TDSQL MySQL版

产品简介

腾讯云

文档目录

产品简介
产品概述

最近更新时间：2022-10-26 11:39:25

简介

TDSQL MySQL版（TDSQL for MySQL）是部署在腾讯云上的一种支持自动水平拆分、Shared Nothing 架构的分布式数据库。TDSQL MySQL版 即业务获取的是完整的逻辑库表，而后端会将库表均匀的拆分到多个物理分片节点。
TDSQL MySQL版 默认部署主备架构，提供容灾、备份、恢复、监控、迁移等全套解决方案，适用于 TB 或 PB 级的海量数据库场景。
TDSQL MySQL版 提供不同的引擎供用户选择，两者均兼容 MySQL 标准协议：

- InnoDB 版采用 InnoDB 作为数据存储引擎，是 MySQL 的默认存储引擎。
- TDStore 版采用腾讯云自研的新敏态引擎 TDstore 作为数据存储引擎，该引擎可以有效解决客户业务发展过程中业务形态、业务量的不可预知性，适配金融敏态业务。

> **说明**
> 数据库内核及 SQL 引擎的新特性可参考 私有云说明，私有云版本后两位与公有云版本后两位相一致表示相同内核版本。

解决问题

单机数据库瓶颈

面对互联网类业务百万级以上的用户量，单机数据库由于硬件和软件的限制，数据库在数据存储容量、访问容量、容灾等方面都会随着业务的增长而到达瓶颈。
TDSQL MySQL版 目前单分片最大可支持6TB存储，如果性能或容量不足以支撑业务发展时，在控制台自动升级扩容。升级过程中，您无需关心分布式系统内的数据迁移，均衡和路由切换。升级完成后访问 IP 不变，仅在自动切换时存在秒级闪断，您仅需确保有重连机制即可。

应用层分片开发工作量大

应用层分片将业务逻辑和数据库逻辑高度耦合，给当前业务快速迭代带来极大的开发工作量。
基于 TDSQL MySQL版 透明自动拆分的方案，开发者只需要在第一次接入时修改代码，后续迭代无需过多关注数据库逻辑，可以极大减少开发工作量。

开源方案或 NoSQL 难题

选择开源或 NoSQL 产品也能够解决数据库瓶颈，这些产品免费或者费用相对较低，但可能有如下问题：

- 产品 bug 修复取决于社区进度。
- 您的团队是否有能持续维护该产品的人，且不会因为人事变动而影响项目。
- 关联系统是否做好准备。
- 您的业务重心是什么，投入资源来保障开源产品的资源管控和生命周期管理、分布式逻辑、高可用部署和切换、容灾备份、自助运维、疑难排查等是否是您的业务指标。

TDSQL MySQL版 支持 Web 控制台，提供完善的数据备份、容灾、一键升级等功能，完善的监控和报警体系，大部分故障都通过自动化程序处理恢复。

产品优势

最近更新时间：2023-01-12 16:04:16

超高性能

- 单分片最大性能可达超24万 QPS，整个实例性能随着分片数量增加线性扩展。
- 计算/存储资源均可独立对业务全透明地弹性扩缩容，单实例支持 EB 级海量存储。
- 计算层每个节点均可读写，单实例轻松支撑千万级 QPS 流量。
- 支持超高压缩比存储，最高可达20倍压缩率，尤其适合大批量写入、写多读少的业务场景。

专业可靠

- 经过腾讯各类核心业务10余年大规模产品的验证，包括社交、电商、支付、音视频等。
- 提供完善的数据备份、容灾、一键升级等功能。
- 完善的监控和报警体系，大部分故障都通过自动化程序处理恢复。
- 提供数据加密能力，支持 AES 算法和国密 SM4 算法，可满足静态数据加密的合规性要求。
- 支持分布式数据库领域领先功能，如分布式多表 JOIN、小表广播、分布式事务、SQL 透传等。
- 数据库实例可用性可达到99.95%；数据的可靠性可达到99.99999%。

简单易用

- 除少量语法与原生 MySQL、MariaDB 不同外，使用起来如使用单机数据库，分片过程对业务透明且无需干预。
- 兼容 MySQL 协议（支持 MySQL、MariaDB 等内核）。
- 支持 Web 控制台，读写分离能力、专有运维管理指令等。

应用场景

最近更新时间：2021-12-20 16:57:00

> **说明**
> TDSQL MySQL版 目前仅适用于 OLTP 场景的业务，例如，交易系统、前台系统；不适用于 ERP、BI 等存在大量 OLAP 业务的系统。

大型应用（超高并发实时交易场景）

电商、金融、O2O、社交应用、零售、SaaS 服务提供商，普遍存在用户基数大（百万级或以上）、营销活动频繁、核心交易系统数据库响应日益变慢的问题，制约业务发展。
TDSQL MySQL版 提供线性水平扩展能力，能够实时提升数据库处理能力，提高访问效率，峰值 QPS 达1500万+，轻松应对高并发的实时交易场景。微信支付、财付通、腾讯充值等都是使用的 TDSQL MySQL版 架构的数据库。

物联网数据（PB 级数据存储访问场景）

在工业监控和远程控制、智慧城市的延展、智能家居、车联网等物联网场景下，传感监控设备多、采样率高、数据规模大。通常存储一年的数据就可以达到 PB 级甚至 EB，而传统基于 x86 服务器架构和开源数据库的方案根本无法存储和使用如此大的数据量。
TDSQL MySQL版 提供的容量水平扩展能力，可以有效的帮助用户以低成本（相对于共享存储方案）存储海量数据。

