Gdevops

全球敏捷运维峰会

阿里分布式数据库服务原理与实践

演讲人：沈洵

自我介绍

- 花名：沈洵
- 新浪微博: 淘宝沈询_WhisperXD
- 阿里分布式数据库DRDS,TDDL负责人
- 参与过阿里集团内大部分的Oracle到MySQL的迁移工作
- 在分布式存储领域经验比较丰富

Agenda

01 DRDS简介

02 DRDS功能特性

03 DRDS原理剖析

04 DRDS实战

DRDS起源与发展

起源

起源：DRDS 脱胎于 alibaba的cobar 分布式数据库引擎

06年上线使用

在alibaba有近百应用在使用目前已经开源

DRDS的40%的代码出自cobar

Server 协议层、Sql解析器

起源

起源：DRDS吸收了Taobao TDDL分布式数据库引擎的大量优秀经验和解决方案

1. 08年上线使用，去IOE的利器
2. 目前在使用的应用近千个
3. 大量实际应用解决方案支持
   - —分布式join
   - —分布式aggregation (group sum max min)
   - —异步索引构建
   - —Auto sharding ,自动扩容缩容

从IOE到TDDL

需求

单个机器早就无法满足应用需要，业务已经拆分了
业务高速发展，小鸡模式成本太高

担心

MySQL是否稳定？安全？
这么多机器，运维成本上升，运维能否扛得住？

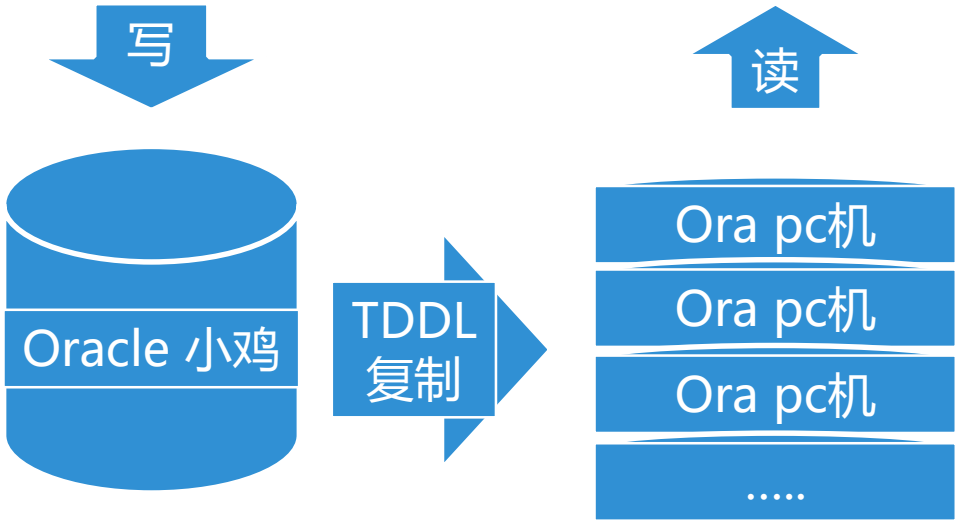
决策

重要应用采用Oracle（写）+mysql（读）架构
—商品、用户、交易、评价、店铺
非重要应用采取MySQL架构
—收藏夹

从IOE到TDDL

转动历史的时刻

- 商品扩展中的失败
- Oracle(写)-> Oracle(读)
  - — Oracle pc机经常挂掉
  - — Oracle 11G压测出现性能问题
  - — 死机后会hung住业务机
- 教训
  - — Oracle不过如此
  - — 架构要简单可依赖



从IOE到TDDL

转动历史的时刻：MySQL(读写)？

新的flash cache方案

写

读

MySQL pc机

MySQL pc机

MySQL pc机

.....

