在「不可靠」硬件上，分布式数据库如何保证数据可靠性和服务可用性？

Original:鸷腾 OceanBase Yesterday

OB君：“数据不能丢，服务不能停”，可以说这句话道出了用户对数据库的核心能力的要求。然而，传统的商业数据库必须依赖高可靠的硬件才能实现数据可靠性和服务可用性。OceanBase作为一款成熟的企业级分布式数据库，基于普通PC服务器，就能够做到传统高端硬件环境下的数据可靠性和服务可用性，而且还能做得更好！跟我们一起看看OceanBase的技术秘诀吧！

作者介绍

乔国治（鸷腾）

蚂蚁金服OceanBase数据库团队高级技术支持专家。曾在亚信、IBM、百度等公司有多年的数据库使用及技术支持经验，现在蚂蚁金服负责OceanBase数据库的外部客户推广及技术支持工作。

Part 1 前言

说到数据可靠性和服务可用性，在数据库领域真是老生常谈的话题，可以说从数据库诞生之日起就如影随形。如果要用一句话来概括数据库对数据可靠性和服务可用性的要求，可以借用OceanBase数据库创始人阳振坤老师的一句话：“数据不能丢，服务不能停”。可以说，这句话也道出了用户对数据库的一个核心能力要求：除了功能完善、使用方便之外，还要**绝对安全、足够健壮**。可以说，为了满足这两个看似简单的要求，在数据库领域诞生了大量的技术和论文，也让无数人绞尽了脑汁。

在传统的商业数据库产品（如Oracle、DB2）中，虽然也有一些行之有效的软件技术（如Redo Log、主从热备技术等）用来提高数据可靠性和服务可用性，但整体来说对硬件的稳定性有很强的依赖。而传统的企业级服务器（如IBM 的Mainframe、AS400、Power等）和EMC、IBM等厂商的高端存储产品，能够很好的保证硬件的稳定性，因此也就成为了Oracle为代表的传统数据库产品的理想平台，这也就是"IOE"一词的由来。可以说，I和E的重要职责就是保障O的稳定运行。

不知不觉间，IT世界进入PC服务器的时代。和传统架构相比，PC服务器能降低成本并带来扩展性上的便利，逐渐成为以互联网为代表的众多企业用户的首选。但PC作为服务器也为用户带来了一个棘手的问题，那就是硬件（服务器、内置磁盘、网卡等）的可靠性明显下降了。虽然也有一些其它机制（比如RAID、外挂磁盘阵列等）可以用来改善这种情况，但不能从根本上解决问题。

近些年来新兴的数据库产品，尤其是分布式数据库，几乎无一例外地采用了PC服务器架构，**那在这种相对不稳定、不可靠的硬件条件下，分布式数据库如何保证数据可靠性和服务可用性呢**？对习惯了传统高端硬件稳定性的企业用户来说，这个问题的答案将直接决定业务要承受多大的风险。

针对这个问题，OceanBase给出了非常明确的答复：**在PC服务器架构下，OceanBase不但能够满足传统高端硬件环境下的数据可靠性和服务可用性，而且还能做得更好！** 可以说，新技术趋势（PC架构）所带来的问题（硬件可靠性降低），反而倒逼了技术自身的演进，大家开始认真思考如何在“不可靠”硬件环境下保证数据的“可靠性”，并进一步保证服务的“可用性”。最终的结果，是更多的从软件层面引入保障机制，来弥补硬件环境的不足。后文将从多个方面来阐述OceanBase具体是如何做的。

Part 2 OceanBase如何保证数据可靠性

在传统数据库中，有几种常用的手段来保证数据可靠性：

1） Redo Log
2） 主从热备
3） 备份/恢复
4） 存储层数据校验

这些技术可以从很大程度上提高数据的可靠性，但似乎都无法做到完美（即RPO=0），也就是无法保证数据完全无损，下面我们简单分析一下原因。

首先看**Redo Log**。采用Write-Ahead-Log（WAL）模式的Redo Log可以保证数据库中已提交的数据不会丢失，如果已提交的数据还在内存中就发生了宕机等意外，利用Redo Log可以恢复这些还未持久化的数据。但这里有一个前提，就是Redo Log自身必须绝对可靠，如果Redo Log所在的存储发生损坏，那么这一前提便不复存在。

因此，Redo Log所带来的数据可靠性其实取决于硬件的可靠性，说到底还是要依赖高端硬件。此外，如果是已经持久化的数据遇到了硬件损坏（比如“坏页”问题），并且对应的Redo Log已经被覆盖，那么Redo Log也无能为力了。