文件索引（万亿行数据毫秒级存取）

一般来说，作为云服务平台，存在大量的图片、文档、视频数据，数据量都在亿级 – 万亿级，服务平台通常需要将这些文件的索引存入数据库，并在索引层面提供实时的新增、修改、读取、删除操作。
由于服务平台承载着其他客户的访问，服务质量和性能要求极高。传统数据库无法支撑如此规模的访问和使用，TDSQL MySQL版 超高性能和扩展能力并配合强同步能力，有效的保证平台服务质量和数据一致性。

高性价比商业数据库解决方案

大型企业、银行等行业为了支持大规模数据存储和高并发数据库访问，对小型机和高端存储依赖极强。而互联网企业通过低成本 x86 服务器和开源软件即可达到与商业数据库相同甚至更高的能力。
TDSQL MySQL版 适用于诸如国家级或省级业务系统汇聚、大型企业电商和渠道平台、银行的互联网业务和交易系统等场景。

基本原理
水平分表

最近更新时间：2023-02-21 11:41:08

概述

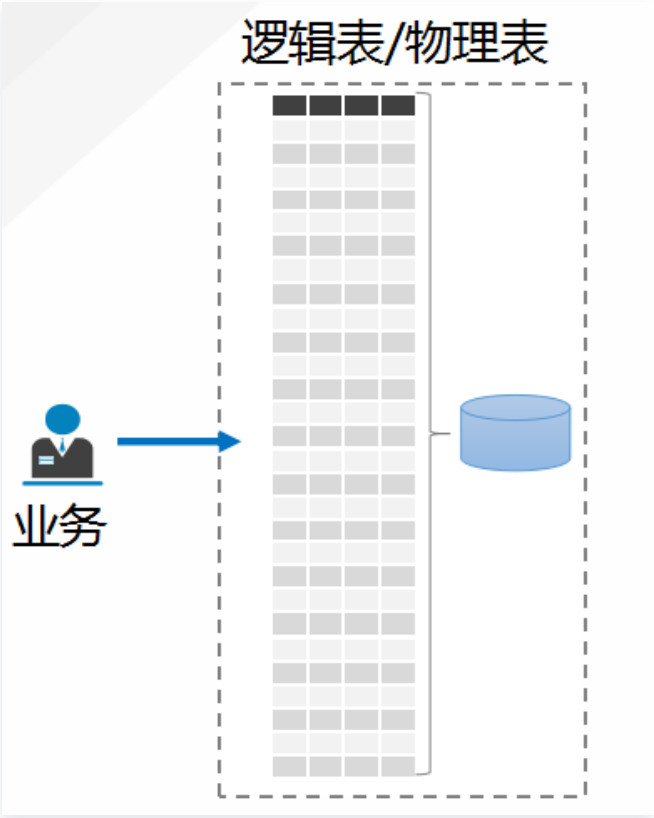
水平拆分方案是 TDSQL MySQL版 的基础原理，它的每个节点都参与计算和数据存储，且每个节点都仅计算和存储一部分数据。因此，无论业务的规模如何增长，我们仅需要在分布式集群中不断的添加设备，用新设备去应对增长的计算和存储需要即可。通过如下视频，您可以了解水平拆分的过程与原理：

