

# 我的经历：Taobao 数据库这5年


@tb丁原



2012-04-10

追風堂

DTCC2012

感谢 

感谢it168 , itpub

感谢Taobao dba team

感谢在场的所有同学

DTCC2012

追風堂 



◆ Taobao数据库这5年

◆ MySQL化面临的问题及应对

# weibo上的讨论



【网友热议：淘宝和阿里巴巴去Oracle化事件】我们对过去这一事件从新进行了整理信息来自于微博，知乎等网络，请进入社区不同的声音<http://t.cn/zOanngE> @红袖添香夜杀猪@mysqllops @阿里八神@tb丁原@何\_登成



hellodba: 淘宝为什么要去oracle，因为它很难适应互联网大规模应用对扩展性的要求，与其说是去I/O/E，更不如说是分布式架构战胜了集中式架构，开源系统战胜了商业系统。至于原来的Oracle DBA，现在都已经是老大了，剩下的DBA都已经转型。 (3月23日 23:38)

回复



YHY: 回复@jametong: 我的观点其实很简单，商业的未必就是好的。考虑购买一个商业产品很多是出于公司政治！ (3月27日 18:59)



老狐狸\_pub: 回复@mysqllops: 跟着领导走，这也是可以理解的，我只是认为使用mysql就一定比oracle省钱的说法，有待商榷，只是我也是猜测而已，没有任何依据 (今天 17:29)



mysqllops: 回复@老狐狸\_pub: 狐狸大哥说的对，确实要从多个角度出发，阿里系内部早是成为一种盲目的行为，现在已经开始冷却的...尤其等能干活的MySQL DBA走光的话，会跌入低谷的！ @IT168企业级频道 (今天 17:22)

DTCC2012

追風堂

# 为什么选择开源 ( MySQL )



- 成本驱动
- 公司自身技术积累，商业软件在淘宝优势逐步弱化
- 其他客观条件

上面3点，哪点是决定性的？



DTCC2012

追風堂

# 成本高，到底有多高



刚开始买服务器（拿个袋子就去了）

后来的成本（得用大卡车运钱去）

DTCC2012

追風堂

# 成本高，到底有多高



某业务真实的数据：

2010年DB+硬件的投入在1100万左右。

2011年DB+硬件的投入在2200万左右。

2012年DB+硬件的投入在4400万左右。

备注：

没有不差钱的公司

这单单只是某个业务的压力，可想整体成本的压力有多大

既然这么贵，我们为什么不尝试免费的呢

DTCC2012

追風堂



# 选择开源初期面临的挑战

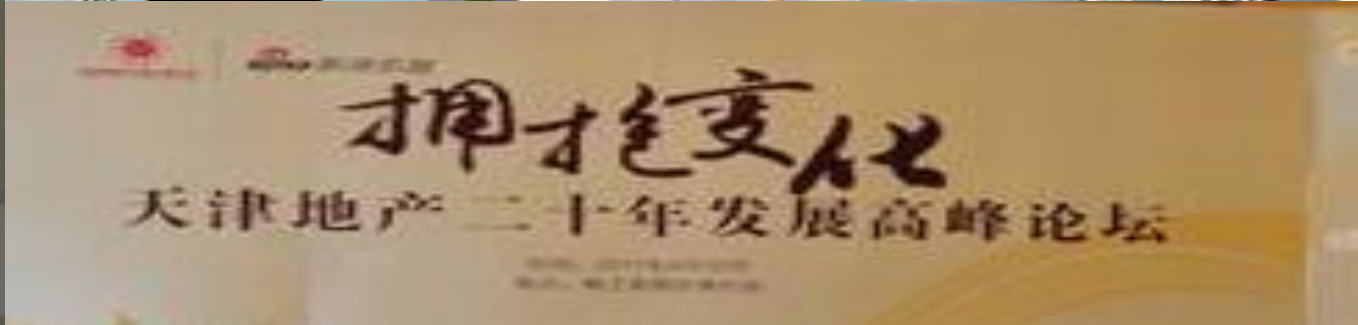


团队



重新开始

团队



4.其



--拥抱变化

[http://media.ifeng.com/news/newmedia/web/201002/0206\\_4266\\_1539529.shtml](http://media.ifeng.com/news/newmedia/web/201002/0206_4266_1539529.shtml)

DTCC2012

追風堂



# 推进过程



- 强推，阻力大？
- 一蹴而就还是按部就班，几年几年必须要完成？



1. 尝试阶段（小业务，小范围开始）
2. 积累阶段（使用过程中开发，dba面临的问题，解决问题）
3. 继续积累，验证（解决新的问题，逐步推广到更多的业务线）
4. 大规模可用阶段

DTCC2012

追風堂



开发易用性：

成熟的中间层，尽量减少开发的难度

DBA易运维：

强大的MySQL底层运维平台

其他：

思想统一，协作，配合


# 关于现在，未来



## 1.关于成本

开源&&商业解决方案，那个成本更加昂贵，现在我们很难知道，各有各的说法，但是未来一定是开源软件的（爱是做出来的，做了就一定有机会看到）。

2.定制化开源软件的趋势（hbase，mysql，hadoop，linux内核。。。）

3.  **阿里八神**: 学习oracle，精读一本大师的作品，在泛读几本其他的，扩充知识面，基本可以成型，学习数据库，也可以精通一款，再根据需要做转型，背后的思路都是想通的，关键还是看个人的心态是否够静 (11分钟前)

DTCC2012

追風堂

# MySQL的淘宝历史



年份	阶段	重要项目（里程碑）
2008年	尝试阶段	始于画报（poster）项目
-2010年	发展阶段	Tddl中间层 zdatasource 数据源 MySQL 监控逐步成熟 核心业务有计划的开始往MySQL上迁移
-现在	继续发展阶段	MySQL秒级别故障切换 MySQL semi sync，online ddl，myawr 定制自己的MySQL，彻底拥抱开源 核心业务全部迁移到MySQL db上 NoSQL尤其hbase成为db的有力补充

