Alibaba Cloud Database Solutions

阿里云数据库

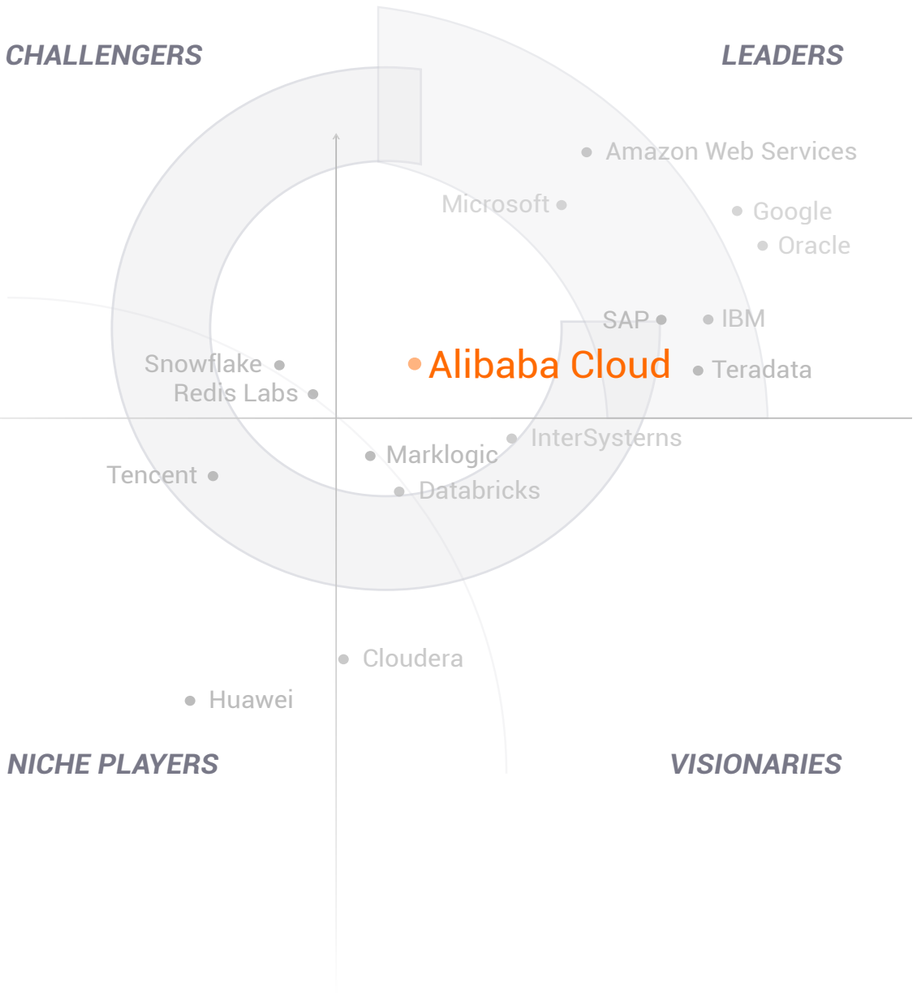
解决方案合集

历史性突破！

阿里云成为首家进入 Gartner 数据库魔力象限领导者的国内企业。

今年Gartner提出“There Is Only One DBMS Market”理念，将OPDBMS(事务处理) 和 DMSA (管理与分析) 融合为统一的 Cloud Database Management System (Cloud DBMS)，以前所未有的全面性覆盖数据管理与分析领域的各个侧面，并融入了云计算的发展趋势。

挺进该报告的「领导者」象限，也证明了阿里云在数据管理与分析领域的产品能力与发展策略在全球市场上已经属于第一集团军。



序言

阿里云智能数据库产品团队一直致力于不断健全产品体系，提升产品性能，打磨产品功能，从而帮助客户实现更加极致的弹性能力、具备更强的扩展能力、并利用云设施进一步降低企业成本。以云原生 + 分布式为核心技术抓手，打造以自研的在线事务型 (OLTP) 数据库 PolarDB 和在线分析型 (OLAP) 数据库 AnalyticDB 为代表的新一代企业级云原生数据库产品体系，结合 NoSQL 数据库、数据库生态工具、云原生智能化数据库管控平台，为阿里巴巴经济体以及各个行业的企业客户和开发者提供从公共云到混合云再到私有云的完整解决方案，提供基于云基础设施进行数据从处理、到存储、再到计算与分析的一体化解决方案。

凭借于此，阿里云数据库产品在云计算这个赛道上的快速演进，已经成长为全球前三的云数据库厂商，根据 IDC 数据库市场分析报告统计，2019 年 H2 阿里云在中国数据库市场份额第一，其中公有云占比超过 50%；根据 Gartner 2019 全球云数据库市场份额统计，阿里云数据库亚太市场份额第一，全球第三。阿里云在 2019 年荣获 Gartner 全球在线数据库魔力象限【挑战者】位置，在 2020 年荣获【领导者】位置；荣获 Forrester 2019 年全球数据库 DBaaS Wave 里强劲表现者位置。在软件领域，阿里云是中国第一家进入 Gartner 魔力象限领导者位置的公司，创造了中国软件的历史，这是阿里巴巴集团和中国软件行业的一次重大突破。

无论是个人开发者，中小企业，还是全球化大规模企业，阿里云数据库服务，都是您可靠、高效的选择。

目录 .

1 游戏行业

多场景云数据库解决方案

放眼全球市场，游戏产业收入增长迅速，且有明显的反经济周期特征。相较于日美等游戏大国，国内游戏市场的付费率和 ARPU（每用户平均收入）值都有很大的提升空间。加强精细化运营和塑造用户传播口碑，以提升用户游戏体验，成为游戏行业商家未来的核心发展方向之一。而精细化运营和用户体验提升，均需要稳定可靠的技术架构支撑，尤其是处于技术架构核心位置的数据库技术的保障。阿里云的游戏行业数据库解决方案通过使用云数据库的产品组合，应对游戏行业多种细分场景，充分利用阿里云数据库高性能、低成本、高可用、易维护的优势，帮助游戏客户适应当前和未来业务发展的需要。

游戏行业的整体特点及其对数据库的需求

通过深入调研，游戏行业的整体特点以及相关企业在使用数据库时常见需求包括以下几方面：

1 游戏行业场景多，对技术架构定制化要求高

游戏类型不同导致技术架构差异较大，很难归纳出一套可复制到所有游戏产品中的单一数据库解决方案，必须根据游戏类型细分方案。

2 突发高峰访问情况挑战数据库承载能力

游戏新开服，或有重大活动时，会有难以精确预期的玩家访问压力。传统的解决方案就是部署尽可能多的数据库来承载高峰访问，其问题是可能因为预测乐观，导致昂贵数据库资源的浪费，或因为预测保守，导致数据库资源准备不足，高峰期超载影响游戏业务，立即扩容时间又来不及，最终影响用户体验。

3 游戏合服需要平滑的数据迁移

当游戏进入平稳期后，出于提升玩家游戏体验和控制成本的目的，会有游戏合服的需求。数据库需要关注的重点就是保证数据的平滑迁移，整个迁移过程不能影响到玩家体验，且数据合并要保证完整准确，符合业务逻辑，解决冲突数据。

4 游戏例行回档需要敏捷的数据恢复能力

游戏运营过程中，可能出现有玩家利用游戏漏洞，批量复制虚拟道具或其他严重破坏游戏公平性的行为，这时就需要业务紧急修复漏洞，同时将受影响数据恢复到漏洞被利用前的状态（所谓回档），如何提升数据恢复的速度成为挑战。

游戏行业细分场景与解决方案

数据库的选型跟游戏分区方式息息相关，市面网络游戏（移动游戏）可分为两大类：

① 全区全服游戏

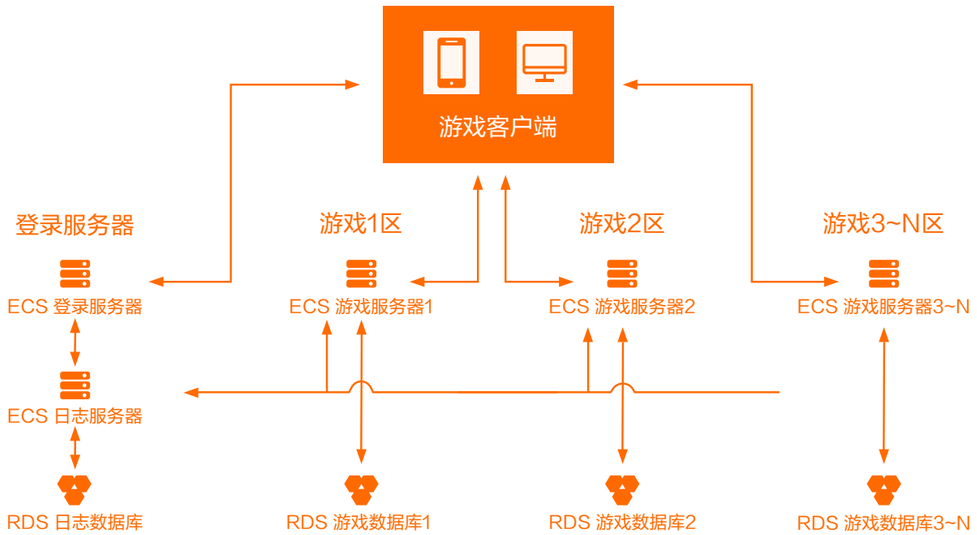
所有玩家数据在一个数据库中，只会在游戏进行时撮合玩家进入游戏房间，像MOBA 类游戏如“王者荣耀”，大逃杀游戏如“吃鸡”，或是玩家直接自己玩的休闲游戏如“开心消消乐”。此类游戏进行中或结束后，游戏状态数据会回写数据库实现持久化保存。

② 分区分服游戏

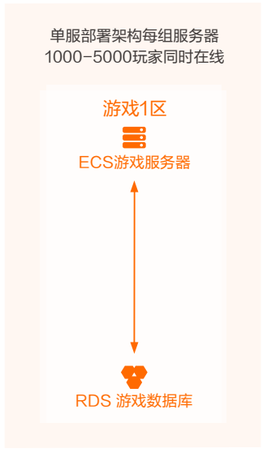
不同区玩家数据在不同数据库中，游戏有很明显的活跃玩家周期，会经历开服、活跃玩家到峰值、活跃玩家进入平稳期、合服这样的过程，如“部落战争”、“阴阳师”等。

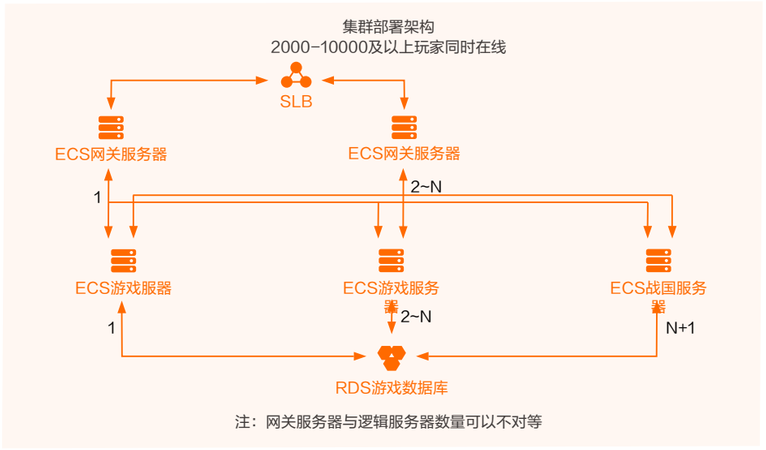
针对两种类型的游戏的数据库架构截然不同。

全区全服：因为玩家数据大集中，对数据库性能要求高；应用上一般采用多级缓存，分布式缓存技术，利用缓存承载高并发读请求，对数据库进行保护。



分区分服：因为在游戏机制上就已经把负载打散，一般一个分区数据存储不超过 10w 玩家的数据，因此游戏平稳期，分区数据库压力不会太大，但也衍生出了合服时的数据合并需求。





在游戏行业，数据库的使用场景非常多，几乎没有一个通用方案能直接适用所有的场景。在一款游戏生命周期中，数据库相关解决方案贯穿从开发到运营的整个过程，针对各阶段场景量身打造，为游戏顺利上线运营和良好用户体验提供保障。

在游戏的整个开发和运营周期，数据库的解决方案可分为以下七种：

1 玩家访问压力应对方案

游戏推广期、新开服或重大活动期间，玩家访问量突增，数据库常成为瓶颈，需要快速变配、扩容能力。

2 游戏合服方案

分区分服游戏进入平稳期后，需定期进行合服。对应的数据库也要进行合并，让数据合并后符合业务逻辑，但不能影响玩家，这是主要挑战。

3 游戏排行榜方案

针对游戏排行榜这种通用的游戏需求，提供简单高效的数据存储方案。

4 游戏回档方案

当游戏漏洞被利用，导致部分玩家数据失去平衡性，需要对相应数据做回档恢复处理。

5 掉线问题解决方案

数据库高可用、例行切换或变配，会导致应用连接闪断，可能导致玩家批量掉线重连，对游戏体验有很大影响。

6 游戏买量优化方案

游戏运营阶段，最大的支出是获客成本，即在多种渠道投放广告获取用户的开销（买量）。通过对海量的获客结果数据进行实施分析，调整不同渠道投放比例，可大幅提升买量行为效率。

7 开放世界类游戏方案

新兴的游戏类型。

1 玩家访问压力应对方案

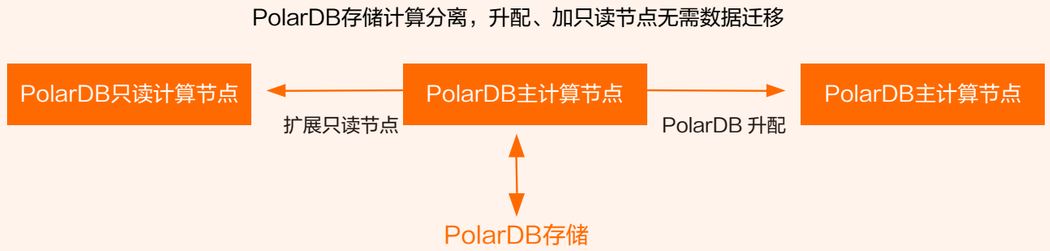
1 方案概述

游戏平稳运行期对数据库压力不大，但在以下三种情况下数据库会面临高并发访问压力：

1. 游戏推广期，买量用户蜂拥而至，频繁开新服；
2. 维护停机后开服，很多玩家等待多时，集中上线；
3. 重大版本上线或有重要活动，推广效果超预期时，会有大量老玩家回归。

由于买量和游戏运营效果具有很大不确定性，很难准确评估玩家压力模型，业内通用做法是部署尽可能多的机器资源应对，数据库购买尽量高规格 RDS，同时业务上自己分库分表，配置尽量多的 RDS 节点，这样很容易造成资源不足或浪费。如下图所示：

阿里云提供以下数据库解决方案：



云原生数据库 PolarDB 新增只读节点最多只需 5min，节点变配最多只需 15min；无需提前购买昂贵的高规格数据库，一旦数据库容量评估不足，玩家并发访问压力超过数据库负载，可快速进行升配保证游戏平稳运行；一旦游戏进入平稳期，业务压力下降，还可对 PolarDB 进行快速降配，实现高性价数据库架构。

② 操作流程与关键步骤

业务压力评估 → 购买合理规格 PolarDB → 数据迁移 → 发现负载风险升配活扩增只读节点 → 业务平稳后降配或削减只读节点

- PolarDB 实例购买后默认自带一个只读节点。
- PolarDB 的默认备份方式是快照备份，备份时间不随数据量的增长而变长。
- PolarDB 的存储默认按存储量计费，如果数据量较大建议购买存储包。

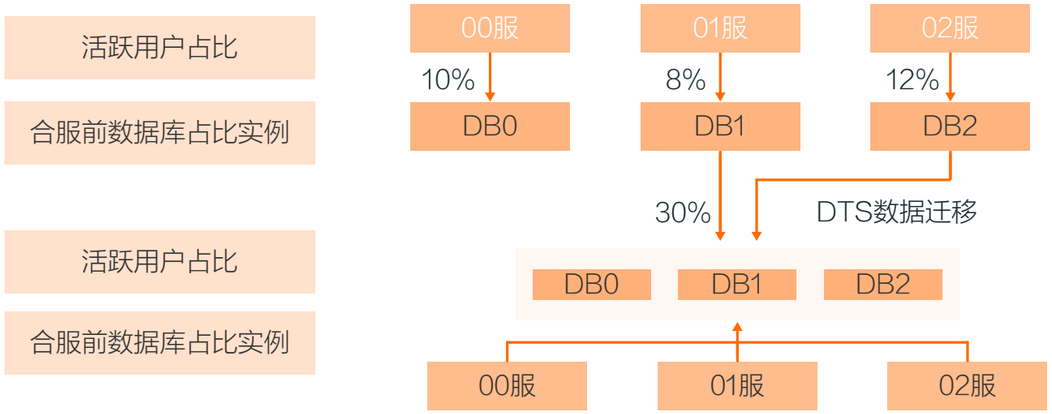
2 游戏合服方案

① 方案概述

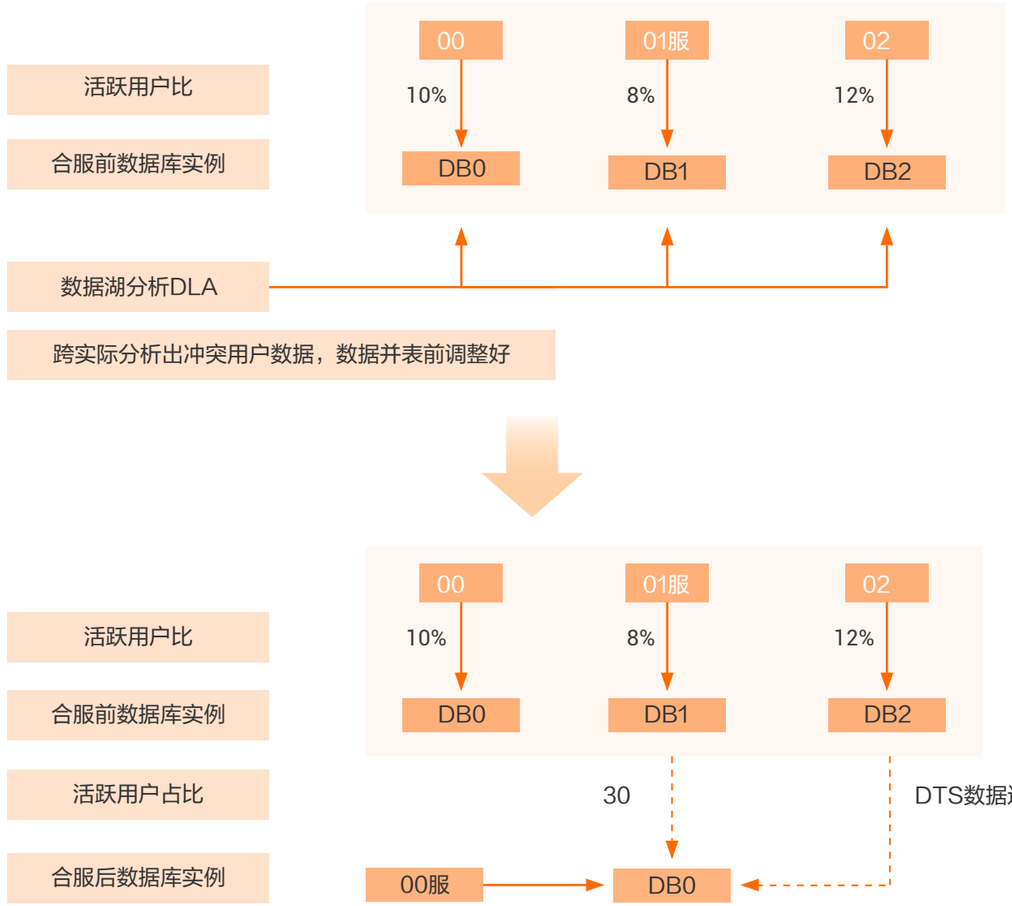
分区分服的游戏往往在游戏的平稳期会有合服的场景，原因如下：

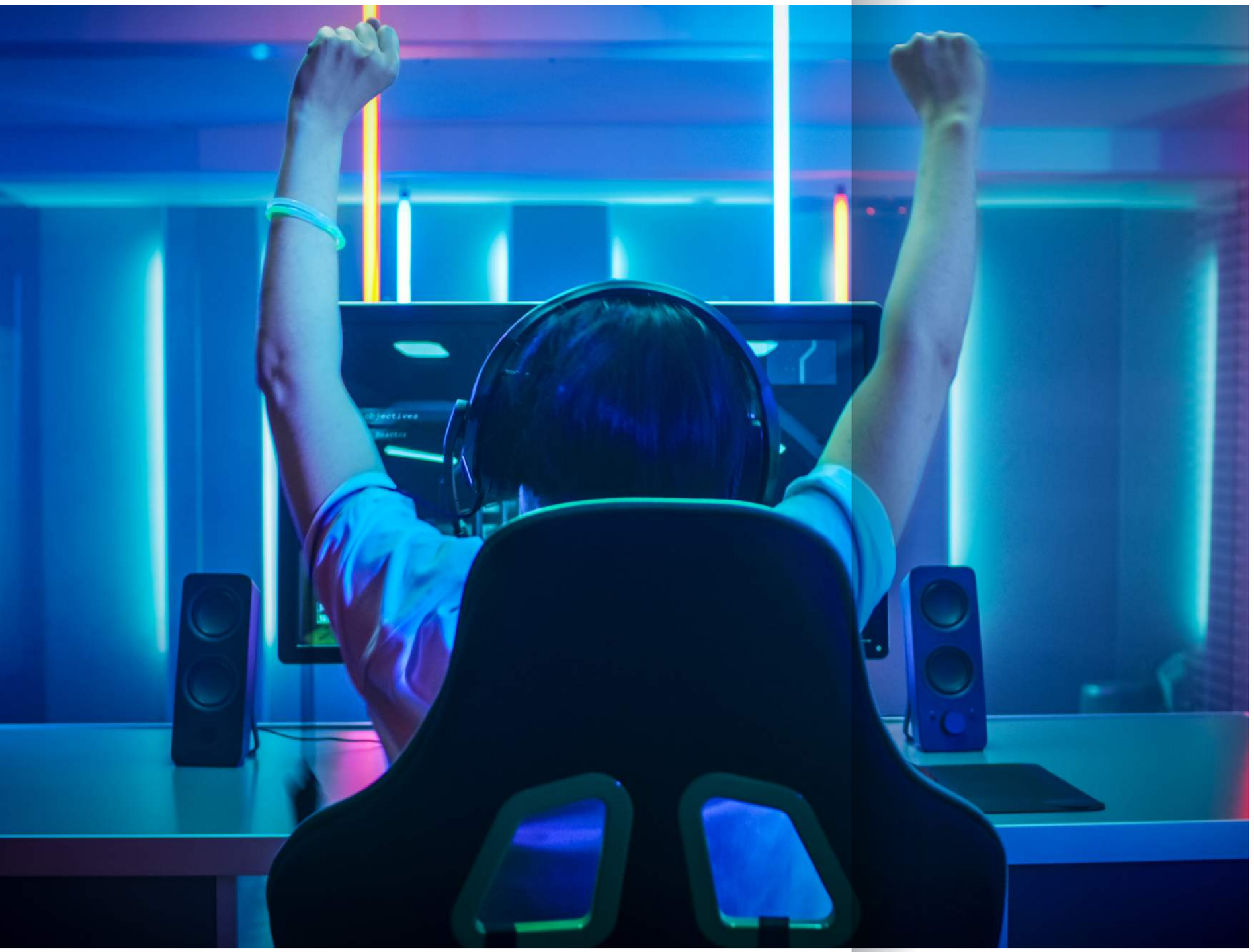
- 节约应用和数据库成本。游戏进入平稳期后，每个分区会流失相当数量的玩家，应用和数据库负载都开始走低，将原来位于多个 RDS 实例的数据库合并到一个 RDS 实例，实例负载也完全可以满足游戏运行需要，多余的实例就可以释放，节约成本。
- 提升留存玩家游戏体验的需要。因为分区玩家流失，在线进行聊天和游戏互动的玩家就会变少，留存玩家就会感觉游戏体验下降，此时合服，将分散在各个区的留存玩家聚到一起，玩家有会找到之前开新服时那种人满为患的游戏热情。

游戏合服场景的关键点是数据的迁移。一种相对简单的情况是将多个物理库从多个数据库实例集中迁移到一个数据库实例，游戏数据在逻辑上并没有真正合并，应用只需要切换数据库链接串即可，如下图所示。



另一种情况是会将多个数据库实例中玩家数据合并到同一张数据表中，不同服玩家真的合并在一个服进行游玩，游戏体验最佳，但可能会涉及玩家名称冲突、联盟名称冲突、任务进度不同步，物价差异等问题，需要游戏运营和研发人员采取相应处理对策。这种方案需要两个步骤，如下图：





阿里云数据库针对游戏合服的方案有如下优势：

- 支持多种类型数据库的数据迁移，比如游戏行业常见的 MongoDB、MySQL 和 Redis 实例间数据迁移。
- DTS 的数据迁移包含结构迁移、全量迁移和增量迁移，迁移过程可限速，不影响游戏业务，在维护器只需要切换数据链接串即可，业务割接时间短。
- DTS 使用高规格服务器来保证每条迁移或同步链路都能拥有良好的传输性能，全量数据迁移高峰期时性能可以达到 70MB/s，20 万的 TPS。
- DTS 底层为服务集群，如果集群内任何一个节点宕机或发生故障，控制中心都能够将这个节点上的所有任务秒级切换到其他节点上，链路稳定性高。内部对部分传输链路提供 7 × 24 小时的数据准确性校验，快速发现并纠正传输数据，保障传输数据可靠性。各模块间采用安全传输协议及安全 token 认证，并具有自动断点续传机制，有效地保证数据传输的可靠性。
- 数据湖分析 DLA 可作为数据处理枢纽，通过标准 JDBC 对不同服数据库实例进行关联查询，很方便的提前定位出数据合并后可能冲突的玩家数据。

② 操作流程与关键步骤

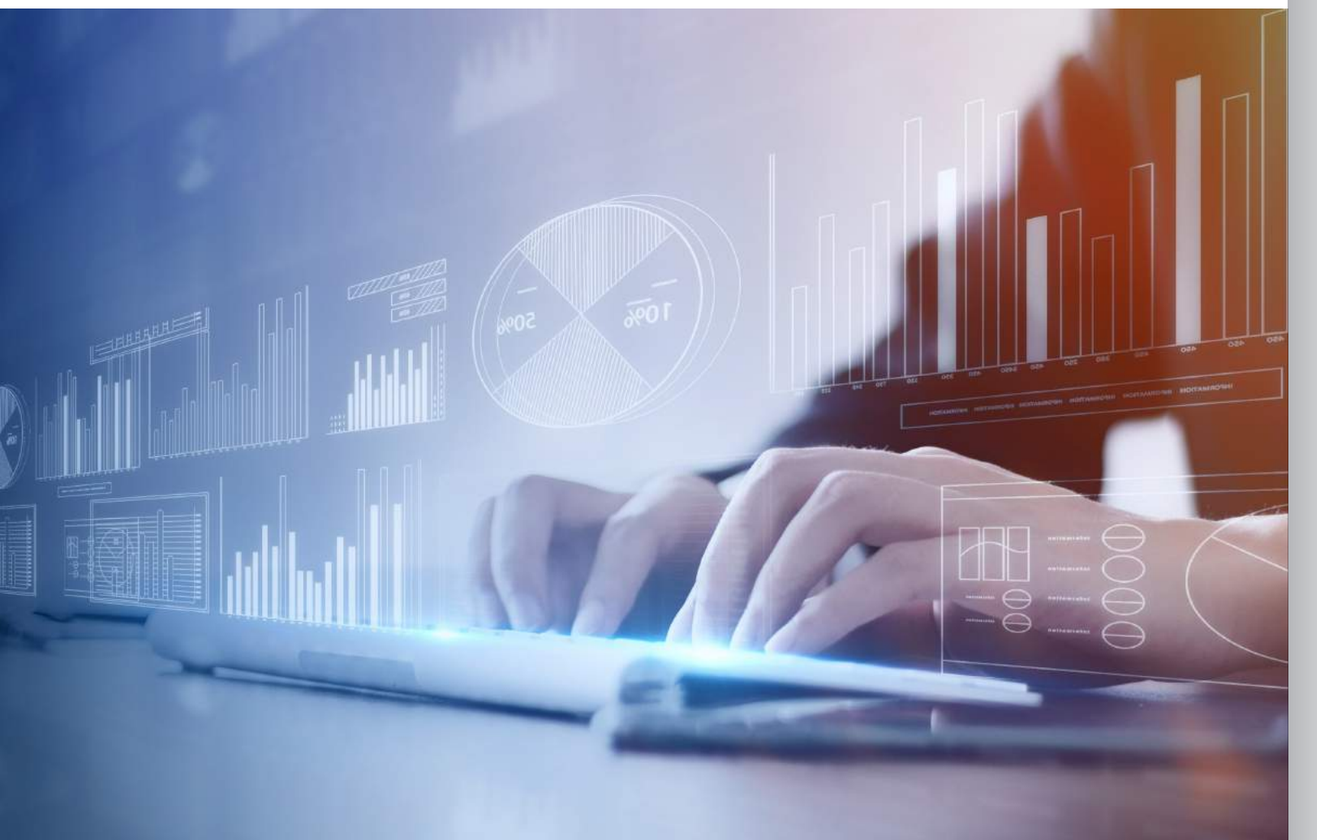
一、数据迁移流程

迁移环境准备 → 创建迁移任务并配置迁移映射关系 → 观察迁移进度 → 游戏应用数据源切换

二、跨数据库实例 SQL 查询流程

在需要进行关联查询的数据库实例白名单中加入DLA的IP白名单 → 再DLA中创建底层映射到数据实例的链接 → 在DLA中创建数据库实例外表 → 通过DLA联合查询多个数据实例的数据

- 如果是进行玩家数据并表的数据迁移，在迁移任务配置界面，需要关闭对目标表是否已有数据的预检查项。
- DLA 只能帮助发现跨服用户数据之间可能存在的冲突，数据调整工作需在数据开始迁移前解决掉，避免数据迁移中遇到主键冲突、唯一键冲突等问题。



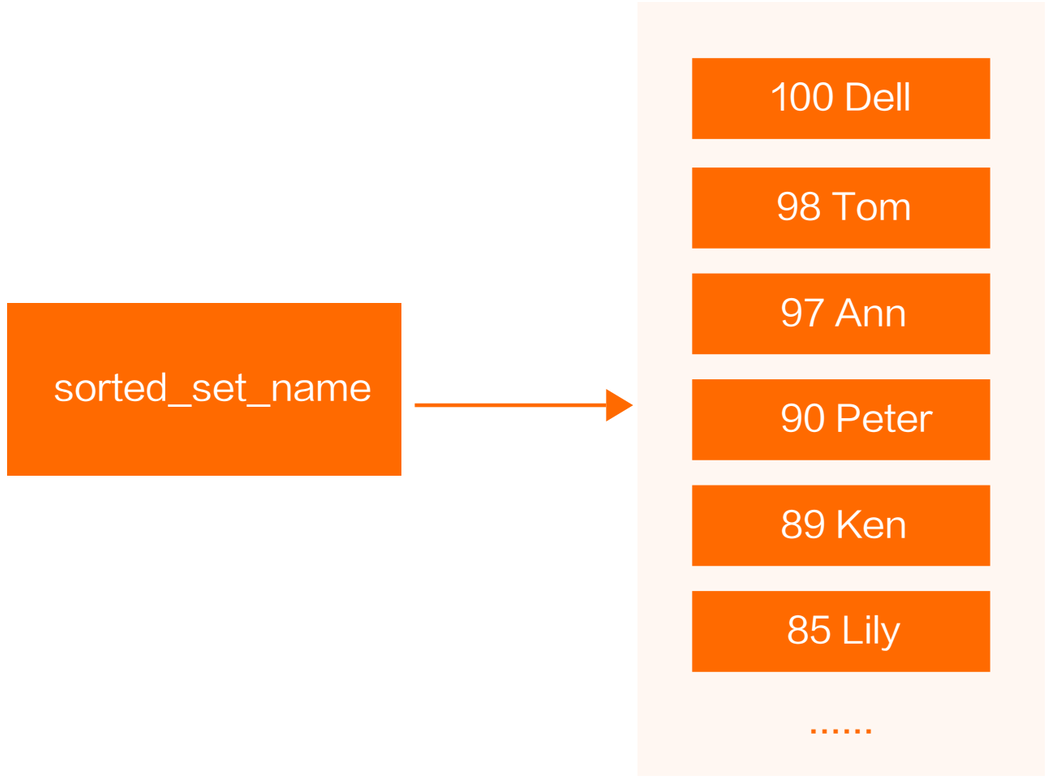
3 游戏排行榜应对方案

① 方案概述

为了鼓励玩家之间互相竞争，游戏中会有排行榜的功能。玩家可根据自己在排行榜上的高排名获得成就感，提升游戏参与积极性。存储排行榜的数据库选型非常重要，如果使用关系型数据库，如 RDS MySQL 来存储用户和积分数据，会有如下几个问题：

- 随着游戏热度逐渐增长，特别是新游戏，积分变化会非常频繁，对数据的 update 更新并发很高，容易导致 RDS 实例过载。
- 积分数据因为实时激励玩家竞争，实时性要求强，玩家查询频率高，每次都要进行排序，在数据量较大情况下会导致 CPU 打满。
- 如果采用只读实例或缓存来缓解 RDS 的排序查询，会面临数据复制延迟和数据一致性的问题。

选用云数据库 Redis 的 Sorted sets 数据类型进行存储，在这个有序集合中，字符串不允许重复，分数可以重复并且是有序存储的。如下图：



本方案具备以下优势：

- 添加、删除和更新玩家积分均有对应接口，对开发非常友好。
- 排行榜数据都在内存存储，可承载高并发更新，且因为数据已经是有序的，查询时延大大优于 RDS 数据库。
- 云数据库 Redis 版提供双机房的同城容灾架构，单机房故障会自动进行切换，保证服务可用性（前提是游戏应用层也要提前做双机房容灾）。
- 云数据库 Redis 版通过在备节点进行 RDB 快照备份的方式，可实现数据的自动、手动持久化保存；且可在控制台使用备份文件进行数据恢复，让排行榜数据具备数据误操作后的快速恢复能力。

2 操作流程与关键步骤

根据要开发的排行榜功能，确认玩家ID和积分字段 → 购买节点“节点类型”为双副本的标准版Redis → 根据Redis标准接口开发排行榜应用

- 云数据库 Redis 版有社区版和企业版两种选择，企业版“性能增强型”单节点可承载 QPS 会是社区版的 3 倍，可根据排行榜数据访问实际压力选择。
- 务必要选择双副本，否则没有高可用保证。
- Java 的积分排行榜代码示例可参见

