02 | CAP理论：分布式系统的PH试纸，用它来测酸碱度

2020-02-10 韩健

分布式协议与算法实战 进入课程 >

讲述：于航
时长 15:40　大小 12.56M

你好，我是韩健。

很多同学可能都有这样的感觉，每次要开发分布式系统的时候，就会遇到一个非常棘手的问题，那就是如何根据业务特点，为系统设计合适的分区容错一致性模型，以实现集群能力。这个问题棘手在当发生分区错误时，应该如何保障系统稳定运行，不影响业务。

这和我之前经历的一件事比较像，当时，我负责自研 InfluxDB 系统的项目，接手这个项目后，**我遇到的第一个问题就是，如何为单机开源版的 InfluxDB 设计分区容错一致性模**因为 InfluxDB 有 META 和 DATA 两个节点，它们的功能和数据特点不同，所以我还需要考虑这两个逻辑单元的特点，然后分别设计分区容错一致性模型。

那个时候，我想到了 CAP 理论，并且在 CAP 理论的帮助下，成功地解决了问题。讲到这儿，你可能会问了：为什么 CAP 理论可以解决这个问题呢？

因为在我看来，CAP 理论是一个很好的思考框架，它对分布式系统的特性做了高度抽象，比如抽象成了一致性、可用性和分区容错性，并对特性间的冲突（也就是 CAP 不可能三角）做了总结。一旦掌握它，你就像拥有了引路人，自然而然就能根据业务场景的特点进行权衡，设计出适合的分区容错一致性模型。

那么问题来了：我说的一致性、可用性和分区容错性是什么呢？它们之间有什么关系？你又该如何使用 CAP 理论来思考和设计分区容错一致性模型呢？这些问题就是我们本节课所要讲的重点了。我建议你集中注意力，认真学习内容，还能学以致用，把 CAP 理论应用到日常工作中。

