DTCC 2010-2022

第十三届中国数据库技术大会

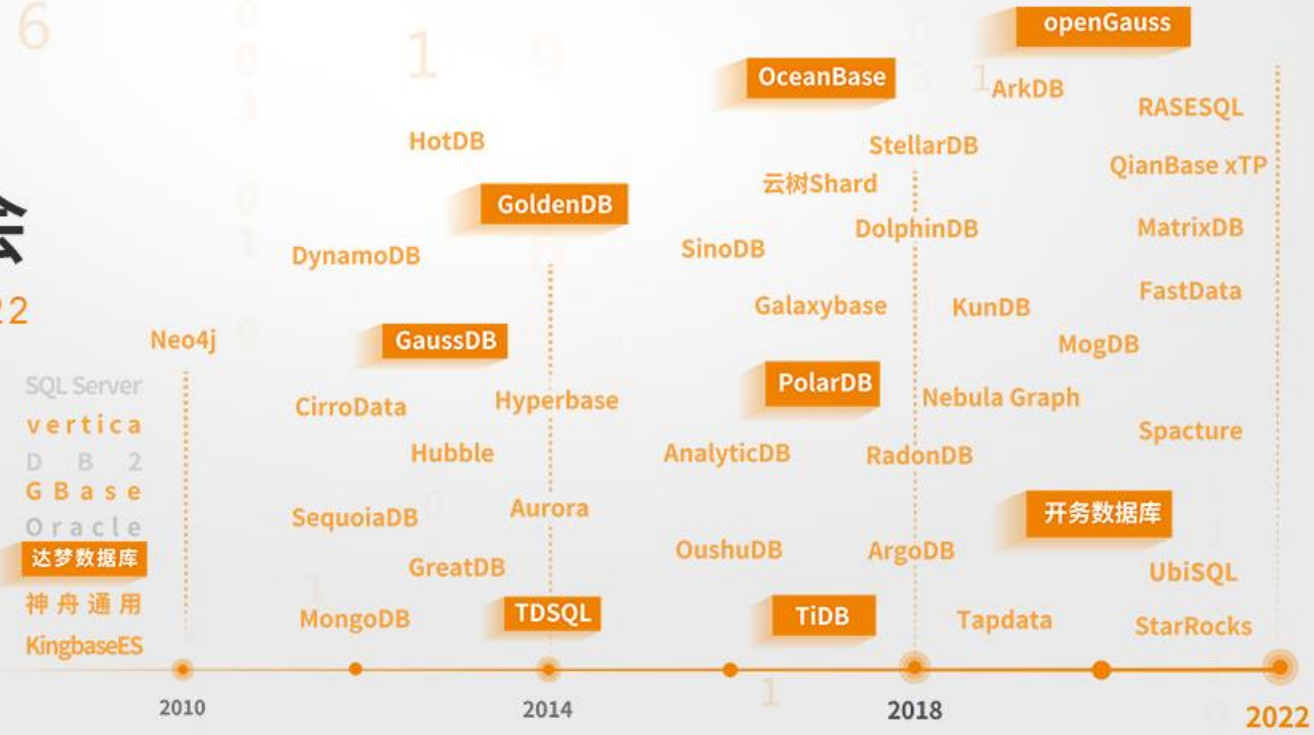
DATABASE TECHNOLOGY CONFERENCE CHINA 2022

数据智能 价值创新

IT168.com ChinaUnix.net ITPUB

线上直播 | 2022/12/14-16

数据来源：数据库产品上市商用时间

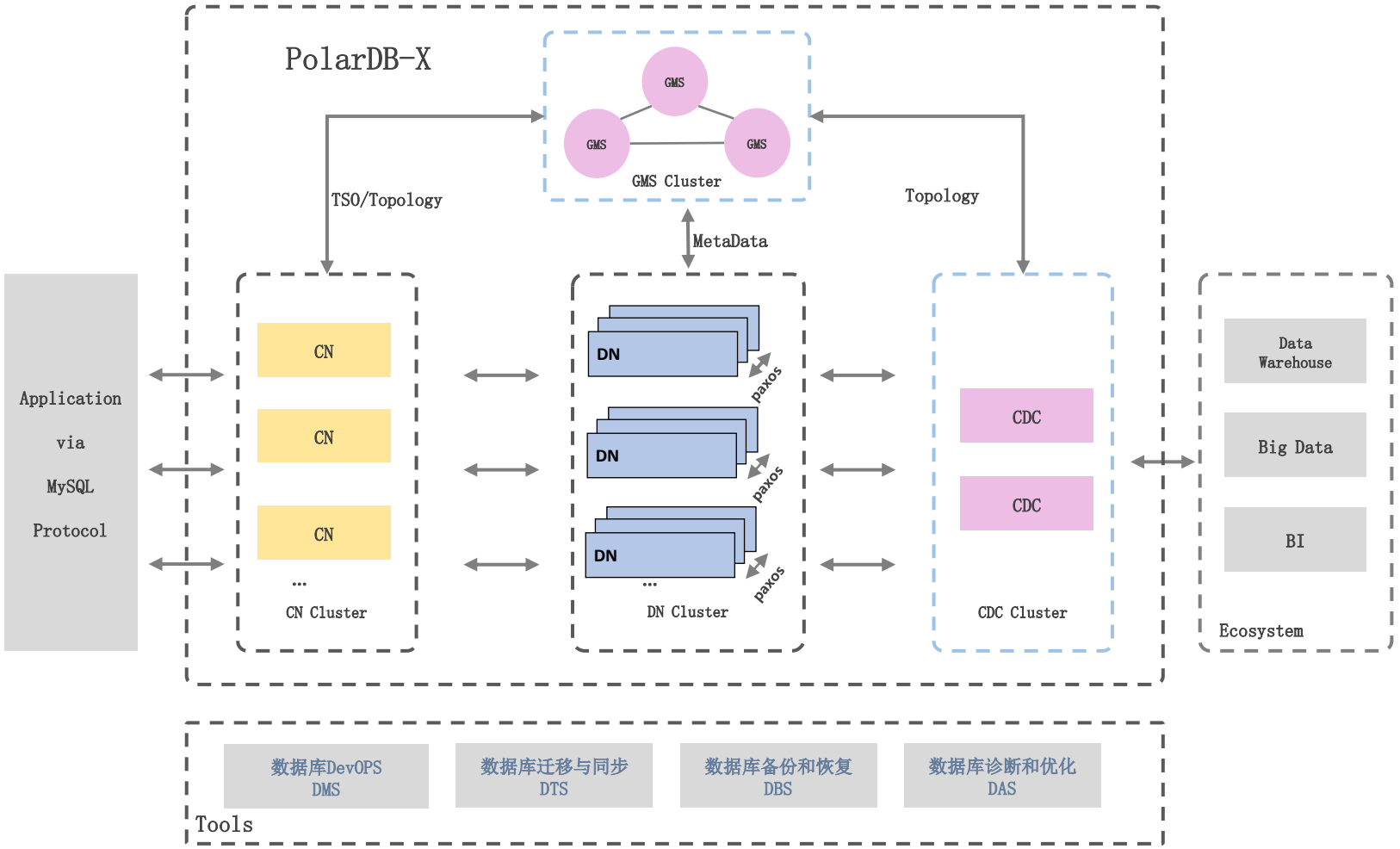


PolarDB-X:云原生分布式数据库的新可能性

黄贵
阿里云资深技术专家

PolarDB-X技术架构

PolarDB-X技术架构:云原生分布式数据库



元数据服务（Global Meta Service，GMS）

- 提供全局授时服务(TSO)
- 维护Table/Schema、Statistic等Meta信息
- 维护账号、权限等安全信息

存储节点 （Date Node，DN）

- 基于多数派Paxos共识协议的高可靠存储
- 处理分布式MVCC事务的可见性判断

计算节点（Compute Node，CN）

- 基于无状态的SQL引擎提供分布式路由和计算
- 处理分布式事务的2PC协调、全局索引维护等

日志节点（Change Data Capture，CDC）

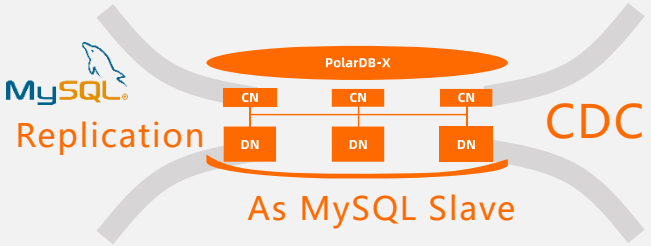
- 提供兼容MySQL生态的binlog协议和数据格式
- 提供兼容MySQL Replication主从复制的交互