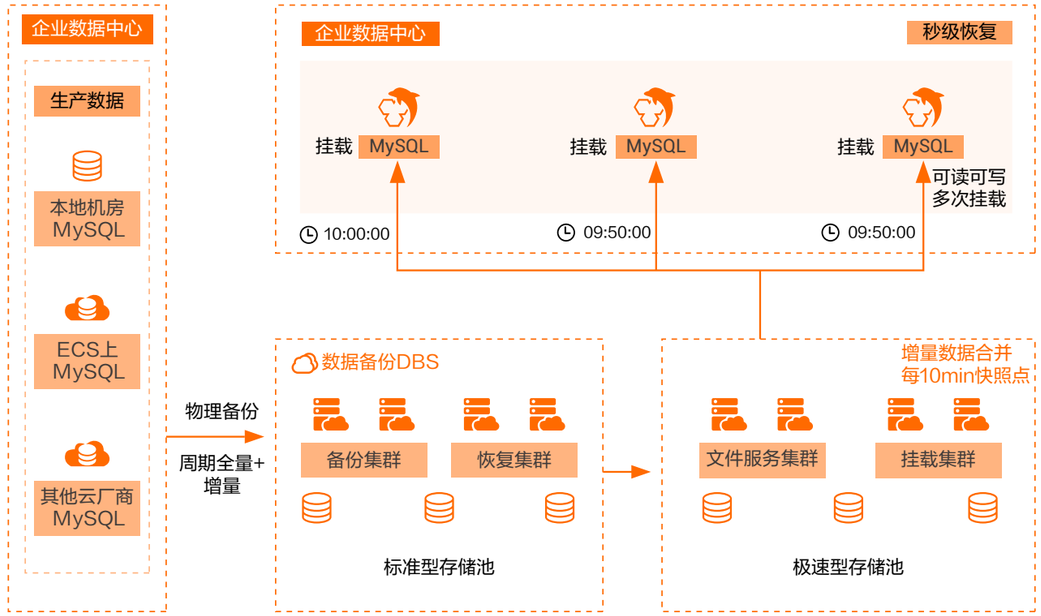
  https://help.aliyun.com/document_detail/26366.html

4 游戏回档应对方案

1 方案概述

网络游戏的回档，给玩家带来的直接感受是最近一段时间内积累的经验值、新获得的道具物品突然消失不见，会给玩家挫败感，带来极坏的游戏体验。但游戏数据的回档依然是有重大价值的，常见于游戏中出现了重大漏洞，有玩家利用漏洞非法复制装备、货币，使游戏公平性遭到破坏。一般数据回档只是将部分玩家数据回滚到之前某个时间点的状态，全部数据回滚的场景极为罕见。

游戏回档会考察 RPO、RTO 两个指标，即希望数据能够回档到过去任意时间点，恢复时间尽可能快。数据库设计架构如下：



本方案具备以下优势：

- 数据库备份 DBS 支持多种环境和多种类型数据库备份。
- 增量日志实时备份，数据可回档到任意时间点。
- 针对游戏回档场景，可选用 DBS 的“极速型存储池”方案，每 10min 生成一份完整的数据快照，需要恢复到任意时间点的增量数据回放秒级完成，数据可直接挂载到 MySQL 实例，然后快速查询出需要回档的完整玩家数据。

② 操作流程与关键步骤

购买DBS备份实例和相应存储

↓

备份任务配置中设置秒级恢复

↓

当需要回档时，选择要恢复的时间点，发起秒级恢复数据任务

↓

获取恢复数据的NFS挂载点

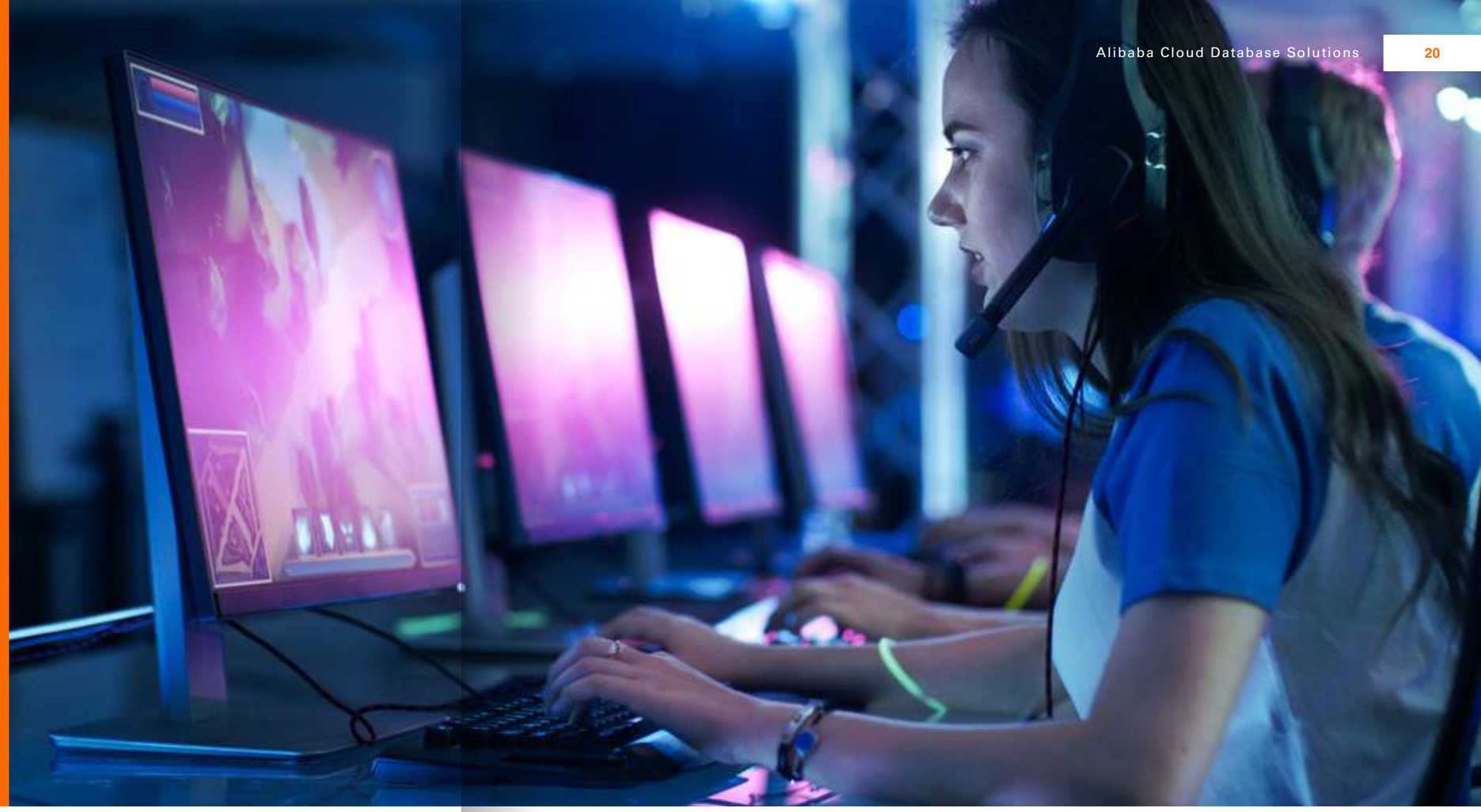
↓

自建立MySQL实例，直接挂在NFS的数据文件目录，启动实例，数据可读可写

↓

查询需要回档用户数据

秒级恢复数据保存的存储池，受限于IO，挂载到 MySQL 实例后，只适合低频的查询，不可作为线上生产库使用。

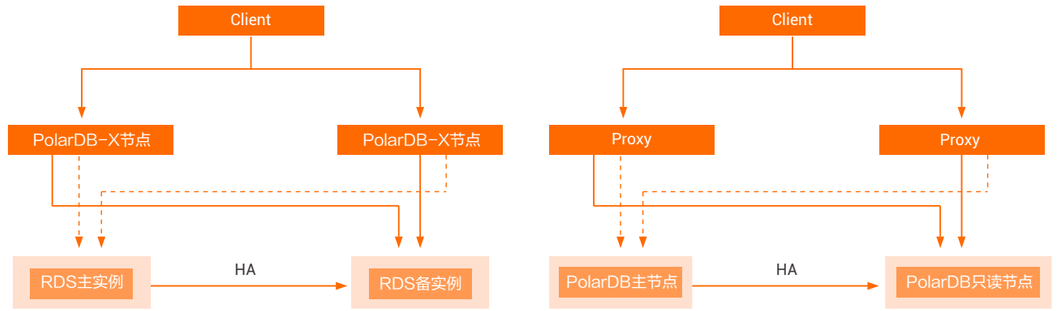


5 掉线问题应对方案

① 方案概述

游戏在研发阶段排期都非常紧，一些容错机制往往不是很健全，比如到数据库的连接保持就是一个典型的例子。一旦到数据库连接发生闪断，由于应用没有很好的连接探活和重试机制，就会导致玩家与数据库失去连接，直接现象就是全服玩家批量掉线，这对游戏运营商是致命的，可能涉及大量玩家补偿甚至导致玩家流失。数据库在运行中发生 HA 切换或变配操作都会导致应用到数据库连接闪断，属于正常运维现象，需要给游戏服务端屏蔽掉这种闪断。

不管是 PolarDB 还是 PolarDB-X 的代理层，均可保持与应用的数据库连接不断，底层发生 HA 切换或变配后，有中间件或代理层重建与数据库实例连接，回放连接会话属性。整个过程应用不会有感知。数据库架构如图所示：



② 操作流程与关键步骤

购买云原生数据库 PolarDB 或 PolarDB-X，无需特殊配置，应用直接连接使用即可具备数据库底层变更防闪断能力。云数据库防闪断能力，仅能保证数据库底层变更不会影响到游戏应用，不影响玩家体验，但应用到数据库的网络存在抖动、闪断的可能性，应用依然需要考虑自己的数据库探活和短线重连机制。

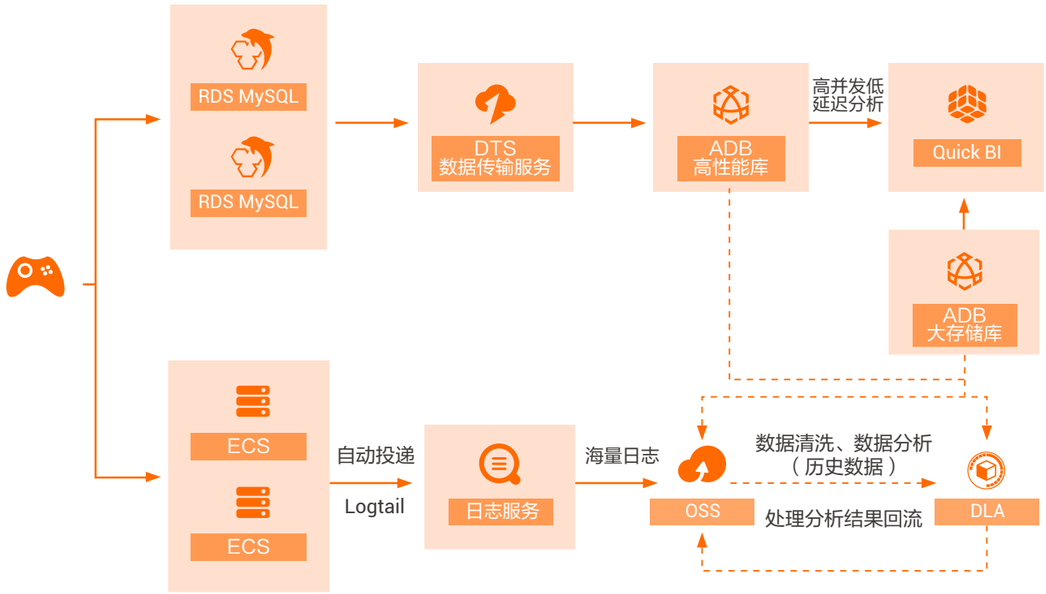
6 游戏买量优化方案

1 方案概述

游戏研发和运营商所关注的成本主要由研发运营成本、推广获客成本和 IT 资源成本三部分构成。其中推广获客成本占一款游戏月流水的 40-50%，通过数据分析技术提供方案帮助客户优化买量的数据质量和降低获客成本的收益巨大。

而广告、买量分析需要数据分析技术，很多游戏客户缺乏大数据技术和人才的储备，短期难以落地，需要易于上手，又易于落地的短平快数据分析方案。

阿里云数据库提供游戏广告实时运营分析平台：



- 秒级实时分析，依托 ADB 预留模式，秒级监控 DAU 等数据，为广告投放效果提供有力在线决策保障。
- 打通了结构化和非结构化数据，广告买量投放效果实时（分钟级）分析，渠道评估更精确。
- 数据湖分析 DLA、ADB 高性能库和大容量库实现了数据冷、温、热三级分层，在充分满足各层数据分析效率同时，有效降低了客户整体数据存储成本。
- DLA 和 ADB 兼容标准 SQL，没有接触过大数据的研发人员可以轻松上手，完成平台开发。

2 操作流程与关键步骤

采用RDS（或PolarDB)实例和ADB实例，用DTS在二者间建立数据同步链路 → 购买日志服务Project和Logstror.将日志数据接入日志服务 → 配置数据投递，验证投递结果 → 购买数据库分析DLA，创建账号，创建OSS外部表，创建schema分别指向ADB高性能库、大容量库，创建ADB外部表 → 用DLA将ADB高性能库和OSS中数据写入到ADB大容量库 → 购买云上报表工具QuickBI，将ADB高性能库和大容量作为其数据源 → 根据具体业务需求，在Quick中创建相应过滤条件，展示表单

详细构建步骤请参见：
https://www.aliyun.com/acts/best-practice/preview?id=572960

7 开放世界类游戏数据库方案

开放世界游戏并非是一个独立的游戏类型，而是近年来越来越受欢迎的一种游戏设计。在之前的网络游戏中，游戏看似宏大，但数据库中并不需要存储庞大的环境数据，因为环境是设定好的，不会因玩家行为而改变，即静态环境，动态玩家。在之前的游戏模式中，数据库仅需要存储玩家信息，相关联的物品、任务信息即可，存储量较小，且研发工程师对单机数据库的能力不会做乐观评估，会通过压缩设计师的数据实体规模来控制数据量，避免数据库成为性能瓶颈。

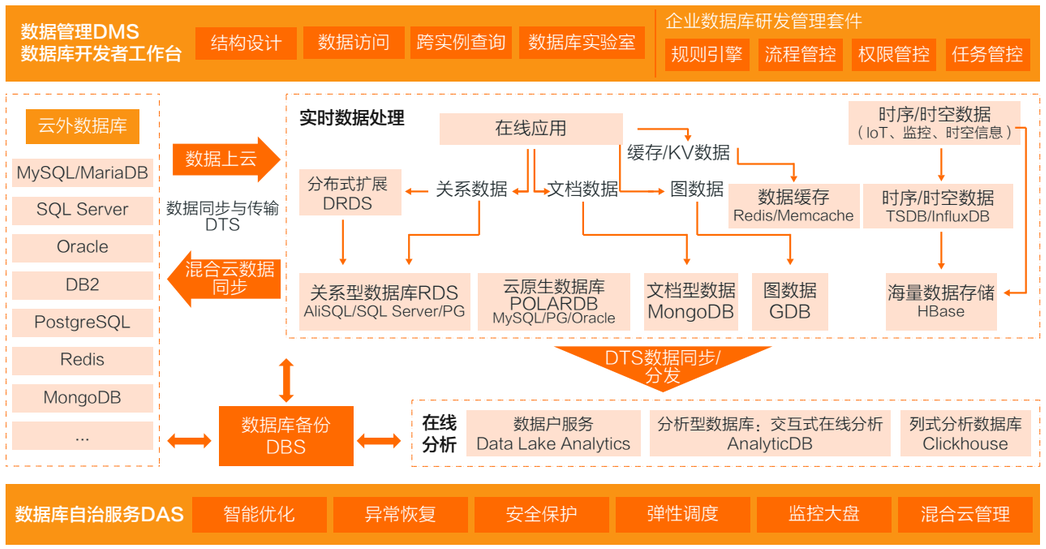
在开放世界游戏中，环境数据是庞大而可变的，留给玩家充分探索和互动的空间。最具代表性的就是大获成功的主机游戏《原神》。这是一个建立在多层系统之上的巨大开放世界，发现不同的系统之间究竟如何交互的过程充满着惊喜和乐趣。

这种开放世界设计扩展到网络游戏中，必然带来海量数据的复杂存储需求：

- 数据库会存储更多的数据，游戏内环境不再是静态的贴图，而会随玩家行为影响而时时发生变化，并需要被记录下来，这就需要弹性数据库如 PolarDB。
- 游戏应用与数据库交互会变的更加频繁，QPS、并发压力都会变大，这就需要云原生分布式数据库 PolarDB-X。
- 游戏空间的扩大，会有接近现实世界的复杂定位场景，可能需要时空数据库存储地理位置。
- 更繁杂的游戏内社交关系可能需要图数据库 GDB 来支持。

方案优势

阿里云云数据库已经形成了完整的技术体系，既有应对在线业务的 OLTP + NoSQL 产品体系，又有对接数据实时分析报表业务的 OLAP 数据库系列。通过 DTS 做为数据总线，可以实现数据顺畅流动；DMS 帮助构建完整数据库研发流程，同时保障内部数据安全；DAS 可实现智能数据库运维管理体系，减轻数据库运维负担。阿里云形成了从数据库内核产品到生态产品的完整闭环，灵活组合，可应对丰富的业务场景。



针对游戏行业不同场景在使用数据库过程中遇到的挑战，阿里云数据库已经有了成熟的应对之策：

- 阿里云数据库产品种类丰富，可结合细分游戏类型，如全球同服游戏，以及具体业务场景，如游戏新开服，设计从缓存到关系型数据库的完整全球数据同步方案，供用户实施。
- 充分利用云上数据库资源弹性的优势，PolarDB 计算存储层分离，可在分钟级别快速升配或增加只读节点；MongoDB 底层存储采用云盘架构，不受单机容量限制，这些都能用来应对游戏新开服等压力场景。
- 针对游戏合服场景，可利用 DLA 跨数据库实例、跨数据库类型对数据进行冲突分析，预先处理冲突数据；而 DTS 可实现数据全量 + 增量平滑迁移，整个迁移过程游戏业务无感知，同时还具备迁移链路高可用、断点续传和数据完整性校验能力。
- 利用云上存储资源优势，构建基于快照原理的备份恢复体系（可选服务），可在分钟级拉起一份历史数据快照，供客户快速查询数据，进行定向数据回档恢复。

客户案例

1 心动网络

① 客户感言

“PolarDB 提供高性能读写能力，100% 兼容 MySQL，使得业务可以无缝迁移，支持 100 万级玩家同时在线，以及游戏服务端软 / 硬件故障导致服务端重启时业务的快速恢复。”

② 案例背景

心动网络，国内极具知名度的游戏公司，中国互联网百强企业。旗下业务涉及游戏研发运营、动画制作、偶像娱乐等多个产业。2009 年起，公司开始打造心动网络的自主品牌，致力于网页及移动客户端游戏的研发与运营。

- 为支持游戏业务快速出海，游戏发行和 TAPTAP 游戏社区全球化运营与交付，需要支撑全球化业务的统一部署。特别是东南亚人口密度较大的地区需要提供低延迟、高稳定的云服务。
- 客户服务于国内、东南亚和欧美等地人口大国时，在活动峰值时需要支撑 100 万级玩家同时在线的高并发压力。
- 游戏运维发布、游戏服务端软 / 硬件故障导致服务端重启，需要数据库支撑更快的数据读取能力，以实现业务的快速恢复。

③ 业务架构

心动网络采用 PolarDB 分布式云原生数据库方案构建了全部业务系统：

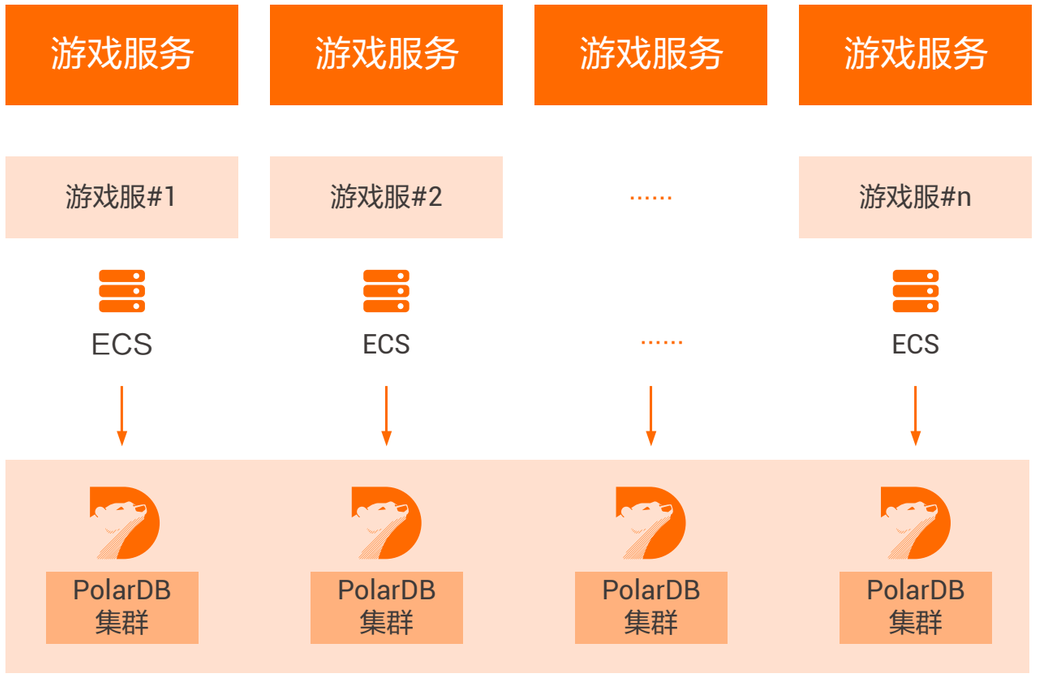
- PolarDB 支持处理海量大数据，同时具备高并发、高可用和很强的弹性伸缩能力；
- PolarDB 特有的高性能读能力，有效支撑游戏服务层的因变更、异常导致的服务端重启业务快速恢复。

④ 业务效果

PolarDB 为千万级用户在线手游保驾护航。

优良的游戏体验

基于存储计算分离架构，所有实例都带有一主一只读节点，提供 3 倍于 MySQL 的性能。基于高性能的读写能力，便于新开服以及应用弹性扩容。在游戏版本发布、服务端重启等场景可以大大缩短维护时间，又确保在极端情况下，不会因为游戏服务端软、硬件故障造成服务恢复过慢而影响玩家体验。



7*24 高可用服务

数据采用三副本一致性存储，很好地保证了数据的可靠性。同时在主实例发生故障的时候，系统能在短时间（30s-60s）内完成快速切换，确保在线业务能够在保证数据完整性的同时快速恢复以提供正常的服务。

丰富的业务支持

100% 兼容 MySQL 5.6 和 MySQL 8.0，完全兼容 MySQL 各种生态和中间件开源工具，非常方便游戏历史战绩、游戏充值和运营活动的开展。

2 某游戏公司

1 客户感言

“ADB 超高性价比助力我们高效地、快速地打造新一代游戏发行实时数据运营平台。”— 某游戏公司首席架构师奕风

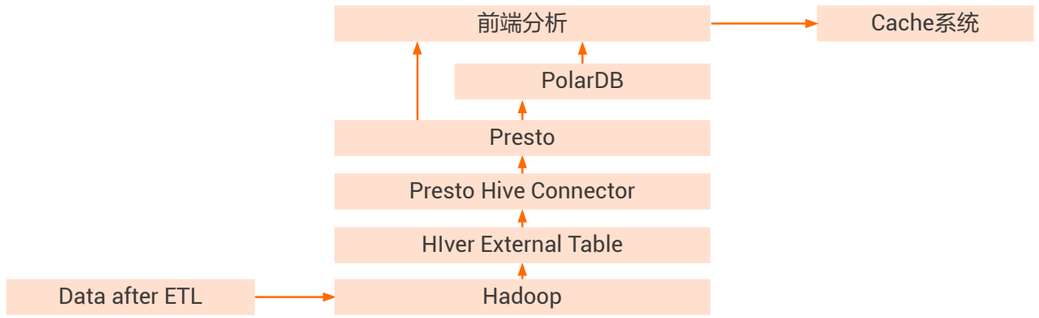
2 案例背景

该客户是一家独特且创新的游戏公司，目前在全球已有超过 1 亿的手机游戏用户。

游戏领域的竞争非常激烈，在互联网高速增长的同时，流量成本不断升高，市场营销开始往精细化发展。

- 流量越来越贵，考验买量更加精准快速调整策略；
- 用户更加成熟，游戏品质要求越来越高；
- 市场竞争愈演愈烈，好玩也需要懂得市场运营。

从而业务上要求游戏运营平台能够做到 3 点要求：精细化运营，效果实时反馈，抢先一步预测。针对上述业务挑战，客户广告大数据分析部门搭建了以 Hadoop 体系为生态核心的大数据架构，用于解决实时分析问题，架构图如下：



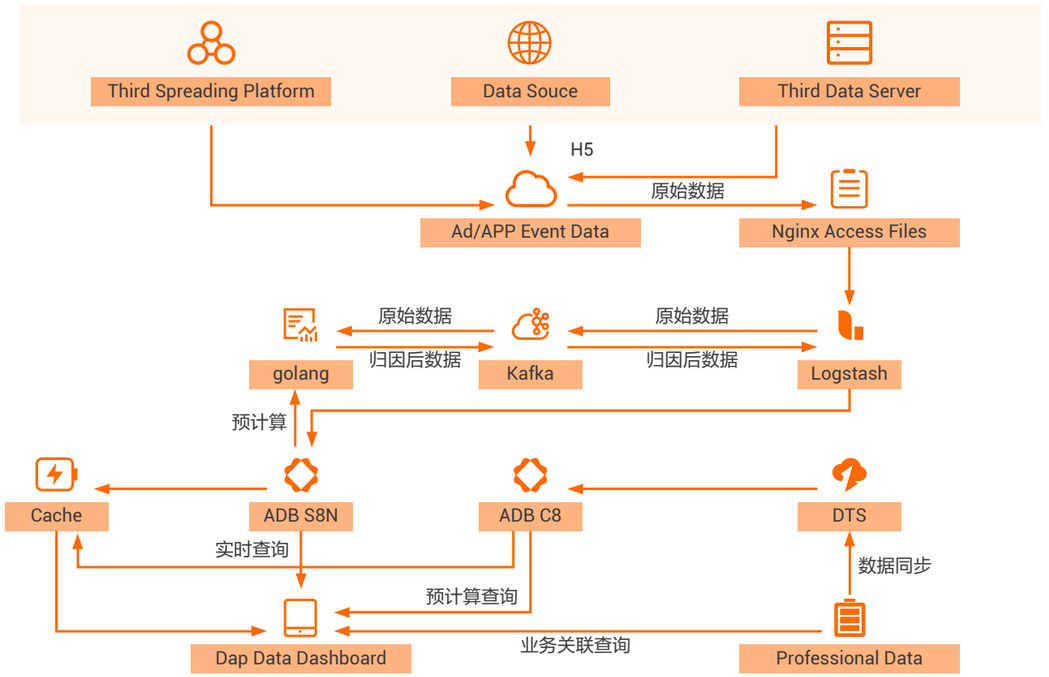
然而采用上述架构和数据流后，随着数据量的迅猛增长以及业务的要求越来越高，存在如下问题：

- 扩展性问题：自建集群的容量拓展，吞吐量、算力提升，没有那么便捷。
- 易用性问题：自己维护 hadoop-hive-presto，有不小的学习与维护成本。ORC 等列存储方案，作为离线分析很不错，但是实时性的方案并不容易实现。
- 实时性问题：数据分析的实时性还不够高，presto 作为直接查询的实时计算引擎，性能不达预期。
- 性价比问题：自建集群，为了保证一定的性能与稳定性的费用花销越来越高。

3 业务架构

针对上述业务挑战和架构遇到的痛点，客户大数据团队开始尝试针对架构和产品选型进行考察，先后考察和尝试了 Kylin 和 Druid 产品，由于 Kylin 本质是预处理，并非实时，查询模式也比较固定，无法满足业务要求。在使用 Druid 时也面临 SQL 支持不好、关联性查询不方便以及不支持查明细等等问题，最终与数据库架构师团队一起考察和 POC 了 AnalyticDB for MySQL 产品，体验到了 AnalyticDB 的快、灵活、易用、规模扩展和高并发的优点特性，成功地打造了围绕 AnalyticDB for MySQL 为核心的新一代游戏广告实时运营分析平台。

DAP 数据架构图 2.0



4 业务效果

通过采用“PolarDB + ADB 大存储 + ADB 高性能”产品组合打造出新一代游戏买量市场实时数据运营分析平台，云原生数据处理、分析闭环，实现了高效的游戏数据运营，助力客户手游市场雄踞海外和国内。

- 分析数据的实时性提升帮助用户更好地挖掘数据蕴含的价值，通过对数据的分析更好地指导业务开展。在构建好新一代平台后，分析性能产生了 5-10 倍的性能提升，极大的提升了业务体验，促进了买量市场的投放效率转化；
- 基于玩家行为日志表日益增长，日增长过亿数据量，通过 ADB 存储密集型实例进行存储和分析，有效地降低的客户的总体使用成本，总成本下降高达 300%；
- 支持该游戏公司从海外市场向国内市场进军，客户作为海外手游市场的佼佼者，目前正在筹划向国内进军，海外和国内玩家市场、买量市场均有各种的特点和差异，有了 DAP2.0 平台，该游戏公司才有信心在国内市场继续所向披靡。

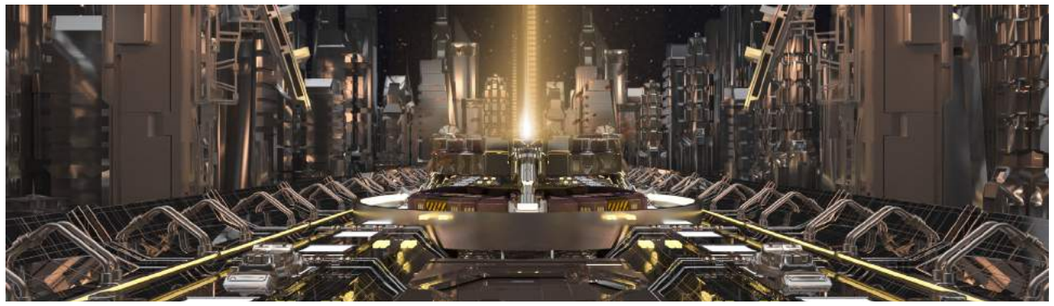
3 某游戏公司

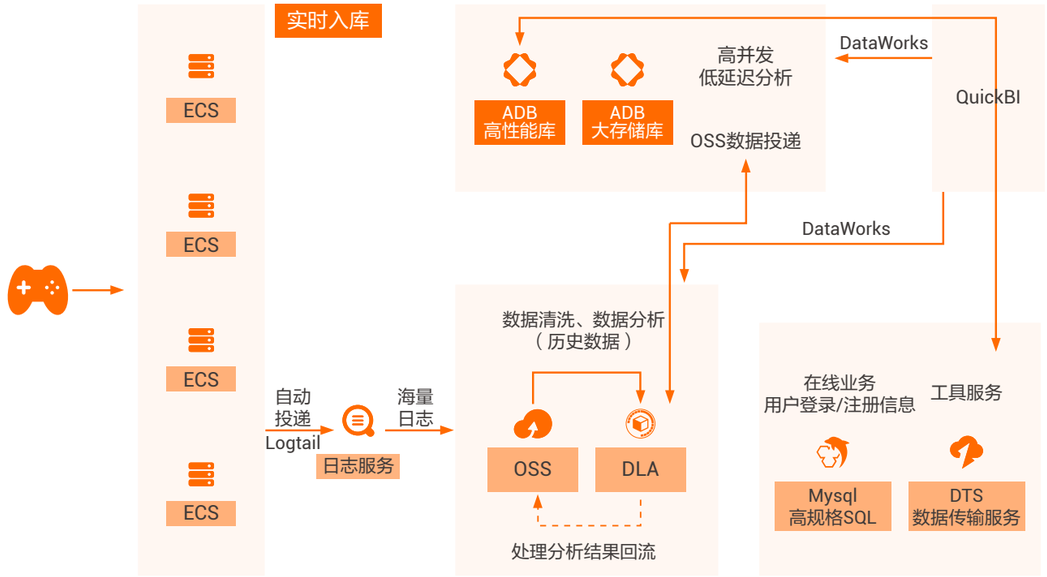
1 案例背景

专业的游戏制作团队、先进的游戏理念以及阿里云的技术支持成为这家游戏创业公司成功逆袭的神器。

公司在发展过程中面临以下业务需求：

- 提供全面的游戏运营指标分析功能：全面提高游戏开发者的日常数据运营工作效率，不仅提供付费用户、付费率、付费金额和 ARPU 等运营指标，还强化了付费用户的留存率、回访率、用户生命周期价值等更加精细化的运营指标，游戏开发者可以更加深入，更加有效率的掌握游戏运营状态。
- 提供有效分析渠道效果，使每分钱都花在刀刃上：实时的分渠道数据统计可以监测到不同渠道用户的增长、活跃、留存状况以及充值状况，更加全面、快速的分析出投资回报率，让开发者对渠道的评估更加准确。
- 对付费用户追踪分析，了解付费用户的习惯：针对付费用户群，通过简单易懂的数据分析模型和图表，跟踪付费用户的留存、流失、回访和充值数据，更好的反映付费用户在整个生命周期的关键行为和价值。
- 细致分析玩家游戏行为，改进产品体验，提高游戏收益：关卡、道具、消费行为分析的功能可以了解道具和物品在使用过程中使用和消耗的总量以及趋势，开发者可以借此来做到恰到好处的数值平衡设计，也可充分利用数据分析的结果来帮助开发者优化游戏内付费商品的收益。





“ADB高性能+ADB大存储+DLA”形成数据分析闭环

ADB高性能库:
（RDS-DTS）玩家登录/注册实时监控，实现展现新增玩家&在线玩家信息（RDS独占10min->ADB3s）

ADB大存储库:
(日志采集投递)玩家行为分析，圈选高价值玩家，通过打点信息分析指导游戏研发优化(百毫秒RT)

DLA数据湖:
（OSS数据投递)将OSS日志数据“T+1”投递到ADB大存储库进行海量热数据分析，日投递量1TB+
（OSS数据分析)将OSS日志数据(含玩家行为历史、API旧志)进行清洗加工，日扫描量5TB+

2 业务架构

构建了以“ADB+DLA”为融合数据仓库分析、以“Redis + RDS”为实时高并发在线业务、以“DTS+DMS”为平滑工具服务的数据库架构。

3 业务效果

- 针对玩家新增设备、新增玩家的实时监控场景，从 RDS 切换到 ADB 后，响应时间从 10 分钟下降至 3 秒钟。
- 基于玩家登陆注册信息单表记录数已经高达数十亿，单表数据量已经高达 TB 级别，通过 ADB 高性能实例进行秒级（<3 秒）监控 DAU 以及玩家活跃状态等数据，为游戏广告投放效果提供了有力的在线决策支撑；
- 基于玩家行为日志表记录数已经高达数百亿，单表数据量已经高达百 TB 级别，通过 ADB 存储密集型实例进行细致分析玩家游戏行为，有效地支撑了对产品改进体验的分析，提高游戏收益；
- 基于 ADB+DLA 进行海量历史数据对比分析，针对游戏广告买量投放效果进行实时分析，实现了每 5 分钟刷新一次投放效果展现，更加全面、快速的分析出投资回报率，使得对渠道的评估更加准确高效。

附录

术语表

ARPU：每用户平均收入，是一个时间段内运营商从每个用户所得到的收入，一般是按照月来计算。

合服：游戏分区分服架构下，当游戏进入平稳期后，不同服务器（可能是逻辑概念）上玩家会减少，出于资源利用率考虑，将不同服务器玩家合并到一个服务器，同时也让单位服务器内玩家密度变大，带来更好玩家间交互体验。

回档：由于游戏 bug 或某些因素导致游戏数据损坏，需要部分或全部恢复到之前正常情况下某个时间点的数据状态，玩家的感受会是最近达到的装备不见了，完成的任务显示尚未完成等等。