CAP 三指标

我刚刚提到，CAP 理论对分布式系统的特性做了高度抽象，形成了三个指标：

一致性（Consistency）

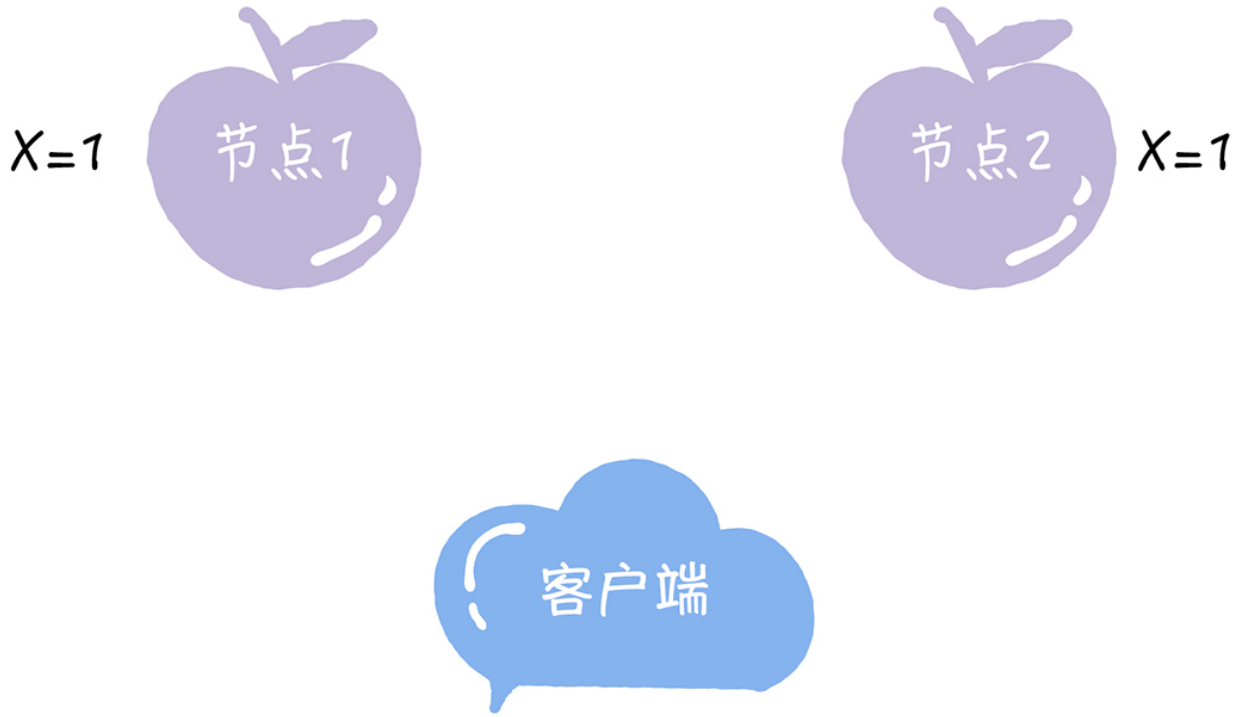
可用性（Availability）

分区容错性（Partition Tolerance）

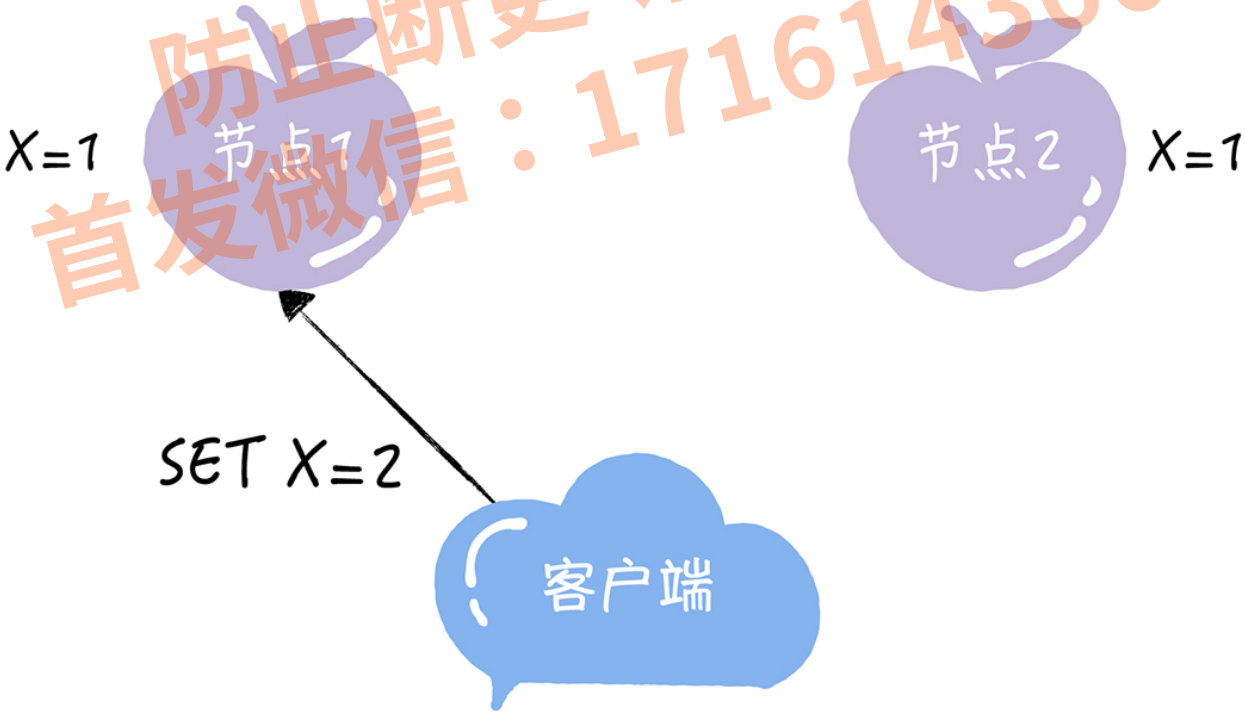
一致性说的是客户端的每次读操作，不管访问哪个节点，要么读到的都是同一份最新的数据，要么读取失败。

你可以把一致性看作是分布式系统对访问本系统的客户端的一种承诺：不管你访问哪个节点，要么我给你返回的都是绝对一致的数据，要么你都读取失败。**你可以看到，一致性强调的不是数据完整，而是各节点间的数据一致。**

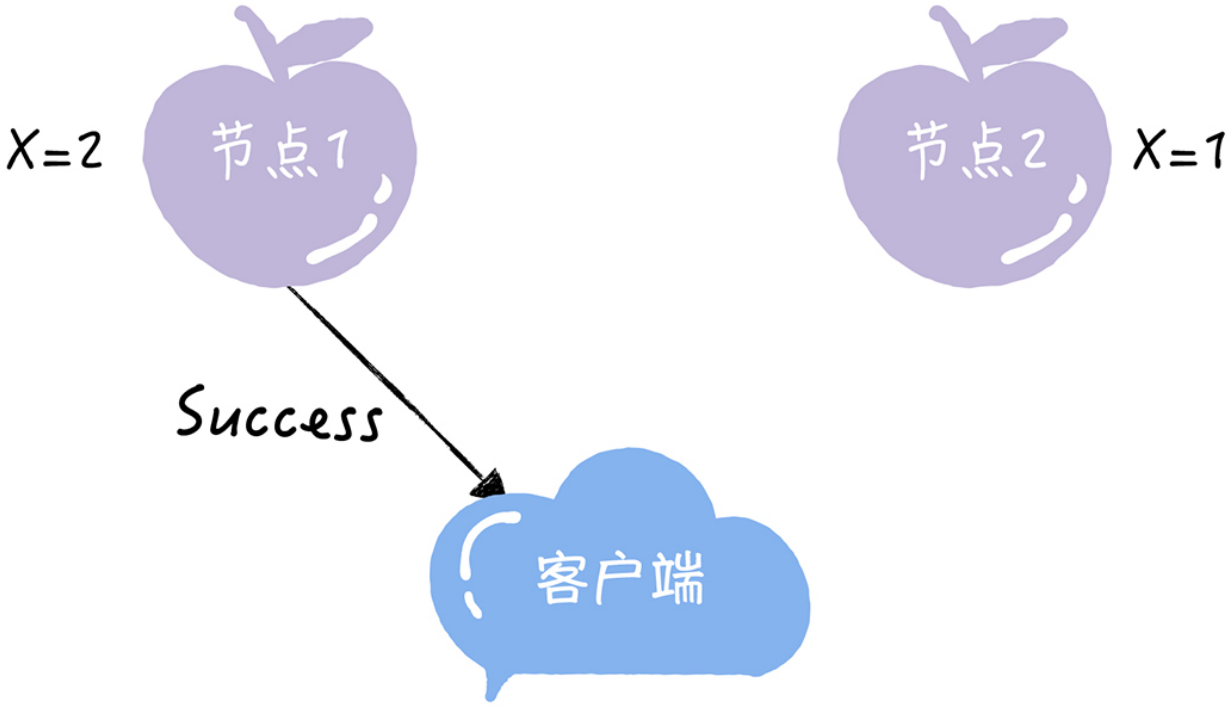
为了帮你理解一致性这个指标，我给你举一个具体的例子。比如，2 个节点的 KV 存储，原始的 KV 记录为“X = 1”。



