腾讯云分布式数据库解决方案

DCDB

[2017 年 8 月 30 日]

[版本 3.1]

【版权声明】

【商标声明】

【服务声明】

本文档意在向客户介绍腾讯云全部或部分产品、服务的当时的整体概况，部分产品、服务的内容可能有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或模式的承诺或保证。

目录

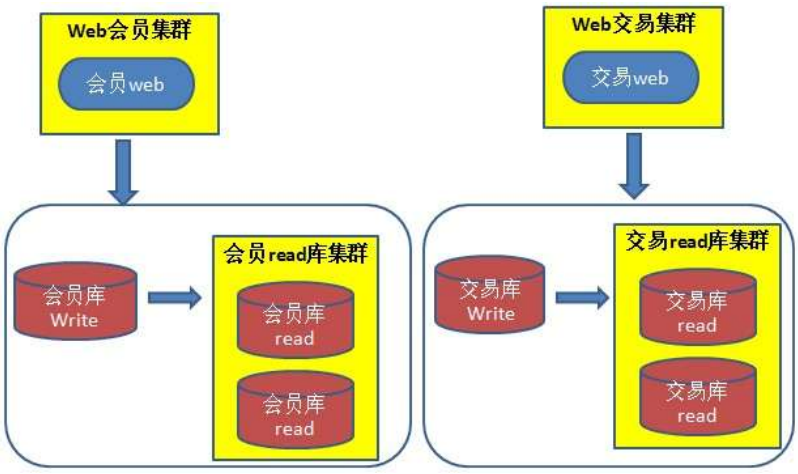
1 概述

1.1 腾讯云分布式数据库（DCDB）发展历史

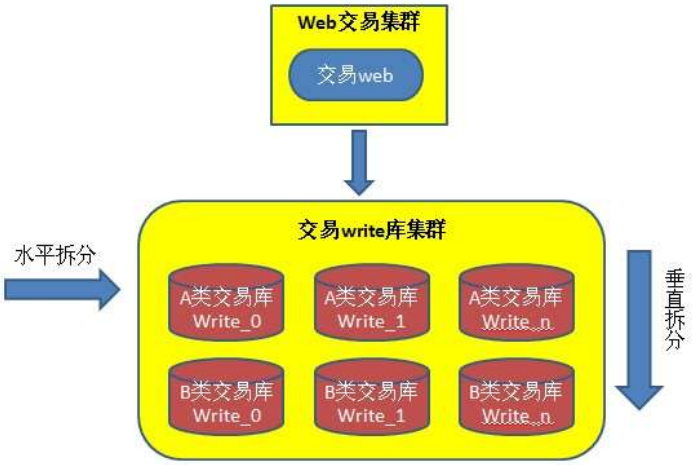
分布式数据库（DCDB-Distributed Cloud DataBase）是随着腾讯业务规模不断扩大而发展起来的。从 2004 年开始，腾讯部分业务就已经开始遇到单机数据库架构已经无法支撑，进而转为分布式架构，业务发展最终推动了数据库架构技术的不断革新，面对日益复杂的需求。截止到 2017 年，分布式数据库经历了 13 年的发展，包括微信支付，腾讯充值，阅文集团等众多业务都使用了腾讯分布式数据库。

1.2 分库与分表

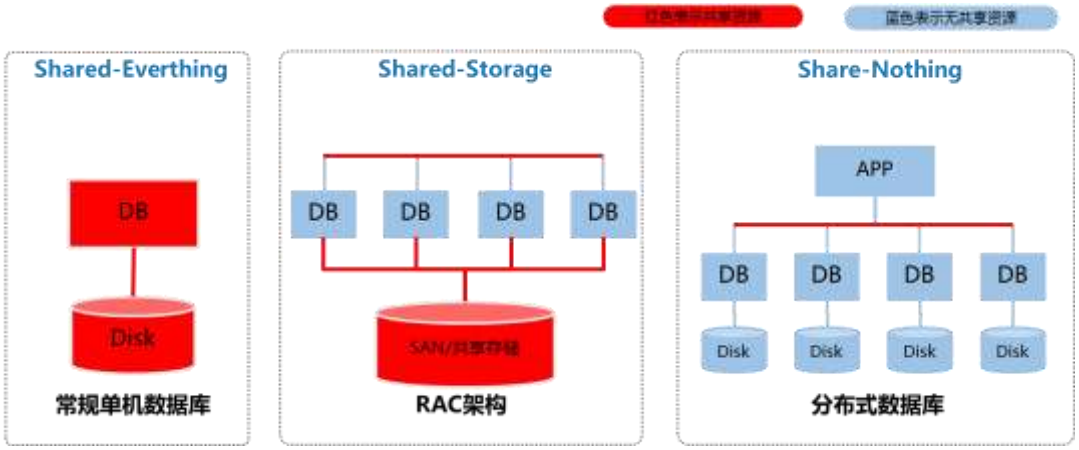
垂直切分(通常也叫做“分库”)也就是按功能切分数据库，这种切分方法跟业务紧密相关，实施思路也比较直接，比如“京东 JD”等电商平台，一个原有一个数据库实例，按功能切分为会员数据库、商品数据库、交易数据库、物流数据库等多个数据库实例，共同承担业务压力。



有时候，垂直拆分并不能彻底解决压力问题，因为单台数据库服务器的负载和容量也是有限的，随着业务发展势必也会成为瓶颈，解决这些问题的常见方案就是水平切分了。**水平切分（又叫做“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，这些“独立”的数据库“分片”；多个分片组成一个逻辑完整的数据库实例。一般来说，分表的前提是分库。



水平拆分的方案，实际上是分布式架构最直接体现，他与 RAC 等方案的最大不同点是，每个计算节点都参与计算和数据存储，每个计算节点都仅存储一部分数据。因此，分布式数据库从架构上来讲，性能是可以线性增长的。



2 业务痛点

业务在采购之初很难准确预测未来业务增长的速度和总量，这就导致业务不得不采购比自己实际需求更多的资源。这可能导致了两个问题：一是资源的浪费，二是难以快速扩展。例如，您可能购买了一台可支撑 10TB/1000W 用户的数据库，可能业务发展只利用了 10%的资源 ,另 90%的资金就浪费掉了。或者 ,随着时间推移 ,数据规模远远的超过了 10TB/1000W，而按照 Scale-up（又名纵向扩展、垂直扩展）的理念，解决方案就是购买更高规格的资源，那么难免面临数据迁移的问题，用户必须停机迁移数据，意味着服务的中断；而且，任何硬件或软件都有性能天花板，这意味着性能和容量扩展是有限的。而 Scale-out （又名横向扩展，水平扩展）架构解决了这个矛盾，用户仅需按需采购存储，刚开始用普通的 x86 服务器就够了，一旦性能或容量不够了，再购置一台接入到集群中就可以了。

从性能角度来讲，互联网业务带来的是动辄千万用户，百亿行的数据规模；而且，随着时间的增长仍在持续增长，即使是某些商业数据库，在如此海量的并发访问，海量的数据规模下都无能为力；而我们所知的所有大型互联网公司，却能用很低的成本，轻易支撑起比这高得多的业务规模——这中间的秘密就是“分布式数据库”。

