

蚂蚁金服分布式事务开源以及实践

绍 辉

蚂蚁金服 分布式事务开源负责人





目 录 contents

- 一、分布式事务产生的背景
- 二、蚂蚁金服分布式事务理论基础
- 三、蚂蚁金服分布式事务实践之路
- 四、蚂蚁金服分布式事务开源



一、分布式事务产生的背景

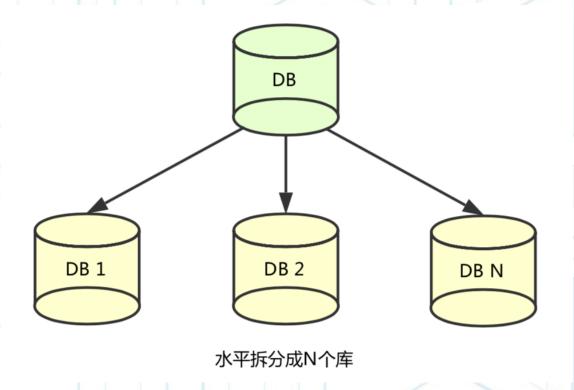






蚂蚁金服架构演进之 数据库水平拆分

分库分表之后,写操作会跨多个数据库



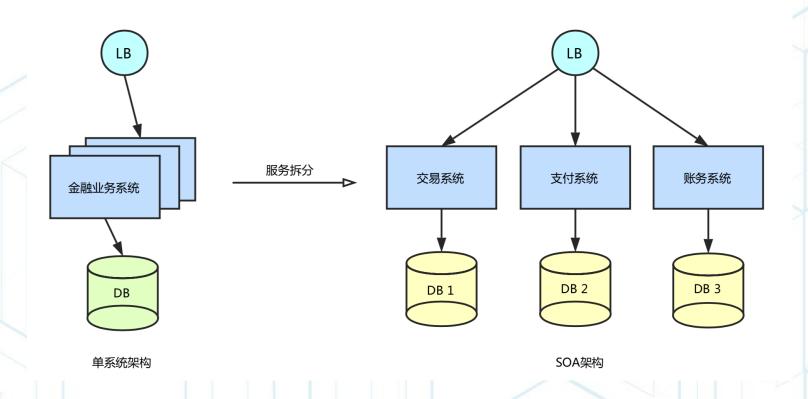






蚂蚁金服架构演进之 服务化拆分

业务服务化拆分之后,原一个服务就能完成的业务操作需要跨多个服务









如何解决数据一致性问题?

以转账场景为例:

一次转账,跨多个服务和数据库

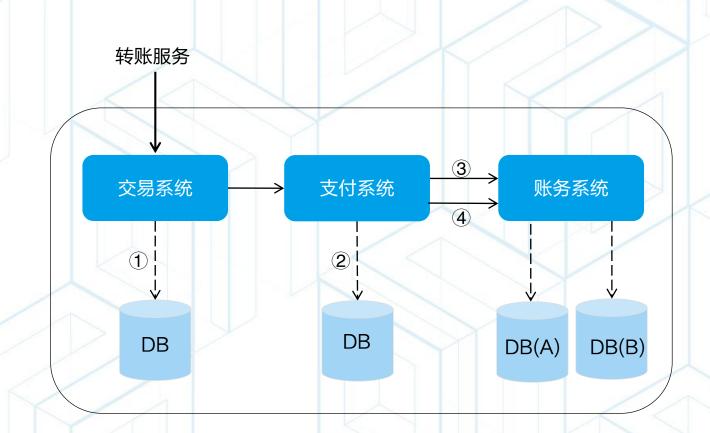
如何保证以下操作的原子性:

▶ 交易系统: 创建交易订单

支付系统: 记录支付明细

▶ 账务系统: A 扣钱

▶ 账务系统: B 加钱





二、蚂蚁金服分布式事务理论基础





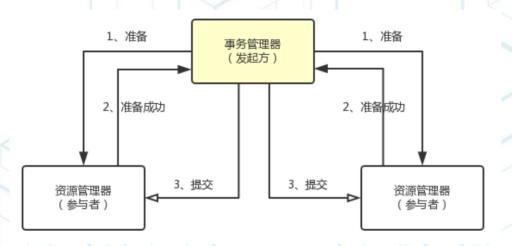
2PC

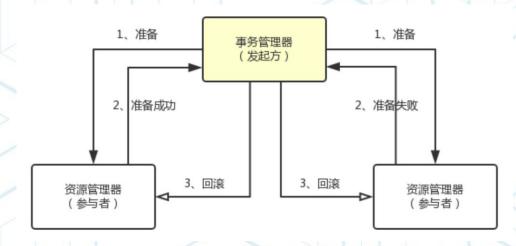
两阶段提交协议

(Two Phase Commitment Protocol)