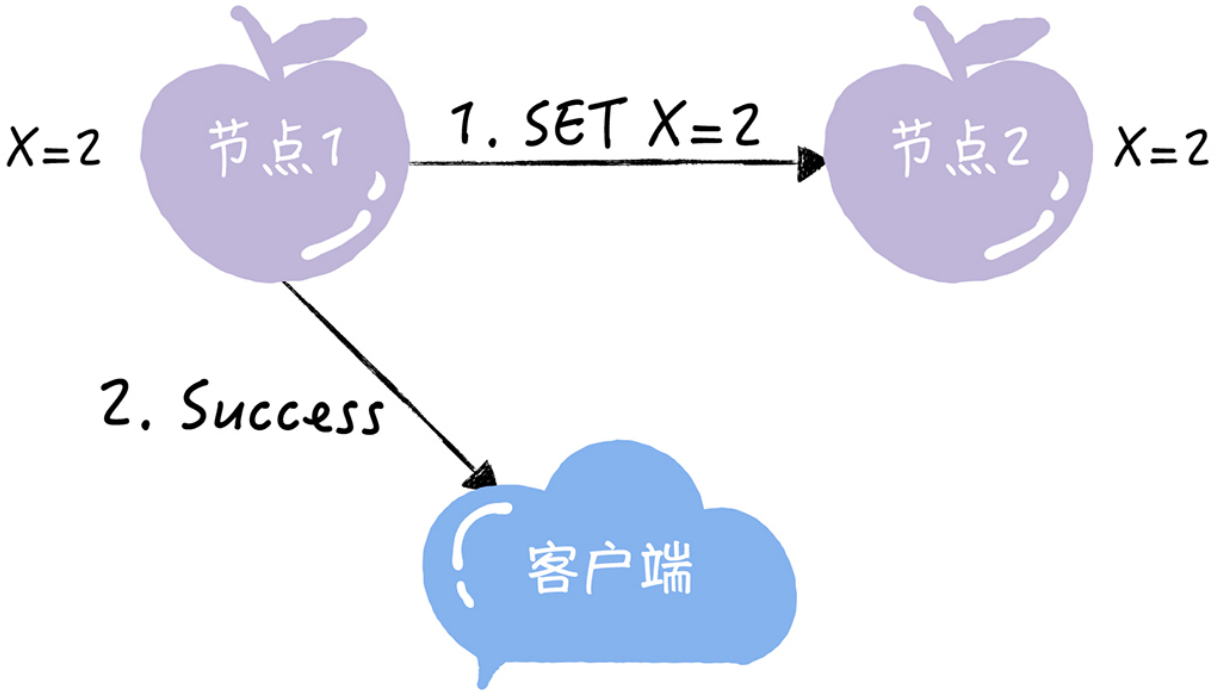
紧接着，客户端向节点 1 发送写请求“SET X = 2”。



如果节点 1 收到写请求后，只将节点 1 的 X 值更新为 2，然后返回成功给客户端，这个时候节点 2 的 X 值还是 1，那么两个节点是非一致性的。



如果节点 1 收到写请求后，通过节点间的通讯，同时将节点 1 和节点 2 的 X 值都更新为 2，然后返回成功给客户端，那么在完成写请求后，两个节点的数据就是一致的了，之后，不管客户端访问哪个节点，读取到的都是同一份最新数据。



