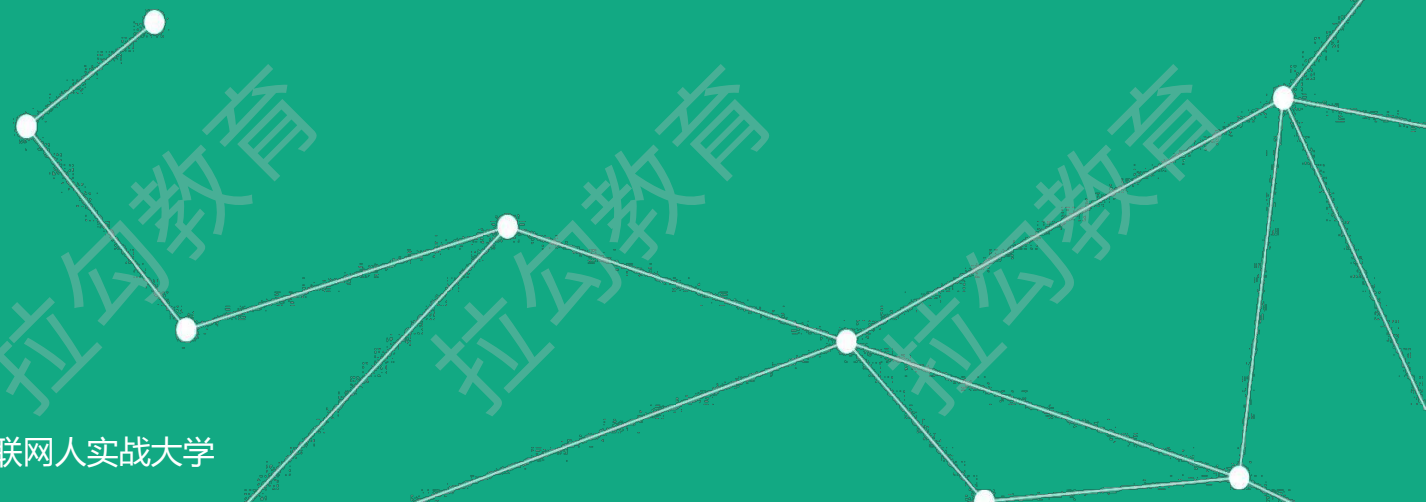
课时 07

哈希碰撞的本质及解决方式

1. 哈希碰撞的情况
2. 开放寻址法（Open Addressing）
3. 分离链接法（Separate Chaining）



哈希碰撞的情况

在概念上哈希表可以定义为是一个根据键（Key）而直接访问在内存中存储位置的值（Value）的数据结构

这里所说的键就是指哈希函数的输入，而这里的值并不是指哈希值，而是指我们想要保存的值

哈希碰撞的情况

现实中

想要有一个完美的哈希函数，将输入值转换成哈希值而不产生哈希碰撞基本是不可能的

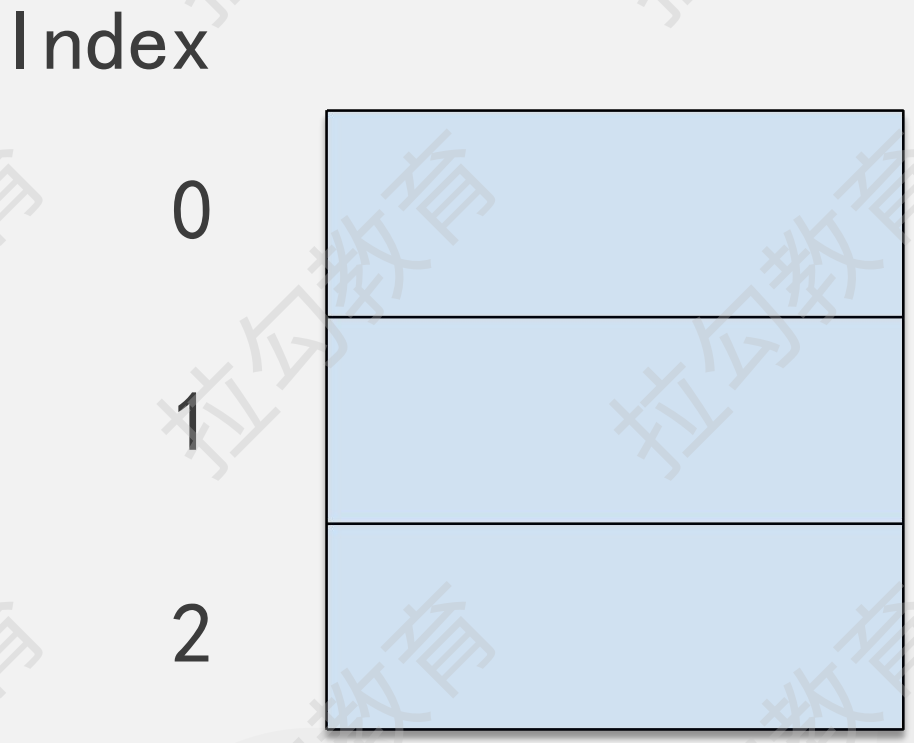
所以哈希表在通过键来访问存储位置的值的时候，是根据我们处理哈希碰撞来决定它自身操作的

