OceanBase 事务引擎介绍

颜然/韩富晟
yanran.hfs@antfin.com

OceanBase

IBM解决方案

System 360

System 390

IBM z14

OceanBase

从小型机到数据中心演化

蚂蚁金服 ANT FINANCIAL

DEC PDP8

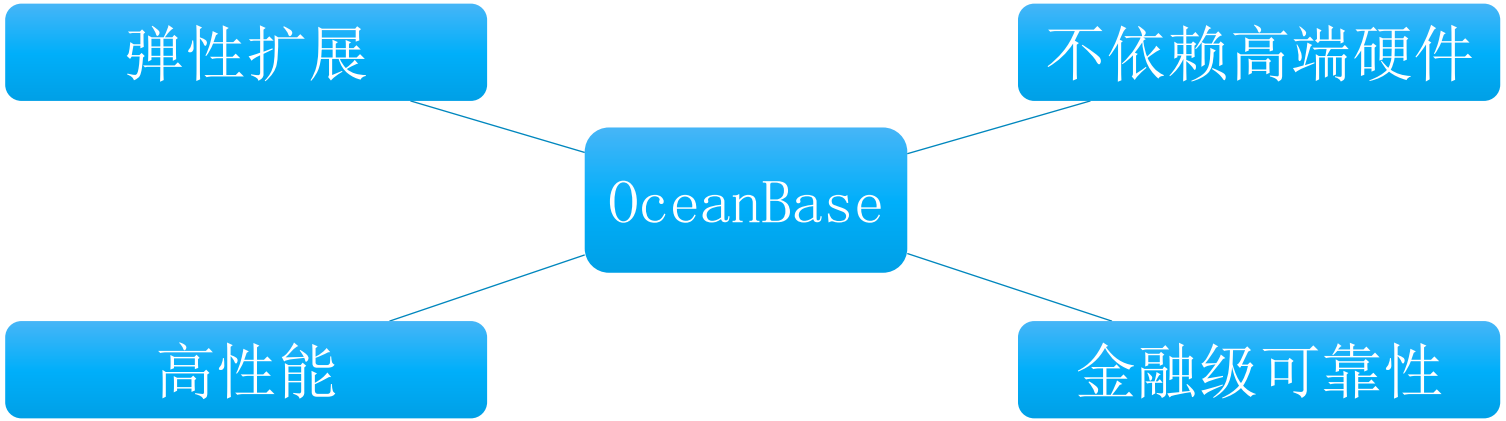
SUN Workstation

PC Server

Datacenter

OceanBase

OceanBase 事务



✓OB 架构下事务 ACID 的实现方式：

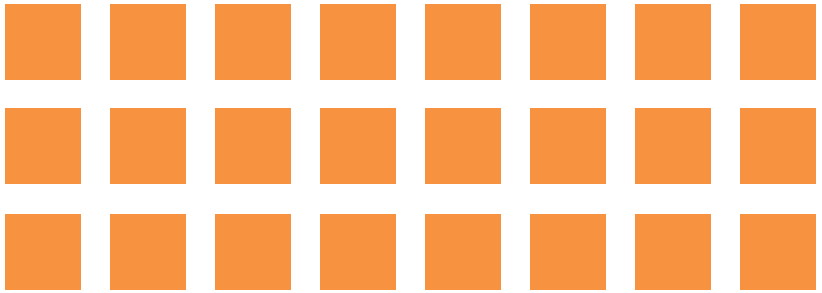
- Durability: 事务日志使用 Paxos 做多副本同步
- Atomicity: 使用两阶段提交保证跨机事务原子性
- Isolation: 使用多版本机制进行并发控制
- Consistency: 保证唯一性约束

