云原生数据库的能与不能

窦贤明，腾讯云数据库专家工程师

云原生数据库的能与不能

窦贤明，腾讯云数据库专家工程师

Agenda

- 为什么从事云数据库？
- 云原生数据库，在做些什么
- 你不一定非要分布式
- Not For ALL, For the MOST

为什么从事云数据库

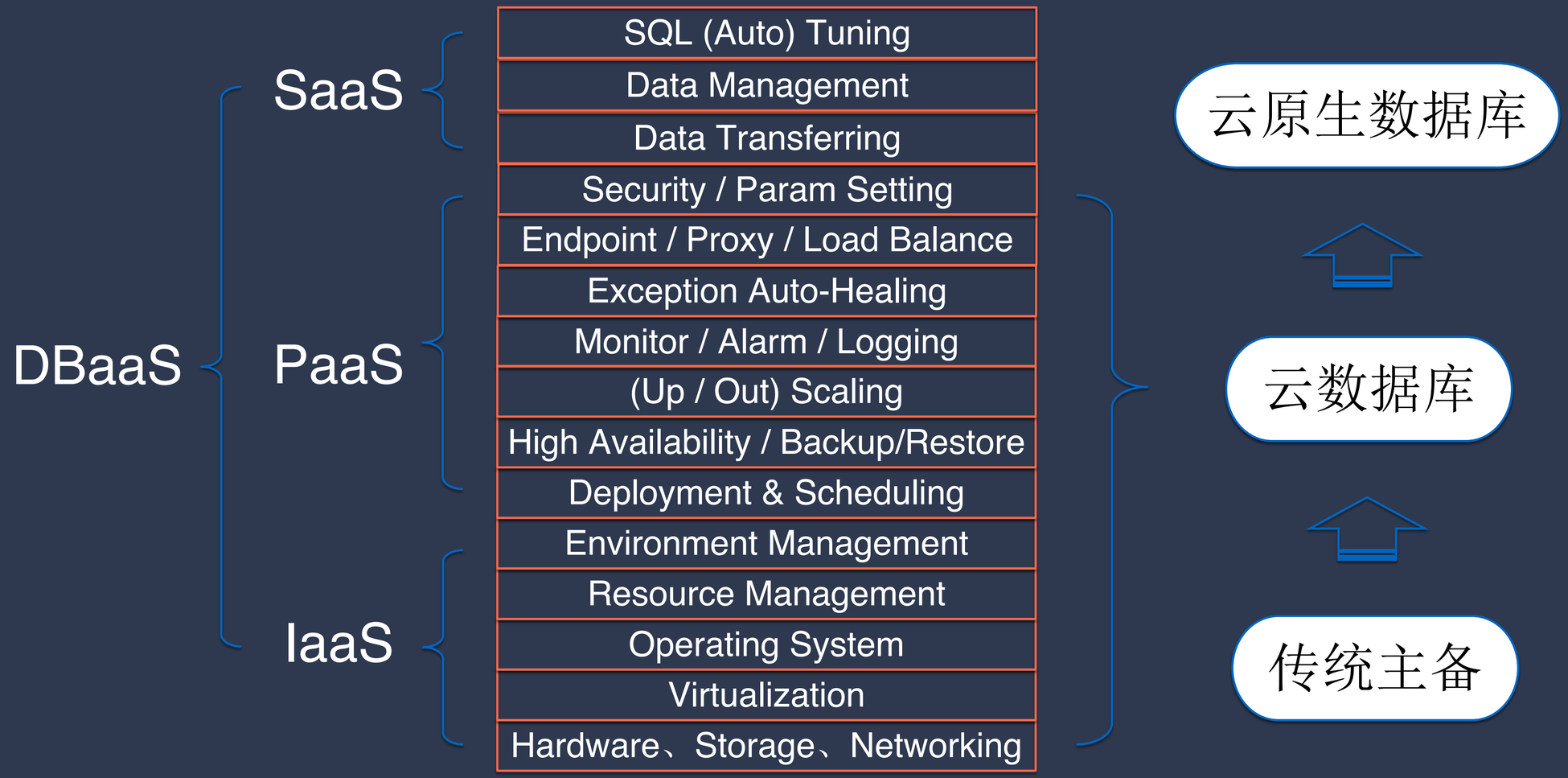
- 从业十余年、*从零到一* 研发多款云数据库产品，**RDS、云原生数据库、HTAP**
  - 关系型数据库，是软件工业的基石
  - 云计算，深度改造各产品链的IT系统，提升社会整体效率、降低整体成本
  - 国产化替代浪潮，不再被卡脖子
- TDSQL-C MySQL / TDSQL-C PostgreSQL 产品研发负责人
- TencentDB For PostgreSQL 产品研发负责人