3 腾讯云分布式数据库

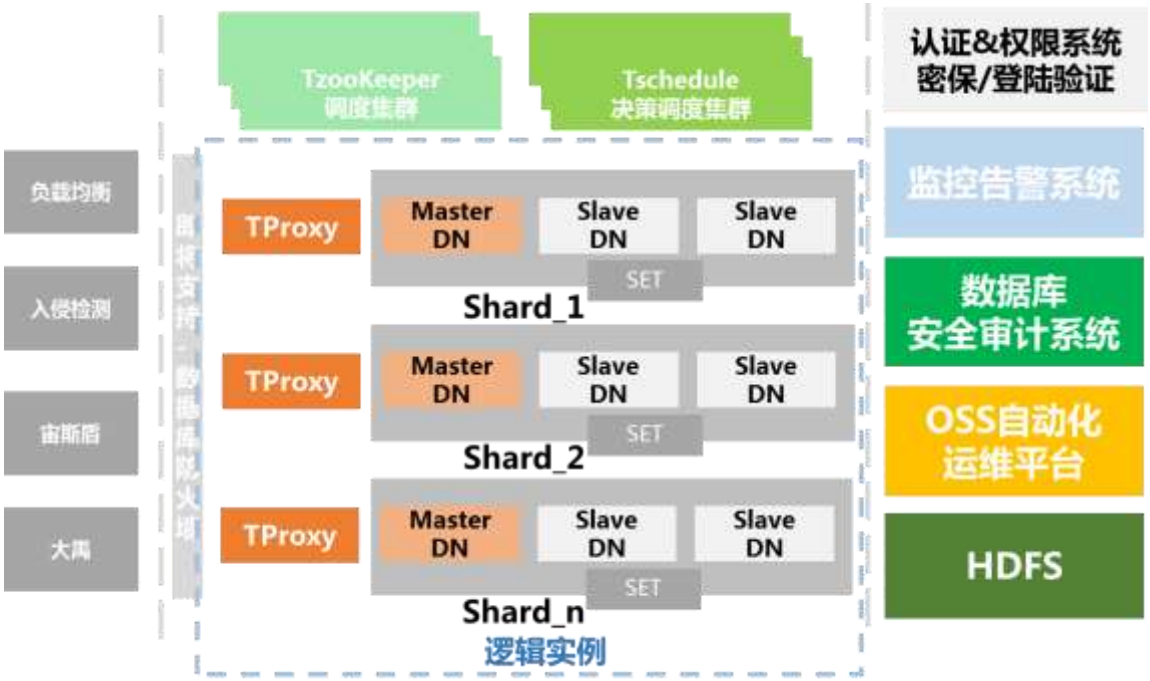
3.1 简介

腾讯云分布式数据库（简称 DCDB）是部署在腾讯云公有云上的一种兼容 MySQL 协议和语法，支持自动水平拆分（分表）的 share nothing 架构的分布式数据库。其前身是腾讯自研 TDSQL 数据库，分布式数据库即业务获取是完整的逻辑库表，后端却将库表均匀的拆分到多个物理分片节点。目前，DCDB 默认部署主备架构且提供了容灾、备份、恢复、监控、迁移等方面的全套解决方案，适用于 TB~EB 级的海量数据库场景。

当前，DCDB 已支持兼容 MySQL5.7（基于 percona 分支），并计划在 2018 年支持 PostgreSQL 分布式方案引擎。

3.2 产品架构

DCDB 采用集群架构，整个集群架构简图如下图。其中，DCDB 最核心的四个主要模块是：决策调度集群（Tschedule）、数据库节点组（SET）和接入网关集群（TProxy），配置调度集群（TzooKeeper）完成。



逻辑实例：从业务视角看到的一个具有完整能力的数据库实例；

物理分片(Sharding)：又简称“分片”，是由数据库节点组(SET)和支撑系统组成一主多从数据库，水平拆分后承载数据的基本单元；

数据库节点组（SET）：由兼容 MySQL 数据库引擎、监控和信息采集（Tagent）组成，通常情况下：

- SET 采用一主多从架构，部署在跨机架物理服务器中；

- 每个节点（DataNode）都部署心跳监控和信息采集模块（Tagent），确保集群的健壮性；

调度作业集群（TScheduler）：帮助 DBA 或者数据库用户自动调度和运行各种类型的作业，比如数据库备份、收集监控、生成各种报表或者执行业务流程等等，DCDB 把 Schedule、zookeeper、OSS（运营支撑系统）结合起来通过时间窗口激活指定的资源计划，完成数据库在资源管理和作业调度上的各种复杂需求，Oralce 也用 DBMS_SCHEDULER 支持类似的能力。

配置调度集群（TzooKeeper）：它是 DCDB 提供配置维护、选举决策、路由同步等，并能支撑数据库节点组（分片）的创建、删除、替换等工作，并统一下发和调度所有 DDL（数据库模式定义语言）操作，整个调度集群大于等于 3 组并跨机房部署。

运维支撑系统（OSS）：基于 DCDB 定制开发的一套综合的业务运营和管理平台，同时也是真正融合了数据库管理特点，将网络管理、系统管理、监控服务有机整合在一起。

接入网关集群（TProxy）:在网络层连接管理 SQL 解析、分配路由。(请注意，TProxy 并非腾讯云网关 TGW 集群)。

- TProxy 与数据库引擎部署数量相同，分担负载并实现高可用容灾；
- 从配置集群（TzooKeeper）拉取数据库节点（分片）状态，提供分片路由，实现透明读写；
- 记录并监控 SQL 执行信息，分析 SQL 执行效率，记录并监控用户接入信息，进行安全性鉴权，阻断风险操作；
- TProxy 前端部署为腾讯网关系统 TGW，对用户提供唯一一个虚拟 IP 服务。

这种集群架构极大简化了各个节点之间的通信机制，也简化了对于硬件的需求，这就意味着即使是简单的 x86 服务器，也可以搭建出类似于小型机、共享存储等一样稳定可靠的数据库。

3.3 高度兼容 MySQL

我们的设计理念就是淡化拆分的概念，无需任何额外设置，就可以让开发者透明的使用 DCDB。因此，对于开发者来讲，DCDB 就是 MySQL Server——您可以用连接 MySQL 的方式去连接 DCDB（除了极少不兼容项以外），大多数情况下，可以用您熟悉的对象映射框架使用 DCDB。对于分表，建议您尽量使用基础的 SQL 语句，因为这样能达到最佳性能，特别是几千万甚至几百亿条记录的情况下。这意味着，您仅需要极小的改造，即可接入 DCDB。

而且，DCDB 支持腾讯云数据库全部运维管理能力，包括读写分离、监控告警、容灾备份。而且基于云的多租户架构、让您的实例具有很强的弹性和灵活。

3.4 数据自动拆分（分表）

3.4.1 分表原理简介

关系型数据库是一个二维模型，数据的切分通常就需要找到一个分表字段（shardkey）以确定拆分维度，再通过定义规则来实现数据库的拆分。业内的几种常见的分表规则如下：

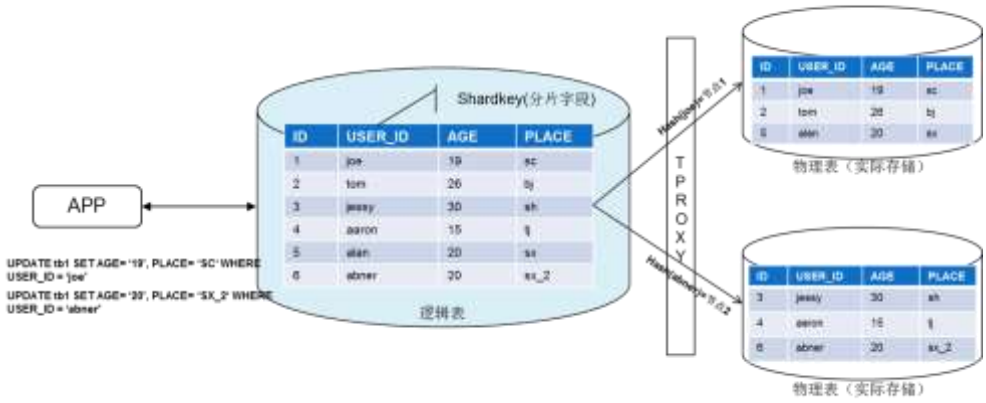
- 基于日期顺序（Time），如按年拆分，2015 年一个分表，2016 年一个分表。
- 基于某字段划分范围（Range），如按用户 ID 划分，0~1000 一个分表，1001~2000 一个分表。
- 基于某字段求模（HASH），将求模后字段的特定范围分散到不同库中。

无论是 Time、Range 都有个主要缺点就是可能导致严重数据倾斜，即多个分片之间负载和数据容量严重不均衡。例如，在大部分数据库系统中，数据有明显的冷热特征——显然当前的订单被访问的概率比半年前的订单要高的多——而采用 Time 分表或 range 分表，就意味大部分热数据将会被路由在少数几个分表中，而存储冷数据的设备性能却被浪费掉了。