买量：游戏的获客行为，一般是在各大渠道发布广告，吸引玩家下载游戏，进行游玩，完成转化，通过在渠道花钱投放广告获得玩家客户，所以叫买量。

数据并表：就是将原来位于不同数据表中数据汇总存储到一张数据表中，这些数据表可能位于不同的数据库实例。

RPO：恢复点目标，在故障发生后，数据能够恢复到的时间点，也反映着业务所能容忍的数据丢失量。

RTO：恢复时间目标，在故障发生后，数据恢复到预期状态所能承受的最长时间。

HA 切换：数据库的高可用特性，比如数据库是主备架构，平时只有主实例对外提供服务，备实例从主实例同步数据，一旦主实例故障，HA 切换后备实例代替主实例对外提供服务，保证数据库服务连续性。

DAU：日活跃用户数。

2 在线教育行业
数据库解决方案

中国产业研究院市场调研结果显示，从 2013 年到 2019 年，中国在线教育市场规模每年环比增长 50%。2019 年，在线教育市场规模达到 3670 亿人民币。在线教育行业发展非常迅猛。其中，中小学在线教育占比每年都在增加，在线教育用户数从 2019 年的 2.3 亿，到 2020 年 Q1 已经突破 4.23 亿，K12（学前教育到大学教育之间群体）成为目前最具前景的教育市场。基于艾瑞咨询 2019 年初的统计，目前在线教育在整个教育市场中的占比不足 10%。相对于其他互联网行业细分领域，在线教育仍然存在巨大的增长空间。在业务快速扩张的情况下，在线课堂的顺利开展，教学质量的稳定，以及教学数据的实时分析和运营能力的提升，都离不开底层数据库的保障。本解决方案基于阿里云产品组合，针对 K12 在线教育的上课、考试和运营场景，提供灵活可靠的数据库解决方案。

在线教育行业对数据库的需求

面对快速增长的在线教育市场，在线教育类企业面临着诸多问题和挑战。例如，数据的爆发式增长、在线人数的剧烈增加、以及越来越多样化的线上即时互动场景。同时，在线课堂一旦故障，影响面大且企业往往需要赔付用户损失，数据容灾和多活功能必不可少。从在线教育的业务角度来看，对数据库有以下需求：

1 运营业务

在线教育十分看重营销获客活动，获客前的活动信息和获客后的集中注册和下单操作都是对数据库的巨大冲击。开始上课后，家长都希望能第一时间看到学生上课的各项统计指标。为了让家长放心，需要支撑并提升学生上课数据实时分析和运营能力。

2 课堂业务

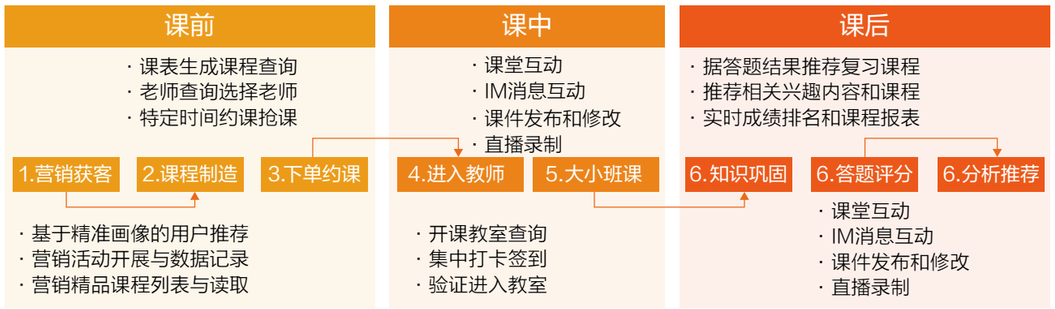
在线课堂经常带来高并发读写请求，数据库需要提高在应对课堂峰值的高并发读写请求能力。同时，在名师、名课效应越来越明显的背景下，需要保障最为核心的上课场景下数据的容灾和多活。在全球教全球学场景下，数据库还需要满足全球数据存取的需求。

3 考试业务

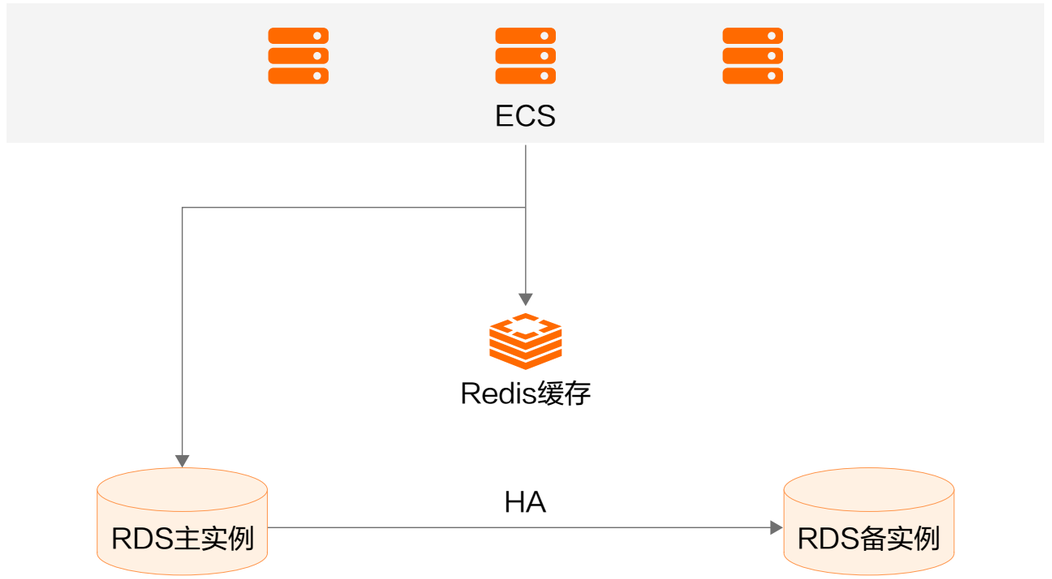
在线教育经常发生动辄数万人同时涌入的考试场景，需要保障题库试卷快速生成，以及提交后的即时判分，数据库将面临频繁的数据请求。同时，为了实现丰富的互动和教育手段，使用数据库设计系统时需要考虑业务需求的不断增长。

K12 教育行业细分场景与解决方案

教育行业从年龄维度可以划分为学前教育、K12 教育、大学教育、成人和职业教育，K12 教育是最核心也是最具活力的群体。整个线上教育围绕在线课堂展开，从时间维度可以划分为课前、课中、课后这三个部分。K12 教育的完整流程图如下图所示，整个流程场景非常多，这里只列举与数据库操作相关的场景，而每个场景对数据库的依赖和需求也截然不同。



1 营销获客解决方案



1 方案背景

典型的营销获客活动场景包括发送名师讲课、精品课程、特价课程给家长，吸引新客注册老客购买。活动发送覆盖的用户比较多，而且通常会集中发送。在获客场景中，发送记录批量落库、同一时间大量用户查询活动内容、新客集中注册、新老客户下单，里面每一个环节都对数据库带来冲击。

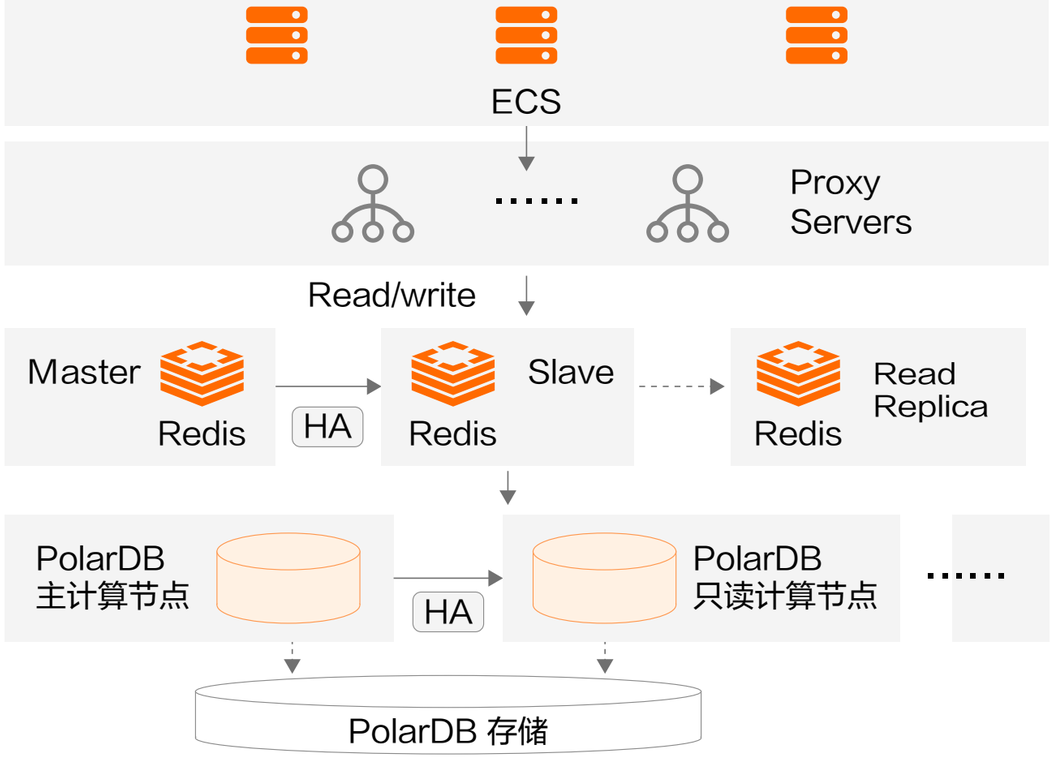
- 在活动发送和用户参与、提交活动信息时，数据库需要具备高 IOPS 写入能力。
- 活动统一推送后，用户请求比较集中，数据库需要具备高并发支持能力。
- 活动请求页内容一般放在缓存里，对缓存的超大连接数和 QPS 有很大挑战。

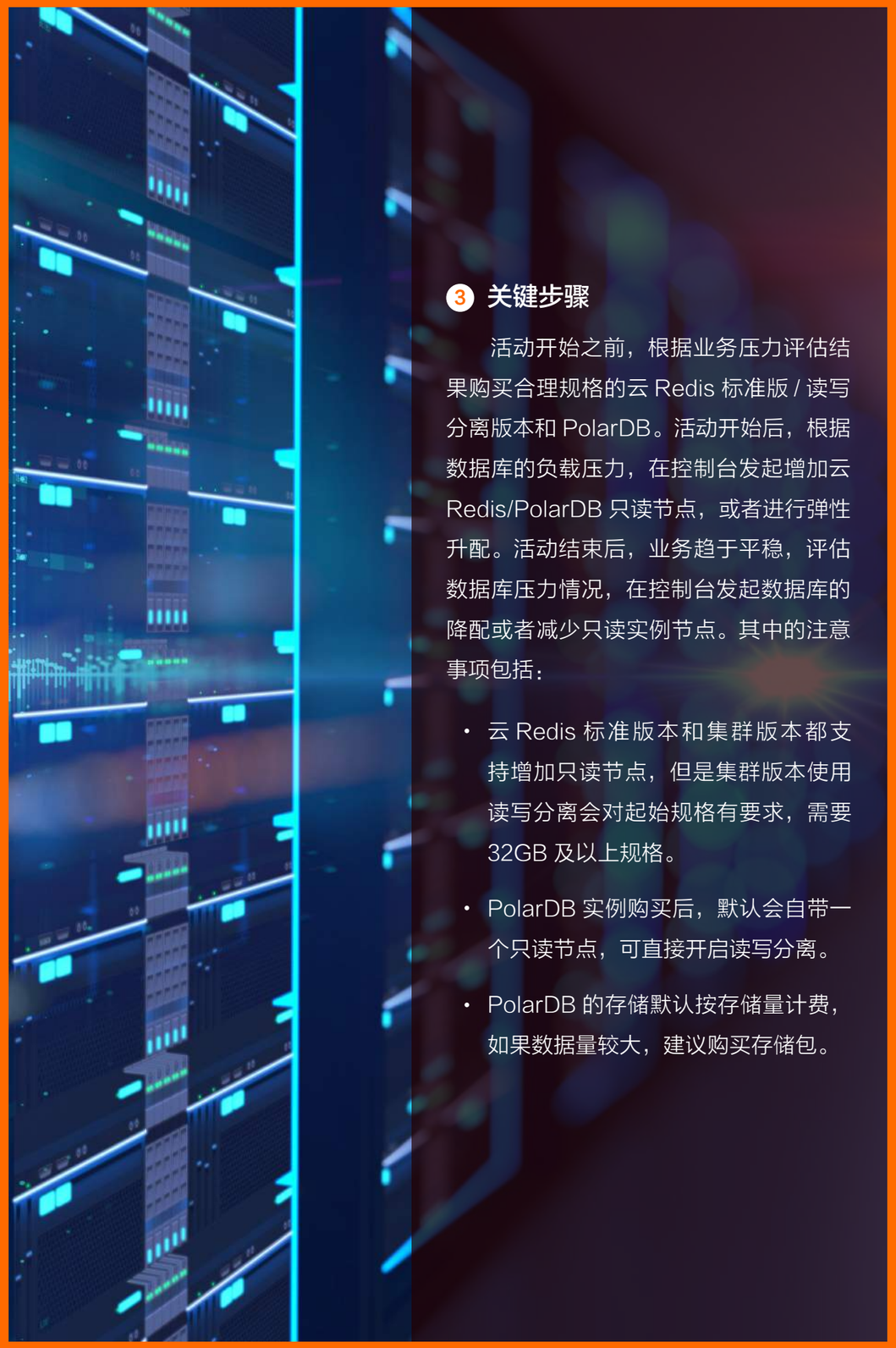
2 方案架构

营销活动内容一般固定后不会实时修改，内容适合放在缓存中。同一时间大量用户访问活动内容，会表现为热点 KEY 伴随高并发场景；用户一旦感兴趣参与活动后，即为获客成功，用户会访问并跳转到具体课程或者活动内容，填写资料提交信息，此新用户信息对客户业务非常关键，需要落库存储，形成对关系型数据库的高并发读写。常用的典型架构如上图所示：

- 在线教育营销活动请求数据量小，查询高，非常适合用 Redis 缓存，阿里云 Redis 支持透明读写分离，活动一旦玩法确定，业务基本不会频繁变动，不用担心只读节点数据读一致性问题。
- 阿里云 Redis 企业版（Tair）的多线程性能增强模式，可以极大缓解突发大量请求导致的连接问题，高并发下运行更稳定。
- 使用云 Redis 读写分离架构能很好的解决营销场景下，数十万人同时请求同一活动内容，形成的热点 KEY 场景。
- 当营销获客成功，目标用户确定参与活动，此时会跳转请求课程或者活动内容，大量用户集中提交信息，然后查询，形成高并发读写数据库场景。
- PolarDB 具备计算存储分离架构配合透明读写分离，主实例在应对高并发、高 IOPS 写入会表现的很好，同时读计算节点能分担大量读请求。
- 一旦营销获客活动反应较好流量剧增，得益于PolarDB的存储计算分离架构，能在最多 5Min 就增加只读计算节点来分担更多读请求。如果主实例写压力也很大，同样在 5Min 就快速实现规格快速弹性升配，不需要类似 RDS 或者自建MySQL一样拷贝底层数据。

综上考虑，推荐优选方案如下图所示：





3 关键步骤

活动开始之前，根据业务压力评估结果购买合理规格的云 Redis 标准版 / 读写分离版本和 PolarDB。活动开始后，根据数据库的负载压力，在控制台发起增加云 Redis/PolarDB 只读节点，或者进行弹性升配。活动结束后，业务趋于平稳，评估数据库压力情况，在控制台发起数据库的降配或者减少只读实例节点。其中的注意事项包括：

- 云 Redis 标准版本和集群版本都支持增加只读节点，但是集群版本使用读写分离会对起始规格有要求，需要 32GB 及以上规格。
- PolarDB 实例购买后，默认会自带一个只读节点，可直接开启读写分离。
- PolarDB 的存储默认按存储量计费，如果数据量较大，建议购买存储包。

2 课程查询解决方案

1 方案背景

老客户或者新获客户都会频繁的进行课程查询或者名师查询，以筛选出最适合孩子的课程和老师，这些课程列表会分别存储在缓存和落在数据库中。由于课程查询属于在线上课之前用户必然会进行的操作，而且频度很高，所以是典型的高并发、高 QPS 的场景。

2 方案架构

课程查询内容固定，如果直查数据库，必然需要极大的资源来支撑查询请求，所以推荐放在缓存中，来为后端关系型数据库抗住流量冲击；因为键值分布相对均衡，适合使用集群模式。阿里云 Redis 集群具备百万 +QPS 的能力，且扩展方便，统一代理层能实现对业务透明，方便业务快速迭代。一旦课程内容变更，业务更新数据库，同时刷新缓存，可以保障缓存、数据库的数据一致性。架构图可参见 1.1.2

3 关键步骤

缓存推荐使用阿里云 Redis 企业版（Tair）。Tair 是多线程版本，在超大连接数和高并发情况下更能发挥多线程的优势，特别是在极端情况下也能保持稳定，可以避免缓存击穿，流量打到数据库引起业务雪崩等场景。

一般课程列表、名师介绍等等内容生成 KEY 对应的 Vaule 可能较大，高 QPS 情况下对流量吞吐要求较高，Redis 企业版 2 倍于社区版，可以避免流量吞吐负载高导致网卡排队等情况。

底层关系型数据库采用 RDS 读写分离架构，在课程查询场景中，读写比往往可能是 N:1，读占据绝大部分，增加只读实例能快速扩展读能力，而较少的 DML 操作也不用担心只读延迟问题。

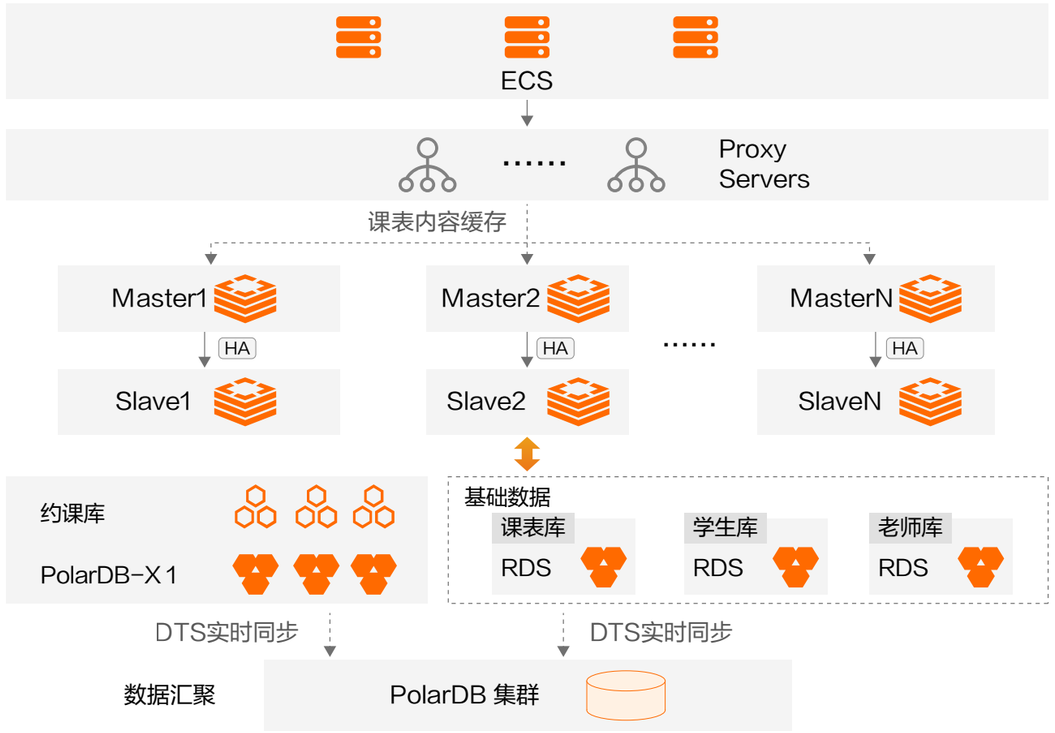
3 约课报名解决方案

1 方案背景

由于在线教育不再依赖传统教育方式的线下教室大小或者出行成本的限制，家长和学生很容易就能直接和名师或者明星课程学习，造成名师和明星课程一推出就疯抢的局面，导致约课报名场景的高并发读写需求。对于一些头部在线教育客户来说，不管是一对一模式还是大小班课模式，约课场景都是压力最大的场景。诸如周二的约课开放日，上百万的课程预约会在很短的时间内高并发完成。此外，大量的约课数据需要保留在数据库中，所以这也是一个海量的数据存储场景，同样需求底层数据能够切片存储，分散数据，降低单库存储数据，保证查询性能。综合来讲，约课报名是高并发、高吞吐结合海量存储的场景。

2 方案架构

针对约课场景高并发读写、海量数据存储的需求，且主线都是围绕学生和老师这两个维度展开的场景，非常需要和很适合分布式数据库这样同时具备 scale-up 和 scale-out 能力的架构来承载。分布式数据库具备很好的扩展读、写和并发的能力，多个节点分散承载读写压力，多个底层数据节点切片存储，数据打散。当然，一旦采用分布式数据库之后，数据的完整统计分析、业务的关联查询会造成不便利性，对业务设计有一定要求。因此，成熟的方案是形成汇聚库，将前端多页业务数据汇聚到一个“仓库”中，将这些操作统一放在汇聚仓库中进行，能极大便利业务查询需求和性能。针对约课场景高并发读写、海量数据存储的需求，且主线都是围绕学生和老师这两个维度展开的场景，非常需要和很适合分布式数据库这样同时具备 scale-up 和 scale-out 能力的架构来承载。分布式数据库具备很好的扩展读、写和并发的能力，多个节点分散承载读写压力，多个底层数据节点切片存储，数据打散。当然，一旦采用分布式数据库之后，数据的完整统计分析、业务的关联查询会造成不便利性，对业务设计有一定要求。因此，成熟的方案是形成汇聚库，将前端多页业务数据汇聚到一个“仓库”中，将这些操作统一放在汇聚仓库中进行，能极大便利业务查询需求和性能。



3 关键步骤

- 使用缓存数据库来承载读流量，降低关系型数据库的负载，提升读取响应时间。通过缓存失效或者订阅底层 binlog 的方式来实现缓存和数据库存储之间的一致性。
- 采用分布式数据库 PolarDB-X 来承载数据，按照学生或者老师维度进行数据拆分，PolarDB-X 的前端节点能承载高并发和海量连接请求，SQL 经过 PolarDB-X 解析后，并发下放到底层数据库物理节点进行读写。
- PolarDB-X 的数据路由能力，底层物理节点数据切片存储，也即分库分表能力可以保障维持单库在一个推荐以内的合理范围（如 500GB），也可以保障单表在推荐的 500W 行以内存储的性能最佳。
- 一个或者多个 PolarDB-X 分布式数据库的数据可以通过 DTS 数据同步工具实时同步到 PolarDB 集群中。PolarDB 的存储计算分离架构，能提供超高 IOPS，在汇聚后并发大表查询中性能保障优异。这样不仅解决数据拆分后的聚合问题，同时会 N 对 1 的形成聚合查询库，业务直接在一个库中做业务查询或者关联，避免跨库等复杂需求带来开发成本。

4 上课场景解决方案

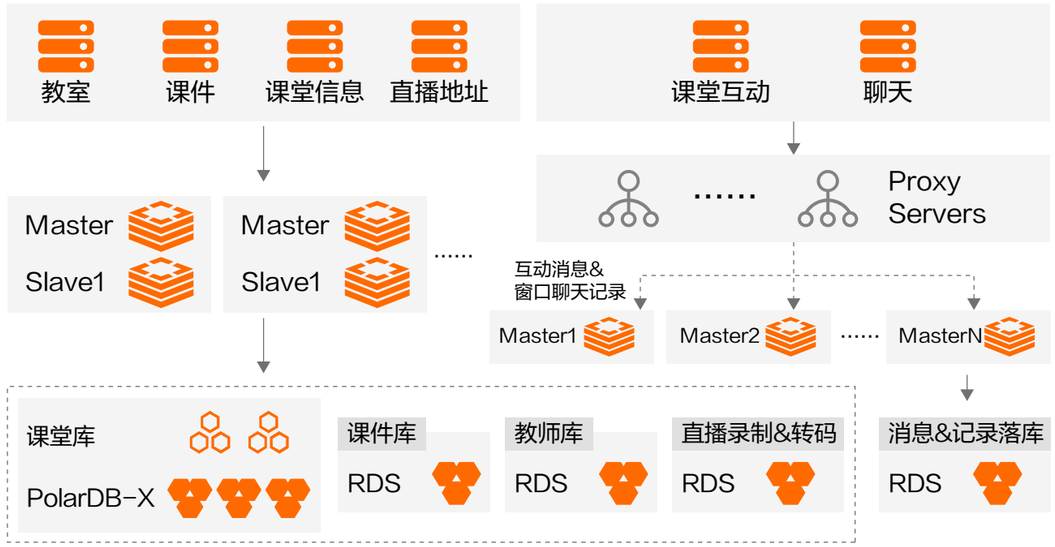
1 方案背景

在学生家长购买课程并且成功预约后，K12 在线教育中围绕课前的动作基本完成，将进入上课环节。具体场景为，学生在指定时间打开 APP 或者电脑客户端，点击“进入教室”，开始既定课程学习。K12 在线教育的上课形式，可以分为三类：一对一模式、小班课（1V4/6/8）、大班课（最大可达万人）。目前在线教育的整体发展趋势是越来越多的向大班课模式靠拢。学生点击进入教室，需要给学生初始化教室，查询当前教室元数据，生成教室背景（部分结合 AI 智能教室画布等能力），列出当前教室课程重点等信息，打卡签到等。同时一些重要的节点数据，比如进入教室、开始上课、课程问答、课堂每小节打卡等内容，还需要有数据记录，在课堂中或者课堂结束后生成课堂报表，来给到家长端查看，让家长能实时或者汇总看到孩子的每堂课学习完成率和精彩问答等，这也是各在线教育机构非常看中的一环，让家长能很轻松的就实时掌握学生的学习状态，增加家长体验和购买意愿。

- 课堂开课时间固定，大量学生会在固定时间同时进入教室，形成高并发场景，尤其是一些名师开课和明星课程，对数据库的并发承载能力是很大的考验。
- 课程可能持续几个月甚至年许，课程记录需要保留，在课程轨迹等业务场景中使用，形成海量课堂记录存储，对数据库带来存储压力。
- 上课作为在线教育最核心的场景，对容灾能力要求非常高，学生如果无法按时进入教室开课，老师如果无法正常开始讲课，随着时间的推迟，会带来非常大的经济损失，这对数据库高可用、容灾双活提出要求。
- 在课堂互动场景中，学生和老师的互动、问答等，都会以消息的形式发送接收，对时效性和不丢消息要求非常高。越来越多的客户使用 Redis 作为内容保留或者直接使用 PUB/SUB 方式实现，对 Redis 的流量带宽和热 key 性能带来很大的挑战。
- 在大班课场景，聊天窗口聊天信息，一般保留在 Redis 缓存及时存取。为了保障聊天信息的实时刷新且不丢失，业务往往不断的请求 Redis 的某一个关联了当前课堂聊天的 KEY，导致热点 KEY 和流量带宽打满等情况。

② 方案架构

在一对一场景中，可能每小时都有数万节课在开始和进行，所以对数据库的压力更多的在于大量课堂信息的查询和存储在数据库中音视频（比如视频通道 URL）信息的读取。而在课堂较少的大班课模式中，由于价格亲民、覆盖面广，同时进入教室的学生人数可能达到几十万的规模，对教室等数据库热点数据的读取和签到等数据写入都会带来瞬间峰值，并且伴随着平均 20-40 分钟课 堂的开始和结束，数据库伴随有明显的波峰波谷。



③ 关键步骤

- 采用 PolarDB-X 分布式数据库，按照比如课堂维度分布式存储，在约课阶段分配好课堂后，上课可按课程维度查询。PolarDB-X 的分布式架构能够横向扩展，轻松面对高并发场景。
- PolarDB-X 分布式数据库底层分片存储，分库分表可以非常有效的解决海量课堂记录的存储压力，避免大库大表拖累性能。
- PolarDB-X、阿里云 Redis、阿里云 RDS 都具备数据库高可用能力，天然接入阿里云数据传输服务 DTS，能够方便、成熟的实现跨机房、跨 Region 容灾和双活架构，阿里云数据库团队能协助用户一起建立容灾双活能力。
- 阿里云 Redis 企业版（Tair）在 PUB/SUB 场景中已经实现集群方式打散请求，解决传统 Redis 在 PUB/SUB 场景只能使用集群节点 1 的困扰。支持打散后，消息数据积压和消息推动时效会大幅提升。

5 课后考试作业场景解决方案

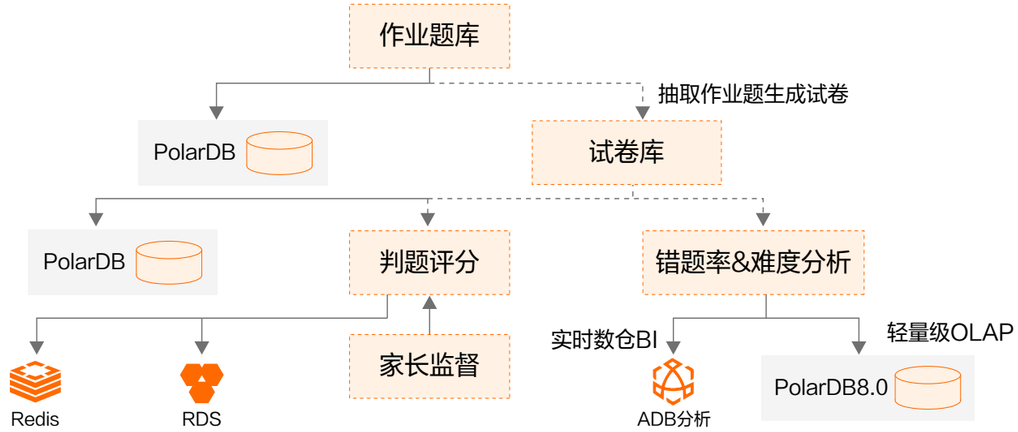
① 方案背景

学生完成课堂学习，正常结束后会通过试题、作业来体现课堂学习效果和巩固课堂知识。考试作业是从海量题库中按照当前课堂打标知识点关联后提取出来，比如题库中有和数学加法联系相关的上万道题，提取其中 20 道生成作业。答题的快速多样性也是体现在线教育对比传统线下统一试卷的优势，不同学生答题各异，同一学生答题可多种难度，这增加了答题质量和体验，但是给数据库带来了频繁的数据请求。作业在规定时间内完成后，会判题评分，在作业答题完成后，会有更多难度让学生挑战，对应生成不同难度和关联作业，并在最终提交后，生成排名。在家长端，家长们也能方便的随时查看学生的作业完成和评分排名情况。

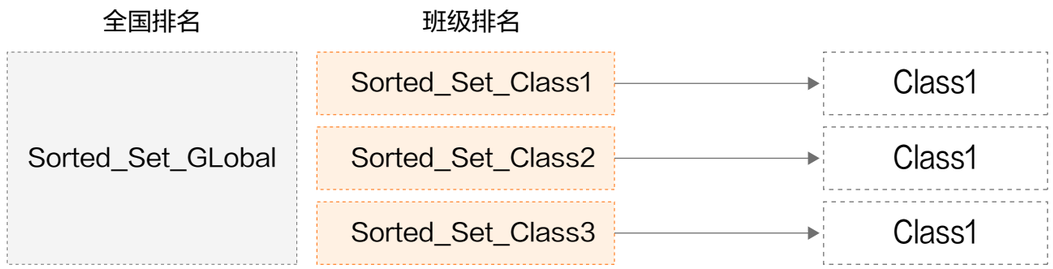
② 方案架构

- 作业明细、错题记录、答题记录、题目收藏、易错题等这些课后作业相关的数据量都非常大，普遍以亿为单位，导致数据库存储几乎都是 TB 级别。
- 统一答题时间，不同学生同时生成不同答题内容，对作业库的查询和试卷库的读取量非常大，批量提交又会形成超大 TPS 场景。
- 班课的作业、答题排名，需要在有限的作业时间结束后快速生成。而随着答题的推进，分数的 update 是非常频繁的，不断按新的分数刷新学生端排名对数据库 CPU 计算能力消耗非常大。
- 对于答题统计、答题计数、易错题分析、题目难易度评估，这些 OLAP 场景，对数据存储和分析能力要求很高。

对于作业答题场景：



对于排名场景：



3 关键步骤

- PolarDB 的存储计算分离架构，能保障单表亿级别的时候依旧具备高并发低延迟的读取和批量提交能力。
- PolarDB 读写分离架构，新增只读节点最多只需 5Min，在定期大型班课场景，能在开始前几分钟快速弹性升配和扩容，提升数据库容量，日常保留较低规格。
- 云数据库 Redis 所支持的 Sorted set 数据结构，学生分数数据写入后天然有序存储，非常适合排名场景。
- 使用分析型数据库 ADB 来承载诸如试卷表、课题表、错题表等多表关联，分析错题分布，以及统计答题率、错题率、平均答题时长的 OLAP 场景。
- 同样使用ADB数据库构建实时数仓，将试卷内容、答题记录等通过数据同步工具 DTS 实时同步到 ADB 数据库中，使用 SQL 语句简单快速的实现数据分析和 BI 报表产出。

6 课后分析场景解决方案

1 方案背景

学生的课程报名、行为日志、上课、签到打卡、答题准确率、排名等数据，都是生成学生或者课堂报表、分析课堂质量，以及课程推荐等场景的重要数据。例如，按照学生的错题分布，实时推荐学生课外读物或者其他课程来巩固知识点。随着课程的推进，这些数据分析的实时性和灵活性是很多客户考验分析质量和分析结果转换率的重要因素。建议在线数据 OLTP 的场景在关系型数据库如 MySQL、PolarDB 中产生和查询，而 OLAP 分析类的场景，特征如海量数据、复杂查询、多表关联和函数计算等，由专门的分析型数据库 AnalyticDB 来做实时分析，中间使用 DTS 做数据实时流转，或者基于 Data Lake Analytics 数据湖的设计思路来汇聚多种数据源来做准实时或者偏离线分析。基于以上 OLTP、OLAP 分离的思路设计分析场景，不影响在线业务，也让分析更实时，数据价值化。

2 方案架构

在线数据大量在关系型数据库中产生，称为热数据，通过 DTS 数据同步工具实时同步到分析型数据库 AnalyticDB 来做实时分析。除此之外，在线数据的产生除了结构化的数据库数据，还可以是半结构化或者非结构化的日志数据、消息数据、OSS 文件，都可以汇聚到 AnalyticDB，形成数仓来做分析。

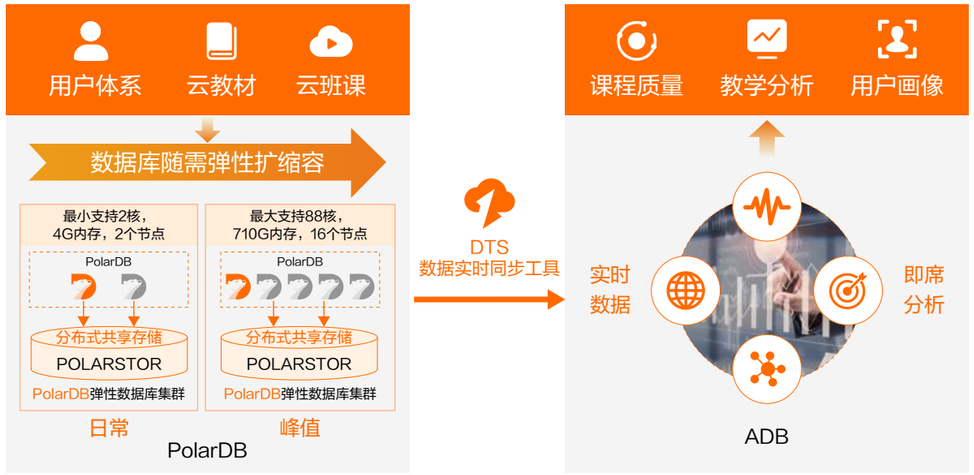
准实时或者偏离线数据，称为温数据或者冷数据，数据一般存放在日志、OSS 存储或者数据库里，建议使用 Data Lake Analytics 数据湖来直接进行分析或者清洗，不用再冗余一份数据，将清洗后的结果直接输出到比如报表或者业务系统，也可以存储到 AnalyticDB，基于 AnalyticDB 的高性能查询引擎进一步做分析查询。

