# Java 基础

## HashMap的实现结构

数组 + 单链表

## ConcurrentHashMap怎么理解

1）线程安全的Map，

2）基于分段锁实现，默认是16级分段，这里面的每个Segment都相当于一个HashTable

3）每个Segment只允许一个线程写，但是不限制读。

## ThreadLocal怎么理解？

1. 空间换时间的一种做法；
2. 数据隔离在每个线程内部，数据不共享。

3）专门一个博文整理了这个问题：<http://blog.csdn.net/u010853261/article/details/55105173>

## NIO和BIO和AIO怎么理解？

1. BIO 同步阻塞IO，一个连接一个线程处理；
2. NIO 同步非阻塞IO，

# 数据结构：

## 数组和单链表查找速度比较：查找target

1. 首先应该是问数据是不是有序的，如果是无序的：

对于无序的数据，应该是数组查询速度快一些，我们可以借助 快速排序 的思想，通过一轮快排，就可以将序列分为两个序列，左边的比基数小，右边比基数大， 通过比较基数与target的大小，将序列定位在其中一部分。时间复杂度是O（logn）。

对于链表时间复杂度是O(n)

1. 如果数据时有序的：

数组快。需要我们自己先找到target的定位，这时候数组通过二分查找，时间复杂度是O(logn), 链表只能顺序遍历，时间复杂度是O（n）

## 2. 数组和单链表删除元素。

a)知道目标数据的位置：就是链表快，只涉及指针改变，但是数组涉及到数组数据的部分移动。

b)如果不知道目标数据，就是还是要先查找，再删除，数组虽然通过快排或则二分查找快，但是需要移动大量数据，还是性能不高。这时候链表不需要移动数据，一般情况下链表快。

# Mybatis

## 1.$ 和 # 号的区别。

1）${}是通过直接将变量字符串 赋值给 查询条件 可能会引发SQL注入的问题。

2）#{} 是通过占位符 ？ MyBatis会进行预编译，防止SQL注入，更加安全。

# 数据库

## Select for update

## 2.去重：

单表去重使用 distinct

多表去重使用 group by

## 3.左连接

# 设计模式：

## 代理模式；

## 责任链模式；

# 分布式：

## 淘宝秒杀—》分布式锁。

## 怎么理解MVC