一致性这个指标，描述的是分布式系统非常重要的一个特性，强调的是数据的一致。也就是说，在客户端看来，集群和单机在数据一致性上是一样的。

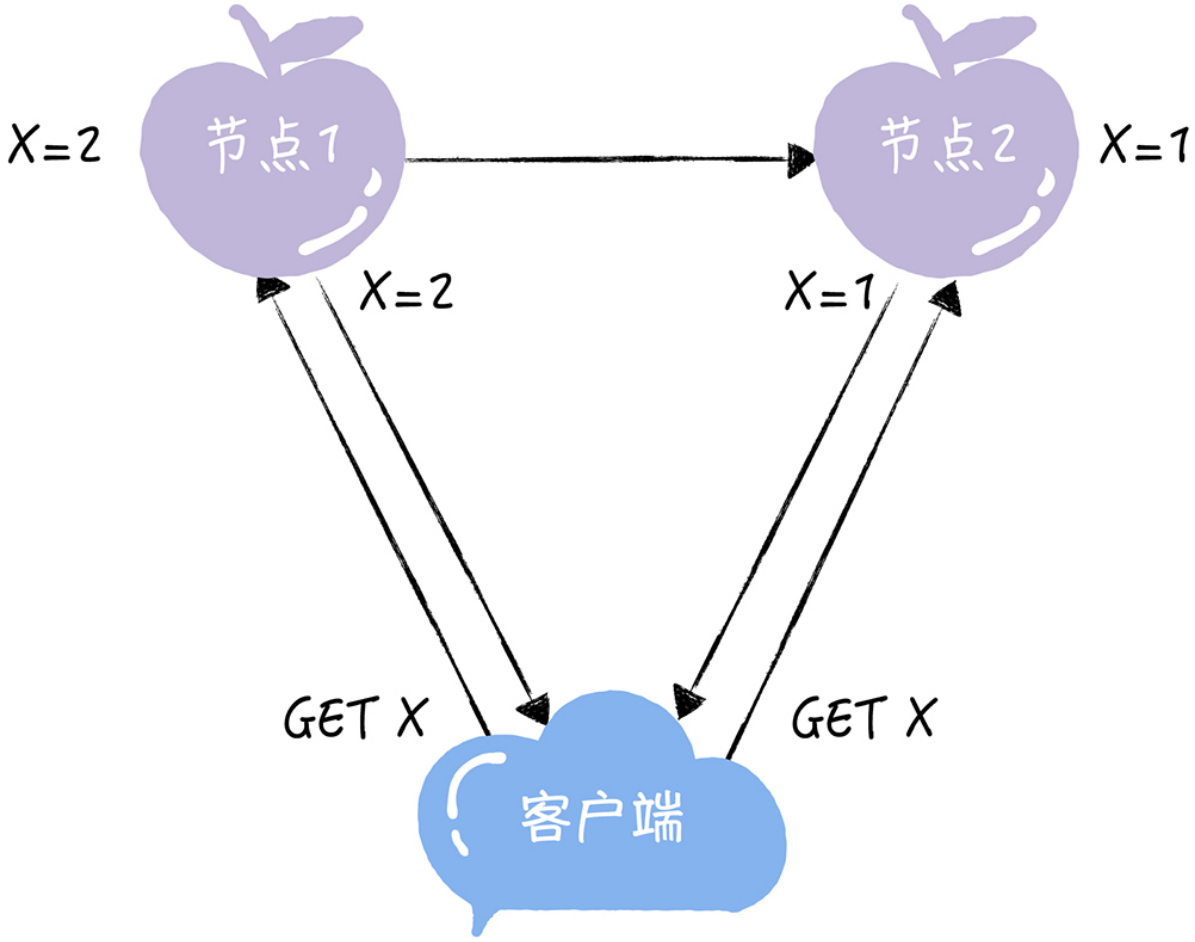
不过集群毕竟不是单机，当发生分区故障的时候，有时不能仅仅因为节点间出现了通讯问题，节点中的数据会不一致，就拒绝写入新数据，之后在客户端查询数据时，就一直返回给客户端出错信息。这句话怎么理解呢？我来举个例子。

业务集群中的一些关键系统，比如名字路由系统，如果仅仅因为发生了分布故障，节点中的数据会不一致，集群就拒绝写入新的路由信息，之后，当客户端查询相关路由信息时，系统就一直返回给客户端出错信息，那么相关的服务都将因为获取不到指定路由信息而不可用、瘫痪，这可以说是灾难性的故障了。

