DTCC

2015中国数据库技术大会

DATABASE TECHNOLOGY CONFERENCE CHINA 2015

大数据技术探索和价值发现

TDSQL架构分享

腾讯 - 计费平台部

harlylei

个人简介

- **harlylei(雷海林)**
- 腾讯 / TEG / 计费平台部
- 2007年加入公司，10年以上的Linux后台Server开发经验，目前主要从事分布式Cache，实时大数据处理引擎，分布式MySQL(TDSQL)设计和开发工作。

业务场景

米大师
数据层解决方案

联机交易
数据层解决方案

金融云
敬请期待...

目录

我们需要什么样的MySQL

百亿级的账户，订单数据
百亿级的日交易流水
十万级别每秒并发
毫秒级交易响应
——***易伸缩，高并发***

一分不差的银行级业务
——***高一致性的容灾***

7 * 24 小时的不间断服务
——***自动容灾，自动扩容***

如果：

MySQL性能足够强大
MySQL一致性切换足够完善
MySQL不需要关心分库分表
MySQL不需要关心容量不足

那么：

代码会比现在简单
运维会比现在简单

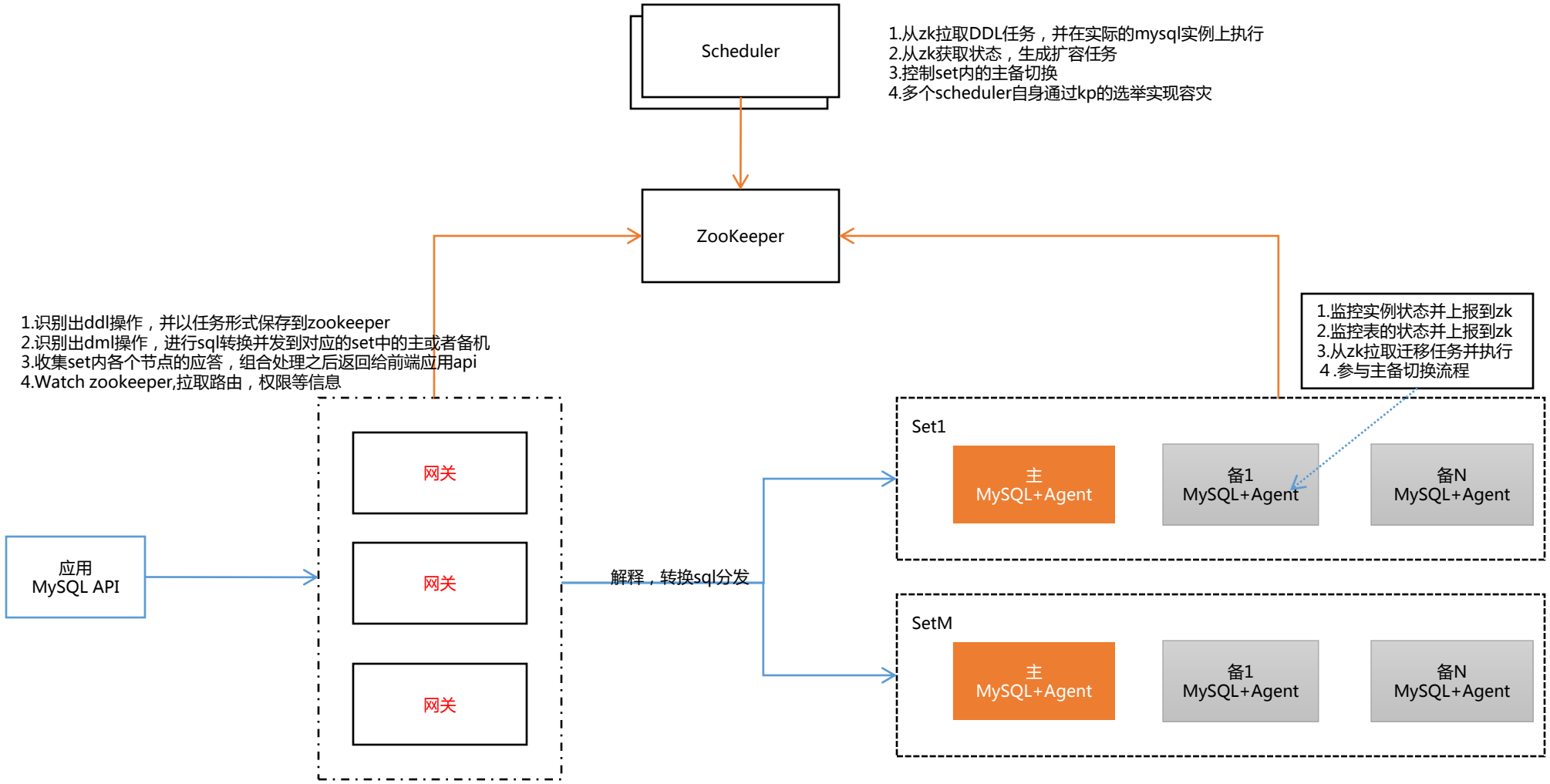
而简单意味着

——***健壮***

将MySQL打造成存储集群

- 继续通过MySQL API和sql接口访问集群
- 节点异常自动切换，切换过程保证数据零丢失，管好钱袋子
- 按需自动扩容/缩容，以支撑业务爆发式增长，扩容过程对业务基本上无感知
- 业务之间支持隔离，集群自身具备流控机制
- 对SQL语句做实时的时耗统计，慢查询分析，异常SQL拒绝等

