**应聘者面试前，应多刷题、多思考**

**对于刷题，这里有两大误区：一是不按照岗位海量刷题，平白浪费了精力；二是背下大量答案**

[Mysql优化、索引的实现 1](#_Toc20956)

[讲几个设计模式，哪些地方用到了，为什么要用 1](#_Toc29382)

[HTTP有没有状态 1](#_Toc26706)

[Cookie和Session的区别及使用 1](#_Toc15940)

[如何给一个双向链表排序？ 2](#_Toc6751)

## Mysql优化、索引的实现

<https://www.cnblogs.com/yuyue2014/p/3662005.html>

## 讲几个设计模式，哪些地方用到了，为什么要用

## HTTP有没有状态

http协议无状态中的【状态】到底指的是什么？

先来看这句话的另外两个概念：（标准的http协议是无状态的，无连接的）

<https://www.cnblogs.com/bellkosmos/p/5237146.html>

1、 标准的http协议指的是不包括cookies, session，application的http协议，他们都不属于标准协 议，虽然各种网络应用提供商，实现语言、web容器等，都默认支持它

2、 无连接指的是什么

每一个访问都是无连接，服务器挨个处理访问队列里的访问，处理完一个就关闭连接，这事儿就完了， 然后处理下一个新的

无连接的含义是限制每次连接只处理一个请求。服务器处理完客户的请求，并收到客户的应答后，即 断开连接

http协议是无状态的，但cookie和session实现了有状态这个功能，这个状态是指，**客户端和服务器在临时会话中产生的数据**！

## Cookie和Session的区别及使用

<https://blog.csdn.net/liyifan687/article/details/80077928>

## 如何给一个双向链表排序？

## redis的主从复制怎么做的？

## 进程和线程的区别？（什么样资源在两个线程共享）

——进程是资源分配的最小单位，线程是cpu调度的最小单元

——一个进程有多个线程，多个线程可以并发执行

——进程的创建退出操作系统为其分配资源，切换线程需要保存这些资源，进程切换开销大，同一个进程的线程共享该进程的资源，故线程仅仅有一些自己独立的寄存器堆栈资源，线程之间切换开销小

——进程之前有专门的通信方式：管道、信号量、消息队列、共享内存、socket套接字，线程之间通信主要靠锁和信号量来完成。

## 单核上是否需要多线程

纯计算不要开多线程。有 io 的话性能瓶颈可能不在 cpu 上

CPU 快， IO 慢，不多线程， CPU 效率不高，操作系统效率也不高。

## 单核机器上写多线程程序，是否需要考虑加锁？

需要，你的非线程安全的代码执行了一半，这时一个时钟中断信号来了，切到别的线程去了，然后另一个线程就把你这个线程用的变量破坏了

单核线程冲突的概率应该会小一些，但不会消失，毕竟不是原子性的操作会有可能被分配到 CPU 的两个时间片被两个线程分别抢占。  
其次，在多核 CPU 出现之前多线程的同步问题就出现了。

除非你的单核并发是不允许抢占式的，否则一样会产生线程不安全

## 写代码，去掉字符串中的空格空格

#include <iostream>

using namespace std;

int main()

{

char str[40] = " abc 123 456 ";

int num = 0;

int i;

for(i = 0; str[i] != '\0'; ++i)

{

if(str[i] == ' ')

++num;

else

str[i-num] = str[i];

}

str[i-num] = '\0';

printf("%s\n",str);

}

## C++的多态

## 虚函数实现

## 协程

答：定义：协程是一种用户态的轻量级线程。

协程拥有自己的寄存器上下文和栈。协程调度切换时，将寄存器上下文和栈保存到其他地方，在切回来的时候，恢复先前保存的寄存器上下文和栈。因此：协程能保留上一次调用时的状态（即所有局部状态的一个特定组合），每次过程重入时，就相当于进入上一次调用的状态，换种说法：进入上一次离开时所处逻辑流的位置；

