TDSQL的
分布式事务处理技术

李海翔 @那海蓝蓝 @腾讯金融云

PostgreSQL, MySQL, Greenplum, Informix, CockroachDB, etc

@那海蓝蓝 Blog : http://blog.163.com/li_hx/

《数据库查询优化器的艺术: 原理解析与SQL性能优化》

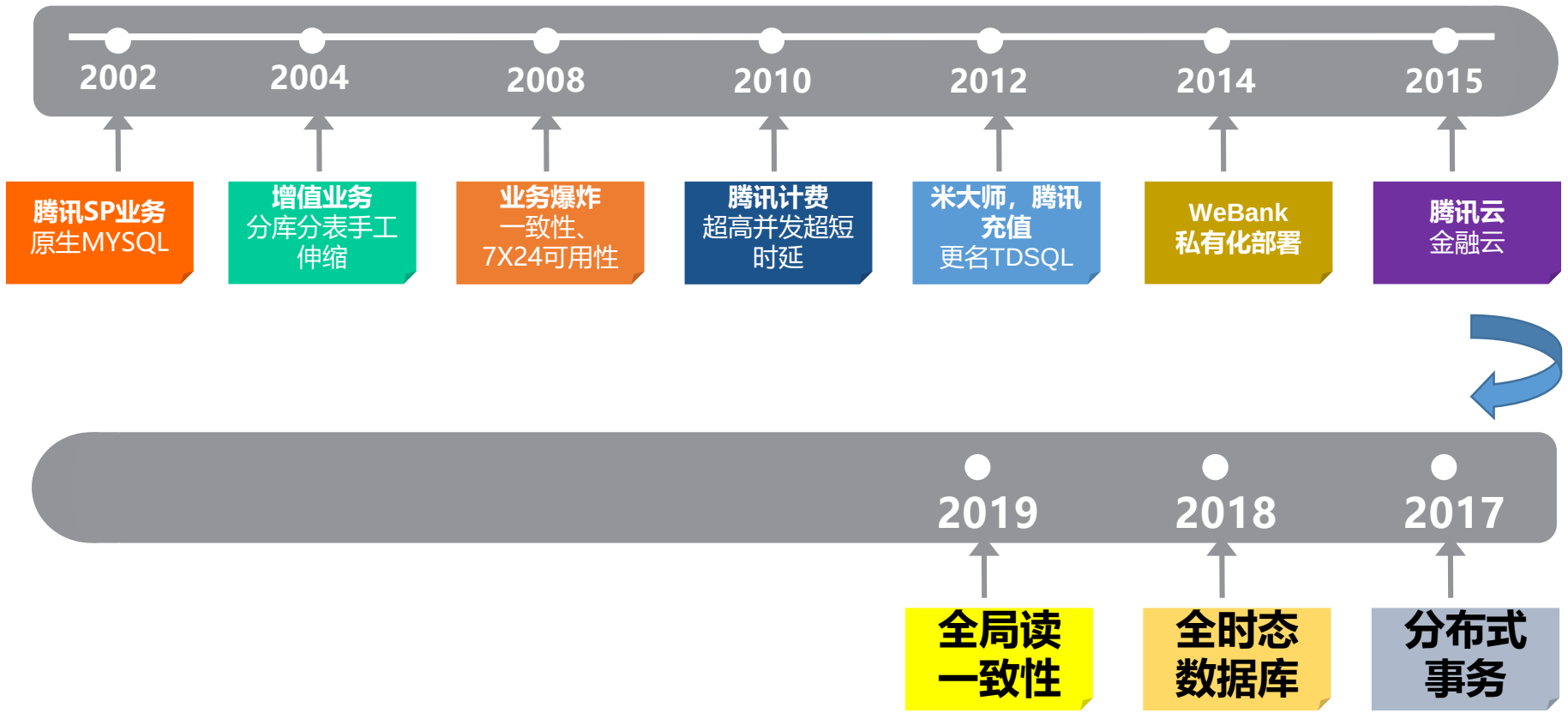
《数据库事务处理的艺术: 事务管理与并发访问控制》

目录

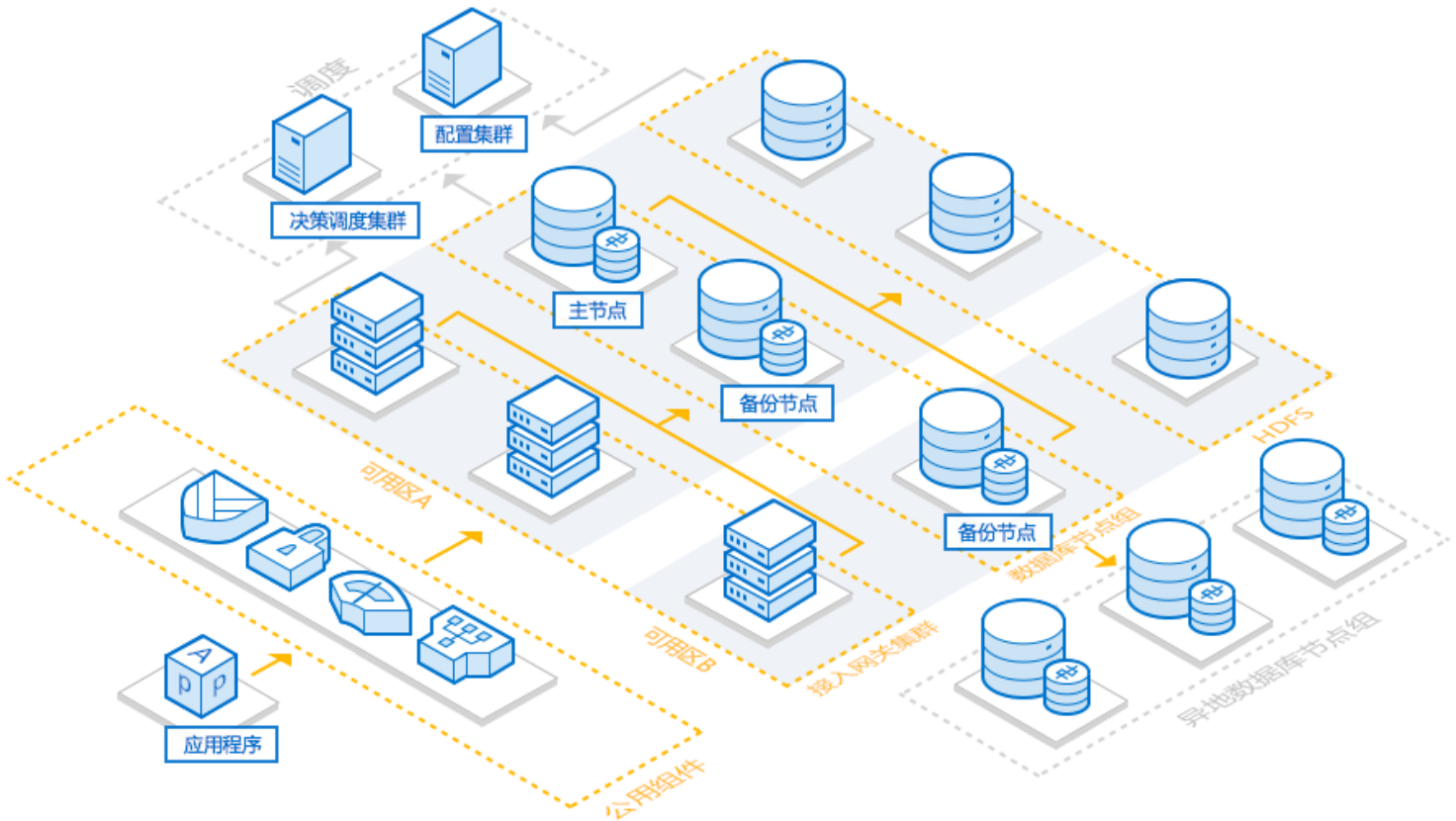
CONTENTS

面向金融类业务，十年积累，亿级账户验证