近100个系统的运维经验积累

—TDDL的稳定和可用性很高

—MySQL出现故障的可能性与Oracle一样低

从TDDL到DRDS

场景广泛

互联网应用
企业内大数据应用
政务类应用
物联网应用

产品化，提升SQL兼容性、Join、gui工具的兼容性
像单机数据库一样运维分布式数据库
自动化读写分离

DRDS专门针对外部用户
进行了配置的重新设计

应用场景

1

面对全中国13亿用户，以及全世界50亿的用户

2

单个数据库的最大实例也会出现瓶颈
— 容量瓶颈
— 事务数瓶颈
— 读取瓶颈

应用场景

Scale out（多机水平扩展）：使用廉价数据库阵列来满足用户需求--DRDS

优势

更轻量的使用数据库，未来更换的成本小
一次重构，以后基本再无需担心系统瓶颈

重构迁移需要付出成本
分布式环境下一些查询会被限制不允许执行
完成相同功能需要比单机扩展付出更多成本

劣势

DRDS简介

DRDS功能介绍

01 自动读写分离

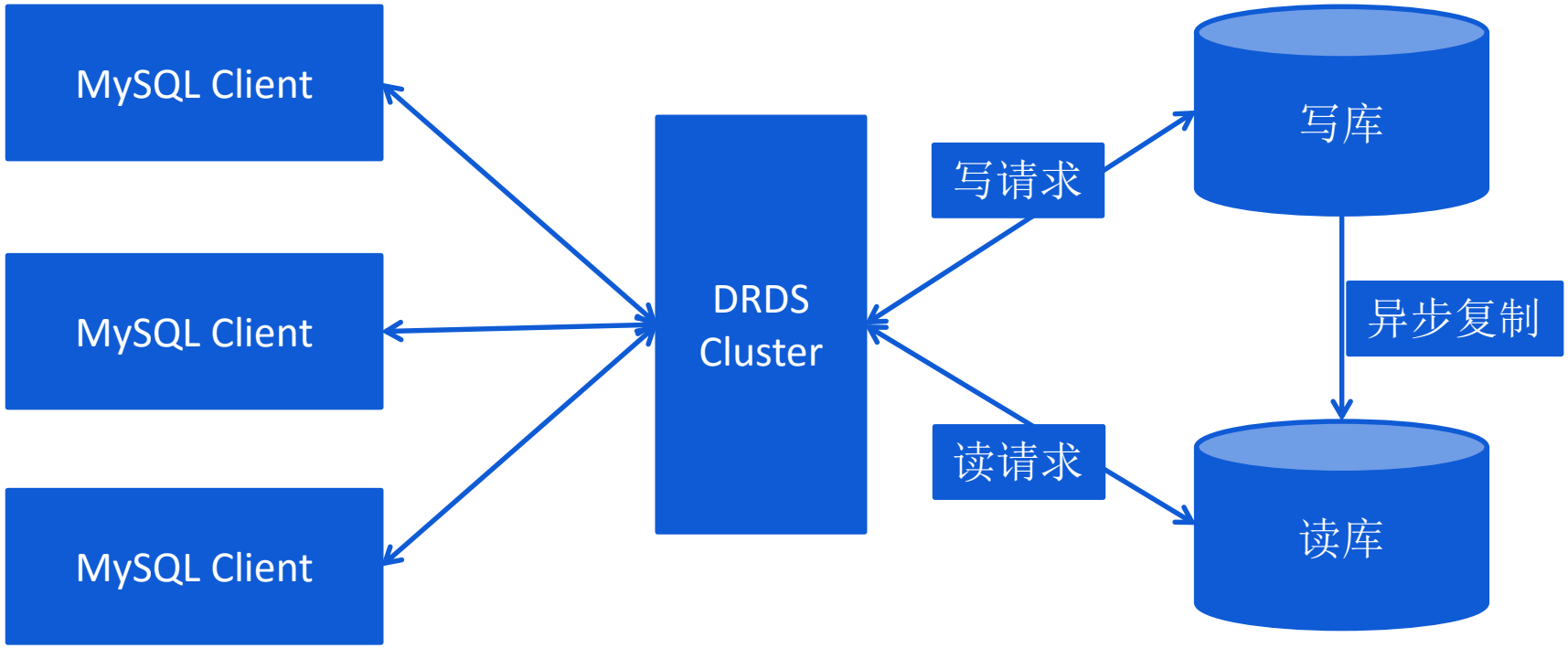
02 分布式MySQL执行引擎

03 弹性扩展

04 小表异步广播

在线动态读写分离

按需“动态”读写分离



应用场景

1

高兼容性：MySQL 5.5 的各类复杂查询

— Join
— 嵌套
— 函数

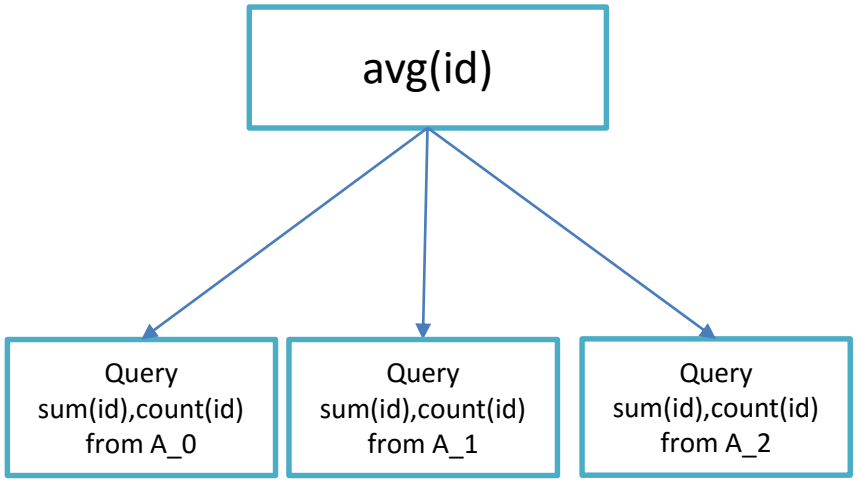
2

智能下推

— 减少网络传输
— 减少计算量
— 充分发挥下层存储的全部能力

执行引擎

- 智能下推
  - 表A 分库分表3个
  - select avg(id) from A



Merge

avg (id)