线程是抢占式，而协程是协作式；

协程的优点：

跨平台

跨体系架构

无需线程上下文切换的开销

无需原子操作锁定及同步的开销

方便切换控制流，简化编程模型

高并发+高扩展性+低成本：一个CPU支持上万的协程都不是问题。所以很适合用于高并发处理。

协程的缺点：

无法利用多核资源：协程的本质是个单线程,它不能同时将 单个CPU 的多个核用上,协程需要和进程配合才能运行在多CPU；

进行阻塞（Blocking）操作（如IO时）会阻塞掉整个程序：这一点和事件驱动一样，可以使用异步IO操作来解决。

## TCP握手与释放

## http和https的区别？

答：HTTPS协议是由SSL+HTTP协议构建的可进行加密传输、身份认证的网络协议，要比http协议安全。

HTTPS（Secure Hypertext Transfer Protocol）安全超文本传输协议，与http主要区别在于：

#http是超文本传输协议，信息是明文传输，https 则是具有安全性的ssl加密传输协议；

#http和https使用的是完全不同的连接方式用的端口也不一样,前者是80,后者是443；

## 虚拟内存的概念与介绍

## 单链表的反转算法

答：思想：创建3个指针，分别指向上一个节点、当前节点、下一个节点，遍历整个链表的同时，将正在访问的节点指向上一个节点，当遍历结束后，就同时完成了链表的反转。

ListNode\* ReverseList(ListNode\* pHead) {

ListNode \*p,\*q,\*r;

if(pHead==NULL || pHead->next==NULL){

return pHead;

}else{

p=pHead;

q=p->next;

pHead->next=NULL;

while(q!=NULL){

r=q->next;

q->next=p;

p=q;

q=r;

}

return p;

}

}

## 数据库索引

答：定义：数据库索引是对数据库表中一列或多列的值进行排序的一种结构，使用索引可快速访问数据库表中的特定信息；

举例：employee 表的人员编号列（id）就是数据库索引，select \* from employee where id=10000即可查找编号10000的人员信息。如果没有索引，必须遍历整个表直到id=10000；

数据库索引作用：

一，大大加快 数据的检索速度，这也是创建索引的最主要的原因；

二，保证数据库表中每一行数据的唯一性；

三，可以加速表和表之间的连接，特别是在实现数据的参考完整性方面特别有意义；

四，在使用分组和排序子句进行数据检索时，同样可以显著减少查询中分组和排序的时间；

五，通过使用索引，可以在查询的过程中，使用优化隐藏器，提高系统的性能。

数据库索引缺陷：

一，表的增删改查、创建索引和维护索引要耗费时间；

二，索引需要占物理空间；

## C++——4个默认成员函数详解

<https://blog.csdn.net/tianzez/article/details/79636250>

<https://www.cnblogs.com/chengkeke/p/5417362.html>

计算框架mpi

常见RPC框架

linux操作系统

加密算法与安全通信

## IM通讯机制

即时通讯

精通服务器逻辑框架，系统架构，性能分析，测试，安全性等技术，对海量、高性能、分布式开发

<https://www.cnblogs.com/secondtonone1/p/7076769.html>

<https://blog.csdn.net/yishizuofei/article/details/81843124>

## 常用的密码学算法和原理

<https://blog.csdn.net/williamfan21c/article/details/73881195>

<https://blog.csdn.net/Yj_sail/article/details/55261834>

<https://blog.csdn.net/xk_qian/article/details/40860655>

<https://blog.csdn.net/qq_34227896/article/details/84028740>

<https://www.cnblogs.com/LittleHann/p/3738141.html>

<https://www.cnblogs.com/sddai/p/5667472.html>

<https://www.cnblogs.com/wqhwe/p/5407468.html>

<https://www.cnblogs.com/wudaoyongchang/p/6253451.html>

## 分布式缓存系统(如memcached/redis)

**Memcache与Redis的比较**

（1）数据结构：Memcache只支持key value存储方式，Redis支持更多的数据类型，比如Key value、hash、list、set、zset；

（2）多线程：Memcache支持多线程，Redis支持单线程；CPU利用方面Memcache优于Redis；

（3）持久化：Memcache不支持持久化，Redis支持持久化；

（4）内存利用率：Memcache高，Redis低（采用压缩的情况下比Memcache高）；

（5）过期策略：Memcache过期后，不删除缓存，会导致下次取数据数据的问题，Redis有专门线程，清除缓存数据；