系统结构


Scheduler
1.从zk拉取DDL任务，并在实际的mysql实例上执行
2.从zk获取状态，生成扩容任务
3.控制set内的主备切换
4.多个scheduler自身通过kp的选举实现容灾
ZooKeeper
1.识别出ddl操作，并以任务形式保存到zookeeper
2.识别出dml操作，进行sql转换并发到对应的set中的主或者备机
3.收集set内各个节点的应答，组合处理之后返回给前端应用api
4.Watch zookeeper,拉取路由，权限等信息
1.监控实例状态并上报到zk
2.监控表的状态并上报到zk
3.从zk拉取迁移任务并执行
4.参与主备切换流程
网关
网关
网关
应用
MySQL API
解释，转换sql分发
Set1
主
MySQL+Agent
备1
MySQL+Agent
备N
MySQL+Agent
SetM
主
MySQL+Agent
备1
MySQL+Agent
备N
MySQL+Agent

容量按需自动伸缩

- 规则(水平扩容还是垂直扩容)
  - 标的：**Table**
  - 最小粒度：**SET**
  - 即一个伸缩任务应该是：**将某个Table的容量伸缩n个SET**

扩容决策

谁要扩？

- 如何发现源头？

扩到哪？

- 如何选择目标？

怎么扩？

- 迁移谁？
- 是否分裂表？分裂多少？

监控是基础，**调度**是核心

容量判断

资源级

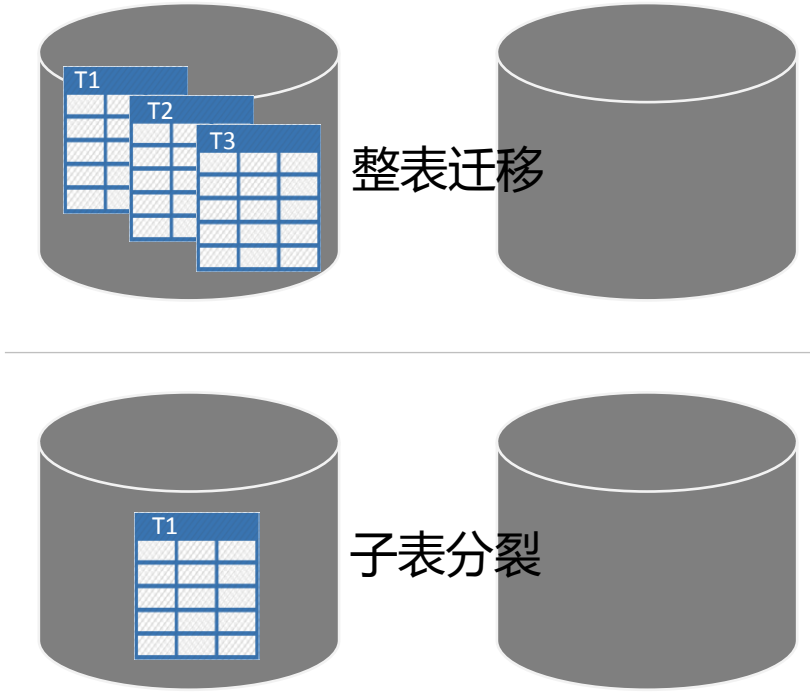
- CPU
- 内存
- 磁盘空间
- 磁盘IO
- 网络IO

业务级

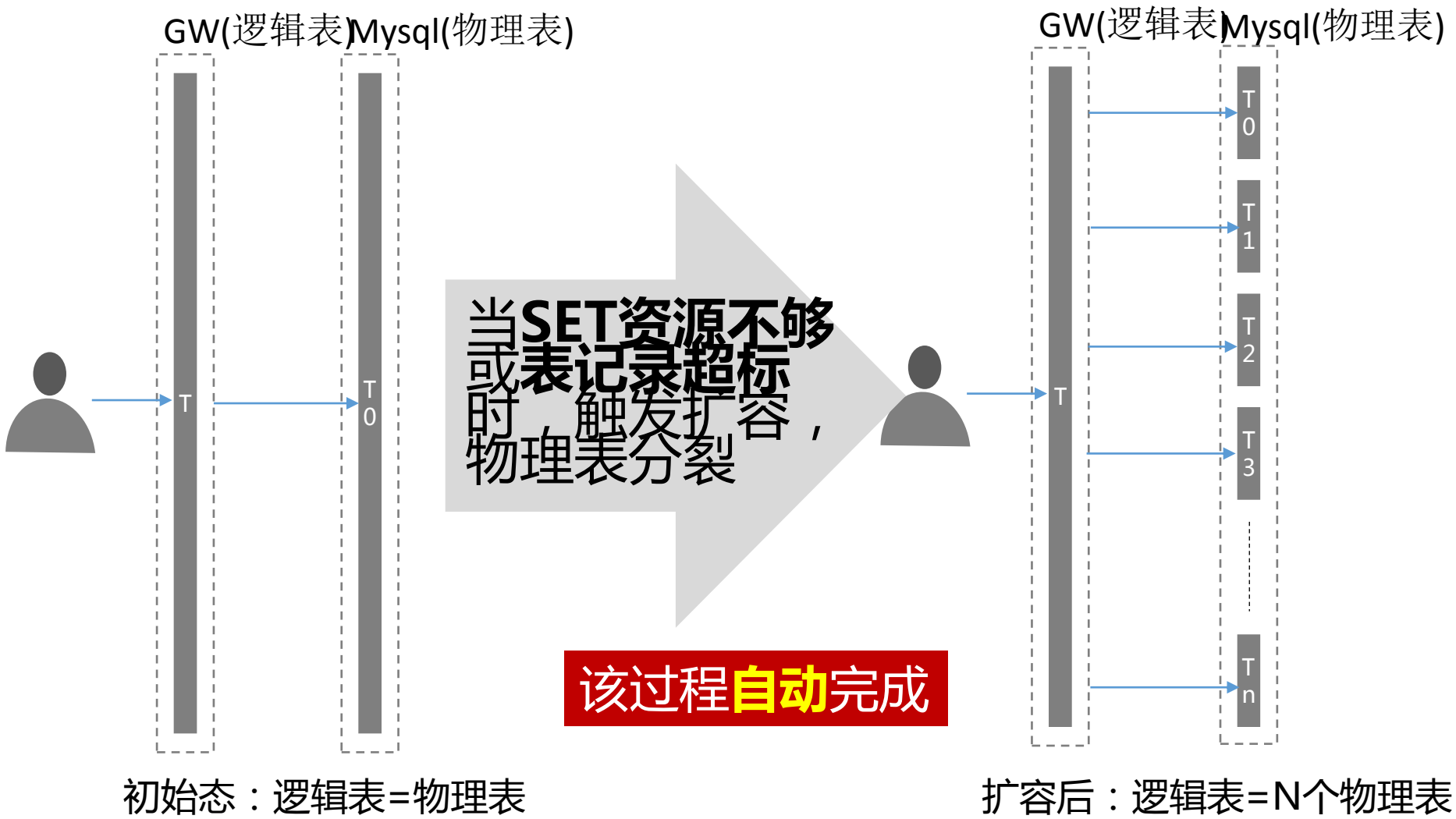
- 时延
- 请求量
- 记录数

伸缩方式

- 伸缩方式
  - 整表迁移
  - 子表分裂
- 原则：**避免表分裂，及时表合并**