观看视频

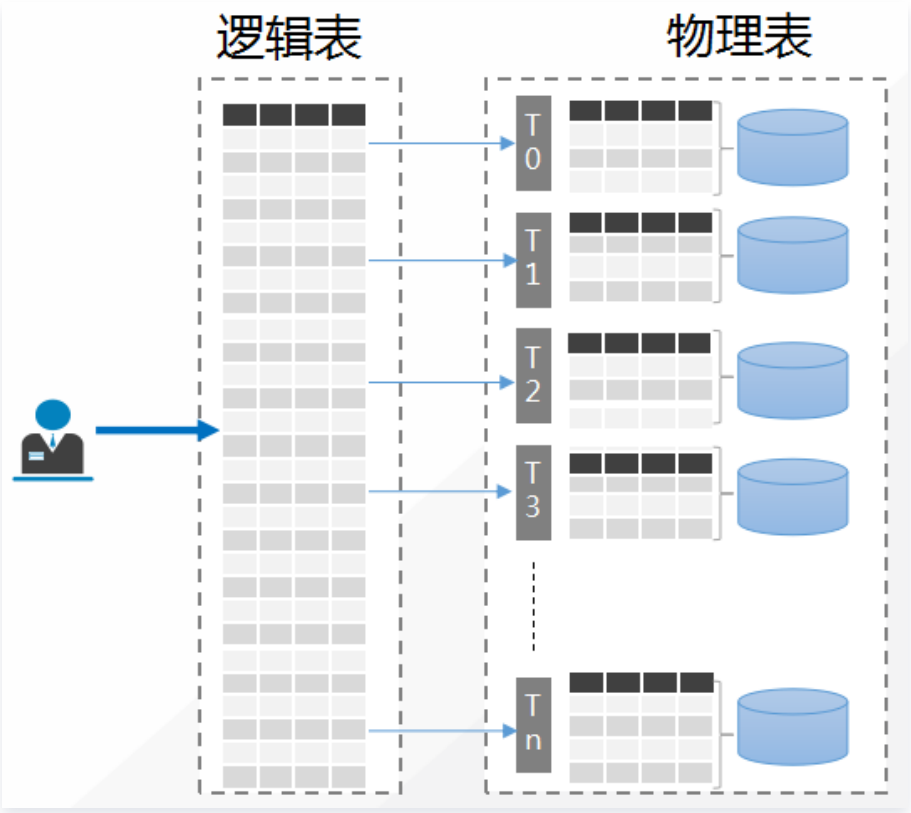
水平切分

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。



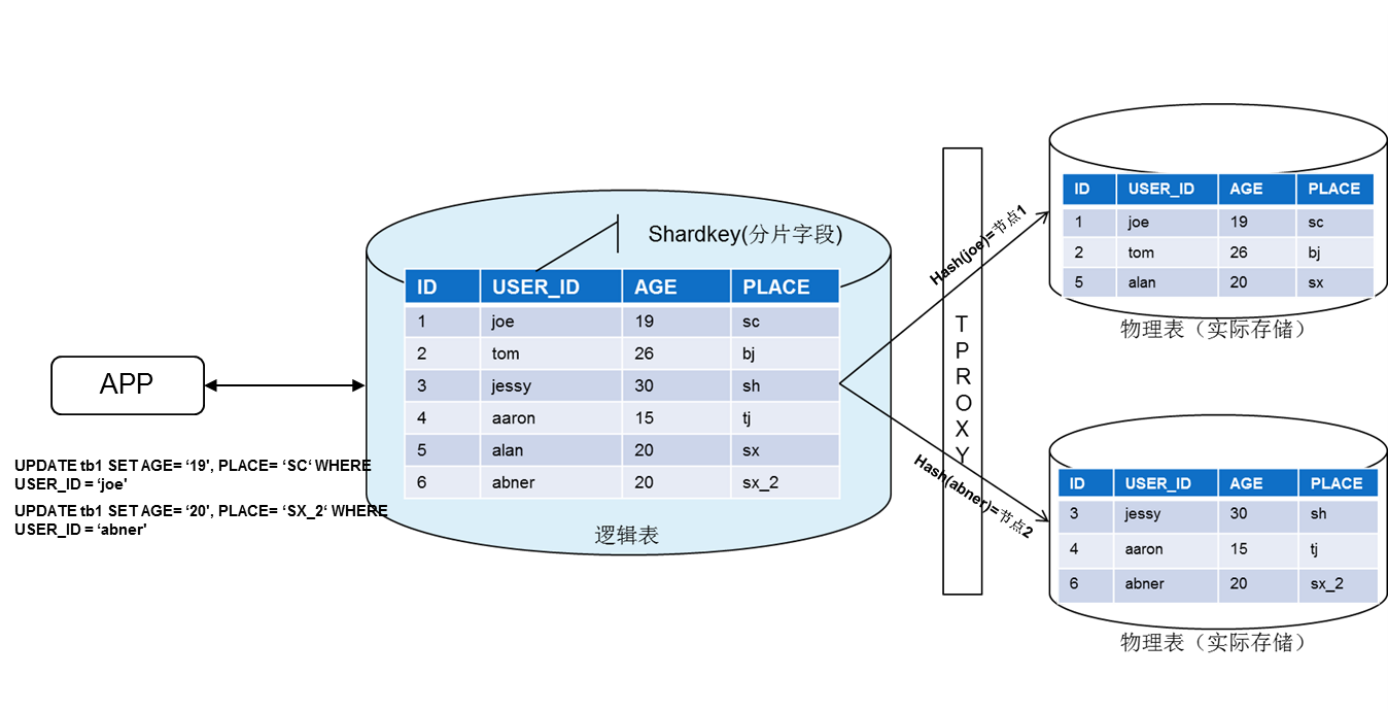
- 分布式数据库中，根据在建表时设定的分表键，系统将根据不同分表键自动分布数据到不同的物理分片中，但逻辑上仍然是一张完整的表。



- 在 TDSQL MySQL版 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 的某个字段就是 shardkey。HASH 算法能够基本保证数据相对均匀地分散在不同的物理设备中。

写入数据（ SQL 语句含有 shardkey ）

1. 业务写入一行数据。
2. 网关对 shardkey 进行 hash，得出 shardkey 的 hash 值。
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，TDSQL MySQL版 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。