subQuery

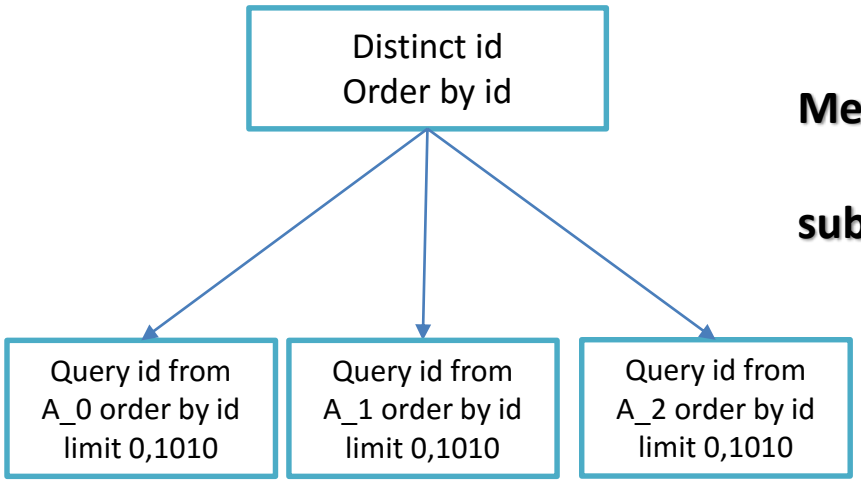
Q1:select count(id),sum(id) A_0

Q2:select count(id),sum(id) A_1

Q3:select count(id),sum(id) A_2

执行引擎

- 智能下推
  - 全表distinct groupby的执行计划
  - Select id from A order by id limit 1000,10



Merge

distinct id , group by id

subQuery

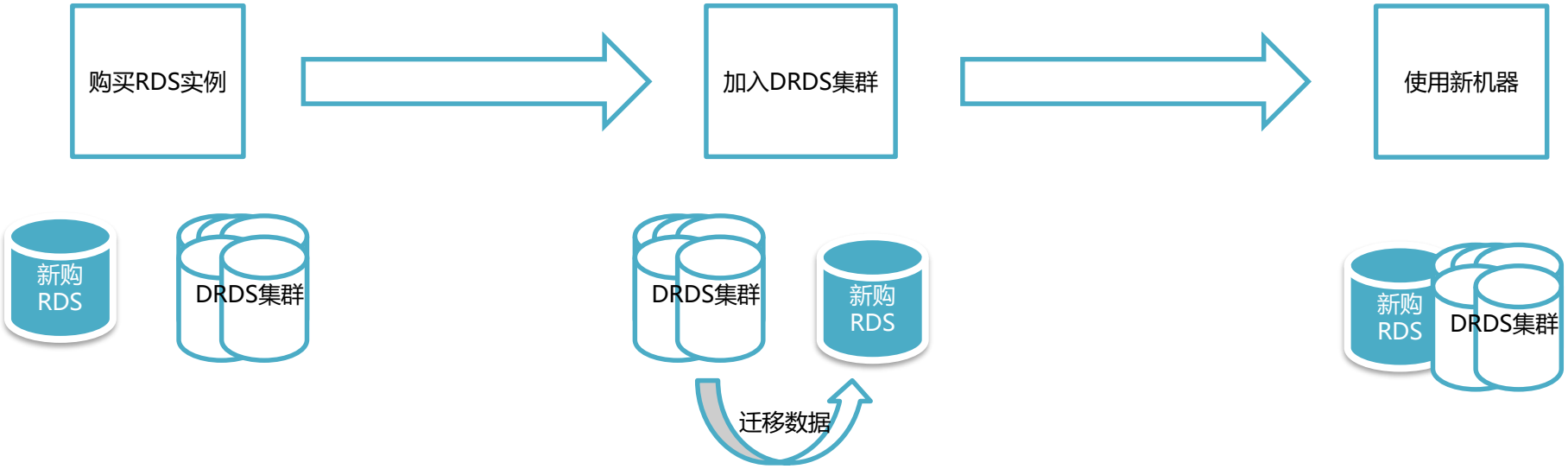
Q1:select id from A_0 order by id limit 0,1010

