微信移动客户端
数据存储优化实践

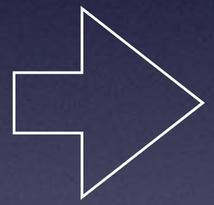
guoling @ QCon, 2017

移动端数据库痛点

- 并发性能低下
- 数据损坏常见
- 访问接口难用

引子

- 启动app超时
- 一个会话两个表
- 线性增长
- 小改动，大成效



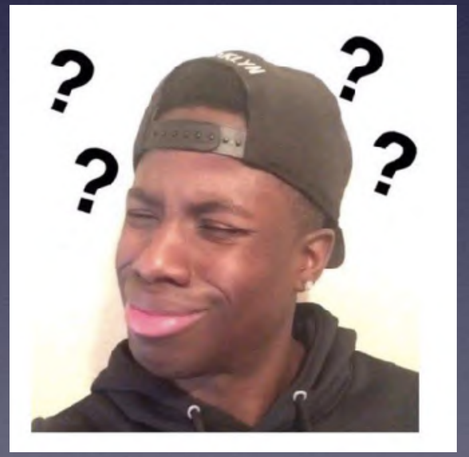
SQLite不为移动端
量身定制

并发性能

微信消息收发场景

消息并发性能

- 用户反馈消息收发卡顿



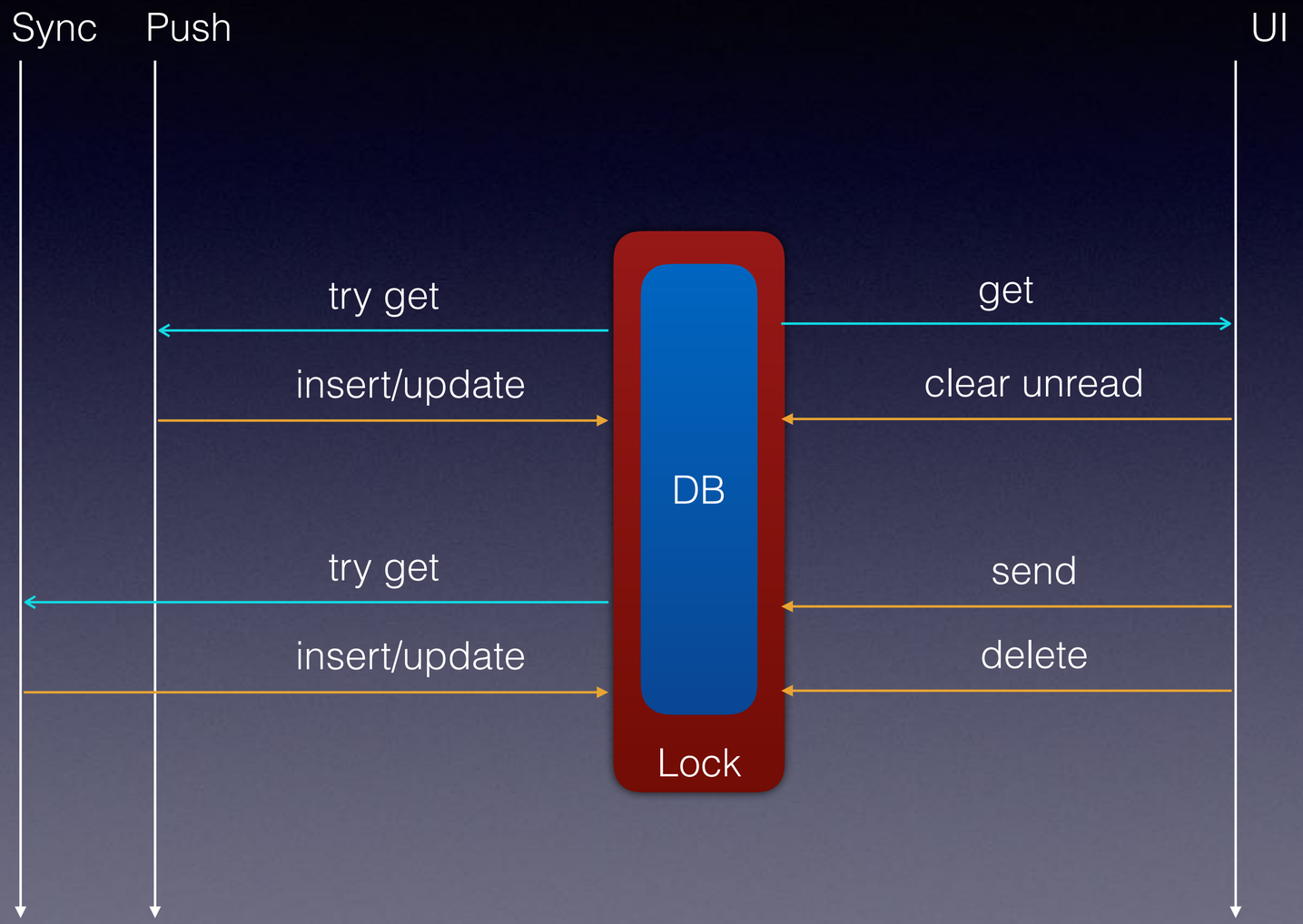
工欲善其事，必先利其器

卡顿监控系统：

Lag函数名	Lag 类型	Lag次数	Lag次数比例
CScopedLock::CScopedLock(NSRecursiveLock*) -[WCDataBase getProperty:ofClass:fromTable:withQueryString:]	11	17993	1.05%
CScopedLock::CScopedLock(NSRecursiveLock*) -[WCDataBase getProperty:ofClass:fromTable:withQueryString:]	6	17120	1%
CScopedLock::CScopedLock(NSRecursiveLock*) -[WCDataBase getObjectsOfClass:fromTable:onProperties:where:orderBy:limit:getError:]	6	9226	0.54%
CScopedLock::CScopedLock(NSRecursiveLock*) -[WCDataBase getObjectsOfClass:fromTable:onProperties:where:orderBy:limit:getError:]	11	6173	0.36%
CScopedLock::CScopedLock(NSRecursiveLock*) -[WCDataBase getCountOfObjectsOfClass:fromTable:withDistinctProperty:where:]	6	3791	0.22%

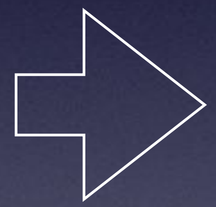
Bugly Demo

线程模型


Sync
Push
UI
try get
get
insert/update
clear unread
DB
try get
send
insert/update
delete
Lock