PolarDB-X 产品特点

原生MySQL生态

兼容性

- MySQL语法、语义、功能、协议兼容



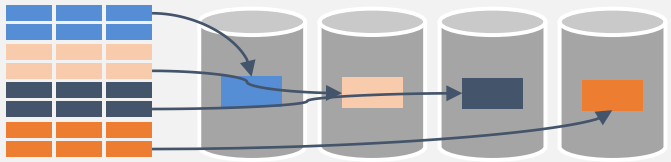
生态融合

- MySQL 生态兼容: 上下游工具无缝接入

一体化透明分布式

自动扩展

- 数据自动分区，线性扩展
- 自动模式无门槛使用分布式
- 手工分区模式按业务需求调优



分布式事务

- 高性能强一致分布式事务
- 全局二级索引满足多维度查询
- 一致性全局备份与快照

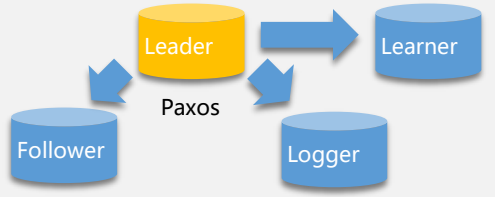
在线数据字典变更

- DML & DDL互不影响
- 修改数据分区规则适应访问特点
- 一致性全局备份与快照

企业级数据库

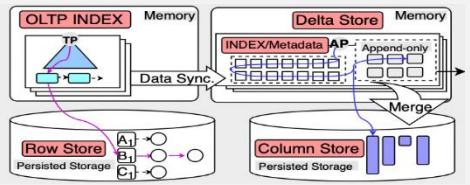
高可用

- 基于Paxos协议，数据强一致
- 少数派节点故障，不影响服务
- RPO=0　RTO<10s



HTAP

- 智能读写分离保障TP流量稳定性
- MPP并行查询只读副本查询隔离
- 全局TSO时间戳保证数据一致性



数据安全

- 三权分立机制权限控制
- 详细审计日志
- 数据脱敏与加密保障数据不泄露

分布式数据库发展方向

分布式技术并不是银弹

分布式数据库面临的问题

兼容性

应用和数据模型0改造

使用门槛

潜在分布式诉求,无需了解过多概念

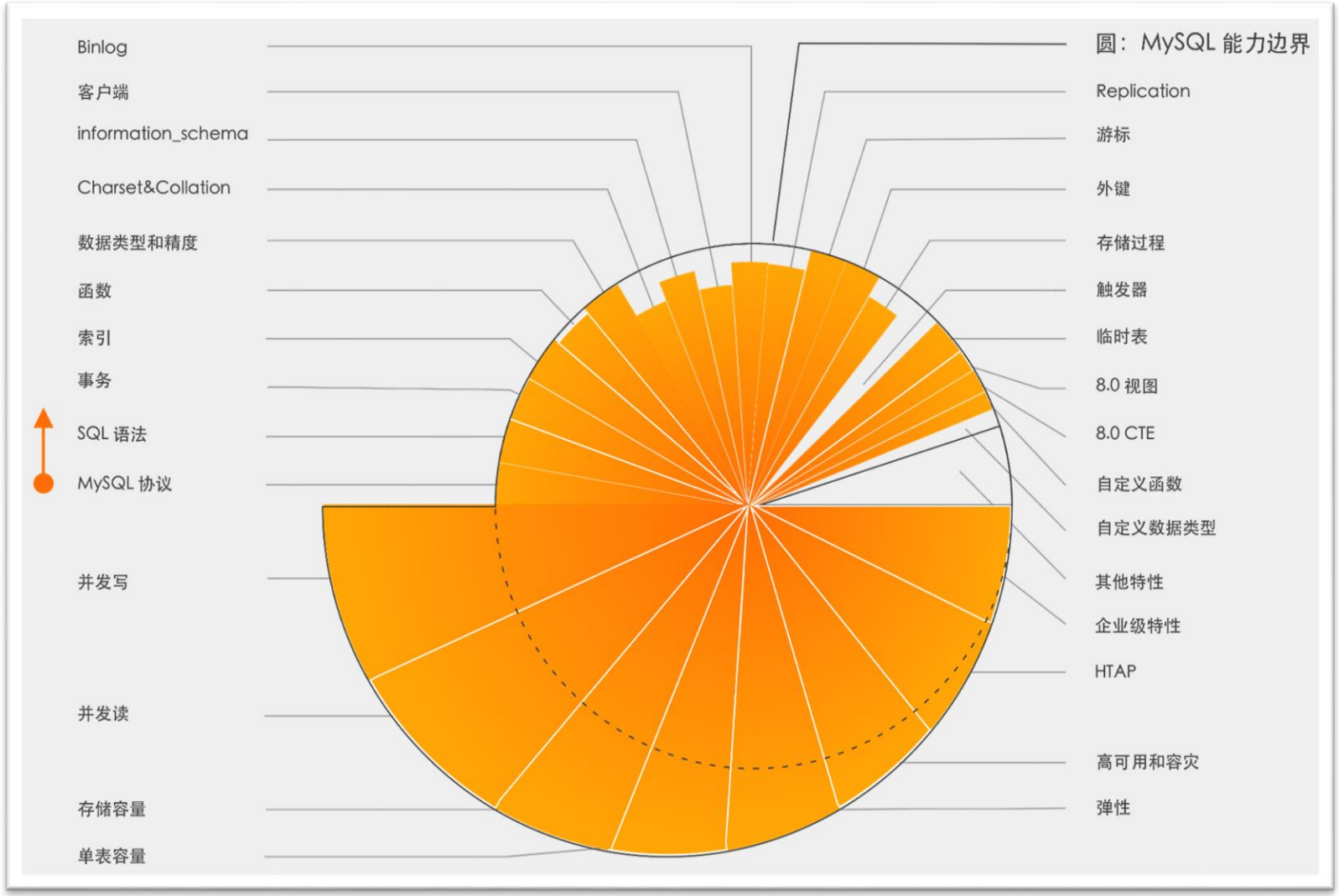
扩展能力

分布式架构线性扩展

运维复杂度

实时可观测&运维自动化

全面兼容MySQL功能特性



功能兼容

1. 兼容MySQL协议和SQL语法
2. 兼容数据类型和编码

性能兼容

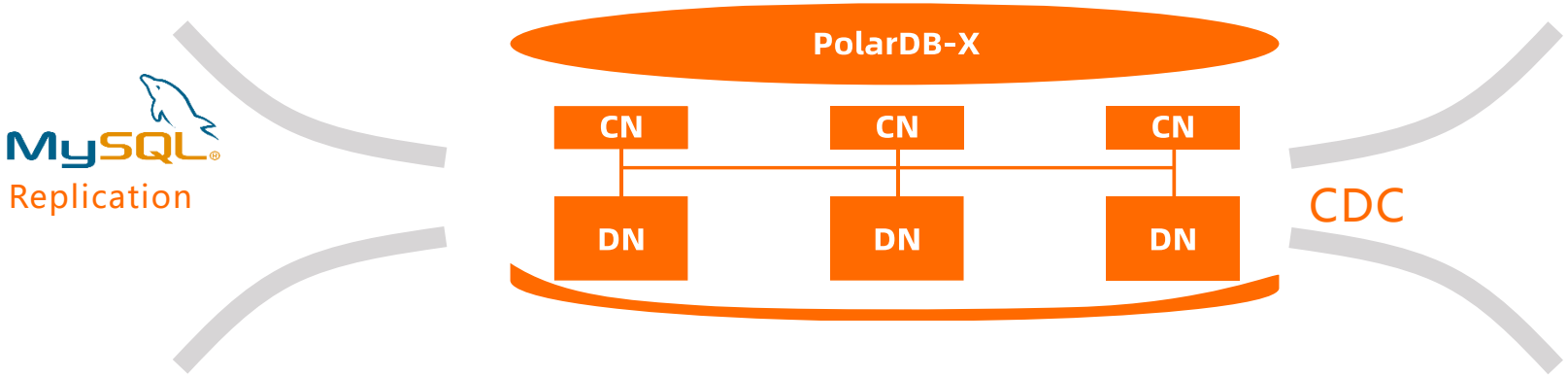
1. 兼容MySQL的事务行为
2. 兼容MySQL的并发读写

生态兼容

1. 兼容MySQL Binlog
2. 兼容存储过程、UDF函数等

全面兼容MySQL生态

无缝集成到现有数据生态，即插即用



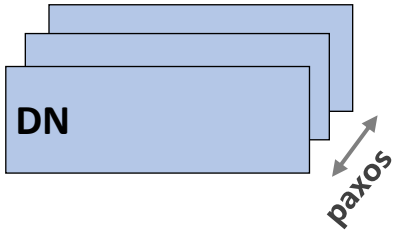
作为MyQL Slave
利用MySQL Replication组成高可用架构

CDC提供完全兼容单机MySQL Binlog
无缝接入现有生态工具同步到下游生态

集中式与分布式的一体化，降低使用门槛

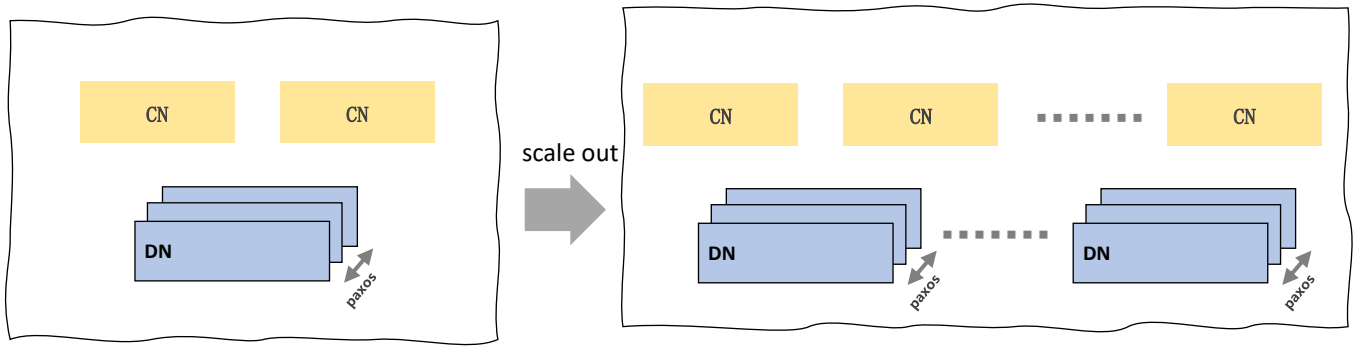
PolarDB-X 标准版

(100%兼容单机MySQL)



- 全方位优化MySQL高可用、存储引擎、查询链路等
- CPU<=32核，相比于分布式有性能优势
- 问题：B+Tree大表、以及scale up的限制

PolarDB-X 企业版



- 全方位优化分布式透明性，兼容查询、事务、binlog生态等
- 细粒度scale out，支持单表DDL动态变更为分区表
- 单机平滑演进，提供单机MySQL一键迁移