为什么从事云数据库

- 云计算，对数据库行业带来前所未有的冲击，五到十年高速发展期
- 云原生数据库、分布式数据库，走向深度的应用
- AI、自动驾驶、智能制造等，底层都是 数据库技术

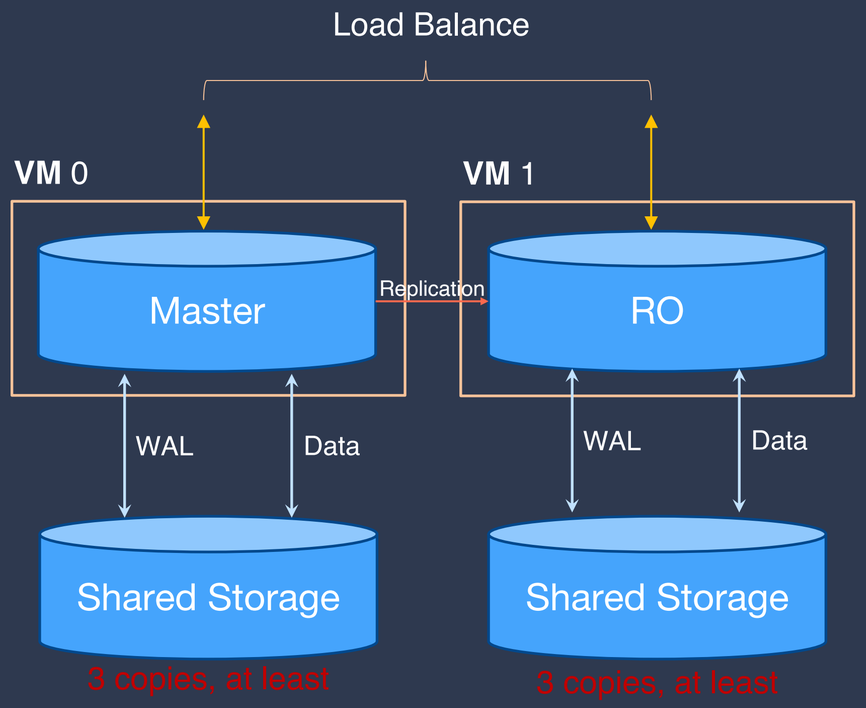
云原生数据库，在做些什么

云原生数据库，在做些什么



云原生数据库，在做些什么

传统数据库 → 云数据库



云数据库相比传统数据库：

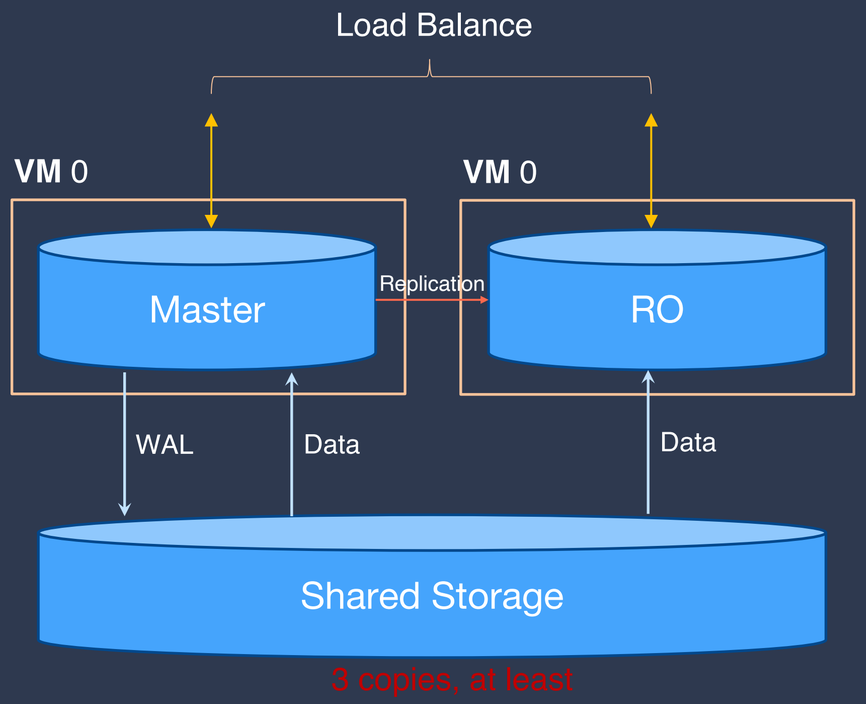
- 成本更低
- 扩容能力更好
- 体验更好

云数据库未解决的问题：

- 存储成本，（3 + N * 3）份数据存储，N 为 RO 节点数量；WAL同理
- （高吞吐数据处理）网络成为瓶颈，共享存储侧有大量网络浪费
- HA切换问题
- RO建设成本高