消息并发性能

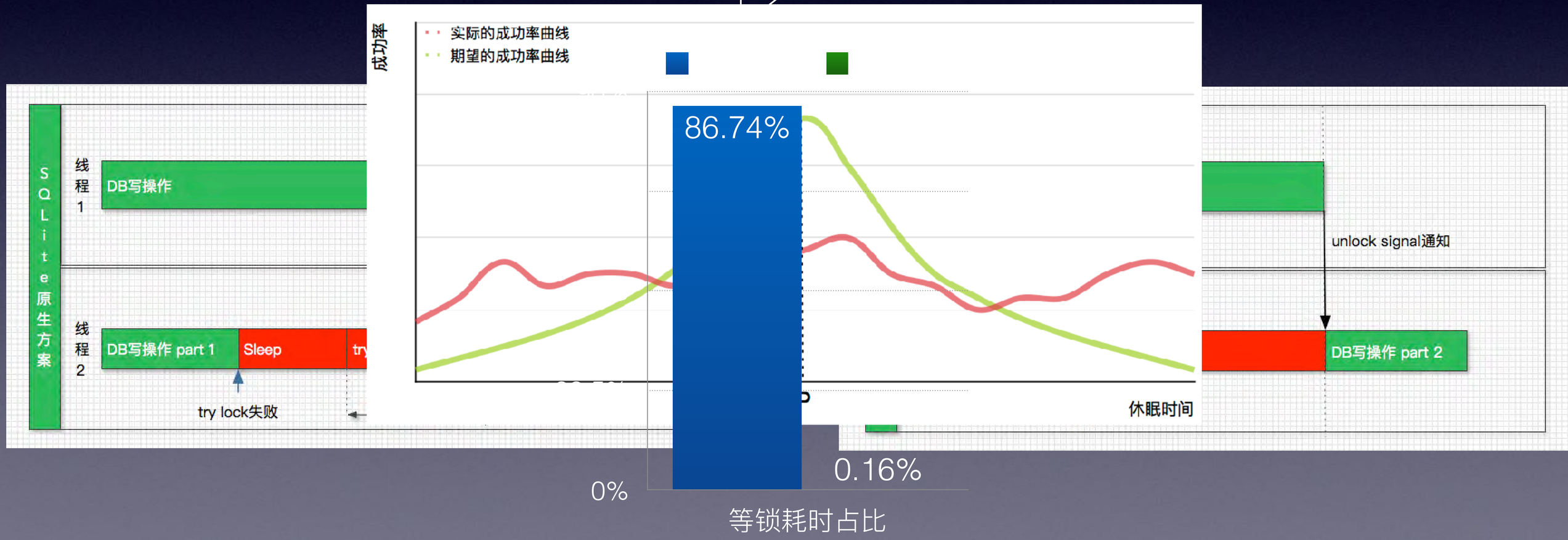
- 单handle，大锁
- Rollback Journal



- 多handle，共享锁
- WAL (Write Ahead Logging)