2 方案架构

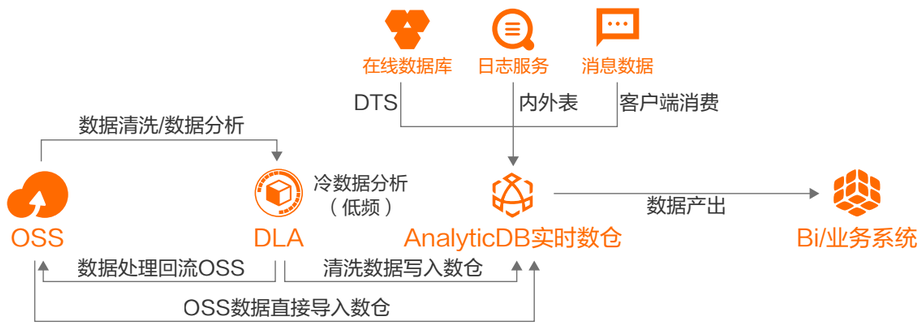
- 在线数据大量在关系型数据库中产生，称为热数据，通过 DTS 数据同步工具实时同步到分析型数据库 AnalyticDB 来做实时分析。除此之外，在线数据的产生除了结构化的数据库数据，还可以是半结构化或者非结构化的日志数据、消息数据、OSS 文件，都可以汇聚到 AnalyticDB，形成数仓来做分析。
- 准实时或者偏离线数据，称为温数据或者冷数据，数据一般存放在日志、OSS 存储或者数据库里，建议使用 Data Lake Analytics 数据湖来直接进行分析或者清洗，不用再冗余一份数据，将清洗后的结果直接输出到比如报表或者业务系统，也可以存储到 AnalyticDB，基于 AnalyticDB 的高性能查询引擎进一步做分析查询。

基于AnalyticDB实现OLTP、OLAP场景分离：

OLTP 与 OLAP 分离的设计理念

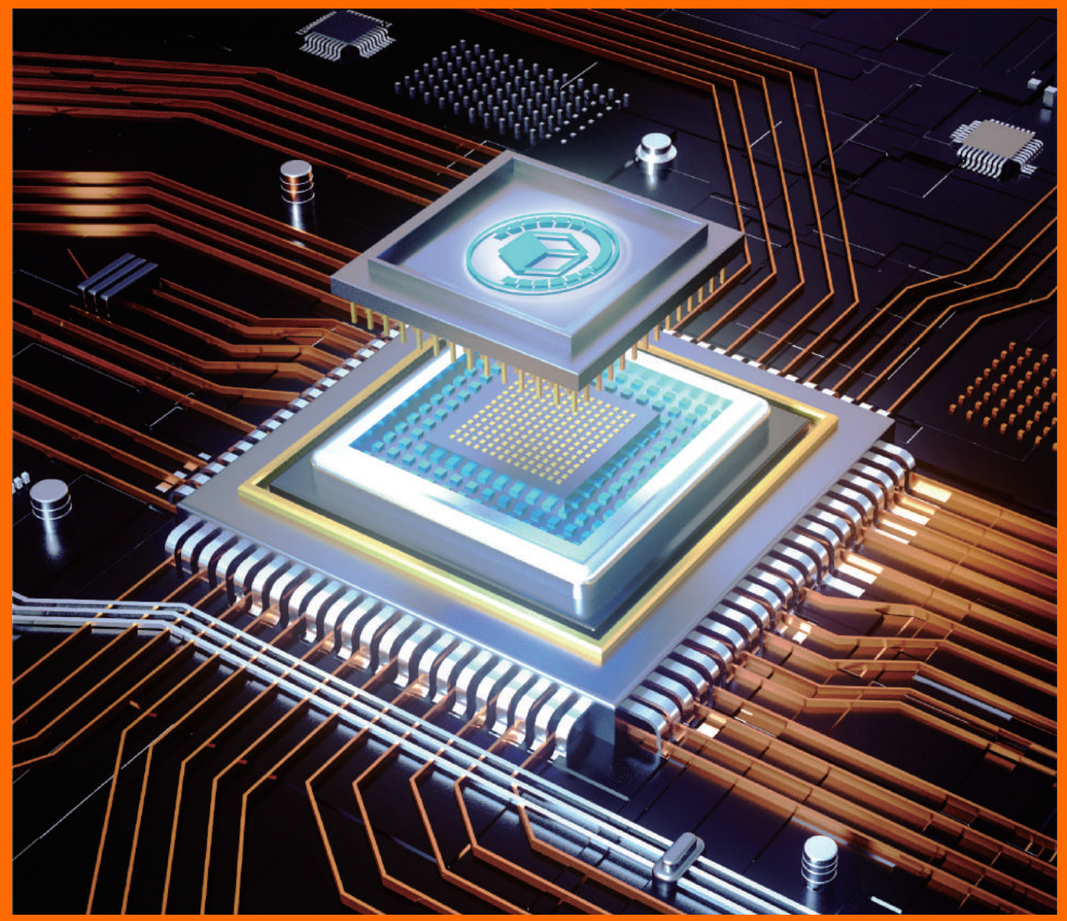


基于 AnalyticDB+数据湖 DLA 实现实时数仓:



3 关键步骤

- 按场景选择弹性或者计算型 AnalyticDB，弹性能实现冷热数据自动分离，冷数据存储在 HDD 设备上，存储成本大量降低，保障查询性能，同时计算资源和存储资源分离，可以按业务需求和数据量选配组装。计算型 AnalyticDB 优先保障性能，在高并发和大计算场景表现优异。
- 配置 DTS，完成数据库到 AnalyticDB 的数据同步，选择分布键，让数据散列在 AnalyticDB 分布式存储里。
- 创建数据湖 DLA，DLA 可以对接多种数据库和日志等存储，运行为 serverless 模式，在发生查询的时候才计费，成本非常低，可以使用在数据清洗、冷数据查询和离线计算中。
- 配置业务系统或者 BI 系统连接 AnalyticDB 做数据产出，如果有 DLA，配置 DLA 结果数据写入 AnalyticDB 或者 OSS 对象存储。

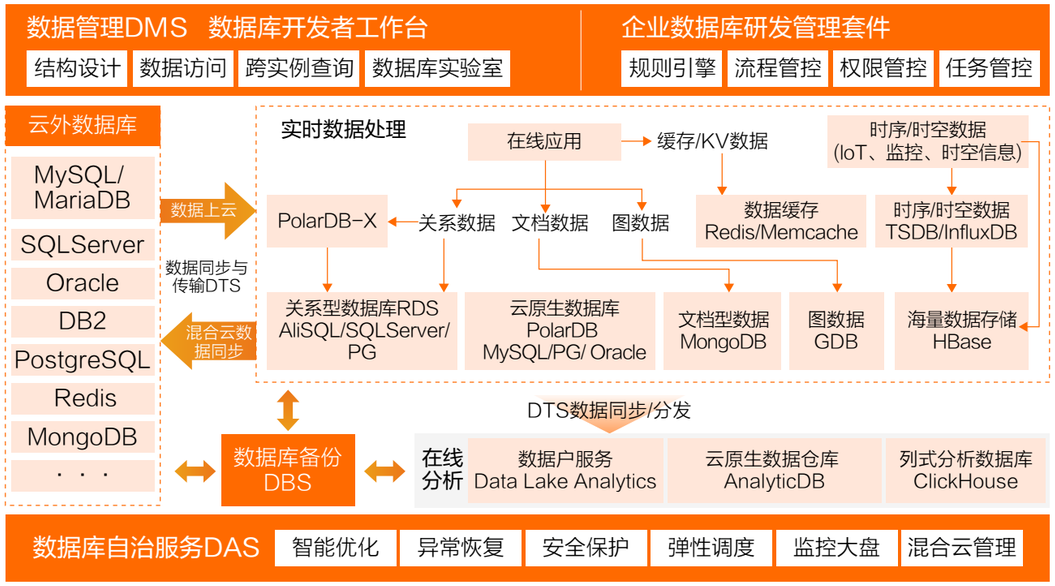


方案优势

阿里云数据库以真实业务场景出发，面对在线业务 OLTP 场景，提供了 MySQL、PolarDB、PolarDB-X 等数据库类型，来面对高流量、高并发、超大连接、海量数据、低成本等不同业务场景诉求，配合读写分离、快速弹性扩容等特性服务于前端业务。同时结合缓存来帮助数据库承担数据读流量、缓存层也有读写分离、集群等丰富的架构来支撑请求。此外 MongoDB、HBase 等在日志、高写入等场景大量应用。

在同样非常重要的 OLAP 场景，基于数据库构建实时数仓的方案，轻量、开发简单、快速实现。在轻量大数据场景、分析场景，能帮助业务快速实现数据分析实时化。

以 DTS 作为数据流转通道，云上已经实现了多种数据来源流向多种数据存储的成熟方案，让数据顺畅流动，产生价值；DMS 帮助构建起完整的数据库研发发布流程和可自定义的权限管理、审计、流程审批机制，保障公司内部数据安全，帮助研发和运维更高效的使用数据库，提升效率；DAS 智能数据库运维管理系统，能降低数据库运维负担，贴合业务场景，快速定位和发现解决问题。阿里云形成了从数据库内核到生态产品的完整闭环，应对不同业务场景和数据需求。



在线教育场景中对数据库的挑战尤为突出，阿里云数据库经过大量的实践，形成场景化的解决方案来很好的支撑。

- 针对营销、查课等波峰波谷场景，非常好的对应到云上弹性能力。PolarDB 存储计算分离架构，能实现分钟级别的弹性升降配和加减节点。Redis 企业级缓存（Tair）在超大连接和高并发场景下，表现尤为稳定，都能用来应对超大用户数的波峰波谷场景。
- 针对答题、约课等瞬间高并发、海量数据、集中查询的场景，PolarDB-X、Redis 集群版等架构，能实现数据和查询的打散，让多个节点来承载数据和查询请求。
- 对于广告投放、课程质量管理、课后实时统计汇总、营销等场景，基于 AnalyticDB 和 DLA 能实现海量数据基于 SQL 语言的快速分析和实时产出。数据经过 DTS 实时同步，DTS 具备的断点续传、服务高可用、数据一致性保障和校验等能力让数据流转更顺畅。
- 上课可用性要求越来越高，云上数据库产品在面对数据容灾、多活等业务需求，产品之间已经高度打磨，能快速构建出数据层的多 AZ、跨 Region、单向、双向数据同步等架构。

客户案例

1 智启蓝墨

1 案例背景

智启蓝墨是一家智能互联网教育公司，提供基于人工智能、云技术、大数据和移动互联技术研发推动教学模式变革的智能云教学平台。公司在发展过程中，面临以下业务挑战：

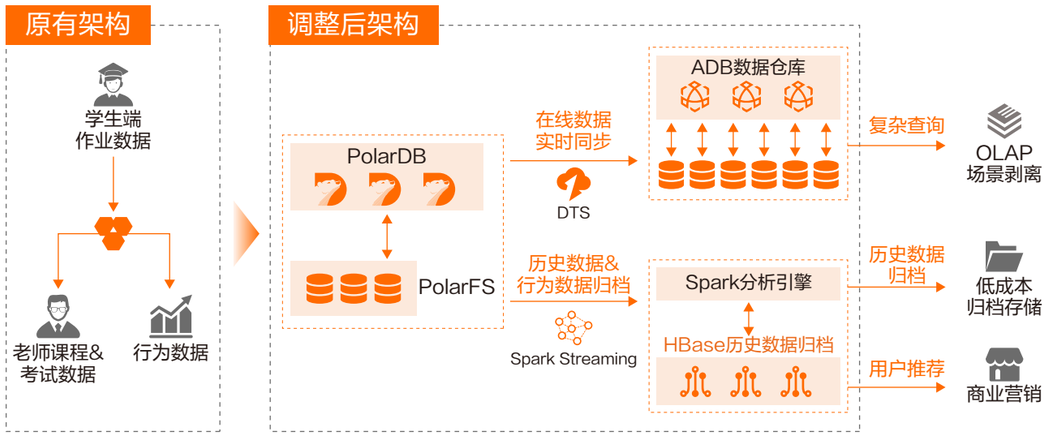
- 单库在线数据 6TB，单表 10 亿，存储和性能都碰到瓶颈。
- 业务和分析共用一份数据，报表分析对在线业务影响越来越大。
- 希望构建用户推荐平台从而挖掘数据价值。

2 业务架构

- PolarDB 超大存储规格满足单库 6TB 和单表 10 亿的存储和查询性能需求。
- OLAP 复杂查询切换到 AnalyticDB，构建清晰规范的 OLTP、OLAP 分离架构。
- 历史数据归档 HBase，上层构建 Spark，构建用户推荐场景。

3 业务效果

- 无需改造迁移到 Pola rDB，满足支撑未来几年业务增长。
- OLTP、OLAP 分离架构，复杂查询业务从在线库切换到分析库，减少业务与分析相互影响。调整后基本无业务相互影响。
- 使用 HBase 做推荐，发现历史行为数据价值，实现推荐场景。



2 猿辅导

1 案例背景

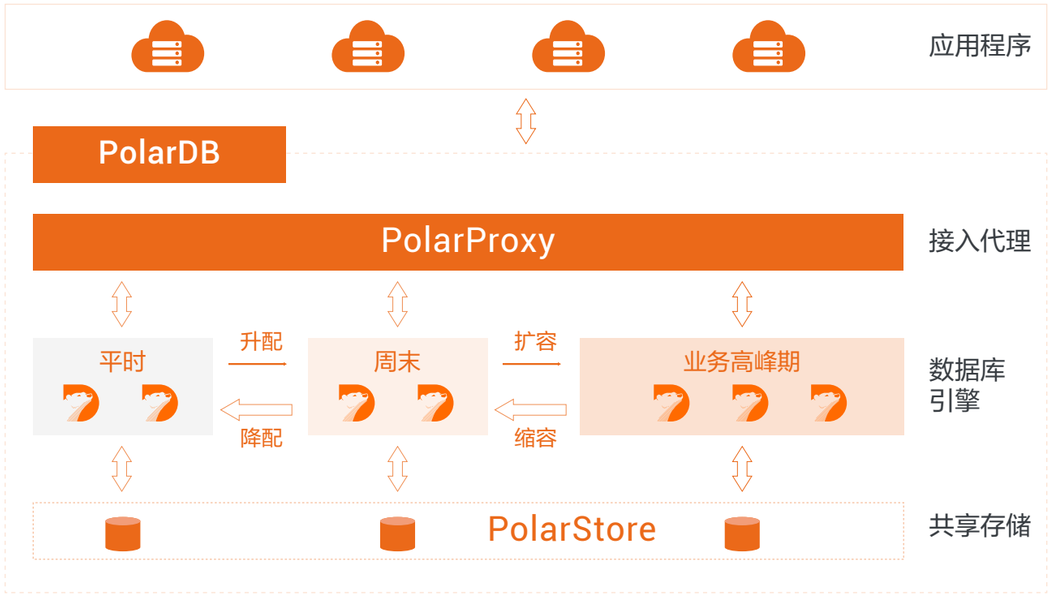
该国内知名的 K12 直播课程平台面临的业务挑战包括：

- 周末或者在线模考期间，同时在线用户数会瞬间激增，自建 MySQL 实例无法承受翻倍的流量增长，会有大约三分之一的学生无法正常进入在线考试。同时答题延时也从正常情况下的 1 秒之内增长到平均 5 秒，用户答题体验急剧下降。
- 用户量每年飞速增长，自建 MySQL 数据库的 CPU 利用率已经达到了 70% 以上，实例扩容对业务影响较大。

2 业务架构

在核心业务系统中，借助云原生数据库 PolarDB 快速弹性的能力，在业务的高峰期临时增加计算节点的配置和集群规模，与之前的方案相比整体成本大大降低。

- 纵向扩展（计算节点升配 / 降配）：得益于计算与存储分离，PolarDB 实例的计算节点可以单独升高或者降低配置，而且整个升降配过程只需要 5–10 分钟。
- 横向扩展（增 / 减只读节点）：由于存储是共享的，因此增加只读节点的过程不需要任何的数据 copy，整个过程也只需要 5–10 分钟。增加只读节点后，PolarProxy 可以动态感知并自动加入到读写分离后端的读节点中，对于使用读写分离地址连接 PolarDB 的应用程序无须任何改动便可享受到更好的性能和吞吐。
- 灵活的存储空间管理：存储空间支持按使用量后付费，每小时自动结算；同时也支持更优惠的存储包方式。同时，底层数据存储池可以自动动态扩容、平衡，避免存储碎片和数据热点。



③ 业务效果

通过云原生数据库 PolarDB 的使用，在帮忙客户节约成本的同时，又可以轻松应对在线用户流量激增的情况，也为用户提供流畅的在线教学体验。

- 业务角度，课程完成率从 90% 左右提升至 100%。
- 业务平峰期，轻松支撑 30 万用户同时在线学习；应对业务高峰时，只需提前一小时准备，便可临时使系统具备百万用户并发访问的能力。
- PolarDB 计算资源可以按需弹性伸缩，从 MySQL 迁移到 PolarDB 之后节省了 5 个只读库，整体节省了约 70% 的成本支出。

附录

术语表

术语	解释
IOPS	每秒对磁盘的读写次数，一般业务查询越多越分散（无法在内存中直接命中），该指标越大，也表示磁盘的读写压力越大，出现指标较高需要扩容磁盘性能。
QPS	每秒钟对数据库产生的查询次数。是评估数据库压力的重要指标，一般是秒为单位，比如单次查询耗时50ms，有100个活跃连接在发生查询，那么QPS等于1秒(1000ms)/50ms*100=2000。
DML	数据操作语句，包含增删改，insert、update、delete。
PUB/SUB	Redis里消息的通信模式，PUB代表消息发送，SUB代表消息接收。
OLAP	联机分析处理系统，这里通常是指SQL语句相对复杂，以多维分析、报表、海量数据的业务场景。
OLTP	联机事务处理系统，这里通常是指业务对数据库的请求以短平快的小请求为主，查询简单，响应要求高，请求量和并发量大。

3 大促营销
数据库解决方案

2009 年，阿里启动双 11 全球购物狂欢节。自此以后，每年双 11 大促业务的峰值和系统的峰值都在不断攀升。据全球知名市场研究机构 eMarketer 报告显示，中国在 2019 年零售额首次超过美国，成为全球最大的整体零售市场。2020 年，零售业和其他传统企业数字化转型更加迅速，将来还会有更多企业或平台开展大促业务。权威咨询机构 Gartner L2 分析师 Danielle Bailey 认为“零售商和品牌都应该拥抱双 11，不仅把它作为一个关键的销售时期，而且把它作为展示和扩大客户体验、产品和忠诚度创新的机会，这些创新的持续时间远远超过一天。”大促不仅包括以购物为主要导向的场景，如双 11、618、聚划算 99 等，也包括其他类场景，如几亿人在线协同办公、千万学生并发上课等。根据多年沉淀经验，阿里云提供基于 MySQL 产品系列（RDS MySQL、PolarDB MySQL）的大促解决方案，帮助企业客户轻松应对大促，实现消费者、平台、商家、供应商、物流等大促链路上的所有角色共赢。

大促面临的技术挑战

线上的电商促销从线下衍生而来，但比线下可以覆盖更多用户、场景、商品、主题和活动。电商促销活动等级分为超大规模促销、主题促销和日常促销，像双 11、618、年货节等是属于超大规模促销，并且在大促里面加入了很多新鲜的元素，比如：直播、双 11 狂欢夜、多场景互动等。大促链路上的主要角色有：参与者、平台（交易平台、支付平台、ISV 等）、物流公司等。大促不仅仅是对商家供应链、物流、网络运营商等环节的考验，更是对平台技术能力、全社会协同能力的挑战。　　大促经常遇到的问题大致可以分为以下几类：

1 海量访问

根据市场部分的推广和引流、历年双 11 的经验，大促的起始时刻的峰值呈现接近 90 度上升趋势。在这么大访问流量下，所有核心链路的增删改查都是在数据库上操作，对数据库有比较大的冲击。如果数据库有抖动或者不可服务，就会影响大促，甚至失败，比如：下单失败、网页无法打开、无法支付等。大促带来的巨大流量意味着到数据库里的查询会比较多，同时缓存失效也会导致数据库的并发查询比较多。

2 连接风暴

数据库的连接有限，同一时间创建过多连接会消耗大量的资源导致数据库卡顿，影响数据库稳定性。在大量线程并发工作时线程调度工作过多，缓存失效，缓存查询快，数据库查询慢，大量增删改查请求到数据库执行，导致数据库资源竞争加剧、锁冲突严重。上述问题又会加剧应用和数据库连接释放慢，如何预防数据库的连接风暴，降低创建连接的代价，稳定持续地提供数据库服务是数据库需要应对的问题之一。

3 复杂 SQL 或慢 SQL 问题

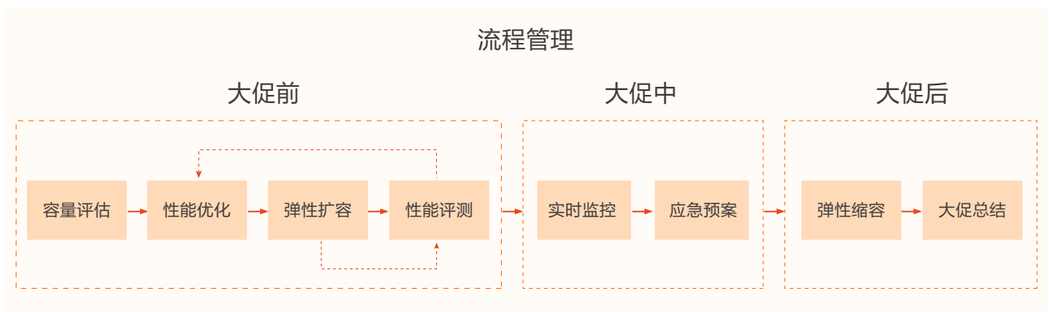
大促期间数据库既要满足业务响应时间又要最大化地提供 QPS。数据库系统通常被认为是一个响应者，需要处理所有接收到的请求。在大促高并发场景下，各方面资源大多数是被打满的状态，数据库高度拥堵，是不大可能实现低并发时的响应时间的。如果有复杂 SQL 或慢 SQL 的话可能导致系统资源耗尽，整个数据库服务卡顿或不可服务，最终导致业务损失。

4 热点商品售卖

所谓热点就是比较火的商品，一上架就被秒杀的商品，这类商品通常在数据库里是一行记录，对于 MySQL InnoDB 引擎而言并发越高行锁竞争越激烈，单行更新的性能会越低，行锁的竞争又会数据库层连接的堆积，连接的堆积又会拖慢整个数据库，数据库慢又会导致应用释放连接慢，进而影响整个应用集群。

解决方案

1 方案概述



大促对整个数据库系统都提出了极高的要求，每个系统在架构设计时都要以支撑大促丝般顺滑为目标。这绝不是大促当天就能全部完成的。所有工作从时间上来看可以分为大促前、大促中、大促后三个阶段。

- 大促前需要做容量评估、性能优化、弹性扩容、性能评测、以及梳理预案。
- 大促中需要关注实时指标和应急预案。
- 大促后需要做一些恢复动作，做些总结。本文不再赘述。

大促前的工作最多，花的时间最长，因为需要从复杂系统中梳理识别风险并提前规避或解决。

2 大促前的准备工作

① 容量评估

把大促做为一个项目来看，风险管理需要量化各种风险，因此需要做容量评估，检查各方面系统资源是否能支撑大促峰值。大促前首先要确认业务指标（包括交易创建和交易支付），由于现在大多数平台都是分布式的，各个业务模块需要根据业务指标换算成业务模块的技术指标和资源指标，然后根据该指标为大促做准备。在大促前可能需要多次优化或扩容动作，比如：性能评测不通过，需要再进行性能优化；如果性能优化已经到位，需要扩容后再做性能评测看能不能满足业务指标。

容量评估的有效方法就是要做压测，可以单独对实例做基准压测或基于真实场景的压测。

实例基准压测

实例既要满足业务需求，又要节约成本。从实例本身出发，可以通过 tpcc, sysbench, mysqlslap 等工具对实例本身能达到的基准值测试，从而了解实例本身的性能。通过基准压测发现从业务角度压测和基准值有偏差时，可以为从哪些方面优化性能提供参考。也可以通过修改压测语句针对秒杀场景单独压测。但是在实际应用中，只有基准测试的结果是不能充分评估真实业务场景的容量的，还需要做基于真实业务场景的压测。

实例级别真实业务场景压测

通常做真实业务场景压测，可以从业务侧用 PTS、loadrunner 等压测实例。阿里云数据库自治服务 DAS 提供一种不需要从业务压测的方法。DAS 从源实例上捕获流量，按照目标实例规格支持的极限速度或指定回放速度，在目标实例上进行回放。回放后可以看监控指标评估能不能支撑住双 11 峰值，如果不能则需要优化性能或扩容资源，或者降级一些不太核心的功能。

全链路业务场景压测

上面两个测试都是从数据库层面单独压测，为了和业务联动，方便评估所有链路工作时的容量可以进行全链路压测。全链路压测是阿里的首创，详细可以参考“阿里是怎么做全链路压测的”。为了更好地服务阿里云用户，阿里也把全链路压测沉淀成了产品 PTS，可以用 PTS 验证全国用户真实业务操作对服务端性能、容量和系统稳定性的影响，让站点提前经历真实的高峰业务场景，确保重大活动平稳支撑。也可以单独做秒杀，红包等典型高并发压测。PTS 可以大幅提升效率和智能化准备大促，全面逼近真实的系统验收能力，全面拉通端对端的链路，提供全方位的的解决方案。

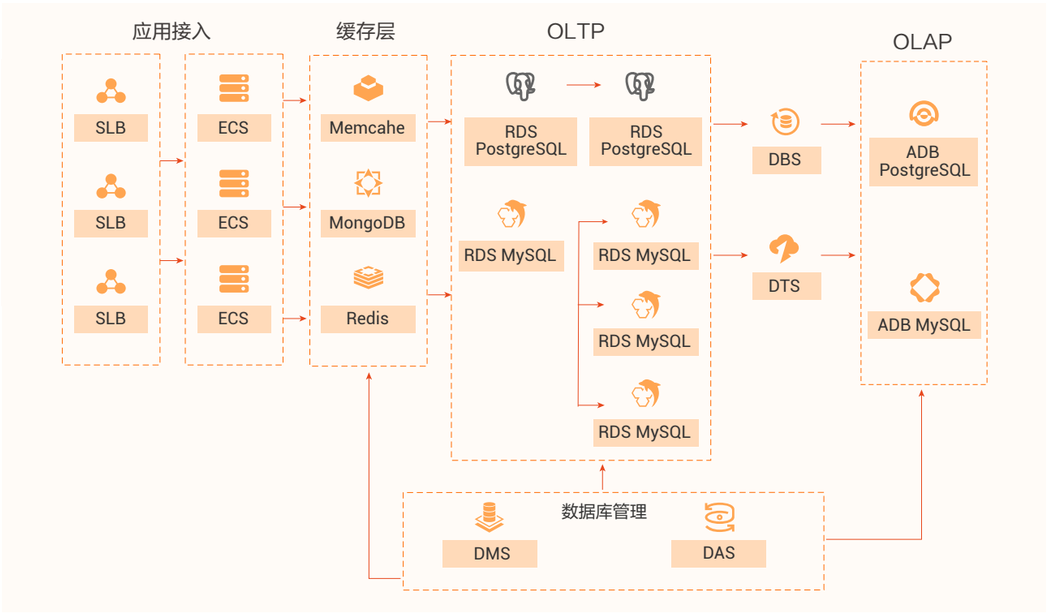
② 性能优化

性能优化是根据性能分析的结果，对系统性能的改进。通常在计算机系统中，这种活动的动机被称为性能问题，它可以是真实的，也可以是预期的。大多数系统当负载增加时会出现某种程度的性能下降。系统接受更高负载的能力称为可伸缩性，而修改系统以处理更高负载就等同于性能调优。系统可以进行性能优化说明某些方面存在瓶颈，期望通过优化瓶颈满足性能需求。

和大促相关的性能优化可以分为架构优化、参数优化和内核优化。阿里云 RDS MySQL 做了很多内核优化的补丁，可以帮助用户很好地解决热点库存、高并发、走错索引等性能方面的问题。更多信息可参见“参考资料”章节。

架构优化

系统要能支撑大促流量，最基本的要做好架构设计。优秀的架构设计通常具备的特点包括：低耦合、可扩展、弹性、容灾。数据库的架构优化可分为几个方面：缓存独立、读写分离、OLTP 和 OLAP 解耦、更新数据库版本，应用不同类型的数据库的特长解决相应的业务需求。



a. 海量查询请求

大促场景下查询请求比较多，应用可以从几个方面做，如：尽可能地使用静态化页面、尽量使缓存命中率 100%、读写分离使读流量不影响主业务。

读写分离是存储里面最简单的架构，顾名思义是把读和写分开。用主实例支撑写流量和对主备延迟要求高的流量，用只读实例支撑读流量，不能从业务侧写。对 RDS MySQL 来说只读实例和主实例之间通过主备复制的方式实现数据同步。

要想实现读写分离，通常可以采用以下几种方案：

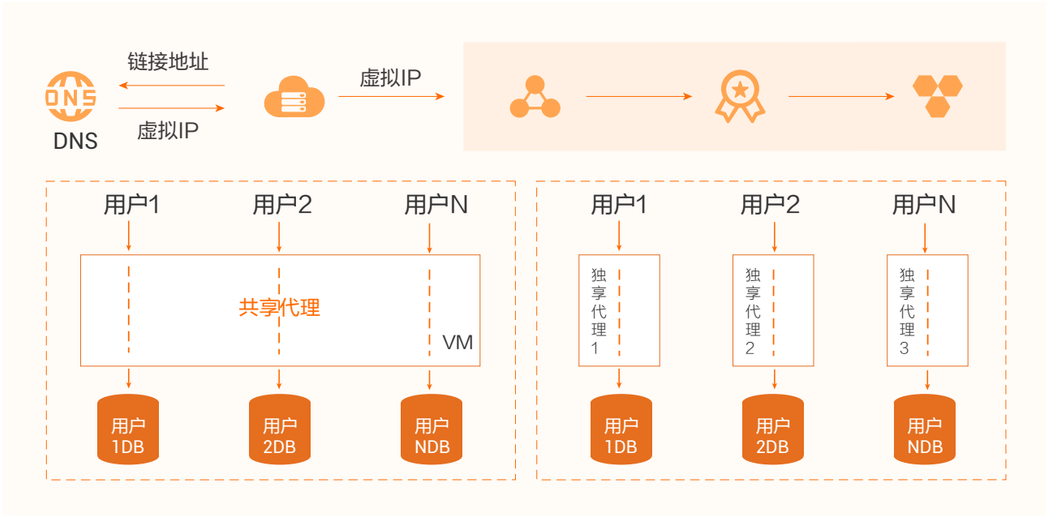
- 方案 1: 应用独立实现
- 方案 2: 开源的中间代理
- 方案 3: 阿里云的独享代理

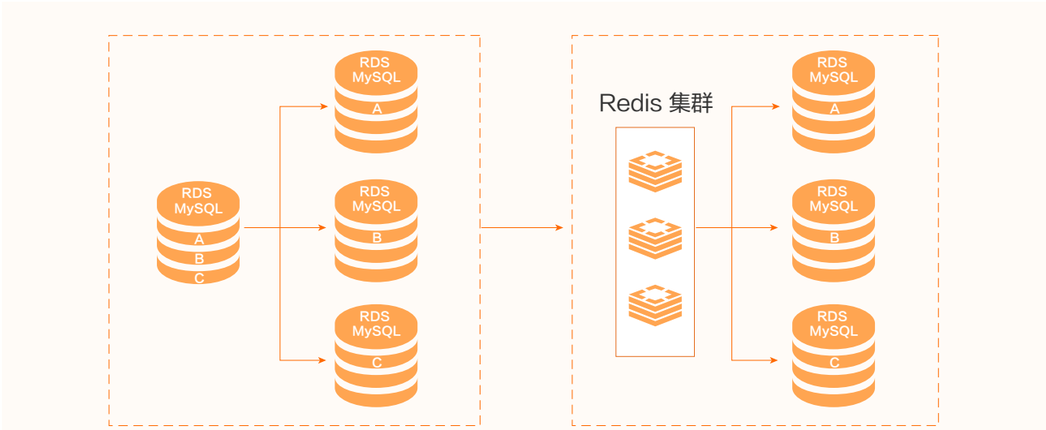
方案 1 的优点是可以把某类特定的业务放到某个只读实例上面，如把类报表业务放到某一只读实例。但同时要考虑只读负载均衡，只读实例延迟过大，不可用等方面的问题。方案 2 和方案 3 相对比较成熟。

b. 独享代理

数据库代理可以在代理层实现读写分离、流量控制、连接保持等方面的优化，避免业务流量同时直接访问数据库，造成数据库瘫痪，进而影响业务不可用。数据库代理的位置可以参考下图：

共享代理是很多用户共享 VM(虚拟机)，如上图所示。共享代理的设计方式决定了它的资源隔离性不是很好，可能会存在多用户争抢资源导致访问服务不稳定的风险。独享代理是相对共享代理而言的，独享代理做了设计优化，首先它的隔离性比较好，不会出现和其他用户争抢资源的问题，资源可扩展，可承载更高流量；其次在大促时期开启读写分离，大促结束后释放只读实例，关闭读写分离，也不用变更应用内的连接地址。





c. 存储架构升级

一个新业务上线到蓬勃发展需要经历 scale up 和 scale out 阶段。在早期阶段，使用数据库的方式如左图所示，几个库分布在一个实例上面。当单个实例不能支撑峰值时，就会把几个库分别放到单独的实例，这几个实例互不干扰。随着业务的进一步发展，数据库很难支撑，就需要使用缓存，如右图所示。后期，如果单个实例也不能支撑 A 库时，就需要使用 PolarDB-X 做数据库拆分了。

- RDS MySQL 升级到 PolarDB MySQL

  如果经过的充分的评估单个 RDS MySQL 不能支撑大促流量，又不想分库分表，且 tps 不超过 6w，可以用 PolarDB MySQL；RDS MySQL 升级到 PolarDB MySQL 特别适用于大促准备时间短、数据延迟要求高、突发流量较多、DAS 数据库智治自动扩容的场景。

  PolarDB MySQL 可以满足数据延迟要求，同时易于弹性升降配置。

- RDS MySQL 升级到 PolarDB-X

  如果 tps 超过 6w，就需要对数据库进行拆分。拆分也有比较多的实现方法，通常可选择分布式中间件 mycat、atlas、oneproxy、cobar 等， 也 可 以 用 PolarDB-X（ 原 DRDS ）。PolarDB-X 是阿里巴巴自研的分布式数据库，在公共云和专有云环境沉淀打磨多年，历经各界天猫双十一核心交易业务及各行业阿里云客户业务的考验。承载大量用户核心在线业务，横跨互联网、金融 & 支付、教育、通信、公共事业等多行业，是阿里巴巴集团内部所有在线核心业务及众多阿里云客户业务接入分布式数据库的事实标准。