腾讯公司内与计费、充值、转账、财务等核心系统90%以上都使用TDSQL！



数据库部署架构



数据库节点组(SET)由MySQL数据库、监控和信息采集模块组成一主二从数据库节点。

调度集群作为集群的管理调度中心，主要管理数据库节点组、接入网关集群的正常运行

接入网关集群账号鉴权、管理连接、SQL解析、分配路由

分布式文件系统(HDFS)提供数据灾备服务，提供至少3份备份

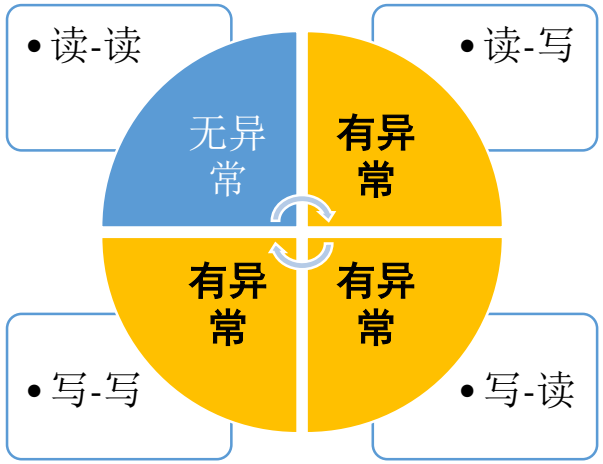
异地容灾数据库节点组部署在主节点以外的异地机房。

TDSQL 数据库的特点

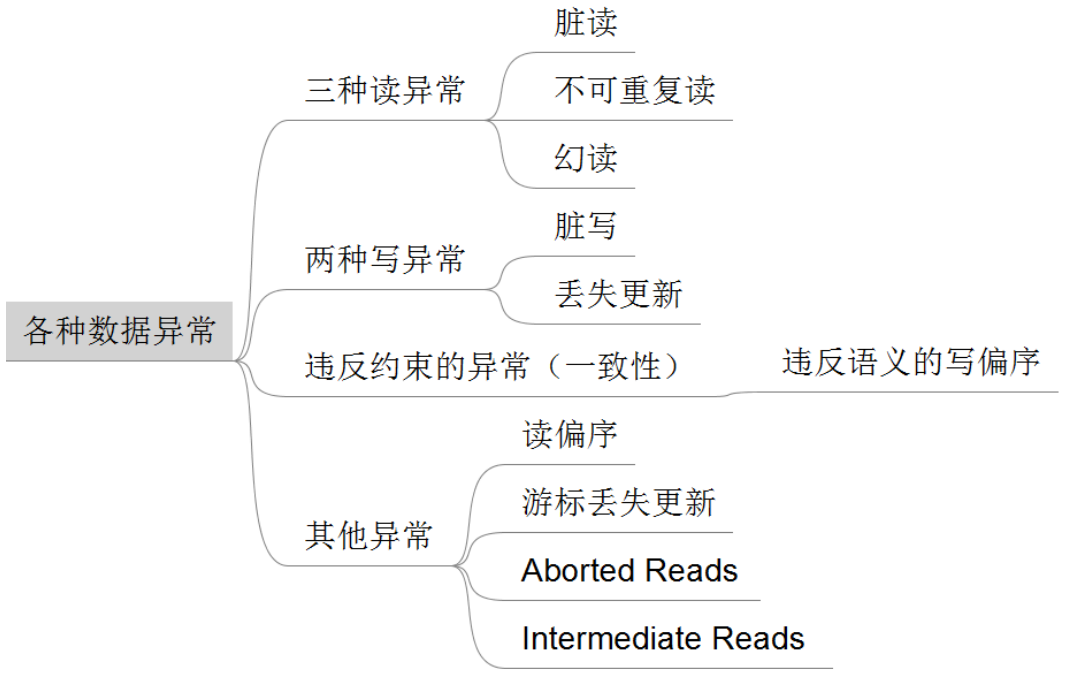
