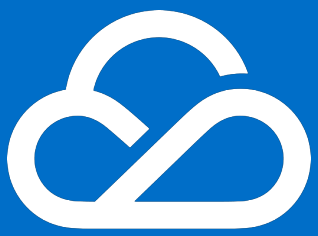
腾讯云分布式数据库DCDB

基本原理

产品文档



腾讯云

文档目录

基本原理

水平分表

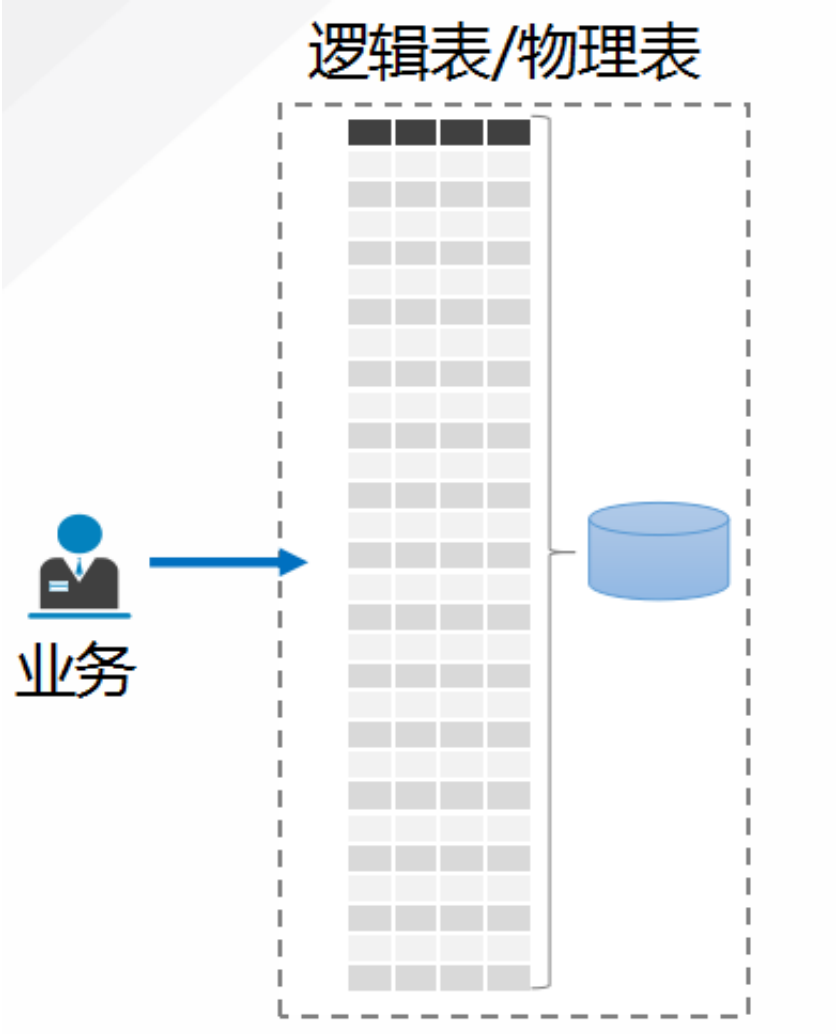
概述

水平拆分的方案，实际上是分布式数据库的基础原理，他的每个节点都参与计算和数据存储，而且每个节点都仅计算和存储一部分数据。因此，无论业务的规模如何增长，我们仅需要在分布式集群中不断的添加设备，用新设备去应对增长的计算和存储需要即可。

