Tencent 腾讯

后台开发通道面试陈述

申请晋升职级：T10

申报人：梁承希（cyrilliang）

来自：视频产品技术部/媒资产品中心/媒资短视频应用组

时间：2020/3/19

梁承希（ cyrilliang ）

项目经历

* 2009/09---2016/03 西安电子科技大学 硕士

* 2016/03---2017/12 腾讯视频-媒资平台组

  媒资基础资料开发管理

  乘风-内容抓取系统、地域播控系统、游戏推荐项目等

* 2018/12---2019/1 腾讯视频-速看后台开发组

  速看 APP 后台服务开发

  速看推荐系统引擎搭建&开发

* 2019/01---至今 腾讯视频-媒资短视频应用组

  短视频业务开发

  媒资重构项目

考核情况

最近一年考核 四星+四星

五星两次 四星三次 三星两次

评审项目

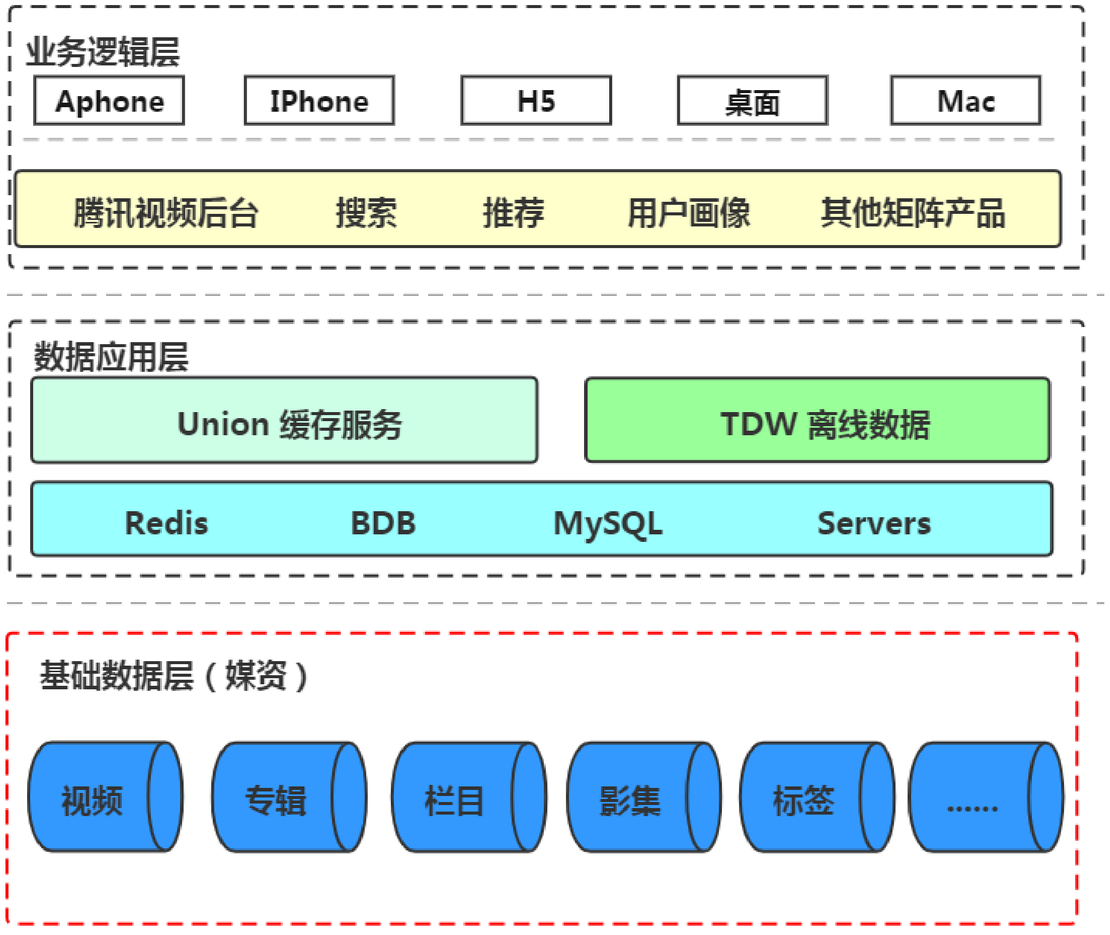
《腾讯视频媒资重构》 第一负责人

目录

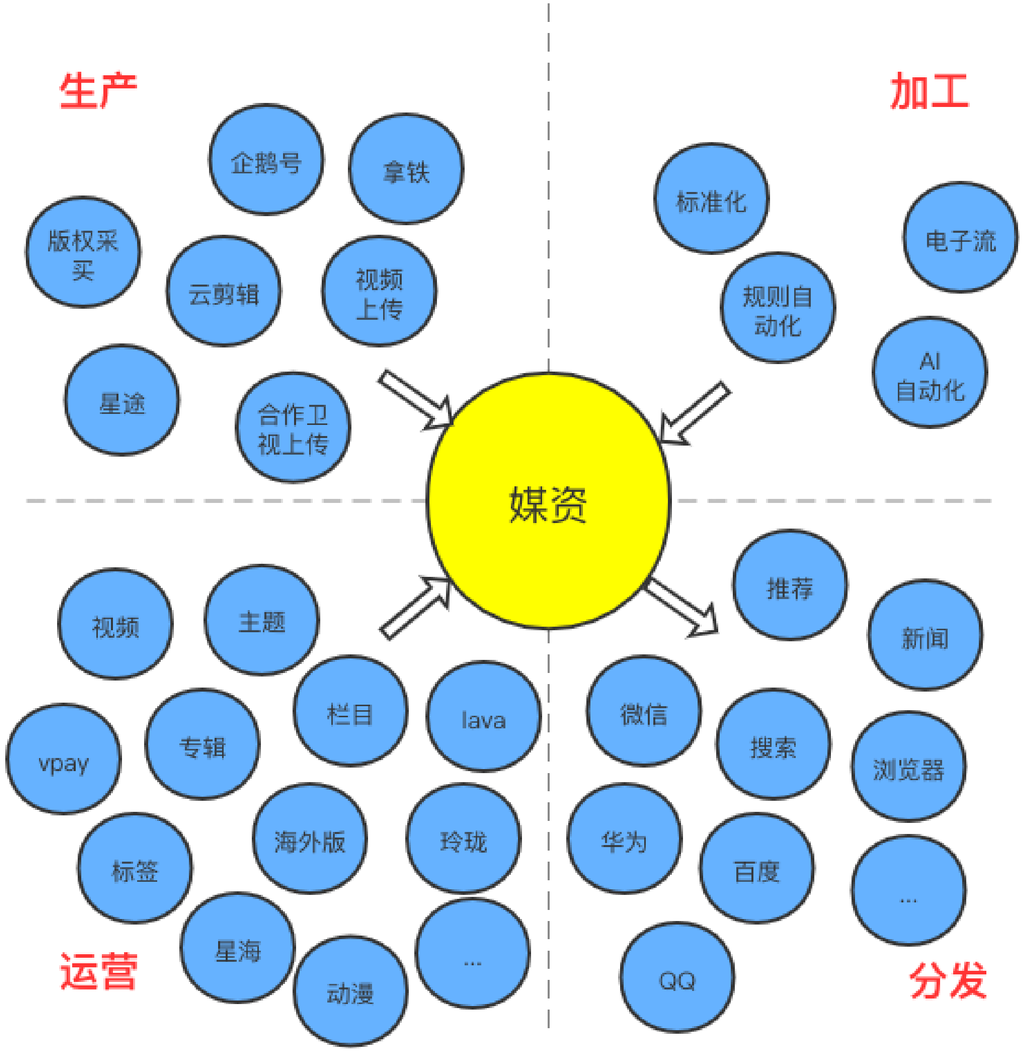
1 媒资重构项目概述

媒资是什么

腾讯视频架构



『 媒资平台承载内容的生产与分发，为频道运营提供最重要的基础运营系统，为各条业务线输送最基础最核心的媒资数据。』



重构背景

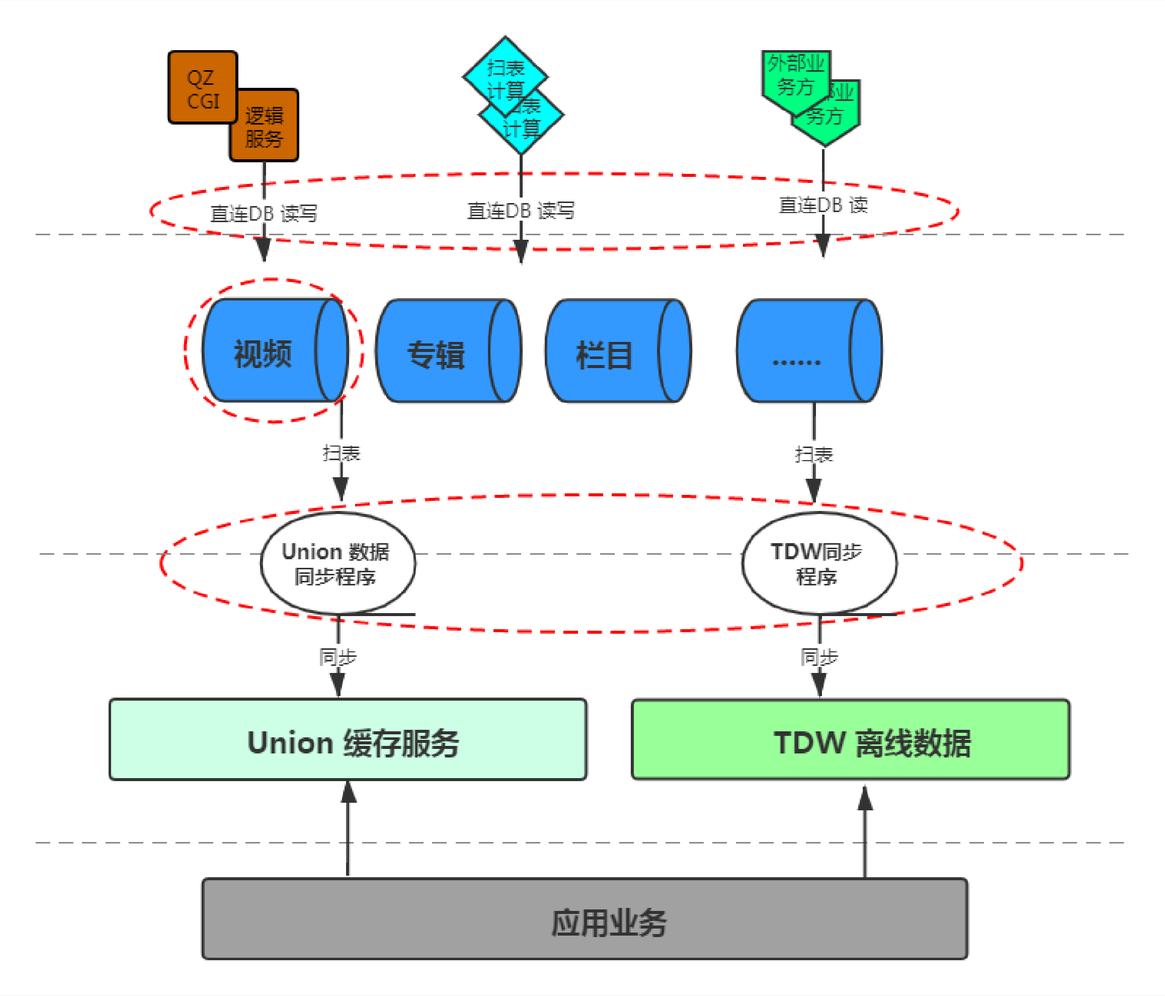
业务调整	存储数据容量告急	性能问题严峻	系统间耦合严重	系统可维护性差
16年起，开始发力短视频业务，19年起，开始进行向综合视频平台转型（netflex+youtube） 15 年前，vid日增量10W以内 17 年日增30W，19年日增80W 整体的架构不满足发展要求	视频旧DB容量已经使用 95%，无法直接扩容 离爆库还有三个月的时间 超过2T的实例无法直接迁移	数据修改接口平均耗时百毫秒左右，5%请求超过 1s 消息更新量已经达到 union 同步程序性能瓶颈 TDW 同步程序同步延迟 5~10 小时，且不稳定	扫表逻辑繁多，达十数个 CGI 直接访问 DB，数量数十个 直接暴露 DB 从库给业务方使用	烟囱式开发，代码冗余，特殊逻辑多，抽象性差 实现方式五花八门，数据访问不收敛 基础技术老旧，老框架/平台 不再维护