目录

CONTENTS

TDSQL单机事务处理--原理



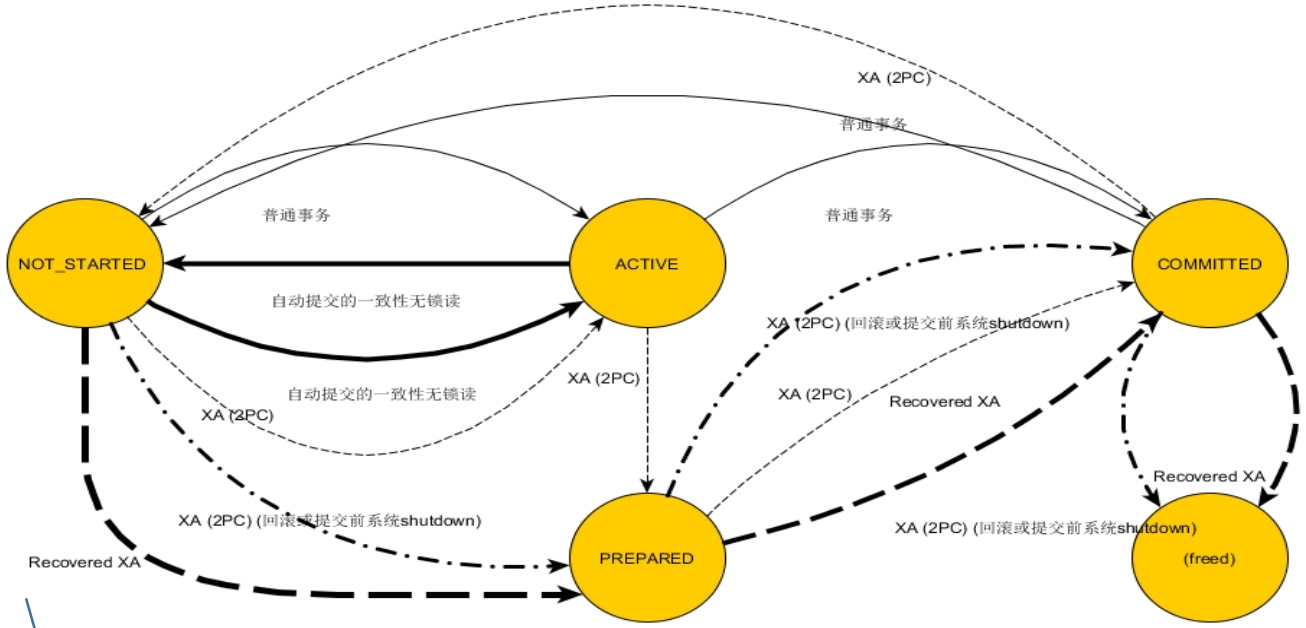
并发操作可以被区分为四种：读-读、读-写、写-读、写-写



TDSQL单机事务处理--原理

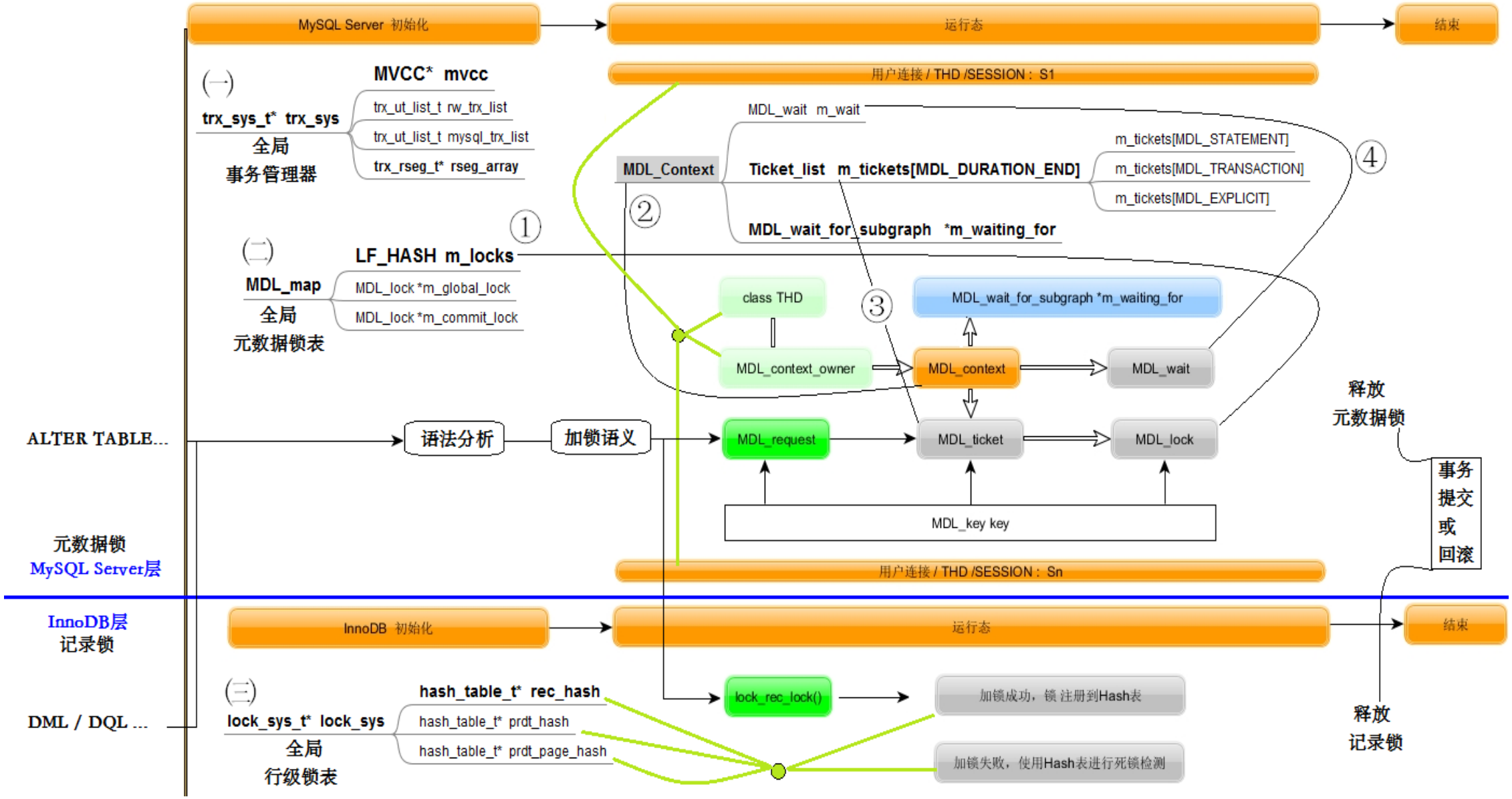
表 11-6 记录锁事务锁相容表

		Granted Mode，已经授予的锁				
		AI	IS	S	IX	X
Requested Mode 正申请的锁	AI		Y		Y	
	IS		Y	Y	Y	
	S		Y	Y		
	IX		Y		Y	
	X					