云原生数据库，在做些什么

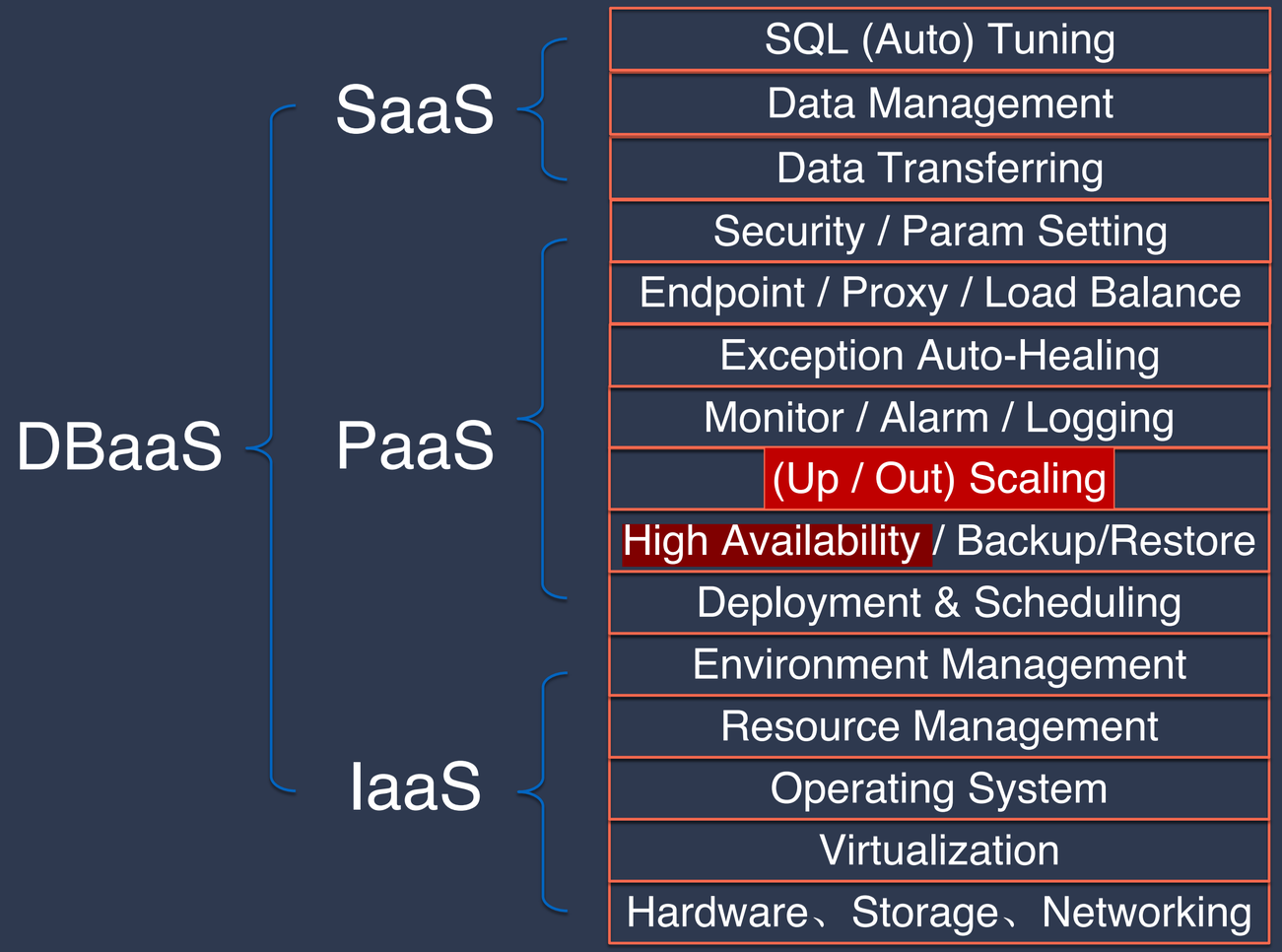
云原生数据库



TDSQL-C MySQL / PostgreSQL：

- Master 和 RO 基于一份数据，放在共享存储
- Master，仅将WAL写入共享存储、Page 不写
- RO，只从共享存储中读取所需 Page，无须写入存储
- RO，从主库接收 WAL，并在缓存中重放，保持缓存中 Page 持续更新
- WAL 在 共享存储中进行重放，实现存储节点上 Page 页的修改
- 存储层以 Page 为单位维护数据

云原生数据库，在做些什么



云原生数据库，在做些什么

云原生数据库 TDSQL-C MySQL / PostgreSQL：

- 即开即用，无须运维
- 相比于云数据库，本质上解决的是 Out Scaling 问题
- 当扩展能力达到极致（时间、空间），引起质变，即 ServerlessDB

