

网 易 数 帆 旗 下



MyRocks 在网易使用和优化实践

D I G I T A L S A I L

王刚

网易杭研 – 大数据平台产品部 – 数据基础设施

网 易 数 帆 旗 下



目 录

- 01 MyRocks 技术实现
- 02 MyRocks 优点分析
- 03 MyRocks 网易案例
- 04 MyRocks 调优实践

MyRocks 技术实现

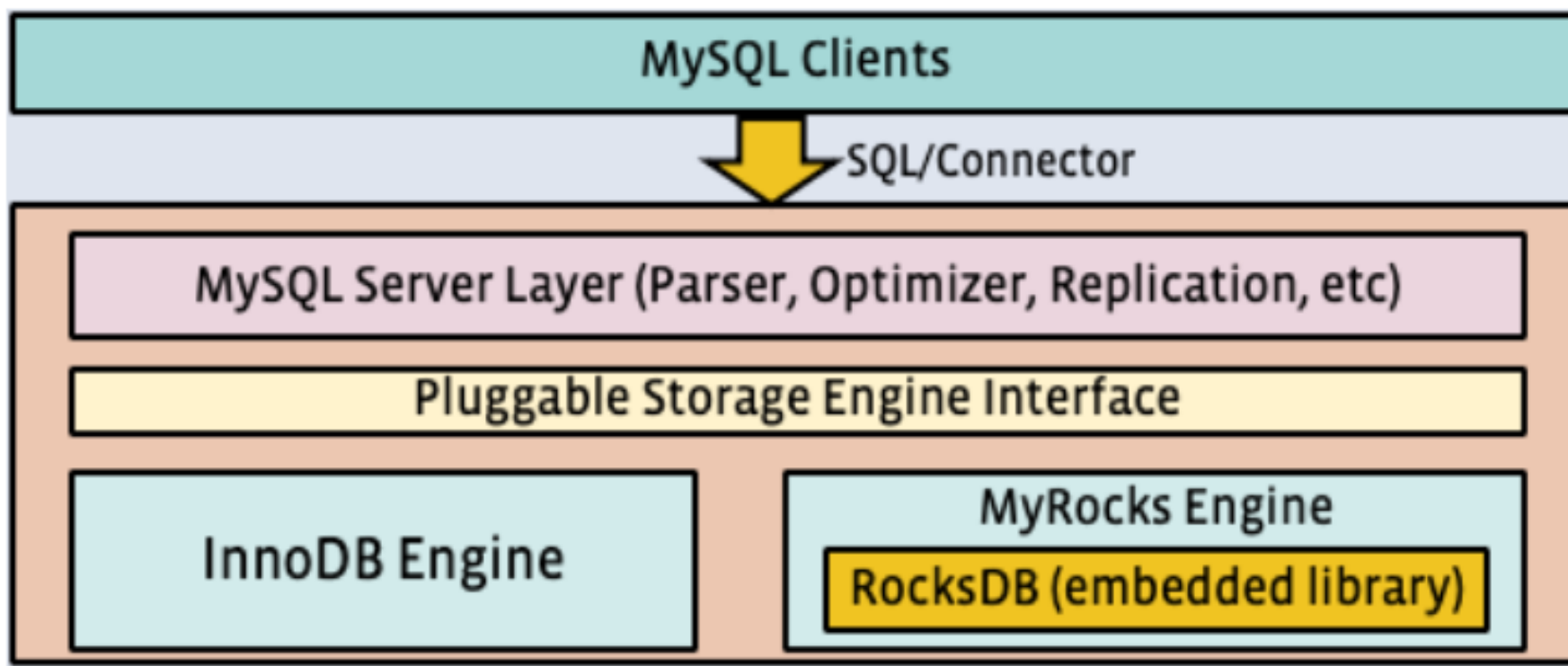
D I G I T A L S A I L

网 易 数 帆 旗 下



MyRocks 是什么

MyRocks, a new MySQL storage engine, was built on top of RocksDB by adding relational capabilities.



注：来自论文 MyRocks: LSM-Tree Database Storage Engine Serving Facebook' s Social Graph.