哈希碰撞的情况

假设存储哈希表的底层数据结构是一个大小为 3 的数组

哈希表的键是好友的名字

哈希表的值是这个好友的电话号码



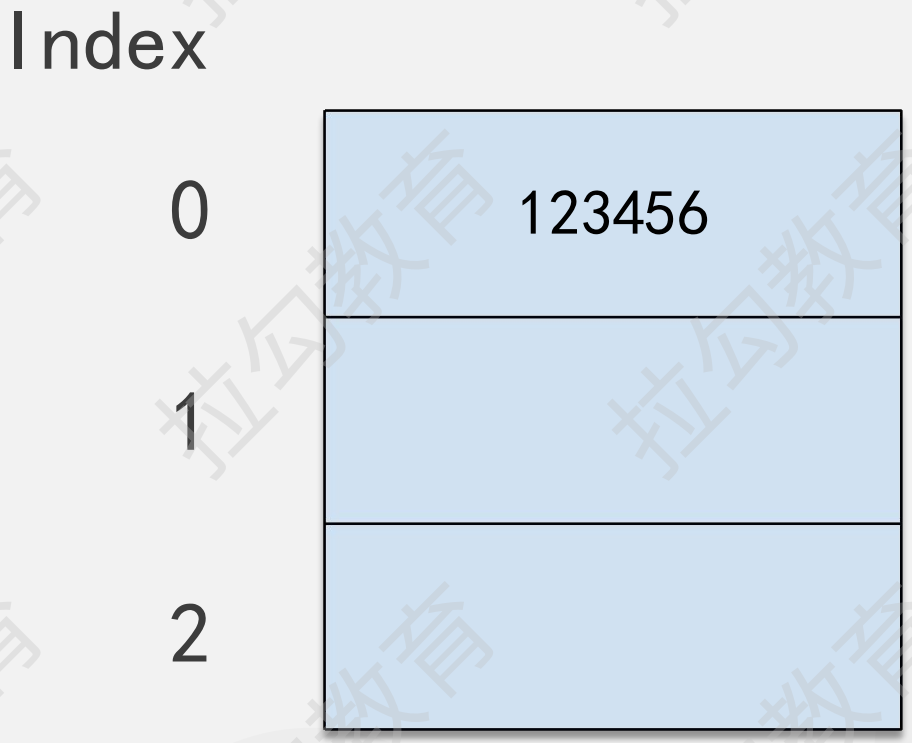
哈希碰撞的情况

假设第一个输入的键值对是（Tom：123456）

表示好友的名字叫 Tom，电话号码为 123456

同时假设 Tom 这个字符串在通过哈希函数之后的所产生的哈希值是 0

此时可以把 123456 这个值放在以哈希值为索引的地方



哈希碰撞的情况

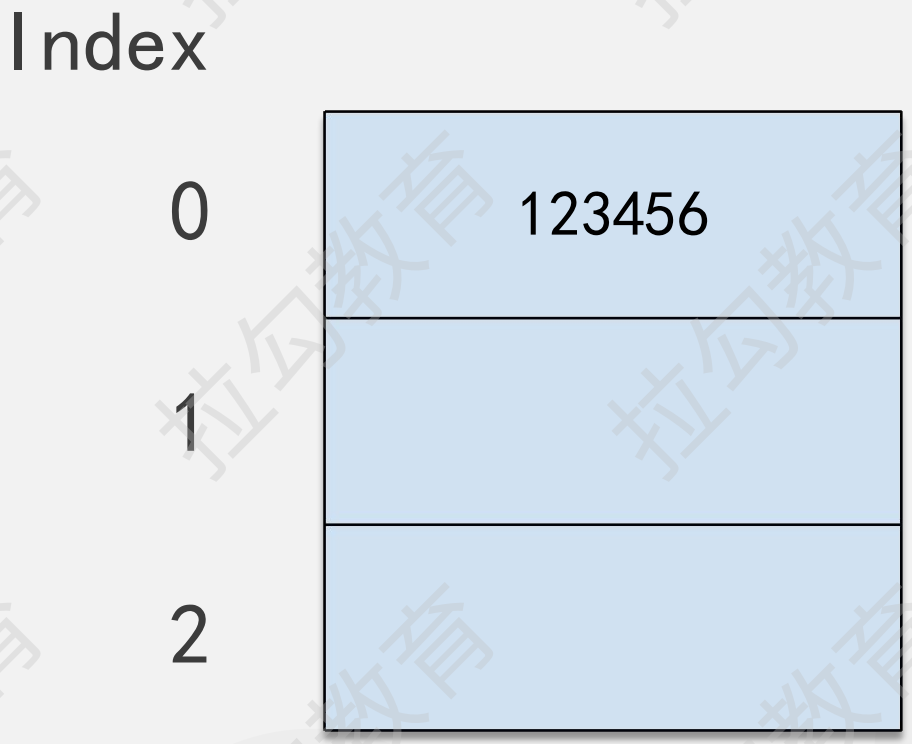
紧接着输入的第二个键值对是（Jack：456789）

同时假设 Jack 这个字符串在通过哈希函数之后的

所产生的哈希值也是 0

而因为索引为 0 的位置已经存放一个值了，也就

