# 背景

云原生数据库在海外被称为「Cloud-Native Database Systems」，前面译成中文就是「云原生」，理解云原生先要理解两个关键词——「云」和「原生」。

在传统数据库的系统架构下，必须是紧耦合的设计方式，才能最大效能地发挥系统的优势。举个生活中的例子，过去每家每户会根据用水量打一口水井使用，这与传统的数据库系统使用计算、存储资源的方式一样，但它是紧耦合的方式。如果水不够了怎么办？对应的，这就是传统数据库系统里经常提到的业务扩容。在金融行业，数据库系统扩容通常需要提前几个月甚至半年去做规划，进而细致部署、缜密实施，上线——灰度——再验证，一整套流程就是为了扩容、缩容。传统扩容过程非常漫长，而业务高峰过后缩容也很痛苦，往往会造成极大的资源浪费，也很难应对业务层需要的快速变化能力，这是传统架构非常大的弊端之一。

如果清楚这些概念和背景就很好理解云原生了。「云」就是使用虚拟化的技术将资源池化。水是资源，不用紧耦合的方式来部署和使用，不用家家户户打水井，而是整个村庄联合起来“打”一个湖或池塘，再修管道连到各家各户，这就是资源池化。资源池化以后可以做到按需按量使用，弹性调度，甚至还可以将资源进行解耦，比如，将厨房当作一个计算节点，水当作存储节点，可以随时切换，如果这个厨房不够了，通过系统调度，可以快速拉起另外两三个厨房做计算。这就是云原生核心逻辑，将不同类型资源解耦，并进行池化。让原来的一口口独立水井，在背后合并成一个看不见的江河湖海。具体的，比如在云原生的计算存储分离架构下，业务节点可以根据需要自由地对计算、存储进行快速的扩缩容等操作。

可以看到，云原生带来的本质性变化就像水井和池塘，随着池塘不断变大，越来越多应用迁移上云，池塘不再是池塘，而变成了江河大海。云原生带来的最大技术红利以及经济红利就是规模化应用后带来边际成本下降效应，因此向云原生技术演进的趋势自然发生并且非常清晰，无论用公共云还是专有云私有化部署。这个边际成本下降效应体现在产品上，客户就会因此受益，TCO也一定会下降。

# 概述

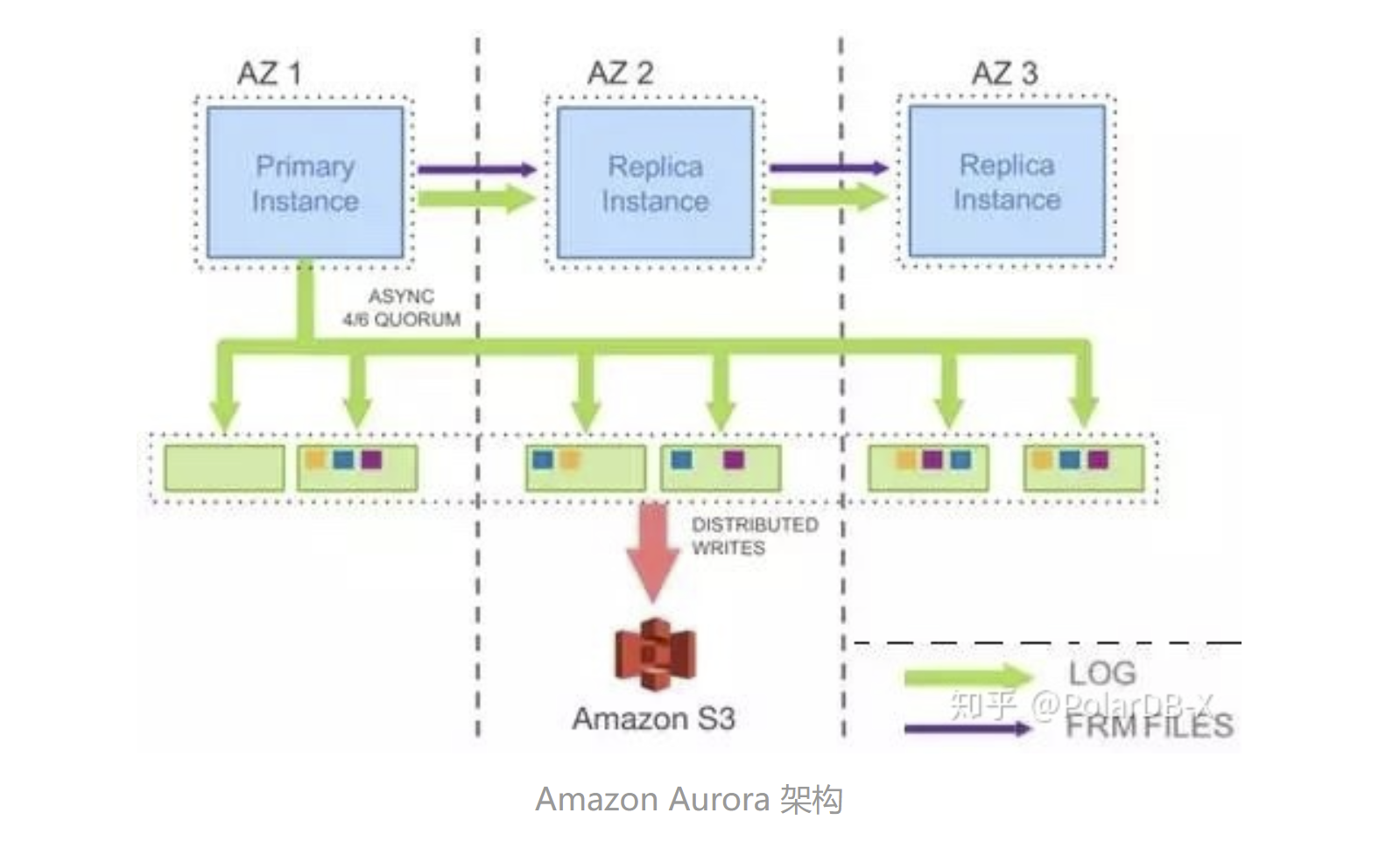
参考：<http://www.uml.org.cn/sjjm/202112171.asp>