消息并发性能

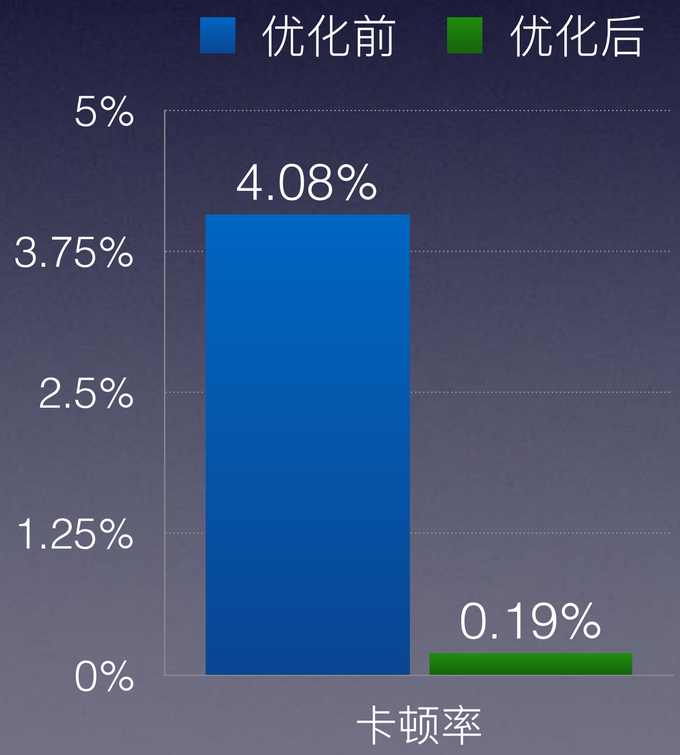
- 进程锁，busy-retry
- 线程锁，sleep-wakeup



消息并发性能

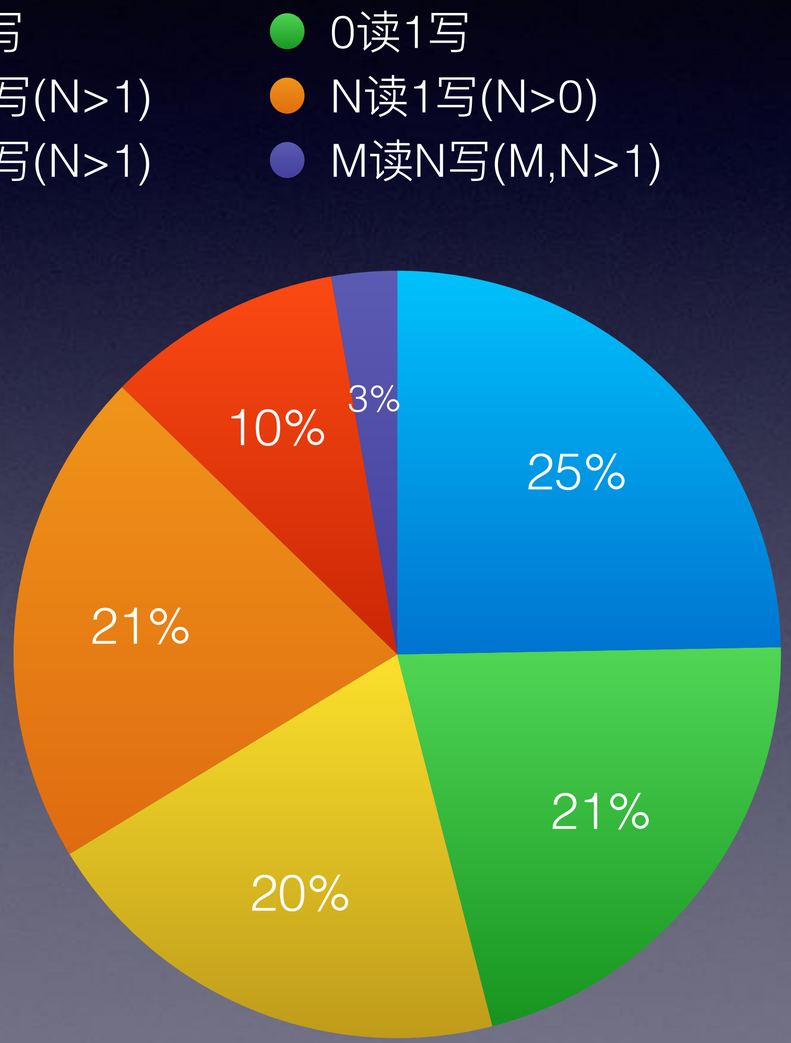
- FIFO抢锁，主线程饥饿

- 优先唤醒主线程



消息并发性能

- 单线程还是多线程?



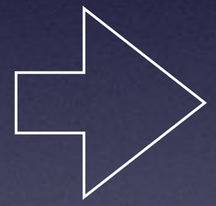
DB损坏

- DB损坏率约0.02%
- DB修复成功率约30%

为何SQLite如此脆弱?