- 表分裂的问题
  - 在一个集群中，每次表分裂，会导致集群表数量的增加；集群中表的数量就是路由的条数，表数量越多，路由的效率就会越低
  - 一个实例上面的表越多，对该实例运行环境的判断就越复杂：同一实例上的子表，表现各异，交叉影响的评估难度增大，可能导致连锁反应
- 扩容：**整表迁移** > **子表分裂**
- 缩容：**子表合并** > **整表搬迁**

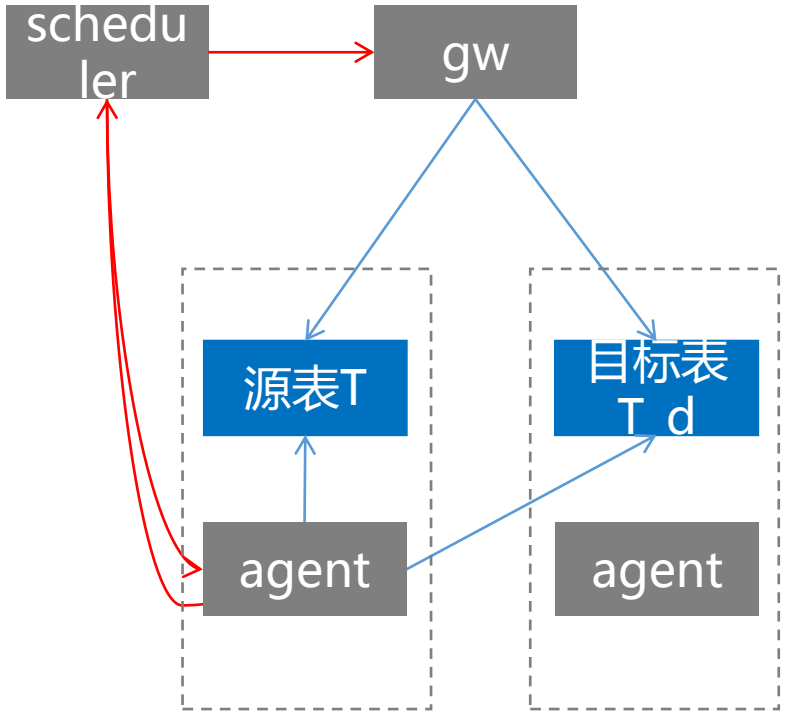


自动分表


GW(逻辑表)
Mysql(物理表)
T
T0
当SET资源不够或表记录超标时，触发扩容，物理表分裂
该过程自动完成
GW(逻辑表)
Mysql(物理表)
T
T0
T1
T2
T3
Tn
初始态：逻辑表=物理表
扩容后：逻辑表=N个物理表

数据搬迁

- 搬迁策略
  - **先切后搬**
  - **先搬后切**
- 搬迁过程
  1. 镜像同步
     - 源agent：记录日志点，导出T中新号段镜像
     - 源agent：向T_d批量插入镜像数据
  2. 追日志
     - 源agent：向T_d追日志
  3. 日志追平
     - 源agent：日志相差<n时，修改T表名为T_s
     - 源agent：追平日志
  4. 切换新路由
     - scheduler ：修改新号段路由至T_d
  5. 完成
- 搬迁过程中的故障处理(快速失败)
  - 源主机宕机，容量伸缩终止，现网不受影响
  - 目标主机宕机，容量伸缩终止，现网不受影响



在线DDL支持

- 传统的DDL操作A->B
  - A表加读锁(影响写)
  - 用A的建表语句创建B，并修改B结构
  - 拷贝A数据到B，锁定B，删除A
  - B rename成A
  - 刷新数据字典并释放锁