下一代云原生数据库详解：<http://mysql.taobao.org/monthly/2020/05/01/>

MySQL中的IO共享操作：<http://mysql.taobao.org/monthly/2019/01/09/>

## 云原生数据库

在放眼未来之前，我们先看看云数据库近几年的现状。AWS作为云计算的开拓者，并未止步于直接售卖开源版本的MySQL（也就是RDS），而是在MySQL基础上研发了Aurora，硬是把MySQL改造成了一个横跨3个可用区、拥有6份replica的高可用分布式数据库。



Aurora 的设计理念和许多AWS产品一脉相承——基于开源项目，用云的方式去改造，保持最大的兼容性。Aurora站在MySQL这个巨人的肩膀上，通过改造IO path提升性能的同时又做到高可用，可以说是非常精巧的设计。

有了先驱者探路，其他云厂商也纷纷跟进。腾讯云的 CynosDB 以及华为的 TaurusDB直接照搬了这套设计。阿里云PolarDB则略有一些区别，PolarDB同样是改造自 MySQL，不过动手术的地方更底层一些，简单来说是将文件系统层替换成了分布式存储。这种架构常被称为shared-storage（共享存储），意为将本地盘存储换成了分布式共享存储。

PolarDB 除了共享存储以外还做了许多其他改进，例如锁管理器、buffer pool 等，官方有时会将其称为shared-everything架构。

共享存储架构很好的解决了数据库存储容量的scale out，几乎可以无限扩展。甚至更进一步，云厂商可以将磁盘资源池化，实现按实际使用量付费（pay-as-you-go），大大降低用户使用成本。但该架构也存在明显的瓶颈：尽管存储方面全面拥抱了分布式，但MySQL Server这一层还保留着单机数据库的一切，尤其是并发事务处理能力（写入吞吐量），受单个节点的性能上限制约。好在对于大多数用户来说这已经足够了。

## NewSQL的崛起

云计算曾经的另一个流派——Google，时常走在Amazon的对立面。在云数据库这个主题上，Google拿出的解决方案是Spanner，它2012年的这篇论文可谓是惊艳了众人，在scale out和高可用性上都无可挑剔，尤其是依靠原子钟实现全球级的外部一致性。

Spanner支撑了Google内部巨无霸级的广告业务，但是作为一款云数据库，在商业上几乎是失败的。与开源生态不兼容就意味着vendor lock-in（厂商绑定），而这是企业CTO们不愿承担的。