表示这时候产生了哈希碰撞



开放寻址法（Open Addressing）

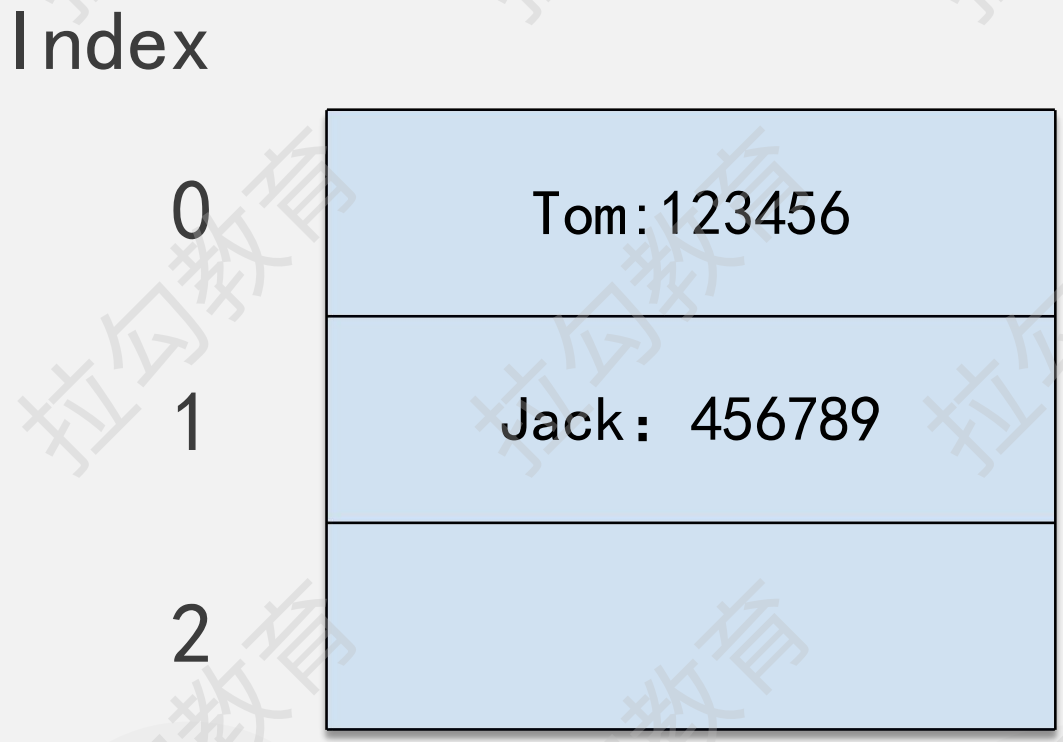
开放寻址法本质上是在数组中寻找一个还未被使用的位置，将新的值插入

这样做的好处是利用数组原本的空间而不用开辟额外的空间来保存值

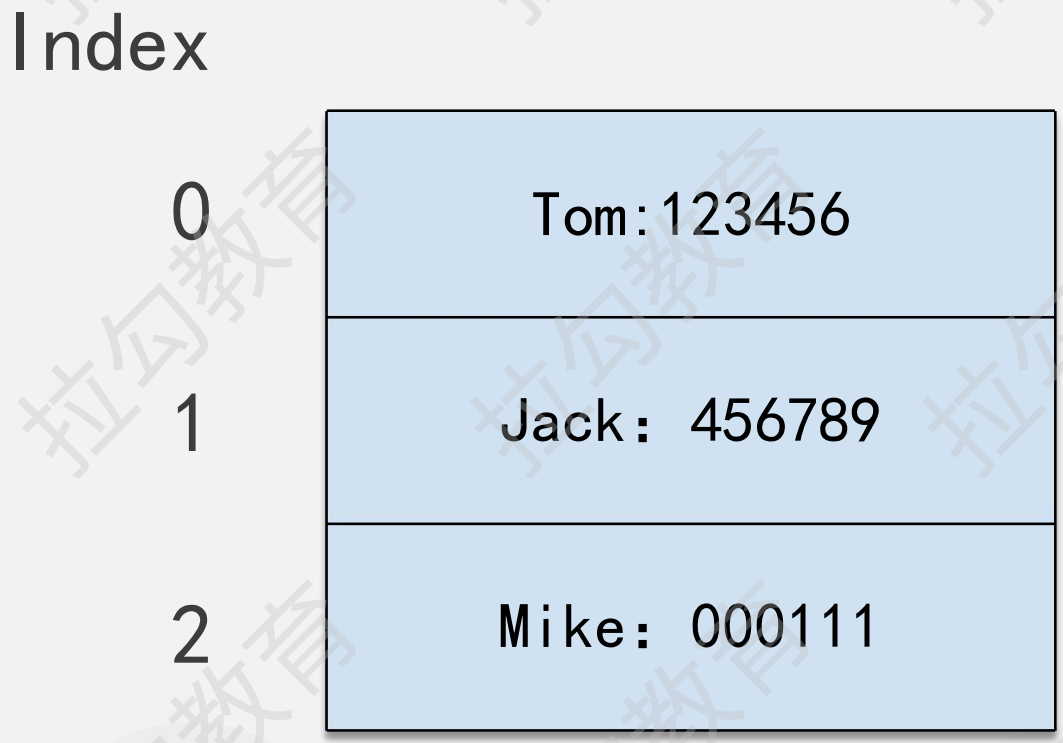
最简单明了的方法就是沿着数组索引，往下一个一个地去寻找还未被使用的空间

这种方法叫做线性探测（Linear Probing）

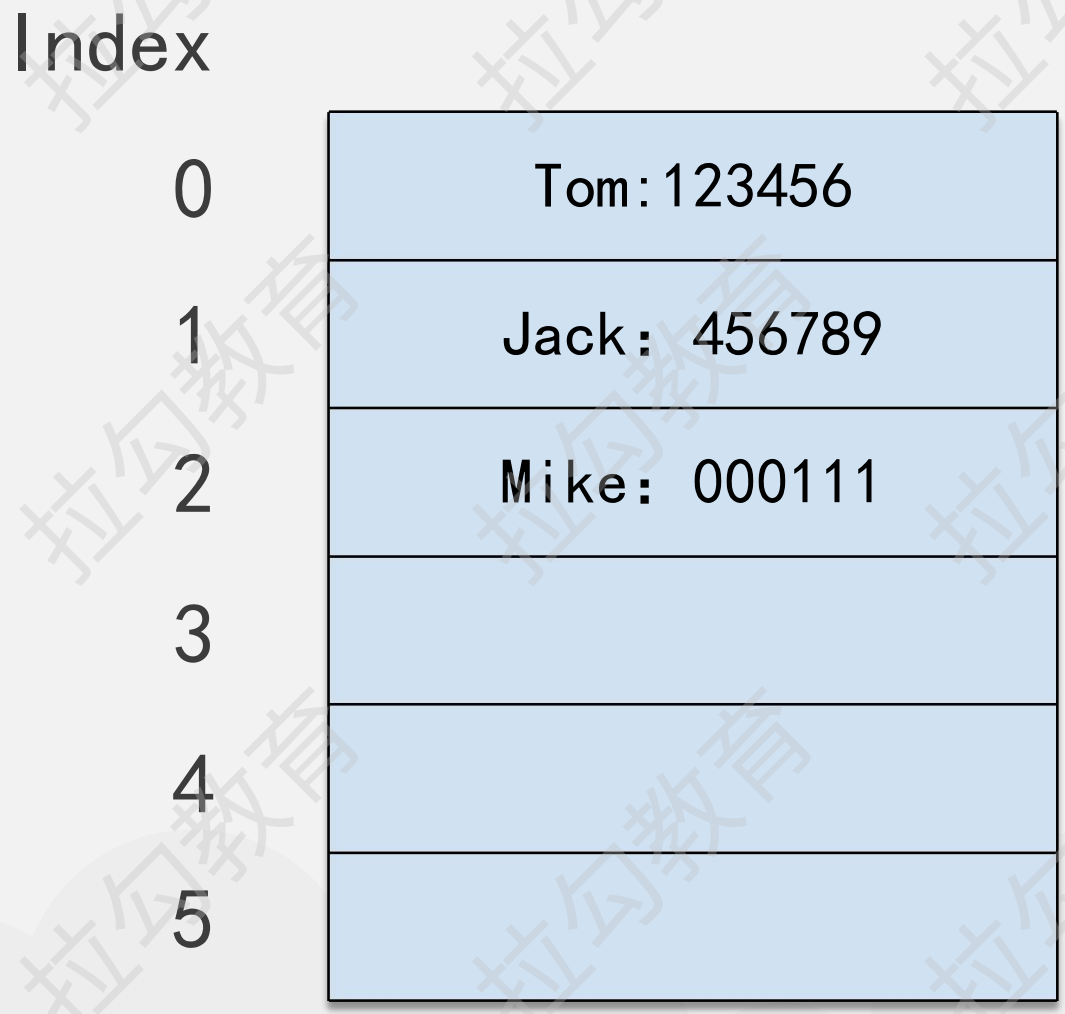
开放寻址法（Open Addressing）



开放寻址法（Open Addressing）



开放寻址法（Open Addressing）