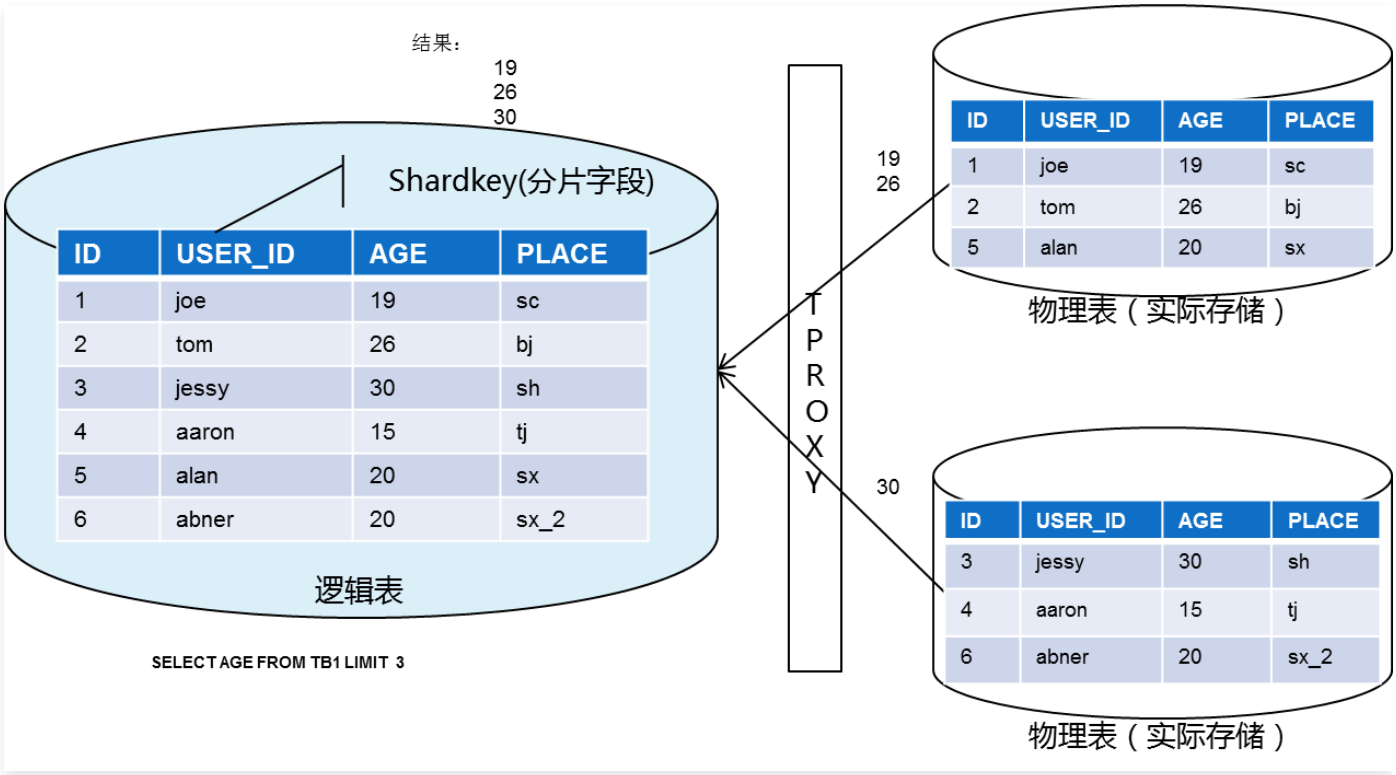
> **注意**
> 执行 SELECT 语句时，建议您在 where 条件带上 shardKey 字段，否则会导致数据需要全表扫描然后网关才对执行结果进行聚合。全表扫描响应较慢，对性能影响很大。

读取数据（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（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。


结果：
19
26
30
Shardkey(分片字段)
ID USER_ID AGE PLACE
1 joe 19 sc
2 tom 26 bj
3 jessy 30 sh
4 aaron 15 tj
5 alan 20 sx
6 abner 20 sx_2
逻辑表
SELECT AGE FROM TB1 LIMIT 3
PROXY
19
26
ID USER_ID AGE PLACE
1 joe 19 sc
2 tom 26 bj
5 alan 20 sx
物理表（实际存储）
30
ID USER_ID AGE PLACE
3 jessy 30 sh
4 aaron 15 tj
6 abner 20 sx_2
物理表（实际存储）

读写分离

最近更新时间：2023-08-09 10:37:05

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。
TDSQL MySQL版 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。