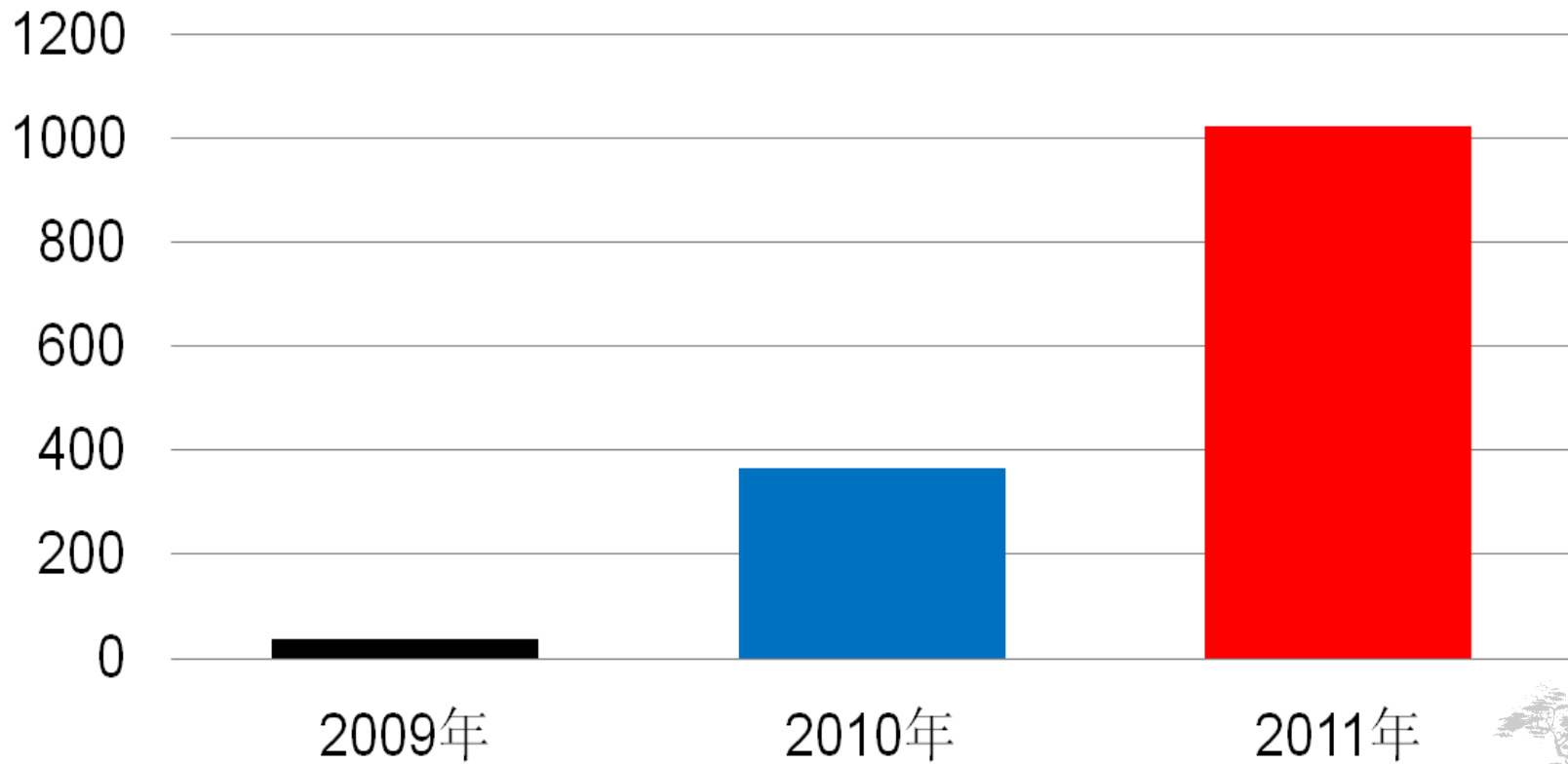
DTCC2012

追風堂

# MySQL线上服务器统计



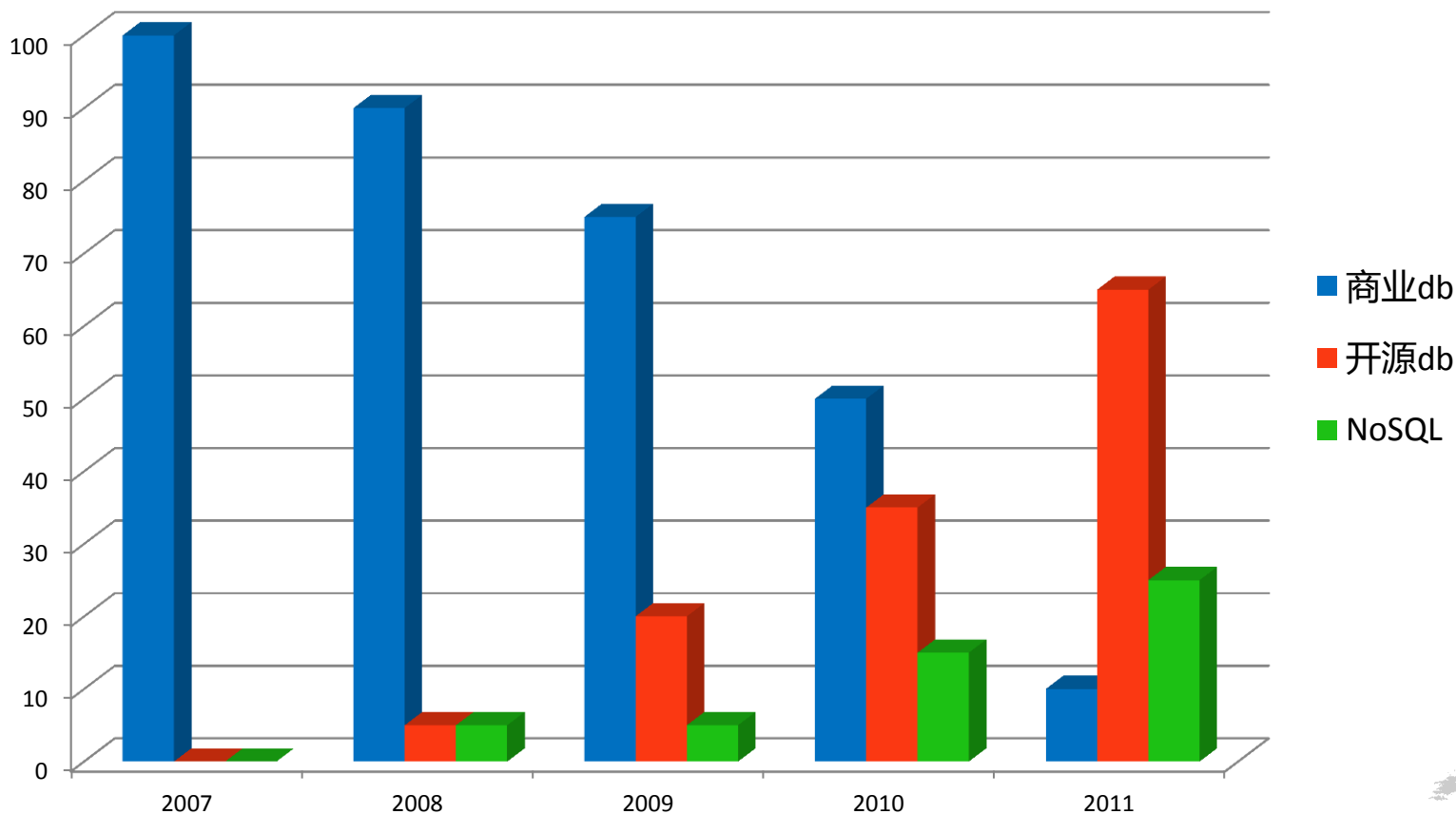
线上新增MYSQL服务器/年



DTCC2012

追風堂

# 这5年，数据存储产品比例



存储趋势上：集中式，分布式，云？

DTCC2012

追風堂





◆ Taobao数据库的这5年

◆ MySQL化面临的问题以及应对

# 商业DB vs MySQL



商业db，MySQL各自的优缺点，适用场景？

DTCC2012

追風堂

# 商业DB vs MySQL



## 商业软件

- ◆ 稳定，功能非常强大，代码非常严谨，不会出现低级的bug
- ◆ license贵，软件黑盒子
- ◆ 确实非常非常好用

## 开源软件

- ◆ 轻量级数据库（连接线程级别）
- ◆ 扩展性好，可以针对具体业务场景进行定制
- ◆ 地雷多（比如ddl出现丢表的情况）

商业软件可能适合传统业务类型，对数据库稳定性要求非常高的业务。

MySQL可能适合于变化非常快的互联网，数据量急剧膨胀，但数据的重要性相对不那么care。

那，那，那我们属于哪一类？

DTCC2012

追風堂

# 使用MySQL,你有什么顾虑



使用MySQL会有很多顾虑，开发的顾虑，dba的疑虑，没有关系，我们一起来解决。

- MySQL会丢数据吗
- MySQL容灾快速切换方案
- MySQL的性能怎么样
- MySQL开源软件自身的稳定性怎么样
- MySQL ddl锁表（阻塞写）怎么解决
- MySQL备库同步延迟，备库跟不上主库
- MySQL主备库数据的一致性校验
- 相比商业软件成熟的解决方案，MySQL+PC架构其高可用如何保证