RocksDB 架构

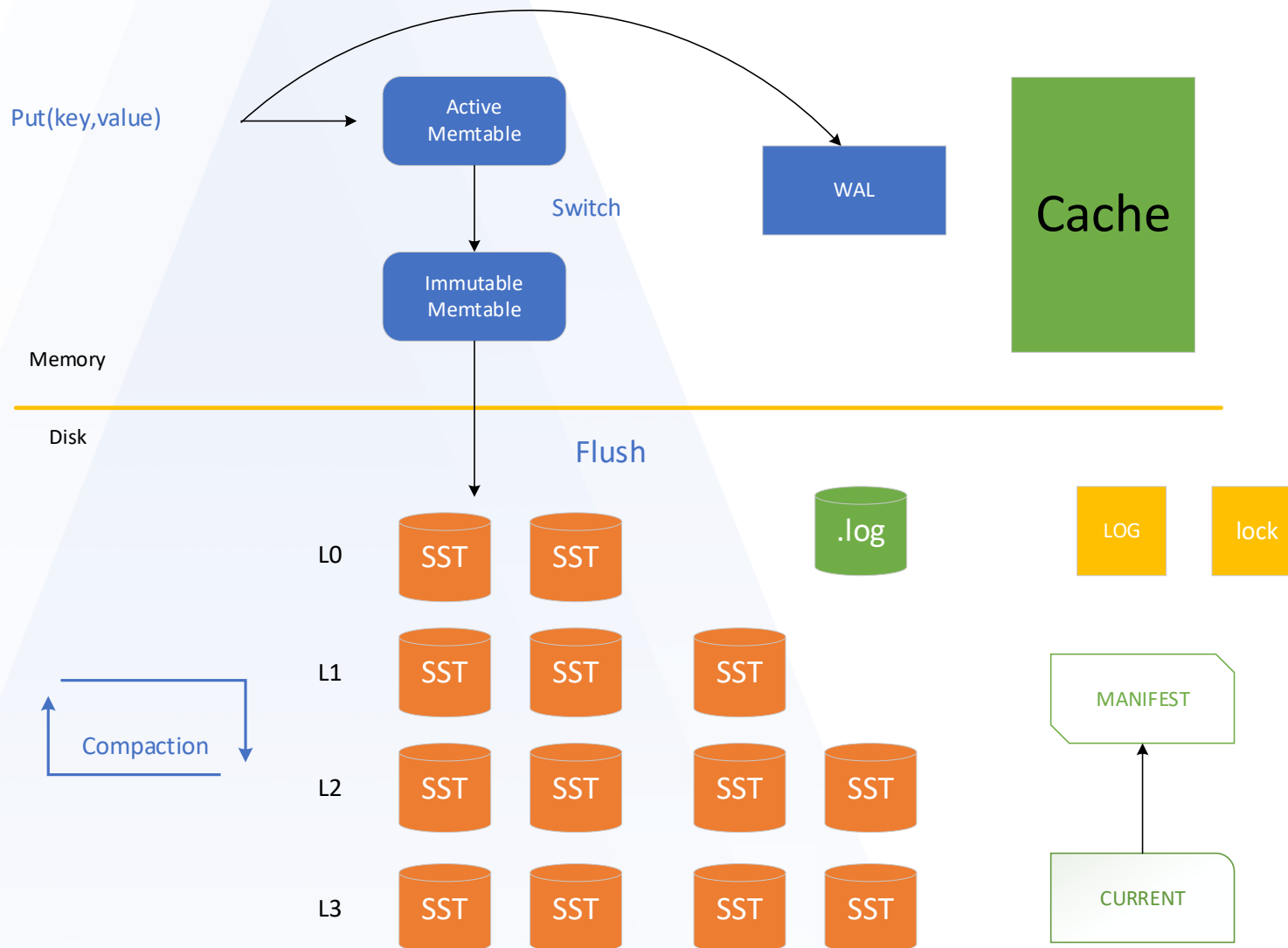
Log-structured merge tree

➤ 写路径

- WAL
- memtable

➤ Memtable 落盘后形成SST文件

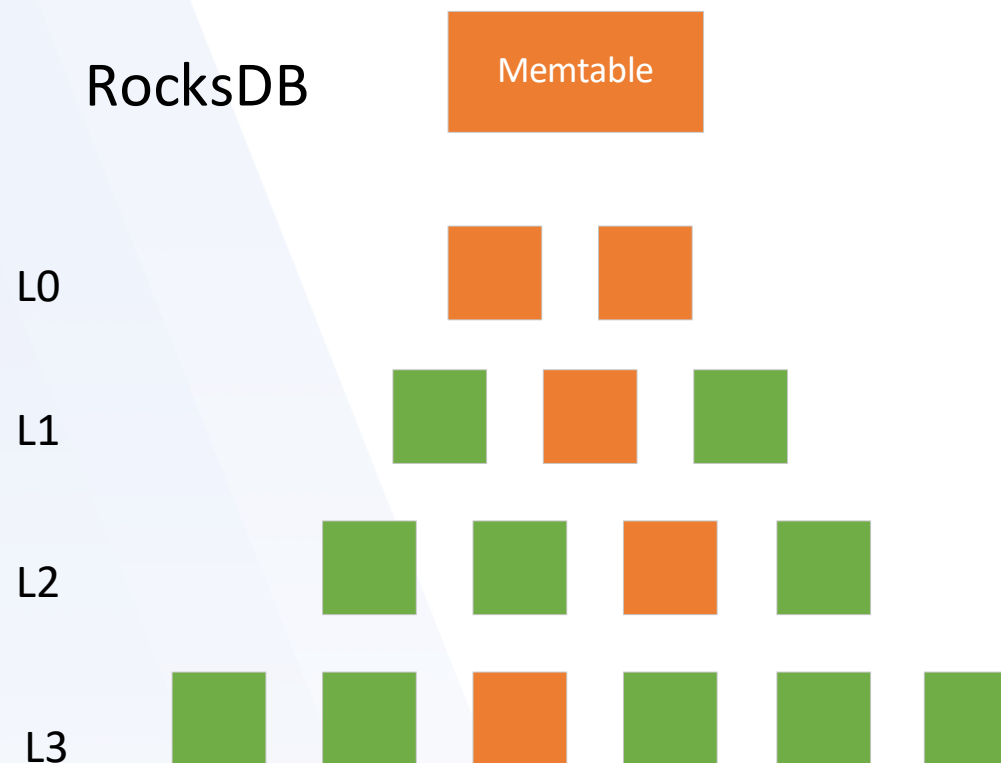
➤ SortedStringTable 不可变



Read in LSM tree

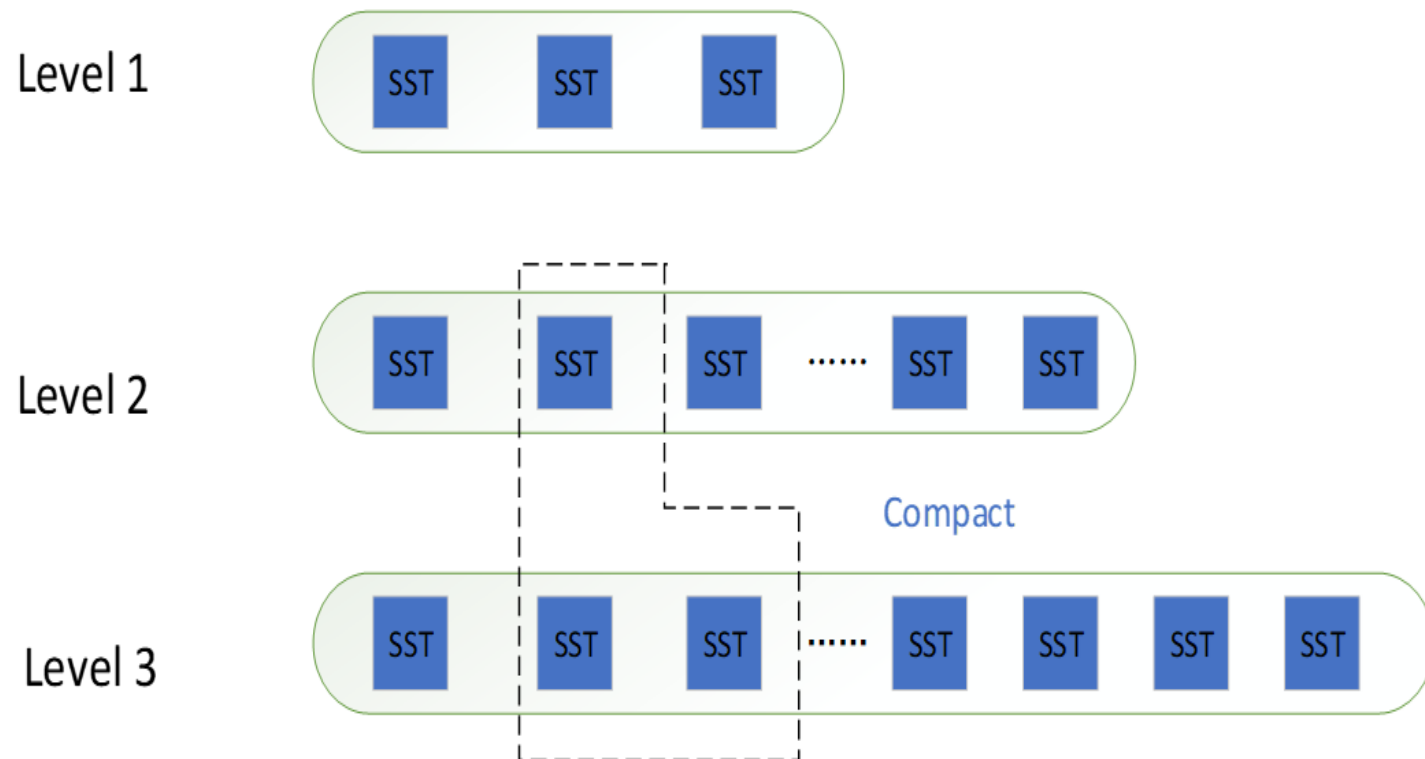
读路径

- 当前事务的 writebatch
- Active memTable
- Immutable memTable
- SSTs file L0 ~ L6



Compact

- 合并多个SSTs
- 移除已经删除的数据
- 减少SST文件数量
- 默认 Level 策略
- 写放大问题



MyRocks 特性

- Features

- RC and RR isolation level
- Row-level locking, MVCC
- WAL based crash safe
- Powerful compression
- Backup(Physical and Logical)

- Limits

- Online DDL
- Only Row-Based Replication
- No Fulltext Index
- Read Performance
- Unstable than InnoDB