因此，DCDB 通常采用某个字段求模（HASH）的方案进行分表。因为 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（某些情况下除外，我们将在后续章节进

行介绍）。

HASH 的过程大致就是，当某条记录(SQL)请求时被发起时，DCDB 会理解 SQL 语句的含义，然后按照拆分键的值和执行策略将 SQL 路由到对应分表进行执行，如下图所示，先通过 hash 算法计算，再路由到各个节点上。



3.4.2 选择拆分键

拆分键是在水平拆分过程中用于生成拆分规则的数据表字段。DCDB 建议拆分键要尽可能找到数据表中的数据在业务逻辑上的主体，并确定大部分（或核心的）数据库操作都是围绕这个主体的数据进行，然后可使用该主体对应的字段作为拆分键，进行分表（该分表方案叫做 groupshard），如下图：

Groupshard 的分表方案，确保可以确保某些复杂的业务逻辑运算，可以聚合到一个物理分片内。例如，B2B/B2C 电商平台订单表和用户表都是基于用户维度（UserID/CustomerID）拆分，平台就可以很容易的通过联合查询（不会存在跨节点 join，或分布式事务）快速计算某个用户近期产生了多少订单。

下面举例了典型应用场景都有明确的业务逻辑主体，可用于拆分键，但我们仍然建议您在选择拆分键前，仔细分析业务特点和数据库中每张表里实体关系，找到一个与大多数表都相关的实体（字段）作为拆分键：

- 面向用户的互联网应用，都是围绕用户维度来做各种操作，那么业务逻辑主体就是用户，可使用用户对应的字段作为拆分键；
- 电商应用或 O2O 应用，都是围绕卖家/买家维度来进行各种操作，那么业务逻辑主体就是卖家/买家，可使用卖家/买家对应的字段作为拆分键；但请注意，某些情况下几个超大卖家占到绝大多数交易额，这种情况会导致某几个分片的负载和压力明显高于其他分片，我们会在后面章节予以说明。
- 游戏类的应用，是围绕玩家维度来做各种操作，那么业务逻辑主体就是玩家，可使用玩家对应的字段作为拆分键；
- 物联网方面的应用，则是基于物联信息进行操作，那么业务逻辑主体就是传感器/SIM 卡，可使用传感器、独立设备、SIM 卡的 IMEI 作为对应的字段作为拆分键；
- 税务/工商类的应用，主要是基于纳税人/法人的信息来开展前台业务，那么业务逻辑主体就是纳税人/法人，可使用纳税人/法人对应的字段作为拆分键。

以此类推，其它类型的应用场景，大多也能找到合适的业务逻辑主体作为拆分键的选择。

3.4.3 拆分键的限制

为了提高语法解析效率，避免因为 shardkey 设置导致路由错误，DCDB 规定了拆分键设定的技术限制（更多详情，请参考腾讯云官方文档）：

1. 如存在主键或者唯一索引，则 shardkey 字段必须是主键以及所有唯一索引的一部分；
2. shardkey 字段的类型必须是 int,bigint,smallint,char,varchar；
3. shardkey 字段的值尽量使用 ascii 码，网关不会转换字符集，所以不同字符集可能会路由到不同的分区（且尽量不要有中文）；
4. 不能 update shardkey 字段的值，如必须则先 delete，再 insert；

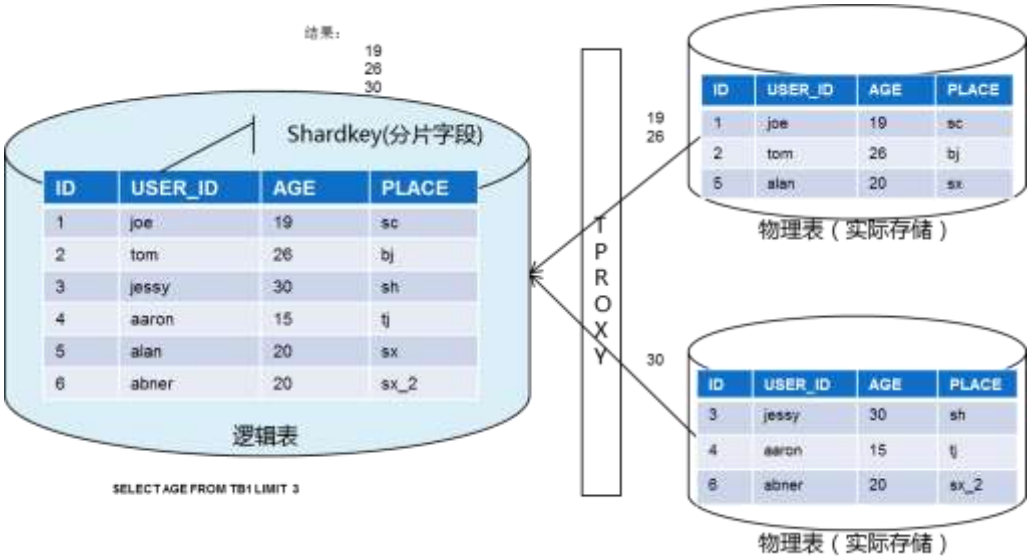
5. `shardkey=` 放在 create 语句的最后面，如下示例：

```
mysql> create table test.right ( a int not null,b int not null, c char(20) not null,primary
key(a,b) ,unique key(a,c)) shardkey=a;
   Query OK, 0 rows affected (0.12 sec)
```

6. 访问数据尽量都能带上 shardkey 字段；

3.5 查询自动聚合

而如果一个查询 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。



读取数据时（如果有明确 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表
3. 数据根据分表算法，将数据从对应的分表中取出

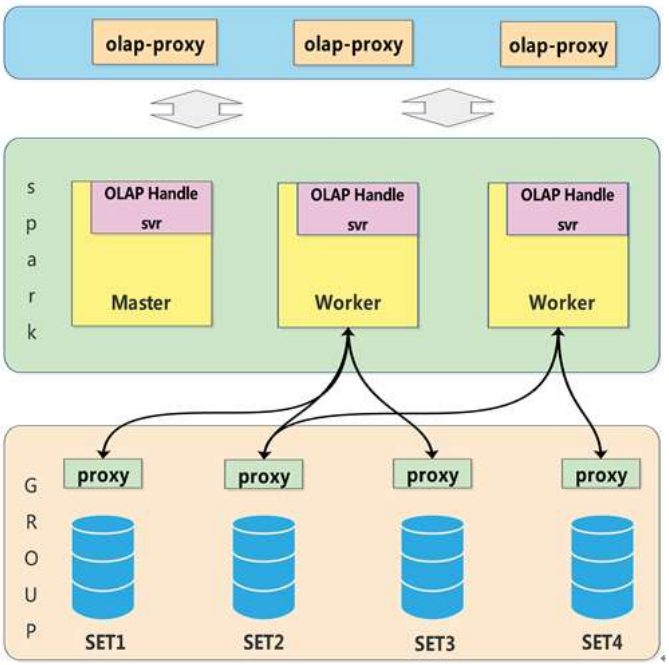
读取数据时（如果没有明确 shardkey 值）：

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

从上述原理来看，查询 SQL 中含有 shardkey 值比不含 shardkey 值效率将会更高。

3.6 数据分析能力（内测中）

DCDB 也推出了为解决用户复杂 OLAP 需求的方案；该方案基于 Spark 平台对于 OLAP 的优势，将其融合到 DCDB 中，为用户一站式解决大数据或数据分析(Analytical Processing ）需求。当您需要使用该功能时，只需要单独申请一套计算集群（ sparksql ），我们就讲自动与 DCDB 现有的数据打通（如下图）：