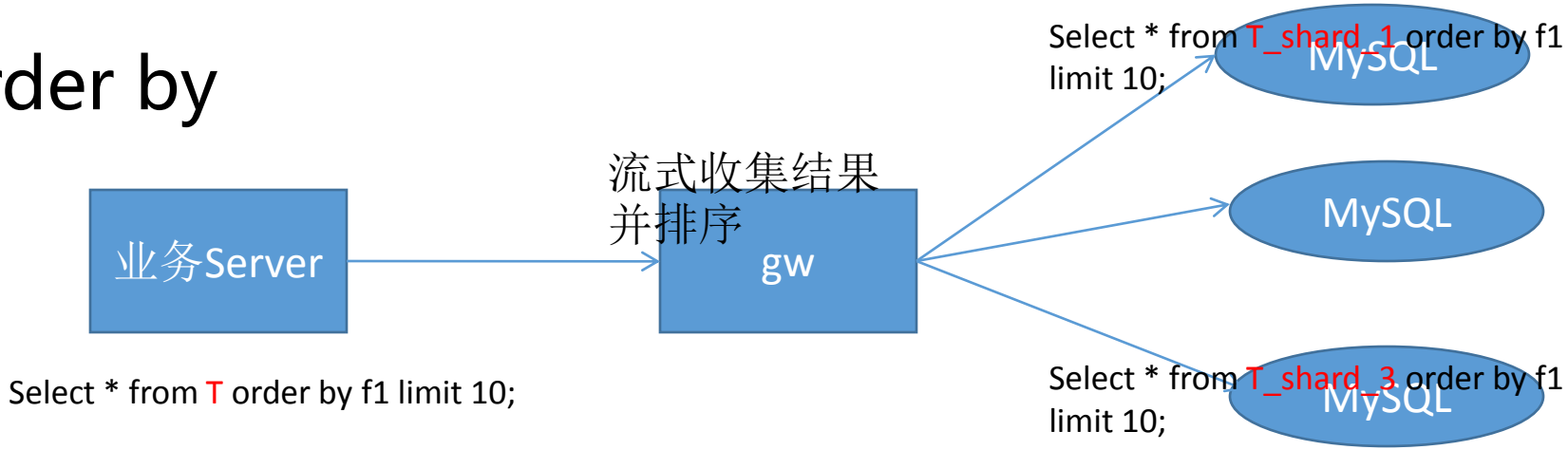
- 新方式
  - 直接采用在线迁移的方式完成

    a)可以立即返回到业务，实际的迁移操作异步完成

    b)整个过程基本上不锁表

聚合类SQL支持

- group by
- Max,sum,min,ave等聚合函数
- Distinct,count(1)
- Order by



主备容灾 - 需求

- 自动切换
- 自动恢复
- 主备一致性
- 跨IDC容灾

容灾决策

谁要切换？

- 如何发现故障？