AUTO模式：数据分布对用户完全透明

自动挡（80%）

自动分区　　全局索引　　单机体验

```
# 典型的电商交易场景
  create table orders (
    id bigint,
    buyer_id varchar comment '买家',
    seller_id varchar comment '卖家',
    primary key(id),
    index sdx(seller_id),
    index bdx(buyer_id)
  )
```

相辅相成

手动挡（20%）

Locality

```
PARTITION BY LIST(省份) (
  PARTITION p1 VALUES IN (“山东”) LOCALITY=‘DN1’,
  PARTITION p2 VALUES IN (“浙江”) LOCALITY=‘DN2’,
  PARTITION p3 VALUES IN (“广东”) LOCALITY=‘DN3’)
```

Local Partition && TTL

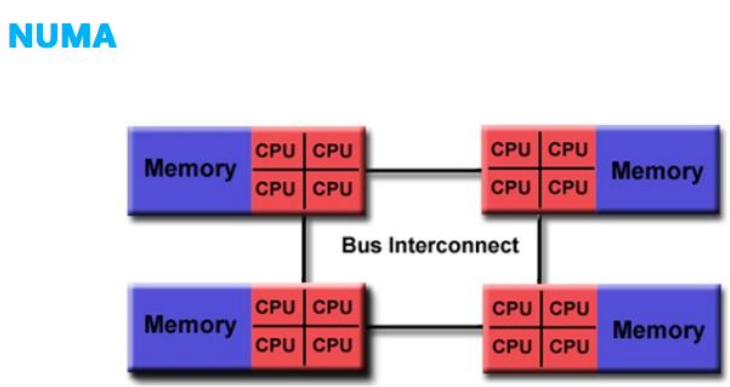
```
LOCAL PARTITION BY RANGE (gmt_modified)
INTERVAL 1 MONTH
EXPIRE AFTER 12;
```

Custom Requirements

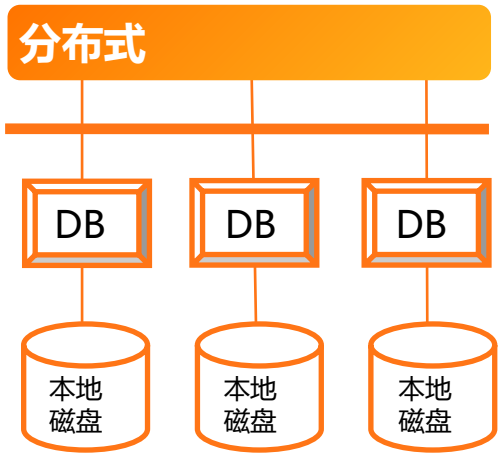
- 特定业务场景下的显示分区需求(Hash/Range/List)
- 热点分区散列，分区分裂、合并、移动、Rebalance
- … , …

分布式系统设计带来的代价

分布式数据库的阿克琉斯之踵：跨数据分区操作带来的巨大开销
将会成为横亘在分布式数据库美好愿景与现实应用中的巨大鸿沟



即便只有2个NUMA节点，跨NUMA NODE访问性能下降至少1倍



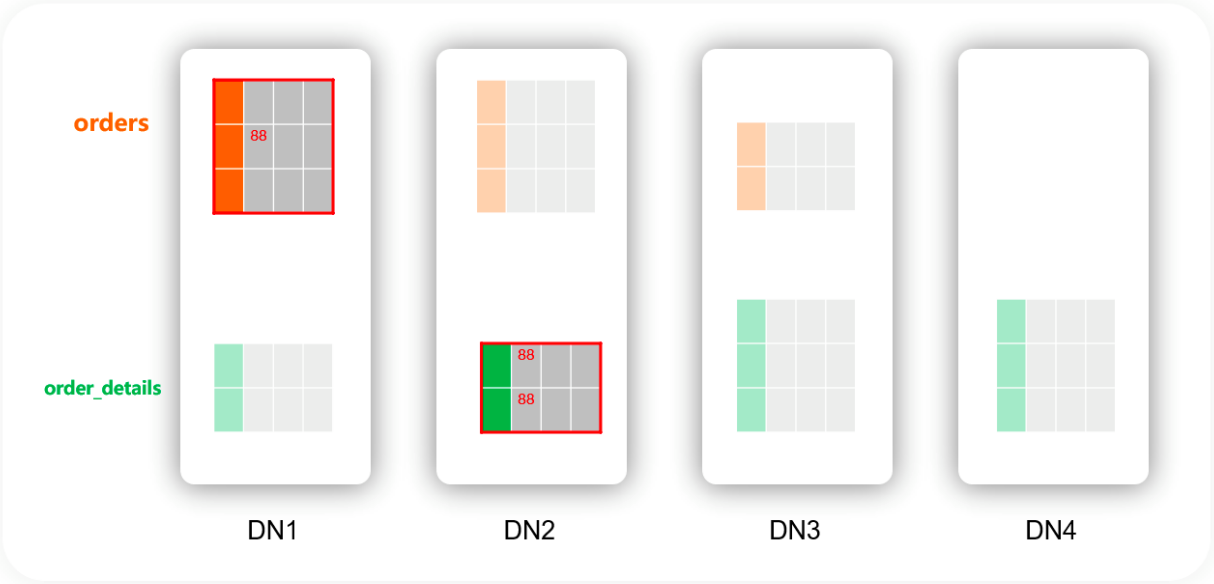
总线变网络，性能下降更加剧烈
+单个全局二级索引，写入性能下降30%
+8个全局二级索引，写入性能降到原来的10%(数量级)

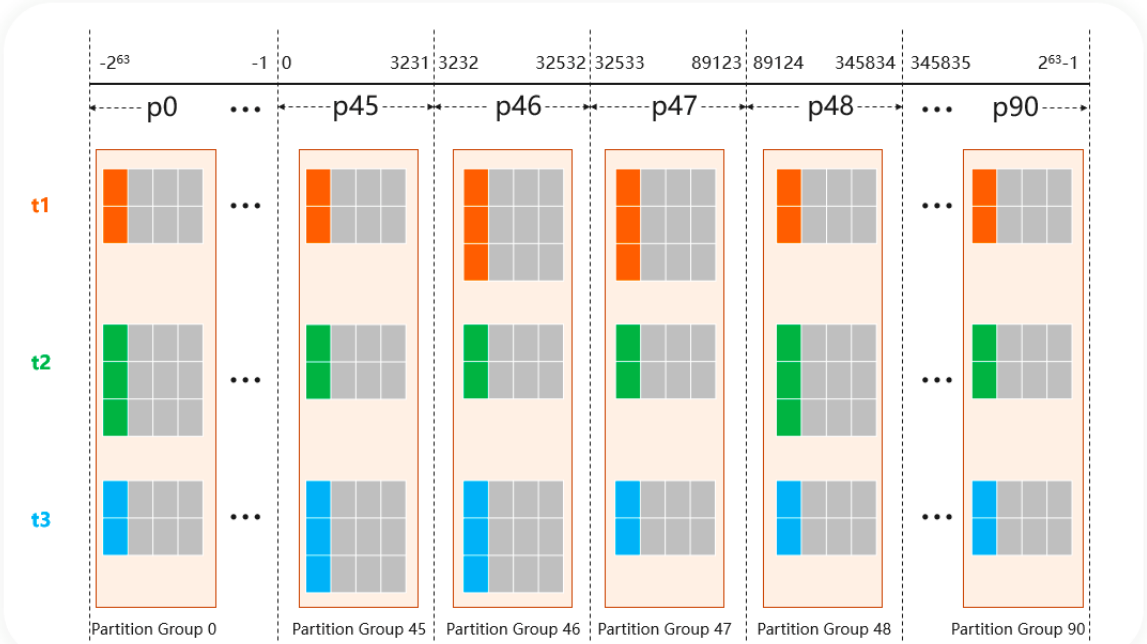
透明体验：性能不达预期
性能可用：精心设计数据分布规则，限制使用特性

根据数据亲和性绑定表组消除分布式事务

表组：数据在业务上存在关系的表，设定同样的数据分区规则，绑定为一个表组，作为调度基本单位

- 分布式事务限制在单个节点，优化为一阶段事务
- 本地索引(Local Index)绑定主表，索引维护无需分布式事务开销
- 本地化Join下推，减少data shuffle