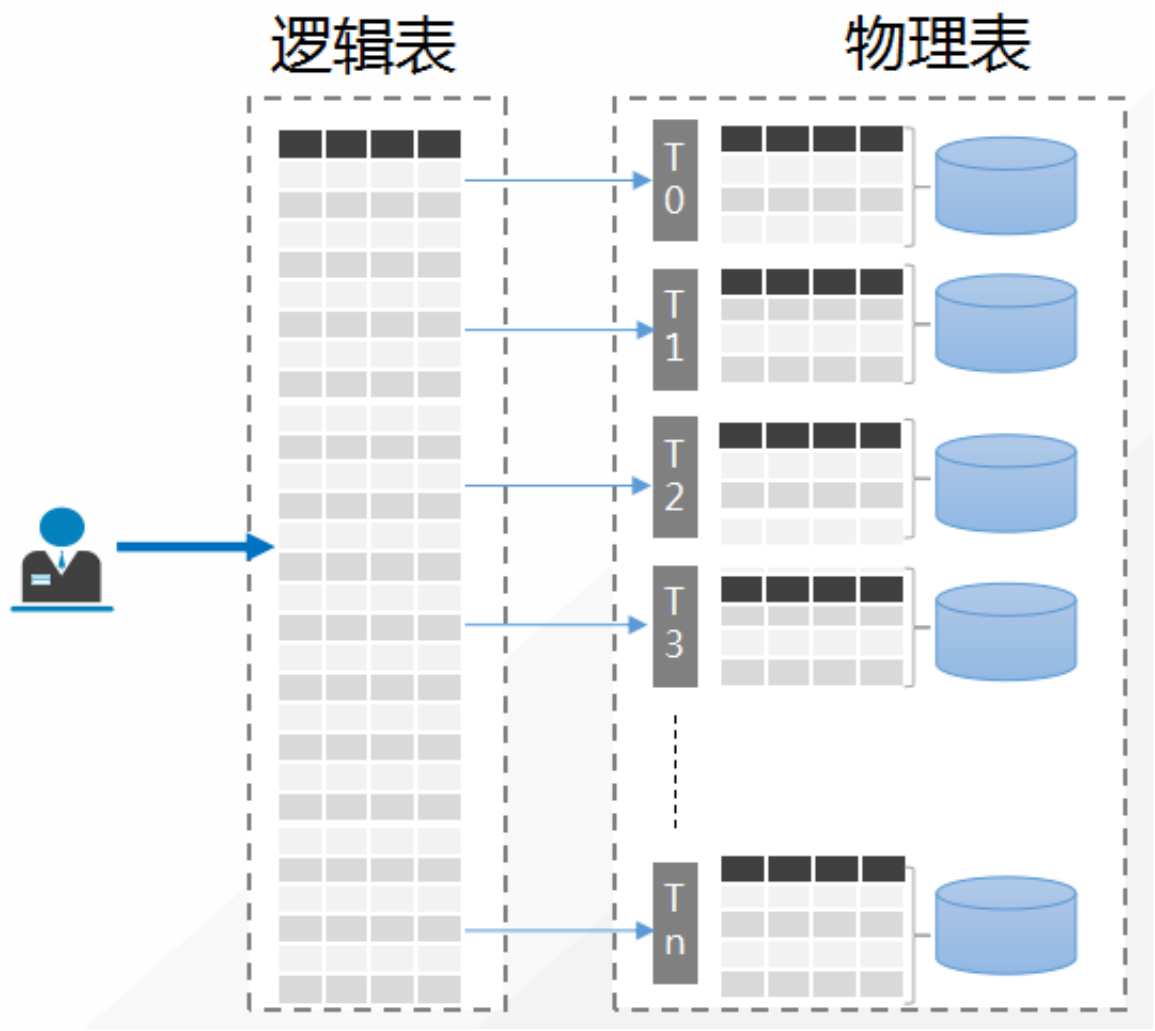
水平切分

水平切分（分表）
是按照某种规则，将一个表的数据分散到多个物理独立的数据库服务器中，形成“独立”的数据库“分片”。多个分片共同组成一个逻辑完整的数据库实例。

- 常规的单机数据库中，一张完整的表仅在一个物理存储设备上读写。



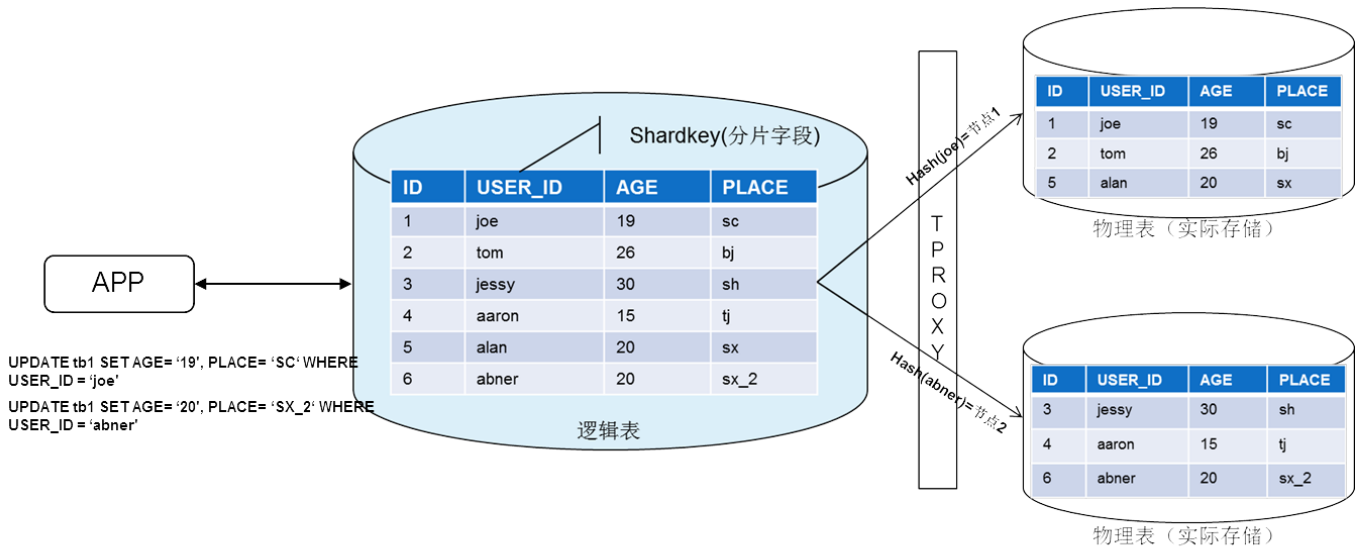
- 分布式数据库，根据在建表时设定的分表键，系统将根据不同分表键自动分布到不同的物理分片中，但逻辑上仍然是一张完整的表。



- 在 DCDB 中，数据的切分通常就需要找到一个分表键（shardkey）以确定拆分维度，再采用某个字段求模（HASH）的方案进行分表，而计算 HASH 的某个字段就就是 shardkey。 HASH 算法能够基本保证数据相对均匀的分散在不同的物理设备中。

写入数据时（ SQL 语句含有 shardkey ）：

1. 业务写入一行数据。
2. 网关通过对 shardkey 进行 hash 。
3. 不同的 hash 值范围对应不同的分片（调度系统预先分片的算法决定）。
4. 数据根据分片算法，将数据存入实际对应的分片中。



数据聚合

数据聚合：如果一个查询 SQL 语句的数据涉及到多个分表，此时 SQL 会被路由到多个分表执行，DCDB 会将各个分表返回的数据按照原始 SQL 语义进行合并，并将最终结果返回给用户。