DTCC2012

追風堂

# MySQL会丢数据吗



## 丢数据场景：

1. MySQL数据库down掉 会丢吗？
2. Mysql 服务器异常down掉（比如CPU，RAM损坏，淘宝这几年发生的几率不到5/1000）
3. 硬盘坏掉会不会丢数据？

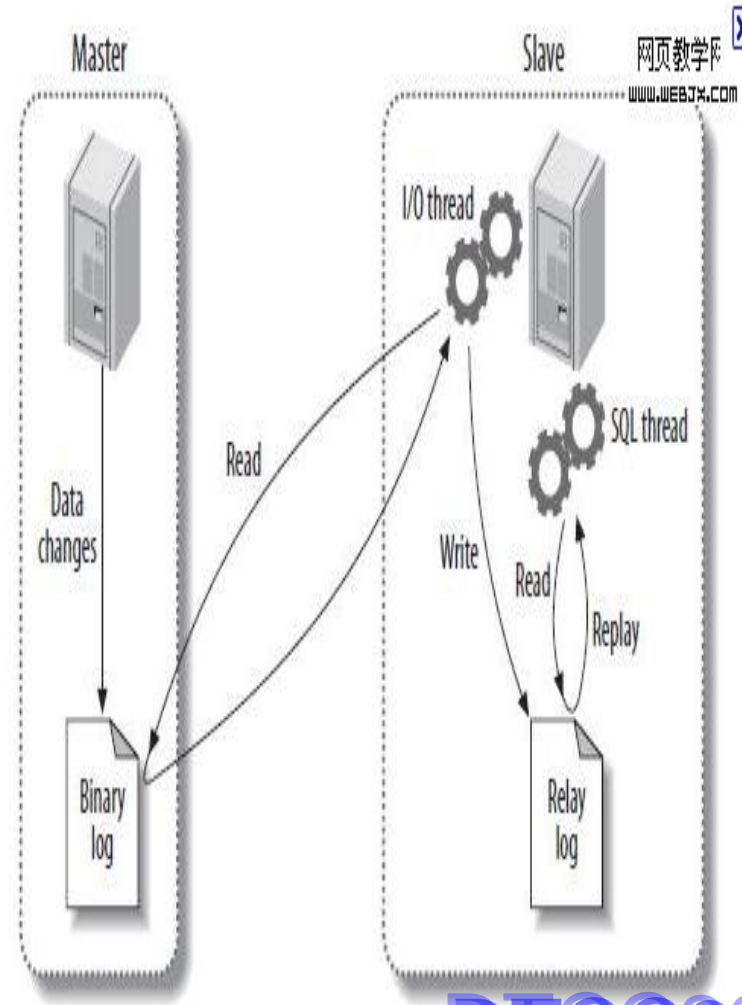
--innodb\_flush\_log\_at\_trx\_commit参数

设置为1（每个事务日志都flush到磁盘），设置为2（每个事务刷到log file中，每秒flush到磁盘中）。

-- Slave 远程binlog：

通过slave来保证数据不丢失，binlog实时传送到远程slave，基本上在毫秒之内。

<http://www.orczhou.com/index.php/2011/11/how-mysql-send-the-binary-log/>



DTCC2012

追風堂

# MySQL会丢数据吗



MySQL丢数据：

更多指MySQL采用pc服务器，pc服务器存在硬件损坏的可能性（比如内存，cpu坏掉），导致丢数据。

怎么来避免这种情况？

DTCC2012

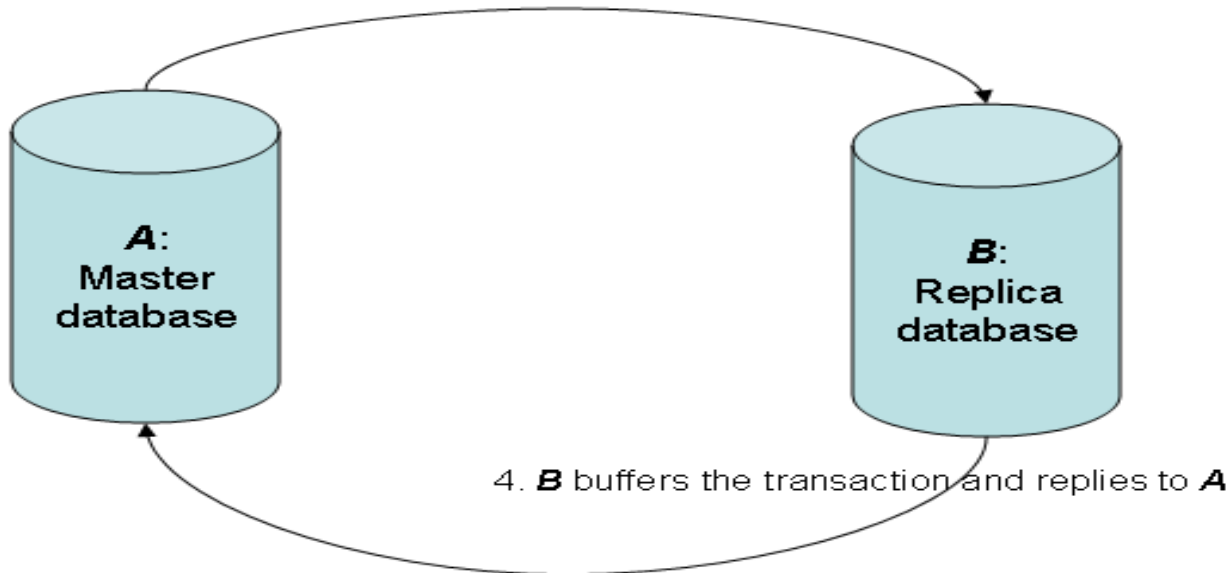
追風堂



# MySQL高可靠方案 ( Semi sync )



1. Application send a transaction to **A**
2. **A** commits transaction
3. **A** sends transaction to **B** and wait for reply



4. **B** buffers the transaction and replies to **A**
5. **A** receives the reply from **B** and can return from the transaction
6. Application returns from the transaction and can continue