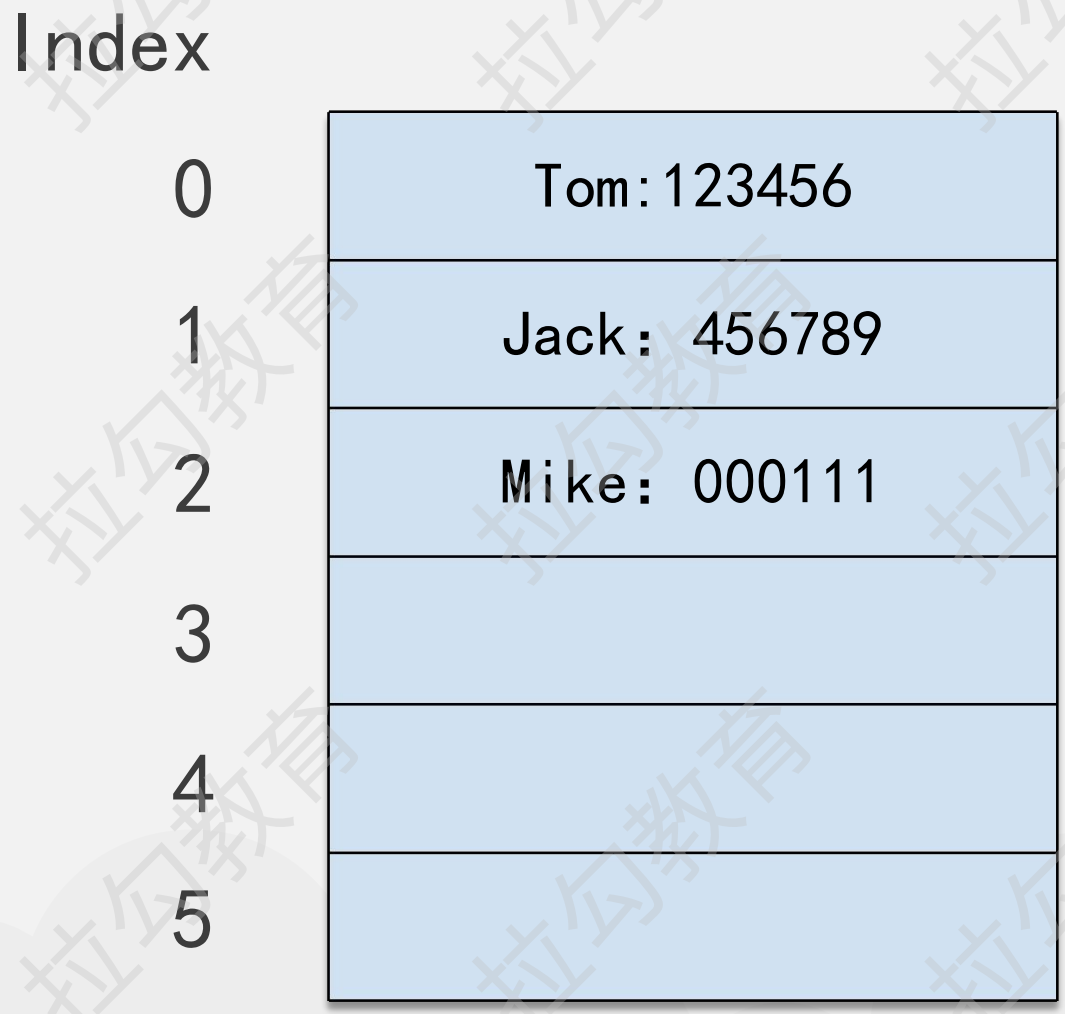
开放寻址法（Open Addressing）

采用这种线性探测的好处是算法非常简单，但它也有自身的缺点

因为每次遇到哈希碰撞的时候都只是往下一个元素地址检测是否有未被使用的位置存在

所以有可能会导致一种叫做哈希聚集（Primary Clustering）的现象

开放寻址法（Open Addressing）



开放寻址法（Open Addressing）

负载因子可以被定义为是哈希表中保存的元素个数 / 哈希表中底层数组的大小

当这个负载因子过大时，则表示哈希表底层数组所保存的元素已经很多了

所剩下的未被使用过的数组位置会很少，同时产生哈希碰撞的概率会很高，这并不是我们想看到的

开放寻址法（Open Addressing）

采用这种线性探测的好处是算法非常简单，但它也有自身的缺点

因为每次遇到哈希碰撞的时候都只是往下一个元素地址检测是否有未被使用的位置存在

所以有可能会导致一种叫做哈希聚集（Primary Clustering）的现象

开放寻址法（Open Addressing）

- 平方探测（Quadratic Probing）指的是每次检查空闲位置的步数为平方的倍数

  例如当新元素插入的键所产生的哈希值为 i，那下一次的检测位置为：i 加上 1 的平方、i 减去 1 的 平方

  i 加上 2 的平方、i 减去 2 的平方、…，以此类推

- 二度哈希（Double Hashing）指的是数据结构底层会保存多个哈希函数，当使用第一个哈希函数算出的哈希值产生了哈希碰撞之后，将会使用第二个哈希函数去运算哈希值，…，以此类推