有了主从热备技术（比如Oracle DataGuard，IBM DB2 HADR等）之后，Redo Log可以写两份甚至多份了，即使主节点遇到硬件故障，仍然可以用备节点的Redo Log来恢复数据，看上去可以做到RPO=0了。但其实不然，虽然主从热备技术通常都提供“数据强同步”的手段（比如Oracle Data Guard中的Max-Protection模式，IBM DB2 HADR中的Sync模式）来保证RPO=0，但实际系统中几乎没有人采用。为什么呢？原因很简单，这种模式下一旦备节点发生故障，或者主备之间的网络发生故障，那么主节点的正常交易就会受拖累；换句话说，备节点不但没有提高整体稳定性，反而降低了整体稳定性，得不偿失。所以，在实际生产系统中部署过主从热备的朋友都知道，几乎没有人会采用数据强一致模式，也就无法做到RPO=0。

除了Redo Log以外，我们还有用来保底的“备份/恢复”手段。但备份/恢复机制在时效性上有明显缺陷，只能最为应对最差情况的“最后一发子弹”，不到万不得已绝不使用。而且，备份文件的可靠性最终依然是依赖硬件的可靠性。

最后，数据库的存储层通常都会有数据校验机制，用来检测存储层的“静默错误”和潜在的软件错误。但数据校验机制更多的是用来在事后发现错误，无法预防或者解决错误，后面我们会介绍OceanBase如何将这种技术和其它机制结合起来，以提高整体的数据可靠性。

综合上面的分析，可以看到传统数据库最终还是要依赖高端硬件的可靠性来保证数据的可靠性，只依靠数据库自身的能力是无法保证RPO=0的。

那么在PC服务器的环境里，硬件的可靠性明显不足，OceanBase这样的分布式数据库又是如何来保证数据可靠性呢？

前面提到过，我们更多的是在软件层面引入保障机制，比如以**Paxos**（以及后面衍生出来的Raft）为代表的“分布式一致性协议”。**OceanBase充分利用了Paxos协议，并将Paxos协议和传统的WAL机制结合起来，每一次Redo Log落盘时，都会以强一致方式同步到Paxos组中多数派（leader+若干follower）副本的磁盘中**，这样做有两个好处：

1）在Paxos组中任意少数派副本发生故障的情况下，剩下的多数派副本都能保证有最新的Redo Log，因此就能避免个别硬件故障带来的数据损失，保证RPO=0。

2）Paxos协议中的数据强一致是针对“多数派”副本而言，而不像主从热备那样要求“所有”副本的数据都保持强一致。如果Paxos组中有少数派follower副本发生故障，剩下的多数派副本（leader+若干follower）之间的数据强一致完全不受影响，这就解决了前面提到的问题：主从热备模式下备副本故障拖累主副本的可用性。

综合以上两点，**OceanBase利用Paxos协议可以保证RPO=0**，且不必担心应用的性能会受到影响，这也是OceanBase和传统数据库在数据可靠性方面最显著的不同点。

前面提到过的“数据校验”机制，在众多工程实践中已经被证实为一种行之有效的机制。但是，如果要在采用Paxos协议的分布式数据库中实施数据校验机制，情况将更加复杂：除了要关注单个物理节点的“静默错误”，还要保证多个副本之间数据的一致性。

假设网络传输出了问题，导致错误的数据从leader副本发送到了follower副本；如果follower副本在不知情的情况下，将这些错误的数据当成正确的数据存储起来，一旦leader副本发生问题，而有错误数据的follower副本又接管了服务，那么久直接造成了用户的数据损失。OceanBase也在存储层引入了数据校验机制，但我们加入了更多的技术手段以避免上述问题，大致包含以下内容：

1）Redo Log的数据校验

首先，Redo Log在落盘的时候会加上数据校验信息，用来应对可能发生的磁盘静默错误。此外，为了保证一个Paxos组中多个副本之间Redo Log的一致性，Redo Log在leader发送和follower接受时都会检查数据校验信息，避免网络传输问题导致的数据错误。

2）数据盘上的校验信息

和Redo Log类似，数据盘上的数据也会有校验信息以应对磁盘静默错误。但由于OceanBase是通过Redo Log实现Paxos组中多个副本之间的数据同步，数据盘上的数据并不会通过网络传输在多个副本间同步，因此不需要副本间的实时校验。