挑战

时间有限　　人力较紧张　　迁移数据规模大　　事故影响面广

业务逻辑复杂　　保证上游无感知　　涉及核心业务程序　　性能提升要求高

项目拆解



数据层重构

- 数据接入层设计&开发
- 底层数据存储选型
- 存储容量模型评估

核心同步程序优化

- Union 同步程序优化
- TDW 同步程序优化

逻辑服务改造

- 逻辑写点收敛
- 主写点迁移

数据迁移

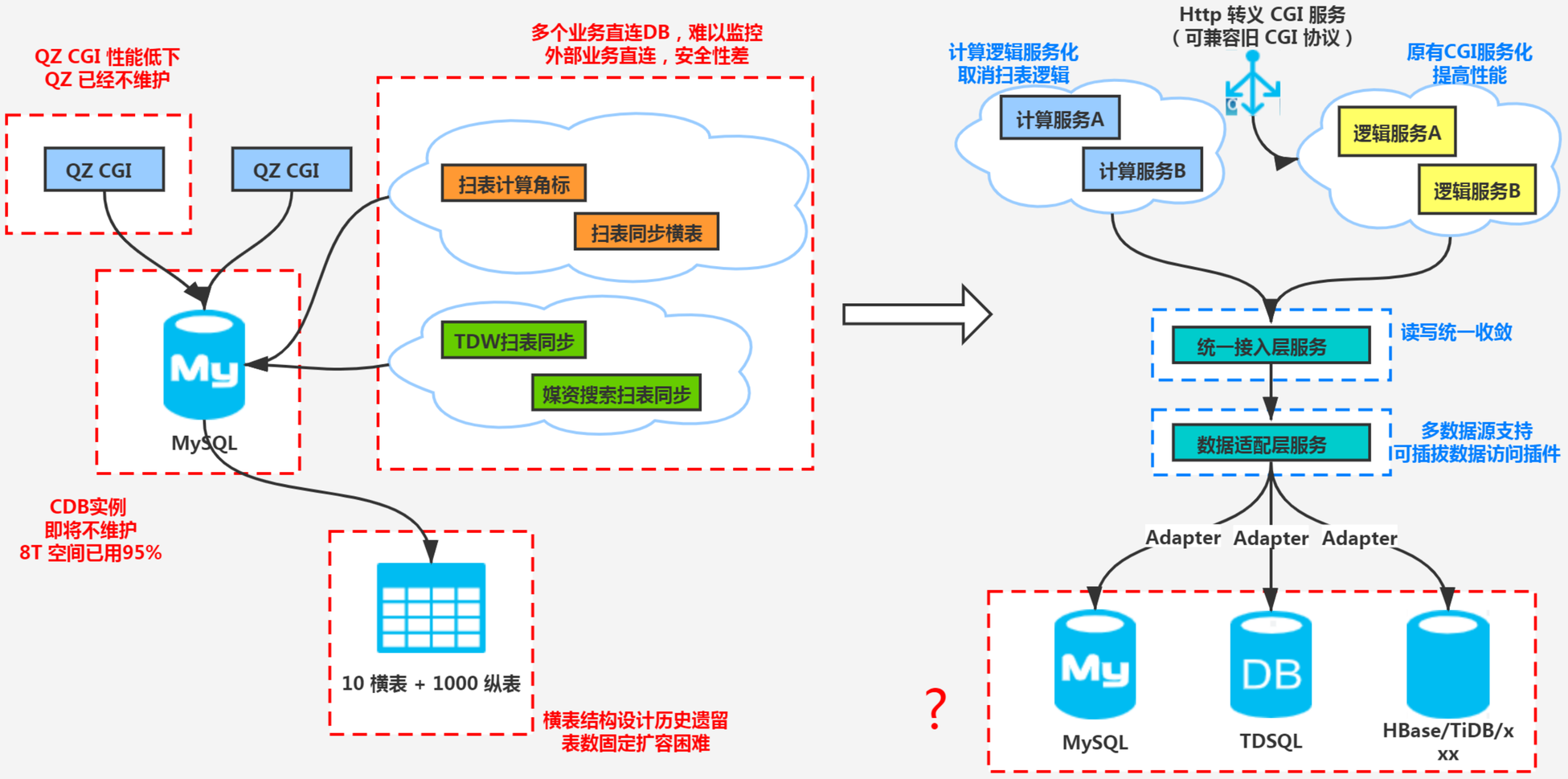
- 媒资精品数据迁移
- 媒资 vv 数据迁移

接入业务逻辑收敛

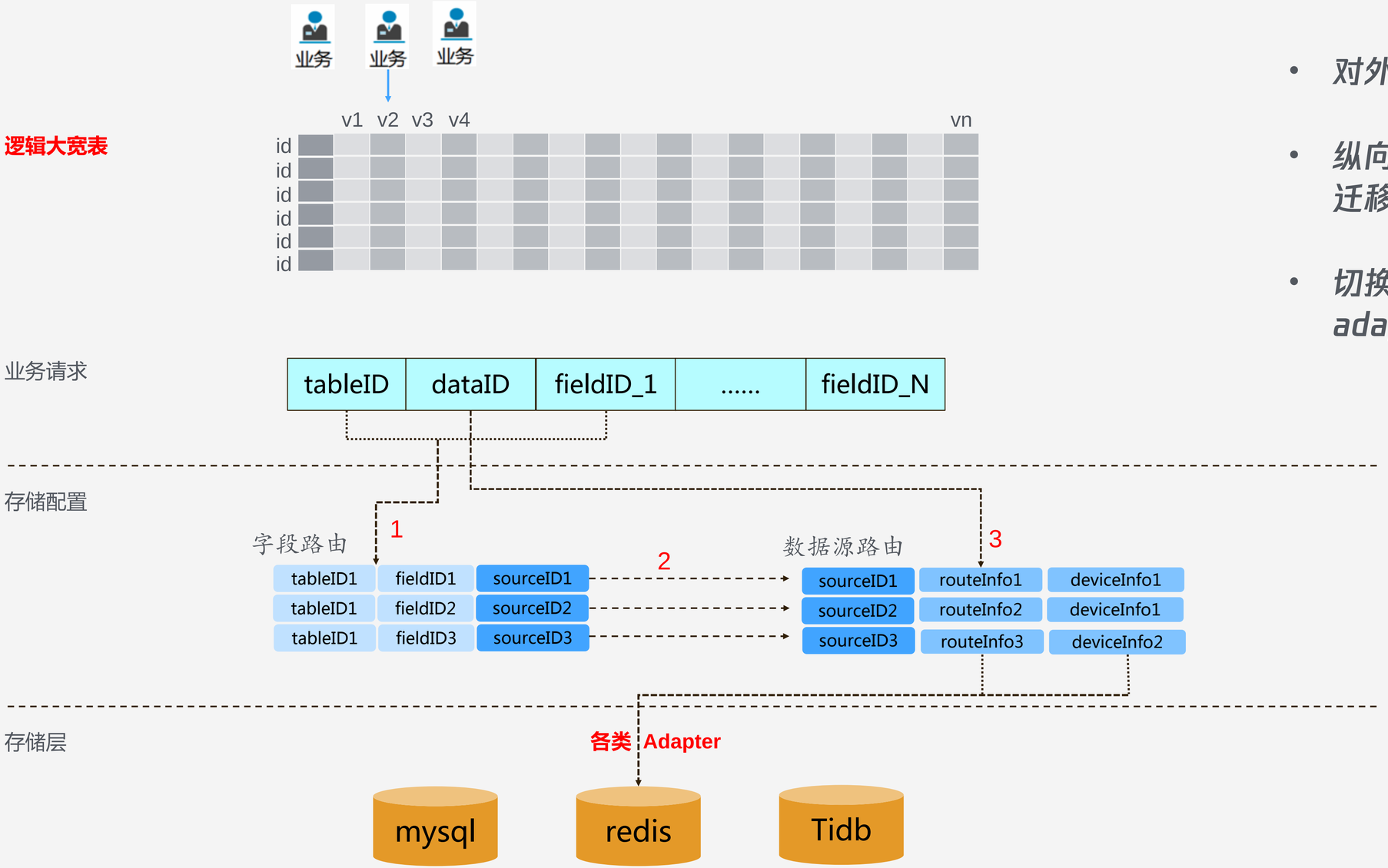
- 数据总线服务
- 媒资统一数据平台

2 难点：接入层与数据存储重构

接入层重构


QZ CGI 性能低下
QZ 已经不维护
QZ CGI
QZ CGI
MySQL
CDB实例
即将不维护
8T 空间已用95%
多个业务直连DB，难以监控
外部业务直连，安全性差
扫表计算角标
扫表同步横表
TDW扫表同步
媒资搜索扫表同步
10 横表 + 1000 纵表
横表结构设计历史遗留
表数固定扩容困难
Http 转义 CGI 服务
（可兼容旧 CGI 协议）
计算逻辑服务化
取消扫表逻辑
计算服务A
计算服务B
原有CGI服务化
提高性能
逻辑服务A
逻辑服务B
统一接入层服务
读写统一收敛
数据适配层服务
多数据源支持
可插拔数据访问插件
Adapter
Adapter
Adapter
?
MySQL
TDSQL
HBase/TiDB/x
xx

2 海量数据存储方案



- 对外表现是一张大横表
- 纵向按列拆分数据，便于扩展和迁移
- 切换底层数据存储，开发新的adapter即可，上游不感知

底层数据存储选型

存储	数据扩展能力	支持存储容量	写性能	读性能	运维支持度	事务支持度	业界使用案例
单实例MySQL	低	低	低	低	高	支持	-
MongoDB	高	中	中	中	低	支持	58同城
Cassandra	高	高	高	中	低	部分支持	Netflix
HBase	高	高	高	中	低	部分支持	腾讯看点
TDSQL分布式	高	高	高	高	高	支持	微众银行
TiDB	高	高	高	高	中	支持	知乎/头条

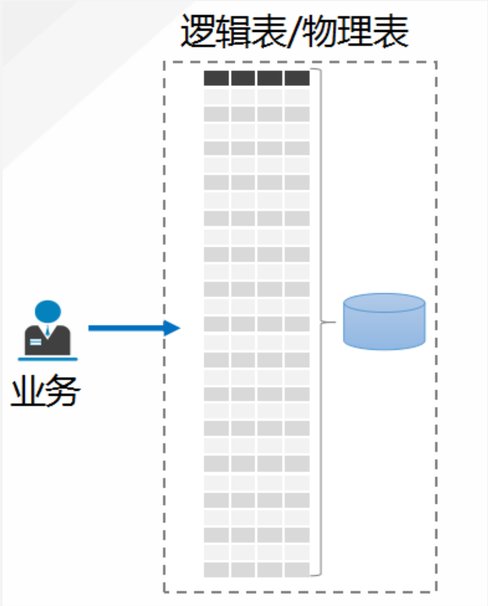
核心精品视频数据 → TDSQL 分布式（MDB 已支持）

非核心视频数据 → TiDB（MDB 已支持）

媒资诉求

- 性能要求 -> 高
  - 写多读多，万级QPS;
- 安全性要求 -> 高
  - 影响外网各个平台外显;
  - 影响视频播放;
- 可维护性可扩展性 -> 高
  - 数据膨胀速度快，需要支持快速扩容能力;
- 业务诉求
  - 媒资一些业务数据需要支持事务;
  - 尽可能兼容SQL，降低维护和迁移成本