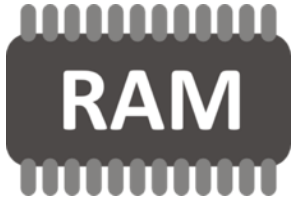
目录

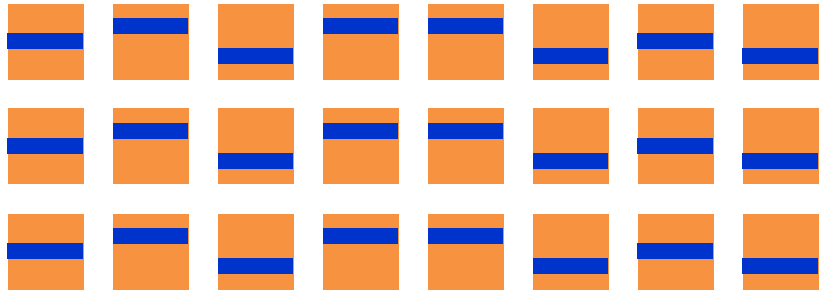
蚂蚁金服 ANT FINANCIAL

OceanBase

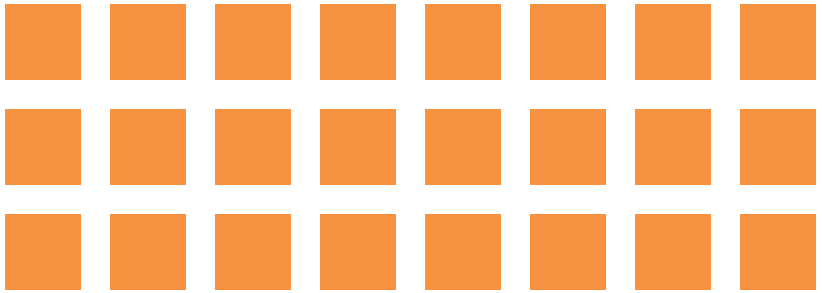
OceanBase 存储引擎







基线数据



增量修改

OceanBase

OceanBase 存储引擎

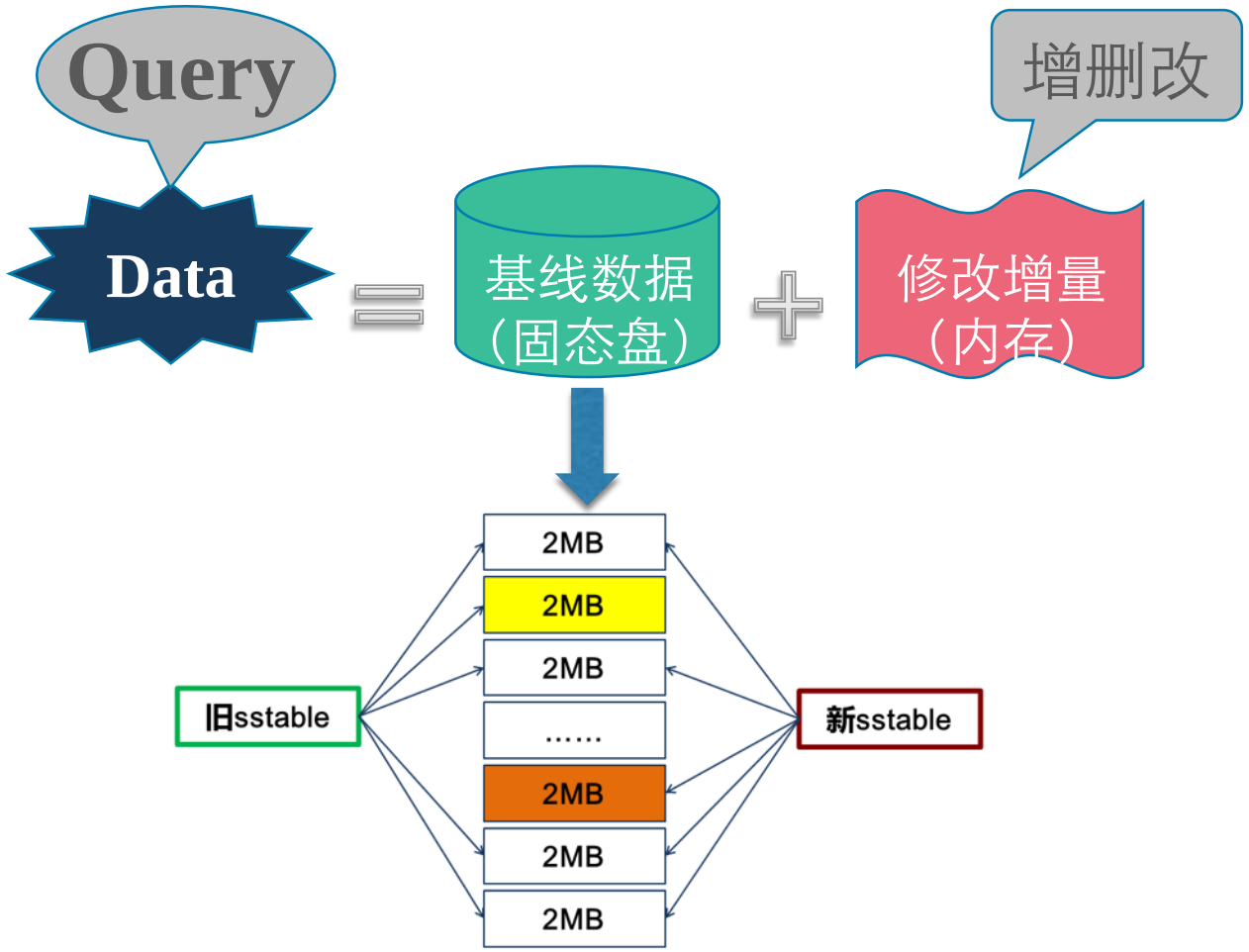
蚂蚁金服 ANT FINANCIAL

✓适应存储介质的数据组织方式

- 固态盘随机读效率高，随机写效率差
- 硬盘存储基线数据以批量的方式写入
- 内存存储增量修改，存储到一定量后与基线数据合并。合并后形成新的基线数据，已合并的内存增量清空。

✓为短事务优化

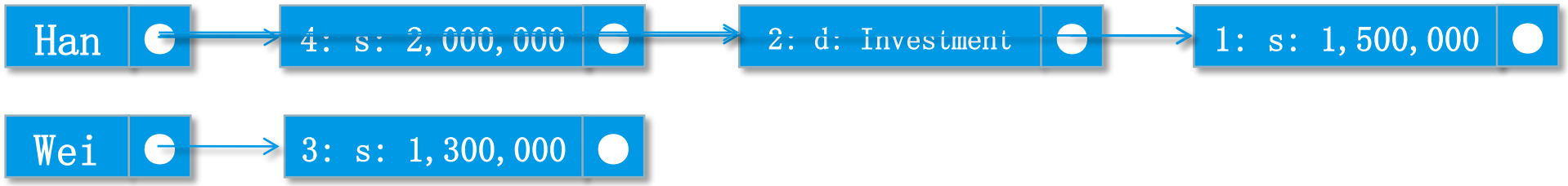
- 实际系统中绝大部分事务都是短事务
- 事务状态不持久化
- 不需要 undo