淘宝MySQL对semisync做了一些改动

<http://code.google.com/p/google-MySQL-tools/wiki/SemiSyncReplication>

<http://code.google.com/p/google-MySQL-tools/wiki/SemiSyncReplicationDesign>

DTCC2012

追風堂

# 我们在用的不丢数据方案



--线上情况

- 1.应用双写（写两份）
2. 应用通过记录log来实现（比如通过notify消息）
- 3.Semi sync方案

DTCC2012

追風堂

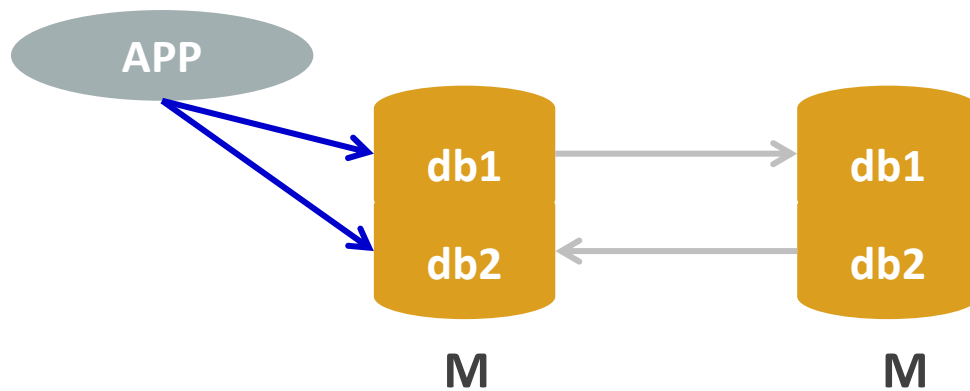
# MySQL高可用方案



硬件故障是很难避免的。

我们要做的是，挂掉的情况下如何快速恢复，减少对业务的影响？

MySQL MM（和MS的区别？）机制，双向复制，数据库秒级别切换



切换步骤上：

1.Db主备库切换

2.App数据源切换，通过zdatasource来做

两者打包，做到db和数据源一键切换，尽量缩短切换时间

DTCC2012

追風堂

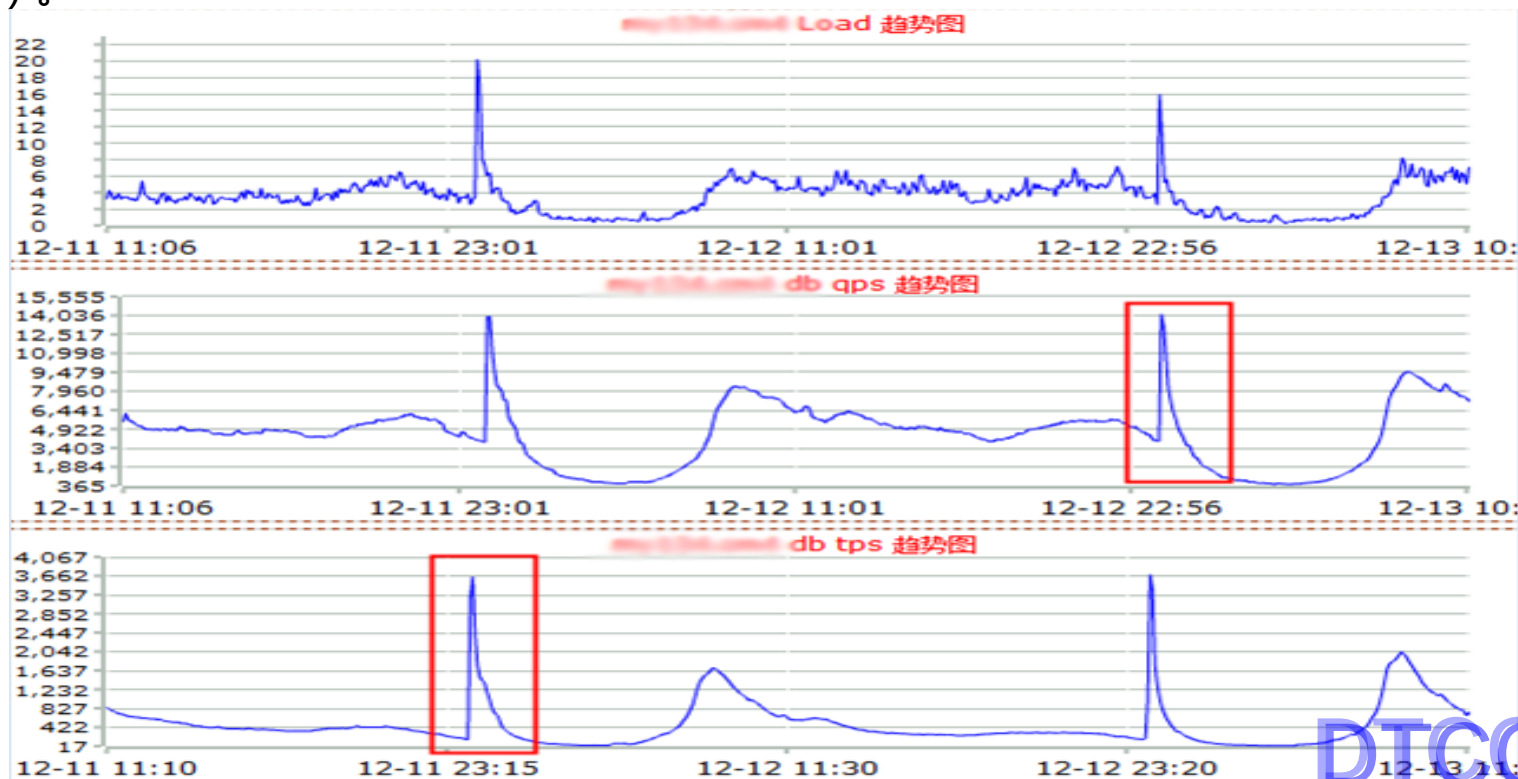
# MySQL的性能表现



线上业务机器：

基本配置：2\*4cpu 2.3G，12\*SAS, 32G RAM, MySQL 5.1.48