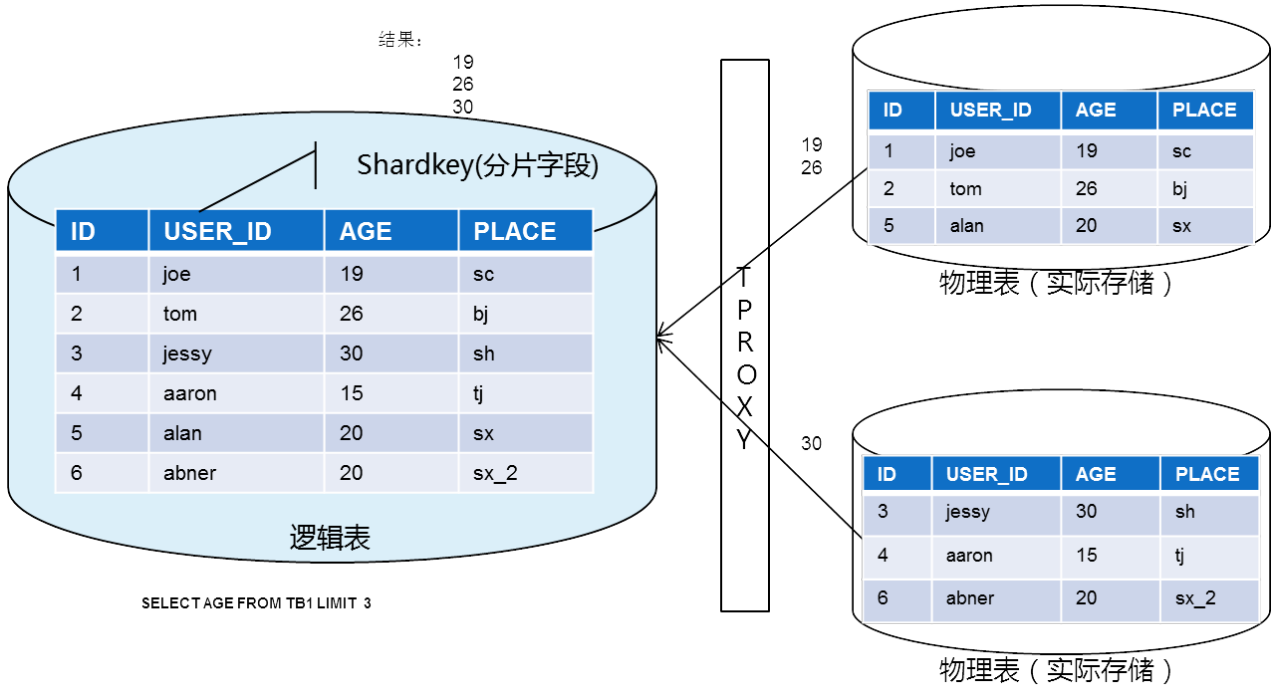
> 注意：
> 在执行 SELECT 语句时，建议您带上 shardkey 字段，否则会导致数据需要全表扫描然后网关才对执行结果进行聚合。全表扫描响应较慢，对性能影响很大。

读取数据时（有明确 shardkey 值）：

1. 业务发送 select 请求中含有 shardkey 时，网关通过对 shardkey 进行 hash。
2. 不同的 hash 值范围对应不同的分片。
3. 数据根据分片算法，将数据从对应的分片中取出。

读取数据时（无明确 shardkey 值）：

1. 业务发送 select 请求没有 shardkey 时，将请求发往所有分片。
2. 各个分片查询自身内容，发回 Proxy 。
3. Proxy 根据 SQL 规则，对数据进行聚合，再答复给网关。



读写分离

注意：
预计8月底支持自助申请，如提前需要，可提交工单。

1. 概述

1.1 功能简介

当处理大数据量“读请求”的压力大、要求高时，可以通过读写分离功能将读的压力分布到各个从节点上。腾讯研发的 DCDB 默认支持读写分离功能，架构中的每个从机都能支持只读能力，如果配置有多个从机，将由网关集群（TProxy）自动分配到低负载从机上，以支撑大型应用程序的读取流量；