从架构来说，DCDB+SparkSQL 有如下优势：

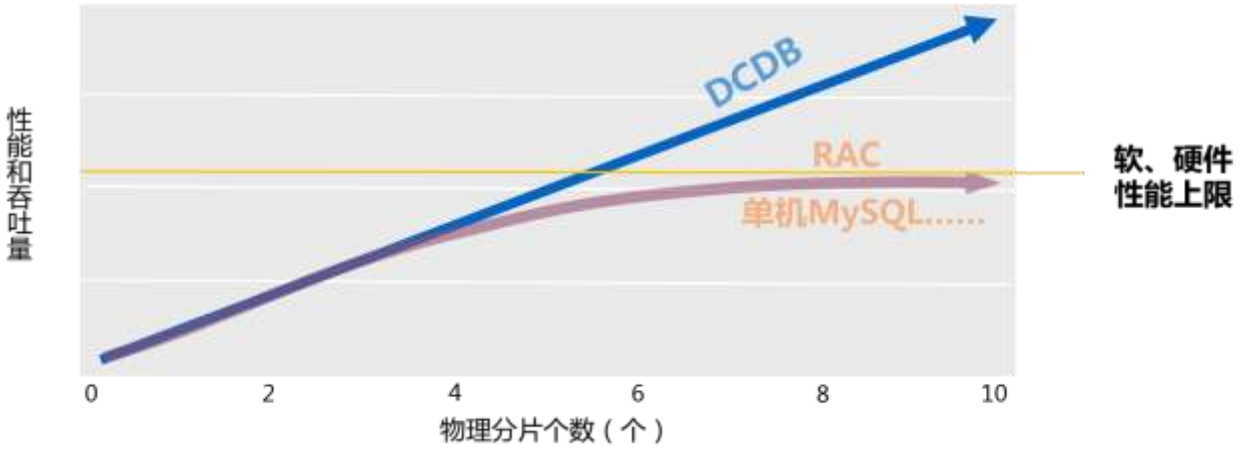
- DCDB 深度整合 Spark 引擎，采用独立的计算节点将极大的提高分析性能；
- 从同一数据源读取数据，无需维护的 ETL，简化了系统架构和运维；
- 借助 Spark 生态圈，DCDB 拥有极大的想象空间。

该功能正在内测中，并计划与 2018 年正式发布，在发布前如需体验，可提交至腾讯云工单申请或加入 QQ 群：571438378 （分布式数据库交流）交流。

4 功能与优势

4.1 性能/容量线性增长

DCDB 是天然的 MPP (Massively Parallel Processing，大规模并行处理系统)架构，这意味着随着 DCDB 分片的增加，每个分片各自承担一部分分布式任务，意味着并发性能、处理能力、存储容量将线性增长。

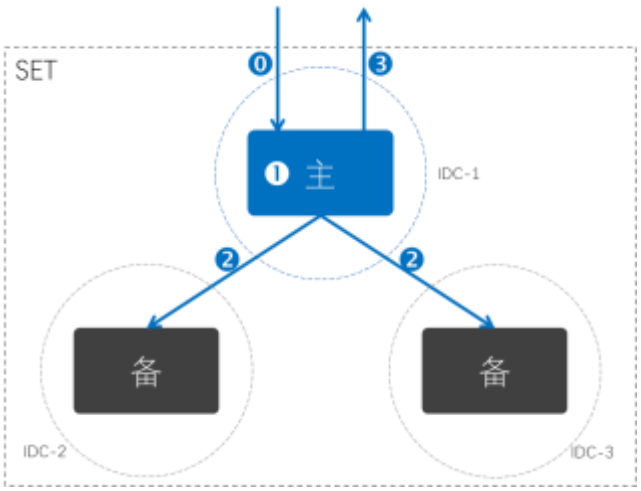


并且 DCDB 默认采用线程池 ,且对调度算法进行了优化 ,改进当系统内核处于重负载时，查询和更新请求在线程组间分布不均衡等极端情况下性能，并且能够更好地利用计算资源，减少无谓的线程切换，减少请求在队列中的等待时间，及时处理请求。类似的内核优化还有很多，通过 sysbench 的压力测试，DCDB 单个分片纯写入操作能超过 12 万+TPS，纯查询操作能超过 48 万 QPS，是 MySQL5.6 性能的 4 倍，MySQL5.7 的 2 倍以上。且腾讯云数据库团队还在持续优化。

4.2 高可用与强同步（MAR）

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；数据库作为系统数据存储和服务的核心能力，其可用要求高于计算服务资源。目前，数据库的高可用方案通常是

让多个数据库服务协同工作，当一台数据库故障，余下的立即顶替上去工作，这样就可以做到不中断服务或只中断很短时间；或者是让多台数据库同时提供服务，用户可以访问任意一台数据库，当其中一台数据库故障，立即更换访问另外数据库即可。



由于数据库中记录了数据，想要在多台数据库中切换，数据必须是同步的，所以**数据同步技术是数据库高可用方案的基础**；当前，数据复制方式有以下三种方式：

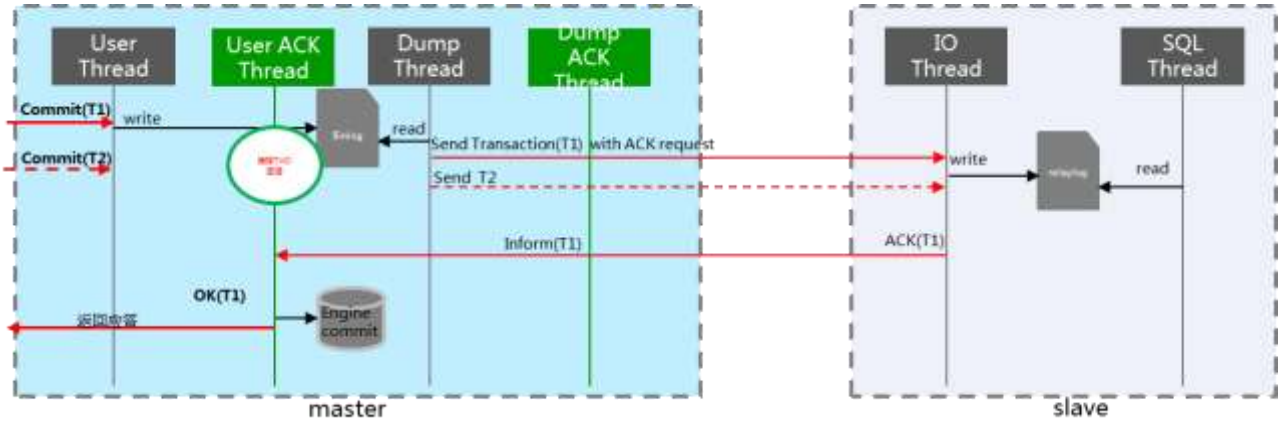
- **异步复制：**应用发起更新（含增加、删除、修改操作）请求，Master 完成相应操作后立即响应应用，Master 向 Slave 异步复制数据。因此异步复制方式下， Slave 不可用不影响主库上的操作，而 Master 不可用有概率会引起数据不一致。
- **强同步复制：**应用发起更新请求，Master 完成操作后向 Slave 复制数据，Slave 接收到数据后向 Master 返回成功信息，Master 接到 Slave 的反馈后再应答给应用。Master 向 Slave 复制数据是同步进行的，因此 Slave 不可用会影响 Master 上的操作，而 Master 不可用不会引起数据不一致。（使用“强同步”复制时，如果主库与备库自建网络中断或备库出现问题，主库也会被锁住（hang），而此时如果只有一个主库或一个备库，那么是无法做高可用方案的。—— 因为单一服务器服务，如果股指则直接导致部分数据完全丢失，不符合金融级数据安全要求。）
- **半同步复制：**半同步复制是 google 提出的一种同步方案，他的原理是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。半同步复制意味着 Master 不可用有概率会较小概率引起数据不一致。

腾讯自主研发了的基于 MySQL 协议的异步多线程强同步复制方案（Multi-thread Asynchronous Replication MAR），相比于 Oracle 的 NDB 引擎，Percona XtraDB Cluster 和 MariaDB Galera Cluster，其性能、效率和适用性更据优势。简单来说，MAR 强同步方案强同

