1. junit 用法，before,beforeClass,after, afterClass 的执行顺序

2. 分布式锁

3. nginx 的请求转发算法，如何配置根据权重转发

4. 用 hashmap 实现 redis 有什么问题（死锁，死循环，可用 ConcurrentHashmap）

5. 线程的状态

5. 线程的阻塞的方式

6. sleep 和 wait 的区别

7. hashmap 的底层实现

8. 一万个人抢 100 个红包，如何实现（不用队列），如何保证 2 个人不能抢到同一个红包，可用分布式锁

9. java 内存模型，垃圾回收机制，不可达算法

10. 两个 Integer 的引用对象传给一个 swap 方法在方法内部交换引用，返回后，两个引用的值是否会发现变化

11. aop 的底层实现，动态代理是如何动态，假如有 100 个对象，如何动态的为这 100 个对象代理

12. 是否用过 maven install。 maven test。git（make install 是安装本地 jar 包）

13. tomcat 的各种配置，如何配置 docBase

14. spring 的 bean 配置的几种方式

15. web.xml 的配置

16. spring 的监听器。

17. zookeeper 的实现机制，有缓存，如何存储注册服务的

18. IO 会阻塞吗？readLine 是不是阻塞的

19. 用过 spring 的线程池还是 java 的线程池？

20. 字符串的格式化方法 （20，21 这两个问题问的太低级了）

21. 时间的格式化方法

22. 定时器用什么做的

23. 线程如何退出结束

24. java 有哪些锁？乐观锁 悲观锁 synchronized 可重入锁 读写锁,用过 reentrantlock 吗？reentrantlock 与 synmchronized 的区别

25. ThreadLocal 的使用场景

26. java 的内存模型，垃圾回收机制

27. 为什么线程执行要调用 start 而不是直接 run（直接 run，跟普通方法没什么区别，先调 start，run 才会作为一个线程方法运行）

28. qmq 消息的实现机制(qmq 是去哪儿网自己封装的消息队列)

29. 遍历 hashmap 的三种方式

30. jvm 的一些命令

31. memcache 和 redis 的区别

32. mysql 的行级锁加在哪个位置

33. ConcurrentHashmap 的锁是如何加的？是不是分段越多越好

34. myisam 和 innodb 的区别（innodb 是行级锁，myisam 是表级锁）

35. mysql 其他的性能优化方式

36. linux 系统日志在哪里看

37. 如何查看网络进程

38. 统计一个整数的二进制表示中 bit 为 1 的个数

39. jvm 内存模型，java 内存模型

【阿里巴巴面试题目】

40. 如何把 java 内存的数据全部 dump 出来

41. 如何手动触发全量回收垃圾，如何立即触发垃圾回收

42. hashmap 如果只有一个写其他全读会出什么问题

43. git rebase

44. mongodb 和 hbase 的区别

45. 如何解决并发问题

46. volatile 的用途

47. java 线程池（好像之前我的理解有问题）

48. mysql 的 binlog

49. 代理模式

50. mysql 是如何实现事务的

51. 读写分离何时强制要读主库，读哪个从库是通过什么方式决定的，从库的同步 mysql 用的什么方式

52. mysql 的存储引擎

53. mysql 的默认隔离级别，其他隔离级别

54. 将一个链表反转（用三个指针，但是每次只发转一个）

55. spring Aop 的实现原理，具体说说

56. 何时会内存泄漏，内存泄漏会抛哪些异常

57. 是否用过 Autowire 注解

58. spring 的注入 bean 的方式

59. sql 语句各种条件的执行顺序，如 select， where， order by， group by

60. select xx from xx where xx and xx order by xx limit xx； 如何优化这个（看 explain）

61. 四则元算写代码

62. 统计 100G 的 ip 文件中出现 ip 次数最多的 100 个 ip

63. zookeeper 的事物，结点，服务提供方挂了如何告知消费方

64. 5 台服务器如何选出 leader(选举算法)

65. 适配器和代理模式的区别

66. 读写锁

67. static 加锁

68. 事务隔离级别

69. 门面模式，类图(外观模式)