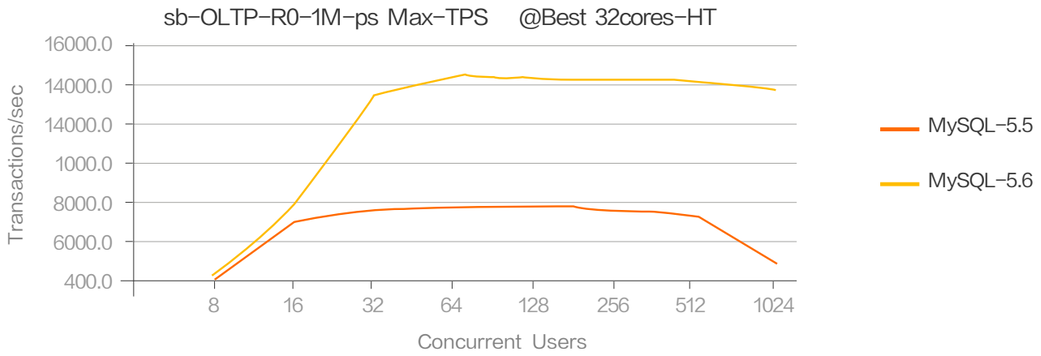
d. 数据库版本升级

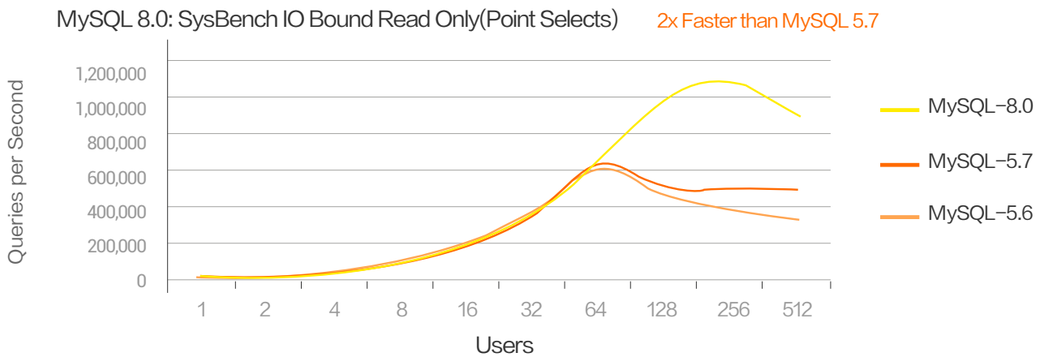
阿里云 MySQL 产品系列在不同的版本上做了不同程度的改进，各个版本的演进可以参考：5.6,5.7,8.0。因为和大促相关主要关注点还是放在性能方面。从下面的性能测试图可以看出比较新的版本性能是不同程度的有提升：

由于不同版本之间的优化器、排序方式等方面可能做了调整，如果不进行验证直接升级的话，可能会导致原来不慢的 SQL 变慢、排序结果不一致等情况，从而影响到业务，所以在升级之前必须进行充分验证。

架构优化从整体层面可以大幅度提高性能，除了添加只读实例比较快外，整体的架构优化是比较长期的方案，适合优化最初上线的业务。如果想要立即提升性能，可以从实例层面进行优化。

实例优化通常包括 2 个方面：SQL 和索引优化、参数优化。SQL 和索引优化需要和业务结合，最好在刚开始进行数据库表设计时就有性能意识，也可以等应用上线后观察，优先解决慢语句，这个在云栖上已经有培训视频在参考资料里面有相关链接。对应用来说最方便的是参数优化，应用不需要做任何改动，只需要调整参数就能提升性能。



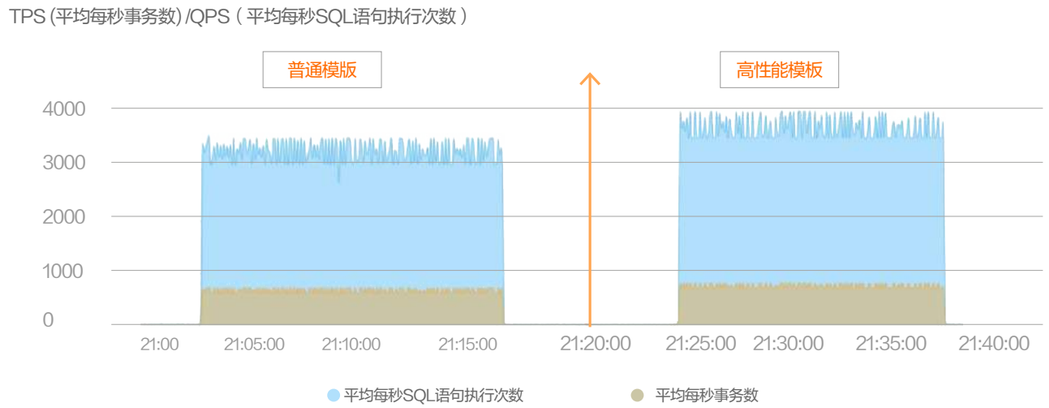


参数优化

以下介绍的是高性能参数模版的参数优化方法，高性能参数模板不一定适用于所有的场景，您也可以根据自己的业务场景有针对性的调整个别参数，比如：修改 binlog 同步方式、或修改 innodb_buffer_pool_instances。

高性能参数模板

高性能参数模板里面沉淀了经过评估后对性能有影响的参数，比较全。RDS MySQL 为了满足不同业务场景对数据库的要求，开放了很多参数模板。例如，为了满足高性能的需求而提供的高性能参数模板。从下图可以看到用性能参数模板可以提升至少 10% 的性能。

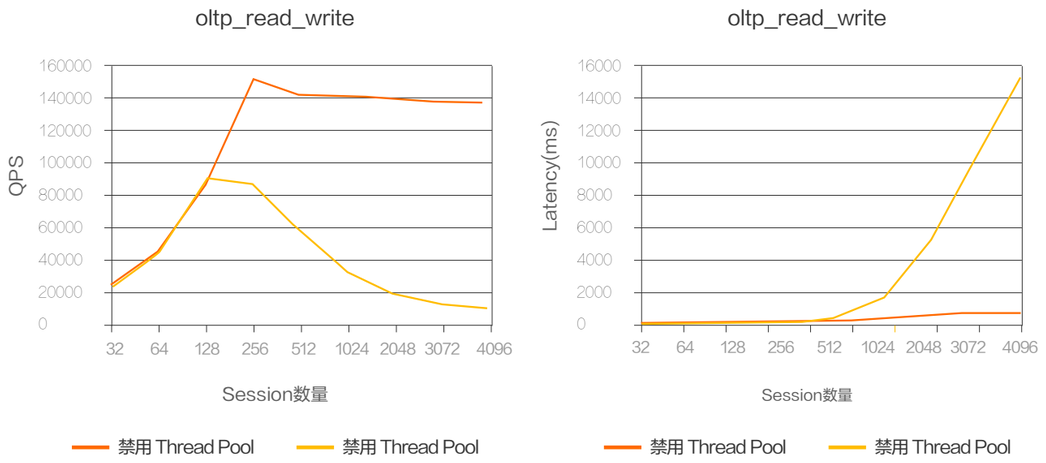


内核优化

阿里云把历年双 11 中遇到的问题沉淀到了内核里以便服务更多客户，特别是零点高峰时的高并发，热点商品扣减，数据库交互性能等挑战，我们有针对性地对内核进行了优化。

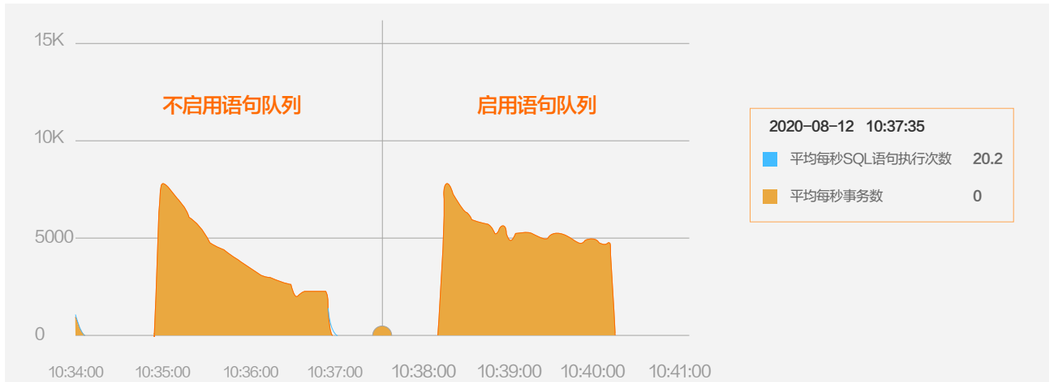
a. 高并发连接风暴

MySQL 默认的线程使用模式是会话独占模式，每个会话都会创建一个独占的线程。当有大量的会话存在时，会导致大量的资源竞争，大量的系统线程调度和缓存失效也会导致性能急剧下降。为了发挥出 RDS 的最佳性能，阿里云提供线程池（Thread Pool）功能，将线程和会话分离，在拥有大量会话的同时，只需要少量线程完成活跃会话的任务即可。阿里云 RDS 的线程池实现了不同类型 SQL 操作的优先级及并发控制机制，将连接数始终控制在最佳连接数附近，使 RDS 数据库在高连接大并发情况下始终保持高性能。优化后的性能如下图所示：



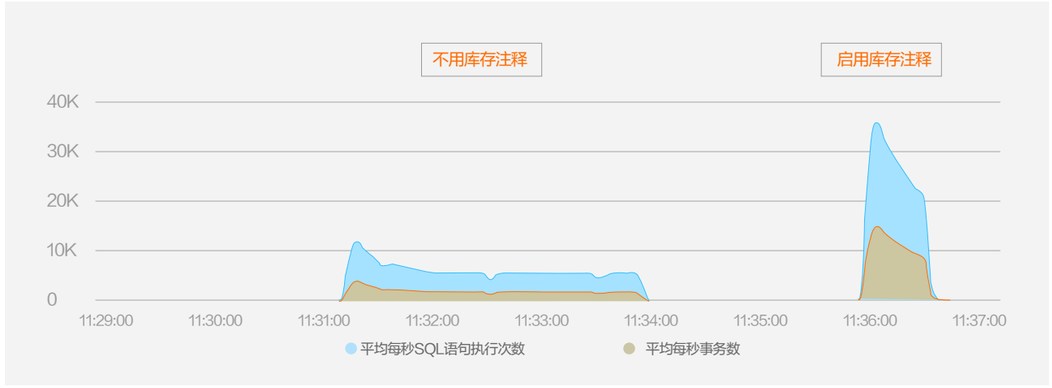
b. 行锁冲突问题

MySQL 的服务层和引擎层在语句并发执行过程中，有很多串行的点容易导致冲突。例如在 DML 语句中，事务锁冲突比较常见，InnoDB 中事务锁的最细粒度是行级锁，如果语句针对相同行进行并发操作，会导致冲突比较严重，系统吞吐量会随着并发的增加而递减。RDS MySQL 提供语句队列补丁，设计了针对语句的排队机制，将语句进行分桶排队，尽量把可能具有相同冲突的语句（例如操作相同行）放在一个桶内排队，减少冲突的开销，有效提高实例性能。如下图所示：



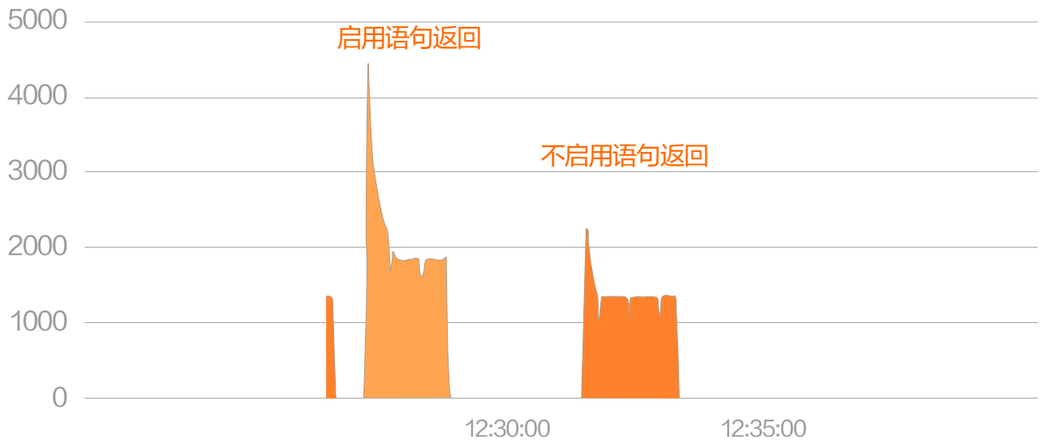
c. 热点商品扣减

在大促中经常会碰到很多用户在极短时间内以较低的价格抢一件商品的情况，这就是热点商品扣减。在这个场景对数据库的挑战比较大，一件商品在数据库里面是一条记录，瓶颈在数据库的行锁。为了减少数据库的锁竞争，阿里云 RDS MySQL 提供了库存注释补丁，使用排队和事务性 hint 来控制并发和快速提交 / 回滚事务，串行任务模型，提高业务吞吐能力。通过库存注释补丁，对单行的更新性能可以提升 3 倍以上。优化后的性能如下图所示：



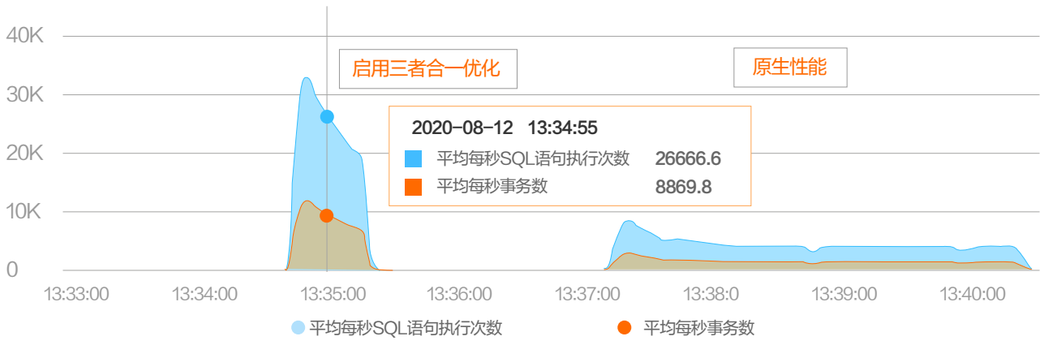
d. 多语句交互问题

在业务请求里到数据库的事务往往不是一条语句，而是好几条，甚至上百条语句，业务需要根据 MySQL 的语句执行结果做出判断后再做下一个业务处理，这些报文通常分为三类：Resultset、OK 和 ERR。针对 DML 语句返回的是 OK 或 ERR 报文，其中包括影响记录、扫描记录等属性，为了减少一次客户端和服务器的交互，returning 功能支持使用 DML 语句后返回 Resultset，这个功能叫语句返回。优化后的性能如下图所示：



上述提到的线程池、语句队列、库存注释、语句返回可以一起使用，性能可以得到大幅提升，如下图所示：

此外，阿里云 RDS MySQL 在回收站、大文件清理、查询缓存、高并发流量控制等都做了很多优化。在此就不一一赘述，如果有兴趣可以参考阿里云官网相关产品介绍。



3 弹性扩容

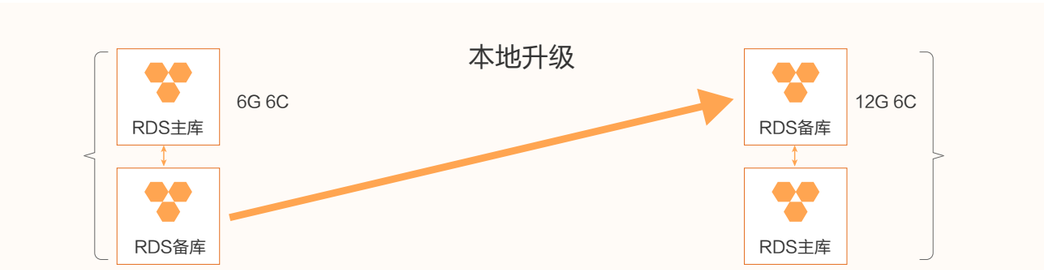
弹性扩容包括本地升级和跨机升级，在您购买实例时如果选择了“本地 SSD 盘”升级时可能会涉及的动作。这两种操作是针对 RDS MySQL 来说的，不包含 PolarDB MySQL。由于 PolarDB MySQL 使用存储计算分离架构，存储层完全 Serverless 的使用方式，弹性扩容会比 RDS MySQL 方便。以下介绍用户较多的 RDS MySQL 的扩容方式。

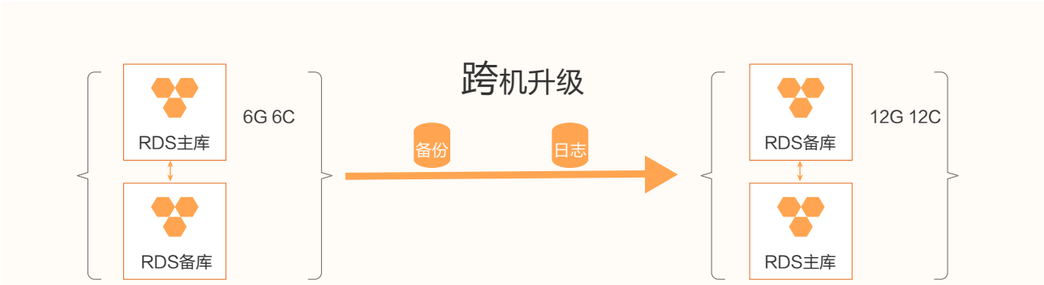
本地升级

不同的存储类型可以参考帮助文档。本地升级不需要拷贝数据，所以本地升级比较快，所有操作大概在半个小时内即可完成。即便有这种方式也不建议在大促当天实例压力撑不下去时升级，因为本地升级的概率比较小，相比大促带来的收益和影响，提前一周升级带来的成本可以忽略。本地升级的流程是先修改备实例的规格，做主备切换，再重启新备实例，具体可以参考下图。

跨机升级

不同的存储类型可以参考帮助文档。由于跨机升级需要拷贝数据，实例升级的时间取决于整个实例的大小，如果实例文件较大升级的时间相对较长。跨机升级的流程是在另外的主机拿全量备份文件恢复，全量恢复之后再用 binlog 日志追增量，具体可以参考下图。





3 大促中的各项工作

如果在大促开始之前已经做好充足的准备工作，大促过程中仅需要完成实时监控和应急两项重要的工作。

对数据库而言监控非常重要。一方面监控是检查实例健康度的重要手段，另一方面实时监控在大促中可方便运维人员发现实时问题，及时采取措施，保证大促顺利进行。

1 实时监控

在阿里云上，对于单个实例的监控您可以在“监控与报警”看到实例的每个指标，详细参考帮助文档。在大促场景里，数据库自治服务 DAS 可以提供一个数据库实时监控大盘把控全局，3 分钟就可以实现所有实例的接入，详情请参考帮助文档。

2 应急预案

为了采取有效的应急措施，大促前需要进行相关的风险梳理，需要统计业务模块、对应负责人、是否可降级及降级影响、执行时间、执行人、恢复时间等等。

即便已经完成上述工作，大促也可能有意料之外的事情发生。大促涉及的每个模块和上述表格里面的内容需要演练多次，在遇到数据库问题如非核心慢 SQL、走错索引 SQL、高并发 SQL 等都需要梳理对应的处理措施，并记录执行实例、执行时间、执行人、是否恢复等。

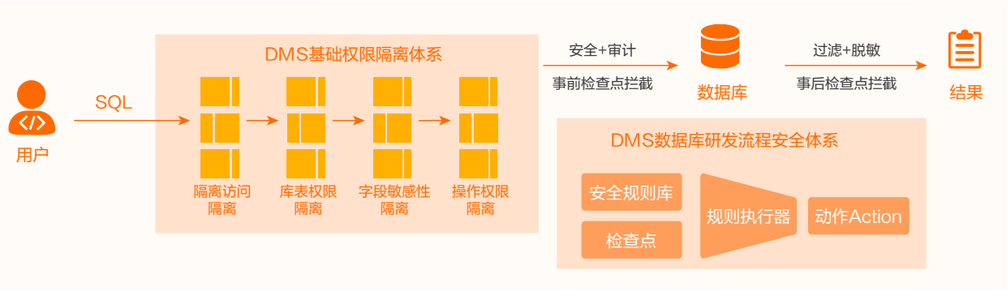
需要注意的是，在执行预案前，一定要和业务方一起评估预案，相关操作是否对业务连续性有影响。

4 流程管理

除了要做到架构和实例优化外，流程管理始终贯穿大促始终。数据库管理工具 DMS 可以把变更流程化、流程规范化，在数据被误删除后又可以快速地恢复回来，降低误操作对业务造成的影响。

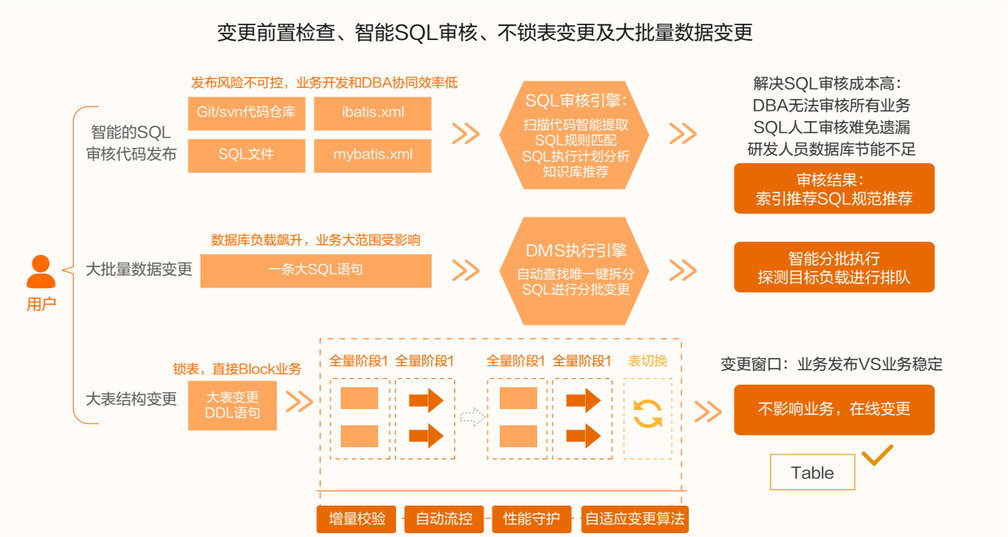
1 安全协同

DMS 对接公司组织架构和账号体系，多种角色协同的研发和数据管理，包括代码发布、数据定制、数据提取等流程。针对暴力查询拖库、删库跑路、数据泄露、慢查询拖垮数据库等问题做了很好的安全控制。



② 变更稳定性

大促期间，需要对数据库表变更（增加 / 删除字段、增加 / 删除索引）、数据的增删改查等进行严格控制，否则可能会造成灾难级故障，影响大促。DMS 在变更方面做了很多优化，针对做大表结构变更经常遇到的锁表问题，提供了无锁变更功能。针对大批量数据变更可能引起数据库负载飙升问题，提供了智能分批执行功能。



③ 数据恢复

数据追踪是通过解析已经转储到 OSS 上的 binlog 或本地热 binlog、逆向 binlog 里面的 DELETE/UPDATE/INSERT 语句，找到执行前后的镜像，反转成恢复语句。在恢复时可以按条件指定日志内容，不需要下载 binlog 手工解析，可以支持逐条恢复数据，更安全。

优势解读

1 技术方案成熟

本篇提供的技术方案经过历年阿里双 11 大促验证，同样也是双 11 背后技术的打磨和沉淀。每年的双 11 都会经历至少 3 个月的技术筹备，进行风险梳理、容量规划、模拟压测等。数据库是其中最重要的一环，所有的下单、支付、库存、优惠、商品等信息都需要数据库进行处理。相应地，阿里云 MySQL 系列产品在技术上也沉淀了热点库存更新、SQL 限流、语句返回、线程池等补丁，这些功能都已经开放。

2 内核增强

相比开源 MySQL，阿里云 MySQL 系列产品不仅加强了数据安全和性能提升方面的工作，还从内核增强层面做了以下工作：

在针对高并发、海量流量场景做了并发控制、线程池补丁；对走错索引、超长执行时间 SQL 场景做了语句注释；对热点库存更新场景做了热点库存更新补丁；针对加索引造成实例抖动、针对小表执行 DDL 消耗十几分钟甚至更长时间开发了 Faster DDL；针对原生 Query Cache 并发处理差、内存管理差、缓存命中率低性能差的问题开发了 Fast Query Cache。

3 生态工具齐全

阿里巴巴提供齐全的数据库产品生态工具。

- 数据库自治服务 DAS，基于机器学习和专家经验实现数据库自感知、自修复、自优化、自运维及自安全的云服务，帮助消除数据库管理的复杂性及人工操作引发的服务故障，有效保障数据库服务的稳定、安全及高效运转。
- 数据管理 DMS，提供免安装、免运维、即开即用、多种数据库类型与多种环境统一的 web 数据库管理终端；提供数据快速恢复、数据管理、大促变更管理等方面的解决方案。
- 数据传输服务 DTS 可以提供异地多活、数据订阅等功能。
- 数据库备份 DBS 可以提供可见的备份、数据清洗等方面的解决方案。

客户案例

1 数云

1 案例背景

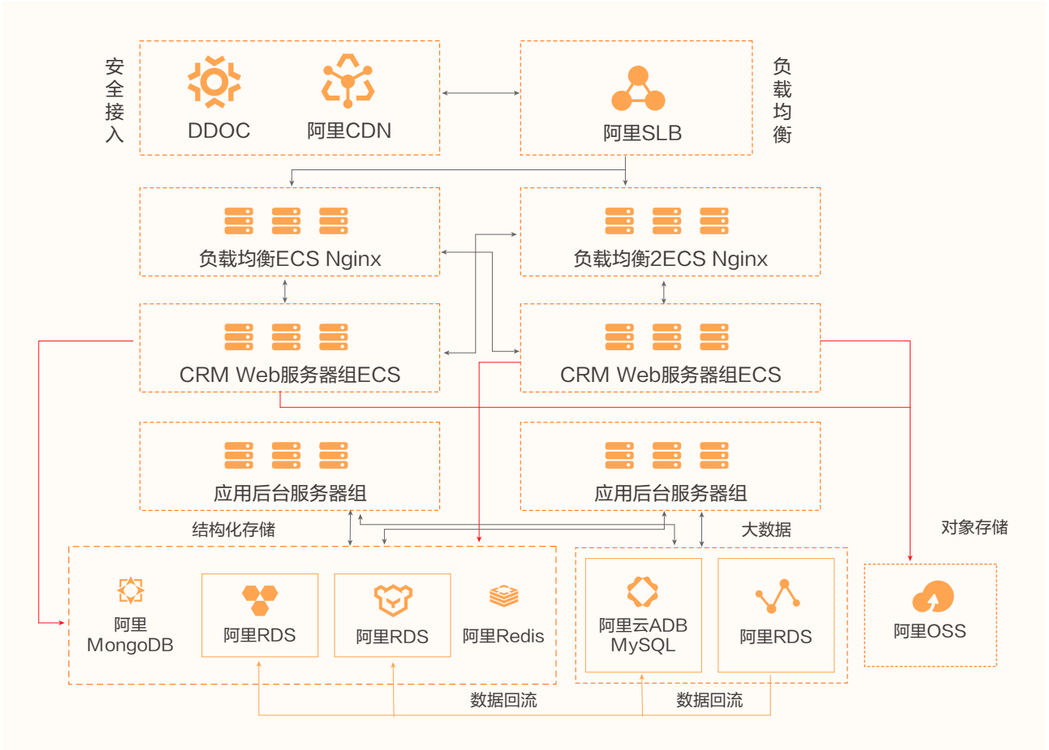
杭州数云信息技术有限公司成立于2011 年，伴随着电子商务、大数据应用和零售企业互联网化的趋势快速发展，目前已成为国内领先的数据化营销软件产品和服务提供商。

数云在业务发展过程中面临以下业务挑战：

- 有大促需求，在 618/ 双十一等大促期间希望能快速弹升 (分钟级) 实例的规格和 IOPS 能力。
- 涉及商家数据一致性读取，希望读写数据库的读节点延迟小，有较大的读写比，希望能够快速增加读节点。
- 单机存储瓶颈，传统数据库单个实例只能存3T 数据，单实例业务数据量大，维护成本高，代码配置复杂。
- 高并发写诉求，对高并发状态下的写能力有较高要求。

2 业务架构

- PolarDB MySQL 在大促期间可以在 20 分钟内完成 10TB 级数据的集群升配，快速弹升 IOPS 的能力。
- PolarDB MySQL 主节点和读节点之间采用物理复制的方式，读节点与主节点延迟大大低于普通的 MySQL 数据库，能满足对读节点的延迟要求；同时新增读节点，5-8 分钟可以完成，保证能快速提升集群的读能力。
- PolarDB MySQL 采用共享存储架构，存储和计算分离，能够最大程度的提供单实例的存储能力，最大可达 100TB，对于历史数据存储，海量在线数据存储都有较好的适配能力，降低代码复杂度和维护成本。
- PolarDB MySQL 通过引擎的优化和超强的 IOPS 能力提供高并发状态下的超强写能力，32 个并发以上，OLTP 写能力能达到普通 MySQL 的 2-3 倍。



3 业务效果

- 天猫聚石塔商家，在双 11 等业务大促的时候，需要对数据库计算能力 (CPU、内存、IOPS) 和存储量做临时升配。客户使用传统 MySQL 数据库的时候，数据库升配时间会随着存储量的大小、宿主机资源的情况而不断上升，最大的实例升配可能要 6-8 个小时。当实例数多的时候，升配时间变长、升配带来的运维成本更高，同时资源冲突可能会导致升配失败。PolarDB 提供节点升配 10-20 分钟、增加节点 5-8 分钟等高弹性能力，解决客户大促期间升配的痛点。这是客户选择 PolarDB 的最重要原因。
- 数云客户业务是天猫 CRM 系统提供商，客户数据库的部署模式是多租户部署，即每个实例上会部署一个或多个数云客户的数据，因此数云使用的数据库实例数据量都比较大。较大的实例一般是 2T-3T，约有上百个实例来满足该业务需求，数据库连接使用比较复杂、管理成本较高，也存在单实例存储瓶颈需要做数据迁移。PolarDB 的共享存储功能，有效解决了客户这三个痛点。
- 数云多租户的数据库部署模式对单实例数据库的写能力有较高要求，PolarDB 的高并发写能力远超过传统 MySQL，解决了客户高并发写瓶颈。

附录

术语表

双 11：是指每年 11 月 11 日的大型促销活动，最早起源于中国电商巨头阿里巴巴旗下购物网站在 2009 年 11 月 11 日举办的“淘宝商城促销日”，现已演变成全行业一年一度的覆盖线上线下的购物活动，及影响全球零售业的消费现象。“双十一”的发起者阿里巴巴 CEO 张勇将双十一称为“商业界的奥林匹克”。2012 年 11 月 11 日网络购物全日销售额超过美国网络星期一，成为全球最大的互联网的购物节日。双十一购物节战场延伸进 12 月，即“双十二”。

TPC-C：是专门针对联机交易处理系统（OLTP 系统）的规范，一般情况下我们也把这类系统称为业务处理系统。

PTS：PTS (Performance Testing Service)， 是面向所有技术背景人员的云化测试工具。有别于传统工具的繁复，PTS 以互联网化的交互，提供性能测试、API 调试和监测等多种能力。自研和适配开源的功能都可以轻松模拟任意体量的用户访问业务的场景，任务随时发起，免去繁琐的搭建和维护成本。更是紧密结合监控、流控等兄弟产品提供一站式高可用能力，高效检验和管理业务性能。

LoadRunner：作为商业性能测试工具，拥有强大的功能。

4 异地多活场景

数据库解决方案

随着云计算的蓬勃发展，越来越多信息系统选择部署在云计算环境下，因此基于云产品为信息系统的服务能力和数据质量提供保障尤为重要。为了防止灾难性的故障如火灾、洪水、地震、区域电力中断或者人为破坏等对信息系统造成不可挽回的破坏，需要构建容灾系统来保障信息系统的可用性和安全性。

2007 年，国务院信息化办公室联合银行、电力、民航、铁路、证券等八大重点行业，制定发布了国家标准 GB/T 20988-2007《信息系统灾难恢复规范》，明确规定了容灾能力的 6 个等级要求。企业在构建容灾系统时往往会参考国标等级，或者以此作为合规要求。然而，大部分传统容灾方案如同城容灾、同城双活、异地容灾、两地三中心等很难达到国标 5-6 级要求，同时还存在成本浪费，灾备单元健壮性不足等问题。

异地多活是新一代的容灾解决方案，在保证业务持续高可用的同时还能实现成本优化、地域级水平扩展、持续高可用等能力，本文会着重介绍阿里云主流数据库产品在异地多活场景下的解决方案。

问题和挑战

1 传统容灾解决方案存在的问题

1 异地容灾成本高

传统容灾方案中，为了提供 1:1 的容灾能力，异地灾备机房要投入与生产机房同等的建设成本。而受限于跨地域带来的延迟问题，灾备机房最多只能承担一些接受最终一致性的读操作，从而造成成本的浪费。

2 同城容灾能力差

同城容灾由于灾备机房和生产机房间隔较近，数据同步延迟可控，甚至可以提供数据强同步功能，但是容灾能力只能保障到数据中心级别，面对地域级的灾难场景无能为力，也无法达到国标 5-6 级能力。

3 灾备机房可靠性差

灾备机房常态不接业务流量，部署的业务系统无法得到充分的流量验证，当发生容灾切换时系统可能无法正常提供业务服务。

2 自行实施异地多活的挑战

1 流量管理难度高

异地多活要实现入口流量基于定制规则的有效管控，同时对于整个系统架构中的接入层、服务层、消息层、数据层等的流量路由与纠错实现统一管理，在规则分发，一致性保障等方面有很大挑战。

2 数据同步策略复杂

为了保障各数据中心实时持有全量数据，多活在数据层需要实现实时的双向数据同步，需要解决同步防环问题。同时异地部署的各个数据中心间隔距离较远，对同步性能、同步带宽有很高要求。

3 容灾切换数据质量保障难

容灾切换要保障 RPO 要求，切换过程中需要对业务架构中的各层进行状态检查，对规则分发的收敛情况以及跨数据中心的同步情况进行准确评估，难度较大。

4 多数据中心统一管控难度大

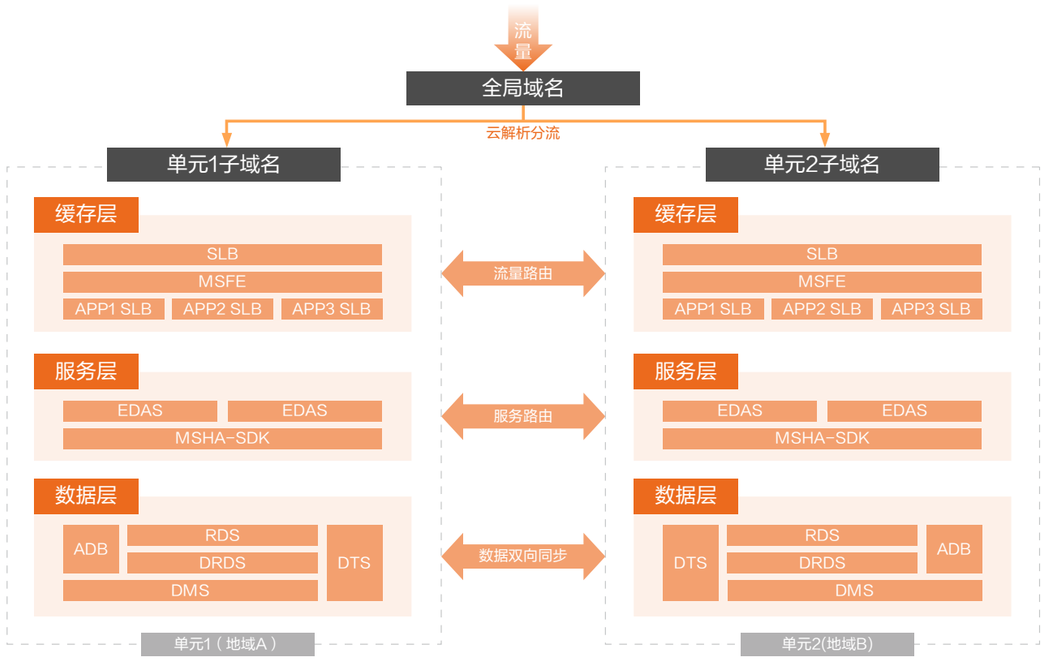
异地多活需要对多个数据中心实施一体化的管理，从新数据中心建站到多数据中心组件联动管理都对多活管控系统提出极高要求。