- 如何确定是否需要切换？

怎么切换？

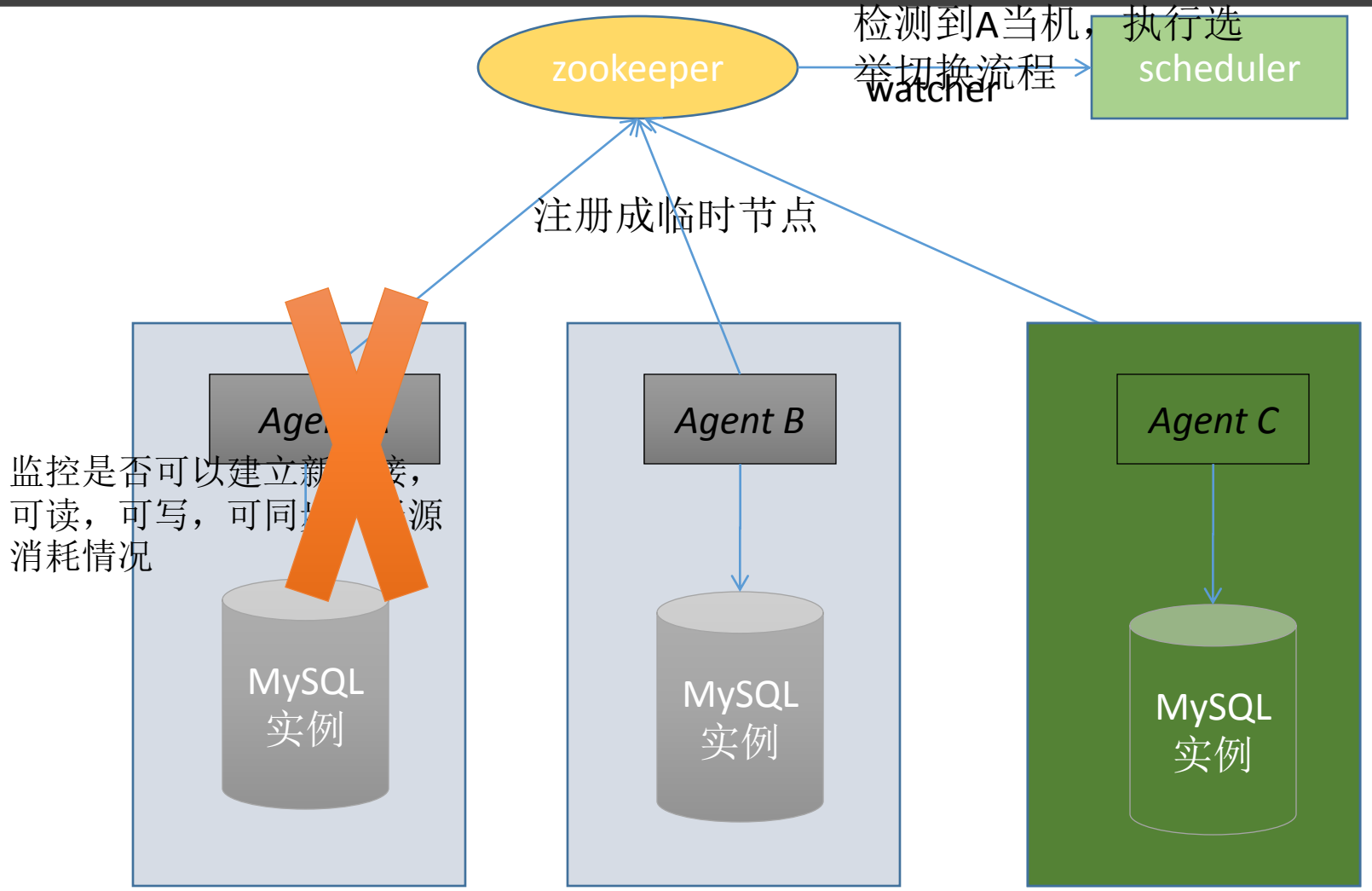
- 如何保障数据一致性？
- 如何切请求？

如何恢复？

- 如何重建SET？

监控是基础，**一致性**是核心

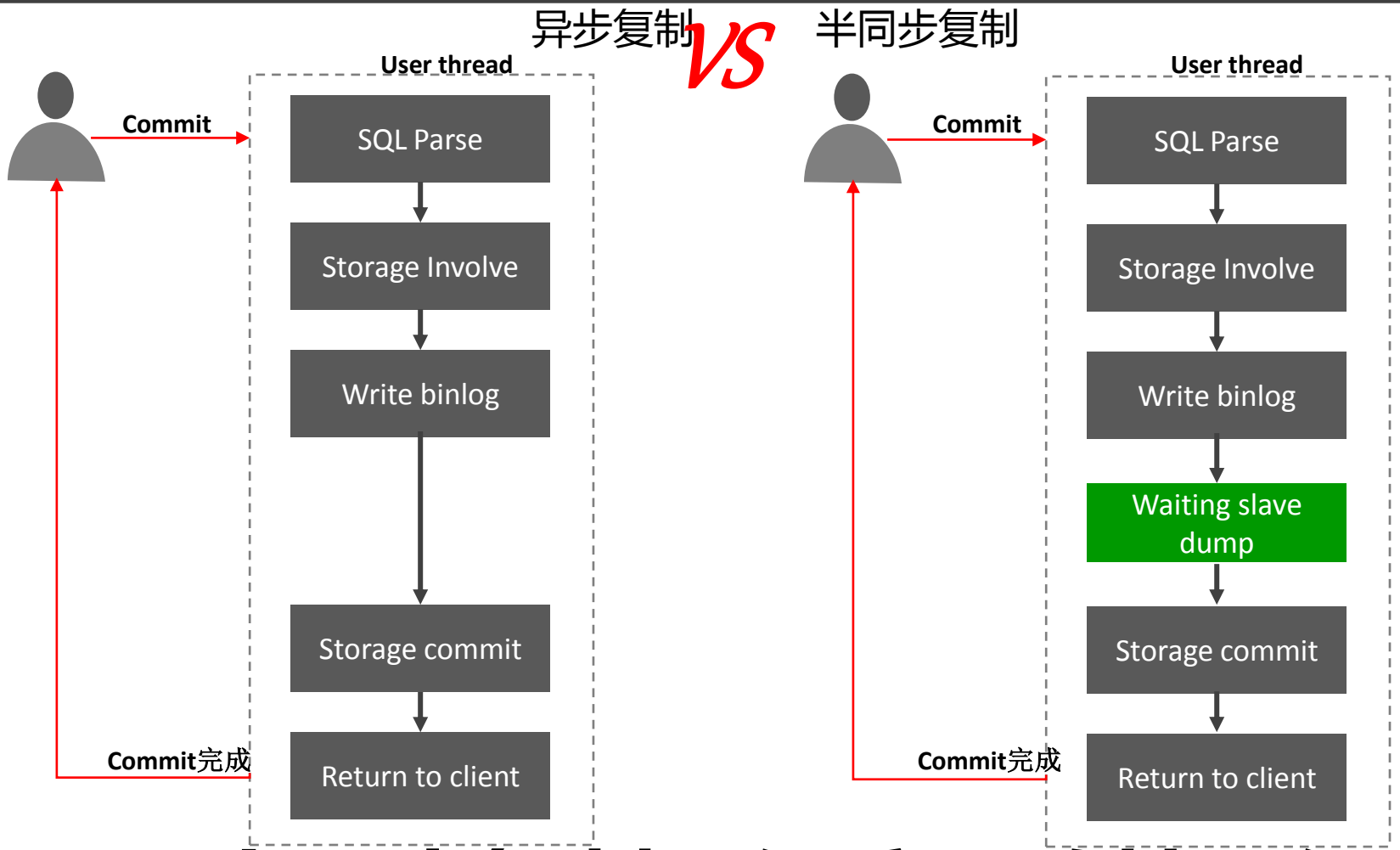
节点存活监控


zookeeper
检测到A当机，执行选举切换流程
watcher
scheduler
注册成临时节点
Agent B
Agent C
MySQL
实例
MySQL
实例
MySQL
实例