Q2:select id from A_1 order by id limit 0,1010

Q2:select id from A_2 order by id limit 0,1010

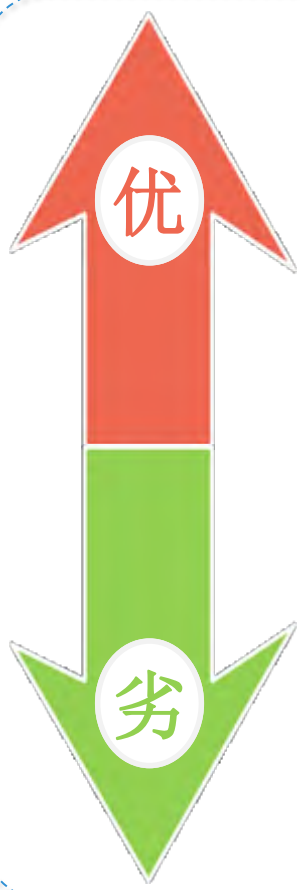
弹性扩展

自动扩容、缩容


购买RDS实例
加入DRDS集群
使用新机器
新购
RDS
DRDS集群
DRDS集群
新购
RDS
迁移数据
新购
RDS
DRDS集群

小表异步广播

跨机JOIN

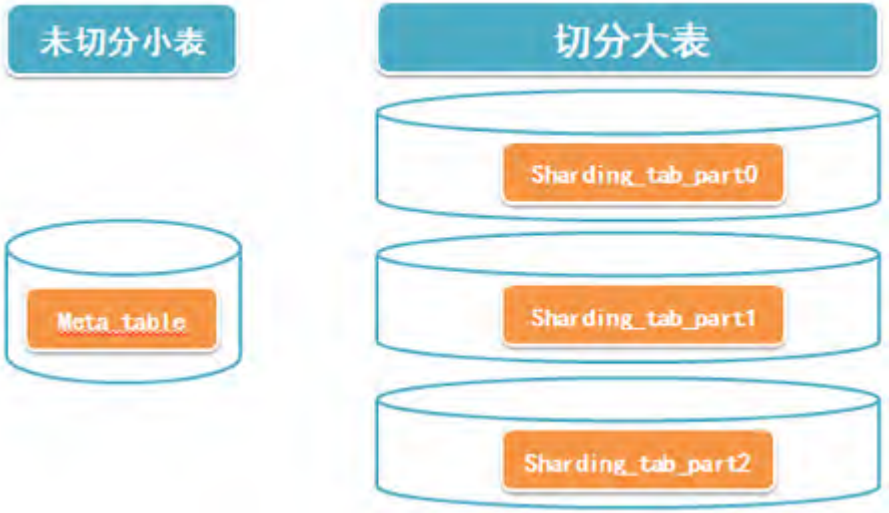


优势

一致性
空间比较节省

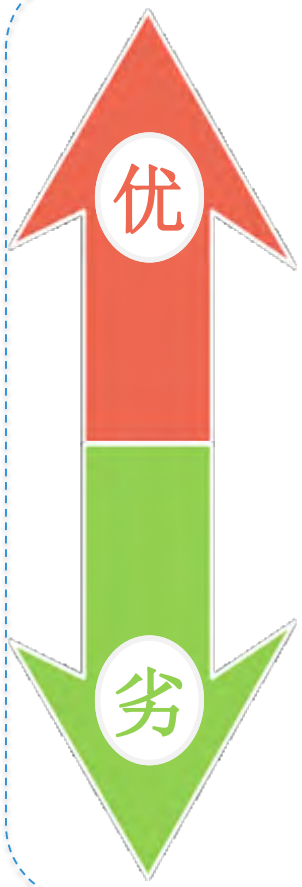
网络消耗
延迟增加

劣势



小表异步广播

小表广播JOIN

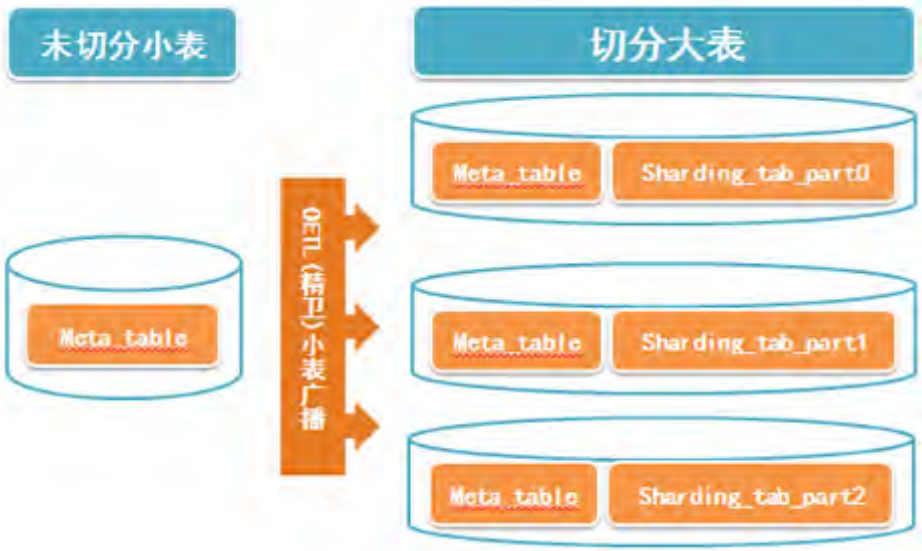


优势

性能高
延迟低
网络消耗小

最终一致性
小表更新量不能太巨大

劣势



DRDS实践

DRDS实践

01 分布式查询优化

02 事务的分布式优化

03 运维平台实践

应用场景

1

原则1：选择的shardingKey要能够让所有存储节点均衡的负载读写请求

— 系统可以简单加机器来扩展

— 没有系统瓶颈

让请求可以水平扩展

2

原则2：查询尽可能带上shardingKey

— 将跨网络请求尽可能减少

— 减少并行查询时的机器消耗，从而节省成本

布式查询优化

CASE1:应该选择哪个列作为切分条件？

按照买家ID的查询
（买家查看自己买了哪些商品）

bizOrderID	buyerID	sellerID	content
0	0	1	床上用品
1	0	2	路上用品
2	0	3	销售路由器
3	0	4	中文书籍
4	0	5	电脑
5	1	0	ipad
6	2	0	笔记本
7	3	0	铅笔
8	4	0	桌面