解决方案

1 系统架构

① 总体架构图

异地多活从业务视角来看是通过对业务做自顶向下的流量隔离来实现的，按照某一个分流维度对业务流量进行划分，并路由到不同的地域。整个部署架构分多个地域，每个地域称之为一个单元，其中某个单元又承担着整个多活架构的逻辑中心角色，提供一些中心化的服务能力（如 sequence 分发，强一致读服务等）。每个单元内的业务架构分为接入层、服务层、数据层：



接入层

业务流量通过租户侧 DNS 解析后按照权重分配到不同单元的接入层，进入接入层后，通过解析请求 header/cookie 中的分流标，对比自定义的分流规则，判断请求是否归属本单元，若归属本单元则走到本单元的服务层。否则流量需要转发到对端单元，根据不同的内网连通性可以走 upstream 或 304 跳转。

服务层

服务层部署的是业务应用系统，包括中间件等。如果使用了阿里云的中间件如 CSB、RPC 框架 、MQ 等，可以通过相应产品的多活接入能力实现在服务层的流量路由和纠错。针对 RPC 服务，在多活场景中分为：单元化服务、中心化服务、普通服务三类：

- 单元化服务：每个单元内完全自治的服务，是多活场景下的主要服务类型，会对流量进行单元归属判断，并进行流量纠错。
- 中心化服务: 强依赖中心单元的服务，流量都会被转发到中心单元，通过中心单元的服务访问中心单元的数据层。
- 普通服务：不做任何改造的服务，在本单元内进行服务调用，并访问本单元的数据层。

支持多活能力的主要服务类型是单元化服务，但是当业务系统比较复杂时，可能难以实现所有业务模块均按某一个维度划分，或者某些业务服务必须要单点部署以避免分布式部署的一致性问题，此时可由中心化服务和普通服务提供相应能力支持。

数据层

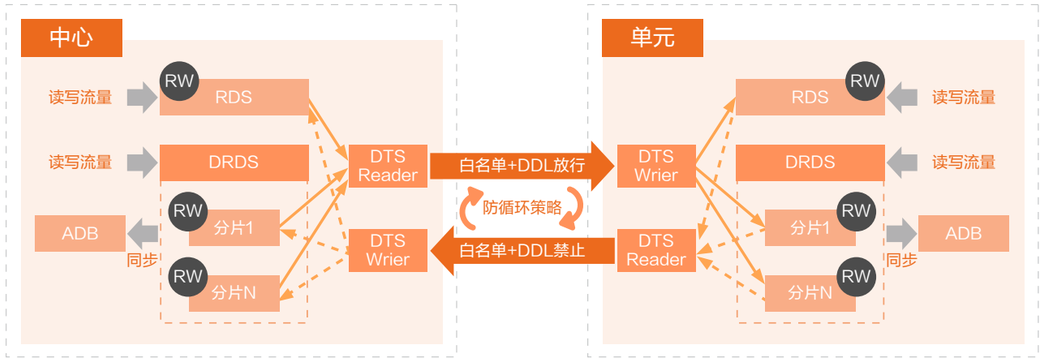
数据层解决数据库跨地域的部署与同步问题，并在灾难发生时对流量切换动作提供相应的数据质量保护策略。针对上层业务不同的服务类型提供 UNIT 和 COPY 两种数据同步策略：

- UNIT 类型：每个单元部署独立的数据库系统，单元之间通过 DTS 进行数据【双向】实时同步，保持每个单元都有全量数据，每个单元均可进行读写操作，读写流量会根据业务定制的分流策略进行单元写保护，这种同步策略用于支持服务层的单元化服务类型，是多活场景的核心同步策略。
- COPY 类型：每个单元部署独立的数据库系统，单元之间通过 DTS 进行数据【单向】实时同步，保持每个单元都有全量数据，中心单元可进行读写，非中心单元只提供读服务。这种同步策略用于支持中心化服务和普通服务，中心化服务路由回中心执行，普通服务可在单元内进行读。

2 数据库多活架构概览

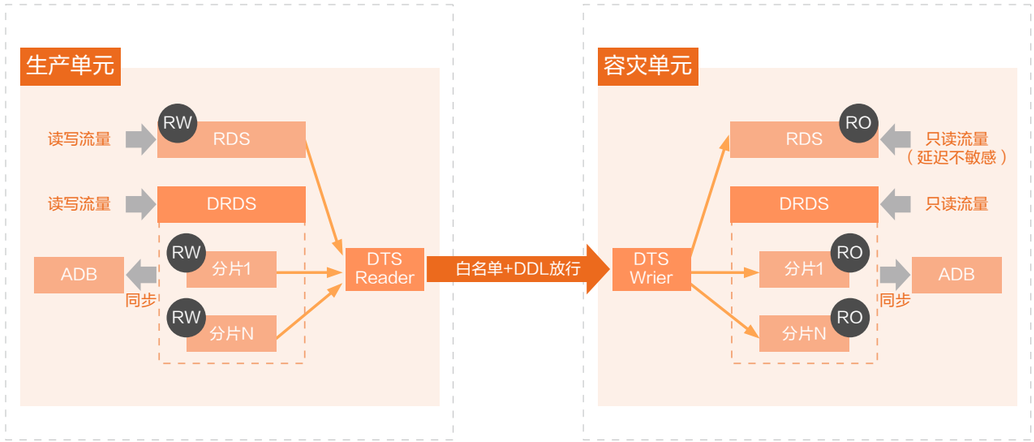
数据跨云同步的原则是尽可能的减小冗余同步，同一份数据在一朵云内可能存在于多个地方，比如在 RDS 中的数据同步到 ODPS，又同步到 ADB 中，那么跨云同步理想的方案是只在 RDS 层面做跨云同步，其他存储产品通过云内的同步能力补全数据。基于这个原则可以有效降低带宽压力和成本，保障数据一致性。本方案针对不同的服务类型场景，在数据库层面提供两种多活同步策略：

UNIT 类型多活架构



- 各单元内部署独立 RDS、PolarDB-X 以支持您的在线业务系统。阿里云 RDS、PolarDB-X 实现了与多活管控系统的无缝对接，实现基于中心的一键建单元能力。
- 各单元数据库为上层业务提供读写服务，每个单元保存全量业务数据，主要用于支撑单元化服务。
- 通过 DTS 的双向同步功能实现 RDS、PolarDB-X 的跨云同步。解决多活场景跨地域（数百公里～数千公里）数据同步的压缩、高效传输、数据防循环等技术难题。
- 实时向上层多活管控系统上报数据同步状态，实施单元间流量切换时的数据保护策略。
- 单元内通过 DTS 实现 RDS、PolarDB-X 到 ADB 的数据同步，实现单元内 AP 业务、TP 业务均可多活。

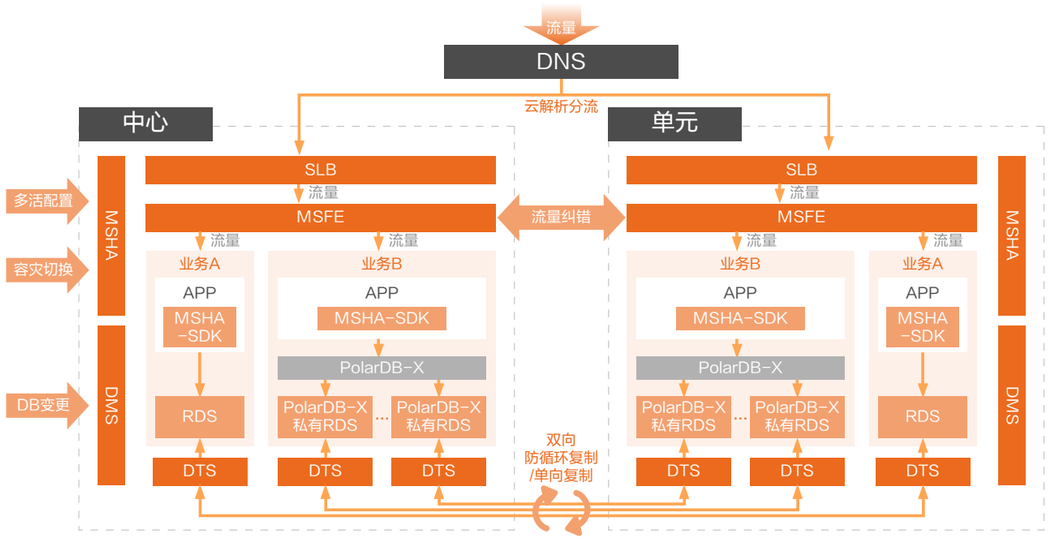
COPY 类型多活架构



- 各单元内同样部署独立 RDS、PolarDB-X 以支持在线业务系统。
- 中心单元为上层业务提供读写服务，其他单元提供只读服务，每个单元保存全量业务数据，主要用于支撑中心化服务和普通服务。
- 通过 DTS 的单向同步功能实现 RDS、PolarDB-X 的跨云同步。
- 实时向上层管控系统上报数据同步状态，实施单元间流量切换时的数据保护策略。
- 单元内通过 DTS 实现 RDS、PolarDB-X 到 ADB 的数据同步，实现单元内 AP 业务、TP 业务均可多活（单元只读能力）。

③ TP 业务多活架构

TP 业务通常流量较大，但查询逻辑简单，对访问延迟比较敏感，同时有事务要求。支撑 TP 业务的主流数据库产品包含 RDS、PolarDB-X 等，下图的多活架构可以实现基于 RDS、PolarDB-X 存储产品的 TP 业务多活能力：



此架构包含了支撑多活的基本要素：

多活流量控制

外部流量进入到业务系统，首先通过 DNS 进行基于权重的分流，流量进入各个单元后，由 MSFE 跟进自定义的流量划分规则（例如基于地域维度）实现多活的流量划分和路由纠错。同时在数据写入数据库时，通过使用 MSHA-SDK 提供的定制化驱动实现写保护功能，避免因为同一行数据在多点写入带来的数据质量问题。

多活数据同步

数据库侧 RDS、PolarDB-X 的多活能力通过 DTS 的异地双向防循环复制实现，DTS 会将在本单元写入的数据实时同步到异地单元，通过网络压缩、多并发消费等手段提高数据传输效率，为业务提供容灾切换时的数据支持。如总体架构中所示，DTS 针对不同的服务场景提供不同的同步能力（UNIT 类型、COPY 类型），并对上层多活管控实时上报同步情况以保障容灾切换时的可靠性。

多活配置

通过 MSHA 的控制台可以方便的进行多活任务配置。当基于中心的业务数据扩建单元时，只需要在单元购买一个同等规格的 RDS 或 PolarDB-X 实例，并在 MSHA 控制台进行简单的配置，就可以快速搭建起多活架构，存量数据的迁移和增量数据的同步都会通过 DTS 的产品能力自动实现。

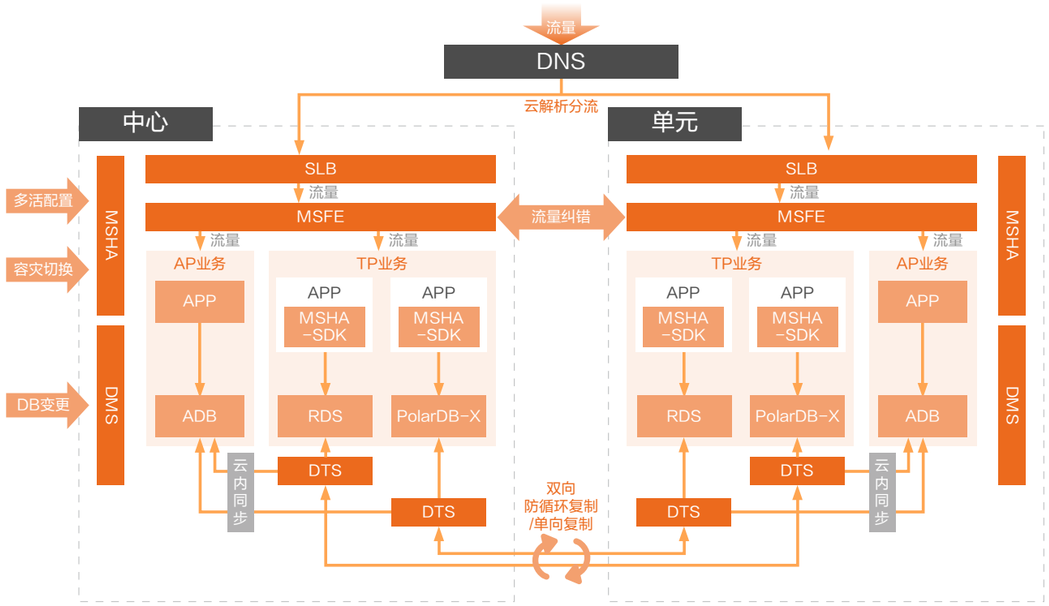
多活场景运维

当实施异地多活之后，日常的数据库运维变更会和过去存在些许差异，大部分变更操作只需要在单边进行（通常是中心），其他单元会通过 DTS 自动同步相应的变更。相反，如果同时在多个单元进行变更可能会导致数据同步异常。借助 DMS 来管理日常的运维变更，能有效控制不同单元的变更行为。

多活容灾切换

当进行容灾切换时，同样只需要在 MSHA 控制台进行流量调整即可，无需干预后端同步逻辑，任务下发后多活管控会通过底层封装的流量切换逻辑保障数据的一致性。

基于此方案可以轻松解决多活在基础设施方面的各类问题，从而让客户更加专注于设计业务的分流策略。分流策略的设计原则是保障核心业务流程能够实现单元内的闭环，即整个业务流程的上下游依赖均能满足某个维度的划分，上下游模块的调用均能在单元内完成，无需或尽可能少地访问其他单元，从而达到最佳的多活效果。



4 AP 业务多活架构

AP 业务通常流量较小，数据时效性要求低，但查询逻辑复杂。支撑 AP 业务的数据库产品以 ADB 为代表，如下是基于 ADB 的多活架构实现的 AP 业务的方案架构图：

通常 AP 业务并不是独立存在的，会以 TP 业务的存储系统作为上游，将 TP 业务实时产生的增量数据批量或实时的同步到 AP 业务的存储系统 ADB 中，供 AP 业务进行复杂的查询（这类场景的解决方案可参考《基于 ADB 构建离线和实时一体化数仓》）。因此，针对 AP 业务的多活实现会复用 TP 业务多活中的数据同步能力，是 TP 业务多活能力的延伸。

鉴于 AP 业务的数据来源是 TP 业务实时产生的增量数据，所以通过 TP 业务的多活能力时刻保障每个单元拥有全量的业务数据，并通过 DTS 将 TP 系统中的数据实时 / 批量同步到本单元的 AP 系统中，从而实现 AP 业务的多活。

此架构支撑多活的基本要素：

多活流量控制

流量控制的实现方式、产品能力、写保护等功能与 TP 业务多活基本一致，这里不再赘述，区别是可以为 AP 业务定义不同的分流策略，以满足 AP 业务的场景需求。

多活数据同步

跨单元数据同步借助 TP 多活的数据同步能力实现，可参考 TP 业务多活架构章节，单元内通过 DTS 实现本单元的 TP 向 AP 数据同步。

多活场景运维

与 TP 业务多活方案一致，不再赘述。

多活容灾切换

AP 业务以读为主，写入通过数据同步实现，并且对数据同步时延不敏感，所以容灾切换策略相对比较宽松，在 MSHA 控制台上对分流策略调整后即时生效。

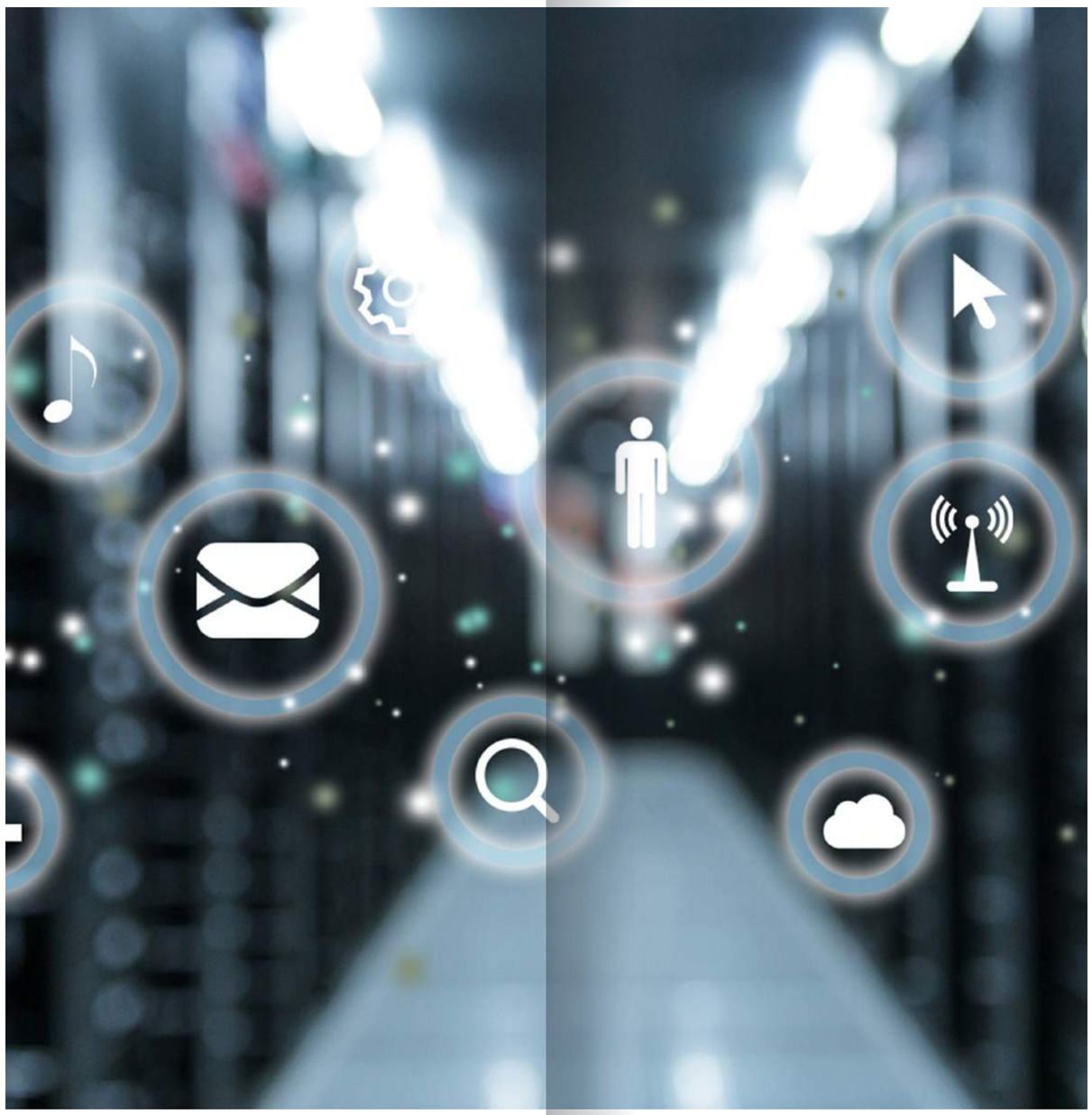
5 多活容灾切换策略

此方案中，针对多活容灾切换的整体流程如下：

预检查 → 开启切流态保护功能 → 推送分流规则 → 启动禁写保护 → 实时同步状态检查 → 关闭切流保护/禁写 → 结束

当发起容灾切换时，多活管控台会联动接入层、服务层、数据层、路由管理模块、写保护模块等进行一系列操作，来保障切换的高成功率及切流的数据质量。核心策略如下：

- 预检查阶段会对 DTS 同步延迟，切流态保护功能有效性等进行全面检查。针对演练场景可以确保多活业务达到了良好的切流条件，针对灾难场景可以跳过此步。
- 切流启动后，首先会打开切流态保护功能，通过 DTS 针对多活场景提供的定制化数据保护功能加强数据质量保护，其保护策略可以实现针对同一行数据，业务写入优先级高于同步数据优先级。
- 在进行新分流规则推送的同时会启动禁写保护，实现新分流规则在接入层、服务层、数据层统一收敛，避免不同节点规则不一致导致的写冲突问题，禁写保护提供了延迟可写、延迟禁写、延迟禁更新等多种策略以方便您针对不同业务场景进行选择。
- 一旦进入禁写保护状态，多活控制台会对同步链路进行实时检查，在状态符合预期后第一时间关闭保护策略，并完成整个切流动作。



6 业务接入多活关键流程

1 多活可行性分析	2 业务适配改造	3 多活架构部署	4 多活切换演练
1.容灾需求分析 2.多活业务选择 3.多活分流策略设计 4.业务服务类型梳理	1.业务系统多活适配改造 2.数据库多活适配改造	1.多活任务配置 2.单元建设完成 3.多活基础能力验证 4.多活架构接生产流量	1.演练方向及场景构建 2.演练实施 3.演练结果复盘

多活可行性分析

根据业务对容灾能力的诉求 (RPO/RTO) 以及对容灾成本的要求，评估是否使用多活容灾方案。明确使用多活方案后需要对业务进行梳理和划分，细化使用多活的具体业务，并设计分流策略，梳理业务在多活场景可能存在的服务类型并基于梳理结果评估改造成本。

- **容灾需求分析：**评估业务容灾能力要求 (RPO/RTO 等)，成本要求等。
- **多活业务选择：**业务系统可能非常复杂，而接入多活是有改造成本的，所以并不是所有业务都需要接入多活，需要根据真实需求进行合理筛选。
- **多活分流策略设计：**设计适合业务的多活分流策略，设计目标是尽可能的实现单元闭环。
- **业务服务类型梳理：**基于设计的多活分流策略，评估业务系统中各个服务接口的类型，不同的服务类型会带来不同的改造成本。
- **多活改造成本评估：**基于前面的分析结果产出整体设计方案，并评估多活整体改造成本。

业务适配改造

根据多活可行性分析的结果进行响应的业务改造适配，可以选取一些逻辑相对简单、改造成本相对较低的业务进行试水，以验证可行性分析结论。

- **业务系统多活适配改造：**业务系统接入多活进行的改造，包括域名改造、引入 MSHA-SDK、分流标透传、单元写保护等。
- **数据库多活适配改造：**数据库接入多活进行的改造，包括 schema 规范改造、sequence 使用改造等。

多活架构部署

正式进行多活架构的部署，包括个单元资源开通，在多活控制台上配置任务，后台搭建同步通道等操作。大部分操作都是由控制台的任务流自动完成的。

- **多活任务配置：**根据可行性分析阶段梳理的多活业务列表，配置对应数据库的多活部署任务。
- **单元建设完成：**任务部署后需要对存量数据进行全量迁移并启动增量实时同步。
- **多活基础能力验证：**部署完成后需要对同步能力，同步数据一致性，MSFE 分流能力等进行相应的原子能力验证。
- **多活架构接生产流量：**在完成了各项能力验证后，把流量割接到多活架构中，通常中心单元就是已有的生产单元，建议从中心单元小批量、逐步把流量割接到多活新建的单元中。

多活切换演练

当业务已经完成多活部署，接下来需要对整个多活架构的容灾能力进行验证，以评估是否符合容灾目标。

- **演练方案及场景构建：**针对业务场景设计合理的演练场景和演练方案。建议演练范围由小到大，灰度放量。
- **演练实施：**按照设计的演练方案进行演练实施，评估演练切换的 RTO、RPO 等指标。
- **演练结果复盘：**复盘和跟进演练结果。

2 应用场景

本解决方案适用于以下业务场景：

容灾能力要求高

异地多活可以达到国标 6 级的容灾能力，适合对容灾方面有较高要求的业务、业务流量比较敏感的业务或业务的某些核心系统。

流量要求精细化管理

异地多活支持多种流量管理策略，适合对流量管理有复杂需求的业务如按地域就近接入，按用户信息分配指定数据中心等。

业务快速发展

异地多活实现业务按照单元级别（数据中心级别）实施水平扩展，业务定义好一个单元的服务部署配比后，可以以此为镜像快速部署多个分布全国的单元，实现容量快速扩充。

业务读多写少

读多写少业务可以从业务行为上天然的规避数据异步复制的各类问题，并且业务接入多活的改造成本也较低，是比较适合实施异地多活的业务场景。

3 方案价值

异地多活不仅实现了传统容灾方案所提供的能力（如低 RPO、RTO），同时还具备更多能力。

异地多活与传统容灾方案的能力对比如下表所示：

	异地多活方案	传统容灾方案
灾备单元健壮性	常态承接业务流量，健壮性有保障	常态承接业务流量，健壮性有保障
业务高速发展支撑	支持业务以地域维度水平扩展	常态承接业务流量，健壮性有保障
流量隔离能力	支持多种维度的流量隔离能力	无隔离能力
成本控制	每个地域都承接流量，有效控制成本	灾备单元长期处于空跑状态，浪费资源

异地多活的价值具体表现在如下方面：

业务即容灾

传统异地灾备或两地三中心的异地灾备中心常态不提供服务，当发生地域级灾难时难以保证异地灾备中心的可用性，存在切换成功率低的问题，异地多活架构下各个地域数据中心不区分角色，全部常态承载业务流量，时刻保证业务系统健壮，各个数据中心既是业务体系也是容灾系统。

业务连续性保障

异地多活架构下各个数据中心常态承接业务流量，故障发生时只需调拨入口流量即可实现容灾切换，实现分钟级的容灾切换。同时随着参与多活建设的数据中心数量增加，参与调拨流量的比例会相应减少，未参与调拨的业务流量可以实现对容灾切换的“0”感知。

业务高速发展支撑

业务高速发展，受限于单个地域的有限资源，尤其是数据层存在单点性能瓶颈，除异地多活以外的全部容灾架构都只能在主生产中心执行写操作。地域多活实现业务级的流量闭环，各个数据中心均可读写，具备水平扩展能力以及跨地域的快速扩建能力。

流量有效隔离

异地多活本质上是提供了一种自顶向下的流量隔离能力，业务具备在数据中心级别完全隔离的能力，各个数据中心承载的流量大小可灵活调配，在最小隔离数据中心内（例如承载 1% 流量），业务可灵活进行风险可控的技术演进，例如基础设施升级、新技术验证等。

成本有效控制

在考虑单个地域发生地域级灾难的场景下，不论两地三中心还是异地灾备都需要在灾备中心部署可承载 100% 流量的系统资源，加上生产中心即达到 200% 的成本冗余。异地多活通过跨城多数据中心部署，有效分摊各个数据中心成本，实现成本小于 200% 冗余。

4 优势解读

阿里巴巴多年的实践沉淀

阿里巴巴从 2012 年开始实施异地多活，有超过 300+ 业务，上万数据库实例的实践经验。

一体化的解决方案

通过统一的管控入口实现接入层、服务层、消息层、数据层的统一管理和路由规则分发。从多活建站到容灾演练实现能力全覆盖。

流量精细化管理

基于阿里提供的异地多活方案，可以实现多维的流量管理策略，能满足业务如就近接入，流量均摊等不同的管理诉求。

分钟级切换

通过在数据同步方案多年的技术积累，可以实现跨数据中心的高效数据同步，保障容灾切换时的RPO最高可达到秒级。通过 " 一键切换 " 能力对各层规则统一管理，可以达到切换 RTO 分钟级。

数据质量保障

为了避免容灾切换时的数据脏写，提供了多种数据质量保障手段，有效控制切流态的数据质量问题。

成功案例

1 某税务系统

① 案例背景

客户一方面要考虑业务对容灾能力提出的强烈且苛刻的要求，另一方面要考虑超大体量下容灾建设带来的成本浪费问题。因此选择多活方案在支持容灾要求的同时需要解决成本、业务高速发展等问题。

② 业务多活架构

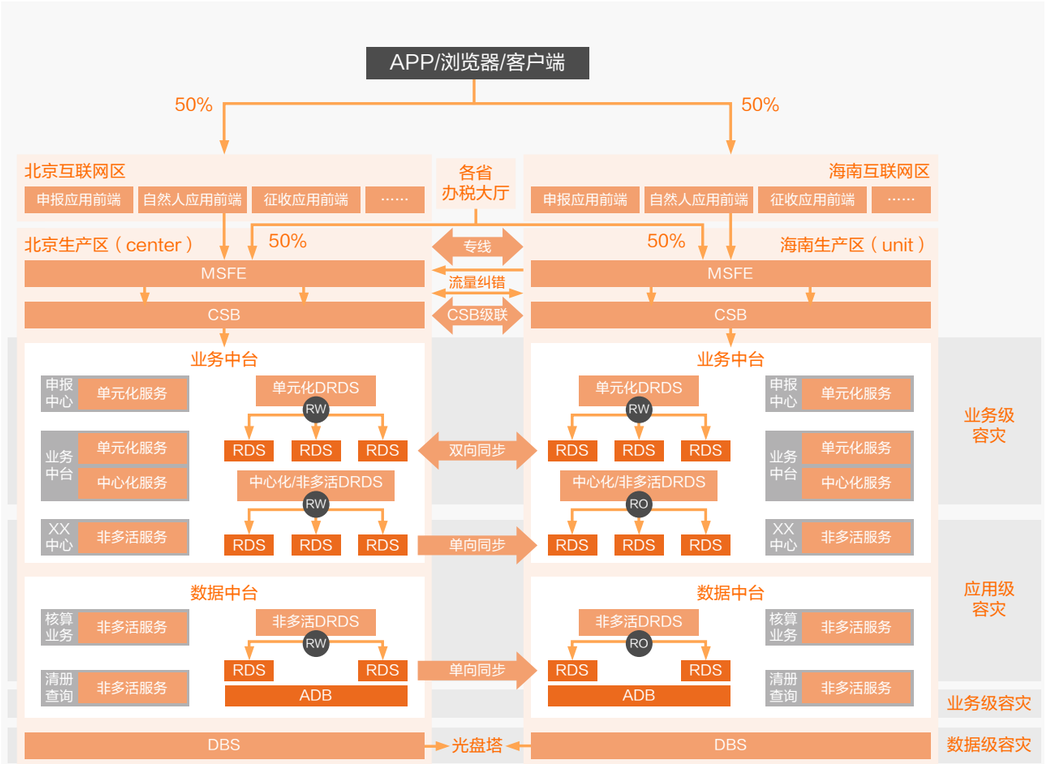
客户基于本方案，整合了 TP/AP 场景的多活能力，借助 RDS、PolarDB-X、ADB、DTS、DMS、MSHA 等产品，有效实施了异地多活容灾能力，达到国标 6 级容灾能力要求：

- RDS、PolarDB-X 承载 TP 业务数据，ADB 承载 AP 业务数据
- 通过 DTS 实现数据的跨域实时同步以及云内同步
- MSHA 实现多活流量管控和容灾切换动作
- DMS 实现日常运维变更和数据变更管理
- DBS 实现数据在第三方平台的备份功能

③ 多活实施效果

客户基于本方案，整合了 TP/AP 场景的多活能力，借助 RDS、PolarDB-X、ADB、DTS、DMS、MSHA 等产品，有效实施了异地多活容灾能力，达到国标 6 级容灾能力要求：

- 针对客户不同的业务模块，实施多种分流策略，自然人电子税务局在线业务实施基于自然人档案号的分流，离线业务清册查询实施按地域分流的多活能力。
- 为客户提供国标 6 级的容灾效果，实现秒级容灾切换并保证数据 0 丢失。
- MSHA 实现多活流量管控和容灾切换动作。
- 客户部署了两单元，常态每个单元承载 50% 的业务流量，充分利用两单元的资源。
- 借助多活管控灵活的流量分配策略，实现重大业务发布时的灰度放量能力。



2 某运营商客服系统

1 案例背景

客户对持续高可用能力有极高要求，其业务特点以 TP 业务为主。

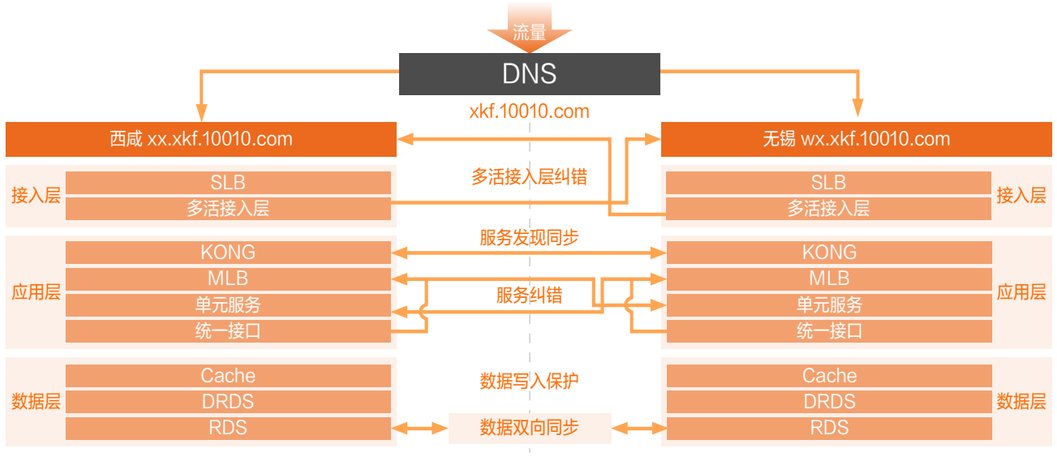
2 业务多活架构

客户基于此方案，整合 RDS、PolarDB-X、DTS、MSHA 产品能力，实现了整个新客服系统 7 个业务中心的多活能力。

- RDS、PolarDB-X 承载业务数据并对接多活管控系统。
- DTS 实现数据的跨城实时同步和状态上报。
- MSHA 实现多活流量管控和容灾切换动作。

3 多活实施效果

- 接入中心、外呼中心、业务支撑等 7 个业务实现按地域多活分流。
- 实现多次容灾演练，对多个省份进行切流，秒级完成切换，数据 0 丢失。
- 客户部署了两单元，常态两个单元均承载业务流量，充分利用两单元的资源。



常见问题解答

Q1. 基于这套方案实施多活，业务是否需要做改动?

A：是的，异地多活是通过对业务流量实施自顶向下的流量隔离实现的，需要业务进行相应的梳理与改造对接流量控制功能。同时在多活场景中为了保障各个单元的数据一致性，需要数据库层面进行相应改造来避免数据冲突等问题。

Q2. 业务如何设计多活分流策略?