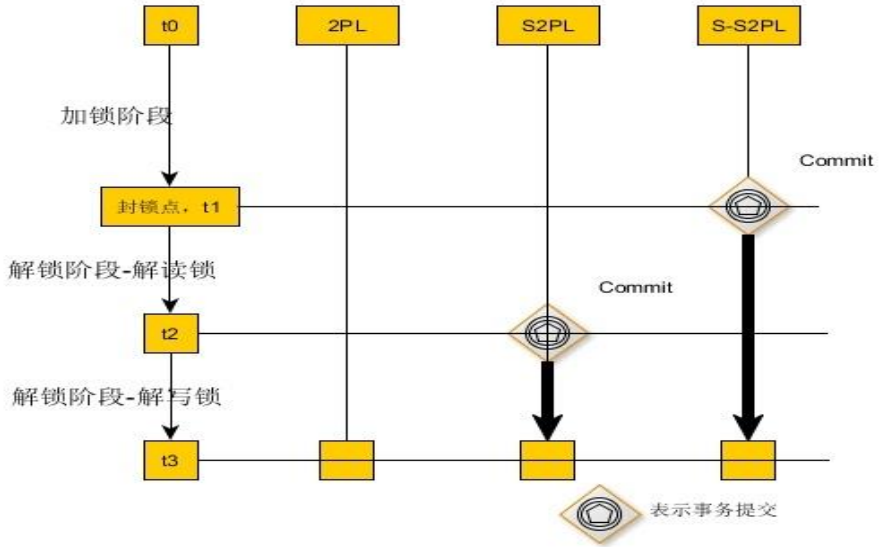


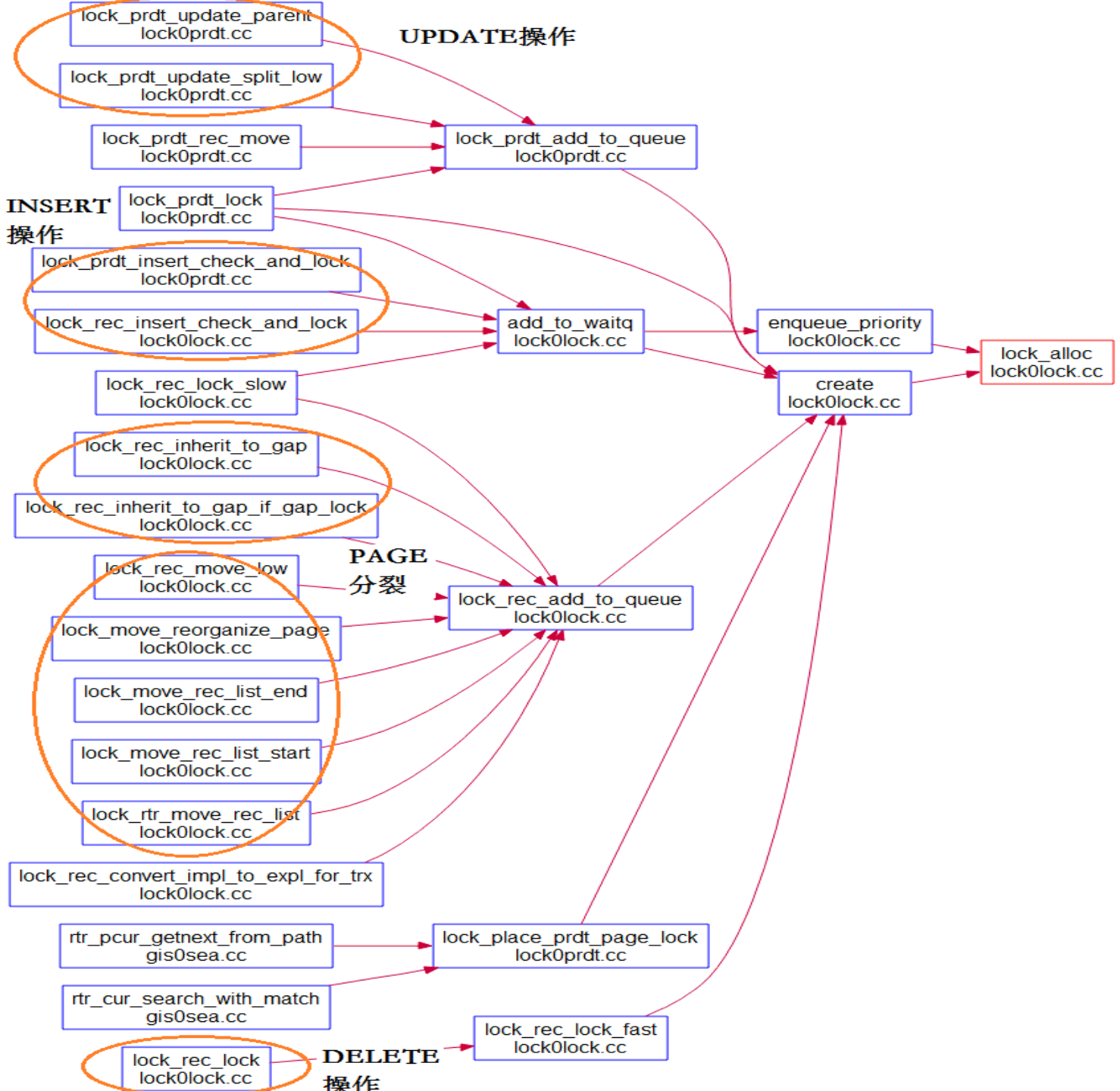
事务处理，有限状态自动机

TDSQL单机事务处理--整体



TDSQL单机事务处理--锁





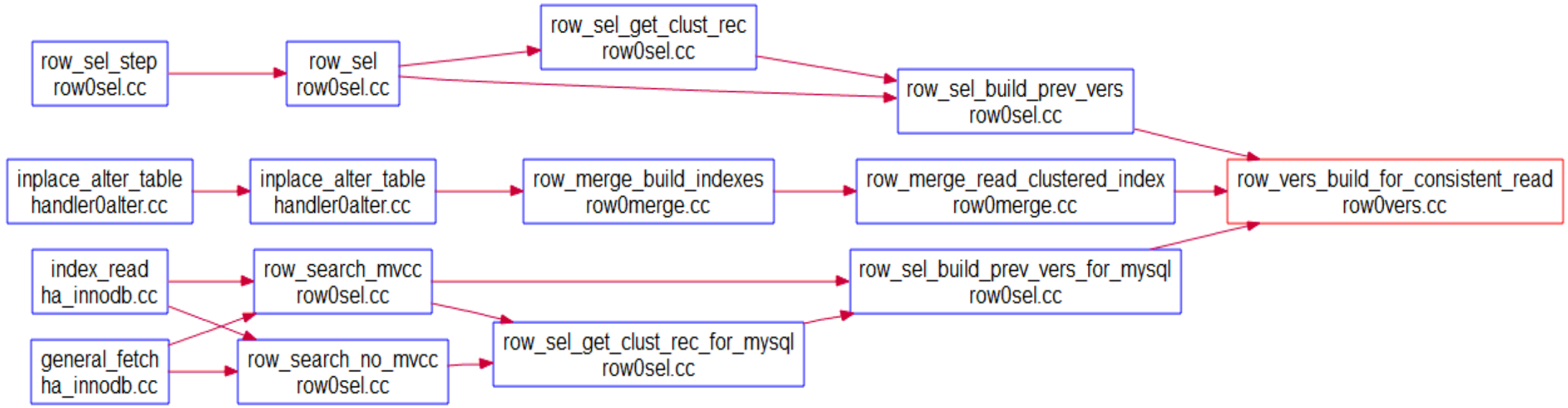
TDSQL单机事务处理--MVCC

多版本生成过程，其方式如下：

- 最老的版本，一定是插入操作暂存到UNDO日志的版本（对于聚集索引，不是元组的所有字段被暂存到回滚段，而是主键信息被暂存）
- 更新操作，把旧值存入UNDO日志。同一个记录反复被更新，则每次更新都存入一次旧值（前像）到UNDO日志内，如此就会有多个版本。版本之间，使用DATA_ROLL_PTR指向更旧的版本。由此所有版本构成一个链表，链头是索引上的记录，链尾是首次插入时生成的UNDO信息。但如果执行过PURGE操作，则链表因被清理过可能链尾不再是首次插入时生成的UNDO信息
- 删除操作，在UNDO日志中保存删除标志（用宏TRX_UNDO_DEL_MARK_REC表示）等信息
- 插入或更新操作，可能的因本地更新的可能（in place），导致trx_undo_page_report_modify函数被多次调用，即插入操作也可能调用此函数（参考row_ins_must_modify_rec()函数）

多版本查找过程，其方式如下：

- 如果是读未提交隔离级别：根本不去找旧版本，在索引上读到的记录就被直接使用，详情参见上节的“读未提交”的实现
- 如果不是读未提交隔离级别：则需要调用row_vers_build_for_consistent_read()等函数进入UNDO回滚段中根据DATA_ROLL_PTR进行查找，边找边根据**changes_visible**()函数判断可见性，如果可见，则返回（请注意，表12-1表明了UNDO信息中记载了DATA_TRX_ID、DATA_ROLL_PTR，所以可以用DATA_TRX_ID作为**changes_visible**()函数的参数判断可见性，用DATA_ROLL_PTR查找每一个历史版本）。



