BASE

jask

2024-08-11

BASE

BASE 的核心是基本可用和最终一致性。软状态描述的是实现服务可用性的时候系统数据的一种过渡状态，也就是说不同节点间，数据副本存在短暂的不一致。

基本可用

基本可用是说，当分布式系统在出现不可预知的故障时，允许损失部分功能的可用性，保障核心功能的可用性。（流量削峰，延迟响应，体验降级，过载保护）

最终一致性

几乎所有的互联网系统采用的都是最终一致性，只有在实在无法使用最终一致性，才使用强一致性或事务，比如，对于决定系统运行的敏感元数据，需要考虑采用强一致性，对于与钱有关的支付系统或金融系统的数据，需要考虑采用事务。

在实践中，你也可以这样思考：如果业务的某功能无法容忍一致性的延迟（比如分布式锁对应的数据），需要实现的是强一致性；如果能容忍短暂的一致性的延迟（比如 QQ 状态数据），就可以考虑最终一致性。

如何实现最终一致性?

以最新写入的数据为准，比如 AP 模型的 KV 存储采用的就是这种方式；以第一次写入的数据为准，如果你不希望存储的数据被更改，可以以它为准。以最新写入的数据为准，比如 AP 模型的 KV 存储采用的就是这种方式；### 实现的具体方式读时修复：在读取数据时，检测数据的不一致，进行修复。比如 Cassandra 的 Read Repair 实现，具体来说，在向 Cassandra 系统查询数据的时候，如果检测到不同节点的副本数据不一致，系统就自动修复数据。写时修复：在写入数据，检测数据的不一致时，进行修复。比如 Cassandra 的 Hinted Handoff 实现。具体来说，Cassandra 集群的节点之间远程写数据的时候，如果写失败就将数据缓存下来，然后定时重传，修复数据的不一致性。异步修复：这个是最常用的方式，通过定时对账检测副本数据的一致性，并修复。

如何使用 BASE 理论

DATA 节点的核心功能是读和写，所以基本可用是指读和写的基本可用。那么我们可以通过分片和多副本，实现读和写的基本可用。也就是说，将同一业务的数据先分片，然后再以多份副本的形式分布在不同的节点上。比如下面这张图，这个 3 节点 2 副本的集群，除非超过一半的节点都故障了，否则是能保障所有数据的读写的。## 总结 1.BASE 理论是对 CAP 中一致性和可用性权衡的结果，它来源于对大规模互联网分布式系统实践的总结，是基于 CAP 定理逐步演化而来的。它的核心思想是，如果不是必须的话，不推荐实现事务或强一致性，鼓励可用性和性能优先，根据业务的场景特点，来实现非常弹性的基本可用，以及实现数据的最终一致性。2.BASE 理论主张通过牺牲部分功能的可用性，实现整体的基本可用，也就是说，通过服务降级的方式，努力保障极端情况下的系统可用性。3. ACID 理论是传统数据库常用的设计理念，追求强一致性模型。BASE 理论支持的是大型分布式系统，通过牺牲强一致性获得高可用性。BASE 理论在很大程度上，解决了事务型系统在性能、容错、可用性等方面痛点。另外我再多说一句，BASE 理论在 NoSQL 中应用广泛，是 NoSQL 系统设计的事实上的理论支撑。