官方解释

- 文件错写
- 文件锁bug
- sync失败
- 设备损坏
- 内存覆盖
- 操作系统bug
- SQLite bug



实际原因

- 空间不足
- 设备断电
- sync失败

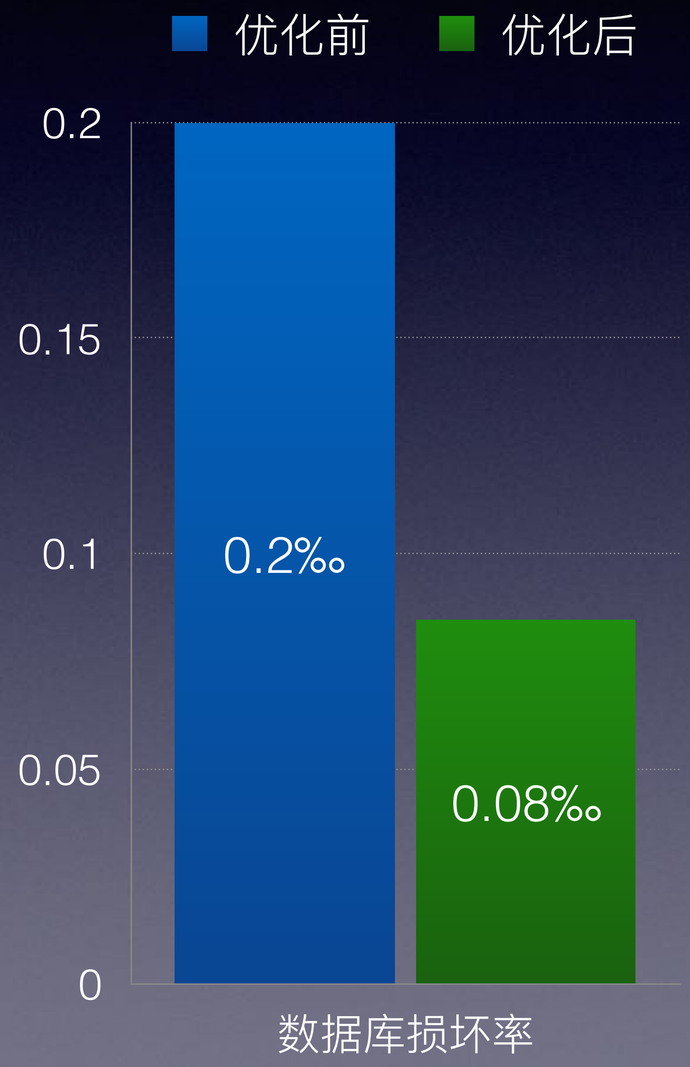
降低损坏率

- 自动清理缓存，减少空间占用
  - 业务文件先注册后使用，内审

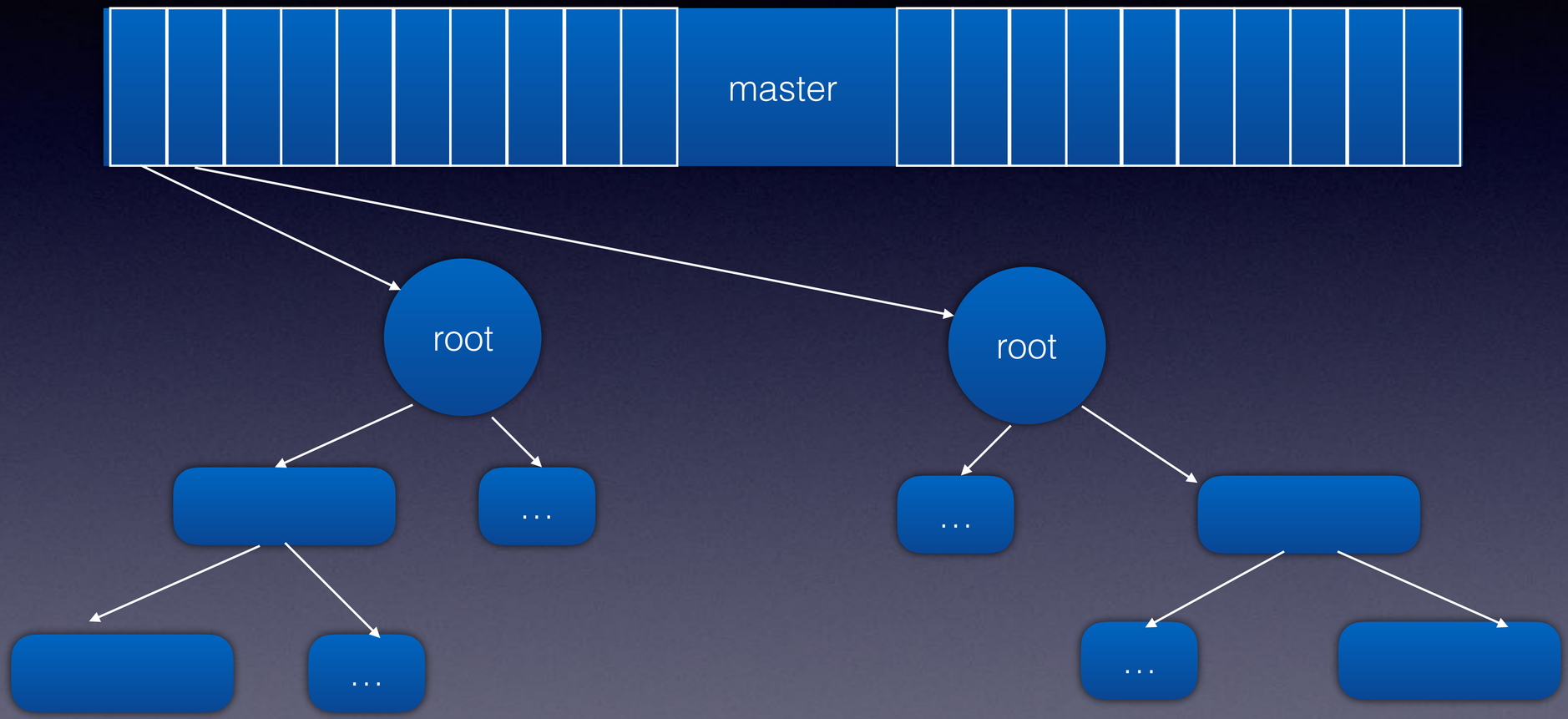
```
// 头像
{MM_DISK_USAGE_BIZ_HEAD_IMG, kDUPathPrefixUserLibrary, @"HeadImg", kDUStorePermanent}, // 头像存储目录，guoling
{MM_DISK_USAGE_BIZ_HEAD_IMG, kDUPathPrefixUserLibrary, @"RoundImg", kDUStoreTwoWeek},  // 圆角头像存储目录，已弃用，在FIC中存储，guoling
{MM_DISK_USAGE_BIZ_HEAD_IMG, kDUPathPrefixUserDocument, @"Usr", kDUStoreTwoWeek}, // 旧头像存储目录，逐步迁移到新头像目录，guoling
```