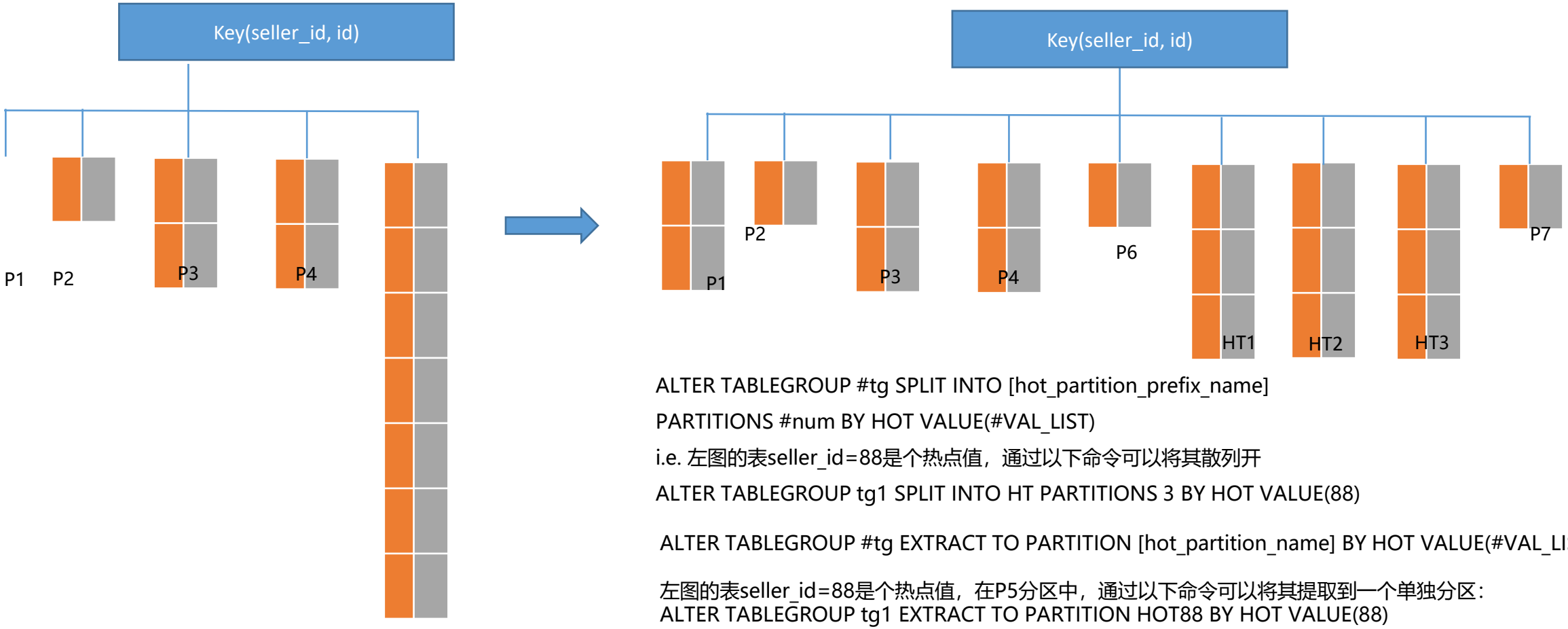


一个Table Group中的数据分布

- 如何接近分布式数据库的性能天花板(真正的线性扩展)
- 在完全透明和完全手工指定之间寻找平衡(根据数据访问特性自动识别数据分布绑定规则)

表组变更：均衡热点让系统变得更平滑

对于Key分区的表，PolarDB-X支持通过命令将热点散列或者提取到多个分区



ALTER TABLEGROUP #tg SPLIT INTO [hot_partition_prefix_name]
PARTITIONS #num BY HOT VALUE(#VAL_LIST)

i.e. 左图的表seller_id=88是个热点值，通过以下命令可以将其散列开

ALTER TABLEGROUP tg1 SPLIT INTO HT PARTITIONS 3 BY HOT VALUE(88)

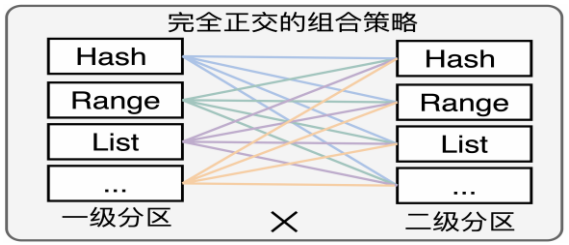
ALTER TABLEGROUP #tg EXTRACT TO PARTITION [hot_partition_name] BY HOT VALUE(#VAL_LIST)

左图的表seller_id=88是个热点值，在P5分区中，通过以下命令可以将其提取到一个单独分区：
ALTER TABLEGROUP tg1 EXTRACT TO PARTITION HOT88 BY HOT VALUE(88)

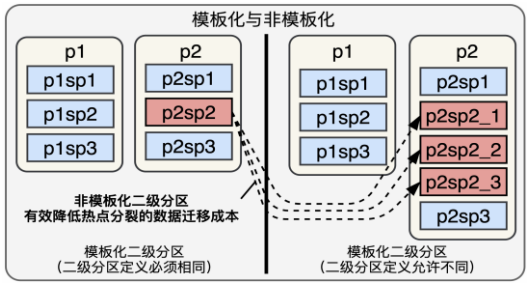
二级分区:更灵活的数据分布方式

核心优势

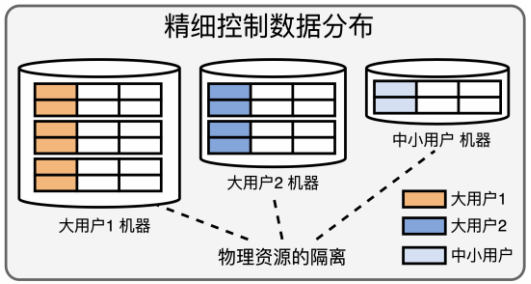
- 灵活多样的分区策略组合



- 支持模板化与非模板化



- 精细化控制数据的物理分布



适合场景

1. 多租户隔离 **2.** 大卖家热点散列 **3.** 历史库或流水型数据库归档 ...

某结算中台的分区方案

(场景特点：多租户模型、商家数据物理隔离、大商家资源定制)

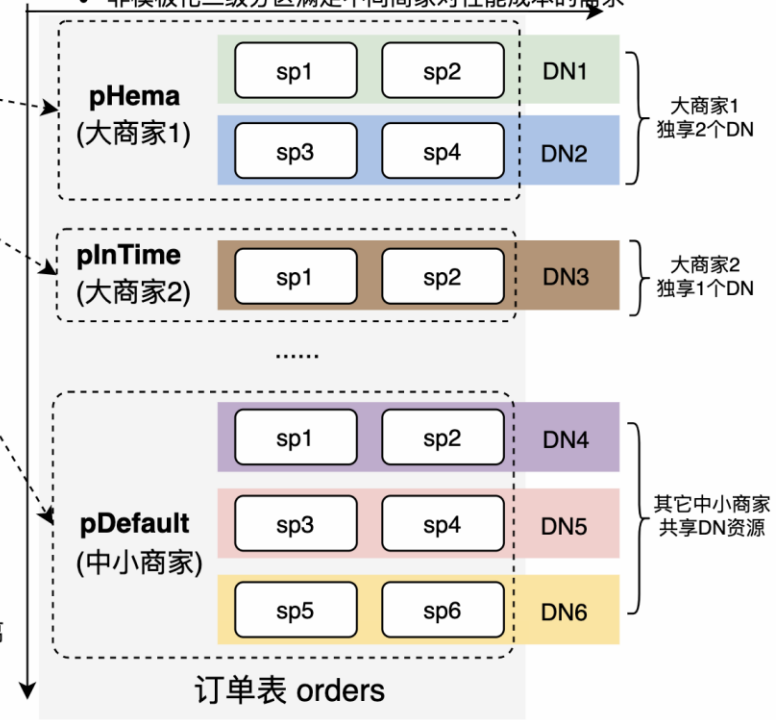
```
CREATE TABLE orders (...)
PARTITION BY LIST(seller_id)
SUBPARTITION BY HASH(order_id)
(
  PARTITION pHema VALUES IN ('Hema')
    LOCALITY='DN1,DN2'
    SUBPARTITIONS 4,
  PARTITION pInTime VALUES IN ('InTime')
    LOCALITY='DN3'
    SUBPARTITIONS 2,
  PARTITION pDefault VALUES IN (default)
    LOCALITY='DN4,DN5,DN6'
    SUBPARTITIONS 6
)
```