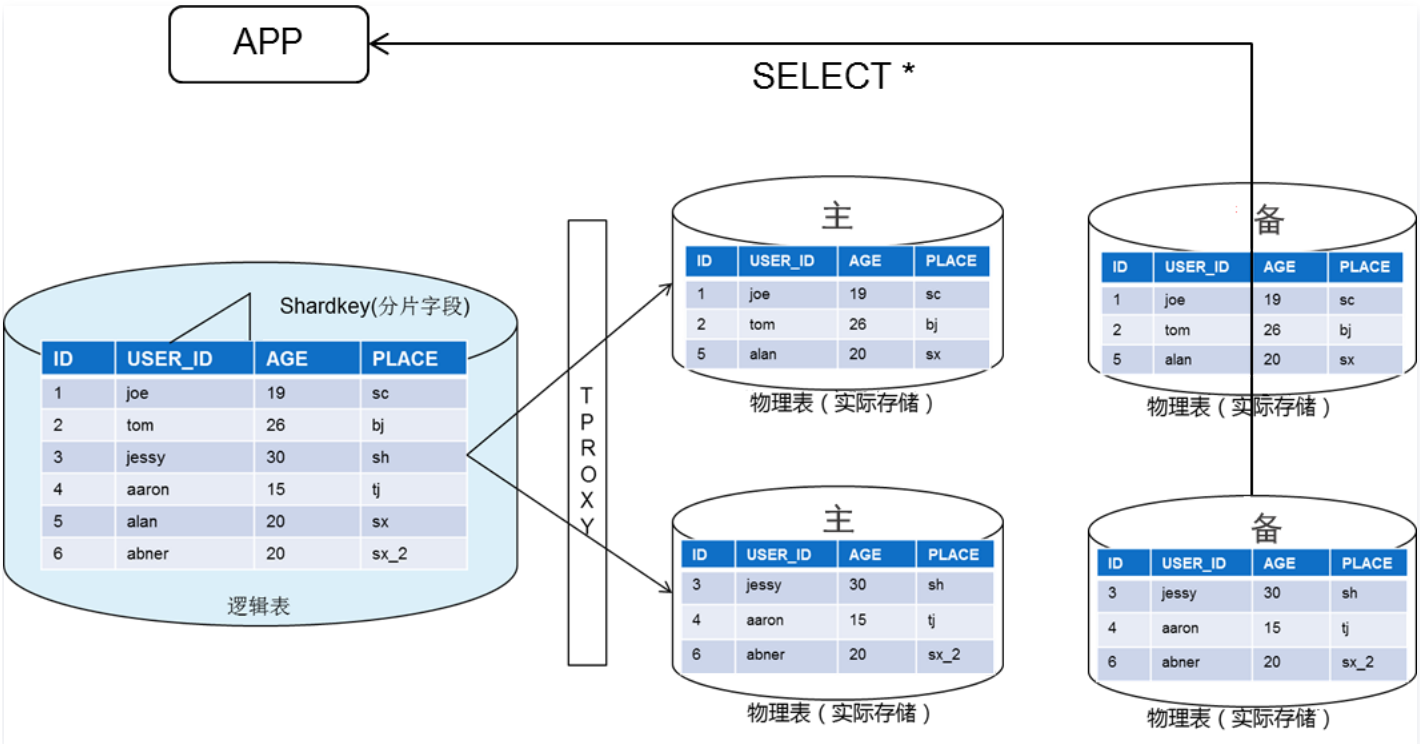
基本原理

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）。

只读账号

> **说明**
> 若您的实例架构为一主一从，只读分离功能仅可用作低负载只读任务，请避免大事务等较高负载任务，影响从机备份任务及可用性。

只读账号是一类仅有读权限的账号，默认从数据库集群中的从机（或只读实例）中读取数据。
通过只读账号，对读请求自动发送到备机，并返回结果。



弹性扩展

最近更新时间：2022-11-24 11:05:54

概述

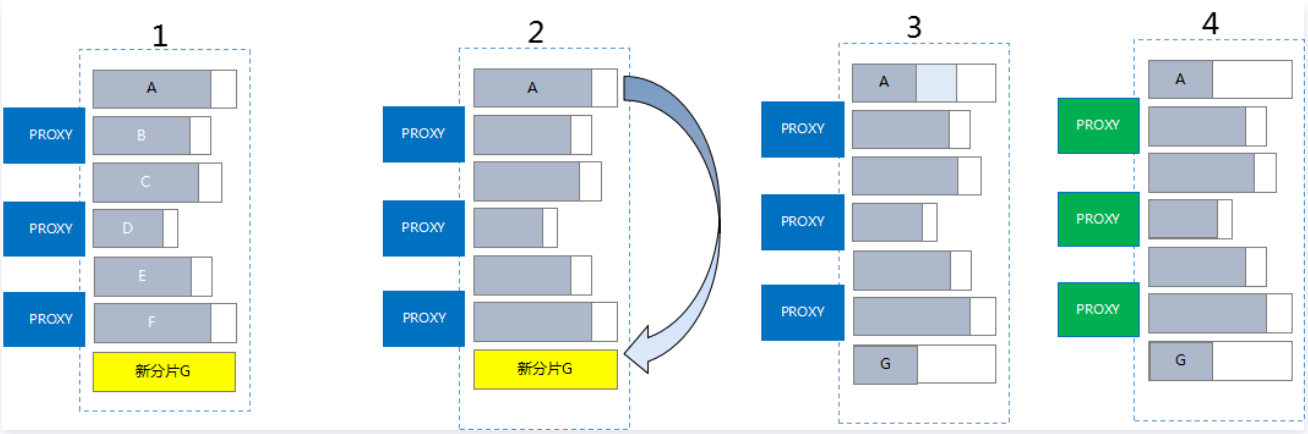
TDSQL MySQL版 支持在线实时扩容，扩容方式分为新增分片和现有分片扩容两种方式，整个扩容过程对业务完全透明，无需业务停机。扩容时仅部分分片存在秒级的只读或中断，整个集群不会受影响。

扩容过程

TDSQL MySQL版 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。

新增分片扩容

1. 在 TDSQL MySQL版控制台 对需要扩容的 A 节点进行扩容操作。
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，A、G 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。