你不一定非要分布式

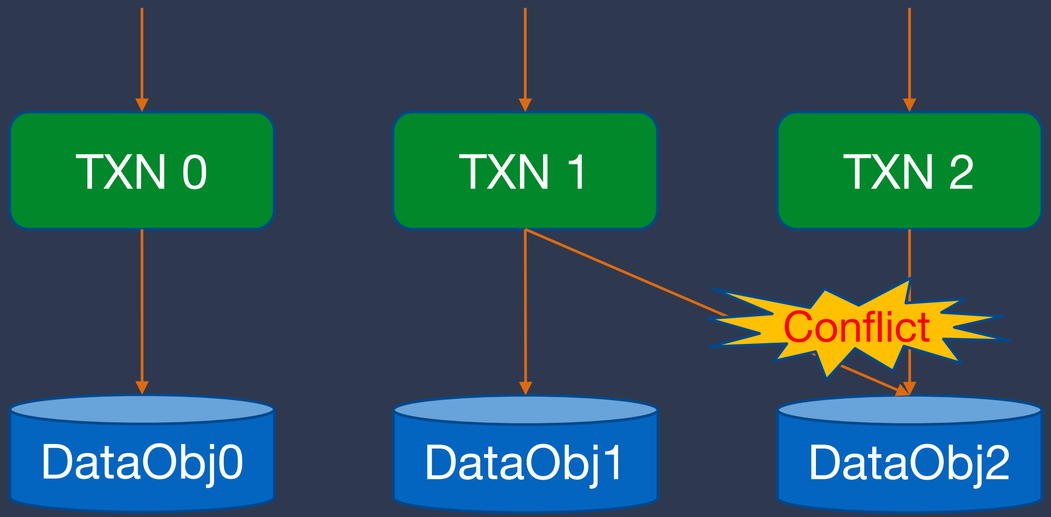
你不一定非要分布式

数据库水平扩展（Out-Scaling）的几个方案：

- 分库分表
- 分布式数据库
- 云原生数据库

你不一定非要分布式

分库分表——基于并发控制的本质



冲突：

1.TXN 0 与其他事务无关

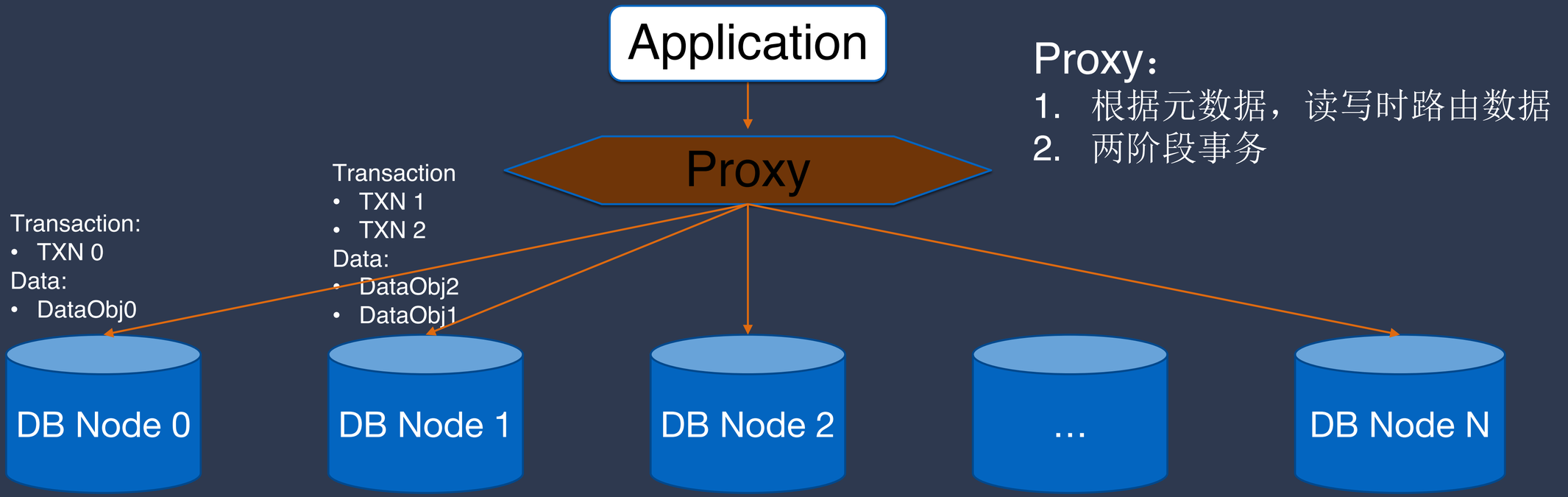
2.TXN 1 与 TXN 2 存在冲突问题，需要解决

3.TXN 1 对 DataObj1 和 DataObj2 的操作需要同时完成或同时失败