步技术具有以下特点

- 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

腾讯 MAR 方案强同步技术，只有当备机数据同步后，才由主机向应用返回事务应答，示意图如下：



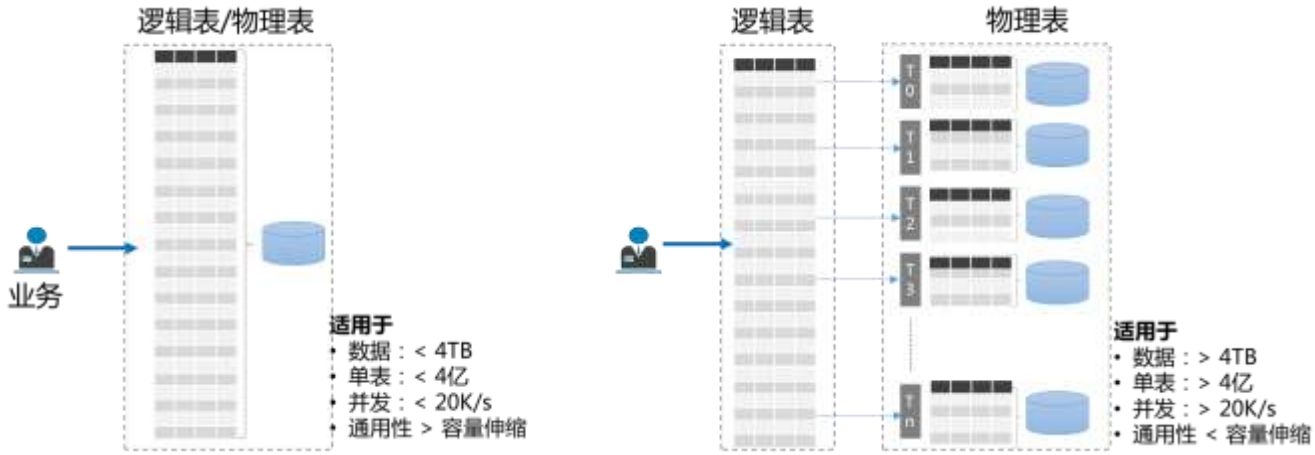
从性能上优于其他主流同步方案，通过对比在**跨可用区(IDC 机房，延迟约 10~20ms)**同样的测试方案下，我们发现其 MAR 技术性能优于 MySQL 5.6 半同步约 5 倍，优于 MariaDB Galera Cluster 性能 1.5 倍（此处测试使用 sysbench 标准用例测试）。

同步方案（跨 IDC 测试）	最大 QPS（100 并发水平）	平均耗时（ms）
DCDB MAR 强同步	486004	26
MySQL 5.7 半同步	386513	32
MySQL 5.6 半同步	10720	42
DCDB 异步同步	486004	13

MySQL 5.7 异步同步	418186	12

4.3 逻辑表

DCDB 对应用来说，读写数据完全透明，对业务呈现的表实际上是逻辑表。逻辑表屏蔽了物理层实际存储规则，业务无需关心数据层如何存储，只需要基于业务表应该如何设计。



DCDB 为用户提供了三种类似的表分表，小表以及单表：

- **分表：**是指那些原有的很大数据的表，需要切分到多个数据库的表，这样每个分片都有一部分数据，所有分片构成了完整的数据。
- **广播表：**即又名小表广播功能，设置为广播表后，该表的所有操作都将广播到所有物理分片（set）中，每个分片都有改表的全量数据。
- **单表：**主要用于存储一些无需分片的表：该表的数据全量存在第一个物理分片（set）中，所有该类型的表都放在第一个物理分片（set）中，语法和使用防范和 mysql 完全一样，您可以把他理解为一个非分布式的表。

4.4 二级分区

DCDB 支持二级分区的方案。第一级即我们常说的水平拆分，原理是使用 HASH 算法，使得数据能均匀的分散到后端的所有节点；第二级分区使用 RANGE 算法（后续还会新增支持 LIST），使得相关的数据能够落在一个逻辑分区。二级分片可以均衡数据分布和访问，为

快速一键扩容提供基础支撑，也可以满足快速删除流水数据等场景。

具体语法如下，例子中以 id 作为一级 HASH 字段，hired 作为二级 RANGE 字段：

```
CREATE TABLE employees (
            id INT NOT NULL,
            fname VARCHAR(30),
            lname VARCHAR(30),
            hired DATE NOT NULL DEFAULT '1970-01-01',
            separated DATE NOT NULL DEFAULT '9999-12-31',
            job_code INT,
            store_id INT
        )
        shardkey=id
        PARTITION BY RANGE ( YEAR(hired) ) (
            PARTITION p0 VALUES LESS THAN (1991),
            PARTITION p1 VALUES LESS THAN (1996),
            PARTITION p2 VALUES LESS THAN (2001)
        );
```

与 MySQL 类似的是，我们也对分区字段做如下要求：

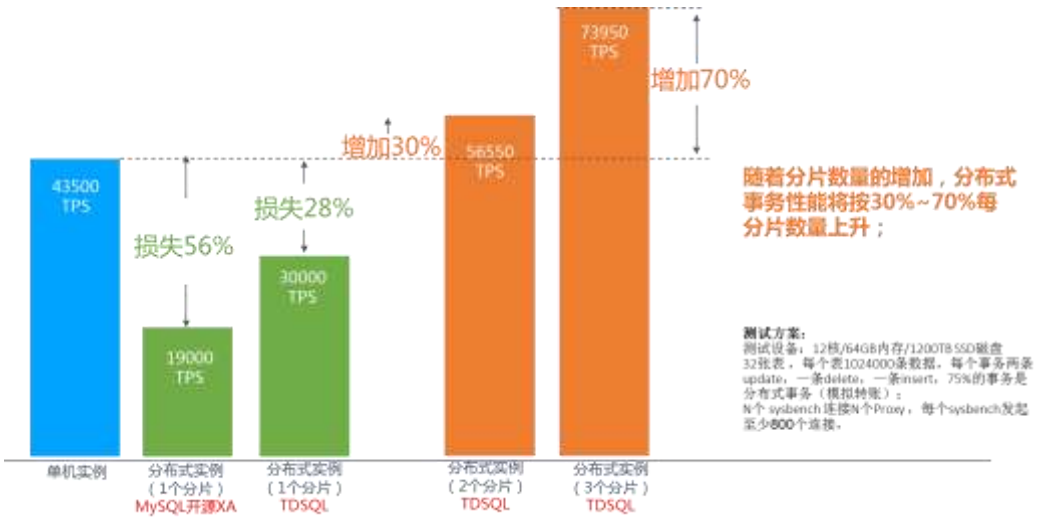
- 一级分区字段，即 shardkey 字段，类型必须是 int,bigint,smallint/char/varchar
- 二级分区字段，类型必须是 DATE，DATETIME，支持的模式为 RANGE，函数类型为 YEAR，MONTH，DAY

4.5 高性能分布式事务

分布式事务，就是一个数据库事务在多个数据库实例上面执行，并且多个实例上面都执行了写入（insert/update/delete） 操作。实现分布式事务处理的最大难点，就是在这些多个数据库实例上面实现统一的数据库事务的 **ACID 保障**，而这里面最重要的算法就是**两阶段提交**算法。分布式事务能力理论虽然很早就被提出，而业内实际工程化实现和大规模业务验证的产品还较少。

DCDB 支持分布事务，可以为银行转账、电商交易等业务提供支持有效支持。当然，分布式事务处理的开销比会比单机架构事务处理开销要大一些，使用分布式事务会导致系统 TPS 降低，事务提交延时增大。而腾讯 DCDB 通过多种优化，提供了高于开源 XA（分布式事务简称）的性能。

由于理论上，一个事务不会操作全部分片，仅操作 1~2 个分片（如转账业务），再加上 DCDB 的 MPP 架构的原因；因此一个分布式实例多个分片的分布式事务性能可以叠加。



所以是否使用分布式事务要根据实际应用需求来定：数据量非常大或者数据访问负载非常高时，分布式事务会大大降低应用开发难度，DCDB 每个事务的查询语句的写法与使用单

机架构实例完全相同，且获得事务的 ACID 保障。然而，业务中可能存在少量特别复杂的事务一次性操作所有分片，这势必会造成分布式事务性能的下降（若需要操作如此多数据，即使是单机实例耗时也会很长）遇到这种情况 ,我们建议业务谨慎平衡性能和开发难度的关系，或将事务拆解，巧妙设计；或引入一些等待机制，以优化用户体验。