大约从2014年开始出现了一些Spanner的开源模仿者，其中的代表包括CockroachDB以及国人研发的TiDB。CockroachDB的设计理念与Spanner如出一辙，不过把原子钟换成了更亲民的NTP协议，并设计了精巧的协议来保证事务一致性；TiDB则干脆放弃了全球部署，仅保留了最关键的scale out特性。

NewSQL系统的架构大多可以分为两层：底层的分布式存储层和上层的SQL计算层。分布式存储层可以类比成HBase，往往通过Paxos之类的一致性协议保证高可用；SQL计算层负责解析用户的查询，然后从存储层访问相应的数据。我们常将NewSQL的这一套架构称为shared-nothing（无共享），意为每个节点都是独立的进程，不存在任何共享资源。

这样的架构中，无论计算层还是存储层都很容易scale out，只需添加更多的节点即可。对于无状态的计算层，依靠容器技术，不难做到秒级启动新节点。而存储层的scale out则略显麻烦一些，新节点需要从原来的节点中复制数据，等数据达到同步后即可对外提供服务。

**NewSQL是云数据库的终态吗？**

我们很高兴的看到，近几年NewSQL的开源社区一直在蓬勃发展，NewSQL这一概念也为更多开发者所接受。但俗话说“没有银弹”，在云数据库这个战场上，NewSQL仍有明显的缺点：

其一是成本偏高。我们知道RPC的代价是远高于进程内调用的，这种低耦合的架构中，每次查询往往包含数次内部RPC，导致单个查询消耗的资源大幅上升。另一方面，为了实现数据存储的高可用，需要将每份数据保持至少3份拷贝，这又进一步抬高了单位成本。

其二是存储层的弹性不足。新加入的存储节点需要从原有的节点上同步大量数据，这一过程必然会耗费较长的时间，而且会挤占现有存储节点的IO带宽。因此实际生产中往往还是需要预先规划好容量，做不到Aurora那样极致的弹性伸缩。

这两点恰好也是“云原生”（cloud-native）的优势所在。云原生的分布式存储本身已经解决了高可用和弹性伸缩问题，而且作为基础设施组件，云厂商通常都在软件、硬件层面做了大量优化，再加上云计算的规模效应，单位性能的成本往往都要比自建划算的多。

# 分类

## Aurora

## PolarDB

## CynosDB

# 特点

## 优点

## 缺点

# 发展方向

全球知名咨询公司Gartner指出，云将主导数据库市场的未来，到2022年，75%的数据库将被部署或迁移至云平台，只有25%的数据库会在本地运行。云化无疑代表了未来，企业如何在云原生架构下使用数据库，就成为必须要思考的问题的。

随着企业业务全面向数字化、在线化、智能化演进，企业面临着呈指数级递增的海量存储需求和挑战，业务有更多的热点和突发流量带来的挑战，企业需要降本增效，进行更智能的数据决策，传统的商业数据库已经难以满足和响应快速增长的业务诉求。

在架构创新上，我们将云原生与分布式结合起来，全新的云原生分布式架构的数据库具备了高扩展性、易用性、迭代快速、成本降低等特点，可以很好的帮助企业解决上述问题。未来数据库也将全面进入云原生加分布式的时代。具体来讲：

1、高扩展性

云原生分布式数据库与底层的云计算基础设施分离，所以能够灵活及时调动资源进行扩容缩容，以从容应对流量激增带来的压力，以及流量低谷期因资源过剩造成的浪费。生态兼容的特点，也让云原生数据库具备很强的可迁移性。

2、易用性

云原生分布式数据库非常易于使用，它的计算节点在云端部署，可以随时随地从多前端访问。因其集群部署在云上，通过自动化的容灾与高可用能力，单点失败对服务的影响非常小。当需要升级或更换服务时，还可以对节点进行不中断服务的轮转升级。

3、快速迭代

云原生分布式数据库中的各项服务之间相互独立，个别服务的更新不会对其他部分产生影响。此外，云原生的研发测试和运维工具高度自动化，也就可以实现更加敏捷的更新与迭代。

4、节约成本

建立数据中心是一项独立而完备的工程，需要大量的硬件投资以及管理和维护数据中心的专业运维人员。此外，持续运维会造成很大的财务压力。云原生分布式数据库以较低的前期成本，获得一个可扩展的数据库，实现更优化的资源分配。