1. **什么是redis？**

Redis是一个开源的使用ANSI [C语言](https://baike.baidu.com/item/C%E8%AF%AD%E8%A8%80" \t "/home/anwc/文档\\x/_blank)编写、支持网络、可基于内存亦可持久化的日志型、Key-Value[数据库](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93/103728" \t "/home/anwc/文档\\x/_blank)，并提供多种语言的API。从2010年3月15日起，Redis的开发工作由VMware主持。从2013年5月开始，Redis的开发由Pivotal赞助。

redis是一个key-value[存储系统](https://baike.baidu.com/item/%E5%AD%98%E5%82%A8%E7%B3%BB%E7%BB%9F" \t "/home/anwc/文档\\x/_blank)。和Memcached类似，它支持存储的value类型相对更多，包括string(字符串)、list([链表](https://baike.baidu.com/item/%E9%93%BE%E8%A1%A8" \t "/home/anwc/文档\\x/_blank))、set(集合)、zset(sorted set --有序集合)和hash（哈希类型）。这些[数据类型](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E7%B1%BB%E5%9E%8B" \t "/home/anwc/文档\\x/_blank)都支持push/pop、add/remove及取交集并集和差集及更丰富的操作，而且这些操作都是原子性的。在此基础上，redis支持各种不同方式的排序。与memcached一样，为了保证效率，数据都是缓存在内存中。区别的是redis会周期性的把更新的数据写入磁盘或者把修改操作写入追加的记录文件，并且在此基础上实现了master-slave(主从)同步。

Redis 是一个高性能的key-value数据库。 redis的出现，很大程度补偿了[memcached](https://baike.baidu.com/item/memcached" \t "/home/anwc/文档\\x/_blank)这类key/value存储的不足，在部 分场合可以对关系数据库起到很好的补充作用。它提供了Java，C/C++，C#，PHP，JavaScript，Perl，Object-C，Python，Ruby，Erlang等客户端，使用很方便。 [1]

Redis支持主从同步。数据可以从主服务器向任意数量的从服务器上同步，从服务器可以是关联其他从服务器的主服务器。这使得Redis可执行单层树复制。存盘可以有意无意的对数据进行写操作。由于完全实现了发布/订阅机制，使得从数据库在任何地方同步树时，可订阅一个频道并接收主服务器完整的消息发布记录。同步对读取操作的可扩展性和数据冗余很有帮助。

redis的官网地址，非常好记，是redis.io。（特意查了一下，域名后缀io属于国家域名，是british Indian Ocean territory，即英属印度洋领地）

1. **Redis数据库都有哪些特点？**

单线程，利用redis队列技术并将访问变为串行访问，消除了传统数据库串行控制的开销

redis具有快速和持久化的特征，速度快，因为数据存在内存中。

分布式 读写分离模式

支持丰富数据类型

支持事务，操作都是原子性，所谓原子性就是对数据的更改要么全部执行，要不全部不执行。

可用于缓存，消息，按key设置过期时间，过期后自动删除

1. **Redis支持的数据类型都有哪些？**

Redis支持五种数据类型：string（字符串），hash（哈希），list（列表），set（集合）及zset(sorted set：有序集合)。

1. **为什么redis需要把所有数据放到内存中？**

Redis为了达到最快的读写速度将数据都读到内存中，并通过异步的方式将数据写入磁盘。所以Redis具有快速和数据持久化的特性。如果不将数据放到内存中，磁盘的I/O速度会严重影响redis的性能。在内存越来越便宜的今天，redis将会越来越受欢迎。如果设置了最大使用的内存，则数据已有记录数达到内存限值后将不能继续插入新值。