1.2 基本原理

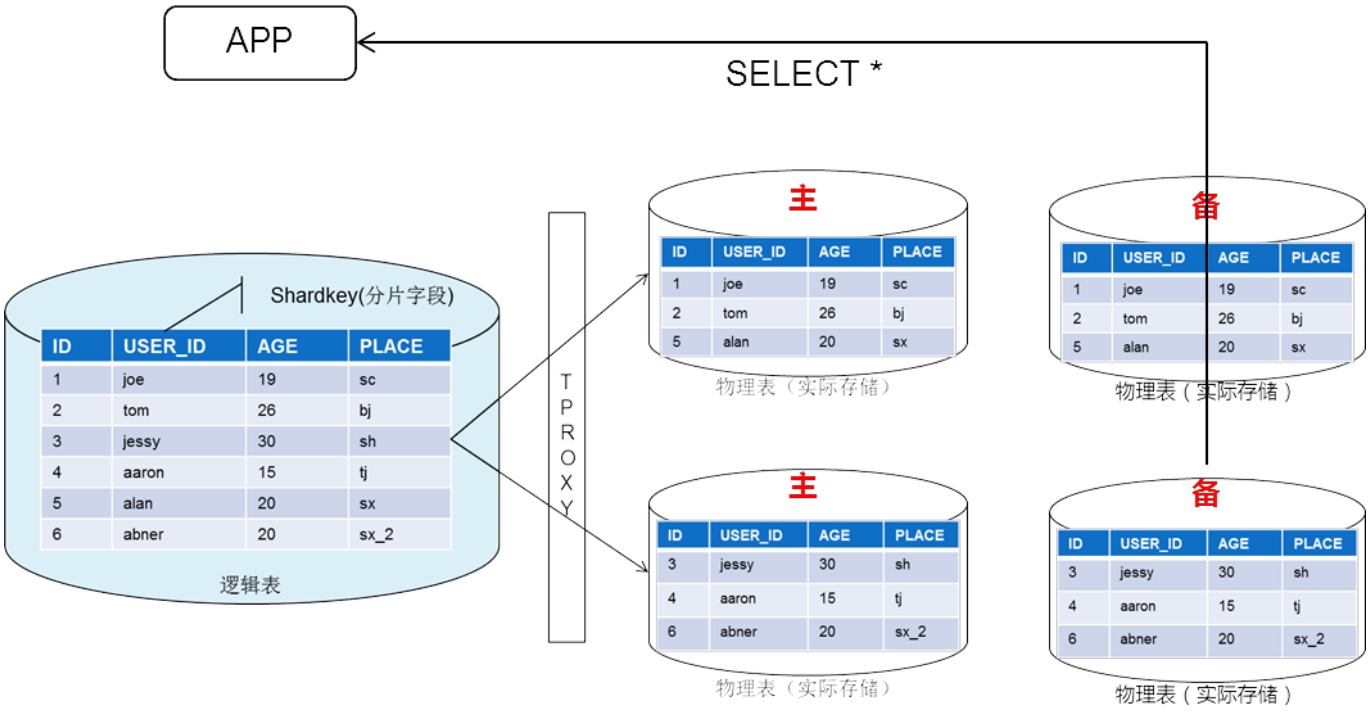
读写分离 基本的原理是让主节点 (master)
处理事务性增、改、删操作（INSERT、UPDATE、DELETE），让从节点 (slave) 处理查询操作（SELECT）。

2. 使用读写分离

2.1基于只读帐号的读写分离

只读帐号是一类仅有读权限的账户，默认从数据库集群中的从机（或只读实例）中读取数据。

通过只读帐号，对读请求自动发送到备机，并返回结果。



2.1.1 读写分离策略

在只读帐号设置选项中，您可以设置【只读请求分配策略】，定义在备机故障（或延迟较大）时的“读“策略。

- 选择【主机】则备机延迟超时时从主机读取。
- 选择【直接报错】则备机延迟超时时报错。
- 选择【只从备机读取】则忽略延迟参数，一直从备机读取（一般用于拉取 binlog 同步）。
- 定义【只读备机延迟参数】，定义数据同步延迟时间，并与【只读请求分配策略】中的【主机】及【直接报错】两种策略配合使用。

2.2 基于注释的读写分离

在每条需要从机“读”的 SQL 前，增加

/*slave*/

字段，并且 mysql 后面增加 -c 参数来解析注释

mysql -c -e "/*slave*/sql"

，即可自动将“读”请求分配到从机，代码示例如下：

//主机读//
select * from emp order by sal，deptno desc；
//从机读//
/*slave*/ select * from emp order by sal，deptno desc；

注意：

1. 该功能仅支持从机读（select），不支持其他操作，非 select 语句将失败。
2. mysql 后面要增加 -c 参数来解析注释。
3.