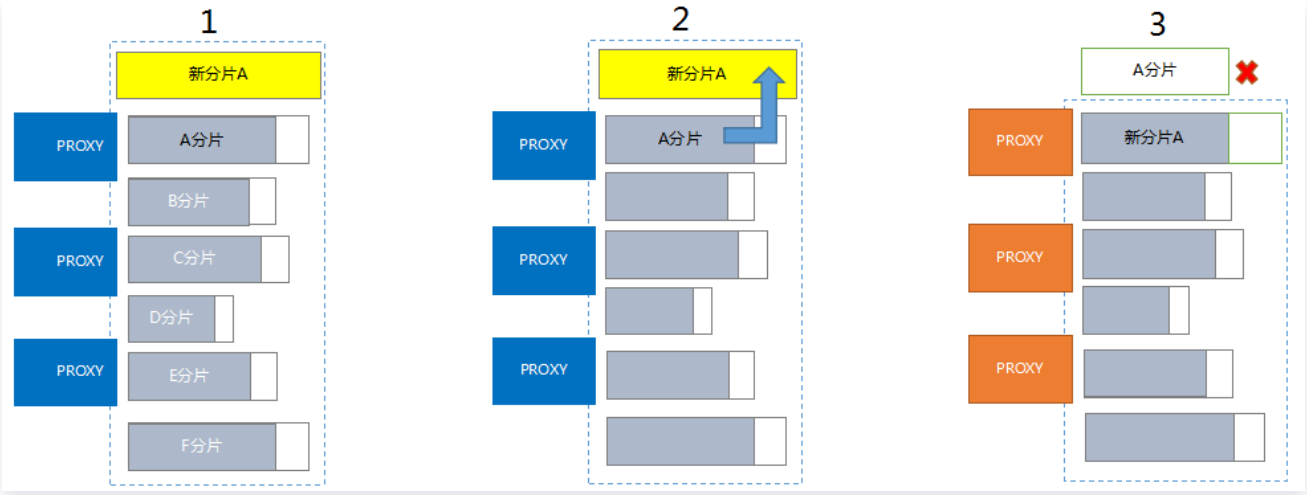


现有分片扩容

基于现有分片的扩容相当于更换了一块更大容量的物理分片。

> **说明**
> 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。



相关操作

分布式数据库由多个分片组成，如您需要将现有 TDSQL MySQL版 实例的规格升级到更高规格，请参见 升级实例 。

强同步

最近更新时间：2022-08-22 11:31:36

背景

传统数据复制方式有如下三种：

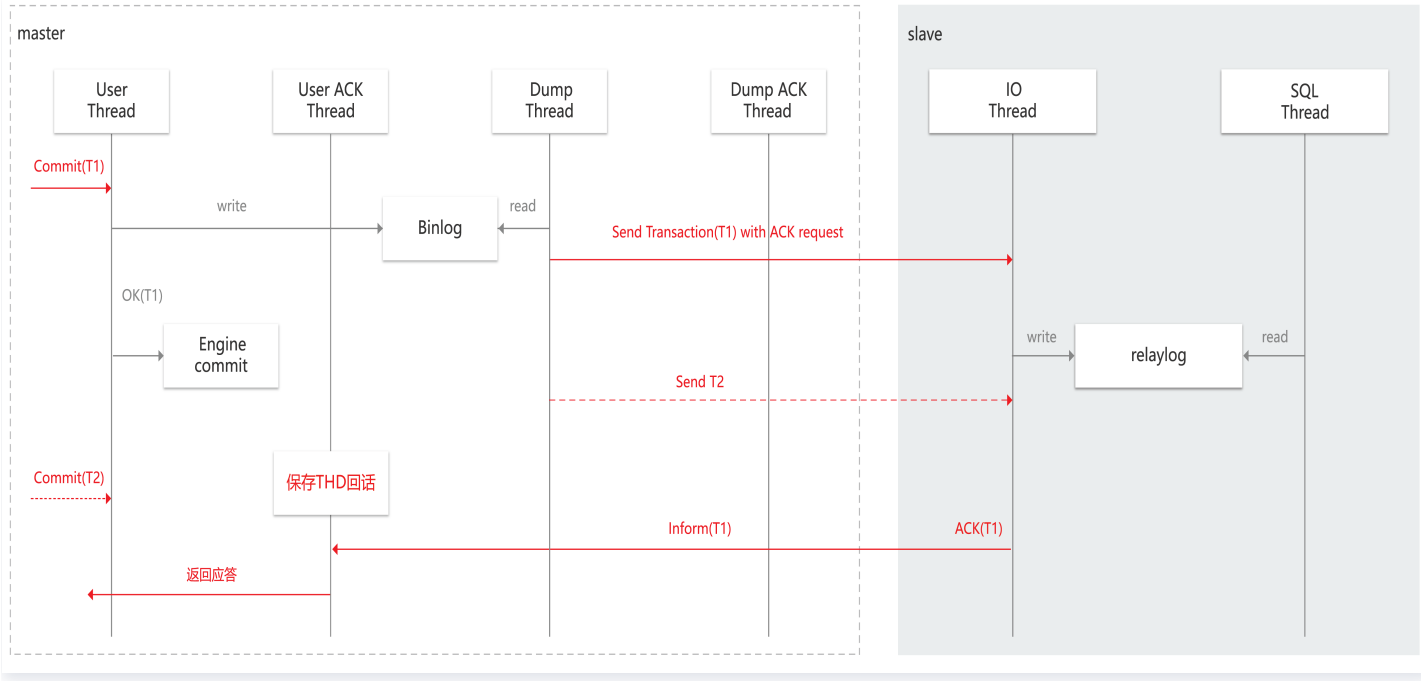
- 异步复制：应用发起更新请求，主节点（Master）完成相应操作后立即响应应用，Master 向从节点（Slave）异步复制数据。
- 强同步复制：应用发起更新请求，Master 完成操作后向 Slave 复制数据，Slave 接收到数据后向 Master 返回成功信息，Master 接到 Slave 的反馈后再应答给应用。Master 向 Slave 复制数据是同步进行的。
- 半同步复制：应用发起更新请求，Master 在执行完更新操作后立即向 Slave 复制数据，Slave 接收到数据并写到 relay log 中（无需执行）后才向 Master 返回成功信息，Master 必须在接受到 Slave 的成功信息后再向应用程序返回响应。

存在问题

当 Master 或 Slave 不可用时，以上三种传统数据复制方式均有几率引起数据不一致。
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。

解决方案

MAR 强同步复制方案是腾讯自主研发的基于 MySQL 协议的并行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，保障数据正确安全。
原理示意图如下：