70. mybatis 如何映射表结构

71. 二叉树遍历

72. 主从复制

73. mysql 引擎区别

74. 静态内部类加载到了哪个区？方法区

75. class 文件编译后加载到了哪

76. web 的 http 请求如何整体响应时间变长导致处理的请求数变少，该如何处理？用队列，当处理不了那么多 http 请求时将请求放到队列

中慢慢处理，web 如何实现队列

77. 线程安全的单例模式

78. 快速排序性能考虑

79. volatile 关键字用法

80. 求表的 size，或做数据统计可用什么存储引擎

81. 读多写少可用什么引擎

82. 假如要统计多个表应该用什么引擎

83. concurrenhashmap 求 size 是如何加锁的，如果刚求完一段后这段发生了变化该如何处理

84. 1000 个苹果放 10 个篮子，怎么放，能让我拿到所有可能的个数

85. 可重入的读写锁，可重入是如何实现的？

86. 是否用过 NIO

87. java 的 concurrent 包用过没

88. sting s=new string("abc")分别在堆栈上新建了哪些对象

89. java 虚拟机的区域分配，各区分别存什么

90. 分布式事务（JTA）

91. threadlocal 使用时注意的问题（ThreadLocal 和 Synchonized 都用于解决多线程并发访问。但是 ThreadLocal 与 synchronized 有本质的区别。synchronized 是利用锁的机制，使变量或代码块在某一时该只能被一个线程访问。而 ThreadLocal 为每一个线程都提供了变量的副本，使得每个线程在某一时间访问到的并不是同一个对象，这样就隔离了多个线程对数据的数据共享。而 Synchronized 却正好相反，它用于在多个线程间通信时能够获得数据共享）

92. java 有哪些容器(集合，tomcat 也是一种容器)

93. 二分查找算法

94. myisam 的优点，和 innodb 的区别

95. redis 能存哪些类型

96. http 协议格式，get 和 post 的区别

97. 可重入锁中对应的 wait 和 notify

98. redis 能把内存空间交换进磁盘中吗(这个应该是可以的，但是那个面试官非跟我说不可以)

99. java 线程池中基于缓存和基于定长的两种线程池，当请求太多时分别是如何处理的？定长的事用的队列，如果队列也满了呢？交换进磁盘？基于缓存的线程池解决方法呢？

100. synchronized 加在方法上用的什么锁

101. 可重入锁中的 lock 和 trylock 的区别

102. innodb 对一行数据的读会枷锁吗？不枷锁，读实际读的是副本

103. redis 做缓存是分布式存的？不同的服务器上存的数据是否重复？guava cache 呢？是否重复？不同的机器存的数据不同

104. 用 awk 统计一个 ip 文件中 top10

105. 对表做统计时可直接看 schema info 信息，即查看表的系统信息

106. mysql 目前用的版本

107. 公司经验丰富的人给了什么帮助？(一般 boss 面会问这些)

108. 自己相对于一样的应届生有什么优势

109. 自己的好的总结习惯给自己今后的工作带了什么帮助，举例为证

110. 原子类，线程安全的对象，异常的处理方式

111. 4 亿个 int 数，如何找出重复的数（用 hash 方法，建一个 2 的 32 次方个 bit 的 hash 数组，每取一个 int 数，可 hash 下 2 的 32 次方找到它在 hash 数组中的位置，然后将 bit 置 1 表示已存在）

112. 4 亿个 url，找出其中重复的（考虑内存不够，通过 hash 算法，将 url 分配到 1000 个文件中，不同的文件间肯定就不会重复了，再分别找出重复的）