降低损坏率

- 自动清理缓存，减少空间占用
- 设置sync，同步回写DB
- 设置fullsync，挖掘平台特性



SQLite

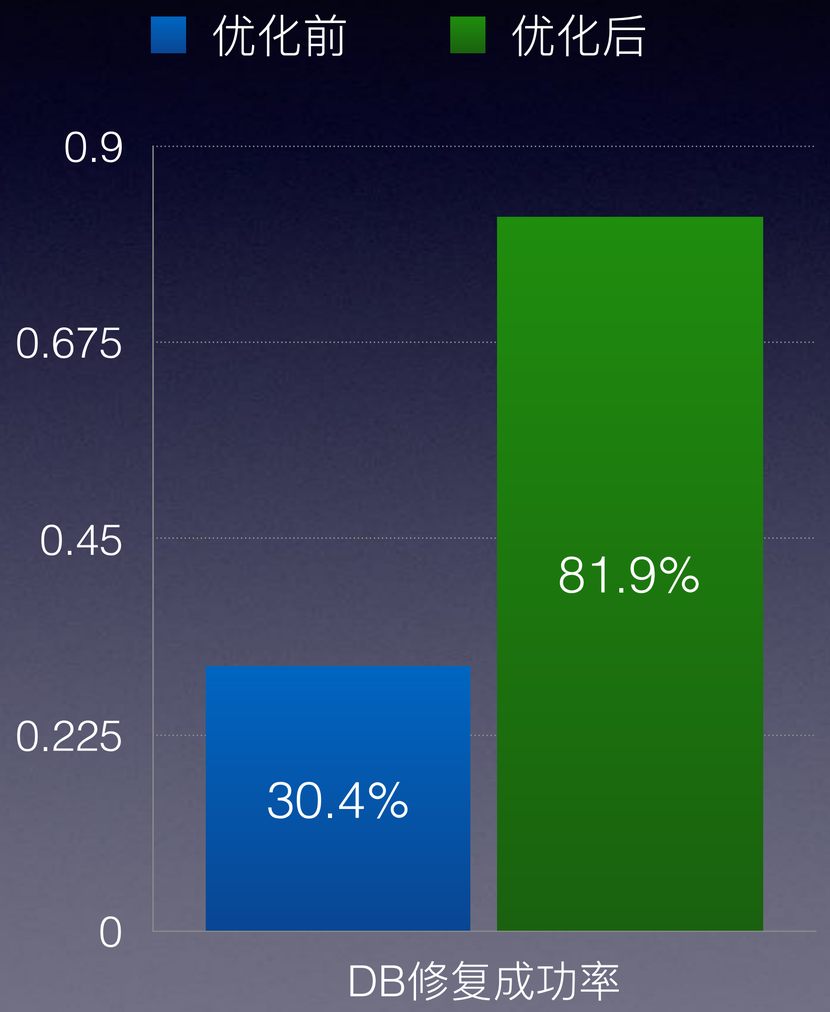


优化修复算法

- master表损坏
- 备份文件损坏
- 文件空间不足

- 轮询备份master表
- 双备份+crc32校验
- 预先分配文件空间

优化效果



WCDB组件

- 高效并发
- 备份修复
- 用户友好

ORM

- 简易的ORM描述

```
767
768 @interface MMInfo : NSObject<WCTTableCoding>
769
770 @property (nonatomic, strong) NSString* username;
771 @property (nonatomic, strong) NSString* nickname;
772 @property (nonatomic, strong) NSString* signature;
773 @property (nonatomic, assign) CGFloat height;
774 @property (nonatomic, assign) CGFloat weight;
775 @property (nonatomic, assign) UInt32 age;
776
777 WCT_PROPERTY(username);
778 WCT_PROPERTY(age);
779
780 @end
781
782 @implementation MMInfo
783
784 WCT_IMPLEMENTATION(MMInfo)
785 WCT_SYNTHESIZE_COLUMN(MMInfo, username, "usrname")
786 WCT_SYNTHESIZE(MMInfo, nickname)
787 WCT_SYNTHESIZE(MMInfo, signature)
788 WCT_SYNTHESIZE(MMInfo, height)
789 WCT_SYNTHESIZE(MMInfo, weight)
790 WCT_SYNTHESIZE(MMInfo, age)
791
792 WCT_PRIMARY(DBContact, userName)
793
794 @end
795
```

CRUD操作

- 面向对象的操作接口

```
MMInfo* info = [[MMInfo alloc]init];
info.username = @"weixin";
info.nickname = @"微信";
info.signature = @"Hello, world!";
info.height = 0.618;
info.weight = 3.1415926;
info.age = 6;

// 建DB
WCTDataBase* db = [[WCTDataBase alloc] initWithPath:nsDBPath];
// 建表
if ([db createTableOfName:@"mminfo" withClass:MMInfo.class]) {
    WCTTable* table = [db getTable:@"mminfo" withClass:MMInfo.class];
    // 插入