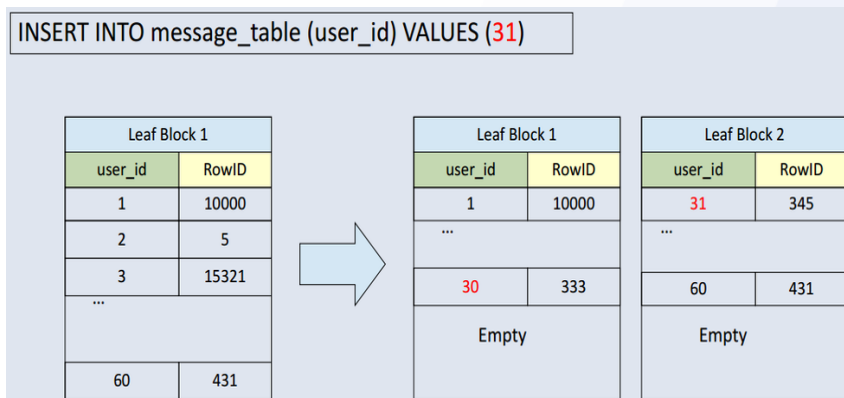
MyRocks 优点分析

D I G I T A L S A I L

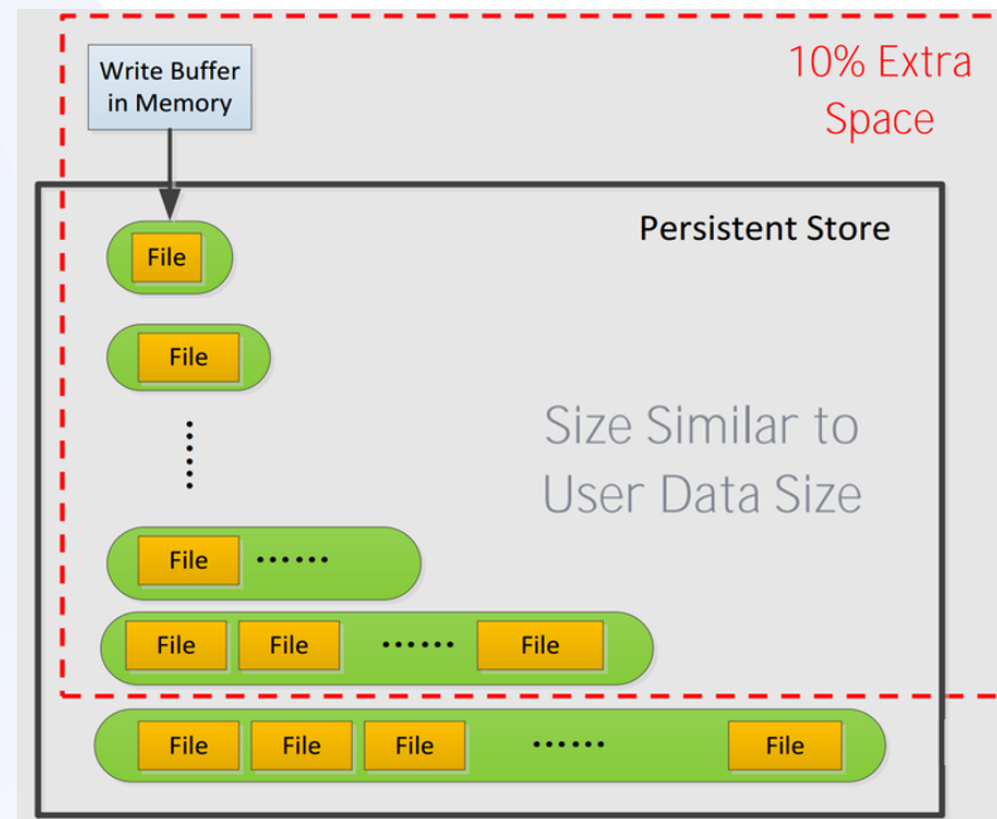
网 易 数 帆 旗 下



- 存储效率高，写性能好
 - InnoDB页填充率低
 - RocksDB不存在页内碎片
 - InnoDB 随机写 vs RocksDB顺序写




InnoDB页空间利用率不高，默认阈值15/16，随机写时碎片严重



RocksDB基于Append-only机制，空间利用率高，默认情况下仅有约10%空间放大

Prefix Key Encoding

id1	id2	id3
100	200	1
100	200	2
100	200	3
100	200	4




id1	id2	id3
100	200	1
		2
		3
		4

存储空间较小

- RocksDB记录可进行前缀编码，默认每16条记录才有一条完整的
- RocksDB每个索引占用7+1 bytes，看似比InnoDB多（每记录6+7 bytes），但Ln SST文件seq id可置0

Zero-Filling metadata

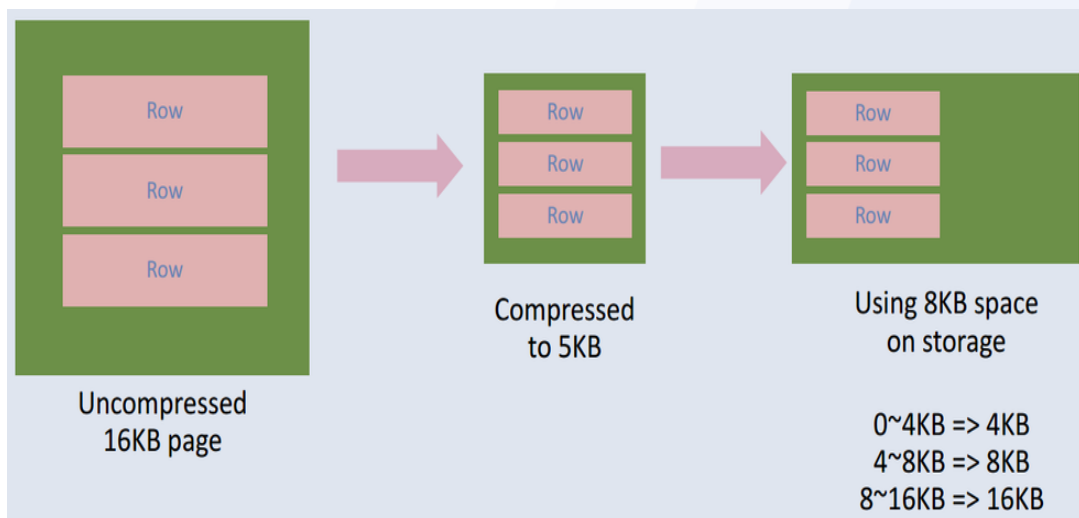
key	value	seq id	flag
k1	v1	1234561	W
k2	v2	1234562	W
k3	v3	1234563	W
k4	v4	1234564	W