/*slave*/

必须为小写，语句前后无空格。

4. 从机出现异常而影响到 MAR（强同步）机制时，从机读操作将自动切换回主机。

弹性拓展

概述

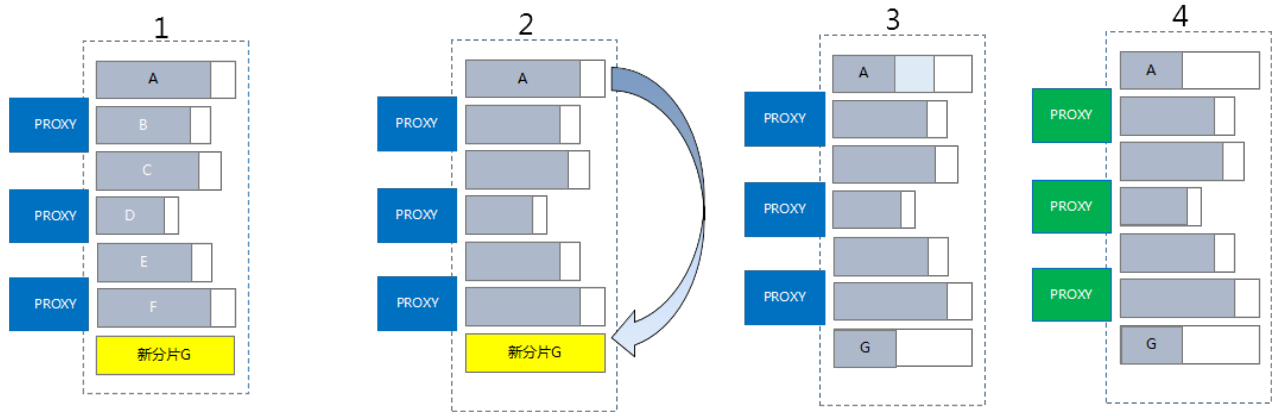
DCDB 支持在线实时扩容，扩容方式分为新增分片和对现有分片扩容两种方式，整个扩容过程对业务完全透明，无需业务停机。扩容时仅部分分片存在秒级的只读（或中断），整个集群不会受影响。

扩容过程

DCDB 主要是采用自研的自动再均衡技术保证自动化的扩容和稳定。

1. 新增分片扩容

1. 控制台点击扩容后，系统根据负载和容量计算出 A 节点（实际上可能影响多个节点）存在瓶颈。
2. 根据新加 G 节点配置，将 A 节点部分数据搬迁（从备机）到 G 节点。
3. 数据完全同步后，AG 节点校验数据库，（存在一至几十秒的只读），但整个服务不会停止。
4. 调度通知 proxy 切换路由。

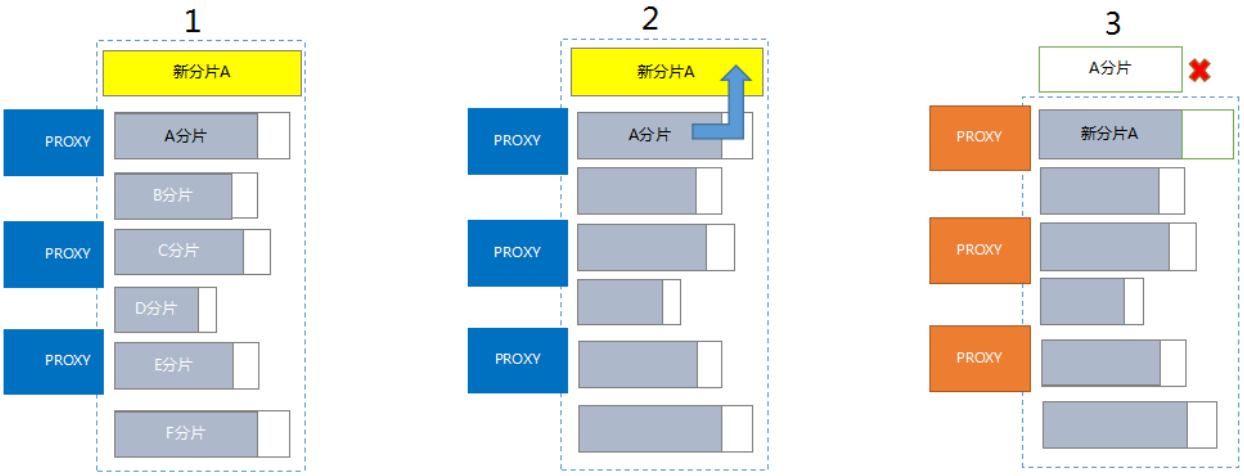


2. 现有分片扩容

基于现有分片的扩容其实相当于更换了一块更大容量的物理分片。

> 注意：
> 基于现有分片的扩容没有增加分片，不会改变划分分片的逻辑规则和分片数量。

1. 按需要升级的配置分配一个新的物理分片（以下简称“新分片”）。
2. 将需要升级的物理分片（以下简称“老分片”）的数据、配置等同步数据到新分片中。
3. 同步数据完成后，在腾讯云网关做路由切换，切换到新分片继续使用。