这个时候，我们就需要牺牲数据的一致性，每个节点使用本地数据来响应客户端请求，来保证服务可用，**这就是我要说的另外一个指标，可用性。**

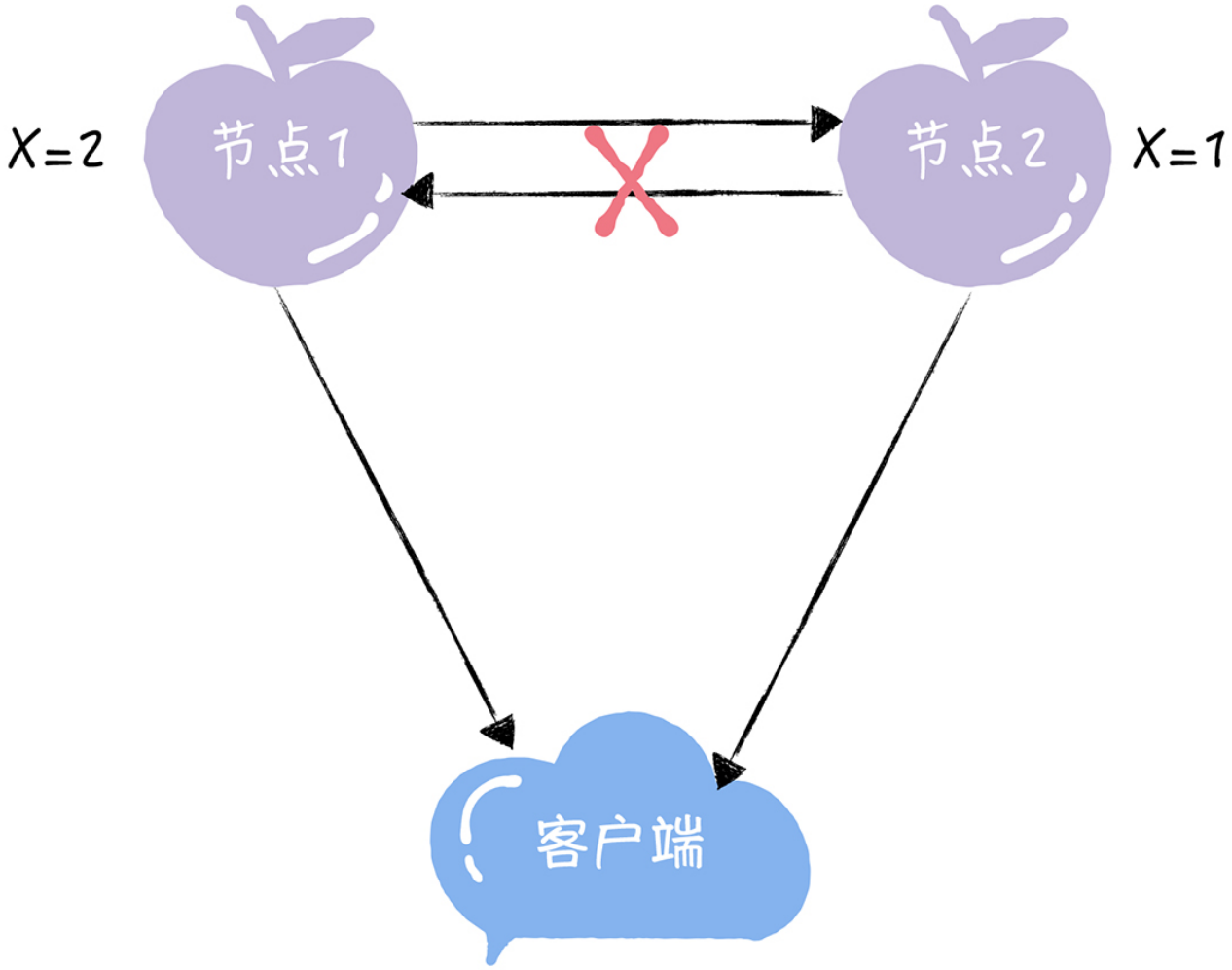
可用性说的是任何来自客户端的请求，不管访问哪个节点，都能得到响应数据，但不保证是同一份最新数据。你也可以把可用性看作是分布式系统对访问本系统的客户端的另外一种承诺：我尽力给你返回数据，不会不响应你，但是我不保证每个节点给你的数据都是最新的。**这个指标强调的是服务可用，但不保证数据的一致。**

我还是用一个例子，帮助你理解一下。比如，用户可以选择向节点 1 或节点 2 发起读操作，如果不管节点间的数据是否一致，只要节点服务器收到请求，就响应 X 的值，那么，2 个节点的服务是满足可用性的。



最后的分区容错性说的是，当节点间出现任意数量的消息丢失或高延迟的时候，系统仍然可以继续提供服务。也就是说，分布式系统在告诉访问本系统的客户端：不管我的内部出现什么样的数据同步问题，我会一直运行，提供服务。**这个指标，强调的是集群对分区故障的容错能力。**