单台机器的dml达到了4000/s，qps超过了15000/s（性能主要看场景和数据量，仅供参考）。



DTCC2012

追風堂

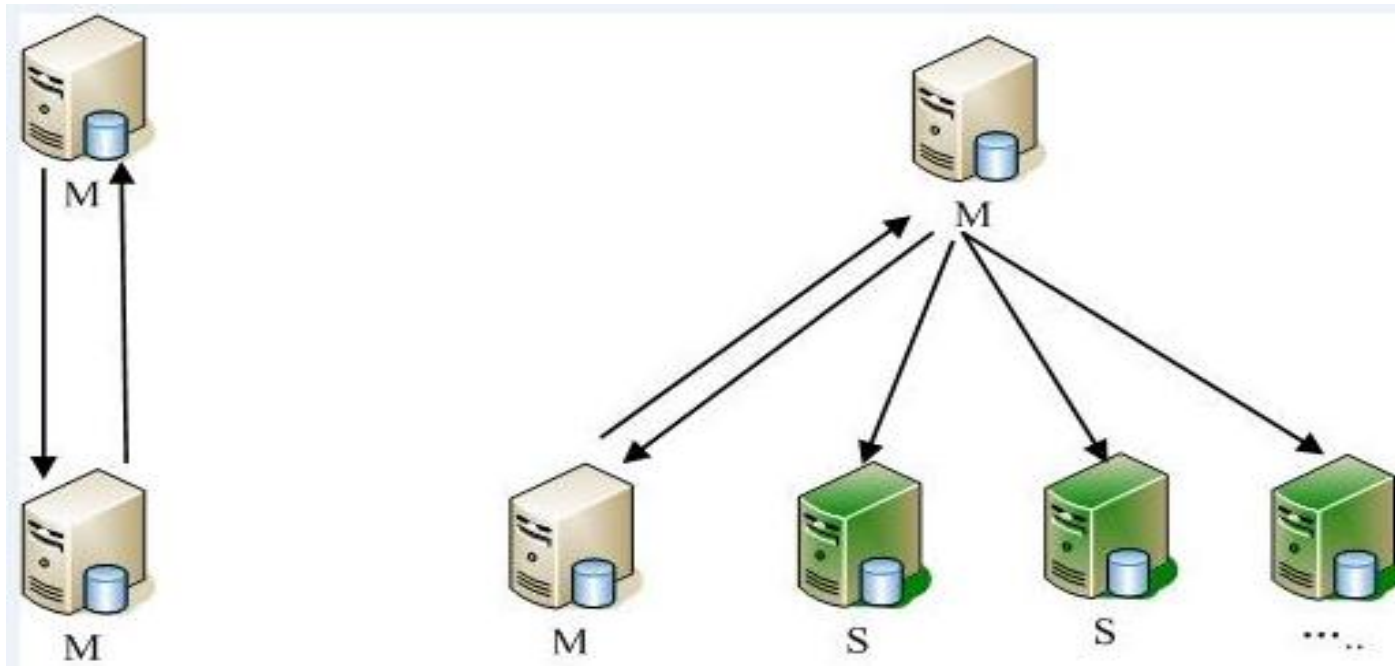
# MySQL性能之\_软件架构



线上MySQL版本：5.1.48 ， Percona Server 5.5.20

MM架构：线上主要架构

MS架构：多Slave可以提供更多的读服务，比如论坛帖子应用



DTCC2012

追風堂

# MySQL性能之\_硬件架构



业务模型决定硬件选型，业务模型是什么？

- 数据量大小
- 读写比例，读多写少还是写多读少

基本硬件配置：

- 内存基本配置24G，48G，96G，根据业务来决定内存选型，内存cache依然为王
- 磁盘：Fusion io，ssd，sas，sata，fio+flashcache，oltp应用最关注的是IOPS，磁盘是否给力直接决定了性能

备注：

这几年硬件更新换代很快，单条PC服务器的性能越来越好

DTCC2012

追風堂



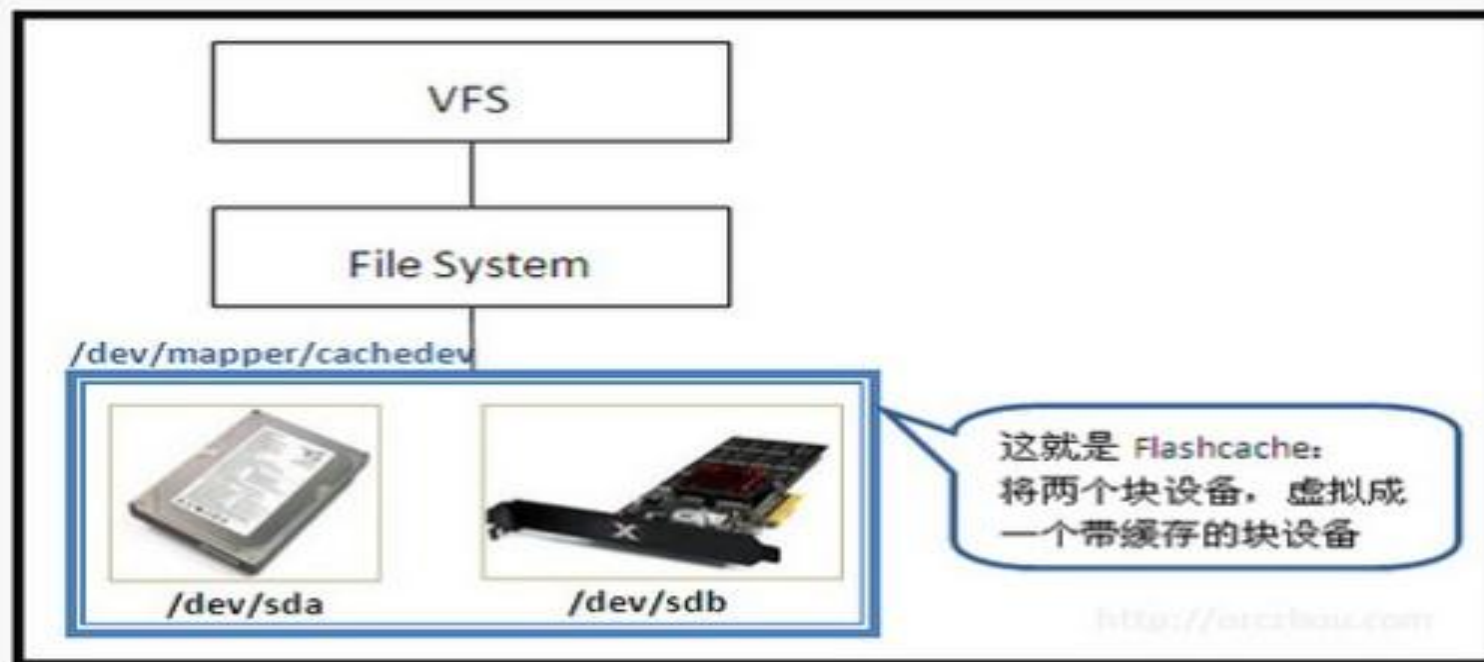
# MySQL性能之\_flashcache



源于facebook，初始用于加速MySQL innodb引擎IO

现在是通用的软件方案块加速设备（土点可以理解为二级缓存）

基本原理图



url :