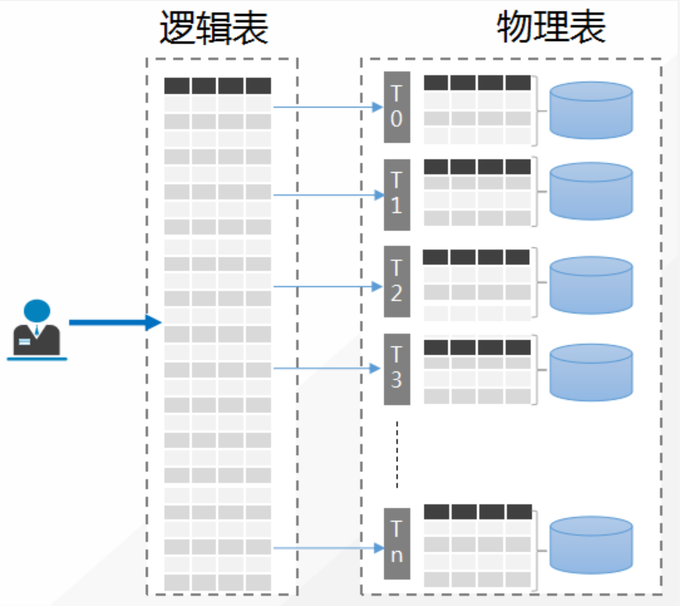
旧MySQL实例



- 10横表+1000纵表结构
- 4亿数据，已快撑满

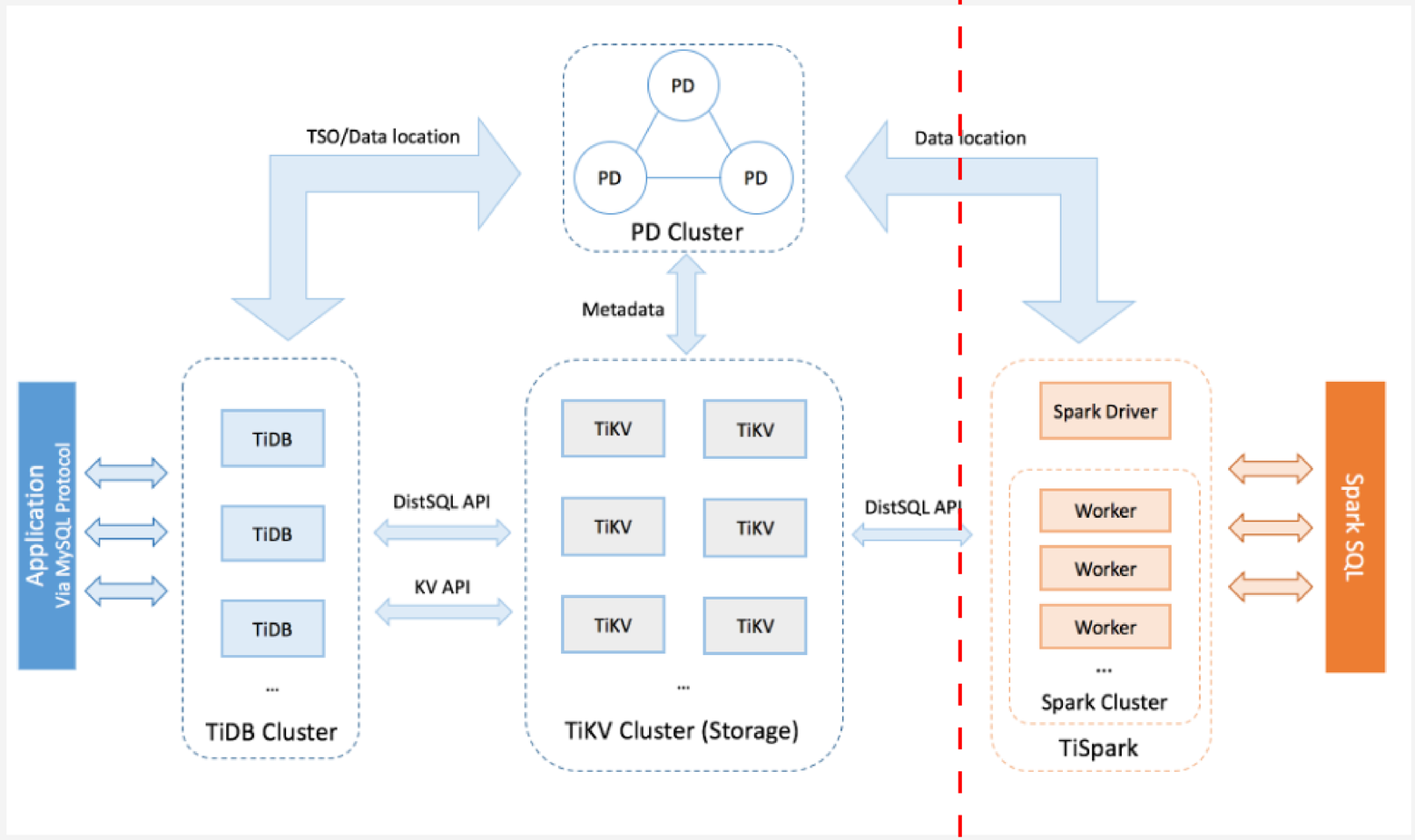
- CDB不再维护
- 需手动迁移、扩容实例

TDSQL 分布式



- MDB 管理平台统一运营
- 自动扩容工具，业务无感知
- 运维建议+业务场景 2库每库64表
- 读写性能 10W QPS 左右
- TP95 5ms

TiDB：NewSQL

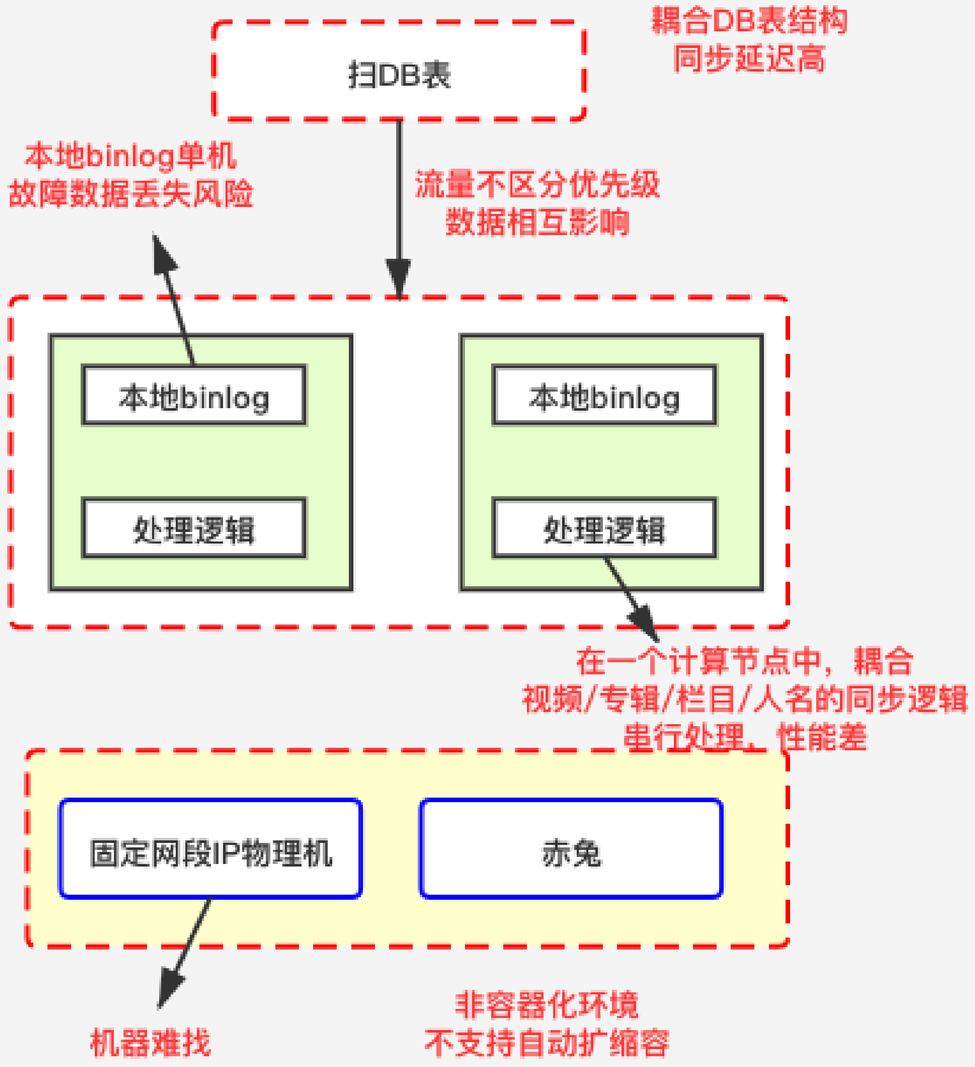


- **TiDB Server**
  - 无状态，平行扩容
- **PD Server**
  - 管理数据集群；
  - Raft 协议保证数据安全高可用；
- **TiKV Server**
  - 数据存储 Key-Value 存储引擎；

单表存储，无需进行再分表 → 应用于20+亿的 VV 视频库 & 下一阶段存储

3 难点：媒资数据同步优化

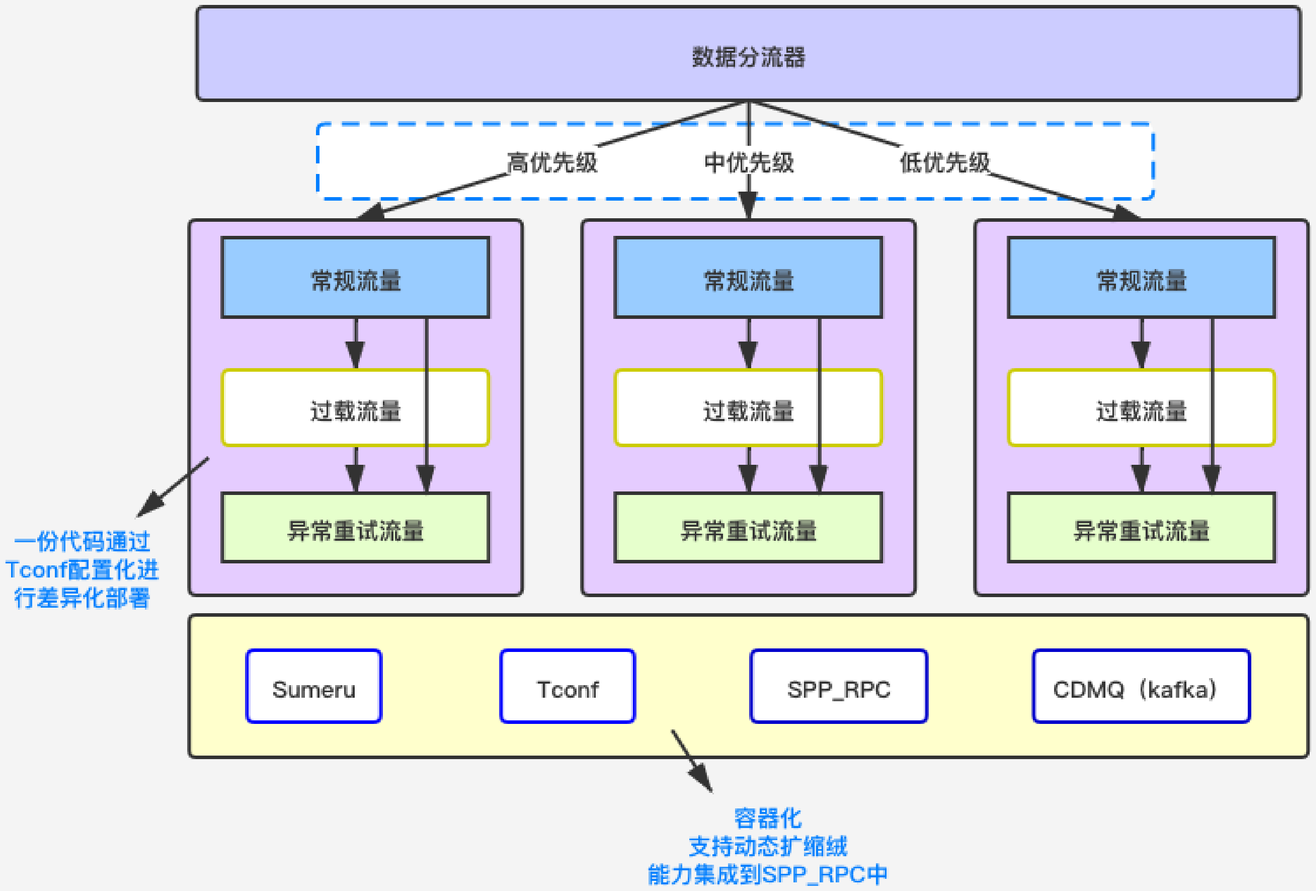
旧同步程序逻辑



业务场景诉求

- 同步要尽可能实时
- 数据不能丢失
- 性能要足够高
- 不应当产生积压导致数据发生同步延迟
- 重点剧重要程度高

新同步程序逻辑



优化点

- 轻重分离，重点剧重点照顾
- 队列分流制，避免异常流量导致数据阻塞
- 充分异步并发，提高单机性能
- 更新数据存储在 kafka，避免单机故障

MORE

- 支持多种队列 kafka/hippo&多种应用特性
- 开源协同，能力集成到部门开发框架SPP_RPC中，给兄弟团队提供使用（媒资/红骑push/视频推荐）
- 配置化实现场景定制，无需修改代码