    [table insertOrUpdateObject:info];
    // 查询
    MMInfo* newInfo = [table getOneObjectWhere:MMInfo.username == @"weixin"];
    // 删除
    [table deleteObject:newInfo];
}
```

链式调用

- 懒加载

```
WCTSelect* select = [[[[table prepareSelectObjects]
                       where:MMInfo.username != @"weixin"]
                       orderBy:MMInfo.age.order(WCTOrderedDescending)]
                       limit:1024];
// get all by once
NSArray* arrResult = select.allObjects;

// or lazy load
while (MMInfo* info = select.nextObject)) {
    NSLog(@"%@", info.nickname);
}
```

SQL条件表达式

- 符合习惯的表达式 (类型检查)

```
// FMDB
NSString* nsWhere = [NSString stringWithFormat:@" %s = %u AND %s = %u AND %s != %u AND (%s != %u OR (%s = %u AND %s & %u))"
            , COL_STATUS, MM_MSGSTATUS_DELIVERED
            , COL_DES, DES_TO
            , COL_TYPE, MM_DATA_SYS
            , COL_TYPE, MM_DATA_VOICEMSG, COL_TYPE, MM_DATA_VOICEMSG, COL_IMG_STATUS, AUDIO_DOWNLOAD_BITSET];

// WCDB
WCTCondition condition = (DBMessage.status == MM_MSGSTATUS_DELIVERED
                         && DBMessage.des == DES_TO
                         && DBMessage.type != MM_DATA_SYS
                         && (DBMessage.type != MM_DATA_VOICEMSG
                             || (DBMessage.type == MM_DATA_VOICEMSG && (DBMessage.imageStatus & AUDIO_DOWNLOAD_BITSET))));
```

他山之石

- CoreData（上手难，多线程坑，版本升级坑)
- FMDB（接口简单，面向过程，胶水代码)
- Realm（MVCC，接口丰富，无法跨线程传递对象)

Stay hungry. Stay foolish.

ISBN: 0-14-303950-3