事务管理器分 2 阶段协调资源管理器:

- ▶ 一阶段:准备
- ▶ 二阶段:资源提交/回滚









TCC

TCC (Try-Confirm-Cancel)

是服务化的两阶段,三个操作均需编码实现

一阶段: Try

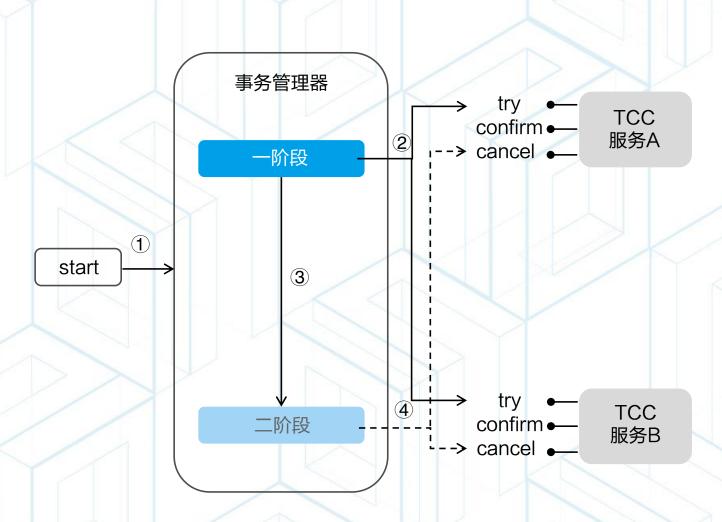
二阶段: Confirm/Cancel

三个操作描述:

➤ Try: 检测预留资源

➤ Confirm: 真正的业务操作提交

➤ Cancel: 预留资源释放





三、蚂蚁金服分布式事务实践之路

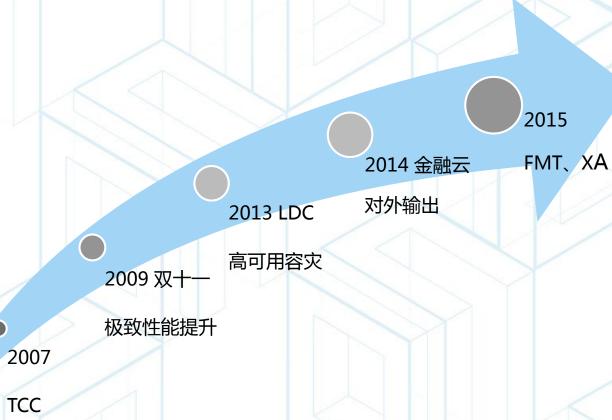






蚂蚁金服分布式事务

- ❖三种模式,丰富的应用场景:
 - TCC模式
 - FMT模式
 - XA模式
- ❖极致性能提升
- ❖高可用容灾
- ❖无侵入解决方案



SOFA Meetup#1 < 北京站 >

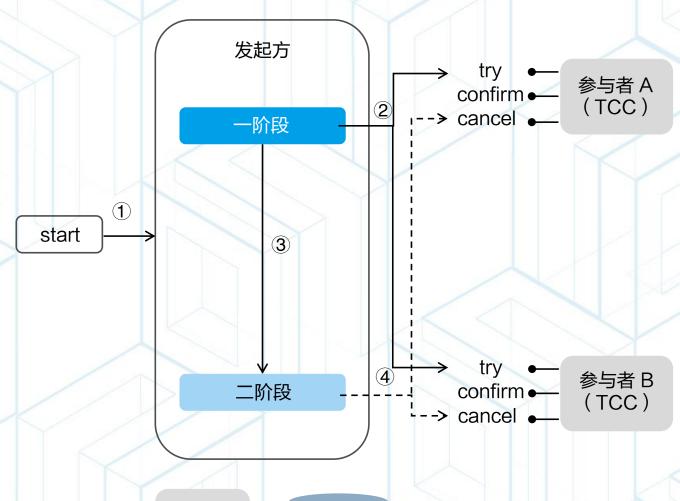






TCC 模式

- ❖ 事务管理器(发起方):
 - > 编排参与者
 - ▶ 两阶段提交
- ❖ 资源管理器(参与者):
 - ➤ 提供TCC服务



事务 恢复服务

事务 日志







TCC 实践经验

蚂蚁金服TCC实践,总结以下注意事项:

- 业务模型分 2 阶段设计
- 并发控制
- 允许空回滚
- 防悬挂控制
- > 幂等控制







TCC 设计 - 业务模型分2 阶段设计

扣钱场景为例: 账户 A 上有 100 元, 要扣除其中的 30 元

-阶段(Try):检查余额,预留其中 30 元;



二阶段提交(Confirm): 扣除 30 元;



二阶段回滚(Cancel): 释放预留的 30 元。









TCC 设计 - 并发控制

扣钱场景为例: 账户 A 上有 100 元, 事务 T1 要扣除其中的 30 元, 事务 T2 也要扣除 30 元, 出现并发

-阶段:检查余额,预留其中30元;

账户A:	T1冻结部分		T2冻结部分	可用余额	
0		30	6	0	100

T1 二阶段提交:扣除 T1 冻结金额;



T1 二阶段回滚:释放 T1 预留的 30 元。

账户A:	户A: 可用余额		T2冻结部分		可用余额		
0		30		60		100	



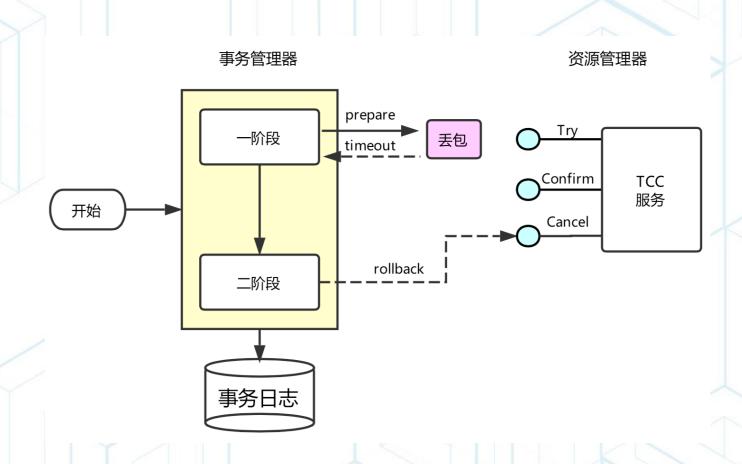
TCC 设计 - 允许空回滚

空回滚: Try未执行, Cancel执行了

出现原因:

- ▶ Try 超时(丢包)
- ▶ 分布式事务回滚,触发 Cancel
- ➤ 未收到 Try, 收到 Cancel

允许空回滚





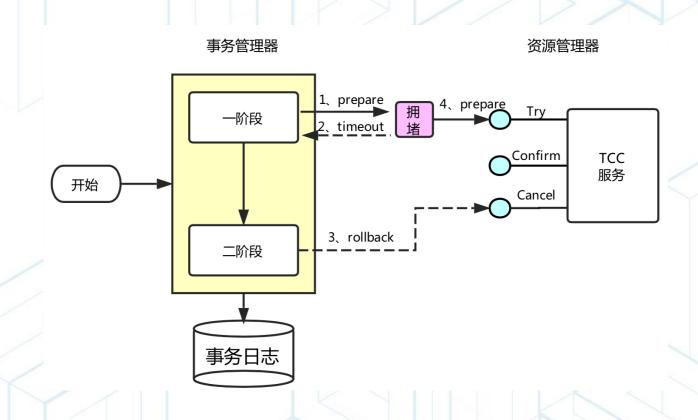
TCC 设计 - 防悬挂控制

悬挂: Cancel 比 Try 先执行

出现原因:

- ➤ Try 超时(拥堵)
- ➤ 分布式事务回滚,触发 Cancel
- ▶ 拥堵的 Try 到达

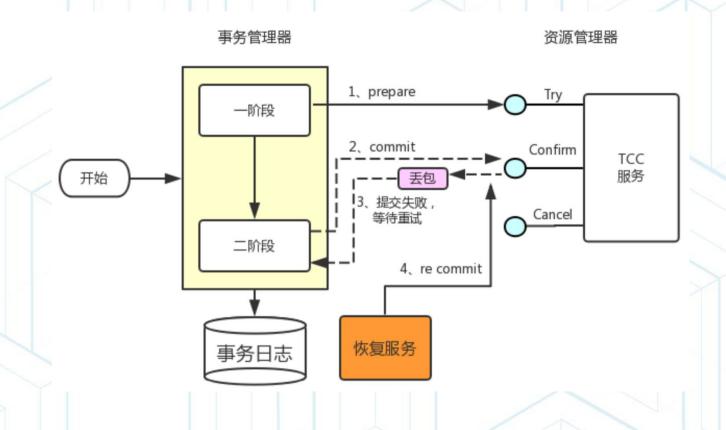
要允许空回滚,但要拒绝空回滚后的Try 操作





TCC 设计 - 幂等控制

Try、Confirm、Cancel 3 个方法均需保证幂等性









极致性能提升 - 支持双十一性能需求

性能提升措施:

❖锁粒度优化:

- ▶ 相对于 XA 等数据库锁, TCC 锁粒度更小
- > TCC 有极大优化空间,性能更强

❖同库模式改造:

- 事务日志与业务数据在相同库存储
- ➤ 优化减少 N+2 次 RPC, 性能提升 50%

❖二阶段异步执行:

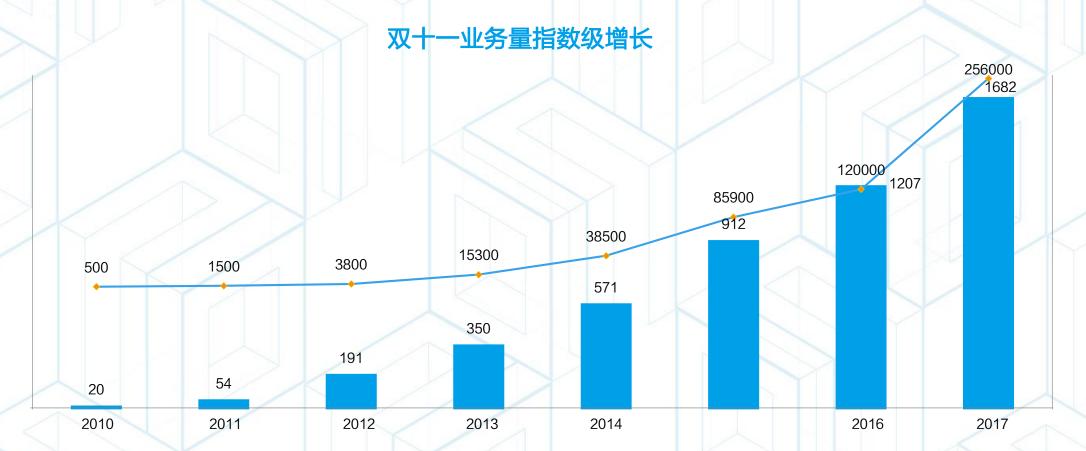
▶ 一阶段同步执行, 二阶段异步执行, 事务耗时减少 50% 左右







极致性能提升成果







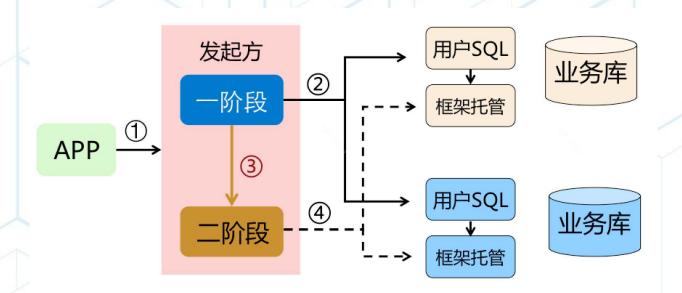


无侵入解决方案 - FMT 模式

FMT (Framework-managed transaction): 框架管理事务,托管事务的所有操作,一阶段和 二阶段操作均由框架自动生成

一阶段:用户SQL

▶ 二阶段:框架自动生成"提交/回滚"操作



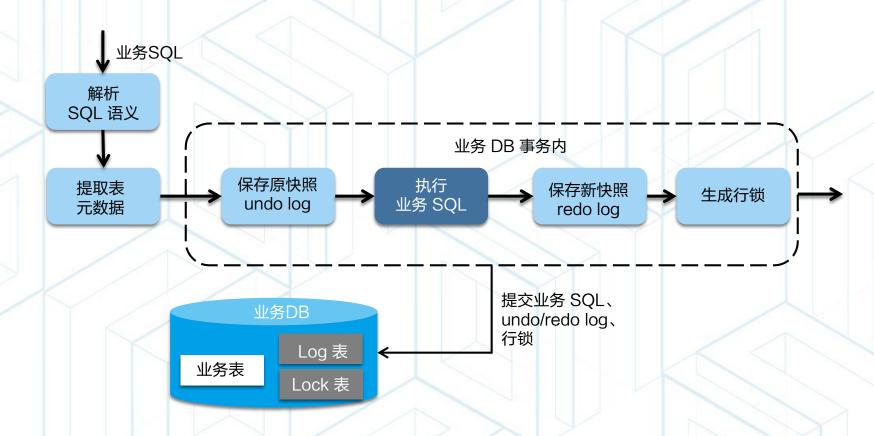




FMT 模式: 一阶段

拦截业务 SQL, 生成 undo log、redo log 等中间数据

- 解析SQL语义
- 提取表元数据
- 保存undo log
- ▶ 执行业务SQL
- 保存redo log
- > 保存行锁

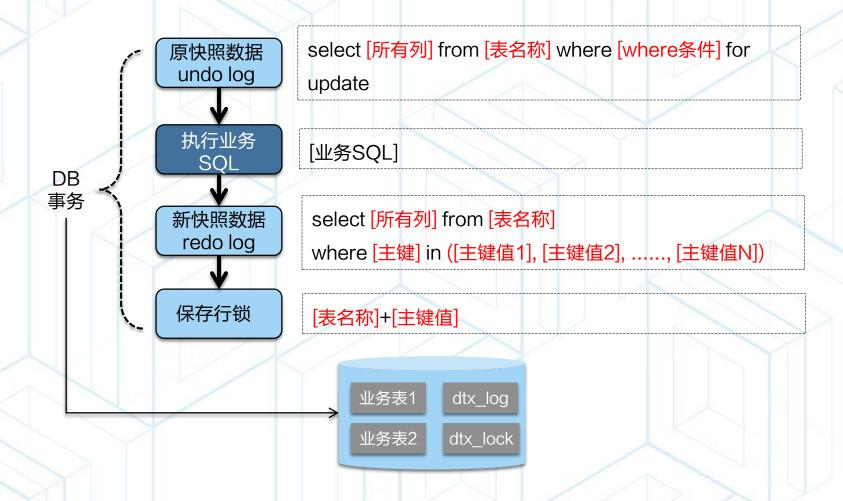








FMT 模式一阶段: undo log/redo log/行锁

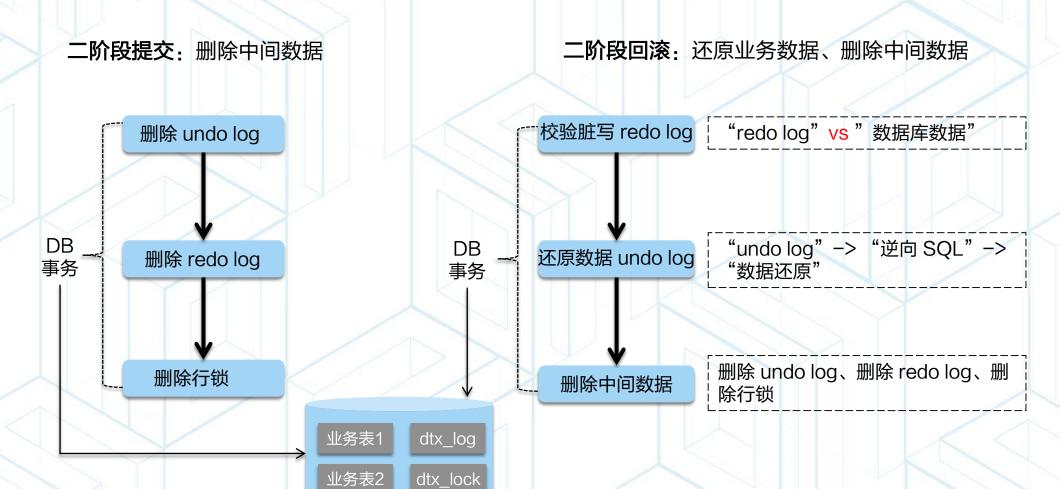








FMT 模式二阶段



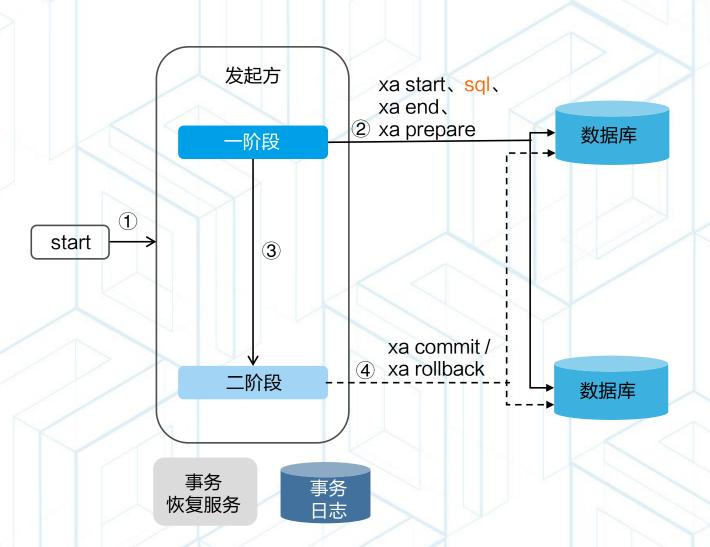






无侵入解决方案 - XA 模式

- > 支持标准 XA 协议,覆盖面广