二级分区：

- 按订单ID字段HASH分区，预建分区，无写入热点，
- 非模板化二级分区满足不同商家对性能成本的需求

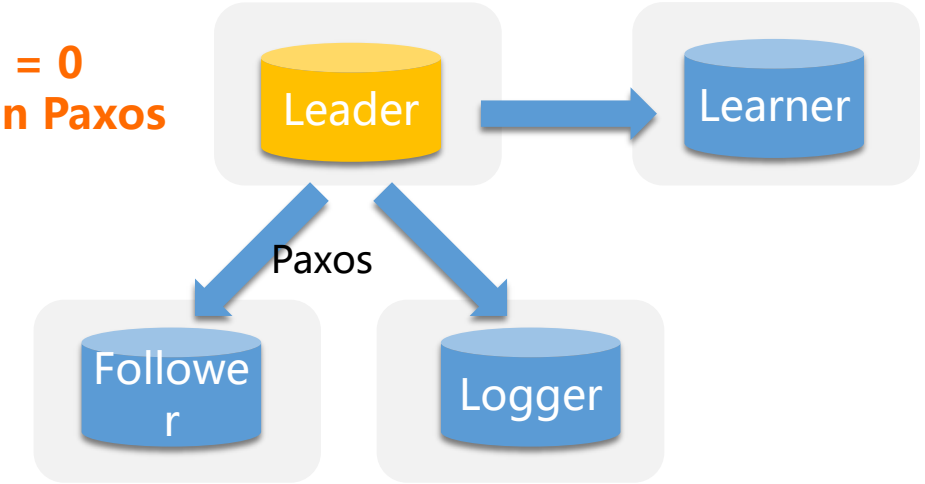
一级分区：

- 按商家ID字段LIST分区
- Localiy 对商家数据实施物理隔离

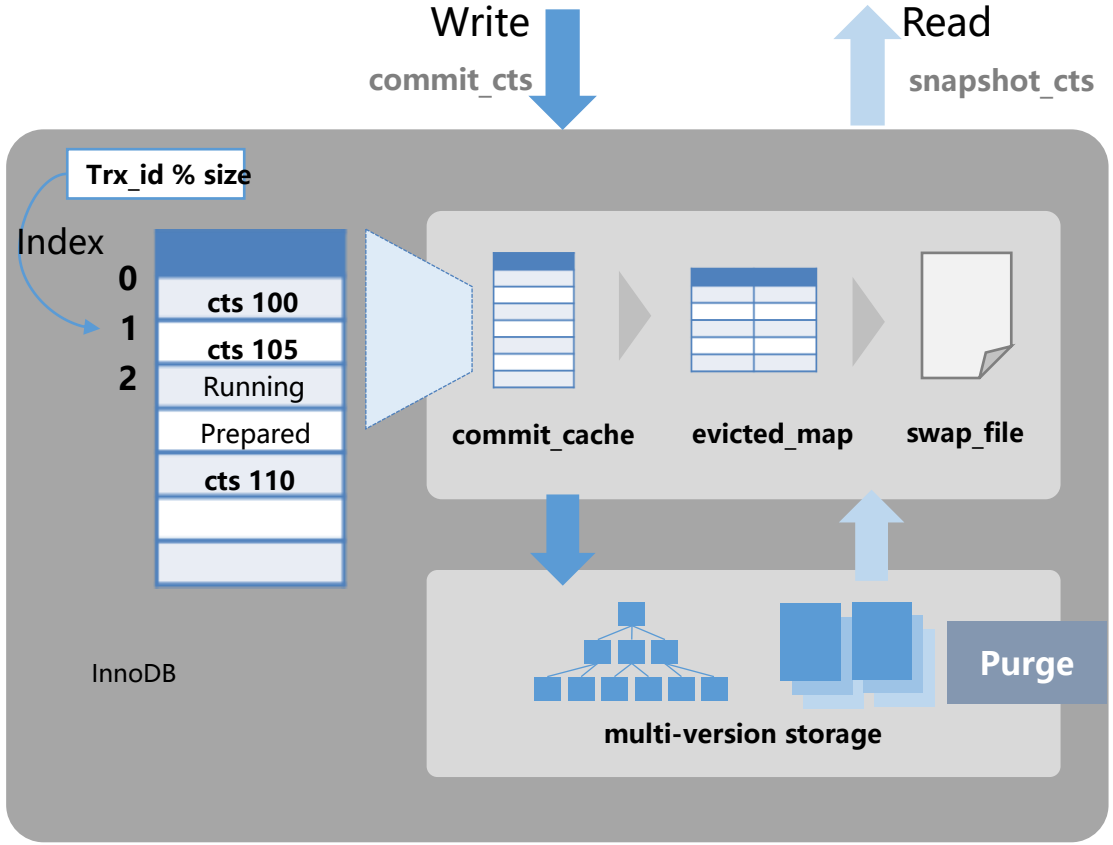


Paxos+2PC:任何时候都有数据强一致保证

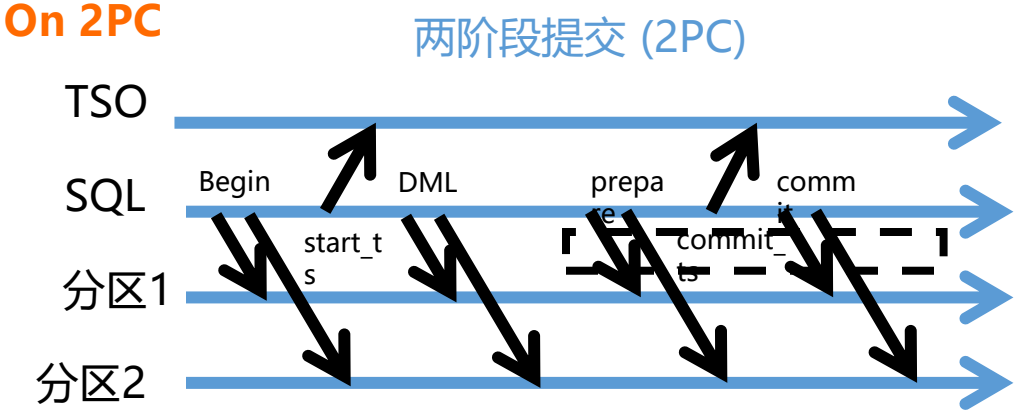
RPO = 0
Based On Paxos



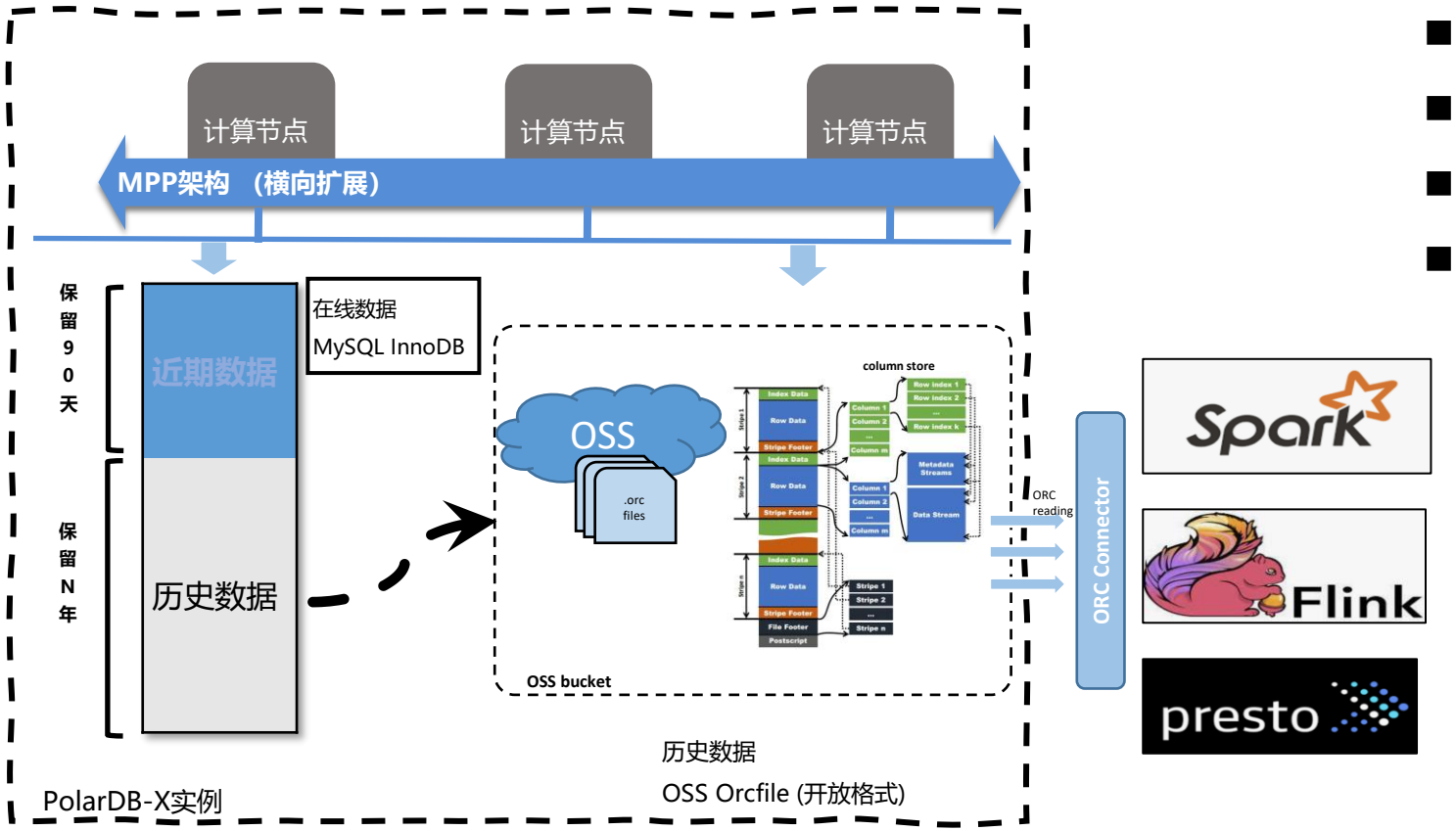
MVCC
Based On InnoDB And TSO



ACID
Based On 2PC



在线与历史归档数据一体化



- 指定数据过期规则，自动进行数据归档(历史数据)
- 统一SQL查询入口 （在线 + 历史）
- 冷数据压缩+低成本存储，降低在线库的成本
- 历史数据开放格式，对接开源大数据生态

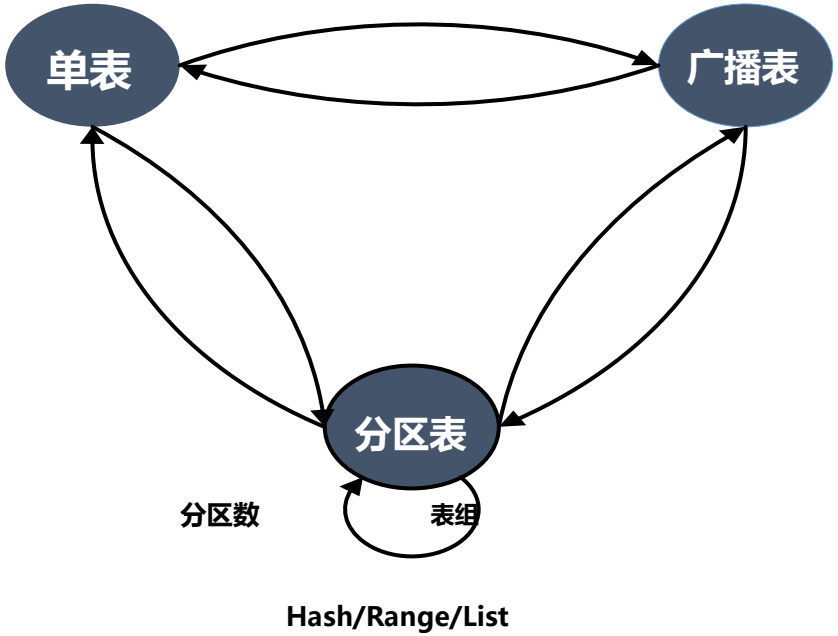
使用例子

```
# 创建TTL
CREATE TABLE t_order (
   ...
)
PARTITION BY HASH(id) PARTITIONS 16
LOCAL PARTITION BY RANGE (gmt_modified)
INTERVAL 1 MONTH
EXPIRE AFTER 12;

# 创建OSS归档
CREATE TABLE t_order_oss LIKE t_order ENGINE = 'OSS' ARCHIVE_MODE = 'TTL'

# 访问在线和历史oss数据
SELECT XX from t_order_oss a union t_order where buyer_id = 'xx' and ...
```