4.冲突则分为：读写冲突、写读冲突、写写冲突

你不一定非要分布式

分库分表——原理



Proxy：

1. 根据元数据，读写时路由数据
2. 两阶段事务

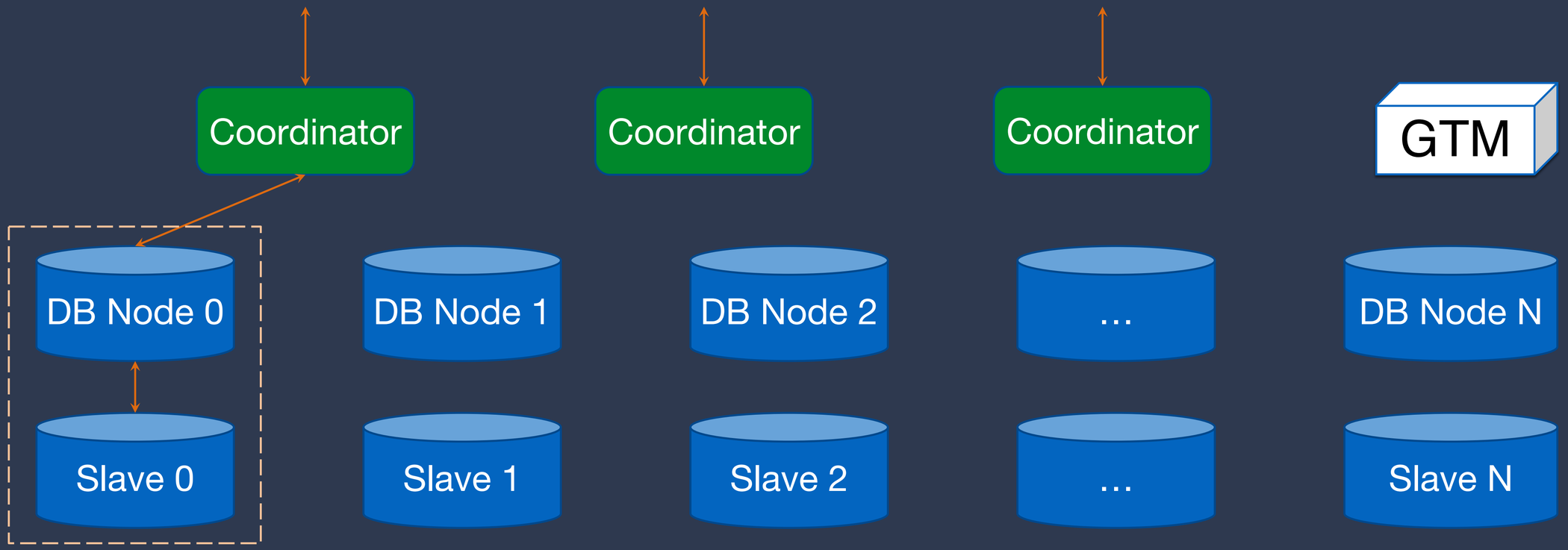
你不一定非要分布式

分库分表——弊端

- 与 业务逻辑 强绑定，需要业务做大量梳理、改造
- 扩容成本高、时间长，需要***新加机器、数据迁移，且计算与存储同步扩容***
- 功能阉割，比如，作为关系型数据库却**无法（或很难）**实现关系运算