布式查询优化

CASE2:应该选择哪个列作为切分条件？

- 按照买家ID的查询
  （买家查看自己买了哪些商品）
- 按照卖家ID的查询
  （卖家查看自己卖了哪些商品）

Table_bid
buyerID % 4

bizOrderID	buyerID	sellerID	content
0	0	1	床上用品
1	0	2	路上用品
2	0	3	销售路由器
3	0	4	中文书籍
4	0	5	电脑
8	4	0	桌面

bizOrderID	buyerID	sellerID	content
5	1	0	ipad

bizOrderID	buyerID	sellerID	content
6	2	0	笔记本

bizOrderID	buyerID	sellerID	content
7	3	0	铅笔

布式查询优化

Table_bid
buyerID % 4

bizOrderID	buyerID	sellerID	content
0	0	1	床上用品
1	0	2	路上用品
2	0	3	销售路由器
3	0	4	中文书籍
4	0	5	电脑
8	4	0	桌面

bizOrderID	buyerID	sellerID	content
5	1	0	ipad

bizOrderID	buyerID	sellerID	content
6	2	0	笔记本

bizOrderID	buyerID	sellerID	content
7	3	0	铅笔

异构复制

Table_sid
sellerID % 4

bizOrderID	buyerID	sellerID	content
5	1	0	ipad
6	2	0	笔记本
7	3	0	铅笔
8	4	0	桌面
3	0	4	中文书籍

bizOrderID	buyerID	sellerID	content
0	0	1	床上用品
4	0	5	电脑

bizOrderID	buyerID	sellerID	content
1	0	2	路上用品

bizOrderID	buyerID	sellerID	content
2	0	3	销售路由器

布式查询优化

CASE3：
卖家在商城销售的所有商品

type	平台名
0	商城
1	专卖店

Table_bid
buyerID % 4

bizOrderID	buyerID	sellerID	type	content
0	0	1	0	床上用品
1	0	2	1	路上用品
2	0	3	0	销售路由器
3	0	4	1	中文书籍
4	0	5	0	电脑
8	4	0	0	桌面

bizOrderID	buyerID	sellerID	type	content
5	1	0	1	ipad

bizOrderID	buyerID	sellerID	type	content
6	2	0	0	笔记本

bizOrderID	buyerID	sellerID	type	content
7	3	0	1	铅笔

布式查询优化

小表异步广播

Table_bid
buyerID % 4

type	平台名
0	商城
1	专卖店

bizOrderID	buyerID	sellerID	type	content
0	0	1	0	床上用品
1	0	2	1	路上用品
2	0	3	0	销售路由器
3	0	4	1	中文书籍
4	0	5	0	电脑
8	4	0	0	桌面

type	平台名
0	商城
1	专卖店

bizOrderID	buyerID	sellerID	type	content
5	1	0	1	ipad

type	平台名
0	商城
1	专卖店

bizOrderID	buyerID	sellerID	type	content
6	2	0	0	笔记本

type	平台名
0	商城
1	专卖店

bizOrderID	buyerID	sellerID	type	content
7	3	0	1	铅笔

布式查询优化

最近1周内所有卖家销售的商品量？

CASE4：
应该选择哪个列作为切分条件？

bizOrderID	buyerID	sellerID	content	GMT_MODIFIED
0	0	1	床上用品	2014-09-01
1	0	2	路上用品	2014-09-01
2	0	3	销售路由器	2014-09-01
3	0	4	中文书籍	2014-09-01
4	0	5	电脑	2014-09-02
5	1	0	ipad	2014-09-02
6	2	0	笔记本	2014-09-04
7	3	0	铅笔	2014-09-03
8	4	0	桌面	2014-09-05

事务的分布式优化

- 目标：
  - 完整的事务支持
    - 像传统单机事务一样的操作方式
    - 可按需无限扩展

- 快醒醒~~别做梦了
- 容易理解的模型往往性能都不好，性能好的模型往往不容易理解

这就是生活

事务的分布式优化

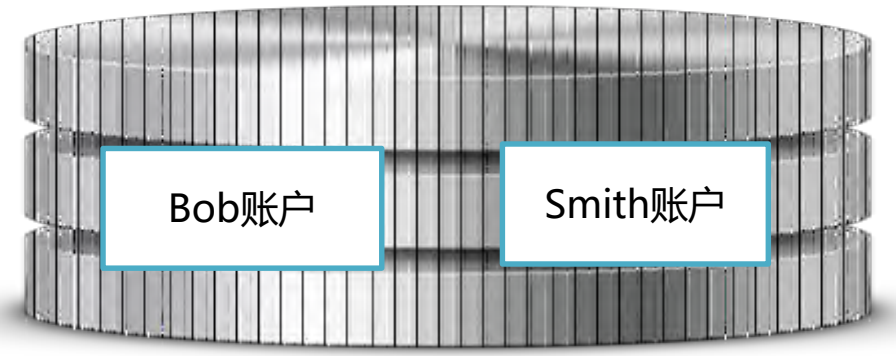
- 分布式事务的优化原则
  - 缩小锁的范围
  - 缩小锁的持有时间
  - 取消回滚