OceanBase

内存事务引擎

name	salary	department	phone
Han	1,000,000	Research	17012341234
Sun	800,000	Sales	17013571357
Wei	1,200,000	Marketing	17014701470




```
UPDATE staff SET salary = salary + 500000 WHERE name = 'Han'
```

```
UPDATE staff SET department = 'Investment' WHERE name = 'Han'
```

```
UPDATE staff SET salary = salary + 100000 WHERE name = 'Wei'
```

```
UPDATE staff SET salary = salary + 500000 WHERE name = 'Han'
```

内存事务引擎

蚂蚁金服 ANT FINANCIAL

✓内存存储

- 修改产生的多版本以内存链表形式表达，效率高

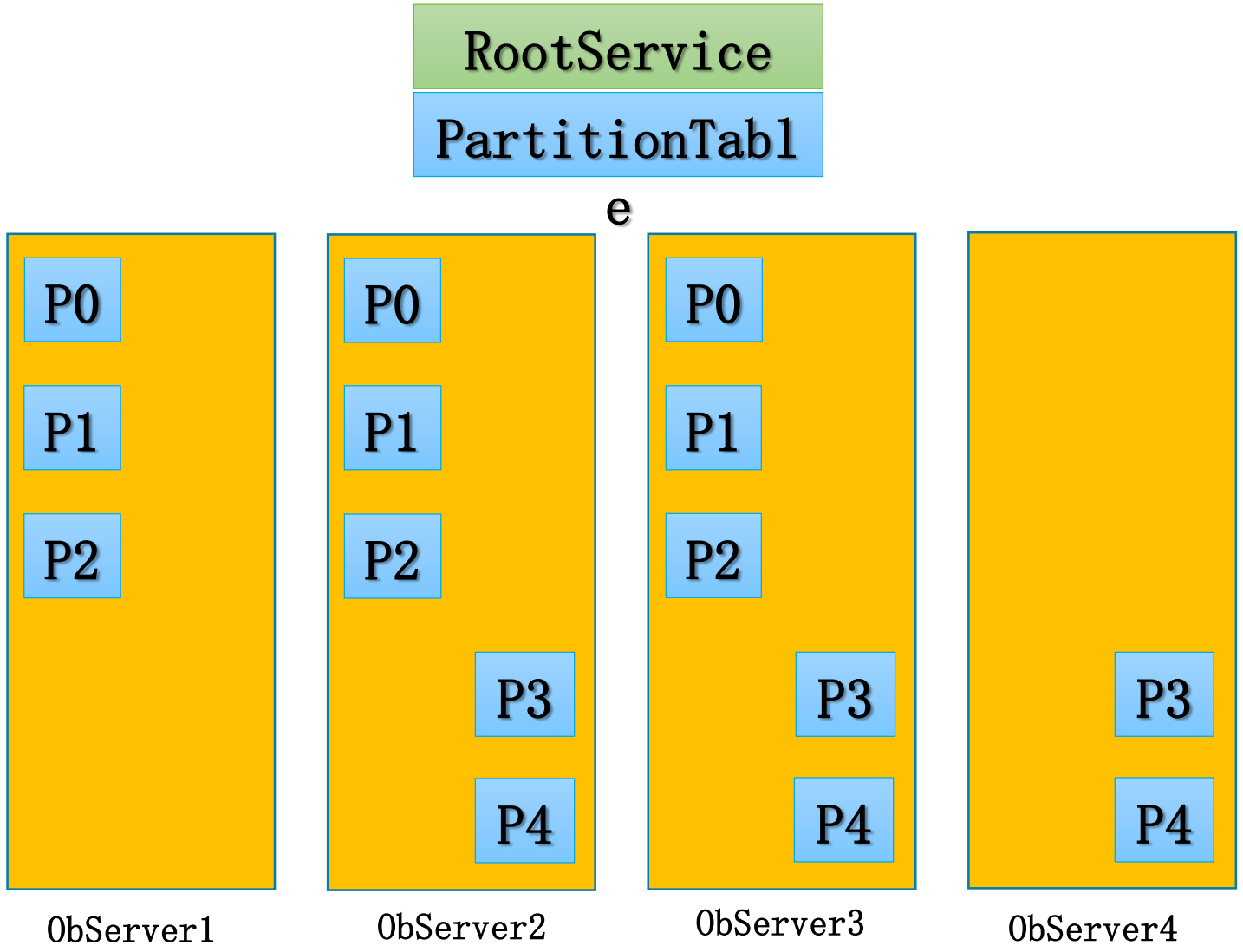
OceanBase

OceanBase 数据组织与分布

蚂蚁金服 ANT FINANCIAL

✓数据组织与分布

- 数据以 Partition 进行组织（一个 Table 有一个或者多个 Partition）
- 全局的 PartitionTable 是 Partition 的元信息，记录了 Partition 的位置
- 每个 Partition 拥有独立的日志服务
- 涉及多个 Partition 的事务都是分布式事务