存储成本相对在线数据下降**20**倍

随时可执行的在线数据字典变更

One Sql，One Enter

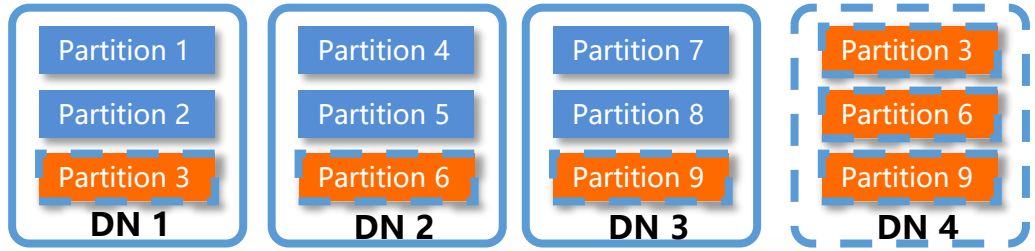


ALTER TABLE t1 PARTITION BY HASH(id) PARTITIONS 4 ;

ALTER TABLE t1 PARTITION BY HASH(name) PARTITIONS 8 ;

ALTER TABLE t1 Split into px PARTITIONS 4 by hot value (xxx) ;

Rebalance cluster ;



可实时观测的运维平台

产品能力

Monitor
- 性能趋势
- 实时监控
- 监控告警

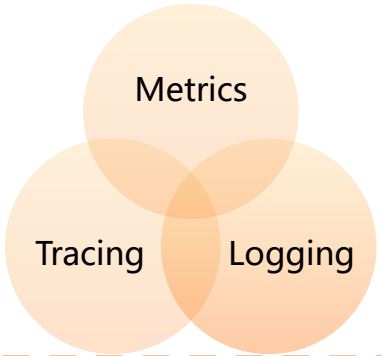
SQL
- 慢SQL
- SQL洞察
- Top SQL

Analysis
- 分区热力图
- 实例会话
- 空间分析
- 死锁分析

Profiling
- 全链路 SQL Tracing
- 智能压测
- 扁鹊

Tuning
- SQL 限流（自动）
- 索引推荐
- 扩缩容
- 分区变更

数据源



Metrics
- 资源指标
- 引擎指标
- 分区信息
- 空间指标
- Perf 指标

Logging
- SQL Log
- Slow Log
- Error Log
- Event Log

Tracing
- 逻辑SQL埋点
- 物理SQL埋点
- binlog 埋点

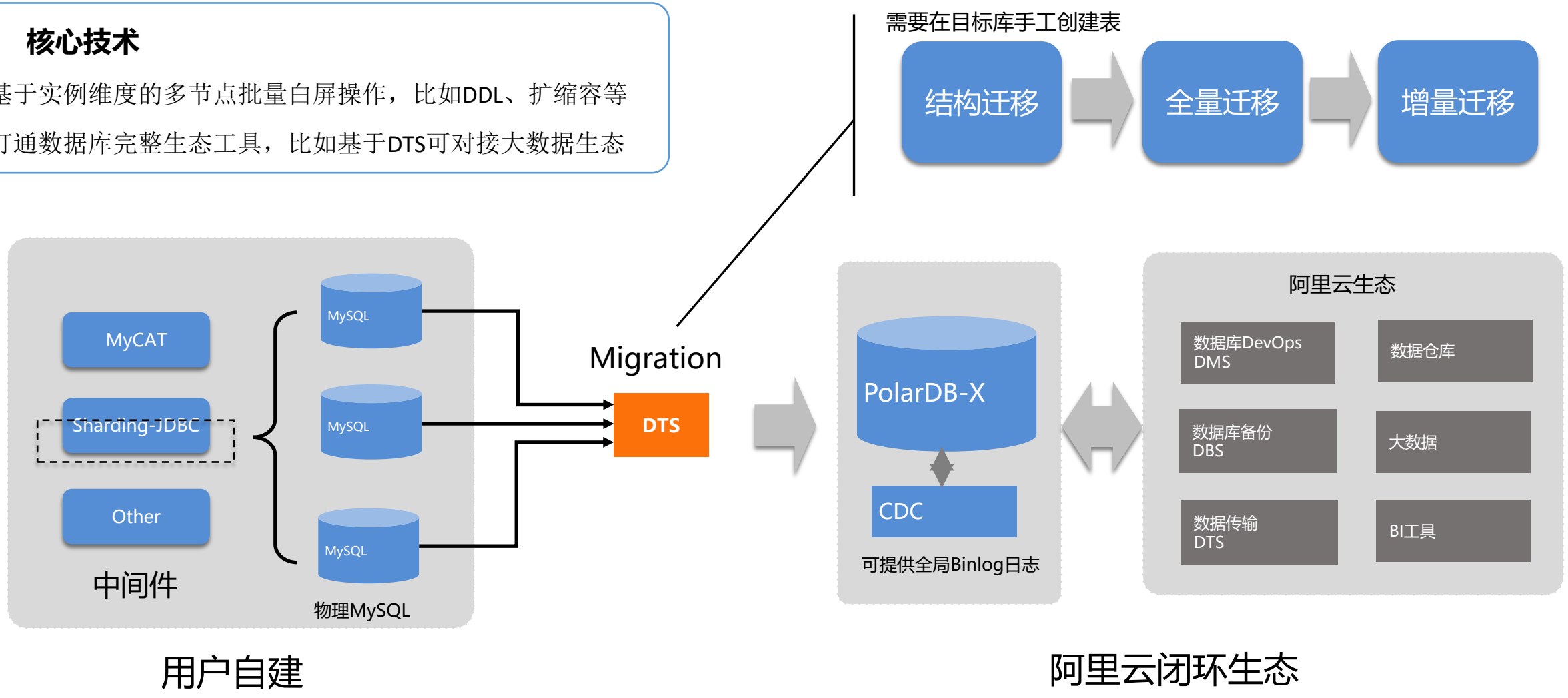
PolarDB-X主要业务场景

海量数据大集中、单机演进分布式、金融级高可用...

替换开源分库分表

核心技术

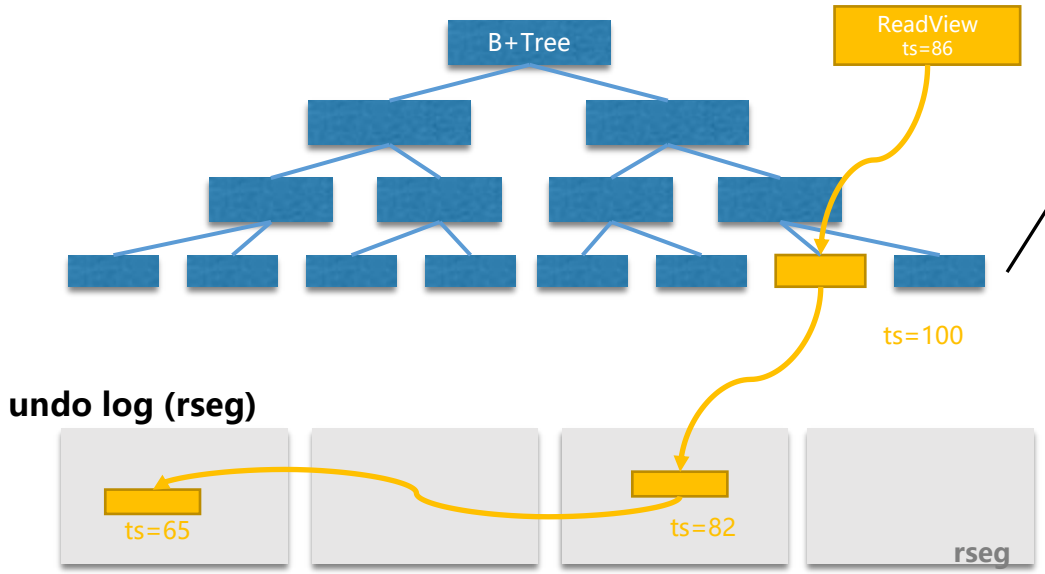
- 基于实例维度的多节点批量白屏操作，比如DDL、扩缩容等
- 打通数据库完整生态工具，比如基于DTS可对接大数据生态



单机平滑演进到分布式

核心技术

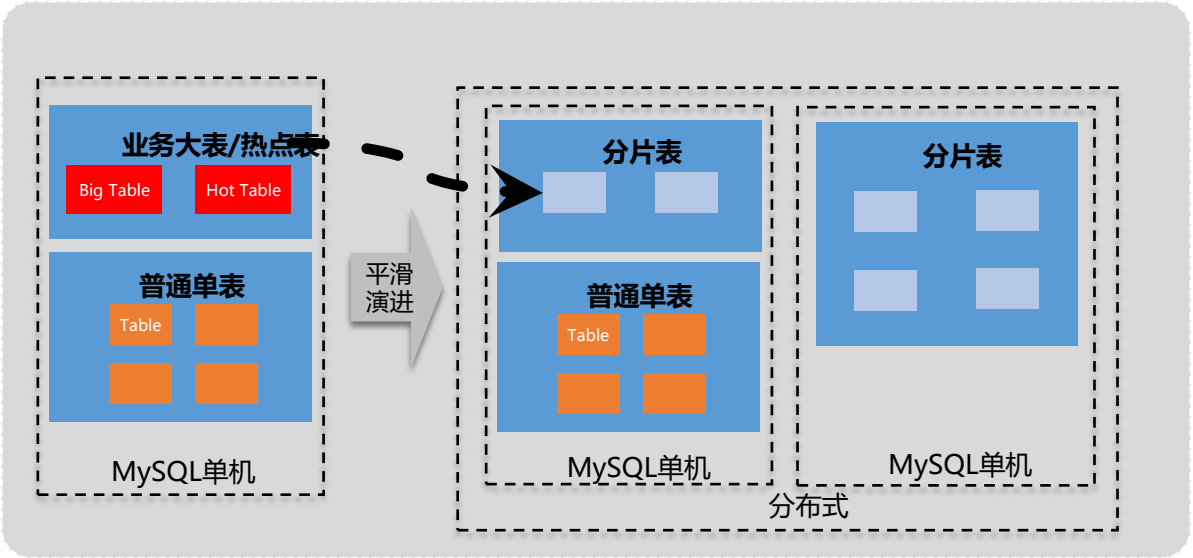
- 基于分布式的数据拆分和Online DDL，优化单机的大表和DDL
- 引入拆分变更DDL、全局binlog回流单机，确保分布式可上可下