高一致性切换-MySQL异步复制分析

高一致性切换-MySQL 5 .5半同步复制分析

数据复制



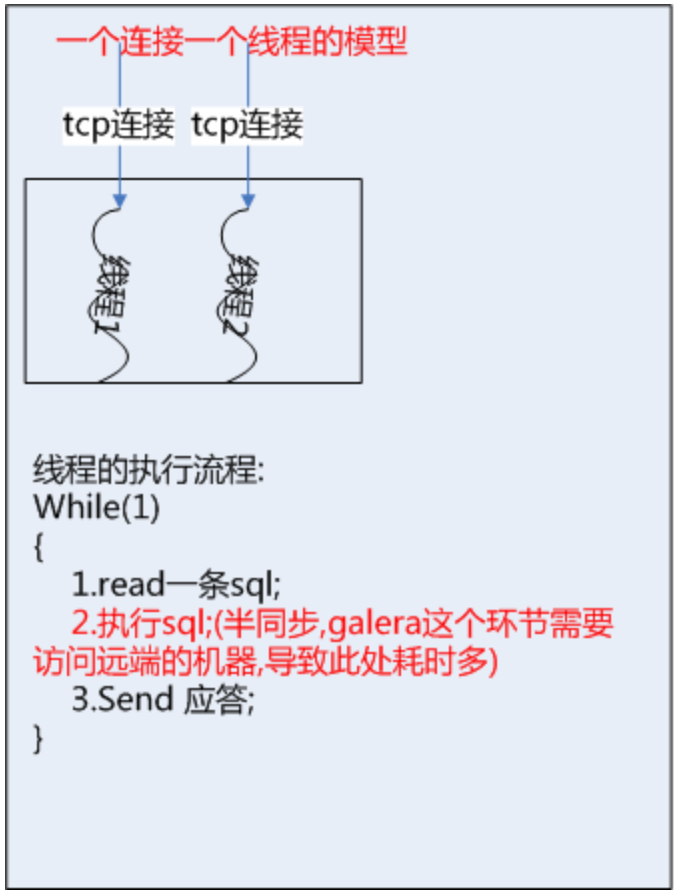
半同步复制可保障一致性，但是...

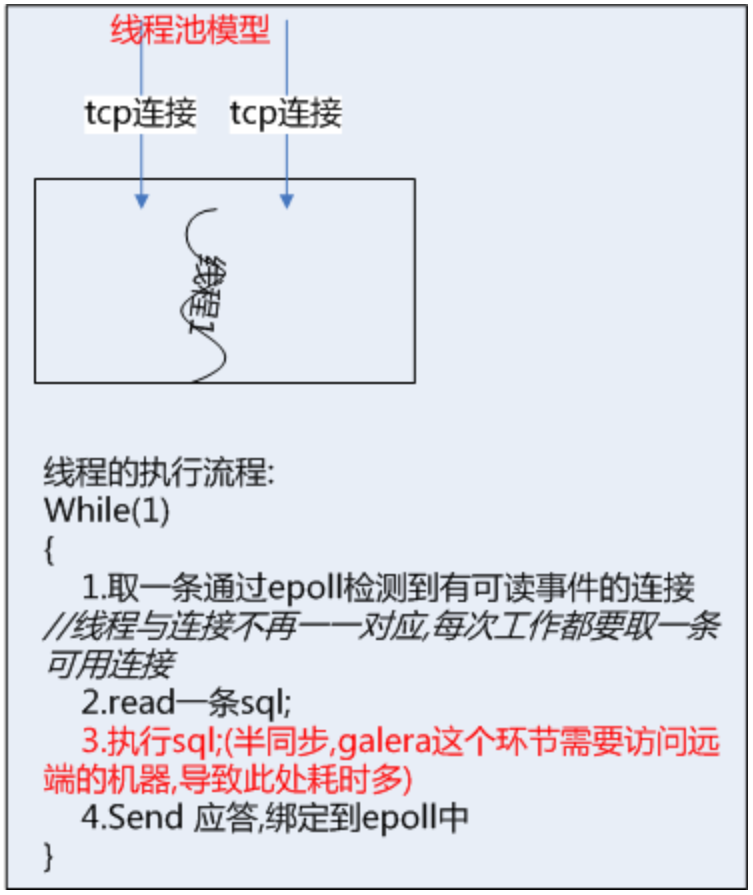
半同步复制的问题

- 1.超时后蜕化成异步，金融场景不合适
- 2.跨IDC的情况下性能不乐观

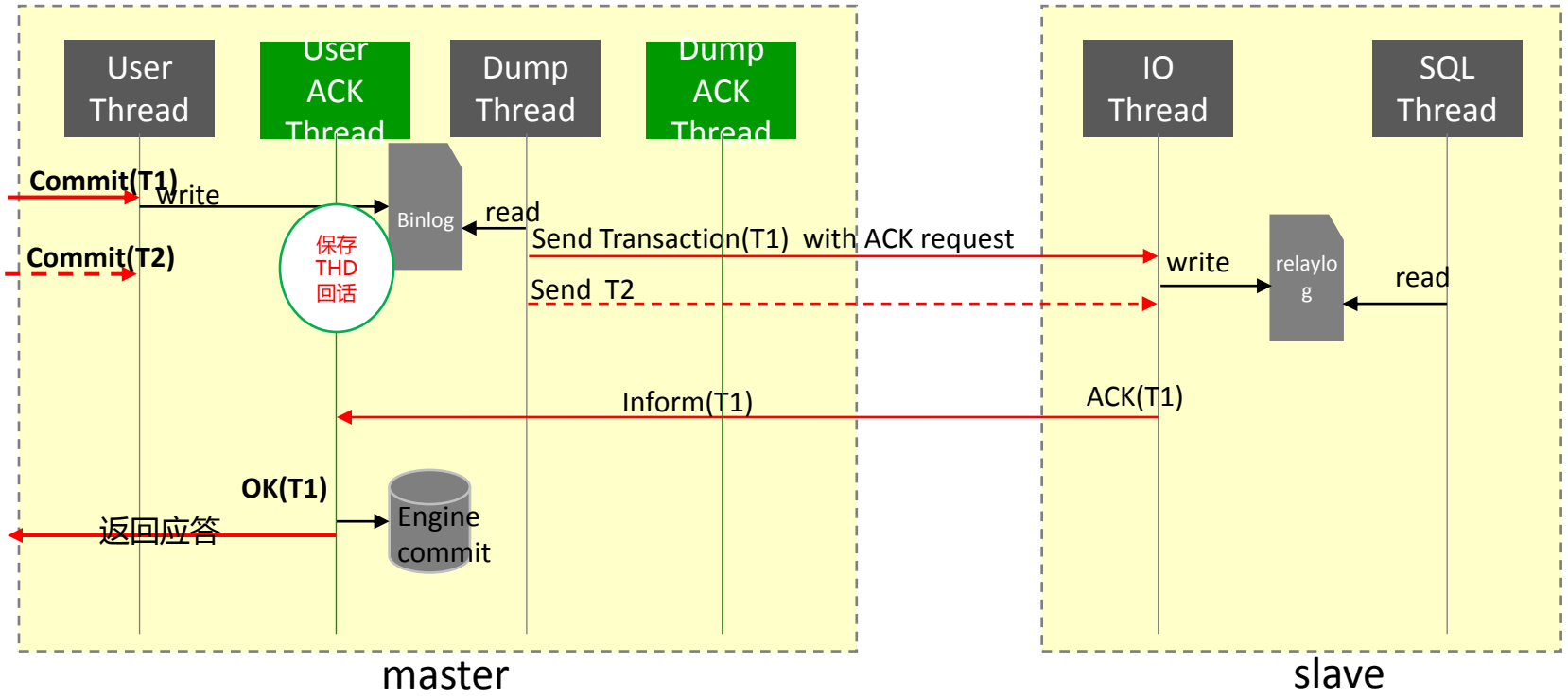
主备复制方案（跨IDC）	TPS	时耗(ms)
异步	20,000	<10
半同步	2,200	4~600
网易innosql（半同步）	4,500	4~500
MariaDB Galera Cluster	6,000	4~10000