来看下面的图，当节点 1 和节点 2 通信出问题的时候，如果系统仍能提供服务，那么，2 个节点是满足分区容错性的。

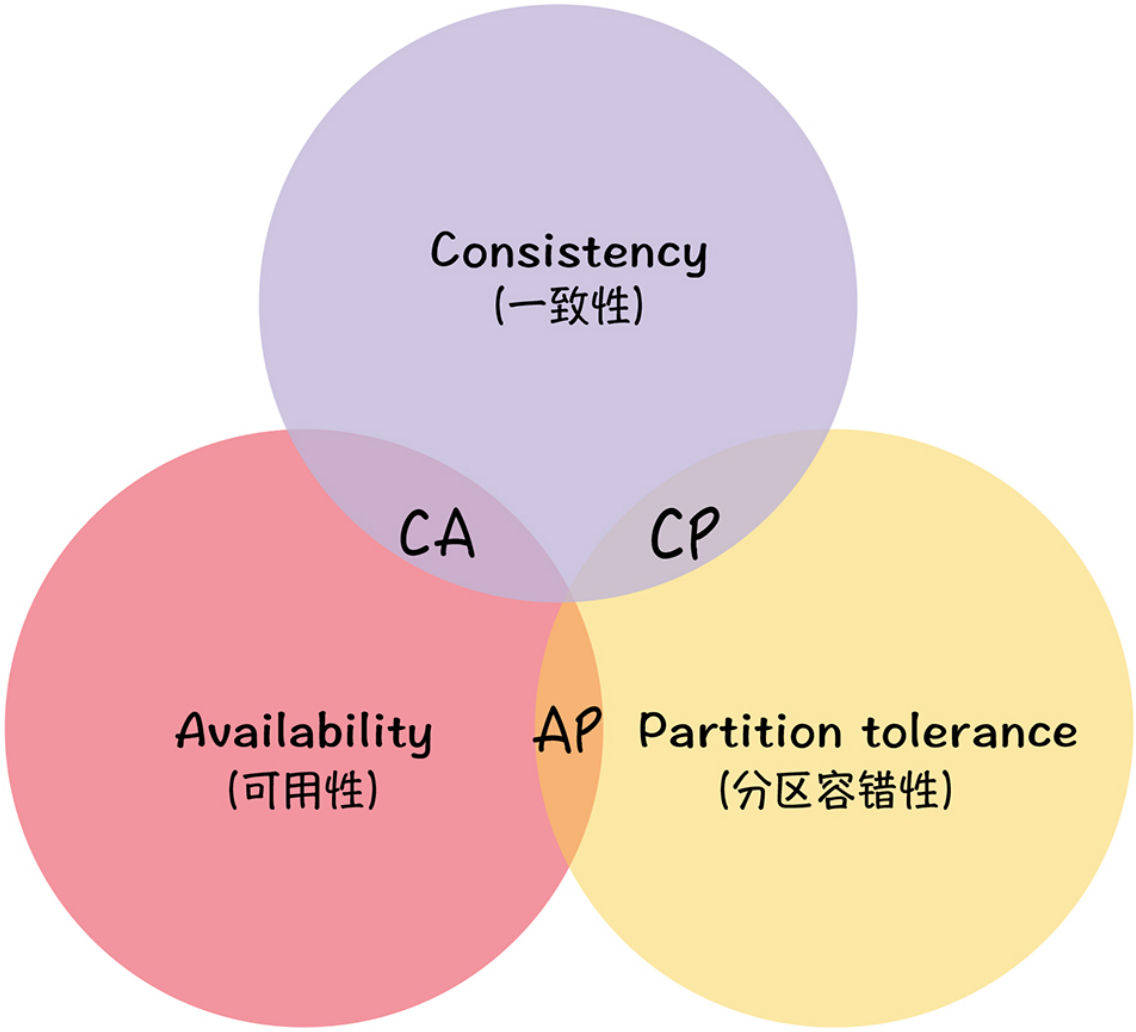


因为分布式系统与单机系统不同，它涉及到多节点间的通讯和交互，节点间的分区故障是必然发生的，**所以我要提醒你，在分布式系统中分区容错性是必须要考虑的。**

现在你了解了一致性、可用性和分区容错性，那么你在设计分布式系统时，是选择一致性？还是可用性？还是分区容错性？还是都可以选择呢？这三个特性有什么冲突么？这些问题就与我接下来要讲的“CAP 不可能三角”有关了。

CAP 不可能三角

CAP 不可能三角说的是对于一个分布式系统而言，一致性（Consistency）、可用性（Availability）、分区容错性（Partition Tolerance）3 个指标不可兼得，只能在 3 个指标中选择 2 个。



CAP 不能三角最初是埃里克·布鲁尔（Eric Brewer）基于自己的工程实践，提出的一个猜想，后被赛斯·吉尔伯特（Seth Gilbert）和南希·林奇（Nancy Lynch）证明，证明过程可以参考论文《Brewer’ s conjecture and the feasibility of consistent, available, partition-tolerant web services》，你记住结论就好了。不过，为了帮你阅读论文，我补充一点：