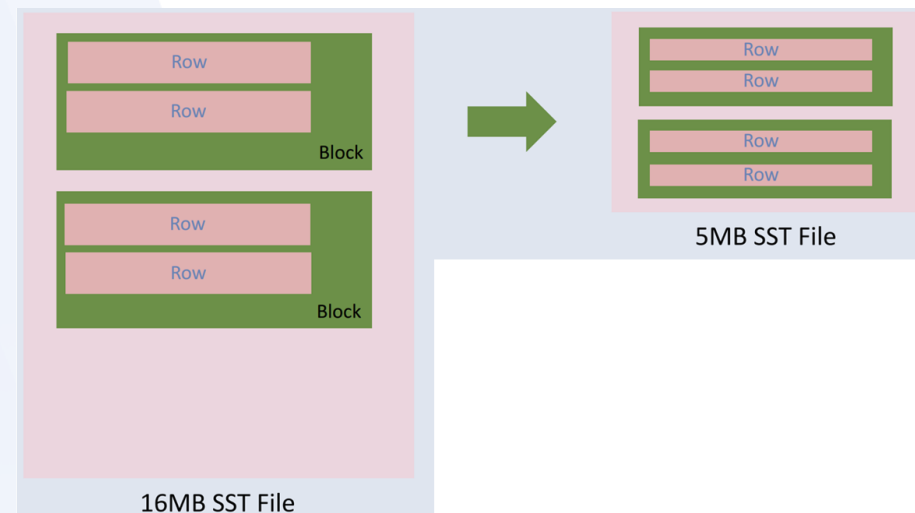
key	value	seq id	flag
k1	v1	0	W
k2	v2	0	W
k3	v3	0	W
k4	v4	0	W