1. **简单说说虚拟内存**

虚拟内存别称[虚拟存储器](https://baike.baidu.com/item/%E8%99%9A%E6%8B%9F%E5%AD%98%E5%82%A8%E5%99%A8" \t "/home/anwc/文档\\x/_blank)（Virtual Memory）。[电脑](https://baike.baidu.com/item/%E7%94%B5%E8%84%91" \t "/home/anwc/文档\\x/_blank)中所运行的程序均需经由[内存](https://baike.baidu.com/item/%E5%86%85%E5%AD%98" \t "/home/anwc/文档\\x/_blank)执行，若执行的程序占用内存很大或很多，则会导致内存消耗殆尽。为解决该问题，[Windows](https://baike.baidu.com/item/Windows" \t "/home/anwc/文档\\x/_blank)中运用了虚拟[内存](https://baike.baidu.com/item/%E5%86%85%E5%AD%98" \t "/home/anwc/文档\\x/_blank)技术，即匀出一部分硬盘空间来充当内存使用。当内存耗尽时，电脑就会自动调用硬盘来充当内存，以缓解内存的紧张。若计算机运行程序或操作所需的[随机存储器](https://baike.baidu.com/item/%E9%9A%8F%E6%9C%BA%E5%AD%98%E5%82%A8%E5%99%A8" \t "/home/anwc/文档\\x/_blank)([RAM](https://baike.baidu.com/item/RAM" \t "/home/anwc/文档\\x/_blank))不足时，则 Windows 会用[虚拟存储器](https://baike.baidu.com/item/%E8%99%9A%E6%8B%9F%E5%AD%98%E5%82%A8%E5%99%A8" \t "/home/anwc/文档\\x/_blank)进行补偿。它将计算机的[RAM](https://baike.baidu.com/item/RAM" \t "/home/anwc/文档\\x/_blank)和[硬盘](https://baike.baidu.com/item/%E7%A1%AC%E7%9B%98" \t "/home/anwc/文档\\x/_blank)上的临时空间组合。当RAM运行速率缓慢时，它便将数据从RAM移动到称为“[分页](https://baike.baidu.com/item/%E5%88%86%E9%A1%B5" \t "/home/anwc/文档\\x/_blank)文件”的空间中。将数据移入[分页](https://baike.baidu.com/item/%E5%88%86%E9%A1%B5" \t "/home/anwc/文档\\x/_blank)文件可释放RAM，以便完成工作。 一般而言，计算机的RAM[容量](https://baike.baidu.com/item/%E5%AE%B9%E9%87%8F" \t "/home/anwc/文档\\x/_blank)越大，程序运行得越快。若计算机的速率由于RAM可用空间匮乏而减缓，则可尝试通过增加虚拟内存来进行补偿。但是，计算机从RAM读取数据的速率要比从硬盘读取数据的速率快，因而扩增RAM[容量](https://baike.baidu.com/item/%E5%AE%B9%E9%87%8F" \t "/home/anwc/文档\\x/_blank)（可加[内存条](https://baike.baidu.com/item/%E5%86%85%E5%AD%98%E6%9D%A1" \t "/home/anwc/文档\\x/_blank)）是最佳选择。

虚拟内存是Windows 为作为内存使用的一部分硬盘空间。虚拟内存在硬盘上其实就是为一个硕大无比的文件，文件名是[PageFile.Sys](https://baike.baidu.com/item/PageFile.Sys" \t "/home/anwc/文档\\x/_blank)，通常状态下是看不到的。必须关闭资源管理器对系统文件的保护功能才能看到这个文件。虚拟内存有时候也被称为是“页面文件”就是从这个文件的文件名中来的。

内存在计算机中的作用很大，电脑中所有运行的程序都需要经过内存来执行，如果执行的程序很大或很多，就会导致内存消耗殆尽。为了解决这个问题，WINDOWS运用了虚拟内存技术，即拿出一部分硬盘空间来充当内存使用，这部分空间即称为[虚拟内存](https://baike.baidu.com/item/%E8%99%9A%E6%8B%9F%E5%86%85%E5%AD%98" \t "/home/anwc/文档\\x/_blank)，虚拟内存在硬盘上的存在形式就是 PAGEFILE.SYS这个页面文件。

1. **使用redis都有哪些好处？**

答案见第2题

1. **Redis相比memcached有哪些优势？**

（1）memcached所有的值都是简单的字符串，redis作为其代替者，支持更为丰富的数据类型。

（2）redis的速度比memcached快很多。

（3）redis可以持久化其数据。

1. **Redis中有哪些常见的性能问题？对应都有哪些解决方案？**

1.Master写内存快照

save命令调度rdbSave函数，会阻塞主线程的工作，当快照比较大时对性能影响是非常大的，会间断性暂停服务，所以Master最好不要写内存快照。

2.Master AOF持久化

如果不重写AOF文件，这个持久化方式对性能的影响是最小的，但是AOF文件会不断增大，AOF文件过大会影响Master重启的恢复速度。

3.Master调用BGREWRITEAOF

Master调用BGREWRITEAOF重写AOF文件，AOF在重写的时候会占大量的[CPU](https://www.baidu.com/s?wd=CPU&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd" \t "/home/anwc/文档\\x/_blank)和内存资源，导致服务load过高，出现短暂服务暂停现象。

4.Redis主从复制的性能问题

第一次Slave向Master同步的实现是：Slave向Master发出同步请求，Master先dump出rdb文件，然后将rdb文件全量传输给slave，然后Master把缓存的命令转发给Slave，初次同步完成。第二次以及以后的同步实现是：Master将变量的快照直接实时依次发送给各个Slave。不管什么原因导致Slave和Master断开重连都会重复以上过程。Redis的主从复制是建立在内存快照的持久化基础上，只要有Slave就一定会有内存快照发生。虽然Redis宣称主从复制无阻塞，但由于Redis使用单线程服务，如果Master快照文件比较大，那么第一次全量传输会耗费比较长时间，且文件传输过程中Master可能无法提供服务，也就是说服务会中断，对于关键服务，这个后果也是很可怕的。

