# TOPS终端优化方案

**文档修订记录**

|  |  |  |  |
| --- | --- | --- | --- |
| **版本号** | **日期** | **修改人** | **摘要** |
| V1.0 | 2023-03-02 | TOPS终端项目组 | 新建文档 |

## 问题背景

鉴于当前门户主要是通过调用三方API进行简单的查询操作，并且代码不涉及到复杂的业务逻辑处理，针对接口响应时间过长的问题，项目组进行了充分的分析与论证。由于数据来源于数仓，其本身的优化空间小，那么项目组就考虑通过在调用方为接口添加缓存以及前端优化来解决响应慢的问题。

## 目标

在较少的工作量下，实现页面数据加载用时在3秒以内，进而满足日常使用。

## 整体优化方案

主要从三个层面来考虑：

1. 前端优化：开启nginx的gzip功能以及缓存静态资源

针对前台静态资源(js、css、font、img)设置浏览器缓存，可通过nginx代理配置实现，可提升页面20%提升，需注意前端发布文件版本问题；缓存高频交互接口（如字典数据、最大交易日查询），登录后请求后缓存，登出失效删除缓存，进而减少单页后端请求数，降低队列请求数。开启nginx传输gzip压缩，可提升2-5倍传输性能。

1. 后端优化：为接口添加缓存

首先针对调用第三方的API的查询场景梳理出性能慢的接口清单,针对重点接口添加查询缓存。当前的方案是缓存首次请求的数据，然后定期清理缓存。

1. 服务平台接口优化：后期会向“一库”团队反馈，提出优化要求

## 后端优化技术选型

1. **后端key设计**

缓存key设计，举例dap:{biz}:{[paramv1:paramv2:...]}，biz代表业务接口编号，{[paramv1:paramv2:...]}代表接口参数值拼接，其他参数不可为空，例如：dap:bondMap:20230201:20230501。

1. **缓存技术选型**

目前考虑使用Jetcache来实现缓存需求，通过redis缓存数据。Jetcache是阿里的一个基于Java封装的开源缓存框架，并且目前支持统一的API访问缓存系统、支持注解实现声明式的方法缓存、支持Key的自动生成以及Value序列化的策略的配置、分布式缓存自动刷新功能、与springboot完美兼容等。考虑到实现简单以及相对较少的工作量，该缓存框架是一个很好的选择。

## demo

* 读取流程
* 更新流程

## 数据一致性问题（解决方案）

考虑数据一致性问题，主要考虑缓存与实际接口平台存在差异问题，原因可能为平台接口数据未跑完情况或应用调整了逻辑。第一步可减少数据缓存过期时间，提升时效。针对应用逻辑调整的情况可手工触发一次清空缓存，兜底方案是每天定时清理清理缓存一次。

不同的业务或接口获得数据注入缓存及清理缓存的策略可能存在差异，因此对应不同的需求，通过配置文件添加不同缓存的缓存策略，并在不同场景的代码中使用注解，完成数据个性化缓存的需求

## 关联系统的影响

应用发布上线或者数据平台重新跑任务需要，根据实际情况是否手工触发一次缓存清理操作。

## 是否有利于后期技术拆分

应用缓存以注解的形式添加在service实现层业务方法上， 对代码浸入性少，如果考虑不适用缓存，则在全局配置文件禁用即可。

## 资源分配情况（人力、硬件以及具体配置）

**工作量预估：**

初步统计需要添加缓存的接口总数是80个上下，经过初步评估工作量在20人天左右，其他如缓存结构设计、定时清理缓存、手工清理单接口缓存等额外功能工作量10人天。总计30人天

**硬件及配置：**

Redis 服务器需要2台做集群，硬件配置如下：

CPU：2个或更多的CPU核心，建议使用高性能的CPU

内存：至少16GB RAM

存储：使用高速磁盘，如SSD，至少50GB的存储空间，用于持久化数据快照和日志文件。

## 技术参考

JetCache：https://github.com/alibaba/jetcache/tree/master/docs/CN