- 压缩效率更高

- InnoDB Page压缩后需要文件块对齐
- RocksDB压缩只需SST File级别对齐

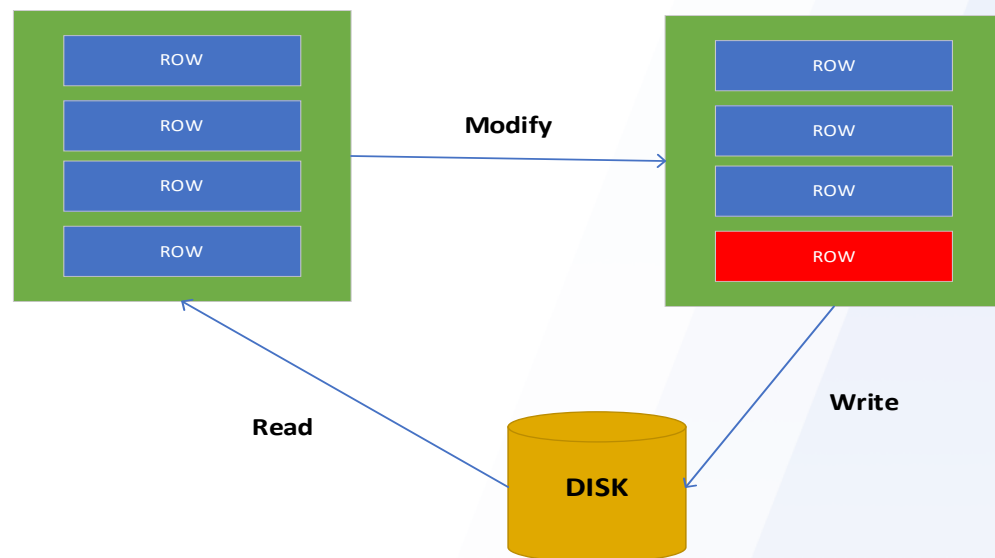


InnoDB支持记录级、页级压缩，压缩后均需文件系统块对齐。

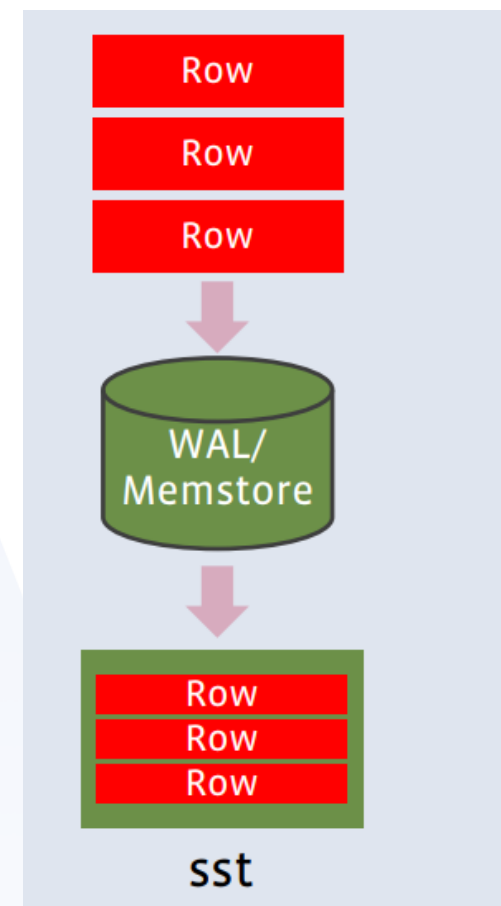


RocksDB虽也是块压缩，但压缩后无需文件块大小对齐，只需要整个SST文件对齐即可

写放大



- ✓ 修改1个byte导致写整个数据页(默认16KB)
- ✓ innodb doublewrite 再写一遍数据页



RocksDB 追加写，更小的写放大

MyRocks 在网易实践

D I G I T A L S A I L

网 易 数 帆 旗 下



案例1-网易号业务线

● 业务属性及痛点

- 1、大部分时间QPS 很低，大多是等值查询