基于证明严谨性的考虑，赛斯·吉尔伯特（Seth Gilbert）和南希·林奇（Nancy Lynch）对指标的含义做了预设和限制，比如，将一致性限制为原子性。

说了这么多，那么 CAP 理论是怎么解决我在开篇提到的问题呢？或者说，你要如何使用 CAP 理论来思考和设计分区容错一致性模型呢？

如何使用 CAP 理论

我们都知道，只要有网络交互就一定会有延迟和数据丢失，而这种状况我们必须接受，还必须保证系统不能挂掉。所以就像我上面提到的，节点间的分区故障是必然发生的。也就是

说，分区容错性（P）是前提，是必须要保证的。

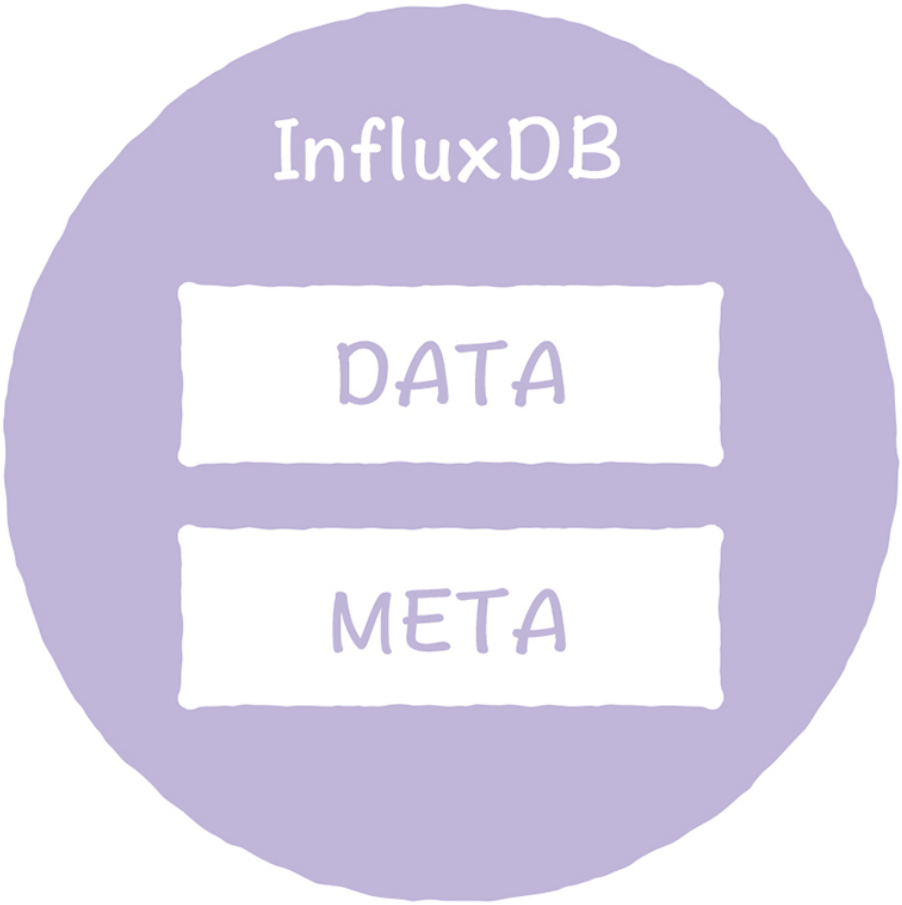
现在就只剩下一致性（C）和可用性（A）可以选择了：要么选择一致性，保证数据绝对一致；要么选择可用性，保证服务可用。那么 CP 和 AP 的含义是什么呢？

> 当选择了一致性（C）的时候，如果因为消息丢失、延迟过高发生了网络分区，部分节点无法保证特定信息是最新的，那么这个时候，当集群节点接收到来自客户端的写请求时，因为无法保证所有节点都是最新信息，所以系统将返回写失败错误，也就是说集群拒绝新数据写入。
>
> 当选择了可用性（A）的时候，系统将始终处理客户端的查询，返回特定信息，如果发生了网络分区，一些节点将无法返回最新的特定信息，它们将返回自己当前的相对新的信息。

这里我想强调一点，大部分人对 CAP 理论有个误解，认为无论在什么情况下，分布式系统都只能在 C 和 A 中选择 1 个。 其实，在不存在网络分区的情况下，也就是分布式系统正常运行时（这也是系统在绝大部分时候所处的状态），就是说在不需要 P 时，C 和 A 能够同时保证。只有当发生分区故障的时候，也就是说需要 P 时，才会在 C 和 A 之间做出选择。而且如果各节点数据不一致，影响到了系统运行或业务运行（也就是说会有负面的影响），推荐选择 C，否则选 A。

那么我当时是怎么根据场景特点，进行 CAP 权衡，设计适合的分布式系统呢？为了便于你理解，我先来说说背景。

开源版的 InfluxDB，缺乏集群能力和可用性，而且，InfluxDB 是由 META 节点和 DATA 节点 2 个逻辑单元组成，这 2 个节点的功能和数据特点不同，需要我们分别为它们设计分区容错一致性模型。