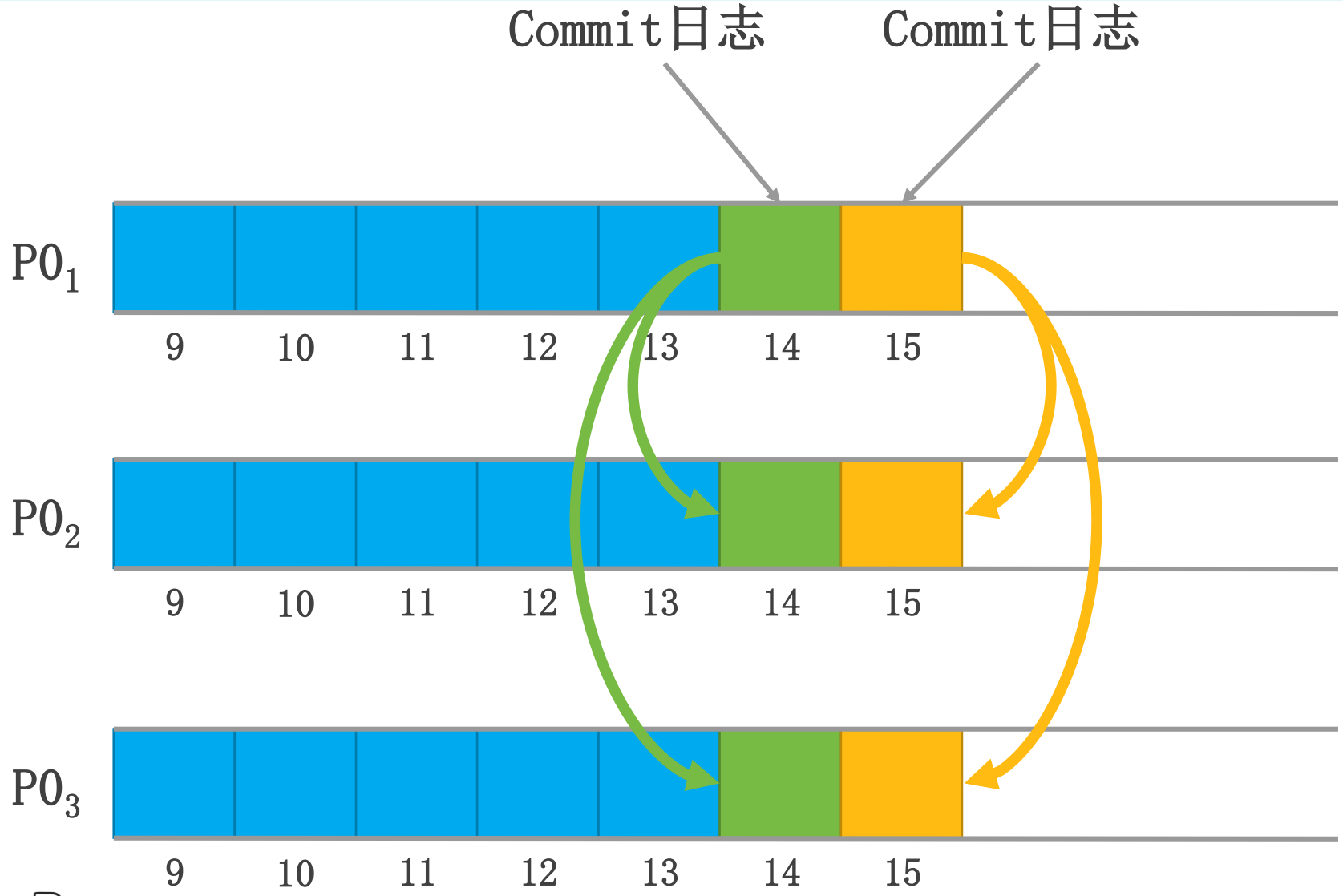
可靠性 vs 可用性

蚂蚁金服 ANT FINANCIAL

- ✓传统数据库的事务模型 ACID 没有描述**可用性（Availability）**
- ✓业务系统切实需要**可用性**来保证服务能力
- ✓传统数据库的解决方案：最大保护（Max Protection），最大可用（Max Availability），最好性能（Max Performance）
- ✓OceanBase 采用 Paxos 协议兼顾**可靠性**和**可用性**