MySQL B+树

MySQL 大表问题 (建议大表控制在500万~5000千万)

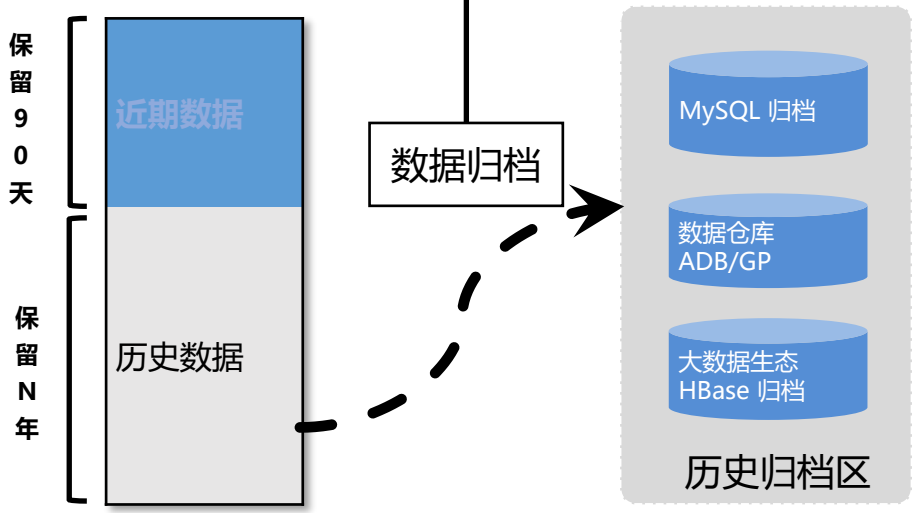
1. B+树深度 (行大小为0.5KB+自增主键，3层B+树大约支持4千万记录)
2. 业务并发度高，会因为B+树深度过深导致串行分裂的阻塞代价偏大
3. MySQL原生Online DDL执行代价过大，会导致主备复制延迟
4. 复杂查询性能不友好，MySQL原生为单线程+迭代执行导致效率慢
5. 历史数据归档不高效，大规模数据删除后需optimize table回收空间



海量数据大集中

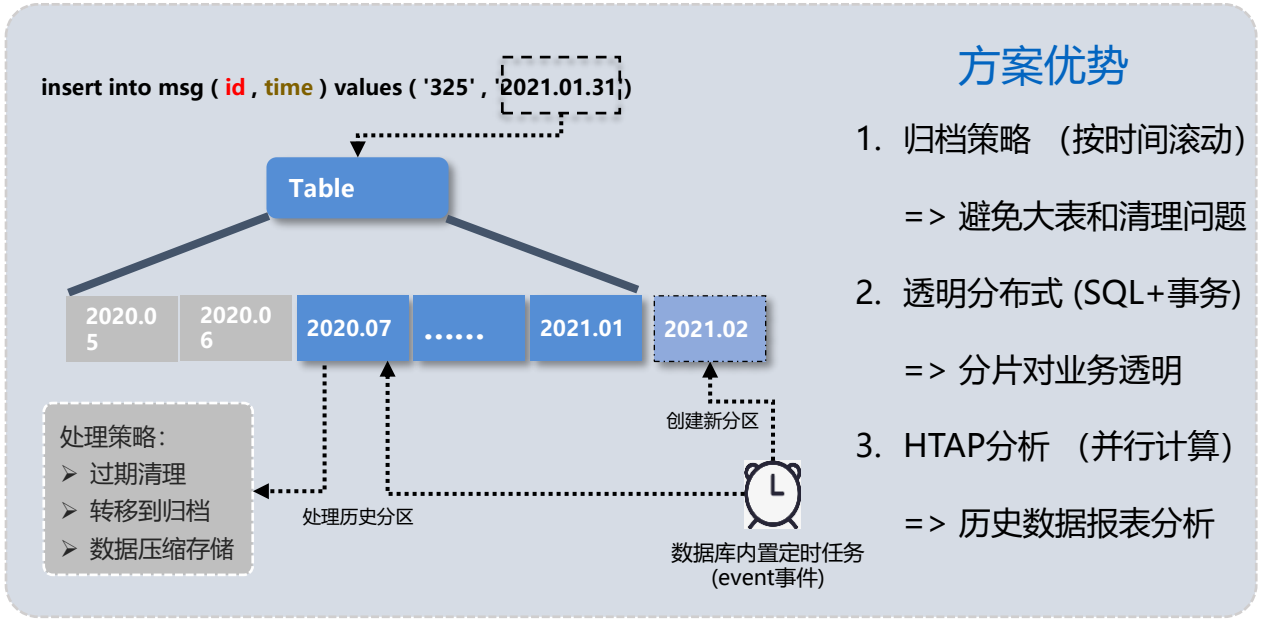
核心技术

- 支持高并发、存储无限扩展，以及提供SQL和事务ACID标准体验
- 引入历史归档的产品设计，提供低成本压缩、清理、转移等方案



数据归档挑战

1. 在线数据库，历史数据的高效清理 (delete操作复杂性和代价偏高)
2. 在线数据清理 和 数据写入归档库，需要有一致性和冗余处理，比如先入库
3. 数据清理和归档，业务需要维护定时任务，确保任务的正常运行
4. 在线和历史归档采用两套存储，业务需要做不同存储特性的适配工作
5. 历史数据库需要针对数据偶发更新、报表更新等做做相应产品选型
6. 在线和历史数据，业务如果要做联合查询，需要业务层面实现数据聚合



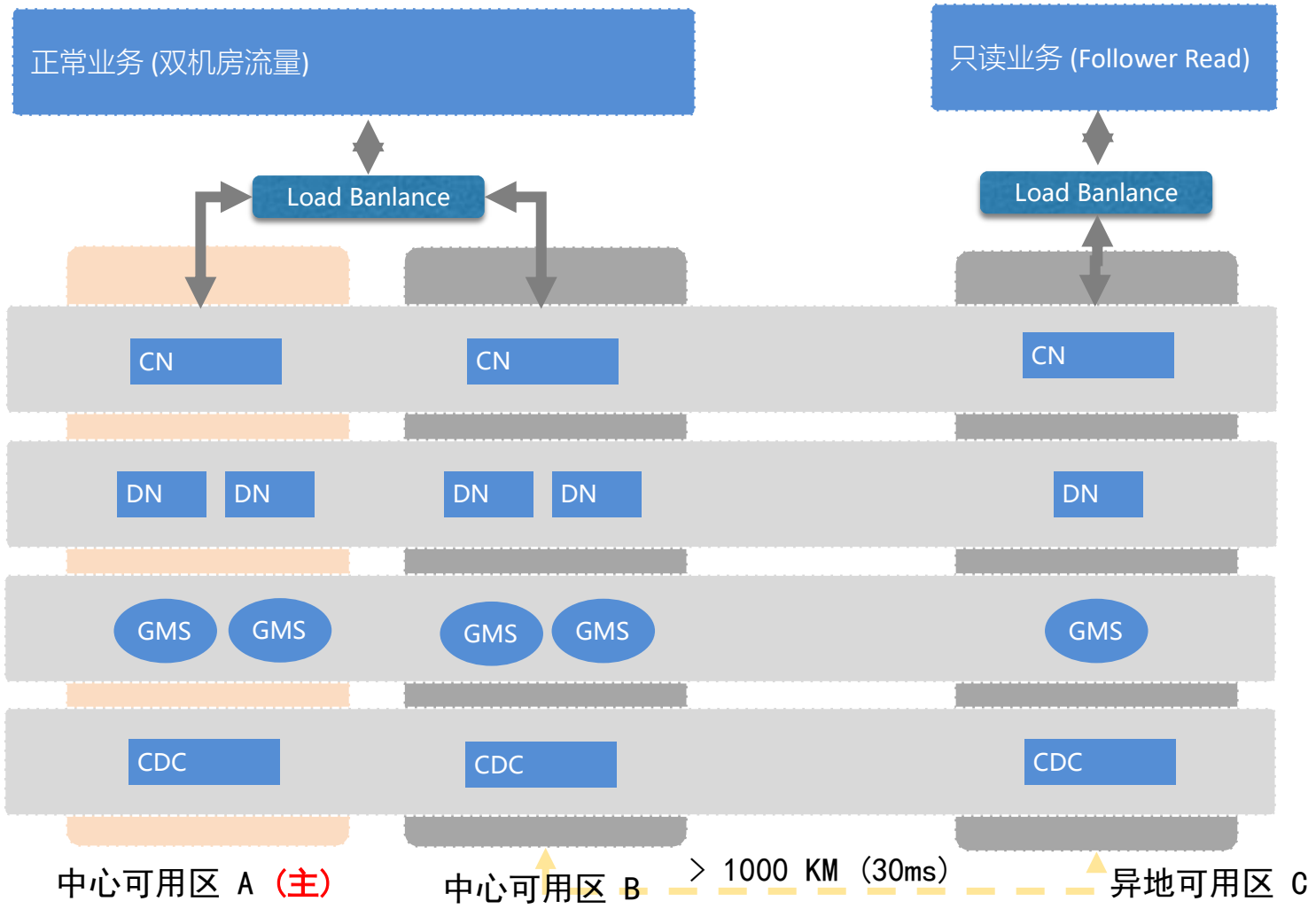
方案优势

1. 归档策略 （按时间滚动）
   => 避免大表和清理问题
2. 透明分布式 (SQL+事务)
   => 分片对业务透明
3. HTAP分析 （并行计算）
   => 历史数据报表分析