半同步性能不好的原因分析





用户线程异步化



主备复制方案（跨IDC）	TPS	时耗(ms)
异步	20,000	<10
半同步	2,200	4~600ms
异步化改造后的半同步	9,500	99.9%的<30ms，少量毛刺，最大达到600
网易innosql（半同	4,500	4~500ms

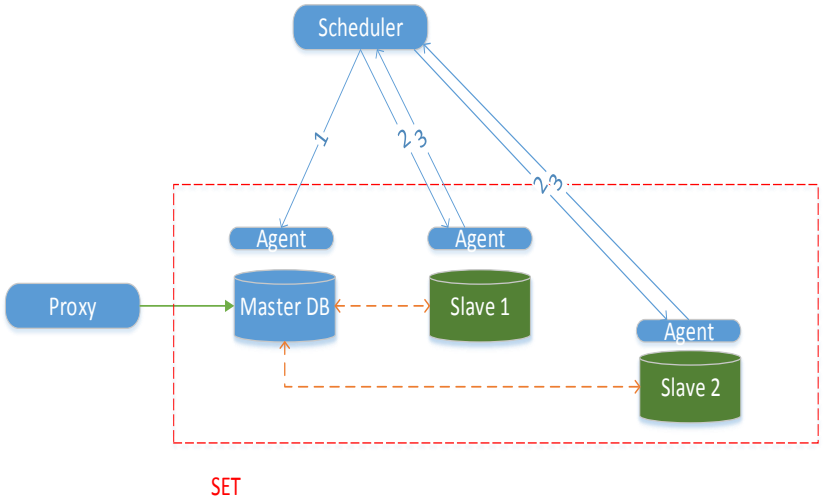
高一致性容灾 — 如何保证没有脏数据

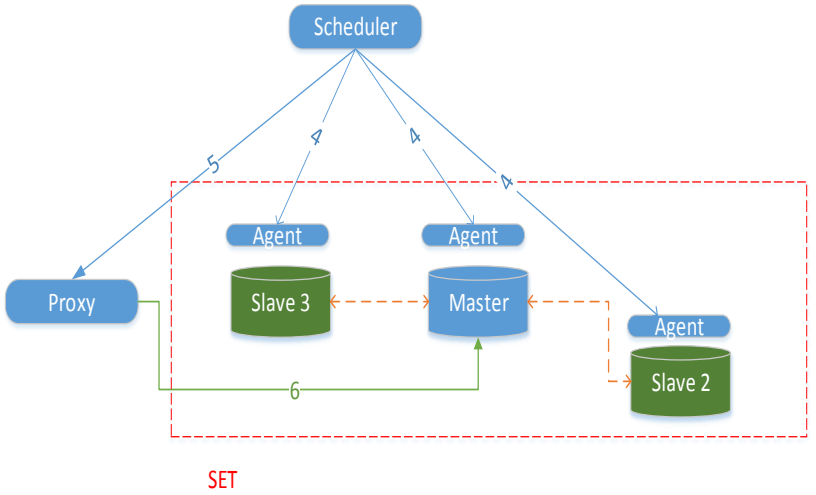
原则：

1、主机可读可写，备机只读，备机可以开放给业务查询使用

2、任何时刻同一个SET不能有两个主机

3， 宁愿拒绝服务，不提供错误的服务，追求CAP中的C，必要的时候牺牲部分A





1、主DB降级为备机（杀死当前所有session,设置只读，如因为当机原因没有收到下次重新启动也会执行这个流程），同时会给网关下发暂时没有主节点的路由

2、参与选举的备机停止io线程之后上报最新的binlog点

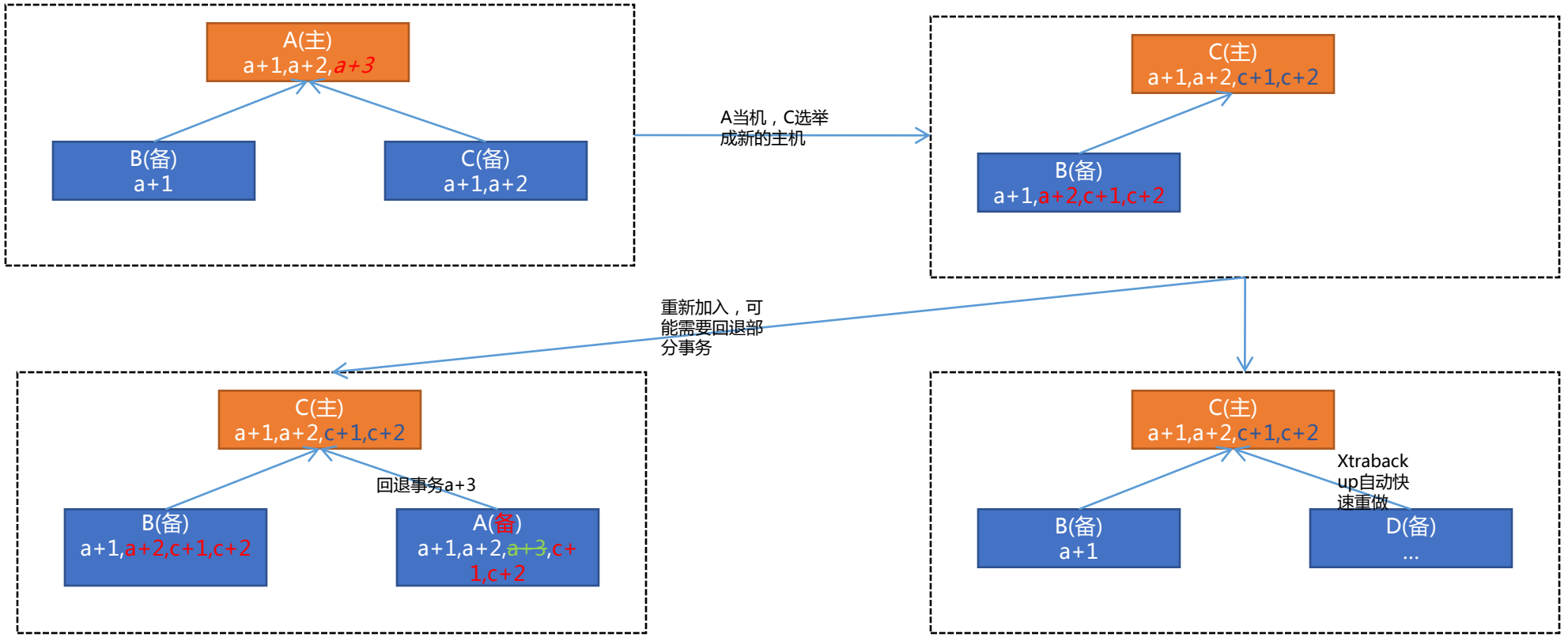
3、scheduler收到binlog点之后，选择出binlog最大的节点(可能同时有 2 个)并要求对应的机器加载完relay log。当收到加载完relay log信息之后，则选择这个应答的节点为主机；

4、重建主备关系

5、修改路由

6、请求发给新的主机

数据高可用性的保障机制（恢复）



应用如何适应TDSQL

- TDSQL当前定位是支撑OLTP类型短事务的业务，不支持join操作
- 弱化了单机MySQL的功能，如存储过程，触发器，视图，自增序列号，Session变量等
- 业务可以将它看成一个加强版的NoSQL系统，优势是存储的是结构化数据，支持SQL操作 ，支持多个索引，数据高一致性访问，持久化非常强，数据自动扩容等
- 所有的表能通过某个shard字段(如QQ号，微信号等)进行数据水平拆分，高频的SQL操作都能带上这个shard字段

下一步的规划

- 按组扩容

  拥有同样shard字段的所有表采用同样的路由规则，以支持同一个shard下所有的sql操作，如join,事务等

- 集群虚拟化

  引入docker来更灵活地管理set和节点，加强资源隔离

运营数据展示

