到“数据一致性”时，第一反应可能都是数据在跨表事务中的数据一致性场景。但是本文介绍的“数据一致性”，指的是“数据在多份副本中存储时，如何保障数据的一致性”场景。

由于在大数据领域，数据的安全不再由硬件来保证，而是通过软件手段，通过同时将数据写入到多个副本中，来确保数据的安全。数据库在同时向多个副本写入记录时，如何确保每个副本数据一致，称为“数据一致性”。

2.关系型数据库如何保障数据一致性

传统的关系型数据库在大数据的场景，对于运行环境–硬件要求都比较高，例如Oracle会建议用户使用小型机+共享存储作为数据库的运行环境，DB2 DPF也同样建议用户采用更好的服务器+高端存储来搭建数据库的运行环境。所以在数据存储安

全的技术要求下，传统关系型数据库在大数据场景更多是依赖硬件的技术来保障数据的安全性。



因为关系型数据库的数据安全是基于硬件来保障，并且数据也不会通过同时存储多份来保障数据的安全，所以关系型数据库的用户默认认为数据存储是一致的。

3.分布式存储如何保障数据一致性

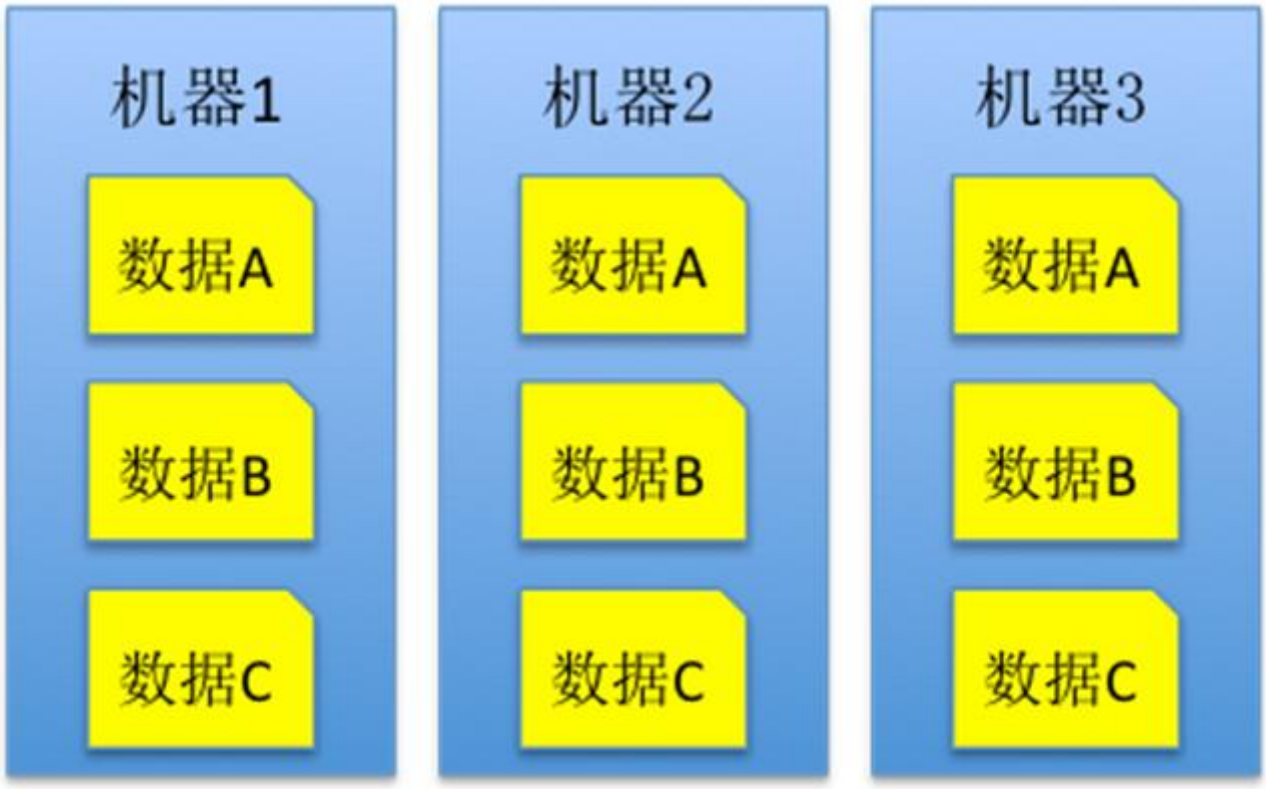
本文在讨论分布式存储时，主要指的是大数据产品中的分布式文件系统和分布式数据库，例如：HDFS。