OceanBase

Paxos 日志同步



OceanBase

分布式事务

蚂蚁金服 ANT FINANCIAL

✓难点：

- 分布式事务依赖**二阶段提交**保证事务的原子性
- 二阶段提交：状态多、复杂、状态机容易卡住

✓OceanBase 二阶段提交协议的特点：

- 所有参与者都是高可用的
- 协调者不写日志，无持久化状态
- commit 操作延迟低
- 全自动处理异常情况

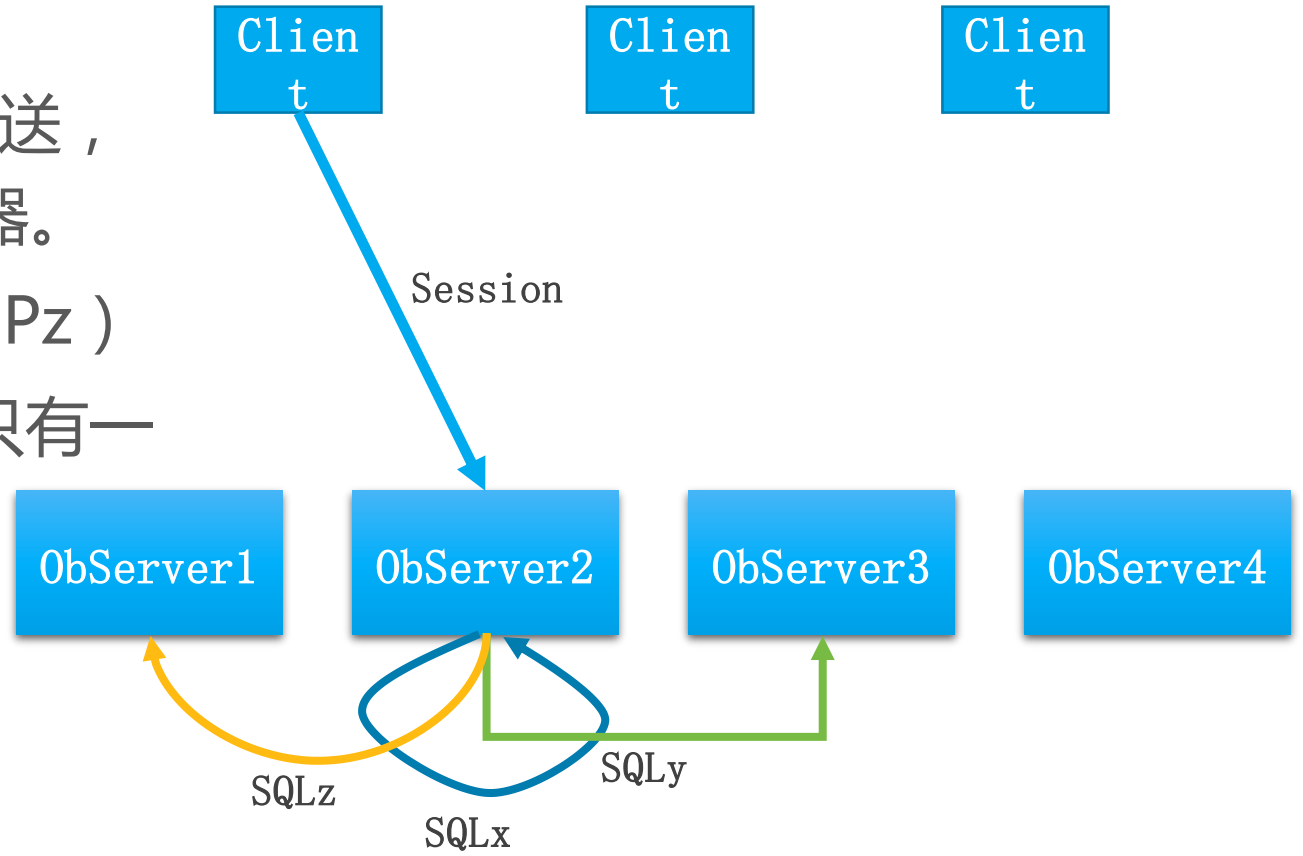
OceanBase

分布式事务的执行模型

蚂蚁金服 ANT FINANCIAL

✓二阶段提交参与者

- 事务的所有 SQL 都通过 session 传送，参与者列表维护在 session 所在机器。
- 参与者的实体是 Partition（Px, Py, Pz）
- 业务不感知是否分布式事务，如果只有一个参与者则使用一阶段提交，否则自动使用二阶段提交



OceanBase

二阶段提交流程

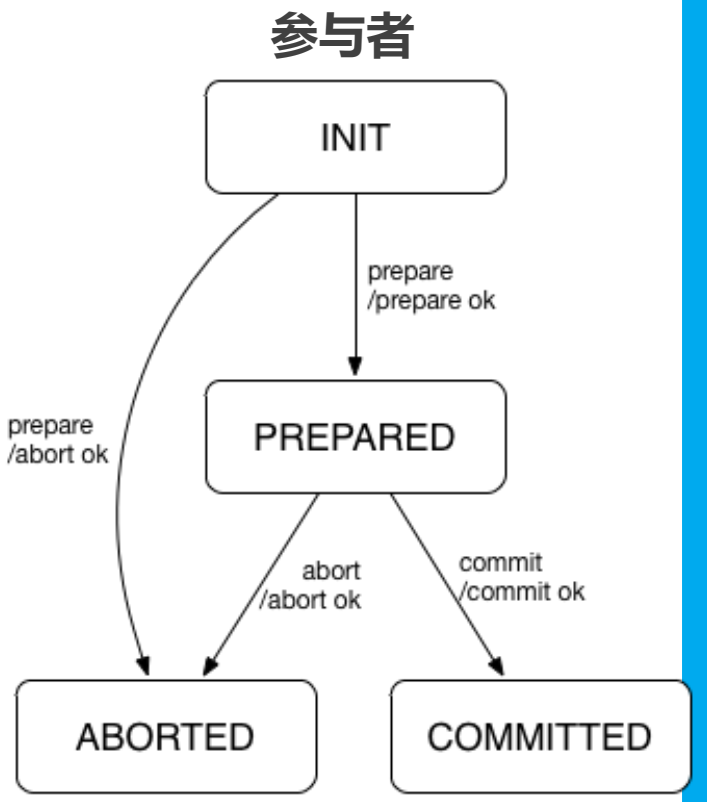
蚂蚁金服 ANT FINANCIAL

✓Commit 命令

- 指定第一个参与者 Px 作为协调者，发送 end_trans 消息给 Px 并告知其参与者列表 Px, Py, Pz

✓协调者

- Px 收到 end_trans 消息创建协调者状态机
- 协调者进入 PREPARE 状态并向所有参与者发送 prepare 消息
- 收到所有参与者 prepare ok 应答后进入 COMMIT 状态并向所有参与者发送 commit 消息
- 如果有一个参与者应答 abort 则进入 ABORT 状态并向所有参与者发送 abort 消息