InfluxDB 程序的逻辑架构示意图

我具体是这么设计的：

作为分布式系统，分区容错性是必须要实现的，不能因为节点间出现了分区故障，而出现整个系统不能用的情况。

考虑到 META 节点保存的是系统运行的关键元信息，比如数据库名、表名、保留策略信息等，所以必须保持所有节点的一致性，这样才能避免由于各节点元信息不一致，导致时序数据记录不一致或者影响系统运行。比如，数据库 Telegraf 的信息在一些节点上存在，在另外一些节点上不存在，那么将导致向某些节点写入时序数据记录失败，所以，我选择 CAP 理论中的 C 和 P，采用 CP 架构。

DATA 节点保存的是具体的时序数据记录，比如一条记录 CPU 负载的时序数据，“cpu_usage,host=server01,location=cn-sz user=23.0,system=57.0”。虽然不是系统运行相关的元信息，但服务会被访问频繁，水平扩展、性能、可用性等是关键，所以，我选择了 CAP 理论中的 A 和 P，采用 AP 架构。

你看，我用 CAP 理论进行思考，并分别设计了 InfluxDB 的 META 节点和 DATA 节点的分区容错一致性模型，而你也可以采用类似的思考方法，设计出符合自己业务场景的分区容错一致性模型。

那么假设我当时没有受到 CAP 理论的影响，或者对 CAP 理论理解不深入，DATA 节点不采用 AP 架构，而是直接使用了现在比较流行的分区容错一致性算法，比如使用 Raft 算法，会有什么痛点呢？

受限于 Raft 的强领导者模型。所有请求都在领导者节点上处理，整个集群的性能等于单机性能。这样会造成集群接入性能低下，无法支撑海量或大数据量的时序数据。

受限于强领导者模型，以及 Raft 的节点和副本一一对应的限制，无法实现水平扩展，分布式集群扩展了读性能，但写性能并没有提升。这样会出现写性能低下，和因为架构上的限制，无法提升写性能的问题。

Raft 的“一切以领导者为准”的日志复制特性，会导致 DATA 节点丢数据，出现时序数据记录缺失的问题。

关于 Raft 算法的一些细节（比如强领导模型），我会在 07 讲详细带你了解，这里你知道有这么回事儿就可以了。

最后我想再次强调的是，一致性不等同于完整性， 有些技术团队基于数据完整性的考虑，使用 Raft 算法实现 DATA 节点的数据的分布式一致性容错，恰恰是这个设计，会导致 DATA 节点丢数据。我希望你能注意到这一点。

那么在这里，我也想考考你：如果 META 节点采用 AP 架构，会有什么痛点呢？你可以思考一下。

内容小结

本节课我主要带你了解了 CAP 理论，以及 CAP 理论的应用，我希望你明确的重点如下：

CA 模型，在分布式系统中不存在。因为舍弃 P，意味着舍弃分布式系统，就比如单机版关系型数据库 MySQL，如果 MySQL 要考虑主备或集群部署时，它必须考虑 P。

CP 模型，采用 CP 模型的分布式系统，一旦因为消息丢失、延迟过高发生了网络分区，就影响用户的体验和业务的可用性。因为为了防止数据不一致，集群将拒绝新数据的写

入，典型的应用是 ZooKeeper，Etcd 和 HBase。

AP 模型，采用 AP 模型的分布式系统，实现了服务的高可用。用户访问系统的时候，都能得到响应数据，不会出现响应错误，但当出现分区故障时，相同的读操作，访问不同的节点，得到响应数据可能不一样。典型应用就比如 Cassandra 和 DynamoDB。

在我看来，CAP 理论像 PH 试纸一样，可以用来度量分布式系统的酸碱值，帮助我们思考如何设计合适的酸碱度，在一致性和可用性之间进行妥协折中，设计出满足场景特点的分布式系统。关于酸（Acid）和碱（Base），我会在 03 和 04 讲带你了解。

最后我想说的是，在当前分布式系统开发中，延迟是非常重要的一个指标，比如，在 QQ 后台的名字路由系统中，我们通过延迟评估服务可用性，进行负载均衡和容灾；再比如，在 Hashicorp/Raft 实现中，通过延迟评估领导者节点的服务可用性，以及决定是否发起领导者选举。所以，我希望你在分布式系统的开发中，也能意识到延迟的重要性，能通过延迟来衡量服务的可用性。

课堂思考

既然我提了 CAP 理论是一个很好的思考框架，能帮助我们思考，如何进行权衡，设计适合业务场景特性的分布式系统，那么你不妨思考一下，CP 模型的 KV 存储和 AP 模型的 KV 存储，分别适合怎样的业务场景呢？欢迎在留言区分享你的看法，与我一同讨论。

最后，感谢你的阅读，如果这篇文章让你有所收获，也欢迎你将它分享给更多的朋友。

上一篇　01 | 拜占庭将军问题：有叛徒的情况下，如何才能达成共识?

精选留言

写留言

由作者筛选后的优质留言将会公开显示，欢迎踊跃留言。