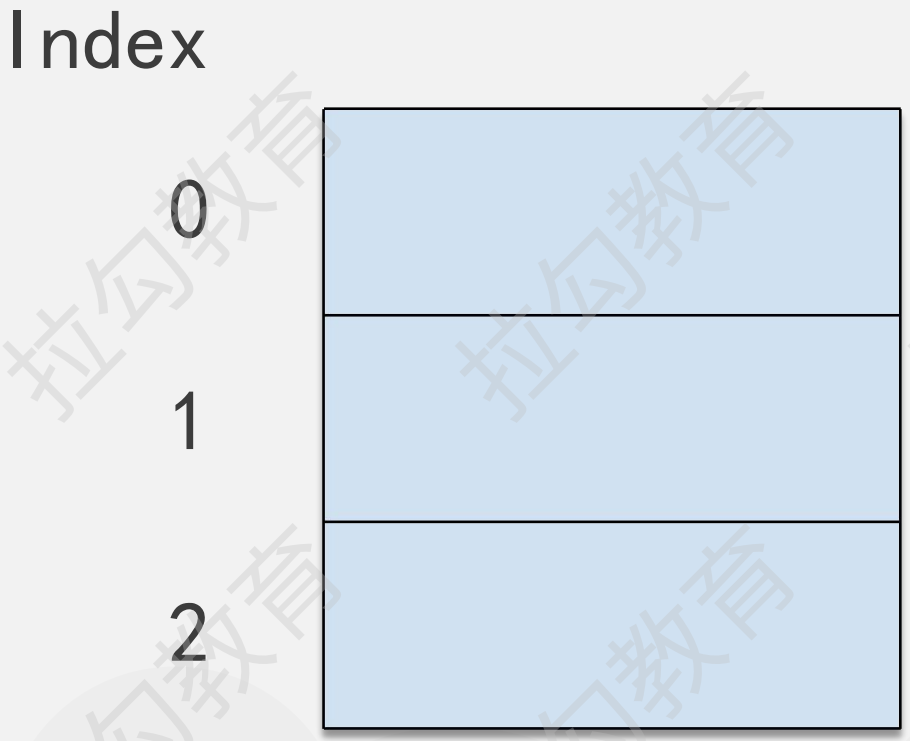
分离链接法（Separate Chaining）

拉勾 LAGOU.COM

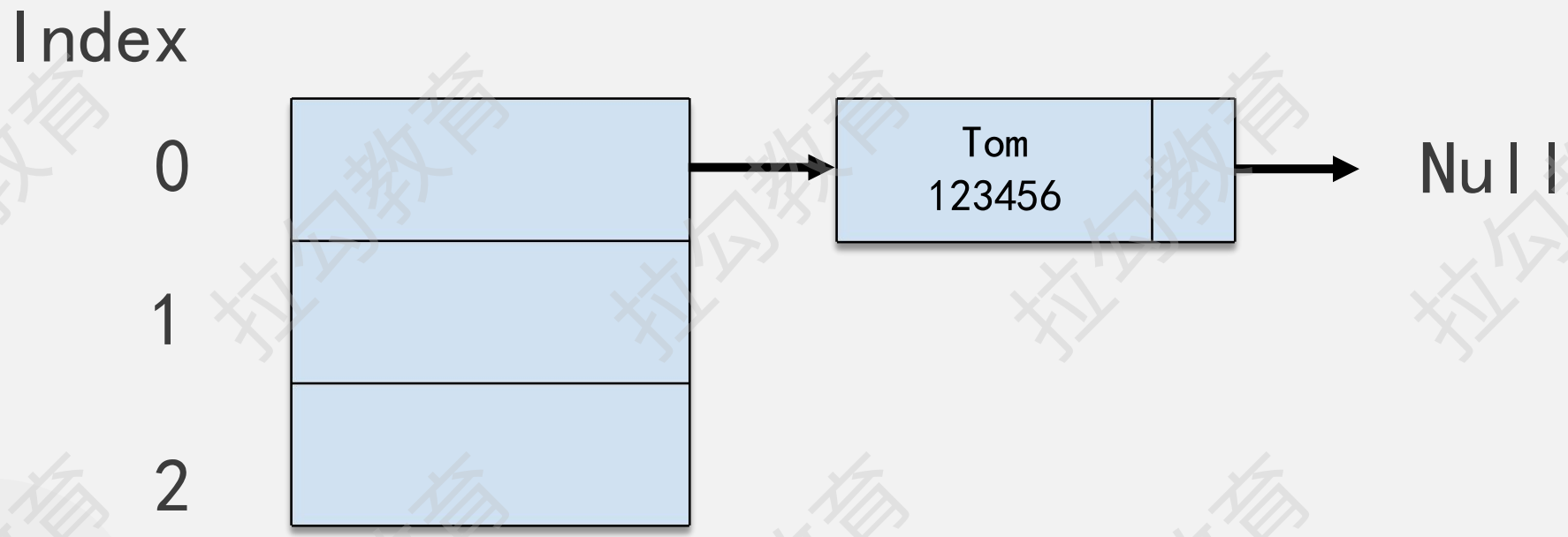
分离链接法与链表有很大的关系，它的本质是将所有的同一哈希值的键值对都保存在一个链表中