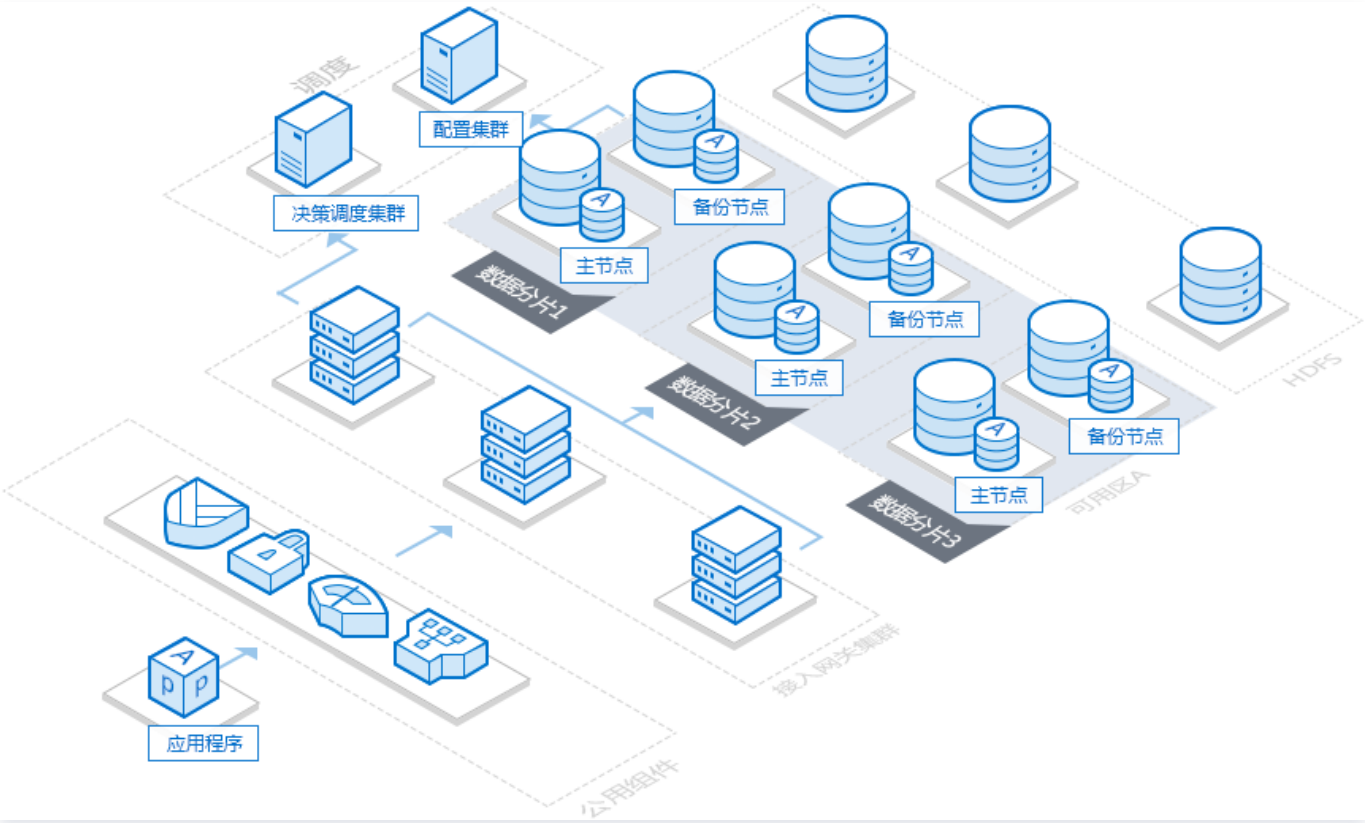
在应用层发起请求时，只有当从节点（Slave）返回信息成功后，主节点（Master）才向应用层应答请求成功，以确保主从节点数据完全一致。
MAR 强同步方案在性能上优于其他主流同步方案，具体数据详情可参见 强同步性能对比数据 。主要特点如下：

- 一致性的同步复制，保证节点间数据强一致性。
- 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。
- 将串行同步线程异步化，引入线程池能力，大幅度提高性能。
- 支持集群架构。
- 支持自动成员控制，故障节点自动从集群中移除。
- 支持自动节点加入，无需人工干预。
- 每个节点都包含完整的数据副本，可以随时切换。
- 无需共享存储设备。

实例架构

最近更新时间：2023-10-07 10:03:01

InnoDB引擎



注意

数据库审计功能重构升级中，敬请期待；在此期间数据库新购实例不再开放审计功能。

实例架构	定义	节点	特点
标准版（一主一从）	每个分片提供主从双活部署的高可用架构	两个节点：一个 Master 节点，一个 Slave 节点	支持从机只读 **说明：** 一主一从架构从机只读仅可用于轻量只读任务，请避免大事务等较高负载任务，影响从机备份任务及可用性。 故障后节点自动恢复 默认监控采样粒度：5分钟/次 最大可配备份时长：7天 操作日志备份：7天 支持数据库审计，审计日志存储15天，规则配置个数暂无限制
标准版（一主二从）	每个分片提供主从多活部署的高可用架构	三个节点：一个 Master 节点，两个 Slave 节点	

金融定制版（一主一从）	每个分片提供主从双活部署的高可用架构	两个节点：一个 Master 节点，一个 Slave 节点	• 支持其他部署方案，需联系对应商务 • 支持从机只读，从机只读时智能负载 • 故障后节点自动恢复 • 默认监控采样粒度：1分钟/次 • 最大可配备份时长：3650天，需 提交工单 • 操作日志备份：默认60天，归档存储1年 • 支持数据库审计：审计日志存储15天 • 监管配合：有
金融定制版（一主二从）	每个分片提供主从多活部署的高可用架构	三个节点：一个 Master 节点，两个 Slave 节点	

TDStore 引擎

- TDStore 实例分为集群版和基础版两种：
  - 集群版：由多个（≥3个）节点构成，以三副本 Raft 集群的形态提供高性能可用的数据库服务，适用于企业生产环境。
  - 基础版：由单个节点构成，以较低的成本提供完整的数据库功能，适用于个人用户。

> **说明**
> 基础版实例创建后可以通过控制台升级为集群版实例；集群版实例创建后不可以降级为基础版实例。

- TDStore 实例内的节点分为对等架构和分离架构两种：
  - 对等架构：计算层 SQL Engine 与数据层 TDStore 合并在一个物理节点中，减少硬件节点数量和跨节点通信，从而降低成本并提高性能。
  - 分离架构：计算层 SQL Engine 与数据层 TDStore 分别在不同的物理节点中。