<https://github.com/facebook/flashcache>

DTCC2012

追風堂

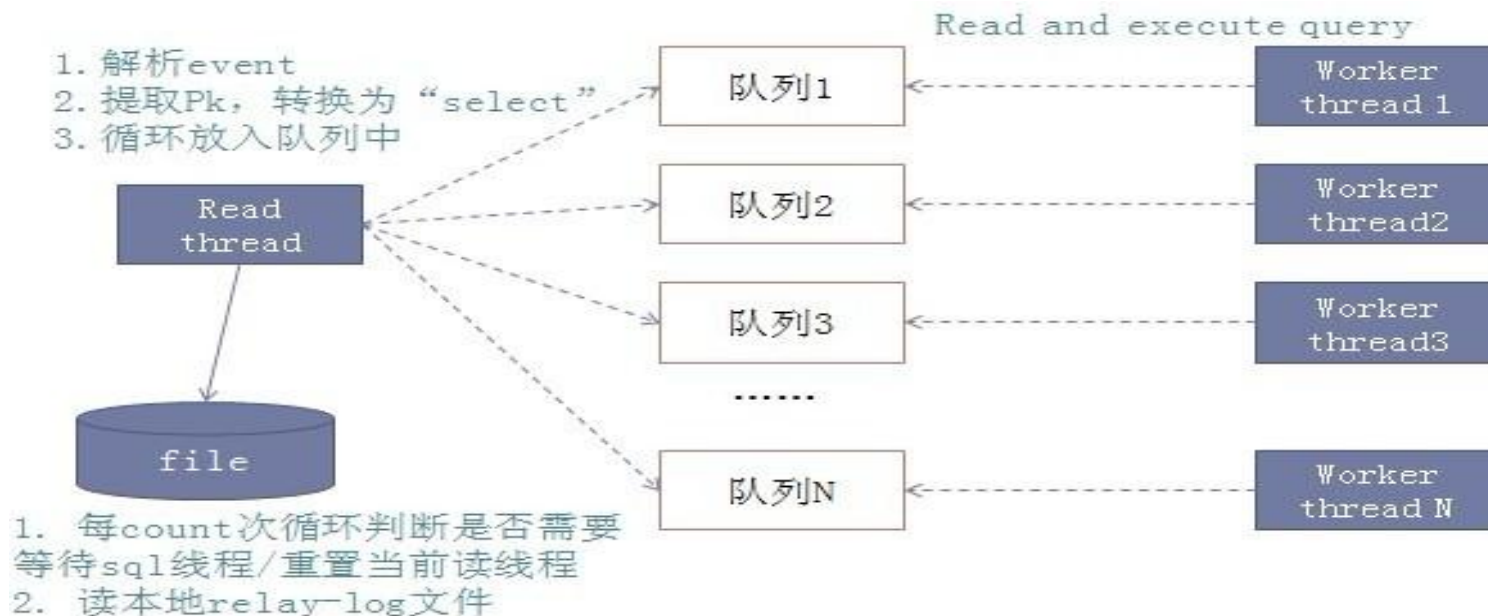
# MySQL主备库延迟解决方案 ( relay-fetch预热 )



基本思路：

在备库sql线程执行更新之前，预先将相应的数据加载到内存中

<http://relay-fetch.googlecode.com/svn/trunk/>



--业界

[http://code.google.com/p/maatkit/issues/list?q=Label:Tool-mk\\_slave\\_prefetch&sort=type](http://code.google.com/p/maatkit/issues/list?q=Label:Tool-mk_slave_prefetch&sort=type)