- 与自研数据库 OceanBase 深度定制, 解决 XA 性能问题
- 支持全局一致性读





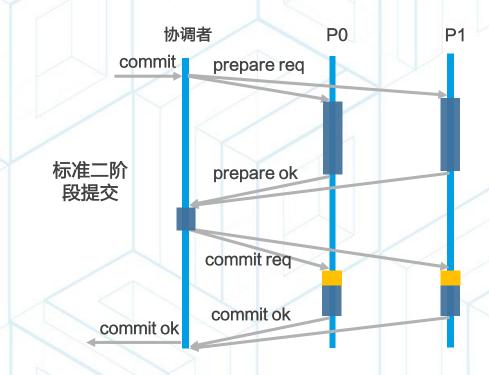


P1

无侵入解决方案 - XA 模式

与 OceanBase 深度定制, commit 性能优化:

- Commit 异步化
- 提交释放锁



commit OB 二阶段提交 commit ok

prepare ok commit req

prepare req

协调者

commit ok

pre-commit ok

clear req

clear ok

事务提交

持久化日志

SOFA Meetup#1 < 北京站 >







蚂蚁分布式事务-金融场景全覆盖

用户覆盖支付、理财、保险、银行等各类金融场景



支付宝



余额宝



花呗



借呗



芝麻信用



蚂蚁保险



信美相互



国泰产险



天弘基金



蚂蚁森林



网商银行



南京银行



重庆农商



天津银行



兴业银行







四、蚂蚁分布式事务开源







蚂蚁分布式事务开源版图

FMT/AT TCC - 集成 MySQL XA - 支持基本 SQL 语法 - 支持跨服务 - 行锁实现 - 集成 Dubbo - MySQL XA - 快照读 - 集成 SOFARPC - Oracle XA - 多隔离级别支持 - 防悬挂控制 - 幂等控制







阿里巴巴分布式事务

TXC 专有云输出 树立了大量的企业、政府、金融 等行业的标杆用户

GTS 公测 (Global Transaction Service) 阿里云服务对外公测

Alibaba Fescar 开源版本正式发布