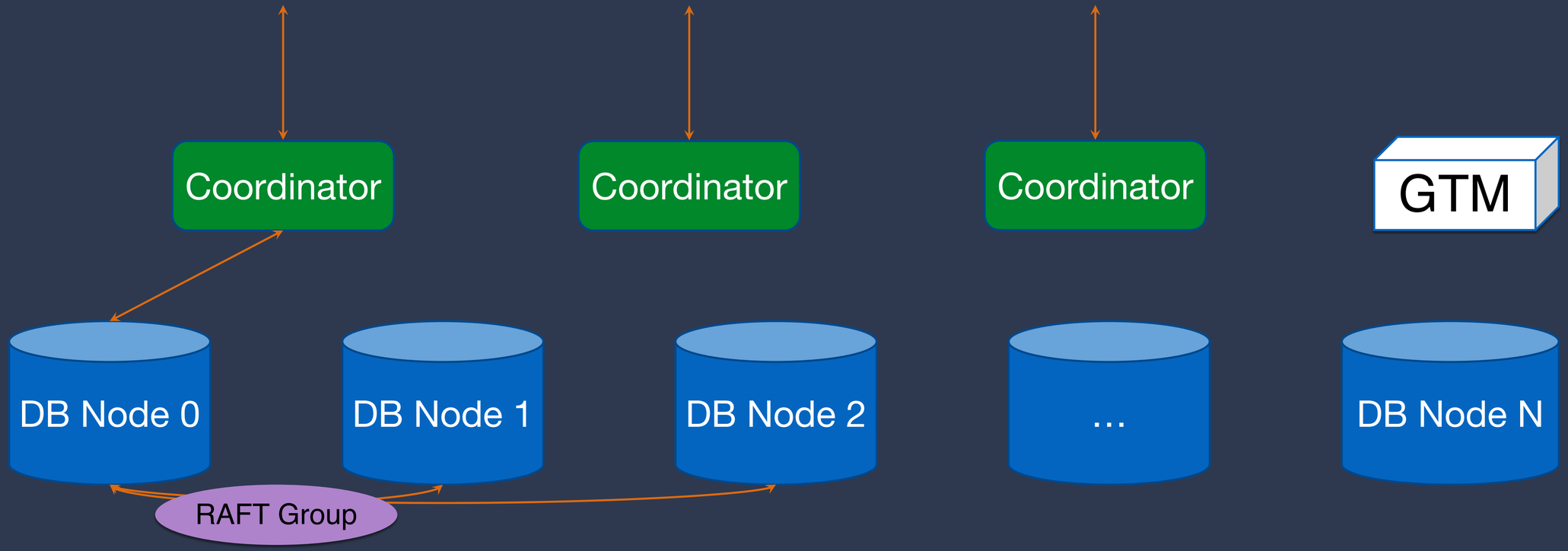
你不一定非要分布式

分布式数据库——数据分片、两阶段、一致性算法



你不一定非要分布式

分布式数据库——数据分片、两阶段、一致性算法



你不一定非要分布式

分布式数据库——弊端

- 架构复杂，维护成本高
- 分片字段选择 极大影响 性能表现，主要是 关系运算（JOIN）
- SQL性能优化 较为复杂
- 不适合 中小 规模

Not For ALL, For the MOST

Not For ALL, For the MOST

那些声称能够解决所有问题的，都是耍流X

No Silver Bullet!

Not For ALL, For the MOST

云原生数据库 能力的边界：

- 存储空间 128 T
- 单节点 计算资源上限
- RO 节点最多 15 个
- 最小规格 1vCore\2GB\10GB

TDSQL-C MySQL
TDSQL-C PostgreSQL

Not For ALL, For the MOST

云原生数据库 能力的边界：

- 存储空间 128 T，适配绝大多数 TP 场景
- 单节点 计算资源上限，主流 96 vCore，满足大部分计算需求
- RO 节点最多 15 个，集群 96 * 16 = 1536 vCore，绝大部分 HTAP 负载
- 最小规格 1vCore\2GB\10GB，不能再小啦

Not For ALL, For the MOST

一些统计：

- 4 vCore 计算规格及以下的实例数，占比 80.85%
- 1 T 存储空间及以上的实例数，占比 19.94%

Not For ALL, For the MOST

选型*建议*：

- **云数据库**
  - 空间小于 6T
  - RO 数量要求较少