- 2、每天定时批量导入数据
- 3、磁盘接近90%, RDS 实例规格为2TB已无法扩容

● 使用 MyRocks 后

1. 单实例数据量缩减近70%。
2. 每天定时导入数据时间由100分钟降低到45分钟

案例2-云音乐实时推荐

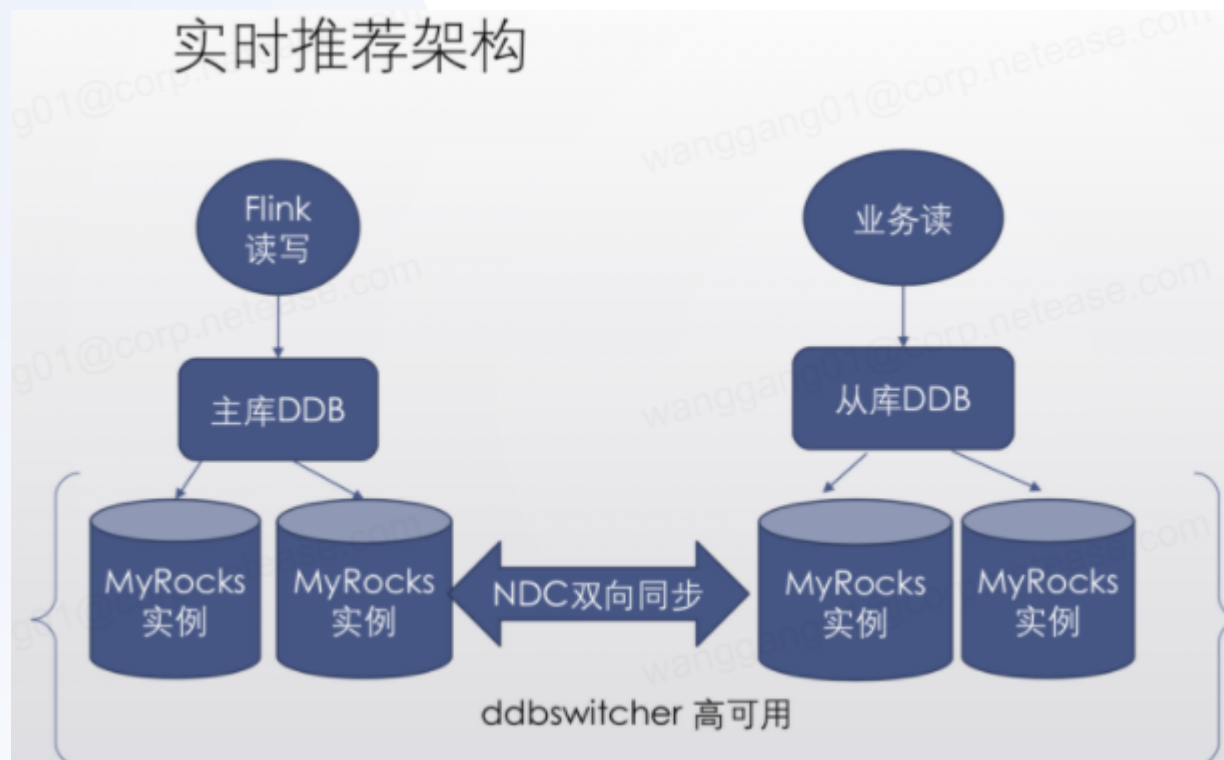
业务属性及痛点

- 推荐类业务，写多读多
- 原为离线推荐，使用Redis缓存
- 10w+算法读写tps，4w+业务读qps
- 升级为实时推荐后，数据量变大
- 仍用Redis，成本会急剧升高

临时方案 - 业务双写

- 难题：主从复制架构无法满足性能要求
- 契机：业务对数据一致性要求不高
- 临时方案：业务层双写

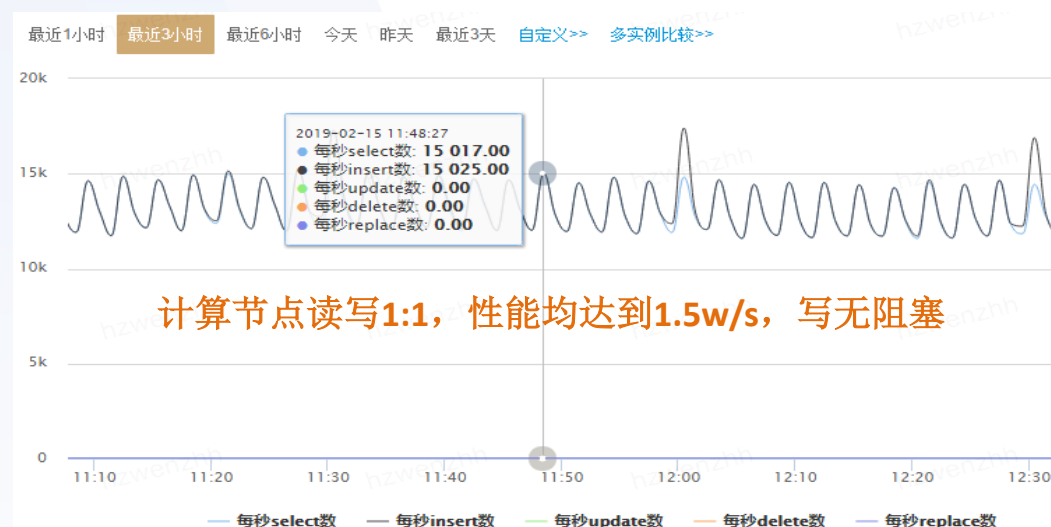
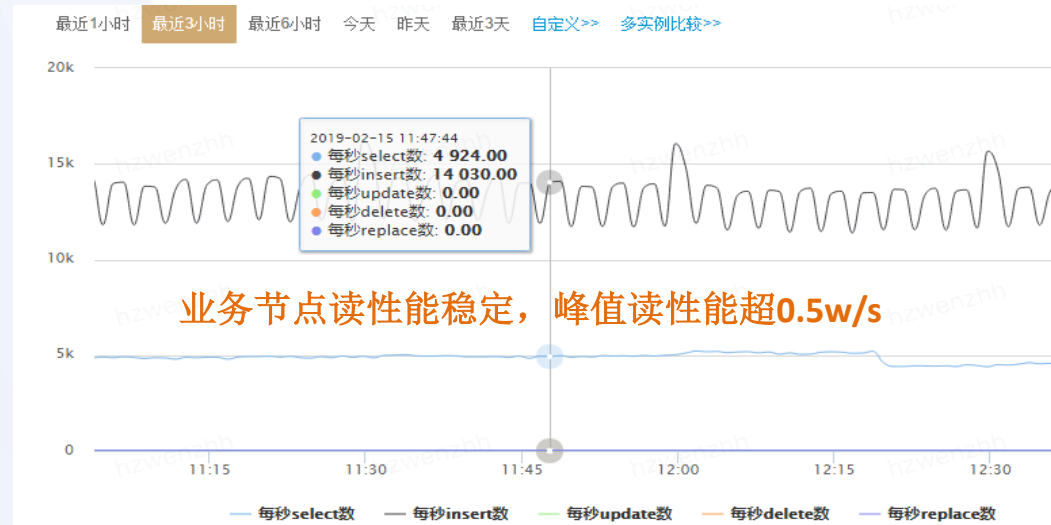
目前的方案