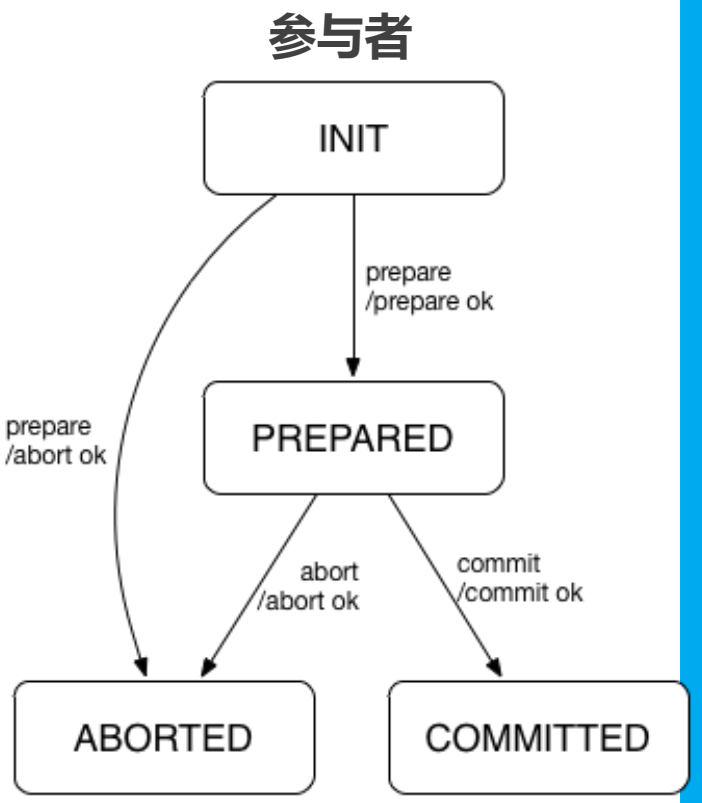
OceanBase

二阶段提交流程

蚂蚁金服 ANT FINANCIAL

✓参与者

- 参与者状态机是在 DML 语句执行过程中创建的
- 收到 prepare 消息后将事务的修改写日志（redo）
- 日志持久化成功后进入 PREPARED 状态并回复协调者 prepare ok 消息
- 收到协调者的 commit 消息则写 commit 日志、进入 COMMITTED 状态并回复 commit ok 消息
- 收到 abort 消息则写 abort 日志、进入 ABORTED 状态并回复 abort ok 消息



OceanBase

二阶段提交过程中参与者宕机

蚂蚁金服 ANT FINANCIAL

✓需要区分是否进入 PREPARED 状态两种情况考虑

✓已进入 PREPARED 状态

- 状态已经 Paxos 同步
- 系统会自动选择一个副本作为新的 Leader 并恢复出 PREPARED 状态，协调者继续推进，协调者可能感知不到

✓还未进入 PREPARED 状态

- 参与者的所有事务状态丢失
- 参与者会应答协调者 prepare unknown 消息
- 事务最终会 abort

OceanBase

二阶段提交过程中协调者宕机

蚂蚁金服 ANT FINANCIAL

- ✓协调者与第一个参与者是同一个 Partition，参与者状态机恢复遵从参与者自己的逻辑，协调者状态机的恢复由参与者回复协调者的消息触发
- ✓参与者未收到协调者消息时，认为上一条回复消息丢失，会定时重发发送上一条消息
- ✓正常流程里 Px 收到 end_trans 消息会创建协调者状态机
- ✓异常流程里 Px 收到 prepare ok, commit ok, abort ok 消息会创建协调者状态机，协调者会继续推动状态机继续运行
- ✓所有参与者都记录参与者列表