用户在搞明白分布式存储的数据一致性原理时，必须要先明白为什么他们就需要数据一致性，和分布式存储的数据存储与关系型数据库的数据存储又有什么区别。

大数据技术的诞生，确确实实让系统的性能有新的突破，并且支持硬件以水平扩展的方式来获得线性增长的性能和存储。

这些都是过去传统关系型数据库所无法提供的。另外，大数据技术也抛弃了运行环境必须足够好的硬性要求，而是允许用户通过批量廉价X86服务器+本地磁盘的方式搭建规模集群，从而获得比过去依赖硬件垂直扩展所提供的更强的计算能力和更多的存储空间。

大数据技术的核心思想就是分布式，将一个大的工作任务分解成多个小任务，然后通过分布式并发操作的方式将其完成，从而提高整个系统的计算效率或者是存储能力。而在分布式环境下，由于硬件的要求降低，必然需要大数据产品提供另外一个重要的功能–数据安全。



大数据产品在解决数据安全的方式上，都比较接近，简单来说，就是让一份数据通过异步或者同步的方式保存在多台机器上，从而保障数据的安全。

分布式存储在解决数据安全的技术难点后，又**引入了一个新的技术问题，就是如何保障多个副本中的数据一致性**。目前SequoiaDB是使用**Raft算法来保证数据在多个副本中一致性**。

05Raft算法

1.Raft算法背景

在分布式环境下，最著名的一致性算法应该是Paxos算法，但是由于它实在过于晦涩难懂，并且实现起来极度困难，所以在2013年，Diego Ongaro、John Ousterhout两个人以易懂(Understandability)为目标设计了一套一致性算法Raft。Raft算法最大的特点在于简单易懂，并且实现起来简单

2.Raft算法概述

与Paxos不同，Raft强调的是易懂，Raft和Paxos一样只要保证n/2+1节点正常就能够提供服务。

众所周知当问题较为复杂时可以把问题分解为几个小问题来处理，Raft也使用了分而治之的思想。Raft算法重点解决三个子问题：

1）选举(Leader election)

2）日志复制(Log replication)

3）安全性(Safety)。

Raft算法强化了Leader节点的功能，Follower节点的数据只能够从Leader中获取，所以Follower节点的实现就变得简单，只要负责和Leader保持通信，并且接受Leader推送的数据即可。

3.Raft算法原理

节点角色

Raft算法中，对节点的状态分为3种角色:

1)分别是Leader(领导者)