通常业务诉求：

1. TB级别的规模 （在线有事务和一定并发写入）
2. 历史数据复杂性 （数据做压缩、数据偶发更新、做报表分析）

金融级数据强一致，跨地域高可用



存储五副本（2+2+1）

- 存储副本leader切换本机房优先
- 机房级别容灾和恢复处理
  1. 异地机房 => 不影响
  2. 中心机房故障 => 5副本降级3副本 (手工)
  3. 中心机房恢复 => 3副本升级5副本 (手工)
  4. 中心地域故障 => 异地单副本启动 (脚本)

     => 此场景下，分布式事务一致性需要额外处理

计算多可用区

- 可用区流量对齐（locality）
  1. 中心主可用区，常态下避免CN/DN跨机房，减少RT
  2. 异地机房，CN需要访问本地域的DN，就近访问
- TSO时间戳分配策略
  1. group (CN节点可以合并请求一次性批量获取)
  2. pretch (GMS利用租约机制支持TSO的异步持久化)

PolarDB-X 发展方向

更加往一体化的方向

HTAP:事务处理与分析一体化

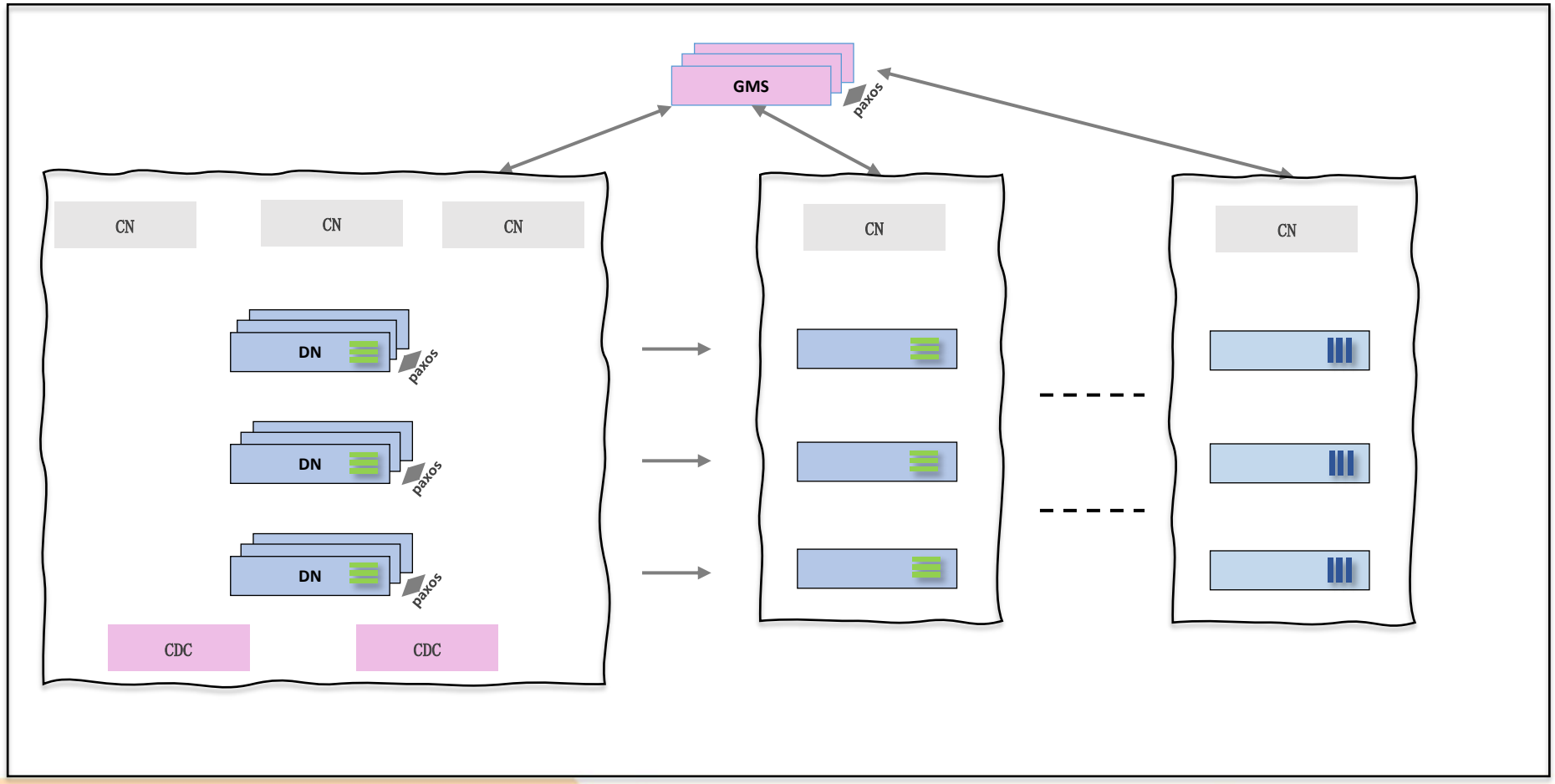
写 读

读写路由 RW + RO
Endpoint (负载均衡)

- 兼容MYSQL生态
- 云原生列索引服务支持亚分钟的更新

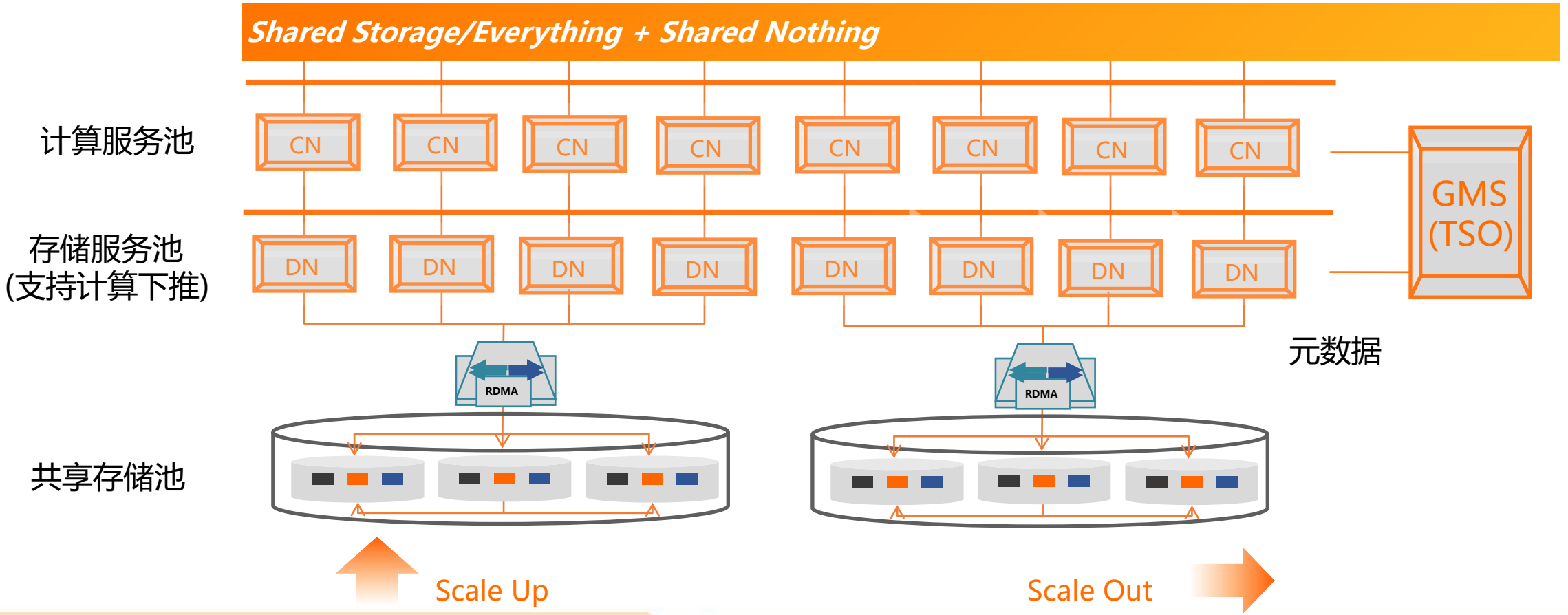
读

只读路由 RO
Endpoint (负载均衡)



云原生与分布式融合一体化架构

云原生资源解耦池化和分布式水平扩展的能力，可提供资源的独立弹性，是云数据库主要发展方向




openGauss
OceanBase
ArkDB
RASESQL
HotDB
StellarDB
QianBase xTP
云树Shard
GoldenDB
DolphinDB
MatrixDB
DynamoDB
SinoDB
FastData
Galaxybase
KunDB
SQL Server
vertica
D B 2
GBase
Oracle
达梦数据库
Spacture
开务数据库
UbiSQL
神舟通用
TDSQL
TiDB
Tapdata
StarRocks
KingbaseES
2010
2014
2018

THANKS