TDStore 引擎介绍

最近更新时间：2023-02-21 11:42:25

TDStore 是腾讯云全自研的金融级新敏态引擎，该引擎可以有效解决对于客户业务发展过程中业务形态、业务量的不可预知性，适配金融敏态业务。

TDStore 引擎特性

高度兼容 MySQL 语法

TDSQL TDStore 引擎版计算节点基于 MySQL 8.0 实现，除个别受限的系统操作，TDStore 可以100%兼容原生 MySQL 语法。单机 MySQL 的业务可以无损迁移到 TDStore 上，真正实现对业务应用无入侵。

存储计算分离/独立弹性伸缩

TDStore 采用计算和存储分离的原生分布式的架构设计，计算层和存储层的节点均可根据业务需求独立弹性扩缩容，而且无须额外的人工运维干预，实现扩缩容过程对业务零感知。

- **计算层**采用多主架构，而且每个计算节点均可读写，用户可以随着业务量的增长而弹性扩展计算节点，单实例可支撑千万级 QPS 流量，帮助用户轻松应对突如其来的业务峰值压力。
- 对于**存储层**资源，用户也可以随着业务数据量的增长而弹性扩展存储节点。数据在不同节点之间的迁移、均衡、路由变更等操作均由 TDStore 实例内部自洽完成。

云原生的管控系统

TDStore 的管控部分采用了云原生的方式，借助云原生的能力，能够快速且方便地管理 TDStore 实例，免除了繁琐的物理机上架，配置等资源管理运维操作，同时也无需关心资源的使用率情况，即买即用，支持高效弹性扩缩容。

原生 Online DDL 支持

TDStore 支持原生 Online DDL 操作，用户在业务运行过程中，有动态更改表结构的需求时，无须依赖如 pt 或 ghost 等外部工具组件，直接使用原生 MySQL DDL 语句便可完成。
TDStore 覆盖 MySQL 原生可支持的 instant 类型 DDL 操作，并且对于大部分类型（除涉及主键外的）DDL，均能以不阻塞业务的正常 DML 请求下完成。同时，TDStore 的 Online DDL 可以在多个计算节点之间保持一致性，不同表对象的 Online DDL 可以并行执行。

完整分布式事务支持

TDStore 以原生分布式的架构完整支持事务 ACID 特性，默认的事务隔离级别为快照隔离级别（Snapshot Isolation），支持全局一致性读特性，整体事务并发控制框架基于 MVCC + Time-Ordering 的方式实现。
分布式事务协调者由分布式存储层节点担任，而当存储节点在线扩容遇到数据分裂或切主等状态变更的场景时，TDStore 均可实现不中断事务，将底层数据状态的变更对事务请求的影响降到最低，从而做到无感知的集群扩缩容。

低成本海量存储

TDStore 存储层基于 LSM-Tree + SSTable 结构存放和管理数据，具有较高的压缩率，能有效降低海量数据规模下的存储成本。对于一些数据行重复度较大的业务场景，对比 InnoDB 存储引擎，TDStore 版最高可实现高达20倍的压缩率，单实例可支撑 EB 级别的存储量。

TDStore 引擎架构

计算节点 SQLEngine

SQLEngine 是计算节点，负责接收和响应客户端的 SQL 请求。SQLEngine 基于 MySQL 8.0 实现，完全兼容原生 MySQL 语法，从原生 MySQL 迁移过来的业务在使用时无须对业务语句进行任何改造。
SQLEngine 采用无状态化的设计方式，节点本身不保存任何用户数据，并将多线程框架替换为协程框架，与集群内的 TDStore 节点进行交互。
一个实例内可以包含多个 SQLEngine 节点，节点之间彼此独立，均可读写。

存储节点 TDStore

TDStore 是存储节点，负责用户数据的存储。它是一个基于 Multi-Raft 协议实现的分布式存储集群。
Region 是 TDStore 存储和管理数据的最小单位，以及 TDStore 节点之间进行数据复制同步的单位，一个 Region 代表一段左闭右开的数据区间，每个 Region 包含一主 N 备的多个副本，不同副本分散在不同的 TDStore 节点上。客户端对某一行数据的访问，在经过 SQLEngine 编码后，会将请求发送到对应的 TDStore 上对应的 Region 上。
在分布式事务中，TDStore 承担协调者的角色，由 Region 的 Leader 副本进行响应。

管控节点 TDMC

TDMC 为管控节点集群，负责实例内部的资源管控调度，以及全局唯一性的数据资源的管理。

- 在资源调度方面，TDMC 主要负责根据 TDStore 心跳上报的信息，对 Region 下发进行分裂、切主、合并、迁移等任务，同时向 SQLEngine 提供最新的全局的 Region 路由信息。
- 在数据管理方面，TDMC 主要负责向计算和存储节点提供全局唯一且严格递增的事务时间戳，用以实现数据多版本管理以及可见性的判断。另外，MC 还负责管理 MDL 锁、系统变量参数等等。

地域和可用区

最近更新时间：2021-12-14 15:43:21

InnoDB 引擎

公有云

腾讯云目前提供多个可选地域，TDSQL MySQL版 支持的地域和可用区可在 TDSQL MySQL版 购买页 查看。

金融云

针对金融行业监管要求定制的合规专区，具有高安全，高隔离性的特点；已认证通过的金融行业客户可提工单申请使用专区，详见 金融专区介绍 。

TDStore 引擎

内测阶段，目前支持公有云华南（广州）、华北（北京）、华东（上海）地域。暂不支持金融云。