有 1 万个数组，每个数组有 1000 个整数，每个数组都是降序的，从中找出最大的 N 个数，N<1000

113. LinkedHashmap 的底层实现

114. 类序列化时类的版本号的用途，如果没有指定一个版本号，系统是怎么处理的？如果加了字段会怎么样？

115. Override 和 Overload 的区别，分别用在什么场景

116. java 的反射是如何实现的

【阿里巴巴面试题目含答案】

1，mysql 的三大引擎是啥？

mysql 常用的引擎有 InnoDB，MyISAM，Memory，默认是 InnoDB

InnoDB：磁盘表，支持事务，支持行级锁，B+Tree 索引

ps:优点： 具有良好的 ACID 特性。适用于高并发，更新操作比较多的表。需要使用事务的表。对自动灾难恢复有要求的表。

缺点：读写效率相对 MYISAM 比较差。占用的磁盘空间比较大。

mysql 的 4 大特性+4 种隔离级别：

MyISAM：磁盘表，不支持事务，支持表级锁，B+Tree 索引

ps: 优点：占用空间小，处理速度快（相对 InnoDB 来说）

缺点：不支持事务的完整性和并发性

MEMORY(Heap)：内存表，不支持事务，表级锁，Hash 索引，不支持 Blob,Text 大类型

ps: 优点：速度要求快的，临时数据

缺点：丢失以后，对项目整体没有或者负面影响不大的时候。

2，redis 的 hash 算法用的是啥？

redis 应该是使用一致性 hash 算法---MurmurHash3 算法，具有低碰撞率优点，google 改进的版本 cityhash 也是 redis 中用到的哈希算法。

现有的主流的大数据系统都是用的 MurmurHash 本身或者改进

3，nosql 为啥比 sql 快？

Nosql 是非关系型数据库，因为不需要满足关系数据库数据一致性等复杂特性所以速度快；

sql 是关系型数据库，功能强大，但是效率上有瓶颈

4，什么是索引为啥 nosql 没索引？nosql 有索引滴

索引分为聚簇索引和非聚簇索引两种，聚簇索引是按照数据存放的物理位置为顺序的，而非聚簇索引就不一样了；聚簇索引能提高多行检索的速度，而非聚簇索引对于单行的检索很快。

聚簇索引：有主键时，根据主键创建聚簇索引；没有主键时，会用一个唯一且不为空的索引列做为主键，成为此表的聚簇索引；如果以上两个都不满足那 innodb 自己创建一个虚拟的聚集索引

非聚簇索引：非聚簇索引都是辅助索引，像复合索引、前缀索引、唯一索引

5，B+树和 B 树区别？

B 树的非叶子节点存储实际记录的指针，而 B+树的叶子节点存储实际记录的指针

B+树的叶子节点通过指针连起来了，适合扫描区间和顺序查找。

BATJ 面试题目

欢迎大家加入 Java 高级架构/互联网：570210627

1，应该怎么封装简历才有 BATJ 面试机会?

2，HashMap 底层执行原理，

3，hashtable 和 ConcurrentHashMap 如何实现线程安全？

4，jvm 的内存布局，垃圾回收机制

5，类加载机制里的，双亲委派模型

6，阐述事务的隔离级别和传播属性

7，高并发下，如何做到安全的修改同一行数据？

8，A 服务调用 B 服务多接口，响应时间最短方案；

9，A 系统给 B 系统转 100 块钱，如何实现？

10，动态代理的几种实现方式及优缺点

11，多线程下读概率远远大于写概率，如何解决并发问题？

12，按线程池内部机制，当提交新任务时，有哪些异常要考虑？

13，@Transaction 注解一般写在什么位置?如何控制其回滚?

14，说说 Spring 的 IOC 容器初始化流程？

15，说说 springboot 启动机制

16，Redis 高性能的原因大概可以讲一些?

17，你是怎么控制缓存的更新？(被动方式/主动方式/增量/全量)？

18，浅析 Http 和 https 的三次握手有什么区别。

19，谈谈 Session/cookie 机制，如何实现会话跟踪？

20，什么是一致性 hash？

21，MQ 有可能发生重复消费，如何避免，如何做到幂等？

22，如何做限流策略，令牌桶和漏斗算法的使用场景？

蚂蚁金服面试

一面

hashmap 的实现原理，多线程并发操作 hashmap 会有什么问题？
原理简述：使用数组加链表的数据结构,根据给出的 key -hash 到数组的一个下标，如果当前下标有值建立一个链表 指向 next， 注意的是新插入的值会在头链表上， 这样的设计思路是 新数据默认更热
发产生的问题：多线程 put 后可能导致 get 死循环

多线程 put 的时候可能导致元素丢失
主要问题出在 addEntry 方法的 new Entry (hash, key, value, e)，如果两个线程都同时取得了 e,则他们下一个元素都是 e，然后赋值给 table 元素的时候有一个成功有一个丢失。
put 非 null 元素后 get 出来的却是 null