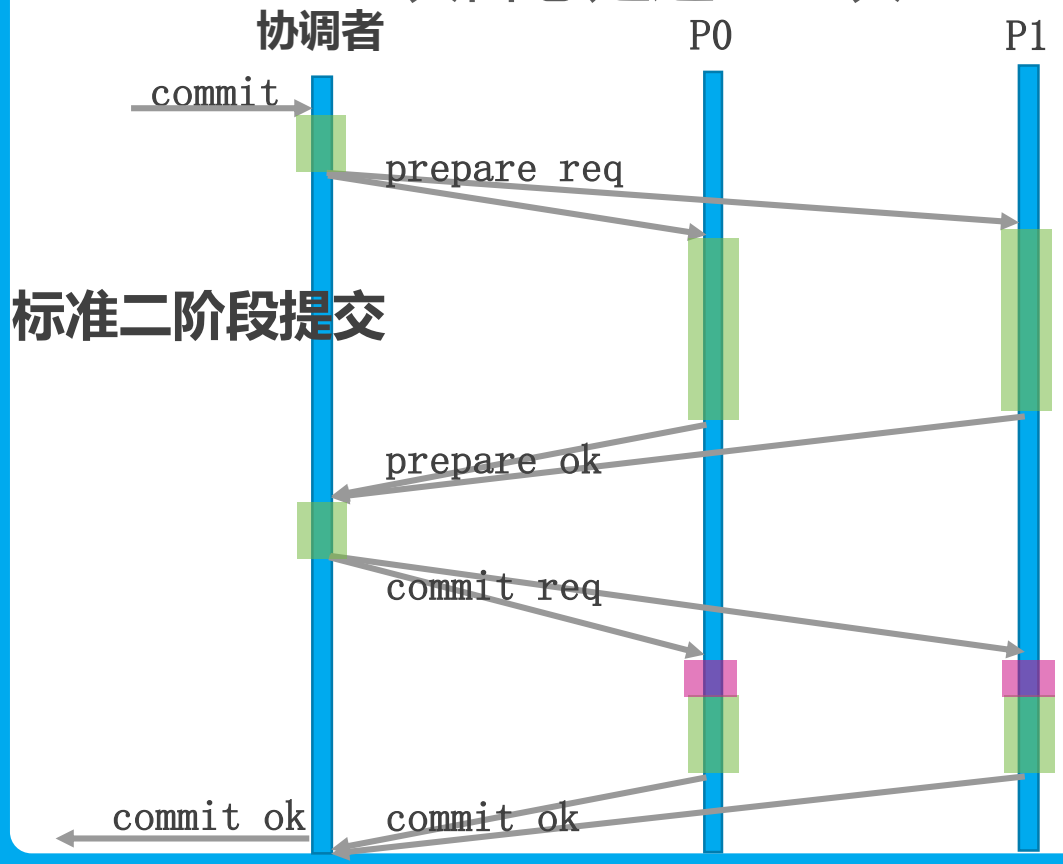
OceanBase

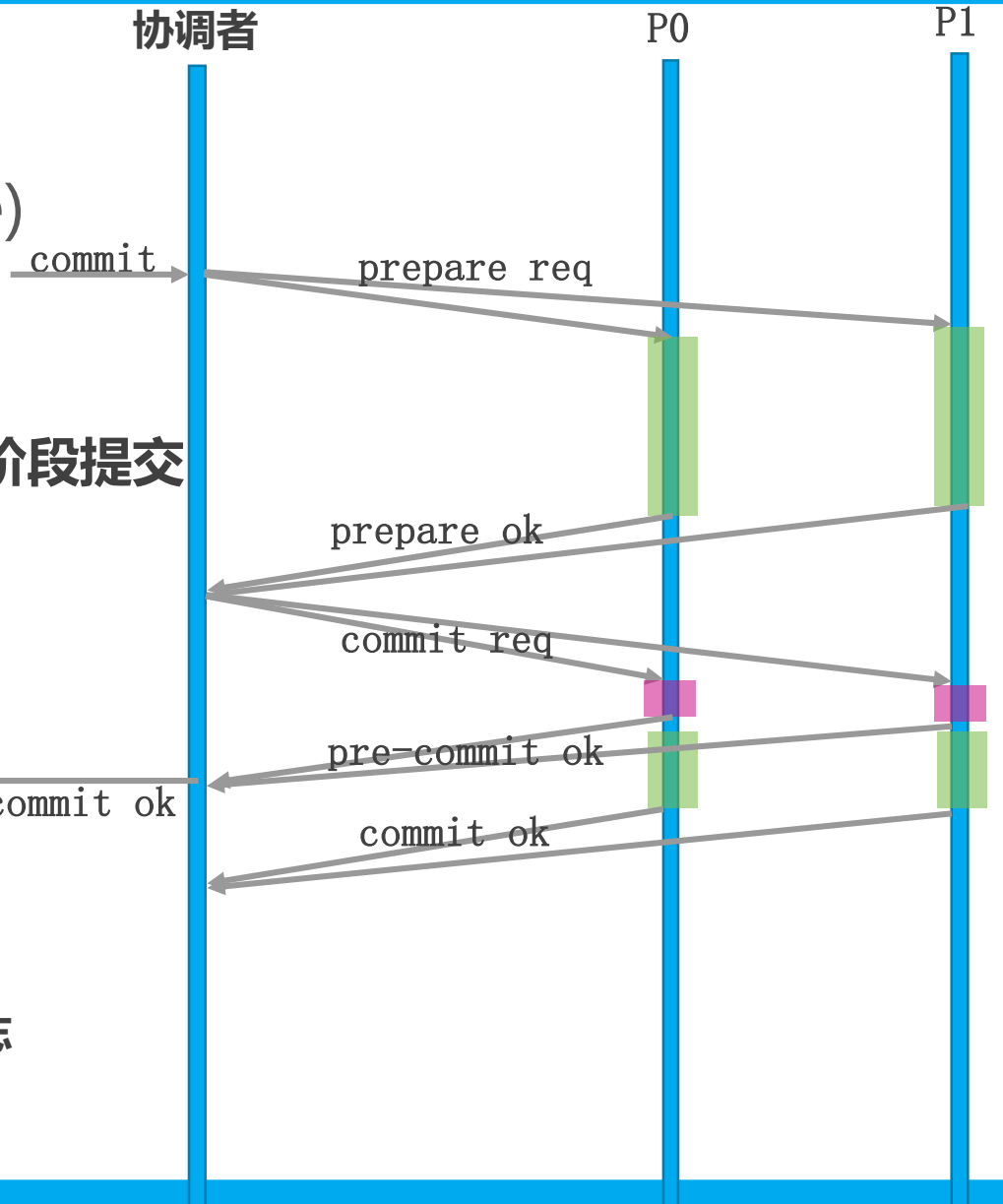
二阶段提交延迟分析

蚂蚁金服 ANT FINANCIAL

✓用户感知的 commit 延迟

- 标准：4次日志延迟 + 2次 RPC 延迟(同zone)
- OB：1次日志延迟 + 2次 RPC 延迟(同zone)