而哈希表底层的数组元素就是保存这个哈希值对应的链表

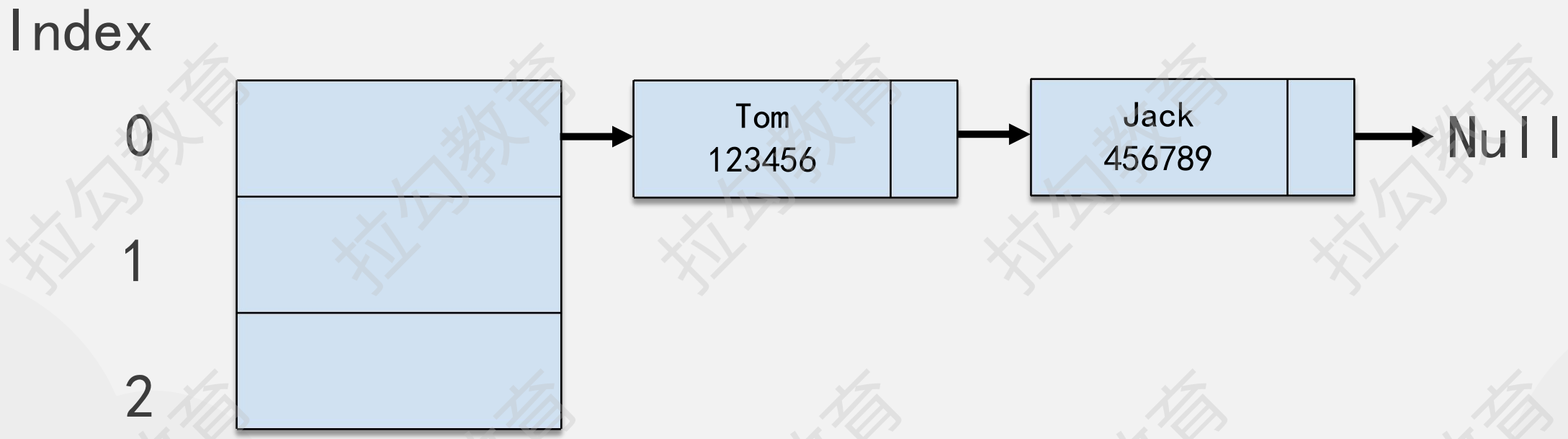
分离链接法（Separate Chaining）



分离链接法（Separate Chaining）



分离链接法（Separate Chaining）



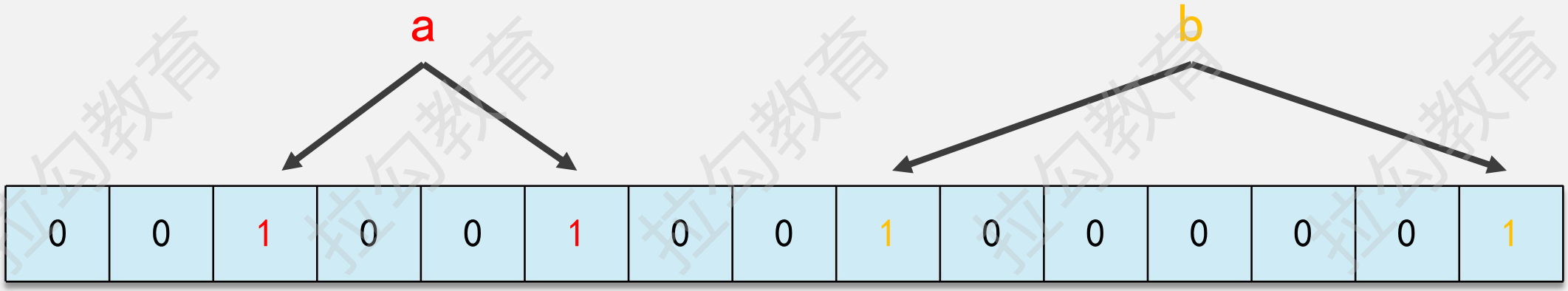
Bloom Filter

Bloom Filter 是一个哈希表和位数组相结合的基于概率的数据结构，由 Bloom 在 1970 年提出