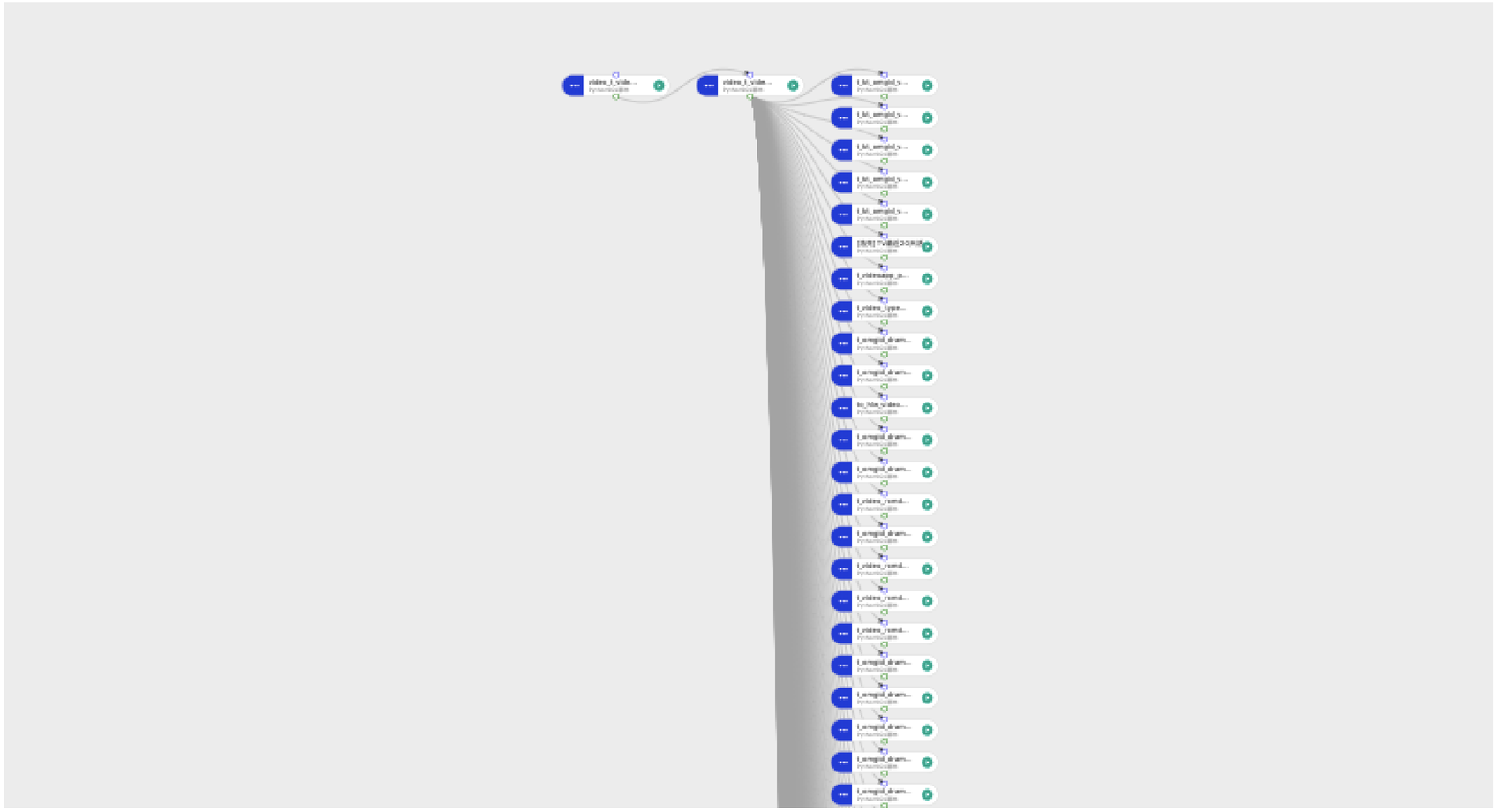
优化后指标对比

比对项目	旧同步服务	新同步服务	
单机并发性能	200 QPS	2000 QPS	
最大支持并发数	3000 QPS	20000 QPS+	
平均同步延迟	1min 左右	亚秒级	
可维护性	低	高	

* 新服务最大并发数受限于下游 Union 写服务最大支持量
* 旧服务已经有三年未进行过代码变更和发布

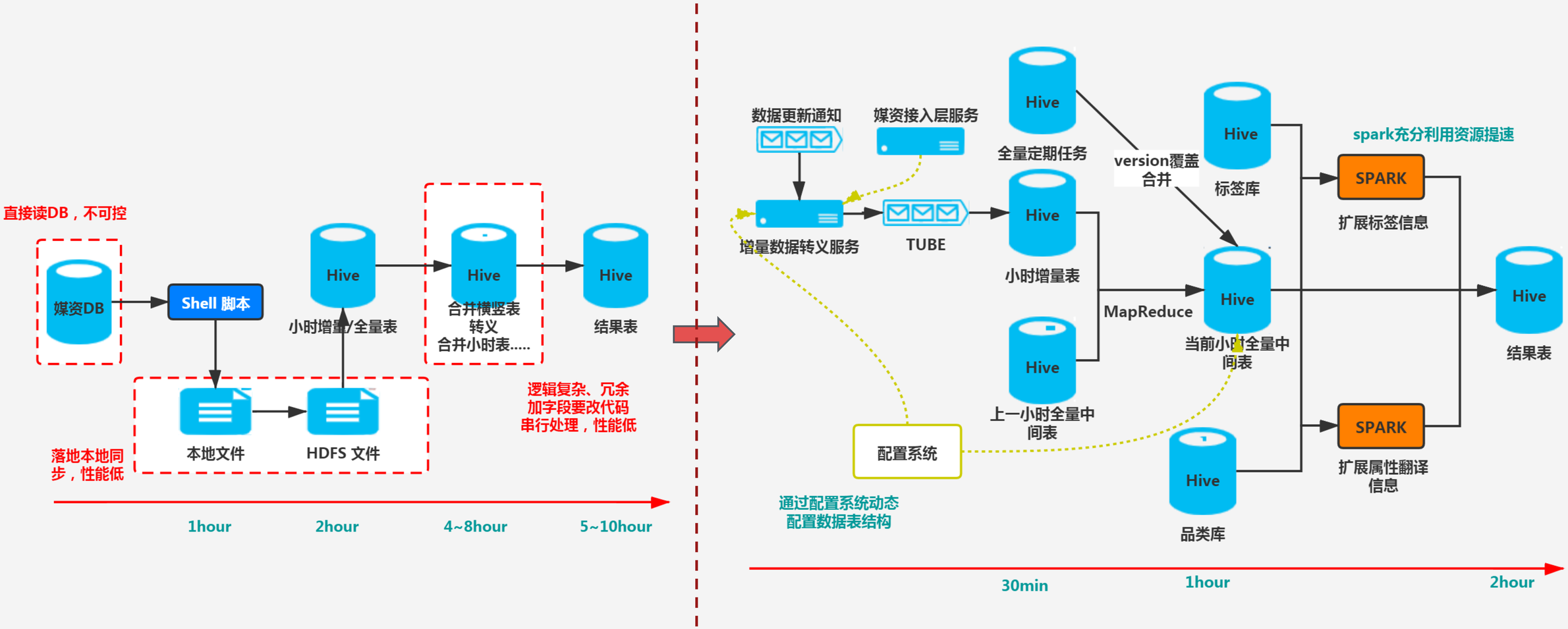
背景

查看父子关联



- 依赖业务方有数百个
- 耗时要求高，要小时表
- 数据同步量大，计算量大
- 维护成本高，经常加字段

同步新旧方案对比



- 性能提升：同步耗时 5~10 小时 缩减到 2小时内（中间表1小时左右）
- 可维护性提升：1、字段管理配置化，无需进行代码修改；2、加字段半天缩减到5分钟；

3 媒资数据同步优化 —— TDW 数据同步

任务ID	任务名称	周期	数据时间	状态	尝试(次)	开始时间	运行时长	查看
20160629100925110	video_t_video_info_extend	小时报	2019-08-23 08:00:00		0/3	预:2019-08-23 09:40:00		父任务 日志 异常
20160629100925110	video_t_video_info_extend	小时报	2019-08-23 07:00:00		0/3	预:2019-08-23 08:40:00		父任务 日志 异常
20160629100925110	video_t_video_info_extend	小时报	2019-08-23 06:00:00		0/3	预:2019-08-23 07:40:00		父任务 日志 异常
20160629100925110	video_t_video_info_extend	小时报	2019-08-23 05:00:00		0/3	预:2019-08-23 06:40:00		父任务 日志 异常
20160629100925110	video_t_video_info_extend	小时报	2019-08-23 04:00:00		0/3	预:2019-08-23 05:40:00		父任务 日志 异常
20160629100925110	video_t_video_info_extend	小时报	2019-08-23 03:00:00		0/3	预:2019-08-23 04:40:00		父任务 日志 异常
20160629100925110	video_t_video_info_extend	小时报	2019-08-23 02:00:00		0/3	预:2019-08-23 03:40:00		父任务 日志 异常
20160629100925110	video_t_video_info_extend	小时报	2019-08-23 01:00:00		1/3	2019-08-23 09:34:04		父任务 日志 异常
20160629100925110	video_t_video_info_extend	小时报	2019-08-23 00:00:00		1/3	2019-08-23 08:58:58		父任务 日志 异常
20160629100925110	video_t_video_info_extend	小时报	2019-08-22 23:00:00		1/3	2019-08-23 08::		
20160629100925110	video_t_video_info_extend	小时报	2019-08-22 22:00:00		1/3	2019-08-23 07::		

TDW 数据延迟前后对比

任务检索 / 任务实例（任务名：video_t_video_info_extend_hour_res 媒资视频基础信息结果表，任务ID:20190814161842596）

数据时间： 2020-03-12 ~ 2020-03-14 检索

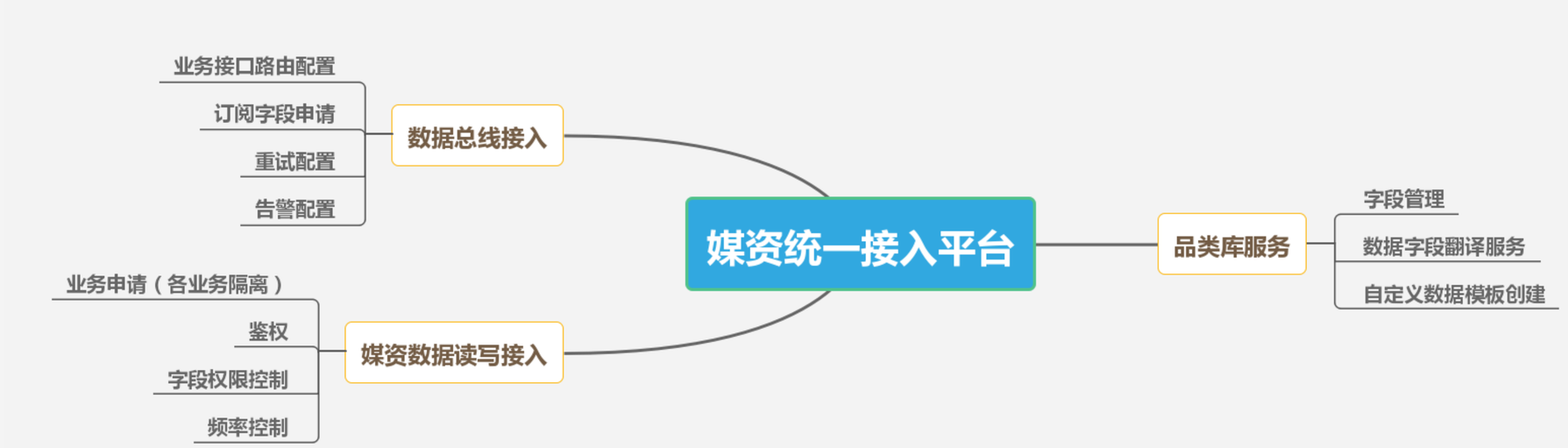
运行状态： 全部 非成功 就绪 等待父任务 运行中 成功 失败 等待终止 正在终止 终止成功 终止失败 永久终止 等待下发