<http://dom.as/2011/12/03/replication-prefetching/>

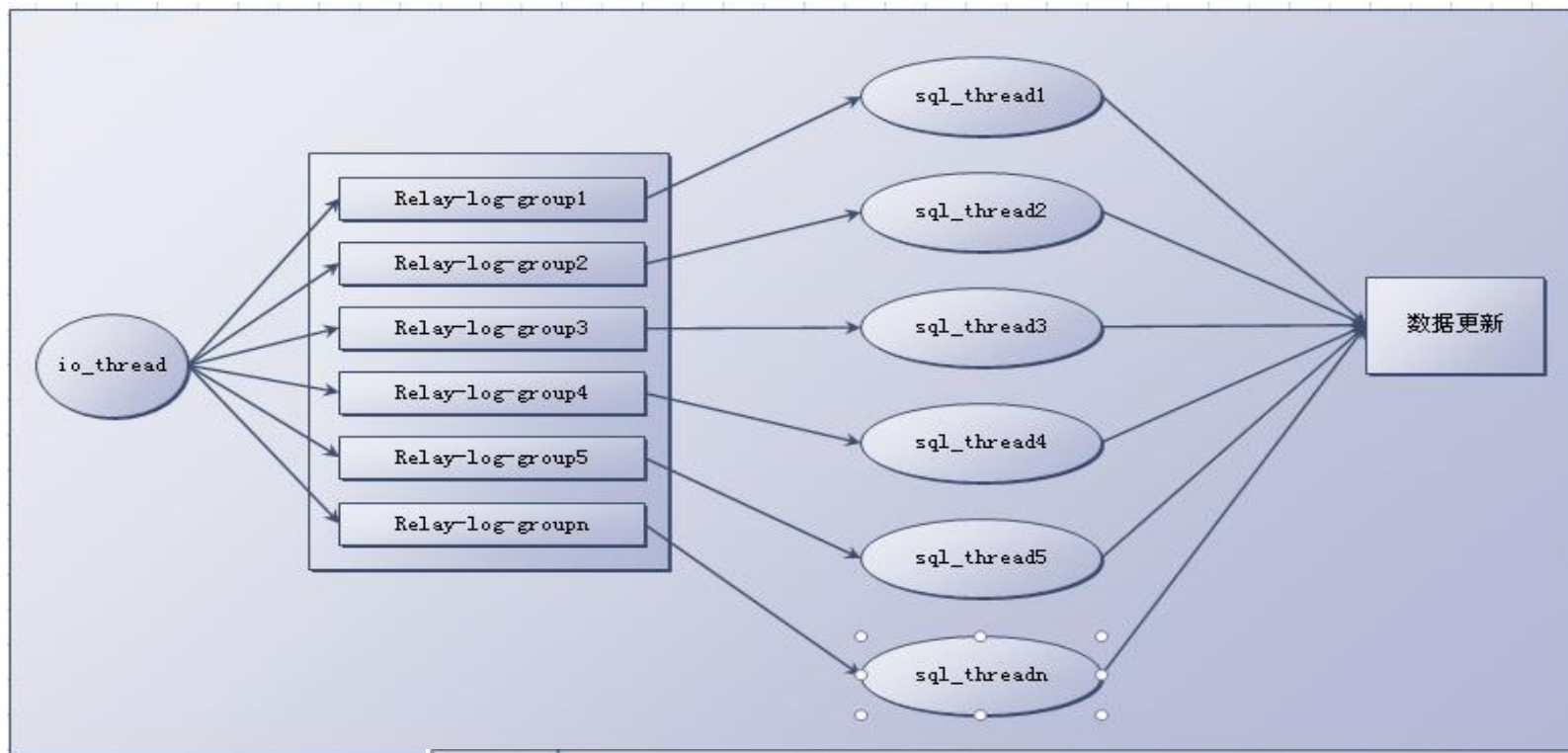
DTCC2012

追風堂

# MySQL主备库延迟解决方案 ( transfer )



改进方案：  
多线程



原有方案：单线程



--相关资料

<http://vdisk.weibo.com/s/2lBnN/1329966761> ( 淘宝丁奇 )

DTCC2012

追風堂

# MySQL online ddl



online ddl工具：

**业界online ddl工具基本实现原理都是一致的，原理基本差不多的。**

<http://www.facebook.com/notes/mysql-at-facebook/online-schema-change-for-mysql/430801045932>

能做到的是：在ddl table的过程中，app依然可以进行读写操作。

--工具链接

oak-online-alter-table：

<http://openarkkit.googlecode.com/svn/trunk/openarkkit/doc/html/oak-online-alter-table.html>

pt-online-schema-change：

<http://www.percona.com/doc/percona-toolkit/2.0/pt-online-schema-change.html> (pecona出品)

myddl：<http://www.ningoo.net> (ningoo开发，已经在用)

追風堂  
DTCC2012

# MySQL主备库逻辑复制的风险



主备库：

MySQL 通过逻辑（statement or row模式）复制来实现主备库数据同步。  
逻辑复制理论上是有风险的，极端情况下可能存在主备库数据不一致。

应对方案：

- 1.尽量采用row模式
- 2.主备库定期数据一致性校验
- 3.数据生命周期的binlog尽量保存下来

DTCC2012

追風堂

# MySQL其他问题应对



- MySQL高可用解决方案

MM，MS机制

动态数据源机制实现异常快速切换（秒级别）

通过改造后的semi sync来保障数据不丢失，或是应用设计上高可用考虑

应用数据补偿方案

- 应用设计往前一步，任何设计都假设MySQL挂掉的应对方案
- 开源软件带来更多灵活性，遇到BUG，及时去修复bug，也可以定制我们自己的MySQL

采用MySQL：

App一定要站在db的角度来考虑问题，db站在app的角度来思考。

DTCC2012

追風堂



# 使用MySQL,你还有什么顾虑



DTCC2012

追風堂

# 不断假设，不断去验证



--测试（假设了很多场景，寻找解决方案，不断的打消我们自己的疑虑）

A	B	C	D	E	F	G	H	I	J
场景							DB	vuser	测试时间
	tps	rt	cpu		load				
	avg	avg (ms)	client	tp	client	tp			
4倍数据量下、6倍读写 (10:1) 压力的性能测试最大值	12184	10	15%	17%	2.1	1.9	select的 qps在3w5	89	09/08/2011
4倍数据量下、6倍读写 (10:1) 压力的稳定性测试	11000-12000	10	20%	20%	3	2.5	select的 qps在3w5	110	01/09/2011

--异常测试

A	D
场景	现象
mysql异常重启	Mysql重启后，命中率在8min左右的时候恢复到了重启前的正常水平（97%）
cpu耗尽	cpu耗尽后，应用tps基本跌0，mysql端恢复后，应用tps慢慢恢复正常
内存耗尽	内存耗尽后，应用的tps基本跌0，mysql端恢复后，应用tps慢慢恢复正常
主备切换	测试环境无法进行主备切换--备库为普通sas盘，tps超过2000之后就出现主备延时

DTCC2012

追風堂



怎么让开源软件好用，怎么让普通pc服务器健壮起来，这需要所有人的参与。

## --现有情况

1. MySQL 源码debug团队
2. 中间层（比如taobao的tddl），尽量降低分布式MySQL下的开发成本
3. MySQL异常快速容灾切换，尽量做到对开发完全透明
4. Os(linux)问题定位非常熟练
5. MySQL online ddl解决方案
6. 团队的支持，易用性稳定性努力（tddl团队，核心研发MySQL团队等）
7. 工具化，能工具的尽量不人肉
8. 非常完善的监控体系



MySQL定位：

就是个开源软件。

轻量级数据库，蛮好用的。

**不要去比较，有优势，也有劣势。**

分库分表，你是否厌倦了？



MySQL化的意义。

对于我们来说，真的只是把db改成MySQL了吗？



思想

思想

打破思想

打破集中式的思维

把db看成只是软件

Db+app，开发和dba一起来实现高可用

# 谢谢大家



DTCC2012

追風堂