为了使用分布式事务功能，需要在使用前运行如下 sql 进行初始化：

```
mysql> xa init;
Query OK, 0 rows affected (0.03 sec)

注意：该 sql 会创建 xa.gtid_log_t，用户在后续使用中万勿对其进行任何操作。
```

为提升功能，DCDB 也新增 sql 命令：

```
1) select gtid()，获取当前分布式事务的 gtid(事务的全局唯一性标识)，如果该事务不是分布式事务则返回空；
   gtid 的格式：
       ‘网关 id’-‘网关随机值’-‘序列号’-‘时间戳’-‘分区号’，例如 c46535fe-b6-dd-595db6b8-25

2) select gtid_state(“gtid”)，获取“gtid”的状态，可能的结果有：
   a) “COMMIT”，标识该事务已经或者最终会被提交
   b) “ABORT”，标识该事务最终会被回滚
   c) 空，由于事务的状态会在一个小时之后清楚，因此有以下两种可能：
          1）一个小时之后查询，标识事务状态已经清除
          2）一个小时以内查询，标识事务最终会被回滚

3) 运维命令：
   xa recover：向后端 set 发送 xa recover 命令，并进行汇总
   xa lockwait：显示当前分布式事务的等待关系（可以使用 dot 命令将输出转化为等待关系图）
   xa show：当前网关上正在运行的分布式事务
```

以转账事务为例，可以支持如下方案：

```
(python 程序)
   db = pymysql.connect(host=testHost, port=testPort, user=testUser, password=testPassword,
database=testDatabase)
   cursor = db.cursor()
   try:
       cursor.execute("begin")
```

```
        #为一个账户 Bob 的余额减 1
        query = "update t_user_balance set balance = balance - 1  where user='Bob' and balance>1)
        affected = cursor.execute(query)
        if affected == 0: #余额不足，回滚事务
            cursor.execute("rollback")
            return

        #为一个账户 John 的余额加 1
        query = "update t_user_balance set balance = balance + 1  where user='John')
        cursor.execute(query)

        # 为了安全起见，建议在这里执行‘select gtid()’获取当前事务的 id 值，便于后续跟踪事务的执行情况

        #提交事务
        cursor.execute("commit")
    except pymysql.err.MySQLError as e:
        # 发生故障，回滚事务
        cursor.execute("rollback")
```

4.6 热点更新

秒杀或大型活动可能导致瞬时超大并发修改数据库某个参数值。传统的方案是将商品库的子库前置在 cache 层或业务层，通过蜕化数据强一致（后通过第三方对账确保库存和抢购一致），而仅保证单个用户看到的库存减少规律一致（确保用户不会一会儿看见商品还有 10 个，过一会儿发现商品还剩 12 个导致投诉）。稍稍研究下，我们就会发现，这种实现方案相当复杂。而 DCDB 通过在数据库层直接实现“热点更新能力”来做到满足业务秒杀的需求，不仅减少了出错的概率，还提升了极大的开发效率。

4.7 灵活的读写分离

DCDB 默认支持读写分离能力，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；我们提供多种读写分离方案供您选择，且您无需关注若干从机是否完全存活，因为系统将根据策略自动调度

- **只读帐号：**您仅需要在创建帐号时，标记为只读帐号，系统将根据策略向将读请求发往从机；
- **/*slave*/注释：**您可以在编程过程中，通过注释/*slave*/，系统将把该条语句发往从机，常用于编程阶段将特殊的读逻辑嵌入代码。

由此为您的应用提高总的读取吞吐量；通过多种只读方案的组合，您可以配置出复杂的只读方案，以满足您各种业务需求和开发的灵活性。

4.8 全局唯一数字序列

数据切分后，原有的关系数据库中的主键约束在分布式条件下将无法使用，因此需要引入外部机制保证数据唯一性标识，这种保证全局性的数据唯一标识的机制就是全局唯一数字序列（sequence）。

DCDB 全局唯一数字序列（以下简称 sequence，使用的是 unsigned long 类型，8 个字节长），使用方法与 MySQL 的 AUTO_INCREMENT 类似。目前 DCDB 可以保证该字段全局唯一和基本有序，但不保证连续性。

4.9 多租户与独享能力

4.9.1 多租户

DCDB 基于公有云，默认采用多租户方式为您提供服务，多租户意味着您的实例并非独占一整套集群，而是与其他租户共用。腾讯多租户方案保障客户的数据安全隔离，当实例被销毁时，数据也将被完全销毁。

4.9.2 基于多租户闲时超用技术

虚拟化让多个租户的业务共享物理设备性能，而传统隔离方案严格限制了每个租户实例的性能大小。这种限制方案很公平，但没有考虑到业务特点：大多数业务仅在一天（一月）的少数时刻有较大的业务压力（如下图）： 该业务日 CPU 平均使用率仅 30%，而一天中仅存在 7 次业务压力较大，CPU 使用率在 80%~100%。虽然云能够基于弹性扩容，然而普通的弹性方案在这种突发性的压力面前，仍然无能为力——可能当您反应过来，您的业务峰值已过；最终，您还得基于业务峰值配置实例。

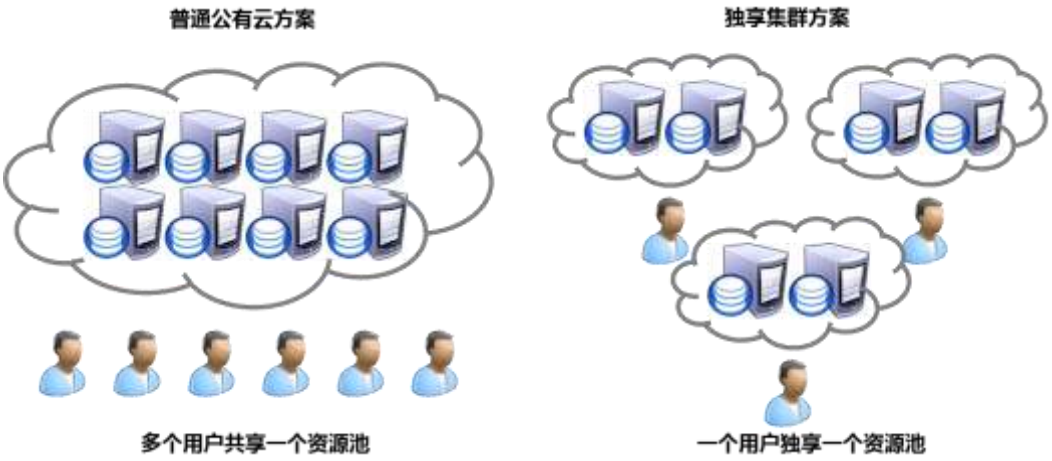
闲时超用技术，即在绝对保证每个实例预分配性能下限的基础上，允许实例使用超过预分配的性能。举个例子：假定 A 实例承载上海股票交易所的业务，B 实例是承载纳斯达克股票的业务，A、B 实例被分配到一台物理设备中，A 可以在 B 的空闲时间，抢占（有限的，并发全部）一部分空闲性能。当然，A、B 同时面对峰值时，我们确保 A 分配的 CPU 基本数量。相对于传统的方案，闲时超用是一种更加灵活的性能隔离方案，让您的业务在面对偶然性峰值时也能游刃有余。

4.9.3 独享集群数据库

DCDB 也支持独享集群数据库（Database Dedicated Cluster）方案。独享集群，可以让您以独享物理集群（完整设备）资源方式购买、创建数据库，以满足您对资源独享、物理安全、

行业监管等需求；购买独享集群后，您可以在其上灵活创建多种自定义规格的云数据库。

独享集群与公有云多租户的方案可以用下图表示：



独享集群方案可以享有 DCDB 的全部能力，但在部署架构上有如下区别：

- 数据库服务器：承载数据库的物理主机，是您使用数据库的主要部分，您将完全独占。
- 腾讯云综合运营管理系统：部署在腾讯云更高等级的管理网络中，是您提供灵活的运维能力的基础，您将与其他用户共享。
- 腾讯云网关/安全网关：部署在数据库前端的提供智能负载均衡和安全管理的虚拟网络设备，为您的数据库提供 VIP（唯一虚拟 IP）、VPC、主备切换、安全防护等能力，您将与其他用户共享。
- 您可能得自留资源冗余，以保障集群的高可用。