实例重跑 实例终止 强制成功

数据时间	开始时间	执行时长	状态	操作人	尝试次数	操作
2020-03-14 09:00:00	--	--	等待父任务	cyrilliang;swing...	0/3	日志 父实例 强制下发 重跑 更多
2020-03-14 08:00:00	2020-03-14 10:18:09	--	正在执行	cyrilliang;swing...	1/3	日志 父实例 强制下发 重跑 更多
2020-03-14 07:00:00	2020-03-14 09:50:08	00:57:46	执行成功	cyrilliang;swing...	1/3	日志 父实例 强制下发 重跑 更多
2020-03-14 06:00:00	2020-03-14 09:14:07	00:57:00	执行成功	cyrilliang;swing...	1/3	日志 父实例 强制下发 重跑 更多
2020-03-14 05:00:00	2020-03-14 07:40:03	00:42:18	执行成功	cyrilliang;swing...	1/3	日志 父实例 强制下发 重跑 更多

4 成果与展望

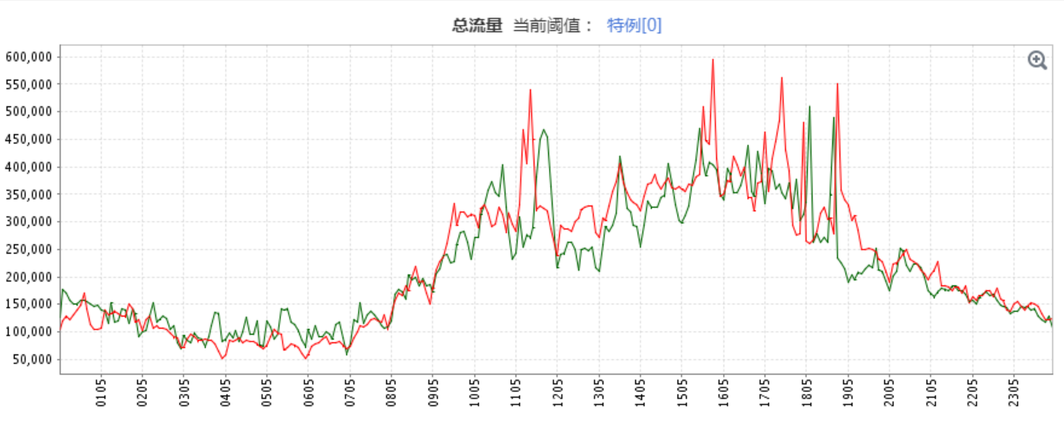
4 成果与展望



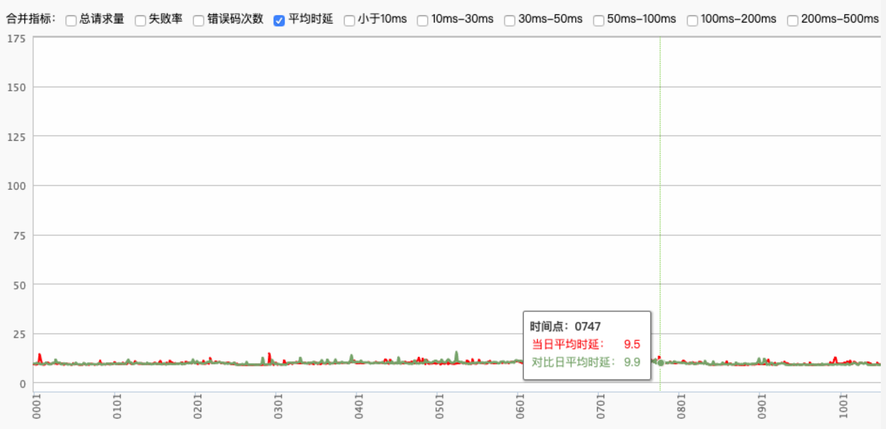
- 数据总线已接入20+个业务，外部接入服务方10+个
- 统一管理媒资数据的消息通知接入和读写接入
- 对接入业务进行鉴权、频控、调用统计

4 成果与展望

新服务 - 总请求量

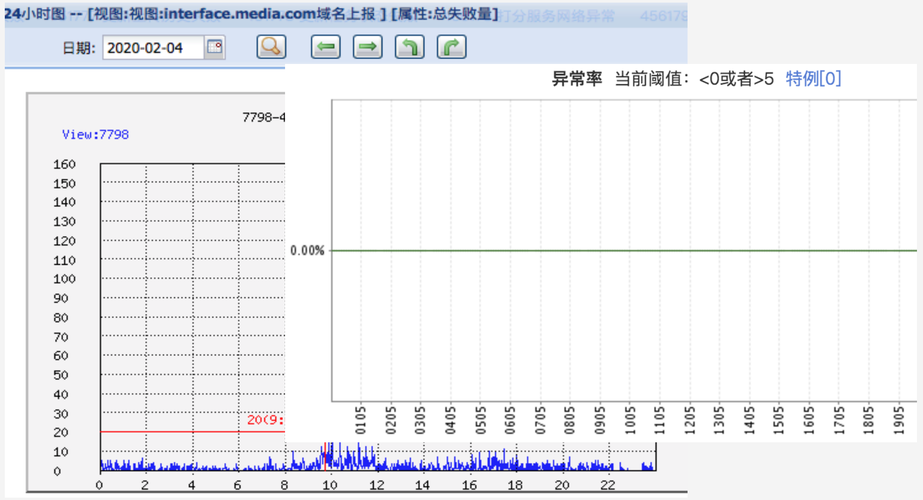


新服务 – 请求平均耗时



视频修改CGI – 平均耗时对比

视频修改CGI – 失败量对比



性能优化效果

* 接口耗时 TP95 10ms

* 数据更新接口耗时

200ms ⬇ 40ms

* 数据更新成功率由

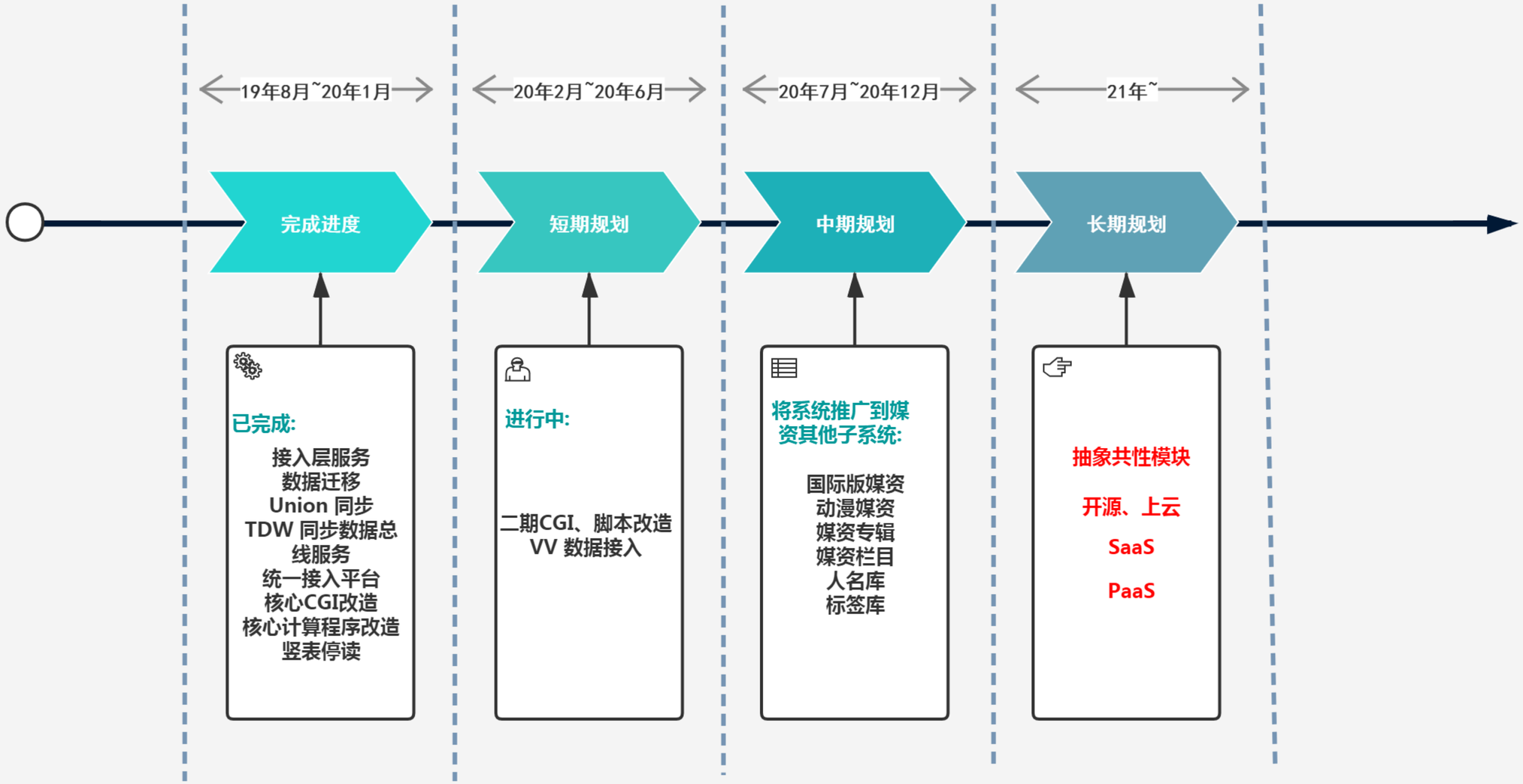
99.9% ⬆ 99.999%

* 视频详情页耗时

500ms ⬇ 20ms

项目获部门技术革新奖


技术革新奖
媒资底层数据重构团队
媒资产品中心
技术革新奖
媒资底层数据重构团队
媒资产品中心
腾讯支票
部门即时激励——优秀团队奖
人民币
(大写)
叁仟圆整
亿仟百十万仟百十元角分
¥300000
腾讯视频
技术革新奖
媒资产品中心
媒资底层数据重构团队
视频产品技术部
二零一九年第四季度


19年8月~20年1月
20年2月~20年6月
20年7月~20年12月
21年~
完成进度
短期规划
中期规划
长期规划
已完成:
接入层服务
数据迁移
Union 同步
TDW 同步数据总线服务
统一接入平台
核心CGI改造
核心计算程序改造
竖表停读
进行中:
二期CGI、脚本改造
VV 数据接入
将系统推广到媒资其他子系统:
国际版媒资
动漫媒资
媒资专辑
媒资栏目
人名库
标签库
抽象共性模块
开源、上云
SaaS
PaaS