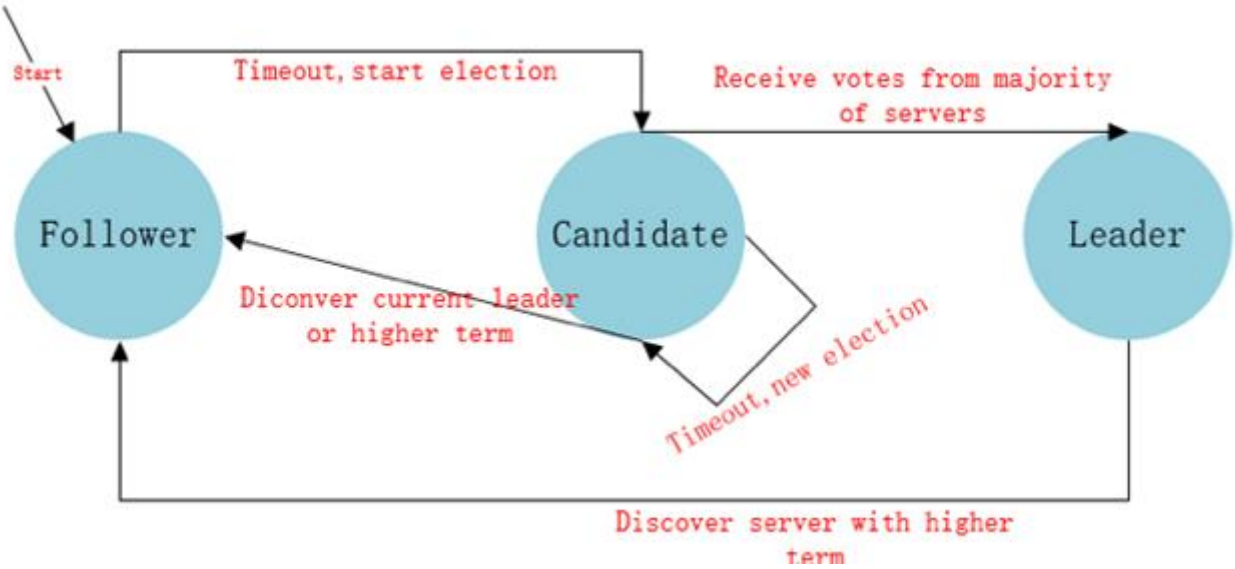
2)Follower(追随者)

3)Candidate(候选者)

Leader，负责处理来自客户端的请求，负责将日志同步到Follower中，并且保证与Follower之间的heartBeat联系;

Follower，当集群刚刚启动时，所有节点均为Follower状态，它的工作主要为响应Leader的日志同步请求，响应Candidate的请求，以及把请求到Follower的事务请求转发给Leader;

Candidate，选举Leader时负责投票，选举出来Leader后，节点将从Candidate状态变为Leader状态。



Terms

在分布式环境下，“时间同步”一直都是老大难的技术难题。Raft为了解决这个问题，将时间划分为一个一个的Term(可以理解为“逻辑时间”)来处理在不同时间段里的数据一致性。