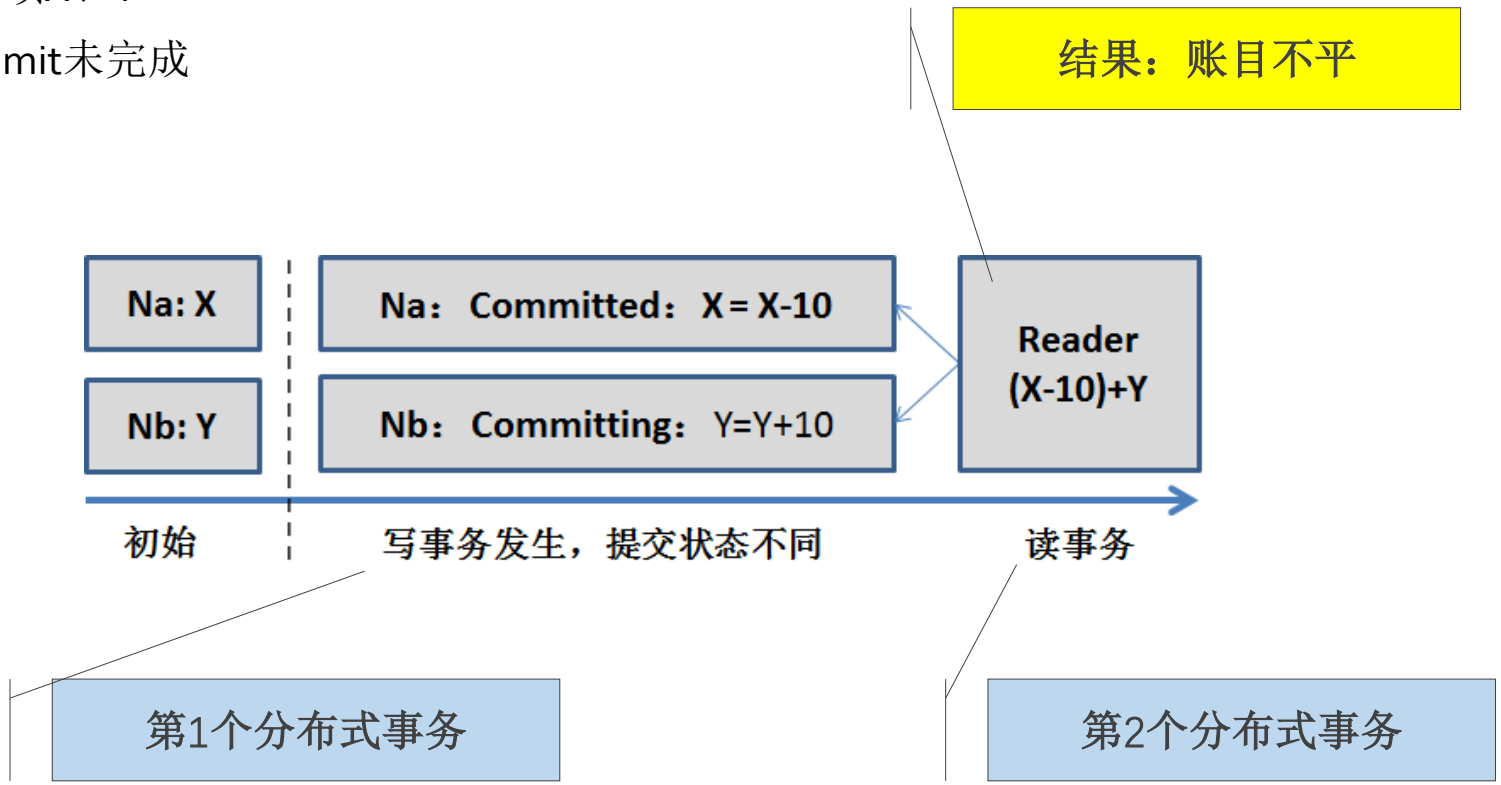
目录

CONTENTS

TDSQL分布式事务--分布式数据异常

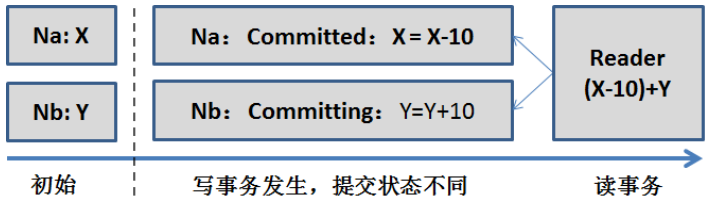
读半已提交数据异常

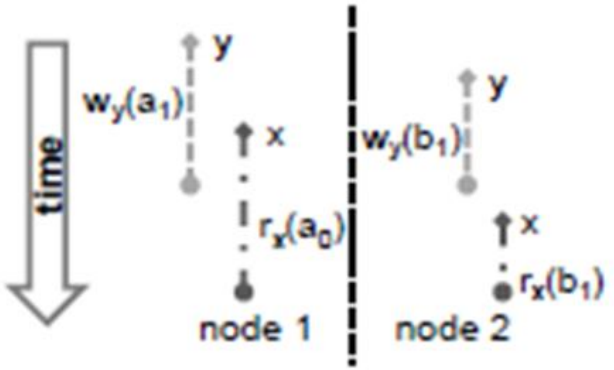
- 两个数据节点Na、Nb；两个数据项X、Y
- Na节点commit完成；Nb节点commit未完成
- 全局该事务处于committing状态