2014.11

2015.06

2015.12

2016.08

2018.05

2019.01

TXC v1.0 (Taobao Transaction Constructor) 阿里系业务分布式事务解决方 案,主要应用于集团电商、新 零售和物流等微服务化场景

TXC v2.0 支持 MT (Manual Transaction) 模式,支持用户自定义补偿

GTS 商业化 阿里云商用版本正式对外提供商 用服务







阿里巴巴 + 蚂蚁金服 社区共建



TCC 模式:

- Dubbo TCC
- SOFARPC TCC
- Local TCC

服务端 HA 集群支

- 数据库存储
- Raft 模式
- 数据路由+Leader

XA

TCC

MySQL XA

• 防悬挂控制

• 幂等控制

Oracle XA

多语言支持

·监控、诊断

web console

容器化及资源调度

(2019.1

v0.2 (2019.2)

(2019.3

v0.4 (2019.3

v0.5 (2019.4

v1.0 (2019.5

NEXT

AT 模式:

- 支持 MySQL
- 支持 Dubbo 无缝集成
- 支持 Spring 注解事务

- Eureka 注册中心
- Zookeeper 注册中心
- 支持 GlobalLock/本 地事务模式

AT 模式:

- 数据库支持扩展: oracle、postgresql 等
- 复合主键支持
- 多事务隔离级别支持
- 行锁优化: 锁粒度细化、乐观锁

SOFA Meetup#1 < 北京站 >



欢迎关注 SOFAStack 公众号 获取分布式架构干货



使用钉钉扫码入群 第一时间获取活动信息