泛型的反射擦除

- 泛型只在编译期有效，在运行期会被擦除掉， 所以通过反射的反射往一个要求泛型的集合添加对象，逃过了编译器的检查，
  是可以正常在运行期使用的

乐观锁了解哪些

B+树和红黑树时间复杂度

二叉树的遍历
前序遍历：根左右
中序遍历：左根右
后续遍历：左右根
算法-java

```
public  void  theFirstTraversal(Node  node){

      printNode(node);

      if(node.getLeftNode()!=null){
```

```
            theFirstTraversal(node.getLeftNode());
        }
        if(node.getRightNode()!=null){
            theFirstTraversal(node.getRightNode());
        }
    }

    public  void  thePostOrderTraversal(Node  node){
        if(node.getLeftNode()!=null){
            thePostOrderTraversal(node.getLeftNode());
        }
        if(node.getRightNode()!=null){
            thePostOrderTraversal(node.getRightNode());
        }
        printNode(node);
    }
    public  void  theInOrderTraversal(Node  node){
        if(node.getLeftNode()!=null){
            theInOrderTraversal(node.getLeftNode());
        }
        printNode(node);
```

```
        if(node.getRightNode()!=null){

            theInOrderTraversal(node.getRightNode());

        }

    }
```

快排的时间复杂度，冒泡时间复杂度，快排是否稳定，快排的过程

讲一下 spring 的启动流程
spring 启动有几种方法，通过 web.xml 的监听器，或者 ClassPathXmlApplicationContext，FileSystemXmlApplicationContext，它启动的核心目标是构建 spring 容器，构建容器需要做以下几个工作 ，实例化对象，
加载配置，装配 bean 对象，
首先根据路径加载配置文件，也叫元数据 和 pojo 对象，根据对应的模板表情进行初始化操作，通过元数据和反射的方式生成对象放入容器中，load 的配置也进行初始化，后面会有 bean 的装配工作和回调工作，
比如实现了 InitializingBean 会在容器所有需要设置属性的 bean 设置完以后调用 afterPropertiesSet
this.postProcessBeanFactory(beanFactory);
this.invokeBeanFactoryPostProcessors(beanFactory);
this.registerBeanPostProcessors(beanFactory);
this.initMessageSource();
this.initApplicationEventMulticaster();
this.onRefresh();
this.registerListeners();
this.finishBeanFactoryInitialization(beanFactory);
this.finishRefresh();

AOP 是怎么实现的？两种动态代理的应用场景？

通过动态生成字节码，增强代码业务功能，
JDK 动态代理， 需要代理的类必须要实现接口
cglib 无需实现接口 速度快
javaassist 也无需实现接口

Java 中的新生代和老年代的垃圾回收算法，对应的垃圾收集器

谈谈 synchronized、ReenTrantLock、volatile 三者的区别
synchronized 互斥锁，
即操作互斥，并发线程过来，串行获得锁，串行执行代码。就像一个房间一把钥匙，一个人进去后，下一个人得等第一个人出来得到钥匙才能进入。如果代码写的不好（A），可能出现死锁！（A 得到锁，B 等待 A 释放锁，A 不释放，B 死锁）.

ReenTrantLock 重入锁
在 JDK 优化之后 synchronized 和 reentrantLCOK 锁的性能其实差不多
唯一的区别是

```
1、ReenTrantLock 可以指定是公平锁还是非公平锁。而 synchronized 只能是非公平锁。所谓的公平锁就是先等待的线程先获得锁。

2、ReenTrantLock 提供了一个 Condition（条件）类，用来实现分组唤醒需要唤醒的线程们，而不是像 synchronized 要么随机唤醒一个线程要么唤醒全部线程。

3、 ReenTrantLock 提供了一种能够中断等待锁的线程的机制，通过 lock.lockInterruptibly()来实现这个机制。
```

如果需要使用 reentrantlock 的这三种独立功能可以使用这个锁 并且 ReenTrantLock 的力度更细

volatile 是一个关键字 对虚拟机声明标识有该关键字的遍历 不允许重排序和修改马上刷新内存~

示例代码：

二面

[Linux]命令 统计，排序，前几问题等
wc、sort、uniq 这三个命令都是用于排序。
wc 命令很简单，在 linux 中用来统计文件中的字节数、字数、行数，并且将结果返回
格式：wc [-clmwL] file
-c：统计字节数
-l：统计行数
-m：统计字符数
-w：统计字数，一个字被定义为由空白、跳格或换行字符分隔的字符串
-L：打印最长行的长度
https://blog.csdn.net/feng973/article/details/73849586

full gc 的发生有哪几种情况？

触发 MinorGC(Young GC)

```
虚拟机在进行 minorGC 之前会判断老年代最大的可用连续空间是否大于新生代的所有对象总空间

1、如果大于的话，直接执行 minorGC

2、如果小于，判断是否开启 HandlerPromotionFailure，没有开启直接 FullGC

3、如果开启了 HanlerPromotionFailure，JVM 会判断老年代的最大连续内存空间是否大于历次晋升的大小，如果小于直接执行 FullGC

4、如果大于的话，执行 minorGC
```

