05 | 倒排索引：如何从海量数据中查询同时...

试想这样一个场景：假设你已经熟读唐诗 300 首了。这个时候，如果我给你一首诗的题目，你可以马上背出这首诗的内容吗？相信你一定可以的。但是如果我问你，有哪些诗中同时包含了“极”字和“客”字？你就不见得能立刻回答出来了。你需要在头脑中一首诗一首诗地回忆，并判断每一首诗的内容是否同时包含了“极”字和“客”字。

很显然，第二个问题的难度比第一个问题大得多。那从程序设计的角度来看，这两个问题对应的检索过程又有什么不同呢？今天，我们就一起来聊一聊，两个非常常见又非常重要的检索技术：**正排索引和倒排索引。**

什么是倒排索引?

我们先来看比较简单的那个问题：给出一首诗的题目，马上背出内容。这其实就是一个典型的键值查询场景。针对这个场景，我们可以给每首诗一个唯一的编号作为 ID，然后使用哈希表将诗的 ID 作为键（Key），把诗的内容作为键对应的值（Value）。这样，我们就能在 O(1) 的时间代价内，完成对指定 key 的检索。这样一个以对象的唯一 ID 为 key 的哈希索引结构，**叫作正排索引（Forward Index）**

一般来说，我们会遍历哈希表，遍历的时间代价是 O(n)。在遍历过程中，对于遇到的每一个元素也就是每一首诗，我们需要遍历这首诗中的每一个字符，才能判断是否包含“极”字和“客”字。假设每首诗的平均长度是 k，那遍历一首诗的时间代价就是 O(k)。从这个分析中我们可以发现，这个检索过程全部都是遍历，因此时间代价非常高。对此，有什么优化方法吗？

我们先来分析一下这两个场景。我们会发现，“根据题目查找内容”和“根据关键字查找题目”，这两个问题其实是完全相反的。既然完全相反，**那我们能否“反着”建立一个哈希表来帮助我们查找呢？**也就是说，如果我们以关键字作为 key 建立哈希表，是不是问题就解决了呢？