事务的分布式优化

事务单元

操作指令	耗时	总耗时
锁定Bob账户	0.001ms	5.004ms
锁定Smith账户	0.001ms	
查看Bob是否有100元	1ms	
从Bob账号中减少100元	2ms	
给Smith账户中增加100元	2ms	
解锁Bob账户	0.001ms	
解锁Smith账户	0.001ms	

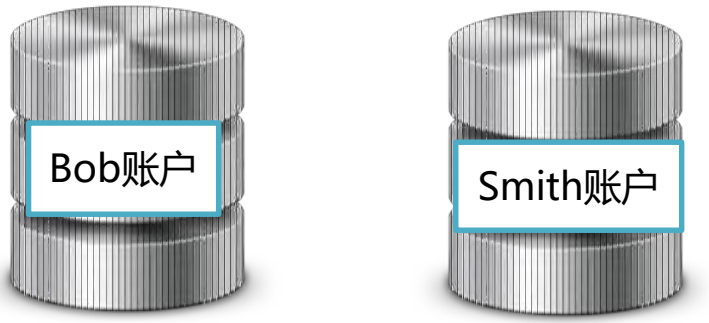
事务时间序



事务的分布式优化

操作指令	耗时	总耗时
锁定Bob账户	0.001ms	11.004ms
通过网络锁定Smith账户	2ms+0.001ms	
查看Bob是否有100元	1ms	
从Bob账号中减少100元	2ms	
通过网络给Smith账户中增加100元	2ms+2ms	
解锁Bob账户	0.001ms	
通过网络解锁Smith账户	2ms+0.001ms	