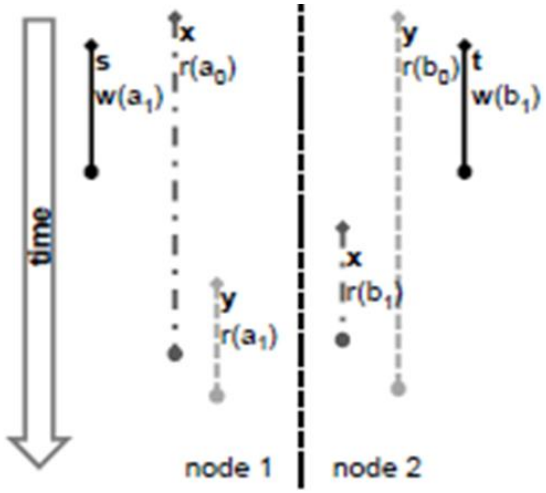
TDSQL分布式事务--分布式数据异常

分布式读半已提交异常





Cross异常

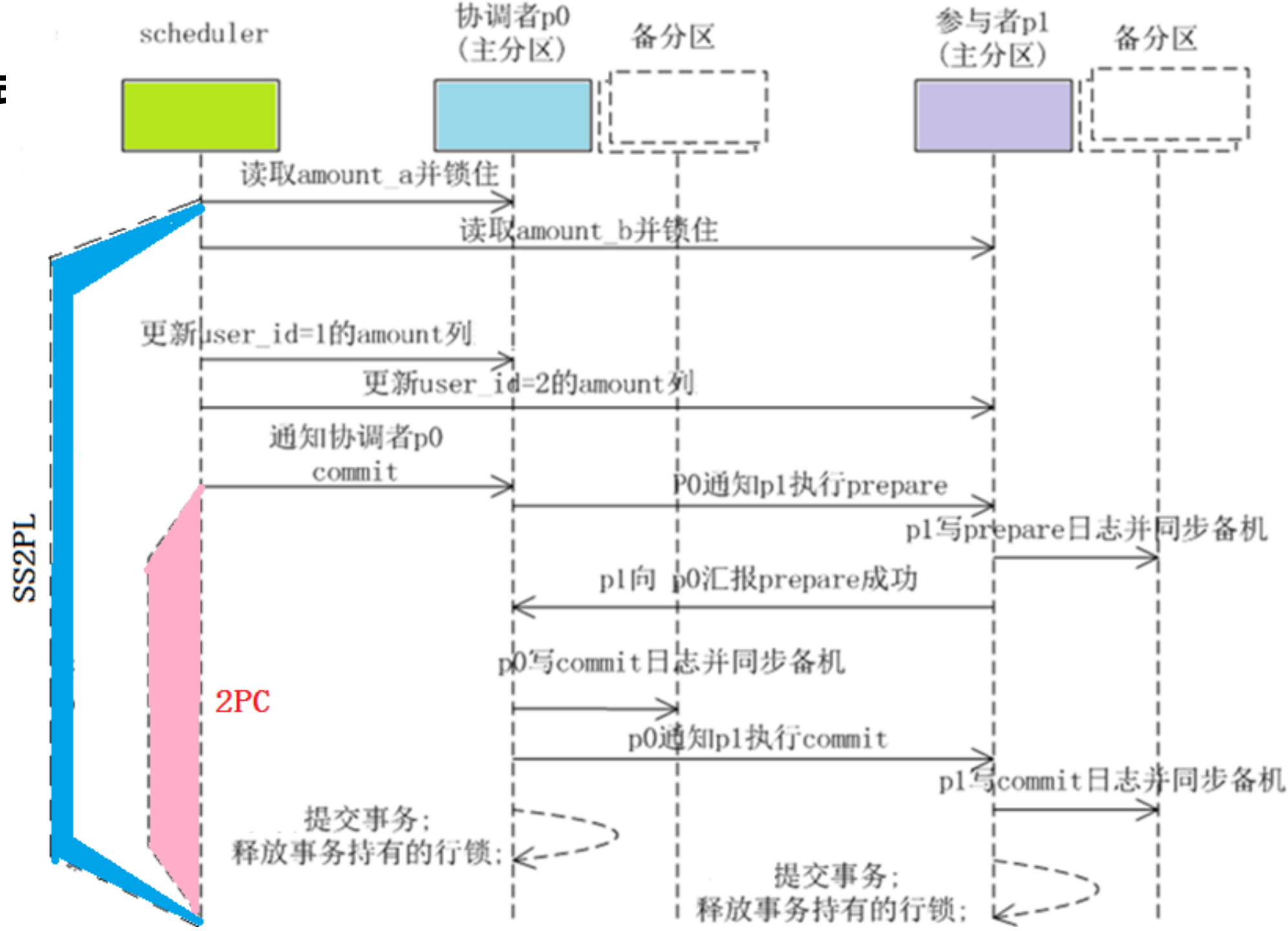


Carsten Binnig, Stefan Hildenbrand, Franz Färber, Donald Kossmann, Juchang Lee, Norman May: Distributed snapshot isolation: global transactions pay globally, local transactions pay locally. VLDB J. 23(6): 987-1011 (2014)

TDSQL分布式

第一代TDSQL
分布式事务处理模型

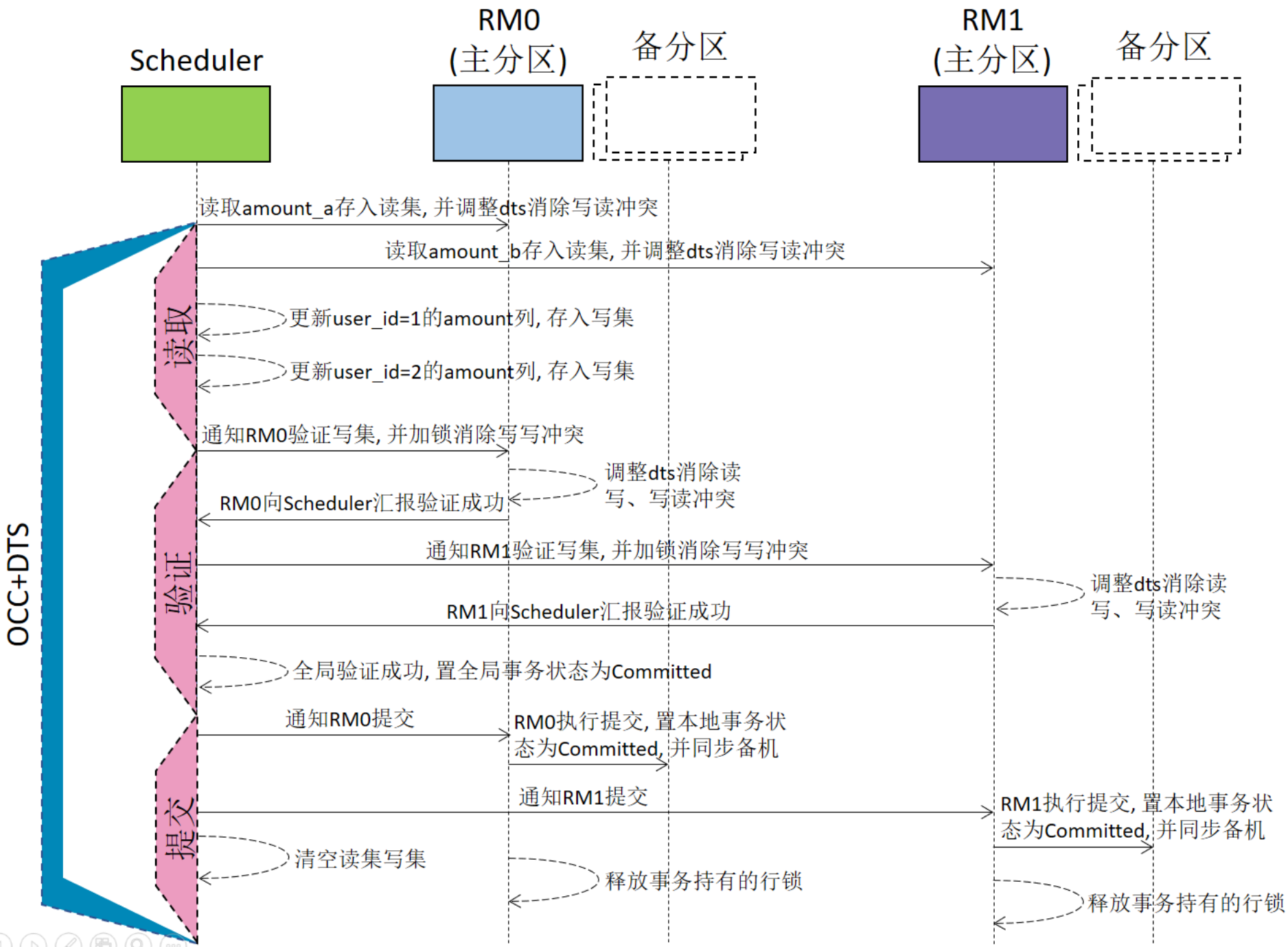
start transaction;
修改user1的金额;
修改user2的金额;
commit;