数据库	表(分区)	记录总数	分裂详情	最新采集时间
caccts	t_acct_water_p201502	15,219,743,248	分裂详情	2015-03-13 16:10:58
	t_acct_water_p201503	6,510,050,700	分裂详情	2015-03-13 16:10:58
	t_acct_water_p201504	12	分裂详情	2015-03-13 16:10:58
	t_acct_water_p201505	12	分裂详情	2015-03-13 16:10:58

分裂情况

数据库名	表(分区)	分裂	记录数	采集时间
caccts	t_acct_water_p201502	caccts.t_acct_water_p201502_shard_9	1,264,871,617	2015-03-13 16:10:58
		caccts.t_acct_water_p201502_shard_8	1,266,229,100	2015-03-13 16:10:58
		caccts.t_acct_water_p201502_shard_7	1,271,304,546	2015-03-13 16:10:58
		caccts.t_acct_water_p201502_shard_6	1,266,786,217	2015-03-13 16:10:58
		caccts.t_acct_water_p201502_shard_5	1,267,456,623	2015-03-13 16:10:58
		caccts.t_acct_water_p201502_shard_4	1,265,627,867	2015-03-13 16:10:58
		caccts.t_acct_water_p201502_shard_3	1,265,010,334	2015-03-13 16:10:58
		caccts.t_acct_water_p201502_shard_2	1,272,322,485	2015-03-13 16:10:58
		caccts.t_acct_water_p201502_shard_11	1,272,682,288	2015-03-13 16:10:58

Q&A


ChinaUnix
ITPUB
THANKS

热烈欢迎各位大牛加入

微信：harlylei

Email: harly@vip.qq.com