案例2-云音乐实时推荐

➤推荐系统使用效果和收益

- ✓ 业务读延迟低（2ms以下），波动小
- ✓ 缓存热点明显，扩展性较好
- ✓ 基于lz4压缩，减少50%+ SSD空间
- ✓ 节省了硬件投入成本



案例3-延迟从库

- 业务核心库：数据误删或损坏代价大
- MyRocks：
 - ①占存储空间小，压缩效率高
 - ②部署成本低
- 无负载：
 - ①延迟从仅回放Binlog，
 - ②无读操作，资源利用少
- MySQL提供了延迟复制功能

问题与挑战 – XA事务

- 现状：使用DDB分库分表，有大量XA事务
- 问题1：MyRocks不支持回放XA事务
`replace_native_transaction_in_thd`
- 问题2：如何将数据从InnoDB迁到RocksDB
`NDC 数据传输工具`
- 问题3：如何复制回放建表操作
`脚本转换`

案例3-延迟从库

➤使用效果

- ✓ 单实例数据量减少70%
- ✓ 单实例消耗内存 < 8G, CPU开销小
- ✓ 每台物理服务器可部署30+个实例
- ✓ 成本可控, 收益比较大

```
mysql> select sum(DATA_SIZE) from information_schema.ROCKSDB_INDEX
_FILE_MAP;
+-----+
| sum(DATA_SIZE) |
+-----+
| 959415012475 |
+-----+
1 row in set (0.06 sec)

mysql> exit
Bye
ddb@db- 34:~/mysql$ ls -lh ./node-1/.rocksdb/ | grep total
total 284G
ddb@db- 34:~/mysql$
```