4.日志复制

日志复制主要的作用就是用来保证节点的数据一致性与高可用性。

当Leader被选举出来后，所有的事务操作都必须要经过Leader处理。这些事务操作成功后，将会被按顺序写入到LOG中，每个LOG都包含一个index编号。

5. 安全性

安全性是用于确保每个节点都是按照相同的日志序列进行执行的安全机制。

06分布式数据库数据一致性技术实现

以国产原厂的分布式数据库SequoiaDB为例，SequoiaDB在多副本的部署中，采用Raft算法保证数据在多副本环境中保持一致。

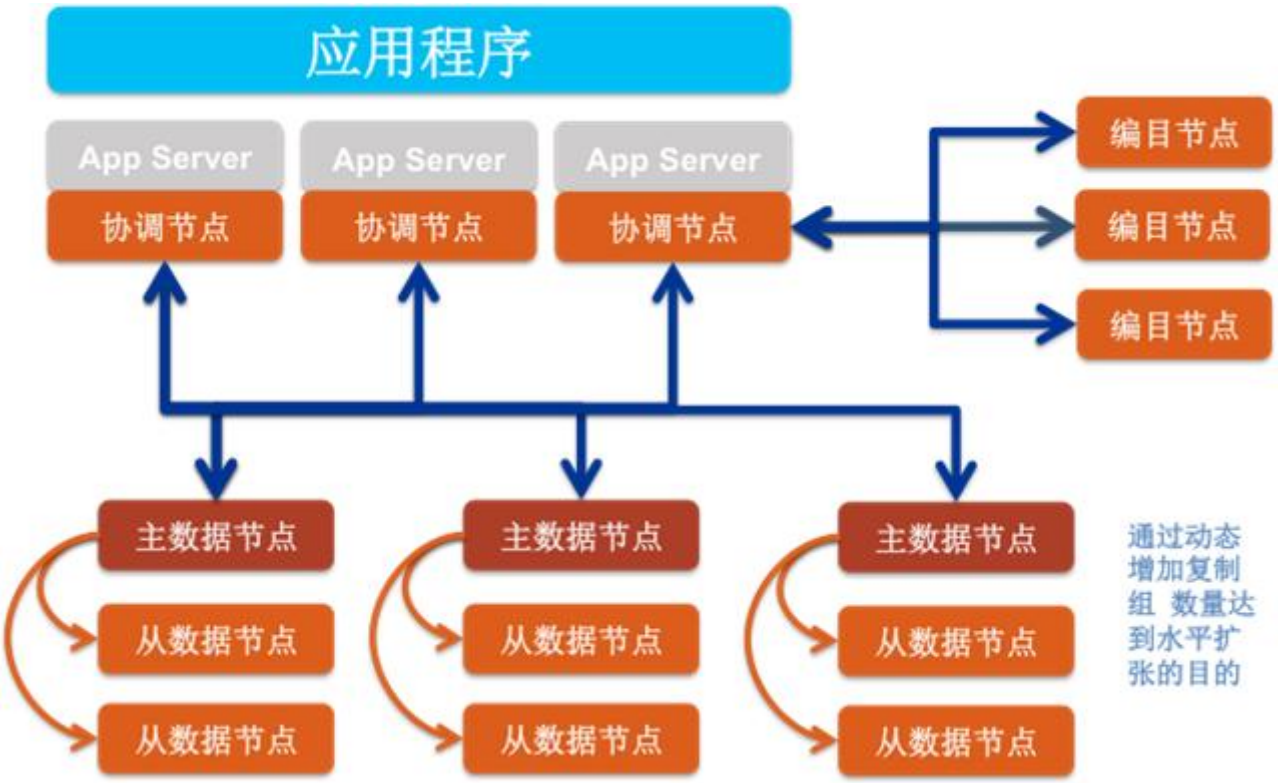
SequoiaDB集群中，总共包含3中角色节点，分别是

1）协调节点

2）编目节点

3）数据节点

由于协调节点本身不存任何数据，所以只有编目节点和数据节点存在事务操作，换言之，编目分区组和数据分区组的副本同步采用Raft算法保证数据一致性。



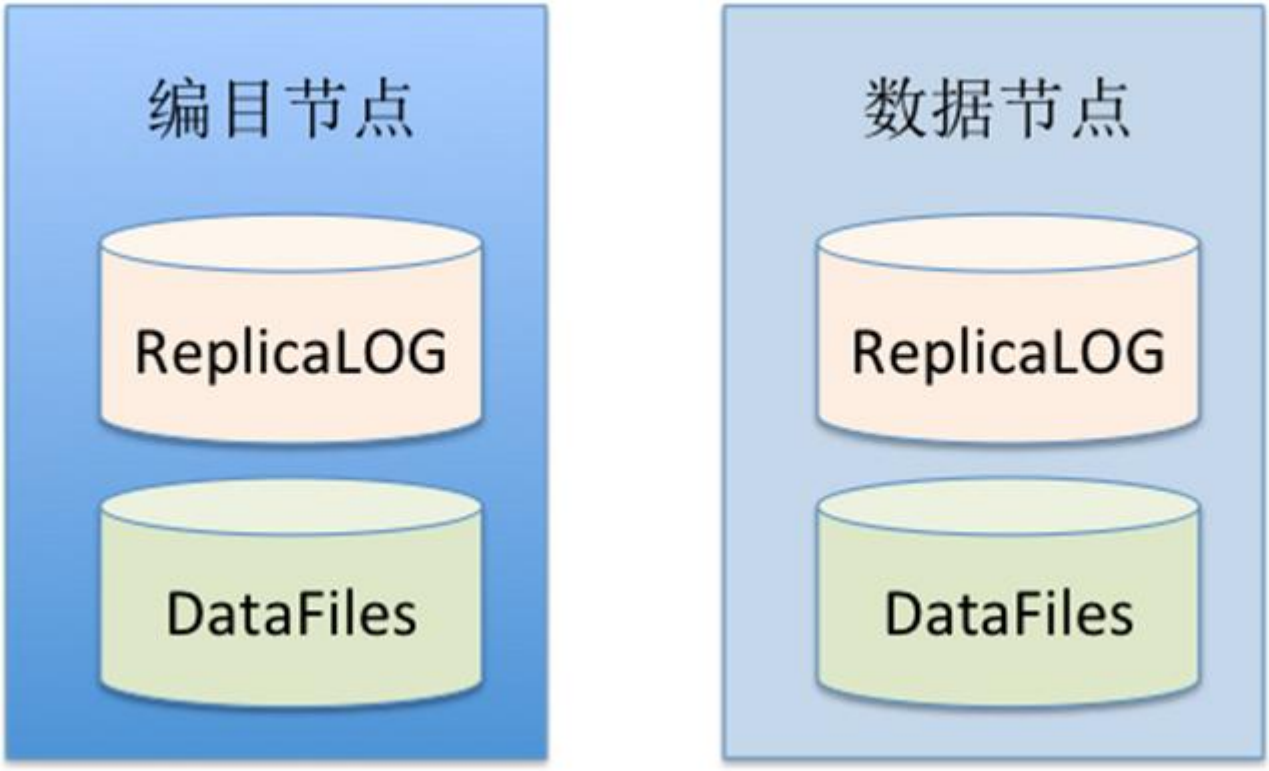
1.编目节点和数据节点的事务日志介绍

编目节点和数据节点由于都是需要存储数据的，并且在集群部署中该，为了确保数据的安全，都是建议采用分布式的方式进行部署，所以在数据同步中，需要采用Raft算法的基本原理进行数据同步。

编目节点和数据节点在存储数据时，共包含两大部分：

1）一个真实的数据文件

2）另一个是事务日志文件



SequoiaDB的节点事务日志，默认情况下由20个64MB(总大小为1.25GB)的文件构成。节点的事务日志主要包含一个index编号和数据操作内容，index编号保持永远递增状态。

另外，SequoiaDB节点的事务日志不会永久保存，而是当所有的事务日志写满后，再重新从第一个文件开始进行覆盖写入。

2.编目分区组的数据一致性