培养&培训

- 培养人：victortang，sullivanzhu，roysun，swingszhang，joyyhe
- 部门课程：乘风系统介绍
- 部门课程：从0到1搭建推荐系统
- 部门课程：Go 语言开发培训

公共组件开发

- SPP 微线程多类型并发工具库（组）
- Python JCE RPC 客户端（中心）
- Python 通用开发工具库（中心）
- SPP_RPC 生产&消费者组件（部门）

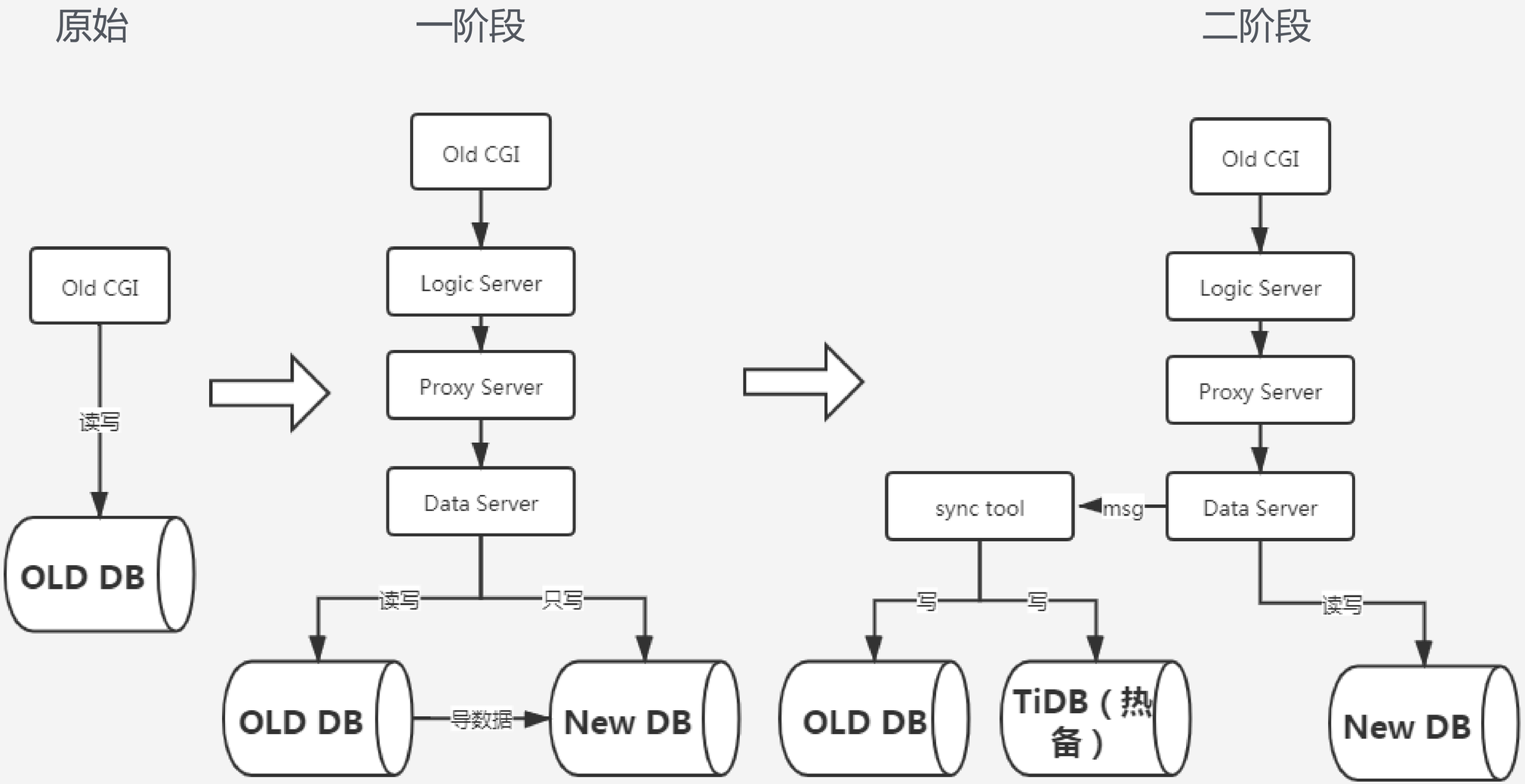
相关文章

- 《网络高性能服务的本质探究》
- 《速看推荐系统搭建》
- 《Continuaion 与 并发》
- 《函数式编程的思维方式》

Thanks

附录

数据迁移流程


原始
一阶段
二阶段
Old CGI
读写
OLD DB
Old CGI
Logic Server
Proxy Server
Data Server
读写
只写
OLD DB
导数据
New DB
Old CGI
Logic Server
Proxy Server
Data Server
sync tool
msg
写
写
读写
OLD DB
TiDB（热备）
New DB

SPP_RPC 消息消费者&生产者组件特性列表

如果你有额外的配置需求，可以在`runtime_conf`节点添加下面这些参数：

- `push_gap_time_ms` 每次推送到消费者的停留时间间隔（毫秒）以防过快消费压死下游业务,缺省为0
- `overload_strategy` 消息过载策略,可以设置的值有：
  - 0：不处理（默认值）
  - 1：直接抛弃
  - 2：转发到其他消息队列
- `overload_gap_sec` 消息过载时差判定（秒），不设置过载保护则置为0
- `overload_prod_id` 消息过载后转发的生产者id（当过载策略为2时需要，相应生产者ID对应配置文件中的“name”）
- `overload_max_gap_sec` 极限过载时间（秒）（当数据延迟超过此阈值时，无论是否设置转发队列，所有消息都将抛弃，以避免阻塞，不开启该功能则置为0）
- `err_forward_prod_id` 失败后转发的生产者id（重试后仍然失败但不希望阻塞且不希望丢弃消息，可以抛到异常队列里进行无限重试逻辑）
- `open_uniq_key_in_batch` 是否开启批次内key去重（当 queue/partition 按照key路由时，可以使用该模式保证全局同一时刻内不会处理同一个key的消息，避免冲突，目前是先来先处理逻辑，处理前提是消息带有key
- `delay_to_consume_ms` 延迟消费，如果需要延迟消费，可以配置上该配置项，即可实现收到消息后等待 xx ms 再进行处理，以满足一些缓存延迟生效需要延迟读取等场景