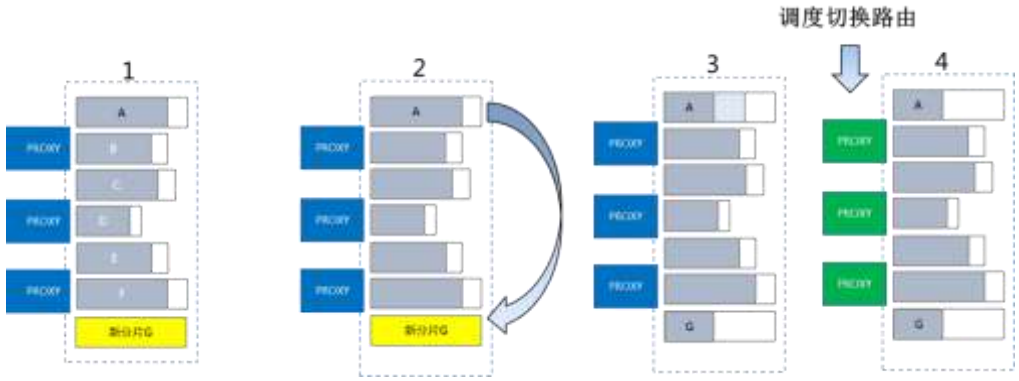
4.10 弹性扩展

4.10.1 自动再均衡技术

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；DCDB 在线扩容仅需管理员到腾讯云 WEB 控制台点击付费即可，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。

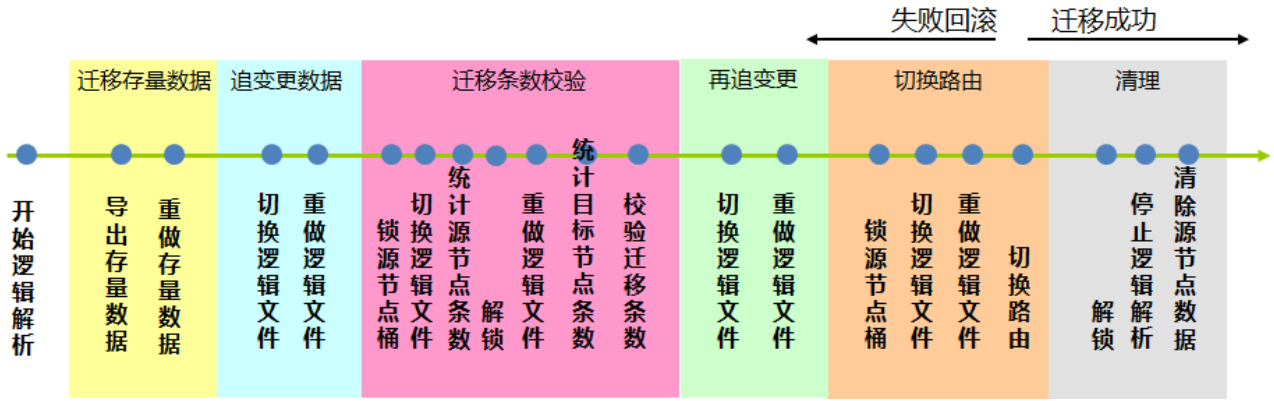
DCDB 主要是采用自研的自动再均衡技术（rebalance）保证自动化的扩容和稳定，以新

增分片为例，扩容过程如下下图：



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 校验数据库，（存在 1~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。

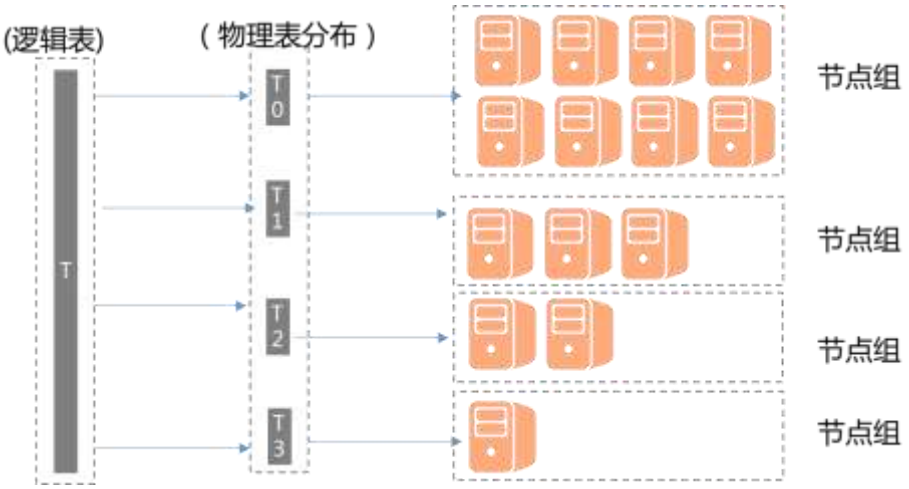
为确保业务不停以及数据一致性，DCDB 的整个迁移过程采用移存量数据、迁移增量数据、数据检验、再追增量、切换路由、清理 六个步骤循环迭代进行。该能力经过腾讯内外海量业务迁移，至今未发生过一次数据异常错误或全集群停机。



4.10.2 解决数据倾斜与分区分表

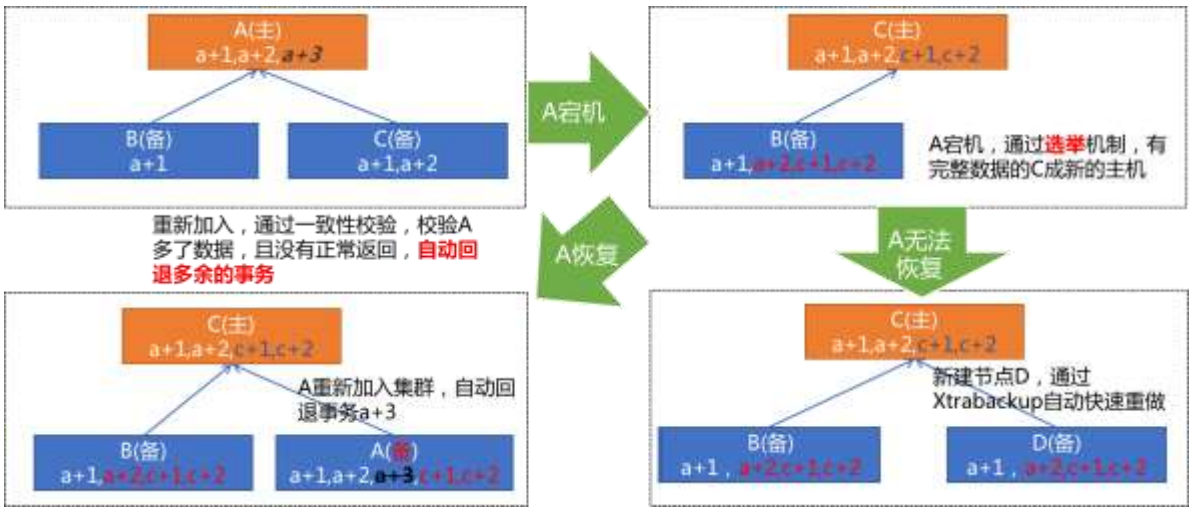
数据倾斜多导致个分片之间负载和数据容量严重不均衡，DCDB 可以针对某一个负载较高的分片单独扩容，单个分片最大容量可到 18TB 或以上（当前公有云默认最大 6TB，超过的需提交工单申请）。

除了利用分片物理性能扩展的方案，DCDB（**计划 2018 年**）支持在分表的基础上，支持分区方案；这种方案可以通过组合，形成虚拟节点组（如下图），节点组方案可以让多个节点共同承担性能和容量。



4.11 容灾与恢复

DCDB 的每一个分片都支持基于强同步的一主多从架构（默认为一主一从，异步同步），主数据库故障时备机立即顶替工作，切换过程对用户透明，且不改变访问 IP。并且对数据库和底层物理设备提供 7X24 小时持续监控。发生故障时，DCDB 将自动重启数据库及相关进程；如果节点崩溃无法恢复，将通过备份文件自动重建节点（如下图）。该功能持续保障数据库可用性在 99.99%以上，以帮助业务持续稳定的运行。