触发 FullGC

- 老年代空间不足 如果创建一个大对象，Eden 区域当中放不下这个大对象，会直接保存在老年代当中，如果老年代空间也不足，就会触发 Full GC。为了避免这种情况，最好就是不要创建太大的对象。
- 持久代空间不足
  如果有持久代空间的话，系统当中需要加载的类，调用的方法很多，同时持久代当中没有足够的空间，就出触发一次 Full GC
- YGC 出现 promotion failure
  promotion failure 发生在 Young GC，如果 Survivor 区当中存活对象的年龄达到了设定值，会就将 Survivor 区当中的对象拷贝到老年代，如果老年代的空间不足，就会发生 promotion failure， 接下去就会发生 Full GC.
- 统计 YGC 发生时晋升到老年代的平均总大小大于老年代的空闲空间
  在发生 YGC 是会判断，是否安全，这里的安全指的是，当前老年代空间可以容纳 YGC 晋升的对象的平均大小，如果不安全，就不会执行 YGC,转而执行 Full GC。
- 显示调用 System.gc

Netty 内部结构
mysql 有几种引擎，区别？

innodb 中聚集索引，非聚集索引

分布式下 redis 如何保证线程安全

MySQL、Redis、MongoDB 各自的应用场景

Redis 的存储数据结构，以及持久化区别

三面

什么情况索引不会命中，会造成全表扫描

1.应尽量避免在 where 子句中对字段进行 null 值判断，否则将导致引擎放弃使用索引而进行全表扫描，如：

1. 应尽量避免在 where 子句中使用!=或<>操作符，否则将引擎放弃使用索引而进行全表扫描。

   3.应尽量避免在 where 子句中使用 or 来连接条件，否则将导致引擎放弃使用索引而进行全表扫描，如：

   4.in 和 not in 也要慎用，否则会导致全表扫描

   5.'%abc%' like 全匹配查询也会导致

   6.应尽量避免在 where 子句中对字段进行函数操作，这将导致引擎放弃使用索引而进行全表扫描

JVM 性能调优会涉及哪些参数?

Restful、SOAP、RPC、SOA、微服务之间的区别
SpringClound 与 Dubbo 的比较，优劣势
mysql 如何实现如何实现分库分表+动态数据源+读写分离
什么是缓存雪崩？服务器雪崩的场景与解决方案
分布式锁的方案，redis 和 zookeeper 哪个好，如果是集群部署，高并发情况下哪个性能更好。
请画一个大型网站分布式集群部署图
秒杀系统的架构设计

Java 程序员从阿里拿到 offer 回来，这些面试题你会吗？

前不久刚从阿里面试回来，为了这场面试可以说准备了一个半月，做的准备就是刷题和看视频看书充实自己的技术，话说是真难啊，不过还算顺利拿到了 offer，有很多面试题我已经记不起来了，这些是当天回家整理好的，下面我来跟大家一起分享一下。