DB 容量存储模型

存储容量模型评估

- 一张逻辑表->128张物理表
- 单物理表最大2G
- 年复合增长量10%

- 日增量100W
- 单 VID 信息大小 18K

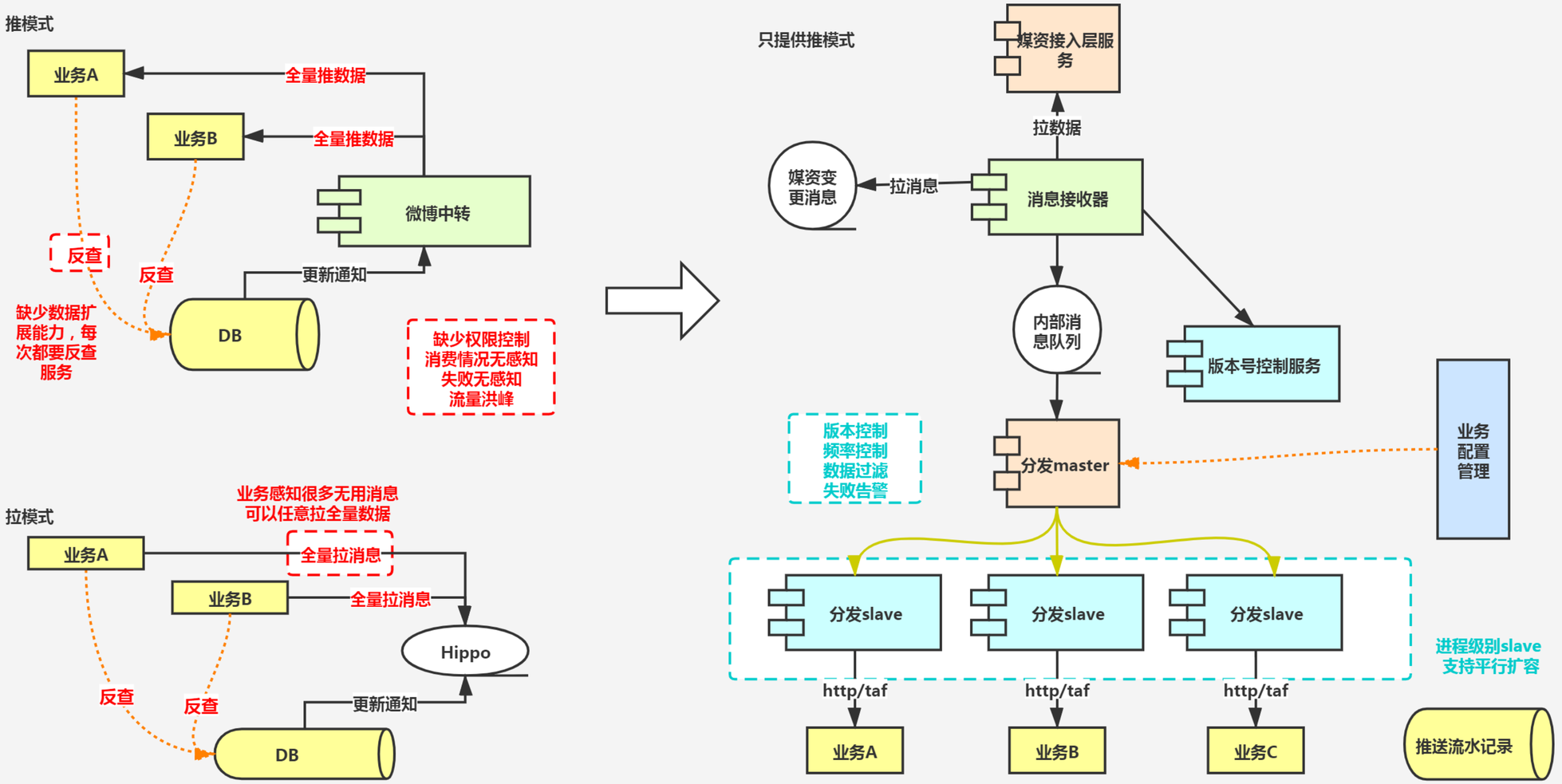
- 全纵表结构
- 未来5年，需支撑数十亿~百亿数据

2库，每库64表

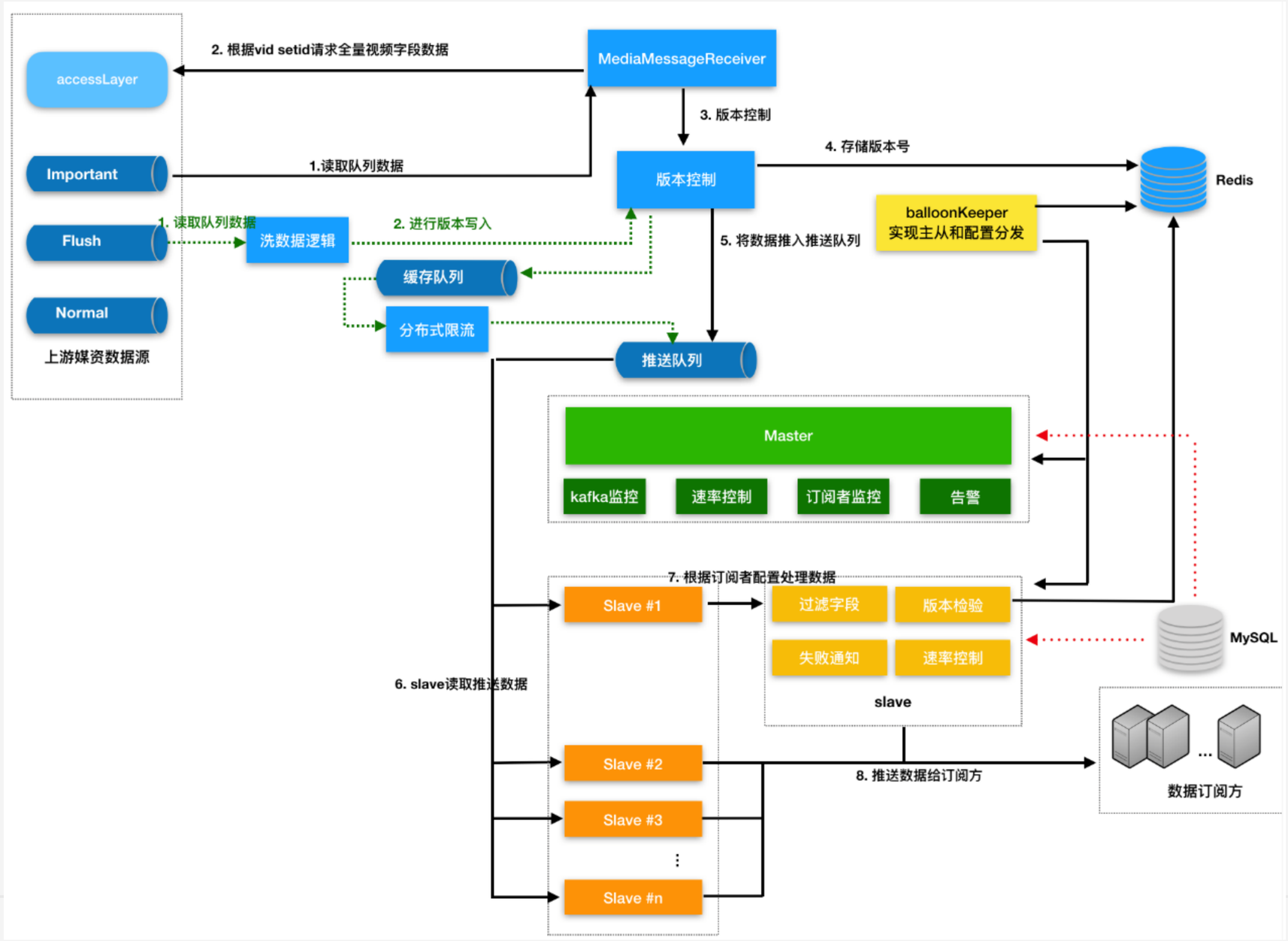
日增量占存储 = 18K*100W = 18G

所需分表数 = 365(天)*5(年)*18(G)/2(G)/128 = 128

数据总线


推模式
业务A
全量推数据
业务B
全量推数据
微博中转
反查
反查
更新通知
缺少数据扩展能力，每次都要反查服务
DB
缺少权限控制
消费情况无感知
失败无感知
流量洪峰
拉模式
业务感知很多无用消息
可以任意拉全量数据
业务A
全量拉消息
业务B
全量拉消息
Hippo
反查
反查
更新通知
DB
只提供推模式
媒资接入层服务
拉数据
媒资变更消息
拉消息
消息接收器
内部消息队列
版本号控制服务
版本控制
频率控制
数据过滤
失败告警
分发master
业务配置管理
分发slave
分发slave
分发slave
进程级别slave
支持平行扩容
http/taf
http/taf
http/taf
业务A
业务B
业务C
推送流水记录

数据总线


accessLayer
2. 根据vid setid请求全量视频字段数据
MediaMessageReceiver
3. 版本控制
Important
1.读取队列数据
版本控制
4. 存储版本号
Redis
Flush
1. 读取队列数据
洗数据逻辑
2. 进行版本写入
缓存队列
5. 将数据推入推送队列
balloonKeeper
实现主从和配置分发
Normal
分布式限流
上游媒资数据源
推送队列
Master
kafka监控
速率控制
订阅者监控
告警