A：多活通过流量的隔离实现每个单元对一部分业务流量的自治能力，所以流量的划分策略要重点考虑如果实现流量在单元内的自治。可以从业务的核心维度入手，评估在此维度下相关的业务逻辑是否都能在一个单元内做到闭环。比如淘宝基于买家维度进行流量划分，可以实现 80% 以上的业务逻辑都是围绕买家进行的，一个买家发生的业务行为大部分都是可以在买家归属的单元内实现，不需要跨到别的单元。又或者该客户是按省份划分业务的，因为其业务逻辑本身就是按省自治的，可以实现几乎 100% 的业务流量在单元内闭环。

Q3. 跨地域的数据同步如果存在延迟对业务有什么影响?

A：由于业务已经实现了单元内的自助，换句话说业务依赖的数据全部都在本单元存在，不需要其他单元支撑，所以跨地域的数据同步延迟并不影响本单元的业务行为。仅仅在流量发生切换时存在影响，比如当要把某个用户的流量从单元 A 切换到单元 B，此时这个用户的数据如果存在同步延迟，那么切换后可能会影响此用户的正常业务行为，为此我们的方案中实现了多种切流保护和预检查行为，保障切换时延迟状态可控。

Q4. 双向数据同步是否会造成数据在多个单元间循环复制?

A：不会，DTS 针对多活场景设计了防循环功能，在单元 A 写入的数据同步到单元 B 之后不会再同步回单元 A。

Q5. 容灾切换对业务会有哪些影响?

A：当触发容灾切换动作时，为保证各层级的各个服务节点统一收敛分流规则，会有秒级的业务禁写（仅对涉及切换的流量），之后如果业务不关注同步延迟则会放开业务流量，即完成切换动作，整体耗时在秒级。如果关注同步延迟则会继续应用延迟禁写策略。

参考资料

1 几种容灾方案的对比

容灾方案	容灾能力	资源利用率	架构复杂度
同城容灾	低（1级）	低	低
同城双活	低（1级）	高	低
异地容灾	中（2级-5级）	低	中
多活管控系统	中（2级-5级）	中	中
异地多活	高（6级）	高	高

2 方案相关的数据库产品

RDS：阿里云关系型数据库 RDS（Relational Database Service）是一种稳定可靠、可弹性伸缩的在线数据库服务，提供容灾、备份、恢复、迁移等方面的全套解决方案。

PolarDB-X：PolarDB-X（ 原 DRDS 升级版）是由阿里巴巴自主研发的云原生分布式数据库，融合分布式 SQL 引擎 DRDS 与分布式自研存储 X-DB，专注解决海量数据存储、超高并发吞吐、大表瓶颈以及复杂计算效率等数据库瓶颈难题。

ADB：云原生数据仓库 AnalyticDB MySQL 版（简称 ADB，原分析型数据库 MySQL 版）是一种支持高并发低延时查询的新一代云原生数据仓库，全面兼容 MySQL 协议以及 SQL:2003 语法标准，可以对海量数据进行即时的多维分析透视和业务探索，快速构建企业云上数据仓库。

DTS：数据传输服务 DTS (Data Transmission Service) 支持关系型数据库、NoSQL、大数据 (OLAP) 等数据源间的数据传输。它是一种集数据迁移、数据订阅及数据实时同步于一体的数据传输服务。

DMS：数据管理 DMS 是一种集数据管理、结构管理、用户授权、安全审计、数据趋势、数据追踪、BI 图表、性能与优化和服务器管理于一体的数据管理服务。

附录

术语表

术语	说明
容灾	容灾系统是指在相隔较远的异地，建立两套或多套功能相同的系统，系统之间可以相互进行健康状态监视和功能切换，当一处系统因意外如，火灾、洪 水、地震、人为蓄意破坏等停止工作时，整个应用系统可以切换到另一处，使得该系统功能可以继续正常工作。
分流标	对业务流量进行划分的维度标识，比如按照省份进行流量划分则分流标可以设置为"省份id"。
写保护	对单元的写行为进判断，判断是否应该在本单元执行，如果不是则不允许写入，从而达到保护单元的效果。
单元	据业务特点在逻辑上分成几个逻辑数据中心（LDC），我们给其命名为“单元”，核心业务在这个数据中心实现自流转。
中心	特殊的单元，是一些长尾没有做单元化改造的业务和一些强中心无法进行单元化改造的业务所在的数据中心。
多活管控系统	实时多活架构部署和容灾切换的管控系统，由MSHA产品提供对应功能。
单向同步	异地多活架构中对数据做从中心到单元的数据同步。
双向同步	异地多活架构中对数据做从中心到单元以及单元到中心的数据同步。
数据防循环	在双向同步场景，一行数据写入单元A后被同步到单元B，不能再从单元B同步回单元A。
数据质量	指在多活场景中，因为数据多点可写和数据双向同步带来的各单元数据不一致问题，保障数据质量既是保障数据一致性。
单元闭环	整个业务流程中，上下游模块的调用均能在单元内完成，无需或尽可能少的访问其他单元。

5 服饰行业
数据库解决方案

随着经济的高速发展，品牌服饰行业增长迅速。一方面，消费者崇尚品牌化、高端化、个性化的需求越来越明显，促使传统服饰品牌商和互联网电商渠道迅速融合，线上渠道业务快速增长。另一方面，品牌商之间的竞争激烈，国内 TOP 服饰品牌商逐步向多元化和国际化发展，并购其他国内和国际品牌，线上线下渠道高速拓展。这些都对业务系统建设提出了较高的要求，传统的割裂式 IT 系统已经不能满足业务的高速发展带来海量高并发业务处理需求和业务快速创新的诉求。基于互联化架构和技术的信息化转型成为服饰行业品牌商的不二之选。阿里云旨在帮助服饰品牌客户实现基础设施云化、全触点数字化、核心业务在线化、运营数据化、决策职能化，其中数据库起到了核心作用。本解决方案介绍如何通过阿里云数据库助力客户构建上下游数据打通方案、亿级订单和千家门店支撑方案、大促弹性解决方案、业务报表加速、新开门店和新渠道接入等方案，普惠云上云下的服饰品牌客户。

服饰行业典型业务场景

经历多次演变，服饰行业的零售迎来了新一次的变革，新零售时代已经到来。随着服饰行业的的变革和发展，服饰品牌商和消费者都发生了巨大的变化。品牌商还面临行业和消费者变革带来的业务痛点：客户流失；库存失衡；利润低迷；销售瓶颈。为了解决这些痛点，服饰行业需要做 IT 零售转型，品牌商需要开始转变思路适应消费者变化。由信息化企业到数字化企业，从业务积累数据到数据驱动业务发展的转变，需要更加灵活多变的 IT 架构，把企业变成以信息、数据来驱动的企业。

总体来讲，服饰行业典型业务场景包括：

1 上下游数据打通

传统服饰品牌商的业务链路较长，从门店或电商渠道售卖到会员、仓储、物流、供应链、财务等系统整体数据链路较长。不同渠道、相同渠道不同环节数据相对割裂，无法形成渠道数据打通、上下游数据的打通。对于会员管理、商品管理、调拨管理、采购管理、订单管理、支付管理、库存实时管理和优化、运营决策等环节非常不利，无法提升业务效率，需要打通上下游数据的统一存储和高效使用。同时运营、报表类数据和业务数据也是割裂状态，需要通过传统的文件、接口方式将数据传输至报表和运营系统，导致运营决策不能实时化。

2 亿级订单以及千家门店支撑

大型服饰品牌商在业务发展趋势好的情况下会考虑扩展门店和线上渠道，尤其是服饰 TOP 品牌客户门店数量大并且订单数量大，部分品牌订单能到亿级，对海量数据存储、高并发写和扩展性有较高要求。

3 快速门店新渠道接入

服饰品牌商业务发展较快，当业务快速发展，短期内需要新开门店和线上电商渠道增加等业务场景。比如线下扩展 200 家门店或者新接入了微商或京东等电商渠道，需要在短时间内 (比如 1–2 天) 提升数据库系统的读写、存储、扩展能力以支撑快速开店、渠道接入和后续的业务持续增加。

4 业务报表加速

服饰行业业务节奏快，需要通过各类报表进行业务决策，通过云原生数据仓库帮门店管理、库存管理、采购管理、销售管理、调拨管理、财务管理、商品进销存等报表提速，提高业务决策效率。服饰行业客户业务数据在业务高速发展下，数据不断增加，核心的订单、销售、结算数据体量可能会达到千万级、亿级甚至更多，用户画像、业务报表、运营决策报表等性能成为业务业务开展和业务决策的瓶颈。

5 大促营销

服饰品牌商平时业务比较平稳，订单量和支付量相对比较平稳，通过常规容量评估和资源扩容可以满足业务发展。但是在 618、双十一等大促期间会有业务高峰，订单量、支付量会是平时的几倍到几十倍，在大促结束后，又需要将资源使用量从高配置降低到原有配置，确保资源充分利用和有效控制 IT 成，对数据库峰值 TPS、QPS 处理能力和灵活快速的弹性能力有较高要求。同时企业希望在起步阶段 IT 投入不要太大。

服饰生产	服饰订货	渠道销售	业务平台	仓储物流
ERP系统		快速开店 新渠道接入	上下游数据 打通方案	第三方WMS
		大促弹性升降配， 节约IT成本	亿级订单千家 门店支撑	
			业务报表加速	

解决方案

1 整体业务架构

数据库主要支撑的部分是业务中台的各业务中心、数据同步链路、数据中台对应的报表、运营、决策分析数据。数据库支撑数据中台的数据存储和数据加速层，支撑客户的报表、分析平台和大数据平台等。服饰行业整体业务架构如下所示。

2 上下游数据打通解决方案

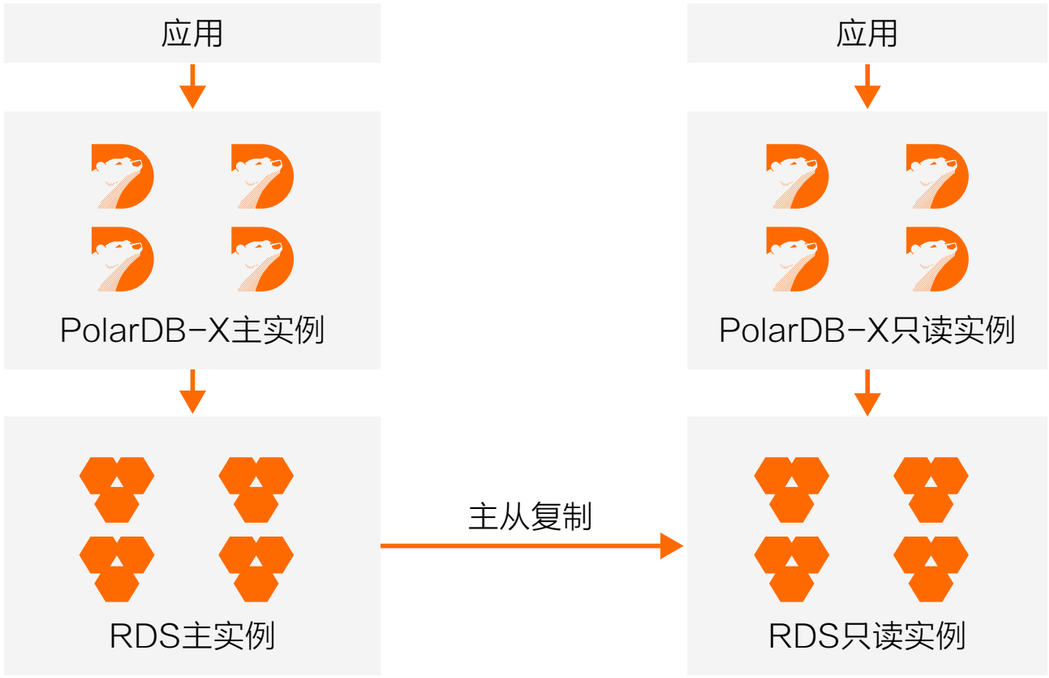
1 方案概述

借助业务中台能力可以将订单、支付、库存、物流、仓储、供应链、财务系统数据实时打通，实现业务数据化，快速响应市场变化、降低 IT 投入成本、提升销售额。云原生分布式数据库 PolarDB-X，有较强的读写能力和海量数据存储能力，能很好的适配业务中台架构，在业务中台构建中起到核心作用，用于承载用户中心、商品中心、交易中心、库存中心、订单中心、物流中心、报表中心等核心业务中心的数据存储。品牌商门店和电商渠道业务量大的情况下，对数据库的性能、容量、存储能力、弹性能力提出较高的要求。借助数据传输服务 DTS 可以将业务数据实时同步至运营和决策报表数据库，提升运营决策能力。借助云原生数据库和 DTS 构建的业务中台方案，可以帮助服饰行业客户解决数据割裂问题，保证业务的快速运行和决策。

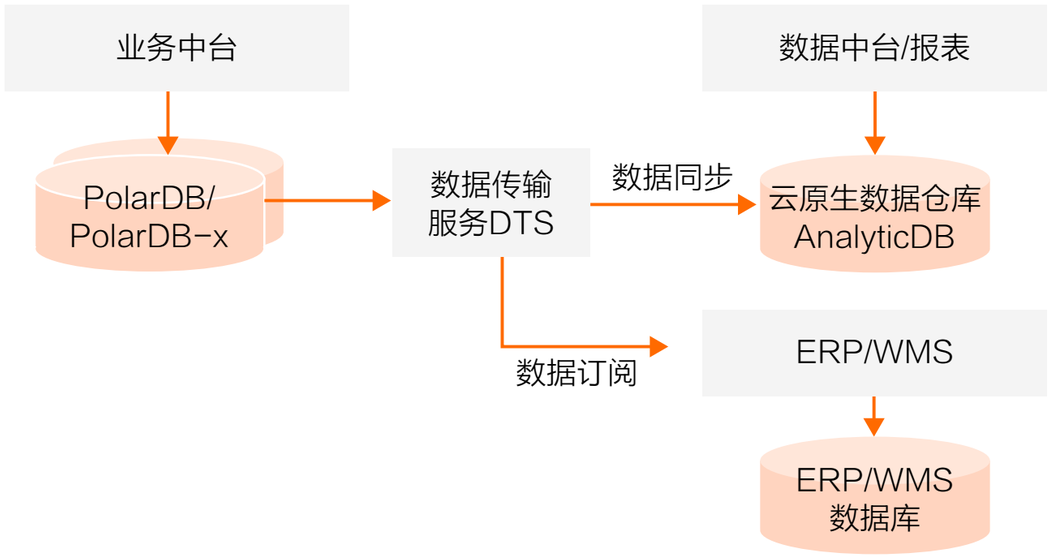
- 会员数据，用于存储门店、电商渠道的会员信息，对数据库主要诉求是数据存储的一致性和安全性，确保品牌的会员信息准确。
- 商品数据，用于存储品牌商所有商品的信息，对数据库主要诉求是数据存储的一致性和安全性，以及实时更新能力。
- 库存数据，用于存储服饰商品库存数据，对高并发写有诉求，同时在大促期间有热点写的场景，通过库存中心打通门店销售、电商销售和库存中心的实时交互，确保库存准确，以指导门店和电商渠道进行库存盘点和补货。
- 订单数据，是服饰品牌数据量最大的业务数据，对数据库高并发写、写扩展、海量存储、存储扩展都有强诉求，在大促期间，订单数据可能是是平时的 3-10 倍，甚至更多。

2 方案架构

PolarDB-X（原 DRDS 升级版）是由阿里巴巴自主研发的云原生分布式数据库，融合分布式 SQL 引擎 DRDS 与分布式自研存储 X-DB，专注解决海量数据存储、超高并发吞吐、大表瓶颈以及复杂计算效率等数据库瓶颈难题，历经各届天猫双 11 及阿里云各行业客户业务的考验，助力企业加速完成业务数字化转型。



数据传输服务 (Data Transmission Service) DTS 支持关系型数据库、NoSQL、大数据 (OLAP) 等数据源间的数据传输。它是一种集数据迁移、数据订阅及数据实时同步于一体的数据传输服务。数据传输致力于在公共云、混合云场景下，解决远距离、毫秒级异步数据传输难题。通过 DTS 可以将业务系统的数据实时同步至报表数据库，实时打通业务系统和报表系统以及服饰商品生产 / 仓储 / 物流系统。



③ 操作流程及关键步骤

- 业务压力评估，根据评估结果购买合理规格的 PolarDB-X。
- 通过数据导入或者 DTS 将现有数据库的数据迁移至 PolarDB-X。
- PolarDB-X 架构设计，设计阶段需要考虑分库数量和分表数量，并控制单表数据量（建议单表 500 万）（关键步骤）
- PolarDB-X 改造和适配，适配需要做业务改造，在建表、SQL 兼容性、分布式查询、分布式事务方面要做改造。（关键步骤）
- 业务量和存储量增长后，发现数据库有超负载风险或已经超载，控制台发起 PolarDB 升配或扩增只读节点，或者在控制台发起 PolarDB-X 的平滑扩容和升配。（关键步骤）
- 业务平稳运行后，评估数据库压力较低情况下，发起 PolarDB 降配或削减只读节点，或者在控制台发起 PolarDB-X 的降配。
- 通过阿里云控制台采购数据同步任务，即可支持从 PolarDB/PolarDB-X 的到云原生数据仓库 AnalyticDB 的数据同步，如果下游是其他类型数据库，可以通过 DTS 订单数据订阅能力支持数据实时传输至下游数据库。

④ 优势解读

服饰品牌 TOP 客户订单量、会员量较大，千万级或者亿级，业务链路较长，对高并发写和实时更新有较强诉求，需要具备海量数据存储和高并发写，并具备一定扩展性的数据库存储产品。

云原生分布式数据库 PolarDB-X 专注解决海量数据存储、超高并发吞吐、大表瓶颈以及复杂计算效率等数据库瓶颈难题；在业务发展初期可以选择小规格 PolarDB-X，在业务发展后可以快速升配 PolarDB-X 到高规格，单集群最大可支持 300 万混合读写 QPS，600 万读 TPS。扩容、升配对业务影响较小。并且具备分布式查询、分布式事务能力，确保在数据拆分后的正常聚合读写。

- 支撑亿级甚至十亿级订单，上千家门店业务的高效读写，保证门店、电商渠道业务顺畅。
- 在保持性能稳定、系统稳定的情况下支撑门店和渠道流量的快速扩展。
- 通过 DTS 提供上下游系统稳定、实时的数据访问和数据传输同步能力。

3 亿级订单以及千家门店支撑解决方案

1 方案概述

服饰品牌商核心业务数据量大，对读写并发要求高，希望能持续无缝扩展，云原生分布式数据库 PolarDB-X 具备这样的能力。PolarDB-X 支持垂直拆分和水平拆分，通过垂直拆分可以支撑不同的业务中心的数据在拆分后可跨库访问，通过水平拆分可以将订单、交易流水的海量业务数据拆分至不同的数据库实例上，提高数据库的高并发读写能力和读写、存储的扩展能力，可以支持 PB 级数据存储，百万级 TPS 和千万级 QPS 混合读写。

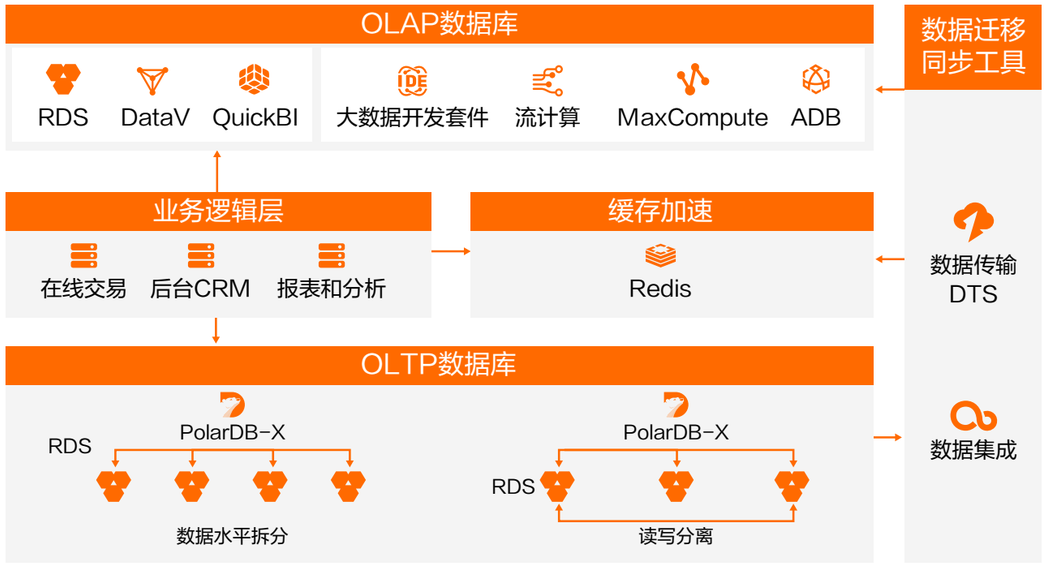
2 方案架构

PolarDB-X 数据库架构：

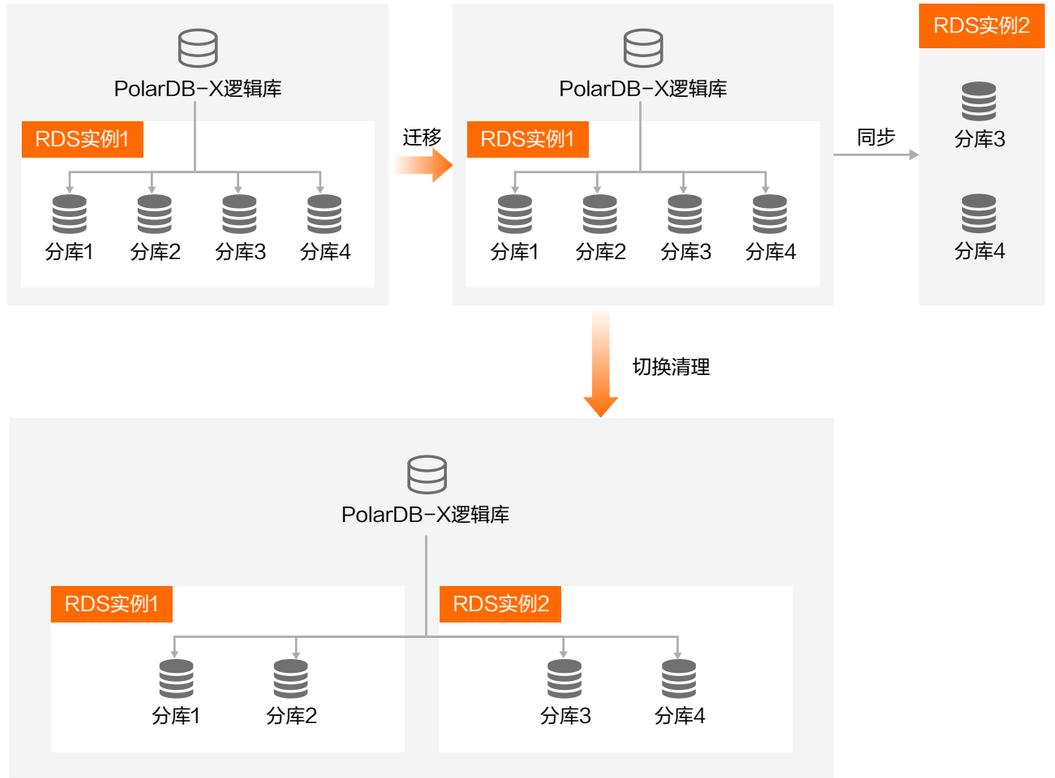
高并发实时交易场景面向客户端的电商、金融、O2O、零售等行业普遍存在用户基数大、营销活动频繁、核心交易系统数据库响应日益变慢的问题，制约业务发展。DRDS 提供线性水平扩展能力， 能够实时提升数据库处理能力， 提高访问效率， 峰值 TPS 达 150 万 +， 轻松应对高并发的实时交易场景。

- 事务支持。单分库严格事务支持，并提供 XA 强一致事务、最终一致性事。
- MySQL 兼容性。兼容 MySQL 交互协议和大部分 SQL，让业务使用更加平滑。
- 大规模。通过分库分表，提供可扩展的服务容量和存储容量。

高并发实时交易场景



PolarDB-X 通过平滑扩容可以将现有数据库实例的中的数据迁移至更多的数据库实例，从而实现存储和读写的水平扩展。见下图：



③ 操作流程及关键步骤

PolarDB-X 计算节点升降配流程：

- 在 PolarDB-X 控制台发起升配流程，将计算节点升配至高配。
- PolarDB-X 计算节点升配会有负载均衡，降配会强制断开流量，需要考虑业务运行期间闪断影响是否可接受。（关键步骤）
- 当前的 PolarDB-X 计算节点和存储节点是分离的，需要分别进行升降配和平滑扩容。（关键步骤）

  升配会逐步做负载均衡，对业务影响很小。
- 如果是跨系列升配，会有闪断。
- 完成升配观察业务使用情况，是否读容量和写容量有提升。
- 业务高峰结束后在 PolarDB-X 控制台进行降配处理，降配会导致闪断，如果业务允许建议在业务低峰做。

PolarDB-X 数据库平滑扩容流程：

- 在 PolarDB-X 控制台发起平滑扩容，增加数据存储的 RDS 或者 PolarDB。
- 平滑扩容最后阶段需要做切换，切换会对访问有一定影响，建议在业务低峰做。

完成平滑扩容后，扩容的数据会分布在新的 RDS 或者 PolarDB，这部分数据的访问压力会由新的实例承载。

业务高峰结束后在 PolarDB-X 控制台进再次进行平滑扩容，可以将扩容出来的数据库再平滑扩容回原有的实例，平滑扩容切换会对业务有影响。

完成再次平滑扩容后，使用的 RDS 或 PolarDB 实例数下降，可以节约 IT 成本。

④ 优势解读

- 该方案支持高并发读写，理论上可以支持百万级 TPS 写访问，千万级 QPS 的读访问。
- 数据存储容量大，PolarDB-X 可以存储 PB 级数据，在业务访问和数据库架构设计较好的情况下可以不需要做数据归档，数据库存储类型如果为 PolarDB 则 PolarDB-X 理论上无需扩展存储。
- 扩展性好，可以分钟级从较小规格将计算规格 (16 核 64GB 内存) 扩展至较大规格 (1024 核 4096GB 内存)，可以在小时级将数据存储实例从 2 个扩展至 16 个。

4 新开门店和新业务渠道接入解决方案

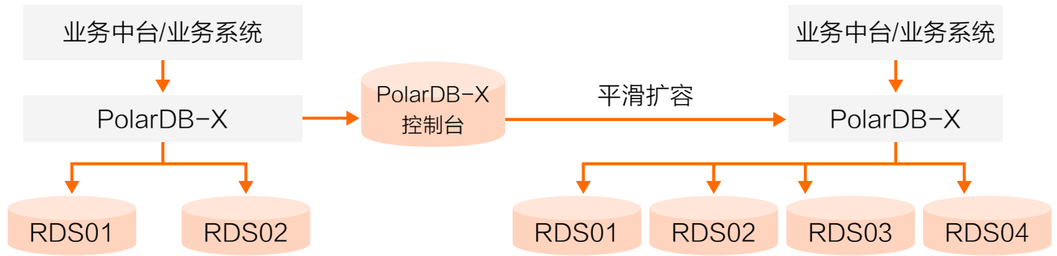
① 方案概述

云原生分布式数据库 PolarDB-X 通过平滑扩容能力可以快速支持数据读写、存储、并提升扩展能力。PolarDB-X 在架构设计阶段通过分库分表水平拆分，可以将数据分布在少量（比如 2 个）PolarDB/RDS 数据库实例上，在业务需要较大的数据库容量的时候，通过平滑扩容可以将现有的数据库实例从 2 个扩展到最多 16 个，从而提高整个数据库系统的读写、存储、扩展能力。

② 方案架构

客户在新开门店和新业务接入场景下，有数据库快速扩容提升计算能力存储能力的诉求，需要扩展性好和快速扩展的数据库。

PolarDB-X 平滑扩容架构图如下所示：



③ 操作流程及关键步骤

PolarDB-X 计算节点升降配流程：

- 在PolarDB-X控制台发起升配流程，将计算节点升配至高配。
- PolarDB-X 升降配会有闪断，需要考虑业务运行期间闪断影响是否可接受。
- 升配会逐步做负载均衡，对业务影响很小。
- 如果是跨系列升配，会有闪断。
- 完成升配观察业务使用情况，是否读容量和写容量有提升。
- 业务高峰结束后在 PolarDB-X 控制台进行降配处理，降配会导致闪断，如果业务允许建议在业务低峰做。PolarDB-X 计算节点升配会有负载均衡，降配会强制断开流量，需要考虑业务运行期间闪断影响是否可接受。（关键步骤）

注意：当前的 PolarDB-X 计算节点和存储节点是分离的，需要分别进行升降配和平滑扩容。（关键步骤）

PolarDB-X 数据库平滑扩容操作流程：

- 在PolarDB-X控制台发起平滑扩容，增加数据存储的 RDS MySQL。
- 平滑扩容最后阶段需要做切换，切换会对访问有一定影响，建议在业务低峰做。
- 完成平滑扩容后，扩容的数据会分布在新的 RDS，这部分数据的访问压力会由新的实例承载。
- 业务高峰结束后在 PolarDB-X 控制台进再次进行平滑扩容，可以将扩容出来的数据库再平滑扩容回原有的实例，平滑扩容切换会对业务有影响。
- 完成再次平滑扩容后，使用的 RDS 实例数下降，节约了 IT 成本。

④ 优势解读

- 扩展性好，分钟级从较小规格将计算规格（16 核 64GB 内存）扩展至较大规格（1024 核 4096GB 内存），可以在小时级将数据存储实例（RDS 或者 PolarDB）从 2 个扩展至 16 个。
- 升降配和平滑扩容对系统影响小，PolarDB-X 升配计算节点可以做到自动负载均衡，平滑扩容切换有短时间闪断。PolarDB-X 降配和删除计算节点会影响缩容节点的访问，对其他节点流量无影响，存储降配和平滑扩容闪断。

5 业务报表加速解决方案

① 方案概述

云原生数据仓库 AnalyticDB MySQL 版可以有效提速复杂报表生成，提升数据运营能力，实时指导业务。ADB 是简单易用的 SQL 数据仓库，能实时洞察数据价值，助力企业完成实时运营转型，可以在帮助企业提速业务场景。

各类业务报表

门店管理、门店销售、门店库存、库存管理、采购管理、销售管理、调拨管理、闪屏管理、财务管理、商品进销存等报表成为指导业务的重要决策工具，对于门店管理、销售管理和决策管理来说非常重要。传统的关系型数据库在报表的时效性和性能方面效果一般，部分复杂报表需要几十秒、十几分钟才能出结果，或者在有些极端场景下跑不出结果。

用户画像分析、用户偏好分析、产品创新预测、客流预测

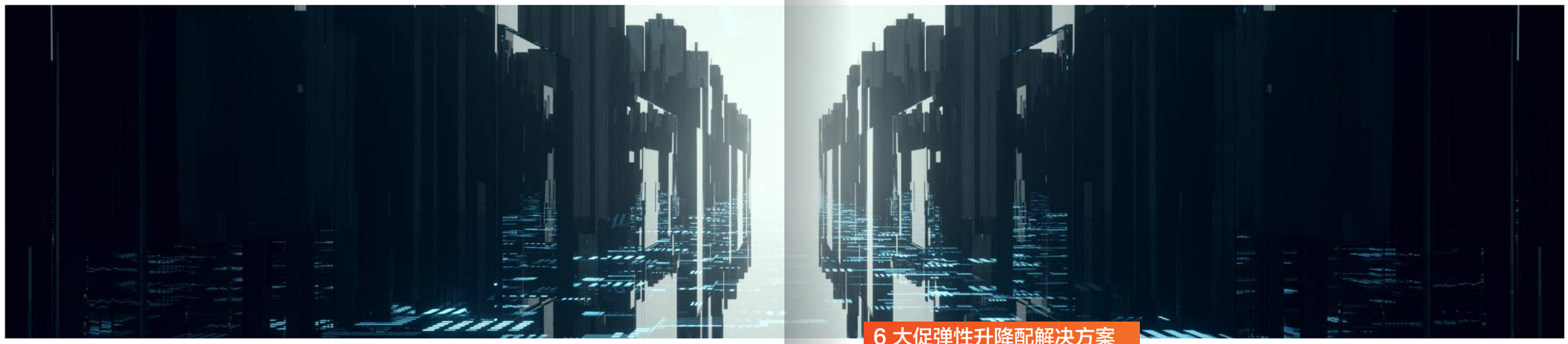
利用 AnalyticDB 可以有效提速用户画像、用户偏好、客流预测等业务场景的返回，协助业务团队进行精准营销。

利用快速分析能力助力战略决策

服饰企业管理者、决策者在面对多种营销链路与产品部署的情况下，往往需要快速发现问题并给出决策意见，以便能够及时调整业务方向，使得整体业务快速前进。AnalyticDB 的快速分析能力结合自助数据决策分析工具具备“高效性”“稳定性”以及“灵活性”三个特性，通过系统预设的分析模板或自助能力，业务人员可以快速聚焦于业务中的潜在问题。

② 方案架构

通过 DTS 将业务数据库 PolarDB、PolarDB-X 的业务数据同步至 AnalyticDB，报表和运营系统通过连接数据库的方式使用 AnalyticDB 可以完成以及数据复杂报表秒级返回。



3 操作流程及关键步骤

- 在云控制台采购 AnalyticDB 集群，并创建数据库、用户名、密码。
- 利用 DTS 构建从关系型数据库 RDS MySQL/PolarDB MySQL/PolarDB-X/ 自建 MySQL 同步数据至 AnalyticDB 的同步链路，快速将业务数据同步至 AnalyticDB。或者通过数据导入工具将已有的报表数据导入 AnalyticDB。
- 业务库到 AnalyticDB 同步数据的过程是关键节点，要确保数据全量和增量同步正常后方可正常使用。（关键步骤）
- 应用服务根据之前设定好的用户名、密码、库名以及 AnalyticDB 的访问地址，以数据库的方式即可访问 AnalyticDB，开启报表业务的实时访问和快速返回。
- 如果因为业务访问量和容量增加导致 AnalyticDB 集群的计算容量、存储容量不够，可以在控制台通过升配解决，升配期间会有闪断，建议在低峰期进行。AnalyticDB 升配可能导致访问异常，需要在业务低峰进行。（关键步骤）

4 优势解读

- 服饰品牌客户业务链路较长，需要门店、库存、采购、销售、调拨、进销存等报表来为业务提速，提高业务效率，对报表快速返回、报表数据库的扩展性和易用性有较高诉求。
- 在线分析报表秒级返回，保证报表的时效性，性能是传统关系型数据库的 5–10 倍。
- 离线分析报表分钟级返回。
- 存储和计算无限扩展，聚合分析不再是难题，可轻松扩展至 PB 级存储。
- 简单易用，全面兼容 MySQL 协议和 BI 工具。