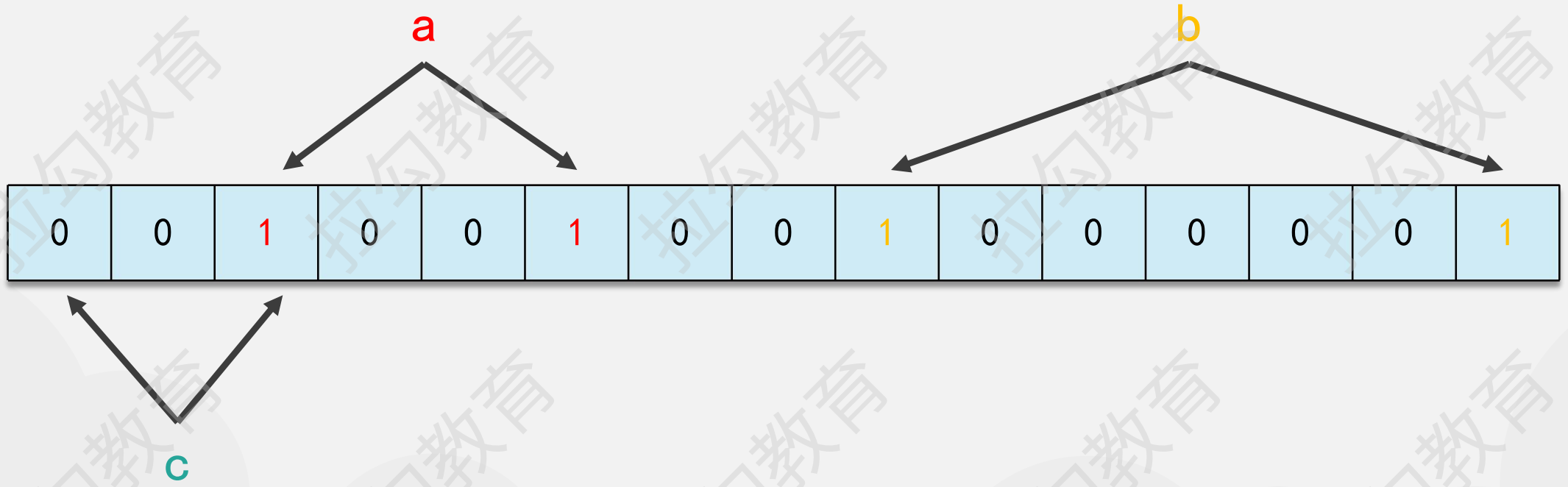
主要用于在超大的数据集合中判断一个元素是否存在这个集合中

像判断一个人是否在黑名单中一样，或者判断一封邮件是否属于垃圾邮件的范畴等等

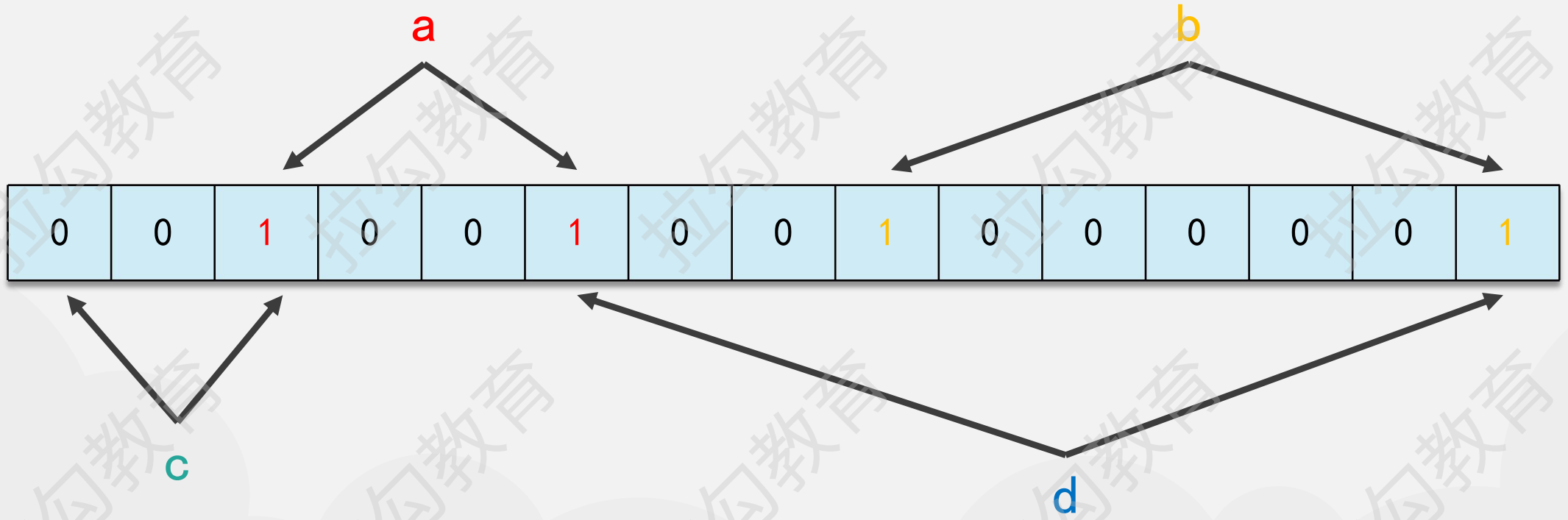
Bloom Filter



Bloom Filter



Bloom Filter



Bloom Filter

误判率的公式

$$\left(1-e^{-kn/m}\right)^k$$

m 表示位数组里位的个数

n 表示已经存储在集合里的元素个数

k 表示哈希函数的个数