以上1.2.3.4根本问题的原因都离不开系统io瓶颈问题，也就是硬盘读写速度不够快，主进程 fsync()/write() 操作被阻塞。

5.单点故障问题

由于目前Redis的主从复制还不够成熟，所以存在明显的单点故障问题，这个目前只能[自己做](https://www.baidu.com/s?wd=%E8%87%AA%E5%B7%B1%E5%81%9A&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd" \t "/home/anwc/文档\\x/_blank)方案解决，如：主动复制，Proxy实现Slave对Master的替换等，这个也是Redis作者目前比较优先的任务之一，作者的解决方案思路简单优雅，详情可见 Redis Sentinel design draft [http://redis.io/topics/sentinel-spec](http://redis.io/topics/sentinel-spec" \t "/home/anwc/文档\\x/_blank)。

总结

Master最好不要做任何持久化工作，包括内存快照和AOF日志文件，特别是不要启用内存快照做持久化。

如果数据比较关键，某个Slave开启AOF备份数据，策略为每秒同步一次。

为了主从复制的速度和连接的稳定性，Slave和Master最好在同一个局域网内。

尽量避免在压力较大的主库上增加从库

为了Master的稳定性，主从复制不要用图状结构，用单向链表结构更稳定，即主从关系为：Master<–Slave1<–Slave2<–Slave3…….，这样的结构也方便解决单点故障问题，实现Slave对Master的替换，也即，如果Master挂了，可以立马启用Slave1做Master，其他不变。

1. **MySQL里有2000w数据，redis只有20w数据，如何保证redis中的数据都是热点数据？**

redis 提供 6种数据淘汰策略：

voltile-lru：从已设置过期时间的数据集（server.db[i].expires）中挑选最近最少使用的数据淘汰  
 volatile-ttl：从已设置过期时间的数据集（server.db[i].expires）中挑选将要过期的数据淘汰  
 volatile-random：从已设置过期时间的数据集（server.db[i].expires）中任意选择数据淘汰  
 allkeys-lru：从数据集（server.db[i].dict）中挑选最近最少使用的数据淘汰  
 allkeys-random：从数据集（server.db[i].dict）中任意选择数据淘汰  
 no-enviction（驱逐）：禁止驱逐数据

1. **Redis集群方案应该怎么做？都有哪些方案？**

详细答案参见以下博客:

<https://blog.csdn.net/kingice1014/article/details/53493287>

1. **Redis都有哪些适合的场景？**

会话缓存（Session Cache）:用Redis缓存会话比其他存储（如memcached）的优势在于：redis提供持久化。当维护一个不是严格要求一致性的缓存时，如果用户的购物车信息全部丢失，大部分人都会不高兴。

全页缓存（FPC）:除基本的会话token之外，Redis还提供很简便的FPC平台。

队列：Redis在内存存储引擎领域的一大优点是提供list和set操作，这使得Redis能作为一个很好的消息队列平台来使用。Redis作为队列使用的操作，就类似于本地程序语言对list的push/pop操作。

排行榜/计数器：Redis在内存中对数据进行递增递减的操作实现的非常好。集合（Set）和有序集合（Sorted Set）也使得我们在执行这些操作的时候变得非常简单，Redis只是正好提供了这两种数据结构。

订阅/发布

1. **如何测试redis的连通性？**

import redis  
 5   
 6 **'''  
 7 这种连接是连接一次就断了，耗资源.端口默认6379，就不用写  
 8 r = redis.Redis(host='127.0.0.1',port=6379,password='tianxuroot')  
 9 r.set('name','root')  
10   
11 print(r.get('name').decode('utf8'))  
12 '''**13 **'''  
14 连接池：  
15 当程序创建数据源实例时，系统会一次性创建多个数据库连接，并把这些数据库连接保存在连接池中，当程序需要进行数据库访问时，  
16 无需重新新建数据库连接，而是从连接池中取出一个空闲的数据库连接  
17 '''**18 pool = redis.ConnectionPool(host=**'127.0.0.1'**,password=**'helloworld'**) #实现一个连接池  
19   
20 r = redis.Redis(connection\_pool=pool)  
21 r.set(**'foo'**,**'bar'**)  
22 print(r.get(**'foo'**).decode(**'utf8'**))

1. **Redis事务相关的命令有哪些？**

详细答案参见博客:

<https://blog.csdn.net/m0_38132420/article/details/78481683>

1. **Redis是单线程的，如何提高多核CPU的利用率？**

edis的读取和处理性能非常强大，一般[服务器](https://www.baidu.com/s?wd=%E6%9C%8D%E5%8A%A1%E5%99%A8&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd" \t "/home/anwc/文档\\x/_blank)的cpu都不会是性能瓶颈。redis的性能瓶颈主要集中在内存和网络方面。所以，如果使用的redis命令多为O(N)、O(log(N))时间复杂度，那么基本上不会出现cpu瓶颈的情况。   
但是如果你确实需要充分使用多核cpu的能力，那么需要在单台服务器上运行多个redis实例(主从部署/集群化部署)，并将每个redis实例和[cpu内核](https://www.baidu.com/s?wd=cpu%E5%86%85%E6%A0%B8&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd" \t "/home/anwc/文档\\x/_blank)进行绑定

1. **Redis的内存占用情况怎么样？如何降低redis的内存使用情况呢？**

详细答案参见博客:

<https://www.cnblogs.com/zhaodahai/p/6824389.html>