持久化日志

事务提交

OceanBase 事务隔离

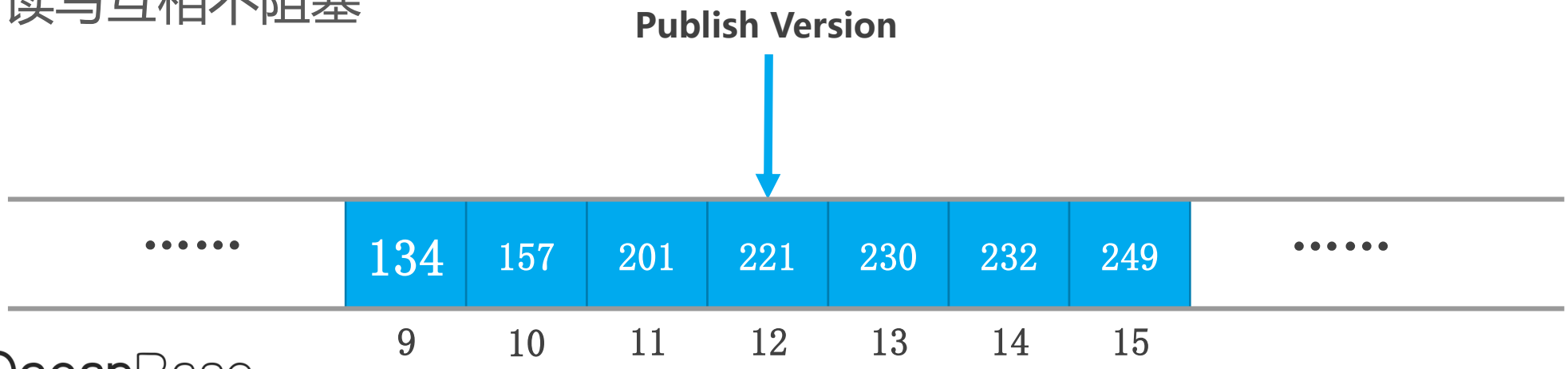
蚂蚁金服 ANT FINANCIAL

✓写-写并发

- 所有修改的行加互斥锁
- 写操作需要等待行上已有的行锁解锁

✓读-写并发

- 读操作读取事务快照版本（Publish Version）对应的数据
- 读写互相不阻塞



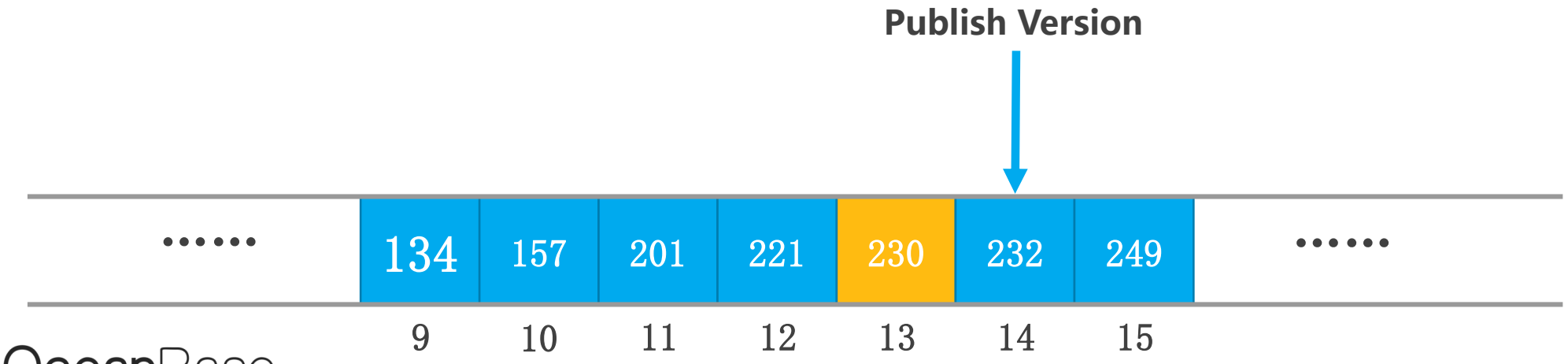
OceanBase

事务版本号

蚂蚁金服 ANT FINANCIAL

✓分布式事务多版本并发难点

- 系统中事务完成的版本号顺序无法严格递增
- 单纯基于快照版本无法做到读到正确的数据
- PREPARE 和 COMMIT 之间的事务已经产生了事务版本，但是事务状态还未确定，需要阻塞更大快照版本的读取



OceanBase

THANKS/感谢聆听

--------- Q&A Section --------

OceanBase 颜然/韩富晟