而 DCDB 也支持定时备份您的数据库实例，自动备份保留期可配置，自动备份存储在统一的安全备份存储内，提供 99.99999%的数据持久性；在业务出现异常时，自动备份和恢复能力可使您的业务损失减少到最小。

4.12 统一的参数管理

您也可以利用腾讯云管理控制台的参数设置统一管理 DCDB 实例参数，一组数据库参数包括一系列自定义的引擎配置值，这些配置应用于实例中的数据库。统一参数管理有利于您对实例的管理，而不需要去操心每个分片的参数是否正确。

4.13 WEB 化管理

您可以通过 WEB 控制中心自主创建、变更 DCDB 实例，查看监控，设置帐号等。为避免安全风险，我们对帐号设置等操作仅支持 WEB 控制中心进行操作，同时记录每位访客的操作记录。

5 安全与资质

5.1 安全性说明

云数据库的管理安全与技术安全要求符合国家信息安全等级保护（三级），部分要求达到金融行业信息安全（四级）标准。

5.2 产品资质

DCDB 现通过多项国家或国际认证，包括但不限于：

- 软件著作权
- ISO22301 认证
- ISO27001 认证
- ISO20000 认证

- ISO9001 认证
- 可信云服务认证
- 信息安全等级保护
- STAR 认证

6 应用场景

6.1 实时高并发交易场景

面向金融、红包、电商、O2O、零售等行业普遍存在用户基数大、核心交易系统数据库单表上亿行且访问日益变慢，制约业务发展。 DCDB 提供线性水平扩展性能，能够极大提升数据库处理能力，可以面对类似全网推广、限时秒杀等营销活动；配合强同步复制能力，即使是金融类业务也可以完全用 DCDB 承载。

6.2 海量数据存储访问场景

面向物联网，交易订单等业务，业务数据增长迅猛，会产生超过单机数据库存储能力极限的数据，数据库实例超过 TB 级别且持续快速增长，造成数据库容量瓶颈，限制业务发展。

DRDS 可以线性扩展存储空间，提供多达 PB 级存储能力，且可以高效利用每一个物理节点的性能，避免瓶颈。可广泛应用于 IOT 场景下的车联网、工业远程监控、智能家居、智能汽车、充电桩加油站等等超大规模传感器数据上报存储访问场景；而对于互联网游戏点券交易、O2O 订单等数据量特别大的场景，也可以轻松应对。

6.3 成为去 O 的中坚力量

企业的核心业务系统一般都是 OLTP 为主的应用场景，在这个领域，Oracle 一直是市场的领导者，而开源数据库 MySQL、MariaDB、PostgreSQL 等似乎仅能提供给中小企业或个人

站长使用。在互联网领域，以 DCDB 为代表的分布式数据库应用非常广泛，用普通 x86 服务器，轻松支撑起上亿的用户访问，经过验证的好的分布式数据库在性能和稳定性上甚至高于用高端设备搭建的 Oracle RAC，而以互联网为代表的去 O 的成功案例比比皆是。。

当然，对于企业而言，由于 Oracle 数据库和上层应用绑定比较紧密，通常会使用到 Oracle 的存储过程、自定义函数、触发器，这就需要涉及到应用迁移，这个工作的工作量和时间周期通常较大，但综合计算下来，即使加上软件改造成本，采用 DCDB 的 TCO 仍然低于使用商业数据库。

6.4 分支业务聚合到总部

由于政务、银行、大型国企的组织架构通常采用总部-分部-分支的架构；因为各种原因，其某些核心 IT 系统建设也采用总部-分部-分支模式。随着业务互通，人员互通，信息互通等需求越来越强烈，业务逐渐向总部聚合。

而业务聚合一个重要问题是数据库性能和容量无法承载。以某部委为例，其省级业务系统数据规模和性能已经在用最高端的商业数据库硬件承载。如果聚合到总部，一是设备性能扩无可扩，二是软件费用和硬件成本将会是天价。因此，到现在为止，不少业务也仅能做到数据汇总，而非业务聚合。

DCDB 此类分布式数据库在微信支付、京东等超大规模业务的应用证明了，一个系统承载全国业务的可能性。—— 2014 年马年春节，总共 800 万微信用户参与争抢 4000 万个红包，最高峰期间的 1 分钟有 2.5 万个红包被领取。2016 年大年初一每分钟有 55 万个红包被发出、165 万个红包被拆开，增长近百倍。

7 案例简介

7.1 米大师

腾讯推出的移动支付组件米大师，专注移动支付解决方案，实现移动终端的更大营收。目前已全面支持微信、手机 QQ、手机 Qzone 等平台手游。当前米大师正在为多个腾讯游戏、电商支付等平台提供服务。目前，米大师分布式物理节点数超过 1000 个，账户总量超过 100 亿，每日请求超 10 亿，平均 99.95%的请求在 5ms 以内响应，连续三年运营零中断、零数据误差。

7.2 汇通天下

汇通天下是目前国内最大的物流解决方案厂商之一，为企业提供物流运输平台服务、帮助企业整合运力资源降低成本、以及全国货运信息平台服务。目前也是全国最大的配货信息平台，拥有注册会员 30 多万，每日及时信息 200 多万条，客户包括 TCL、可口可乐、福田等大型制造企业；中国邮政、德邦快运、华运通、宝供、佳吉快运等大型物流企业，

目前，其物联网平台采用了 DCDB，峰值每秒写入量超过百万级数据，月新增数据 TB 级。这样的规模是传统的集中式架构所不能支持的，投入将是天文数字，且不得不容忍大部分时间空转浪费。

7.3 威富通

威富通作为移动支付解决方案商，对后端对接了包括微信支付、财付通、京东钱包、支付宝、银联等各类渠道。是国内最大的支付服务商之一，原有 Oracle 数据库已经无法支撑。2016 年下半年，威富通启动从从 Oracle>DCDB 业务，仅 2 个月开发实现完毕（其他客户取决于业务架构），目前已经稳定支撑日交易峰值千万+次的业务。

7.4 三一重工（树根互联工业物联网）

树根互联技术有限公司由三一物联网及工业大数据技术及运营核心团队创建，是独立开放的互联网高科技企业 ,致力于打造最具客户价值的工业物联网平台 ,依托深厚的工业积淀，汇聚了大批工业大数据科学家团队，打造了开放的工业物联网生态系统。

工业领域采集的数据量则是消费级的成百上千倍，以三一重工的挖掘机为例，需要采集位置、油温、油位、压力、温度、工作时长等超过 5000 多个参数。如此大量数据，传统数据库根本无法承载，因此三一重工选择了 DCDB 产品。

8 腾讯云简介

腾讯公司成立 17 年，第一个产品 QQ 其实就是一朵云。从 PC 时代第一版的 QQ 到现在，腾讯云始终积极地探寻，从解决如何稳定服务、让用户的 QQ 不掉线；到解决如何满足用户越来越丰富的需求——更多的社交、更好玩的娱乐、更丰富的在线生活；再到如何开放、如何实现一个中国最大互联网生态平台的价值，腾讯云一步未曾松懈，困难始终巨大，阻碍从未变少，但腾讯精神，技术、实力、还有我们对用户永不怠慢的热情，让腾讯云走到今天。

多年来，腾讯云基于 QQ、QQ 空间、微信、腾讯游戏等真正海量业务的技术锤炼，从基础架构到精细化运营，从平台实力到生态能力建设，腾讯云将之整合并面向市场，使之能够为企业和创业者提供集云计算、云数据、云运营于一体的云端服务体验。

云计算为 IT 乃至整个商业市场带来的变革早已不是空谈。传统企业在云时代得以根本意义上的转型，大企业在云端获得源源不断的生命力，中小企业通过云，更快地面向市场获得机遇与发展。未来，会有更多的企业将加入云的世界，腾讯云将致力于打造最高质量、最佳生态的公有云服务平台。让企业更专注业务，而将基础建设放心地交给腾讯云。

联系我们

QQ：800033878

电话服务：4009-100-100