3）副本间的检查点一致性校验

刚刚提到了，OceanBase不会对数据盘上的数据做副本间的“实时”校验，但我们还是会在一些特定的检查点，对多个副本之间的数据盘做一致性检查。这个检查点选在了OceanBase的“每日合并”点，主要的原因，是每日合并动作本身就要对大量数据做归并和重新写入，刚好可以利用这个时机做数据的一致性检查。通过这个检查，进一步在存储层确保了多个副本之间的数据一致性，提高了数据可靠性。

4）数据表和索引表之间的数据一致性校验

对于有关联关系的数据对象，OceanBase会做额外的检查以保证它们之间的数据一致性。比较典型的例子就是索引和它的数据表，OceanBase会在一些特定的检查点（如每日合并点）做索引和数据表之间的一致性检查，进一步提高数据可靠性。

5）定期做数据校验信息检查

上面提到的一些数据校验措施（比如Redo Log和数据盘上的数据校验信息），主要目的还是在数据中埋入校验信息。但光有校验信息是不够的，还要能够利用校验信息及时发现磁盘的静默错误，否则就只能等到访问数据的时候才能发现错误，为时已晚。为了应对这个问题，OceanBase在后台有定期的检查任务，在不影响在线业务的前提下，利用数据校验信息主动检查磁盘静默错误，一旦发现错误会及时通知用户，尽快采取补救措施。

最后，OceanBase也和传统数据库一样提供完善的备份/恢复机制，包括全量备份功能和增量备份功能。而且OceanBase的增量备份是以不间断的后台daemon任务形式持续进行，完全不影响在线业务，降低了运维操作的复杂度。不过从分布式数据库的运行实践来看，在实际系统中极少发生Paxos组中多数派副本同时毁坏的情况，因此基本不会真正用到备份来恢复数据。

Part 3 OceanBase如何保证服务可用性

在数据可靠性有保证的前提下，服务可用性就成为了另一个焦点：**如果某个服务节点发生了故障，用户不但希望数据不丢（RPO=0），而且希望服务能够尽快恢复（RTO越小越好）**。

在历史上，在传统数据库的高可用能力不足时，有很多种高可用方案是结合了硬件和操作系统的高可用能力，不过这些方案通常在架构上过于复杂，而且无法在数据库层面保证数据的一致性。随着数据库内置的高可用能力逐渐完善，用户也转向了数据库自带的高可用方案，典型代表就是“主从热备”技术，比如Oracle Data Guard、IBM DB2 HADR等。

但是，主从热备技术在高可用上有一个很难解决的问题：当主节点故障的时候，如何让备用节点快速接管服务。如果让备用节点判断主节点的状态，并且在主节点故障时“自动”接管服务，那么就会面临一个致命的问题，就是“脑裂（Split-brain）”：备用节点和主节点同时提供服务了，两个节点间的数据将再也无法保持一致。

为了避免脑裂问题，数据库的主从热备机制都不会提供“备库自动接管服务”的能力，备库的接管动作要引入人工决策的流程，需要人为在备用节点发起服务接管的动作。这样一来，RTO就会达到数十分钟甚至以小时计。为了避免脑裂问题，同时又减小RTO，有些数据库产品引入了自动failover的机制，比如Oracle数据库的Fast-Start Failover（FSFO）；但由于引入了额外的组件或者服务，整个解决方案的复杂度明显增大，而且这些组件或者服务自身的高可用又成了新的问题。总体来说，**由于主从热备的本质没有发生改变，只是额外引入了第三方仲裁者的角色，所以这种方案并没有解决根本问题，也不能100%保证避免脑裂**。

前文已经提到， OceanBase利用了Paxos协议中的多数派共识机制来保证数据的可靠性，在高可用方面，OceanBase是利用了同样的机制。首先，根据Paxos协议，在任一时刻只有多数派副本达成一致时，才能推选一个leader，其余的少数派副本则不具备推选leader的资格。

其次，如果正在提供服务的leader副本遇到故障而无法继续提供服务，只要其余的follower副本满足多数派并且达成一致，他们就可以推选一个新的leader来接管服务，而正在提供服务的leader自己无法满足多数派条件，将自动失去leader的资格。因此，我们可以看到Paxos协议在高可用方面有明显的优势：