强同步

概述

MAR 强同步复制方案是腾讯自主研发的基于 MySQL 协议的异步多线程强同步复制方案，只有当备机数据完全同步（日志）后，才由主机给予应用事务应答，保障数据不丢、不错。
数据库作为系统数据存储和服务的核心能力，其可用性要求非常高。在生产系统中，通常都需要用高可用方案来保证系统不间断运行，而数据同步技术是数据库高可用方案的基础。MAR
强同步技术可很好的满足数据库可用性的要求。

传统数据复制方式

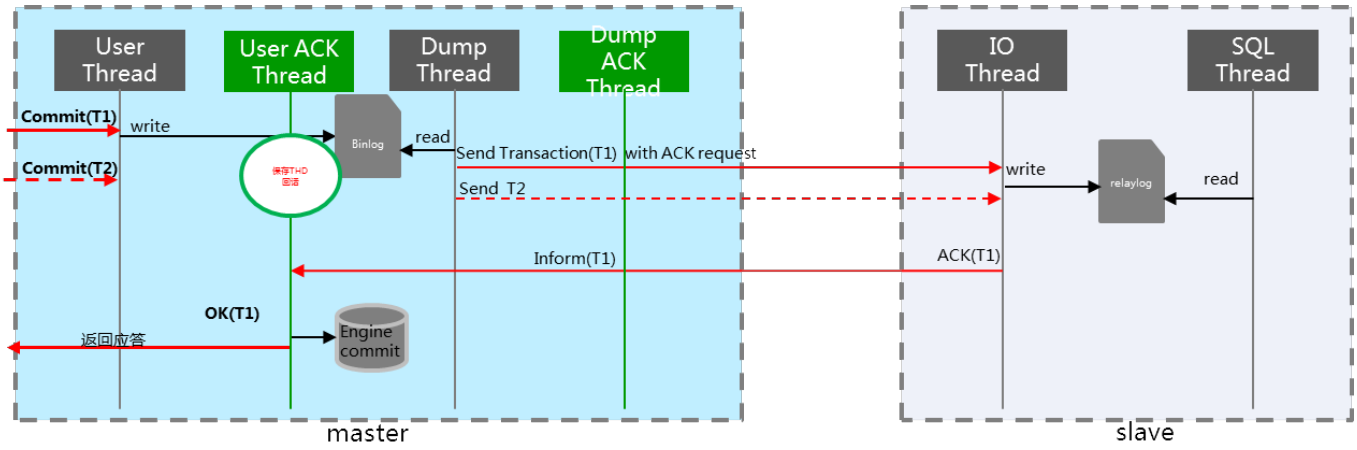
当前，数据复制方式有以下三种方式：

- 异步复制：应用发起更新请求，主节点（Master） 完成相应操作后立即响应应用，Master 向从节点（Slave）异步复制数据。
- 强同步复制：应用发起更新请求，Master 完成操作后向 Slave 复制数据，Slave 接收到数据后向 Master 返回成功信息，Master 接到 Slave 的反馈后再应答给应用。Master 向 Slave 复制数据是同步进行的。
- 半同步复制：正常情况下数据复制方式采用强同步复制方式，当 Master 向 Slave 复制数据出现异常的时候（Slave 不可用或者双节点间的网络异常）退化成异步复制。当异常恢复后，异步复制会恢复成强同步复制。

以上三种方式当 Master 或 Slave 不可用时，均有几率引起数据不一致。

MAR 强同步复制方案

MAR 强同步复制方案能保障数据不丢、 不错，技术示意图如下：



其特点如下：

1. 一致性的同步复制，保证节点间数据强一致性。
2. 对业务层面完全透明，业务层面无需做读写分离或同步强化工作。
3. 将串行同步线程异步化，引入线程池能力，大幅度提高性能。
4. 支持集群架构。
5. 支持自动成员控制，故障节点自动从集群中移除。
6. 支持自动节点加入，无需人工干预。
7. 每个节点都包含完整的数据副本，可以随时切换。
8. 无需共享存储设备。

MAR 强同步方案在性能上优于其他主流同步方案，具体数据详情可参考强同步性能对比数据。