# 布隆过滤器

1.常见的基础数据存储容器有哪些？基础数据结构：数组+链表+map+set（树）

2. 10亿的黑名单email，如何来进行黑名单过滤？垃圾邮件：HashMap 或者 Hash算法

白名单

3.缓存相信大家都知道，但是缓存并不是百分百命中的，通常我们会根据一个id（字符串uuid）判断缓存里面有没有数据，如果没有再去数据库查询，那你有没有想过突然来了一个超级并发去查询一个缓存不存在的id怎么办？缓存击穿：

bitMap：

Set是一种数据结构，它的特点是里面所存的元素是不能重复的。在Java中主要有两种实现方式：

hashMap key-value

Set：key

HashSet：其内部是一个Hash表（HashMap）实现，不能保证元素的顺序。

TreeSet：使用元素的自然顺序进行排序，当然你也可以自己指定排序的规则。其底层是通过二叉树实现的，也就是红黑树。

我们到目前为止主要的容器存储结构有：

List：

可以重复存储对象

插入的顺序和遍历的顺序是一致的

常用的实现方式：链表、数组（ArrayList,LinkedList,Vector）

Set：

不允许重复对象

无法保证每个元素的插入和输出顺序，无序容器。

TreeSet是有序的

常用的实现方式：HashSet，TreeSet，LinkedHashSet（强行保证输出顺序和插入顺序一致，双向链表，耗费空间）

Map：

Map是键值对的形式存储，会有key+value：

Map不允许出现相同的key，出现就会倍覆盖

Map主要实现方式：HashMap，HashTable，TreeMap（也是一个有序的，默认按照自然顺序，其底层结构同样是红黑树）

布隆过滤器是一个非常巧妙的数据结构，在很多高并发大数据项目中都有应用，它的特点就是高效的查找和插入，他的核心一句话就是：

我告诉你不存在的那就肯定不存在，但是我告诉你存在，其实有可能存在的

布隆过滤器结构：它的结构和bitMap非常类似。它是由一个很长的bit数组以及k个hash函数组成。

实现的思想：

插入：将一个插入的元素使用K个hash函数进行k次计算，将得到的Hash值所对应的bit数组下标置为1。

查找：跟插入一样的道理，将查找的元素使用k个函数进行k次计算，将得到的值找出对应的bit数组下标，判断是否为1，如果都为1则说明这个值可能在序列中，反之肯定不在序列中。

为什么是可能在序列中呢？ 每次hash都碰撞

删除：不支持删除

通过以上原理我们知道，这个布隆过滤器其实最重要的还是在这个误判率，怎么样才能降低这个误判率呢？

其实这个是有公式推导的：

以这个为例：假设给你一个1亿的黑名单email，如何来进行黑名单过滤。要求误判率不得高于 0.1%

分析问题：

1.假设布隆过滤器开的长度为m

2.元素个数为N

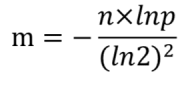
3.哈希函数个数为K

4.误判率为p

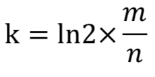
第一步：假设

如何来确定bit数组的大小和哈希函数K的值：概率论+高数推导出来的，p是误差率

：开的空间：



哈希函数的个数：



真实误差率：



有哪些使用场景:前提你得接受一定的误判率 大约0.1%

1.爬虫过滤：

2.缓存击穿：小数据量的直接用hash或者确定id的可以用bitmap

3.垃圾邮件过滤：

4.秒杀系统：重复买

5.Hbase get操作：

BitMap BloomFilter