7. 根据订阅者配置处理数据
Slave #1
过滤字段
版本检验
失败通知
速率控制
MySQL
6. slave读取推送数据
slave
Slave #2
8. 推送数据给订阅方
Slave #3
Slave #n
数据订阅方

TDSQL 压测指标

性能测试

cyrilliang 创建于2019-08-26，cyrilliang 更新于2020-03-10　浏览量 (77)

编辑　关注　评论　更多

总结论

- TDSQL 基本能够满足媒资场景的业务需求
- 当前 3 proxy，8 实例的配置下，单行数据操作的压测情况为：插入性能：9WQPS左右，查询性能：9WQPS左右（利用上从库，可以再*3），修改性能：10WQPS左右，当表的数据存储行数达到3亿时，仍然可以有稳定的性能
- 当前查询性能瓶颈主要在proxy上，增加proxy也可以提升性能
- 单proxy，8实例下的插入性能在 QPS 3W8 时已接近极限，CPU 已达到 80%+
- 尽量不要使用过多的DB连接，连接过多，TDSQL 的性能会急剧下降，请求延迟将会大大增加
- **单表达到4亿存储量级时（8 实例），单表插入性能急剧下降，下降为约3~5k QPS，查询性能和修改性能则基本不受影响**

压测细节

插入性能

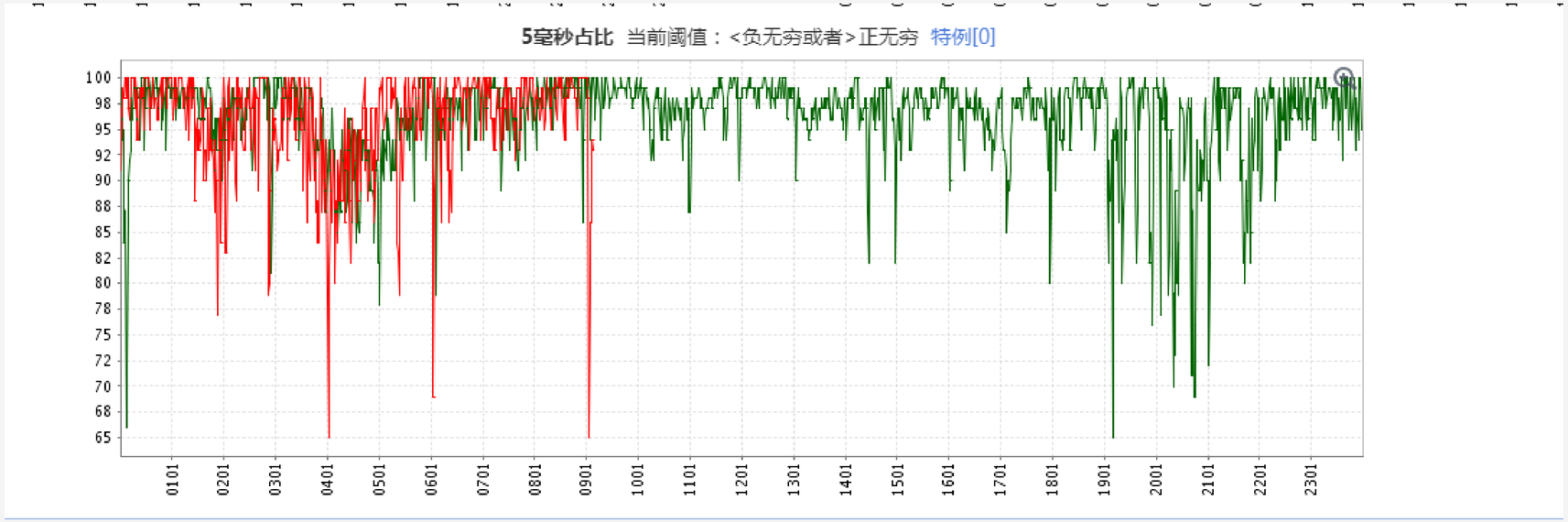
TestCase: 最高链接数测试

测试机器数量：4, proxy 数量 1

执行命令：

目录

TDSQL 压测指标


5毫秒占比 当前阈值：<负无穷或者>正无穷 特例[0]
100
98
95
92
90
88
85
82
80
78
75
72
70
68
65
0101
0201
0301
0401
0501
0601
0701
0801
0901
1001
1101
1201
1301
1401
1501
1601
1701
1801
1901
2001
2101
2201
2301