由于编目分区组是保存SequoiaDB集群的元信息，数据同步要求高，所以编目分区组的数据一致性要求为强一致性，即每次向编目分区组执行事务操作时，必须要确保所有的编目节点操作成功，才计算该操作执行成功，否则该事务操作将在整个编目分区组中回退事务日志，以保证分区组内的数据一致性。

3.数据分区组的数据一致性

数据分区组的数据一致性默认情况下为最终一致性性，即只要求主节点执行事务操作成功即视为操作成功，主节点将在未来异步同步ReplicaLOG到从节点上。

4.主从节点的事务日志同步

SequoiaDB的主从节点是通过事务日志同步来保证数据一致性的，并且主从节点的事务日志同步是单线程完成。

如果当主节点和从节点的LSN差距为一条记录，则主节点会主动将最新的事务日志推送给从节点。

如果主节点和从节点的LSN差距超过一条记录，则从节点会主动向主节点请求同步事务日志，主节点收到同步请求后，会将从节点的LSN号到主节点最新的LSN号对应的事务日志打包一次性发送给从节点。

5.从节点日志重放

当从节点获取到主节点推送过来的事务日志后，就会自动解析事务日志和重放。从节点在重放事务日志时，默认情况下会以10并发来重放事务日志。

从节点在执行并发重放日志时有条件限制，即在集合的唯一索引个数<=1的情况下，INSERT、DELETE、UPDATE、LOB WRITE、LOBUPDATE、LOB REMOVE操作可以支持并发重放事务日志。

从节点在做并发重放时，是通过记录的OID进行打散并发执行，这样就可以保证对相同记录的操作不会由于并发重放导致数据不一致。

5. 总结

分布式的数据库，通过Raft算法来确保在分布式情况上数据的一致性，并且编目分区组和数据分区组对数据一致性要求又有所不同，编目分区组始终要求的是数据在多副本请情况下数据强一致性，而数据分区组则可以由用户在创建集合时来执行数据一致性的强度，强度越高，数据安全性越好，但是执行的效率就会相对较差，反之依然。

2. 扩展内容

- 分布式Session共享的4类技术方案，与优劣势比较
- 分布式事务的解决方案，以及原理、总结

- 分布式锁的由来、及Redis分布式锁的实现详解