6 大促弹性升降配解决方案

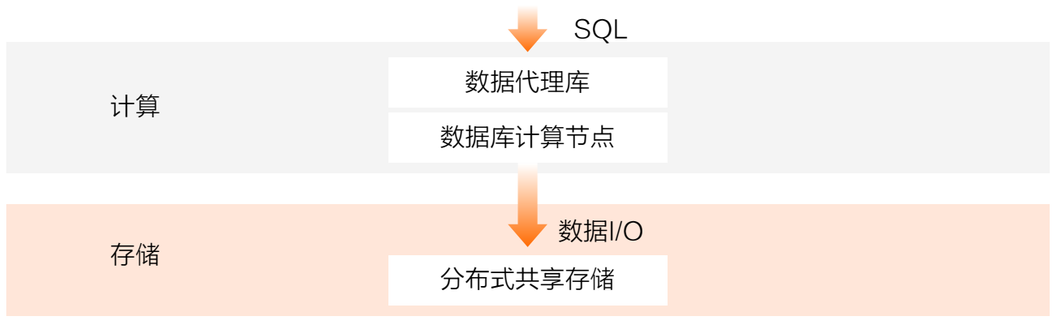
1 方案概述

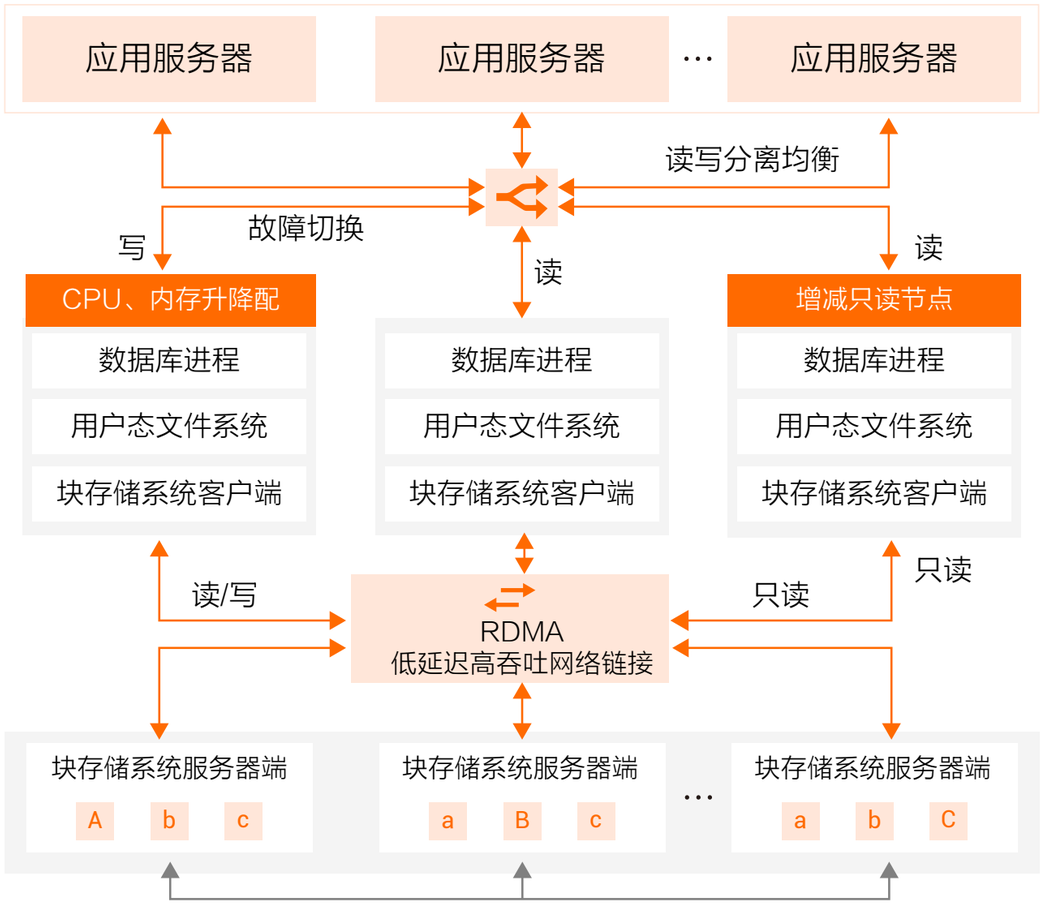
服饰行业大促的业务特征对数据库升降配能力和快速弹性升降配能力提出了较高要求，云原生数据库 PolarDB 具备高弹性能力和快速弹升能力，能很好的适应服饰行业的大促业务诉求。

当业务数据量和写入并发相对不高的时候，选择云原生数据库 PolarDB 即可满足日常业务需求，在大促前或者大促期间，可以通过增加读节点和升配主节点配置来满足业务对读并发能和写入并发能力的诉求。PolarDB 可以在 5 分钟左右完成增加读节点，并在 15 分钟左右完成主节点升配。最新版本的 PolarDB MySQL 8.0 最高支持 2.5 万 TPS，单集群可以增加 15 个从库，最高支持 800 万的读写 QPS，单集群最多存储 50TB 数据。

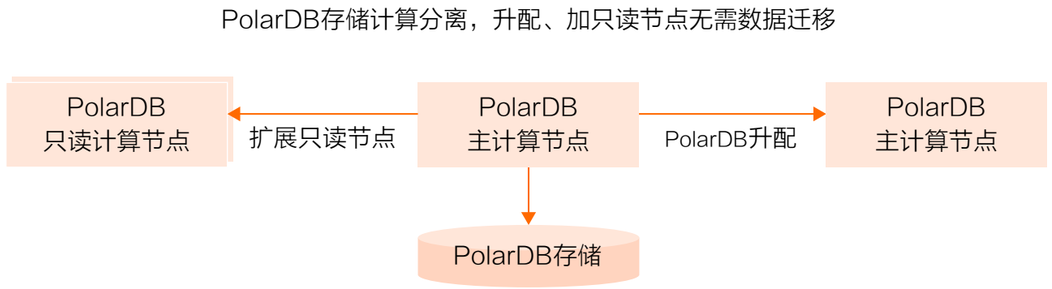
2 方案架构

PolarDB 是阿里巴巴自主研发的下一代关系型分布式云原生数据库，目前兼容三种数据库引擎：MySQL、PostgreSQL、高度兼容 Oracle 语法。计算能力最高可扩展至 1000 核以上，存储容量最高可达 100T。经过阿里巴巴双十一活动的最佳实践。PolarDB 技术架构如下：





PolarDB 快速弹升架构图：



③ 操作流程及关键步骤

PolarDB 数据库升降配流程：

- 在 PolarDB 控制台发起增加节点、升配操作。
- 如果是增加节点，在增加完节点后需要将节点加入集群地址后可以使用，如果是升配等升配完成后即可使用。升配切换会有闪断，如果业务允许，建议在业务低峰做。
- 完成增加节点或升配后观察业务使用情况，是否读容量和写容量有提升。业务高峰结束后在 PolarDB 控制台进行删除节点处理和降配处理，删除节点需要先将节点从集群地址摘除，降配会导致闪断，如果业务允许建议在业务低峰做。
- PolarDB 升降配会有闪断，需要考虑业务运行期间闪断影响是否可接受。

④ 优势解读

- 快速弹升和快速降配，支撑流量洪峰和控制 IT 成本。PolarDB 可以做到 5 分钟完成读容量的扩容，15 分钟完成写容量扩容。
- 数据存储容量大，PolarDB 可以支撑 50TB 数据存储。
- 支持临时升配，可支持小时级临时升配，充分帮企业节约 IT 成本。
- 弹升对业务影响较小，PolarDB 升配期间会闪断，对业务影响较小，增加节点后通过集群地址自动负载均衡，对业务基本无影响，存储按需使用，无需扩展。降配缩容时间段，影响小，降配期间会闪断，对业务影响较小，删除节点可以先从集群地址去掉节点，应用无感知。

方案优势

1 高弹性架构

企业客户在遇到各种大促的时候，轻松应对各种流量洪峰压力。

2 降低成本

在流量洪峰前分钟级升配，在洪峰过后，分钟级降配，数据库资源初始投入成降低，同时减少企业在大促期间的 IT 成本。

3 灵活扩容或缩容

基于共享存储的计算存储分离架构，能快速增加节点（分钟级），对于读多写少的业务，能根据读流量的情况来快速增加和删除读节点，灵活控制数据库集群的 QPS 和成本。

客户案例（特步）

1 业务背景

特步目前有数千家门店，在信息化转型之前有以下痛点：

① 应用层面支持业务需求变革缓慢

- 应用操作繁琐、全端笨重
- 加盟店管控不够细致
- 不同应用使用差异大，用户反馈体验不好

② 无法支撑时效性强的分析

- 对时效性要求高的部门工作量大，靠人工对接数据
- 财务监控处报表困难、耗时长
- POS 完全离线模式，数据上传下载问题多，小票无法及时上传，库存无法实时更新增减

③ 无法满足集团转型诉求

- 无法满足全渠道营销，线上线下一体化的深层会员关系营销
- 业务支持单一，无法支持多元业务需求
- 无法支持多主题、跨级业务结算

④ 数据库使用痛点明显

- 海量订单处理和扩展能力不足
- 上下游数据不能打通，库存不能实时更新
- 报表数据和业务数据不通，不能出实时报表，部分报表运行慢
- 数据库运维成本高，弹性能力不足，不能满足业务的快速变化和大促流量

2 业务架构

基于特步现有 IT 架构痛点，阿里云架构团队给客户推荐了基于业务中台的整体架构。

在特步使用了阿里云提供的业务中台方案和通过分布式数据库、分析型数据库、数据传输服务 DTS 来支撑业务后，帮客户解决了很多痛点，获得以下技术收益。

① 上下游数据打通

通过 PolarDB-X 分布式数据库解决方案支撑客户业务中台系统，打通生产、销售、渠道、库存、订单、财务等不同系统，使客户的O2O全渠道业务数据完全打通，同时结合 DTS 打通了客户的交易业务和报表业务，满足实时分析诉求。

分布式数据库解决方案业务中台，打通客户准实时库存同步能力，指导客户实时调整的库存和供应链。

② 满足以及订单、千家门店业务支撑

通过 PolarDB-X 给特步提供较强的业务和数据扩展能力，通过 PolarDB-X 解决客户海量订单高并发读写和大促业务高峰的诉求，O2O 订单量提升 10 倍以上，数亿订单高性能支撑，同时支撑了特步数千家门店的业务顺利开展。双十一高峰平稳高性能支持特步单日百万级订单的业务读写。

③ 满足大促弹性诉求

PolarDB-X 的平滑扩容和 ADB 的集群架构都具备平滑扩容能力，只需要很短时间（分钟 – 小时级）和很低的人力成本可以提升客户信息系统的扩展性，快速扩展至 5-10 倍当前系统容量。云数据库 RDS 的快速升降配和 PolarDB-X 的平滑扩容，使得客户的具备了快速弹性扩容能力，远低于传统 IDC 的弹升周期。

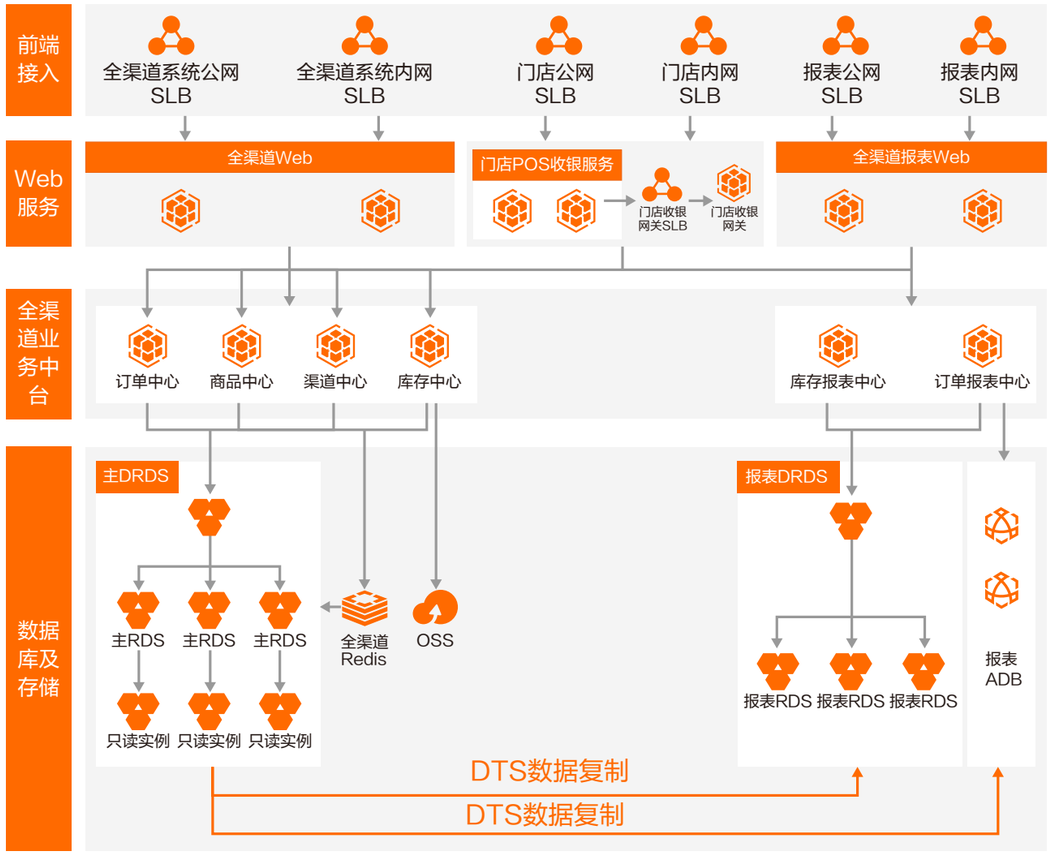
④ 为业务报表提速并保证时效性，提升数据化运营能力

通过数据传输服务 DTS 打通了客户的交易数据和报表数据，帮客户打通核心数据和运营行为，通过运营、销售报表指导运营行为。通过云原生数据仓库 AnalyticDB 的快速分析能力发现客户爆款和滞销商品，分析客户行为，达到精准营销的目的，提升运营能力，降低运营成本。

⑤ 保证销售渠道和门店的库存实时扣减

通过 PolarDB-X 构建业务中台库存中心，门店库存实时扣减，确保库存实时性和统一管理

针对特步客户的业务痛点和整体技术架构设计数据库架构如下：



4.3 业务效果

在获取技术收益的同时也给客户提供了较好的业务价值。

- 支持特步双十一单日 200w-300w 的订单处理，具备单日千万级到亿级订单的读写、存储扩展能力。
- 利用 PolarDB-X 的平滑扩容能力和 ADB 集群扩展能力、云管控快速弹升能力，可以在小时级将系统容量提升到当前容量的 5-10 倍。
- 门店、采购、销售订单、库存、调拨、进销存、财务等报表大部分耗时，从十几分钟到几十分钟降低至 1 秒至 1 分钟。

附录

术语表

术语	解释
RPO	恢复点目标，在故障发生后，数据能够恢复到的时间点，也反映着业务所能容忍的数据丢失量
RTO	恢复时间目标，在故障发生后，数据恢复到预期状态所能承受的最长时间
HA切换	数据库的高可用特性，比如数据库是主备架构，平时只有主实例对外提供服务，备实例从主实例同步数据，一旦主实例故障，HA切换后备实例代替主实例对外提供服务，保证数据库服务连续性。
DAU	日活跃用户数。
TPS	Transactions Per Second，意思是每秒事务数，具体事务的定义，都是人为的，可以一个接口、多个接口、一个业务流程等等。一个事务是指事务内第一个请求发送到接收到最后一个请求的响应的过程，以此来计算使用的时间和完成的事务个数。
QPS	Queries Per Second，意思是每秒查询率，是一台服务器每秒能够响应的查询次数（数据库中的每秒执行查询SQL的次数）。

6 用电信息采集和主站应用系统

数据库解决方案

国家电网公司智能电网建设工作正在快速推进，电网中引入数量庞大的智能电表、电网友好型设备等终端，以实现电网侧与用户侧的双向互动。用电信息采集系统（以下简称“用采系统”）作为国家电网公司提出的“SG186 工程”营销业务应用系统的重要组成部分，是智能电网下营销自动化 / 双向互动应用的核心。用采系统通过对终端用户用电信息的采集、处理和分析，能够为电力营销的自动化抄表、负荷管理、线损分析、费控管理、错峰用电、防窃电等业务提供有效的数据支撑。同时用采系统为运检、安质、发策等不同部门提供营销数据服务，是支撑新型电力营销业务的核心系统。

随着电力用户数量的增加和智能电表的普及，用采系统对电量数据完整性、数据采集频率，数据精度都提出更高要求，并且与外部业务系统的数据交互和共享也日益复杂，基于传统 IOE 架构的用采平台已无法满足日益增长的业务需求。阿里云推出基于云计算技术的电力行业用电信息采集和主站应用系统数据库解决方案，采用分布式数据架构构建新一代用采平台，将为电力营销业务创新提供更为坚实的数据保障。

电力用采系统的现状

电力用采系统大多始建于 10 年前，采用传统 IOE 架构。设备数据采集、电量电费计算、报表统计查询等均基于 Oracle 数据库完成，按照百万级别设备数，每天一次的采集频率进行设计。当前电网中的电力智能终端设备已经非常普及，省级的设备数已达数千万，采集频率提升至每 15 分钟一次，爆发式的数据增长已经让原有系统不堪重负，架构缺陷也日益凸显：

1 烟囱式设计

受制于集中式架构的技术限制，无法集中处理全省的大数据量计算和接入，不同的业务团队只能针对部分数据进行独立开发，系统建设架构按照烟囱式进行设计。

2 扩展性不足

传统架构横向扩展性弱，在采集频率、采集指标大幅提升的情况下，架构上无法支撑全量数据采集，仅能采集数据质量较高的部分数据，通过牺牲数据规模来换取系统稳定性。

3 并发和实时处理能力弱

海量数据高并发采集导致采集过程处理能力存在瓶颈，并发速度难以满足要求，数据入库慢导致大量数据积压；在变压器、计量设备监控等业务场景中，需要对采集数据的有效性、所属台区的合法性等进行数据核查校验。传统架构下只能在采集数据入库之后进行，无法实现数据实时核查，无法满足实时性要求较高的业务需求。目前主站大部分统计类数据均基于前一天历史数据分析，无法做到实时分析现场异常情况。

4 无法自主可控

系统组件大部分使用国外产品和技术，不可控，也存在技术风险。

解决方案（用电信息采集业务场景）

用电信息采集系统涉及设备数据的产生、处理、存储、分析、应用整个完整流程，主要包括前置数据采集、数据存储、数据计算与分析、主站应用等部分。

- 前置数据采集：通过 HPLC 智能电表、智能终端、集中器等设备，采集客户用电量、电流电压、开关和设备工况、终端和电表记录、预付费等数据信息。采集频率为 15 分钟采集一次，单个设备会采集电流、电压等不同指标数据。
- 数据存储：根据实时性、存储量、访问方式的不同特点，将数据分为结构化应用数据、实时采集类数据和缓存数据三部分，使用关系库、列式数据库和缓存分别存储。
- 数据计算与分析：为电量计算、线损分析、电量排名、完整率计算等计算任务，以及费控管理、统计查询等分析任务提供计算和分析能力，对外提供数据接口。
- 主站应用：依据数据层提供的计算结果和服务接口，提供面向终端用户的档案管理、配变检测分析、终端管理、电量分析、负荷分析等多种用采系统主站业务应用。

1 系统架构

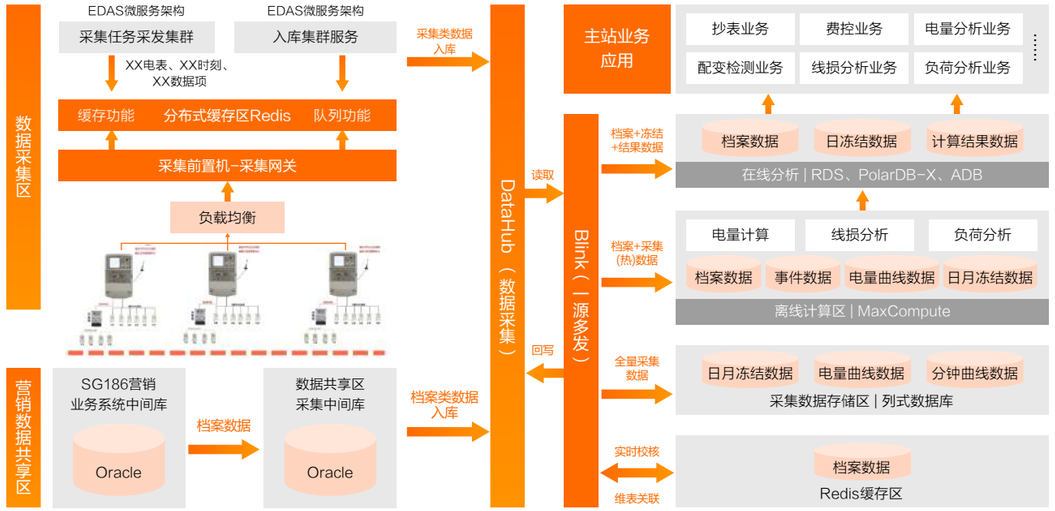
① 总体架构图

用采系统从总体功能架构上主要分为前置数据采集、数据存储、数据计算与分析、主站应用四部分。前置数据采集涉及智能设备的数据读取和上报，电力行业已有成熟方案；数据存储、数据计算与分析部分，是基于数据中台进行架构设计的重点，通过数据分类、分布式改造，使系统具备海量数据实时接入、批量数据分析和实时数据查询能力；主站应用是满足终端用户需求的具体展现，底层对接数据中台提供的数据能力，对外根据国家电网《用电信息采集系统主站软件设计标准》要求完成业务模块和接口设计。

② 技术架构图

基于对用采系统总体架构的分析，在采用阿里云数据中台的基础上，技术架构图如下图所示。针对每个场景所采用的技术组件及实现的功能有：

- 在前置数据采集模块，使用 Redis 作为数据接入缓冲，入库集群服务从 Redis 中读取数据写入数据中台；
- 在数据存储的接入部分，使用数据总线 Datahub 作为接入组件，缓冲所有写入数据，Blink 以流的方式，按不同分类将数据写入不同的后端存储中；
- 以 PolarDB-X、MaxCompute 和列式数据库支持在线查询、离线分析和明细查询等不同场景需求。
- 主站应用系统基于PolarDB -X、ADB、列式数据库提供的数据接口，完成应用迁云改造。



2 应用场景

用采系统有两个显著特点：数据接入量大、支持的业务系统多样。针对这两个特点，需要重点解决两个问题：

- 如何设计数据存储层，不仅要满足海量数据实时入库需求，还要为数据计算和分析提供合适的存储底座；
- 如何进行数据计算层构建，满足应用对实时数据查询、离线数据计算、明细数据查询的不同要求。

因此在用采系统中，重点需要分析数据存储、数据计算与分析这两个模块场景的架构设计方案。

1 数据存储

1.1 场景简介

用采系统主要涉及两大类数据：智能设备采集数据和营销业务档案数据。

采集数据通过采集前置机实时采集，数据类型分为“曲线数据”和“日冻结数据”，“曲线数据”每 15 分钟上报一次，包含 1 个指标值（例如电流、电压值），“日冻结数据”每天上报一次，包含 96 个指标值（15 分钟采集一次，一天共 96 个值）；

营销业务档案数据主要包括设备信息、台区信息、采集点信息、客户信息等关系数据，数据源是营销域数据共享区的关系数据库。

两类数据的特征有比较明显的差异，采集数据由数千万智能电表、终端产生，数据量大，且持续写入，数据格式相对简单，主要记录的是过程明细数据；档案数据针对营销和用采业务设计，包含较为复杂的业务属性，是典型的结构化数据。因此，在用采数据入库的场景下，主要有以下场景特征：

- 采集数据持续写入，没有明显波峰和低谷；
- 以 5000W 设备，3 个采集指标估算，15 分钟需采集入库 1.5 亿条数据，由于数据可能在整点集中上报，会存在瞬间的数据写入高峰，数据不是均匀写入；
- 档案类数据涉及用户关键数据，入库时需严格保证数据一致性和完整性。

1.2 方案特点

传统用采系统的数据存储、计算、分析都为围绕 Oracle 数据库进行设计，由于当时设备数不多，数据写入压力不大，数据入库架构较为简单，前端采集程序将数据直接写入 Oracle 数据库中。但在高并发和海量数据场景下，数据入库方式和后端存储的选择需根据业务需要重新设计。根据用采业务需求，对数据入库主要考虑以下几点：

- 通过缓存应对采集数据存在瞬间的写入高峰；
- 将前端数据写入和后端存储解耦，通过数据总线作为缓冲，实现“削峰填谷”， 使数据以较为均匀的速率写入后端存储。
- 分别使用分布式关系数据库、分析型数据库、列式数据库和离线数仓存储不同类型数据，满足不同业务需求。

1.3 优势解读

- 在数据采集区增加一层Redis 缓存区，Redis 具备百万级别 QPS 能力，能轻松应对设备集中上报的数据写入高峰。
- 使用多种存储引擎满足不同业务场景，相比原来基于一个 Oracle 数据库的方案，实现了计算任务、事务型、分析型的业务场景分离。
- 增加了数据总线 Datahub 作为数据接入缓冲区，采集数据和档案数据统一在 Datahub 中汇聚，再通过 Blink 进行一源多发，分别写入到 PolarDB-X、MaxCompute 和列式数据库中，在源端保证数据的一致性和完整性。

2 数据计算分析与应用设计

2.1 场景简介

底层数据架构的设计目标是为了更好的服务于上层应用，根据国家电网《用电信息采集系统主站软件设计标准》要求，应用层包括档案管理、终端管理、费控管理、线损分析、负荷分析等数十种业务，因此对底层数据的使用也分为不同的场景：

- 电量计算、线损分析、负荷分析等业务，需要根据采集的用电信息，以及用户的档案数据等，按照一定规则进行计算，计算结果用于展示和分析，计算频率每天一次；
- 费控管理、档案管理、终端管理等业务，需实时查看用户档案、用户台账等信息，并对用户状态、设备状态等进行在线修改，是典型事务型应用；
- 报表管理、数据分析、综合查询类应用，主要进行复杂的数据分析查询，几乎没有数据修改，是典型的分析型应用；
- 采集数据明细查询，用于展示指定设备某时间段的历史指标数据，或者某时间点批量设备的指标值。

2.2 方案特点

根据针对不同业务场景的分析，为了更好支持业务开发，在数据计算分析层的架构设计主要关注以下几点：

- 采集的“曲线数据”和“日冻结数据”全量存储在列式数据库上，作为全量数据保存，同时也支持数据的实时查询；
- 每天一次的离线计算任务，在 MaxCompute 中运行，计算需要的档案数据、曲线数据、冻结数据等导入到 MaxCompute 中，由于 MaxCompute 不支持更新，对于数据补召或修改的场景，计算任务可以基于列式数据库的外表方式执行；
- 业务应用在线查询和修改的档案类数据，以及需要与档案类数据进行关联查询的“日冻结数据”、“曲线数据”，写入到 PolarDB-X 中，进行在线更新和查询，在线分析区仅存储热数据（半年或一年内需要查询的数据）；
- MaxCompute 中的计算结果，回流至 ADB 中进行查询加速，对于在线复杂查询，通过 PolarDB-X+DTS+ADB 进行 TP/AP 解耦；
- 使用 Redis 缓存档案数据，作为 Blink 的维表对采集数据进行核查。

2.3 优势解读

- 原计算任务基于 Oracle 存储过程，单点问题严重，进行分布式改造，具备良好扩展性，计算性能有极大提升；
- 用采业务的一个通用场景，是每日新增数千万记录的设备台账表，需要在线实时查询、更新，并与千万级的设备信息表关联，使用 PolarDB-X 进行分库分表，能充分发挥分布式架构的优势，相比原有单库架构，性能差距明显；
- TP 和 AP 业务解耦，充分利用 ADB 在复杂查询上的优势，有效提升复杂报表、数据分析类业务查询响应时间；
- 使用 Blink 流计算关联 Redis 维表，实现了对采集数据的实时校核，提供了原有架构不具备的实时计算能力。

成功案例

1 案例背景

某电力公司的系统每天对日冻结电能示值、电压曲线、电流曲线、功率曲线等数据进行自动采集，与远程费控系统、营销系统接口实现高低压用户自动停复电功能，计量装置在线监测实现计量装置异常诊断及异常处理的闭环管理，台区线损管理模块为线损治理提供数据支撑，对 5 个部门 15 个应用系统提供数据接口服务。

随着用电信息采集系统建设的不断深入及采集覆盖率的逐步提高，系统功能模块，对外接口越来越多，对用采系统的规划和定位有了新的认识：

- 用户规模不断扩大，对系统稳定性、可靠性、实时性提出更高的要求。
- 加强数据再利用，满足各部门业务分析需求。
- 满足智能化运维需求，降低运维成本。

因此，该电力公司基于云计算和大数据技术进行用采系统架构设计，在阿里云平台完成新一代用采系统建设，目前已成功上线运行。

2 系统规模和业务架构

智能电表：4100 余万只；

采集终端：70 余万台；

井井通用户：27 余万户；

水气热用户：20 余万户；

光伏用户：6 万余户。

③ 业务效果

核心指标提升如下表所示：

	原主站用采系统	新一代用采系统
数据采集能力	针对日冻结电能示值数据，采集入库需要1.5小时	40分钟内可完成日冻结电能数据入库，采集能力提升1倍
数据查询能力	页面渲染：10s 常规统计查询：20s 模糊查询：30s	页面渲染：1s 常规统计查询：2s 模糊查询：3s 实时数据查询能力近10倍提升
数据计算能力	4000W户电量计算通过存储过程实现需要4小时	采用分布式计算耗时仅30分钟，计算能力约8倍提升
实时数据处理能力	数据入库后进行全量数据处理，无实时处理能力	基于Blink和Redis可实现抄表数据的实时核查

常见问题解答

① 对于全量采集类数据，是存储在MaxCompute中，还是列数数据库中，或者PolarDB-X中？在产品选型时应如何考虑？

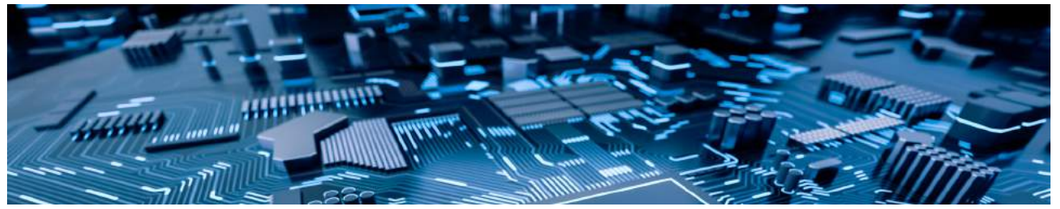
答：采集数据从数据模型上看，主要是时序类数据，结构相对简单；从场景上看，会存在较为频繁的采集数据召测、采集值修改等操作。由于MaxCompute无法实时更新数据，只能定时merge，因此使用MaxCompute存储全量会存在计算任务无法使用最新数据问题。列式数据库和PolarDB-X都可以保证数据修改实时可见，如果仅需要单表明细查询，使用列式数据库性价比更高。

② 如何实现传统数据库的数据（实时数据＋档案）实时上云？

答：数据上云主要有历史全量数据和实时增量数据，全量数据可以通过Dataworks的DI工具进行批量导入；增量数据可分为多种情况，对于采集类数据，由采集程序将增量消息写入Datahub，再由Blink写入到目标端；对于源端是Oracle的档案类数据，可以通过OGG插件将数据写入Datahub，再由Blink分发；如果增量数据过多，OGG无法消费，并且需要将数据实时同步到PolarDB -X/ADB等在线业务库中时，使用DTS进行增量同步；对于分钟级实时性要求，并且源端表中记录更新时间的场景，可用DI进行抽取。

③ 上云以后，业务应该如何使用MaxCompute、PolarDB-X、ADB、列式数据库等不同存储？

答：MaxCompute主要进行离线计算，不具备实时查询能力，因此MaxCompute不直接对应用提供查询接口。对业务提供在线查询服务能力的主要是PolarDB-X、ADB、列式数据库，PolarDB-X适用事务型前台应用，主要是较为频繁的增删改查操作，能支持一定程度的统计类查询；ADB具备更强大的复杂查询能力，统计类报表、数据分析类业务可基于ADB查询；列式数据库提供采集明细数据单表查询，不支持复杂查询。

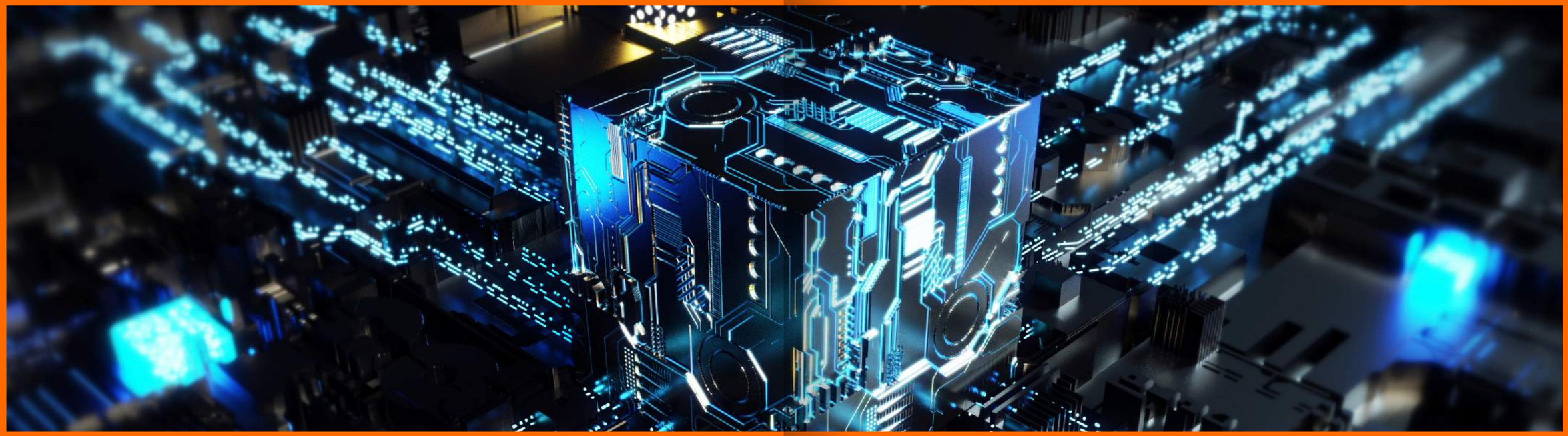


4 上云后用采业务系统使用的关系库主要是 PolarDB-X，如何进行数据拆分比较合适？从业务使用场景看，大多数 SQL 中都带有“区域编码”条件，是否根据区域分区比较合适？

答：PolarDB-X 进行拆分的首要原则是将数据尽量打散，如果仅根据地市级的区域编码拆分，会存在比较明显的数据倾斜（例如省会城市数据量可能会占到 30% 以上），因此仅根据地市分区并不十分合适。从业务场景上，核心 SQL 是十亿级别的设备台账类表，与千万级别的设备信息表进行关联，关联字段多为 ID 类唯一标识，因此拆分条件可考虑设备 ID。但根据设备 ID 拆分，会存在多维度跨库关联查询，会增加业务开发和优化的难度，为减少业务逻辑复杂性，可以进行细粒度的区域划分，例如根据“地市 + 区县 + 供电所”的粒度来划分区域，使数据分布能较为均匀，同时业务 SQL 尽量带有区域条件，尽量将 SQL 下推执行。

5 PolarDB-X 和 ADB 都能提供在线查询能力，两者有什么差别？应用如何区分？

答：简而言之，“增删改”频繁的应用，都基于 PolarDB-X 进行开发；以复杂查询为主，没有或者少量修改的应用，基于 ADB 进行开发；表数据量千万级，5 张表以上不同关联维度的查询，建议都在 ADB 上执行。



附录

术语表

术语	解释
智能电表	智能电表是智能电网（特别是智能配电网）数据采集的基本设备之一，承担着原始电能数据采集、计量和传输的任务，是实现信息集成、分析优化和信息展现的基础。
HPLC	是高速电力线载波，也称为宽带电力线载波，是在低压电力线上进行数据传输的宽带电力线载波技术。
台区	电力台区是指电力系统中一台变压器的供电范围，或一台变压器的供电区域。
日冻结数据	指存储的每天零点的设备指标数据。
数据召测	指主站对电表或者终端里面的数据进行获取。
SG168	2006年4月29日，国家电网公司提出了在全系统实施“SG186工程”的规划。