1）从理论上保证了任一时刻至多有一个leader，彻底杜绝了脑裂的情况。

2）由于不再担心脑裂，当leader故障而无法提供服务时，follower便可以自动触发选举来产生新的leader并接管服务，全程无须人工介入。

这样一来，不但从根本上解决了脑裂的问题，还可以利用自动重新选举大大缩短RTO，可以说完美解决了主从热备技术在高可用上所面临的难题。

当然，这里面还有一个很重要的因素，那就是leader出现故障时，follower能在多长时间内感知到leader的故障并推选出新的leader，这个时间直接决定了RTO的大小。在OceanBase中，为了能够及时感知Paxos组中各个副本（包括leader和follower）的状态，在各个副本之间会有定期探活的消息。另一方面，探活机制虽然能够检测到节点的故障，但是在网络不稳定的情况下，也可能由于偶发的探活消息丢包而产生“误报（False-alarm）”的情况。为了避免误报对系统稳定性带来的影响，OceanBase也采取了很多应对措施：

1）首先，探活消息的周期必须要合理。

如果周期太长就不能及时感知到节点故障，如果周期太短就会增大误报的概率，而且也可能会影响性能。目前在OceanBase中的探活周期为10秒左右，确保能及时感知到节点故障，并且也不会频繁产生误报。

2）其次，要能够容忍偶发性的消息丢包，减小误报的概率。

具体来说，OceanBase不会由于一次探活的失败就认定某个节点发生了故障，而是在连续多次尝试都失败后才确认真正的节点故障。这就有效避免了偶发性消息丢包所导致的误报。

3）如果真的发生了误报，需要将影响范围降到最小。

OceanBase将Paxos组的粒度下沉到了表的分区一级，也就是每一个分区都会有一个Paxos组，用来维护这个分区的多个副本之间的leader-follower关系。如果由于少量网络丢包导致“某个分区”的探活消息没有收到回复，那么受影响的只是这个分区，同一台机器上的其它分区会照常工作，这就有效地控制了问题的影响范围。

4）某些特殊情况的处理。

举个例子，如果某台机器出现间歇性故障（比如网卡或者操作系统出了问题），导致这台机器频繁发生网络传输故障，就会使这台机器上所有的leader副本持续受到影响。这种情况下，OceanBase可以通过设置特定参数限制这台机器暂时不参与leader选举，这样就有效地起到了隔离作用，避免了局部故障对整个集群的可用性造成持续影响。

在具备了上述这些处理机制后，OceanBase目前已经能做到最多10秒钟检测到服务节点异常，并在10~30秒内完成服务的自动恢复。需要说明的是，具体的恢复时间和遇到问题的机器个数、表的分区个数、故障类型（机器硬件故障、网络设备故障等）都有密切的关系，所以上面说的服务恢复时间只是作为一个参考值，在某些特殊情况下也可能发生偏差。

Part 4 OceanBase的容灾方案

前面说到的内容，更多的还是从逻辑架构上介绍OceanBase如何实现数据可靠性和服务高可用。但实际应用中，除了逻辑架构之外，还必须考虑到系统部署时的物理分布情况，否则就无法充分利用Paxos协议所带来的优势。这就衍生出了机架容灾、同城机房容灾、异地城市容灾等诸多和容灾相关的概念。

之前在OceanBase微信公众号中已经介绍了常用的容灾方案，在这里就不再赘述了，有兴趣的朋友可以参考公众号里的文章：

- 一文详解OceanBase的高可用及容灾方案（上篇）
- 一文详解OceanBase的高可用及容灾方案（下篇）

Part 5 总结

到目前为止，我们已经介绍了OceanBase中关于保证数据可靠性和服务高可用的一些基本原理。原理看上去简单，但由于分布式数据库的网络复杂性以及PC硬件的不可靠，在实际使用中会发生形形色色的各种异常情况，任何一个细节处理不好都会造成严重后果。

经过在蚂蚁金服的大大小小、成百上千个系统中多年的打磨和锤炼后，OceanBase已经逐渐完善了众多的处理细节，形成了有效的机制来保证数据可靠性和服务可用性。而且，这套机制已经在众多实际系统中得到了很好的验证，尤其是支付宝和网商银行这种“**线上数据就是全部，一个字节也不能丢**”的金融级核心系统。

因此，今天OceanBase可以说已经突破了传统单点数据库在数据可靠性和服务高可用方面的限制，让用户的数据更安全，服务更稳定！

往期推荐

- 现场剪光缆！支付宝模拟自断一半服务器，26秒一切恢复正常。这一切的背后基于……
- 传统关系数据库高可用的缺失
- 大师专栏 | 如何在「不可靠」硬件上实现金融级高可用？