首先我们需要明白一个事实，招聘的一个很关键的因素是在给自己找未来的同事，同级别下要找比自己优秀的人，面试是一个双向选择的过程，也是一个将心比心去沟通的过程。

和以前一样，只有问题没有参考答案，需要各位小伙伴下来逐一学习！

一、开场白

简单的介绍一下自己的工作经历与职责，在校或者工作中主要的工作内容，主要负责的内容；（你的信息一清二白的写在简历上，这个主要为了缓解面试者的压力）

介绍下自己最满意的，有技术亮点的项目或平台，重点介绍下自己负责那部分的技术细节；（主要考察应聘者对自己做过的事情是否有清晰的描述，判断做的事情的复杂度）

二、Java 多线程

线程池的原理，为什么要创建线程池？创建线程池的方式；

线程的生命周期，什么时候会出现僵死进程；

说说线程安全问题，什么实现线程安全，如何实现线程安全；

创建线程池有哪几个核心参数？ 如何合理配置线程池的大小？

volatile、ThreadLocal 的使用场景和原理；

ThreadLocal 什么时候会出现 OOM 的情况？为什么？

synchronized、volatile 区别、synchronized 锁粒度、模拟死锁场景、原子性与可见性；

三、JVM 相关

JVM 内存模型，GC 机制和原理；

GC 分哪两种，Minor GC 和 Full GC 有什么区别？什么时候会触发 Full GC？分别采用什么算法？

JVM 里的有几种 classloader，为什么会有多种？

什么是双亲委派机制？介绍一些运作过程，双亲委派模型的好处；

什么情况下我们需要破坏双亲委派模型；

常见的 JVM 调优方法有哪些？可以具体到调整哪个参数，调成什么值？

JVM 虚拟机内存划分、类加载器、垃圾收集算法、垃圾收集器、class 文件结构是如何解析的；

四、Java 扩展篇

红黑树的实现原理和应用场景；

NIO 是什么？适用于何种场景？

Java9 比 Java8 改进了什么；

HashMap 内部的数据结构是什么？底层是怎么实现的？（还可能会延伸考察 ConcurrentHashMap 与 HashMap、HashTable 等，考察对技术细节的深入了解程度）；

说说反射的用途及实现，反射是不是很慢，我们在项目中是否要避免使用反射；

说说自定义注解的场景及实现；

List 和 Map 区别，Arraylist 与 LinkedList 区别，ArrayList 与 Vector 区别；

五、Spring 相关

Spring AOP 的实现原理和场景？

Spring bean 的作用域和生命周期；

Spring Boot 比 Spring 做了哪些改进？ Spring 5 比 Spring4 做了哪些改进；

如何自定义一个 Spring Boot Starter？

Spring IOC 是什么？优点是什么？

SpringMVC、动态代理、反射、AOP 原理、事务隔离级别；

六、中间件篇

Dubbo 完整的一次调用链路介绍；

Dubbo 支持几种负载均衡策略？

Dubbo Provider 服务提供者要控制执行并发请求上限，具体怎么做？

Dubbo 启动的时候支持几种配置方式？

了解几种消息中间件产品？各产品的优缺点介绍；

消息中间件如何保证消息的一致性和如何进行消息的重试机制？

Spring Cloud 熔断机制介绍；

Spring Cloud 对比下 Dubbo，什么场景下该使用 Spring Cloud？

七、数据库篇

锁机制介绍：行锁、表锁、排他锁、共享锁；

乐观锁的业务场景及实现方式；

事务介绍，分布式事物的理解，常见的解决方案有哪些，什么事两阶段提交、三阶段提交；

MySQL 记录 binlog 的方式主要包括三种模式？每种模式的优缺点是什么？

MySQL 锁，悲观锁、乐观锁、排它锁、共享锁、表级锁、行级锁；

分布式事务的原理 2 阶段提交，同步\异步\阻塞\非阻塞；

数据库事务隔离级别，MySQL 默认的隔离级别、Spring 如何实现事务、JDBC 如何实现事务、嵌套事务实现、分布式事务实现；

SQL 的整个解析、执行过程原理、SQL 行转列；

八、Redis

Redis 为什么这么快？redis 采用多线程会有哪些问题？

Redis 支持哪几种数据结构；

Redis 跳跃表的问题；

Redis 单进程单线程的 Redis 如何能够高并发?

Redis 如何使用 Redis 实现分布式锁？

Redis 分布式锁操作的原子性，Redis 内部是如何实现的？

九、其他

看过哪些源代码？然后会根据你说的源码问一些细节的问题？（这里主要考察面试者是否对技术有钻研的精神，还是只停留在表面，还是背了几道面经，这个对于很多有强迫症的面试官，如果你连源码都没看过，基本上是会 pass 掉的，比如我也是这样的！）

项目中遇到了哪些比较有挑战性的问题，是如何解决的；（这个很有争议，一方面是你连一个复杂的问题都解决不了，要你过来干什么，还有就是我的能力牛逼啊，但是公司没有业务场景让我展示啊！这个就看你遇到的面试官了，祝你好运！）