事务时间序



事务的分布式优化

事务单元

操作指令	耗时
锁定Bob账户	0.001ms
查看Bob是否有100元	1ms
从Bob账号中减少100元	2ms
解锁Bob账户	0.001ms

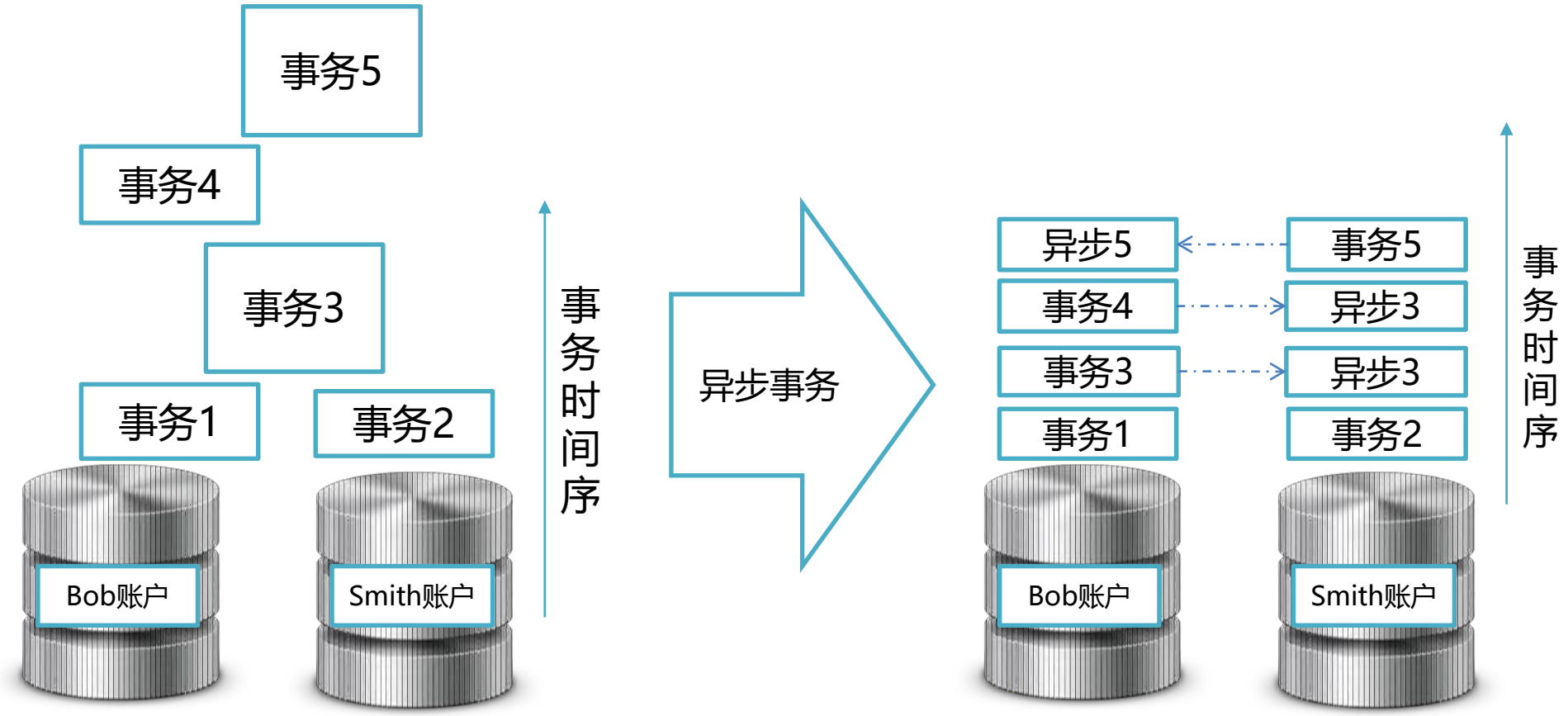
异步并行消息

异步事务单元

操作指令	耗时
锁定Smith账户	0.001ms
给Smith账户中增加100元	2ms
解锁Smith账户	0.001ms

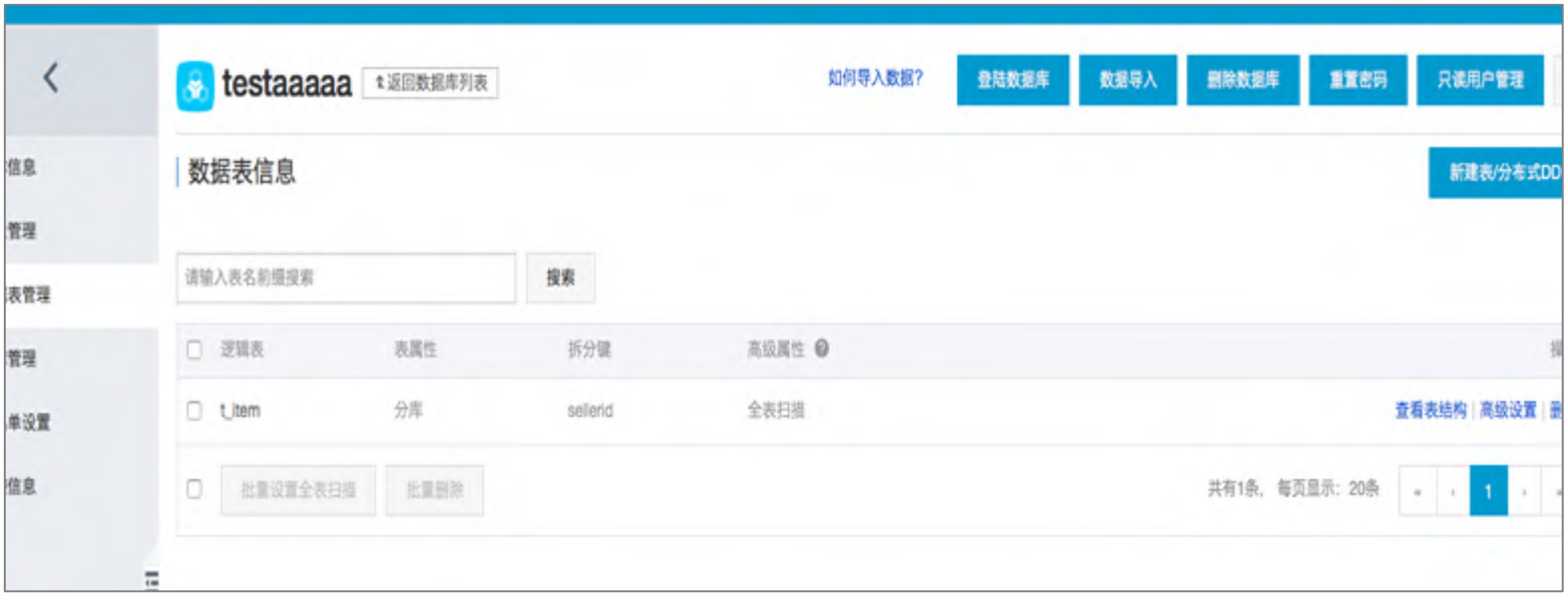
事务时间序

事务的分布式优化


事务5
事务4
事务3
事务1
事务2
Bob账户
Smith账户
事务时间序
异步事务
异步5
事务4
事务3
事务1
事务5
异步3
异步3
事务2
Bob账户
Smith账户
事务时间序