MyRocks 其他落地

- ✓ 云音乐听歌记录
- ✓ 网易传媒阿波罗业务
- ✓ 网易视频历史库
- ✓ 新闻头条推荐

MyRocks 调优实践

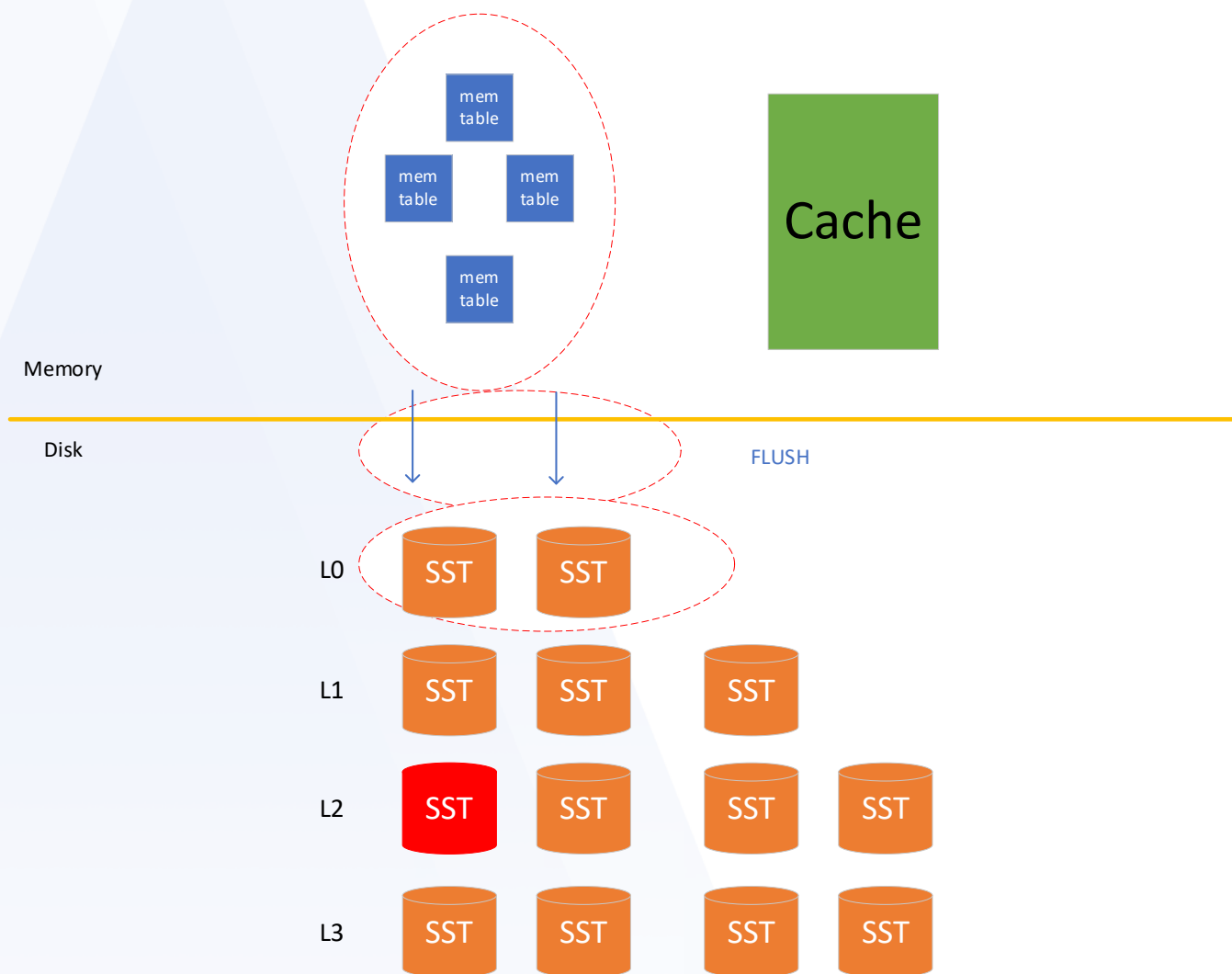
D I G I T A L S A I L

网 易 数 帆 旗 下



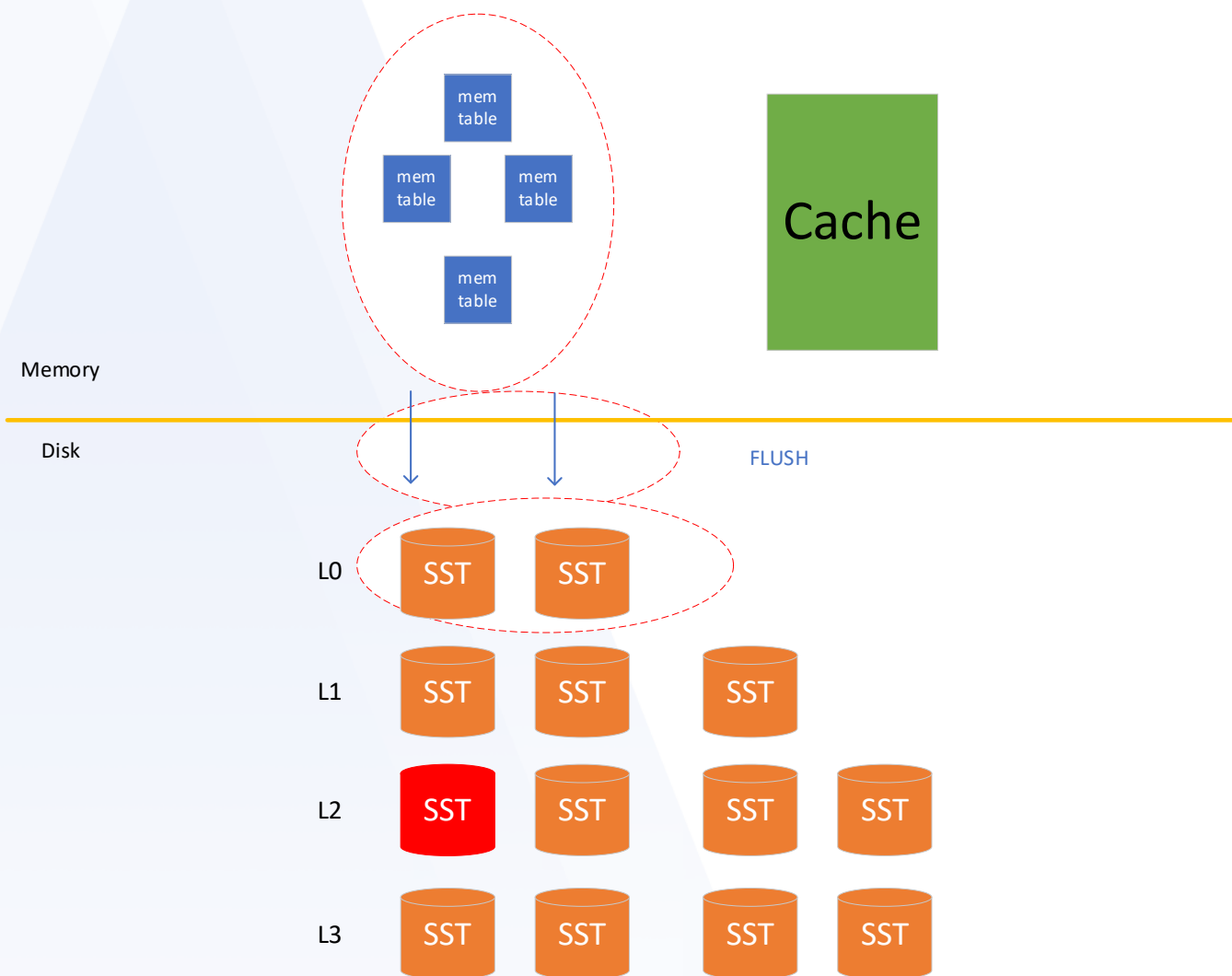
MyRocks 调优

- 内存中 memtable size 及 count
- Cache 大小
- Flush及Compact 线程数量
- 大事务问题



MyRocks 调优

- Level0 层 SST 文件数量
- Compression 算法
- Compact pending bytes



THANK YOU

D I G I T A L S A I L



扫码即可关注