- **TDSQL –C MySQL / TDSQL-C PostgreSQL**
  - **水平扩展下，与传统数据库一致的用户体验**
  - 大于 3T 小于 128T，几十T为宜
  - HTAP类业务
  - 极致弹性要求

- **分布式数据库，如 TDSQL**
  - 数十T、超百T及以上规模的场景
  - 非成本敏感
  - 有一定运维能力

- **分库分表**
  - 不介意业务改造
  - 业务逻辑简单，**没有或较少** 复杂计算
  - 事务简单、作用范围小

Not For ALL, For the MOST

一些故事：

1. 小规格大存储1vCore2GB 2TB
   - 业务复杂度不高
   - 数据需要持续保存
   - **成本敏感**，不愿升级 计算规格

选型*建议*：

- **TDSQL-C PostgreSQL**
  - 放宽 计算与存储 间约束
  - **分级存储**，热数据放共享存储、**冷数据放在COS**
  - 一致的处理方式
- **云数据库**
  - 因为成本问题，计算与存储 互相约束

传统数据库，无法处理、或成本较高

Not For ALL, For the MOST

一些故事：

2. 30TB，少量事务、大量分析
- 腾讯内用户
- 大量分析，且属于临时性
- 事务修改较少、批量导入

选型*建议*：

- **TDSQL-C PostgreSQL**
- 数据分析时，**建立 RO，用完立即释放**
- Master 专用于导入数据，存储随便写

Not For ALL, For the MOST

一些故事：

3. 微信支付

- 超大量、规模庞大，**远超 128T**
- 交易类数据，事务逻辑复杂
- 数据分析的逻辑复杂

选型*建议*：

- **TDSQL**

THANKS

软件正在改变世界

SOFTWARE IS CHANGING THE WORLD