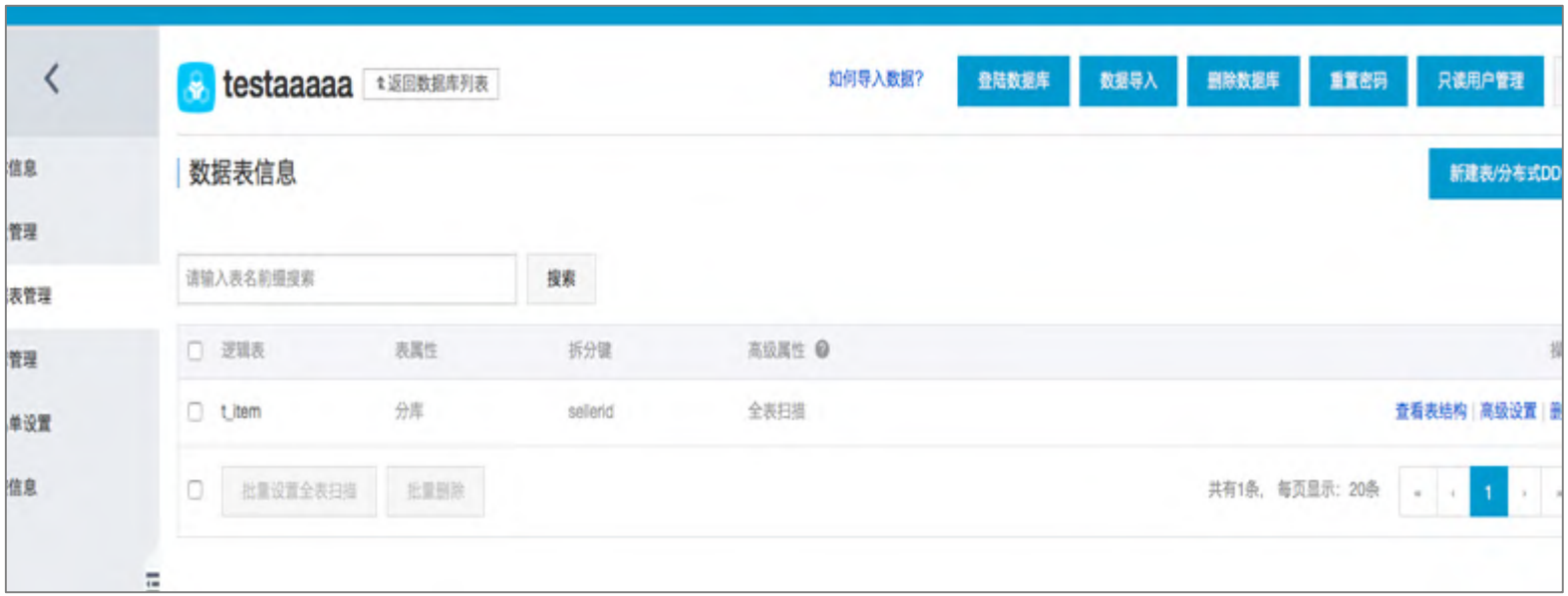
DRDS运维平台

利用控制台可以完成建库建表



DRDS运维平台

利用控制台可以完成建库建表



DRDS运维平台

- 支持标准的DDL扩展

```
CREATE TABLE multi_db_multi_tbl
   (id int auto_increment, id2 int, name
varchar(30), primary key(id))
   dbpartition by hash(id) tbpartition by
hash(id2) tbpartitions 3;
```

DRDS运维平台

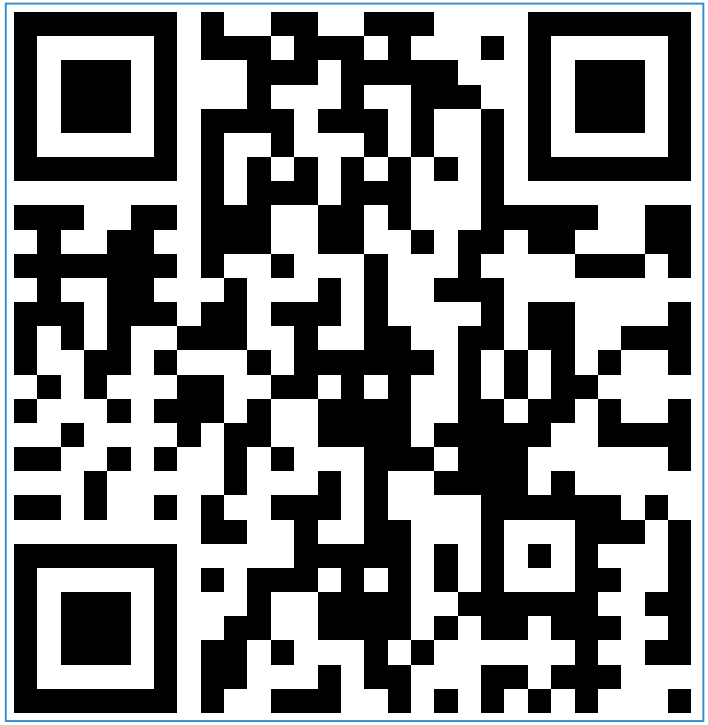
- GUI工具兼容
  - Navicat
  - MySQLWorkbench
  - Etc…

小结

欢迎选择

http://www.aliyun.com/product/drds

期待更多的行业案例



Gdevops

全球敏捷运维峰会

THANK YOU！