TDSQL分布式事

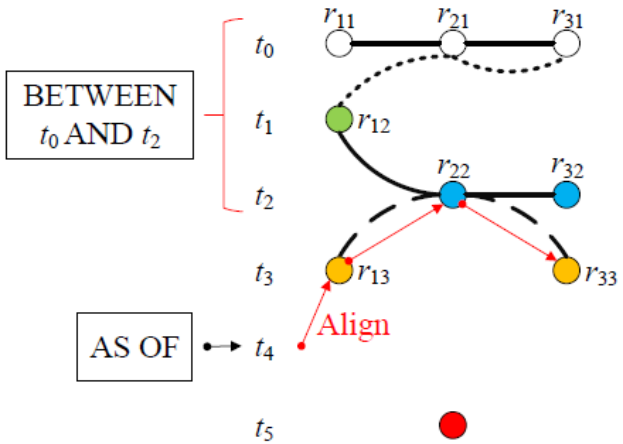
第二代TDSQL
分布式事务处理模型

start transaction;
修改user1的金额;
修改user2的金额;
commit;



TDSQL分布式事务--第二代事务处理技术

基于TDSQL全时态数据库的全局读一致性技术



N1子节点	N2子节点	全局状态	是否可见
Prepared	Prepared	Preparing	不可见，读前一个版本
Prepared	Prepared	Prepared	不可见
Prepared	Prepared	Committed	可见
Committed	Prepared	Committed	可见
Committed	Committed	Committed	可见

核心问题：

- 分布式、全态数据在任何时间点的数据一致性

解决技术：

- 写写冲突封锁机制互斥
- MVCC从新版本到旧版本
- 局部节点处于Prepared状态
- 全局事务Committed/ Prepared状态
- 异步、批量设置本地事务状态
- 全局逻辑时钟（非跨城/洲分布）
- 冲突可串行化

VLDB 2019 腾讯全时态论文《A Lightweight and Efficient Temporal Database Management System in TDSQL》

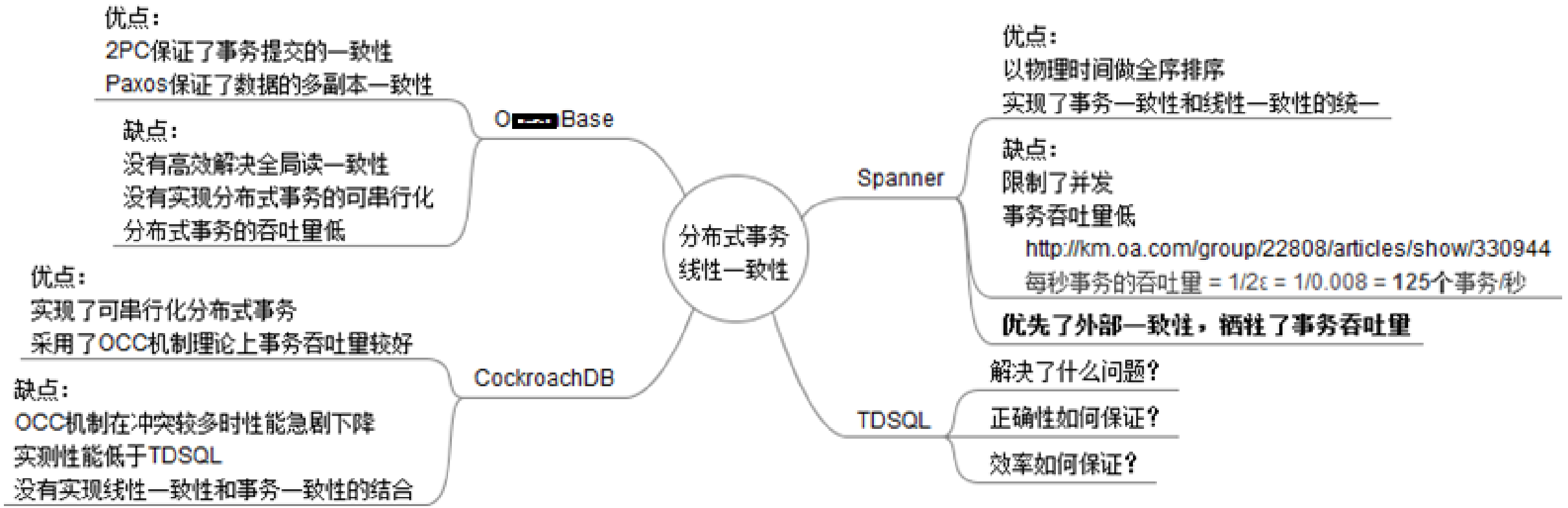
目录

CONTENTS

分布式数据库事务技术

	XxDB	CockroachDB	Spanner	XxxxxBase
事务--ACID	支持	支持	支持	支持
并发控制	乐观/提交时检测冲突	乐观/MVCC	SS2PL/MVCC	SS2PL/MVCC
MVCC--多版本识别/全局唯一特性	事务ID	混合时间戳	物理时间戳 TrueTime	局部
MVCC-隔离特性	snapshot	write-snapshot	snapshot	snapshot
读写事务	乐观机制	乐观/MVCC	2PL	2PL
分布式事务提交/原子性	2PC	可避免2PC（事务状态记录）	2PC	2PC
外部一致性的读写	支持	支持	支持	存在不一致（因果序）
全局一致性的快照读	SI级别	SSI级别		SI级别
只读事务在备机/follower上读	leader上读		支持	leader上读/备机弱一致性读
死锁	无死锁	无死锁	伤停等待（wound-wait）避免死锁	超时检测
预写日志/WAL	支持	支持	支持	支持
隔离级别	SI	SI/SSI	SI	RC

分布式数据库事务技术


分布式事务
线性一致性
OceanBase
优点：
2PC保证了事务提交的一致性
Paxos保证了数据的多副本一致性
缺点：
没有高效解决全局读一致性
没有实现分布式事务的可串行化
分布式事务的吞吐量低
CockroachDB
优点：
实现了可串行化分布式事务
采用了OCC机制理论上事务吞吐量较好
缺点：
OCC机制在冲突较多时性能急剧下降
实测性能低于TDSQL
没有实现线性一致性和事务一致性的结合
Spanner
优点：
以物理时间做全序排序
实现了事务一致性和线性一致性的统一
缺点：
限制了并发
事务吞吐量低
http://km.oa.com/group/22808/articles/show/330944
每秒事务的吞吐量 = 1/2ε = 1/0.008 = 125个事务/秒
优先了外部一致性，牺牲了事务吞吐量
TDSQL
解决了什么问题？
正确性如何保证？
效率如何保证？