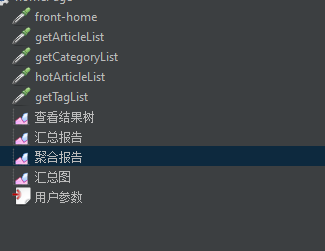
关于本博客在本地运行环境下的性能优化。

1. 关于getArticleList的性能优化。
   1. 背景：用户在“访问首页”时，除前端项目资源外，还会调用后端项目的一些接口，要想对“访问首页”进行优化，其中少不了对后端接口的优化。
   2. 问题分析：用50线程数/秒对“访问首页”接口进行拟并发请求，结果如下图：



（内容为模拟访问“访问首页”接口的线程组，其中front-home是对前端服务的取样器，其余为后端取样器）

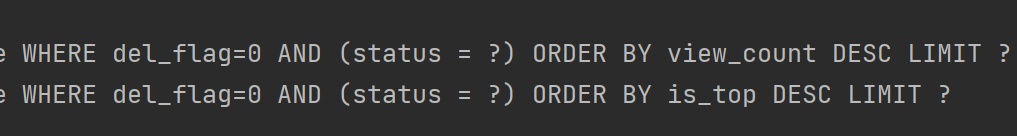


（内容为吞吐量平稳之后，各取样器的相应请求数据）

从上面两图不难看出，在后端接口中，getArticleList接口的平均访问时间为23ms，中位数为14ms，都明显高于其他后端接口，因此，优化getArticleList接口成了优化“访问首页”接口的重要一环。那么如何优化？首先想到的是索引，然后就是关闭后端mybatis日志，当然关闭日志可以暂时放一放。

* 1. 问题分析：尝试使用索引，发现并不合适，考虑缓存；同时对getArticleList的另一种

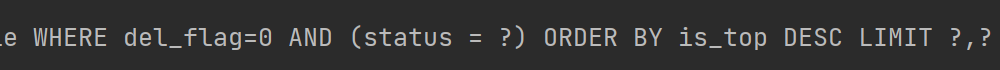
从前端项目进入首页，然后查看后端日志，如下图所示。从下图不难看出，“进入首页”接口涉及到的sql语句包含del\_flag,status,view\_count,is\_top字段。



（“进入首页”接口的后端sql语句）

首先看del\_flag,删除字段只有两个值0和1，因此没有必要对此建立索引，status，is\_top同理。因此，想要使用索引，并不合适。但是鉴于访问首页的这些article经常访问，而且对于数据一致性要求并不高，我们可以考虑将这些article先放入redis缓存中。

同时，从首页还可以点击“查看更多”按钮来进行getArticleList接口的调用。这时的sql语句变为了如下图：



Sql从limit m变成了 limit m, n。这种sql在article数据量特别大时会导致全局扫描太多数据并丢弃，很浪费性能。