## Redis 面试题汇总

#### 1.Redis 使用场景有哪些？

答：Redis 使用场景如下：

* 记录帖子点赞数、点击数、评论数
* 缓存近期热帖
* 缓存文章详情信息
* 记录用户会话信息

#### 2.Redis 有哪些功能？

答：Redis 功能如下：

* 数据缓存功能
* 分布式锁的功能
* 支持数据持久化
* 支持事务
* 支持消息队列

#### 3.Redis 支持哪些数据类型？

答：Redis 支持的数据类型如下：

* String 字符串
* List 列表
* Set 无序集合
* ZSet 有序集合
* Hash 哈希类型

#### 4.Redis 相比 Memcached 有哪些优势？

答：Redis 相比 Memcached 优势如下：

* Memcached 所有的值均是简单的字符串，Redis 支持更为丰富的数据类型
* Redis 的速度比 Memcached 要快
* Redis 可以持久化
* Redis 可以设置过期时间
* Redis 支持主从同步

#### 5.Redis 支持哪些淘汰策略？

答：Redis 淘汰策略如下：

* noeviction：禁止淘汰数据；
* allkeys-lru：尝试回收最少使用的键，使得新添加的数据有空间存放；
* volatile-lru：尝试回收最少使用的键，但仅限于在过期集合的键，使得新添加的数据有空间存放；
* allkeys-random：回收随机的键使得新添加的数据有空间存放；
* volatile-random：回收随机的键使得新添加的数据有空间存放，但仅限于在过期集合的键；
* volatile-ttl：回收在过期集合的键，并且优先回收存活时间较短的键，使得新添加的数据有空间存放。

#### 6.官方为什么不支持 Windows 版本？

答：Redis 官方是不支持 Windows 版的，因为目前 Linux 版本已经相当稳定，如果开发 Windows 版本，反而会带来兼容性等问题。

#### 7.为什么 Redis 是单线程的？

答：因为 Redis 的瓶颈最有可能是机器内存或者网络带宽，而非单线程，既然单线程不是 Redis 的性能瓶颈，并且单线程又比较容易实现，所以 Redis 就选择使用单线程来实现。

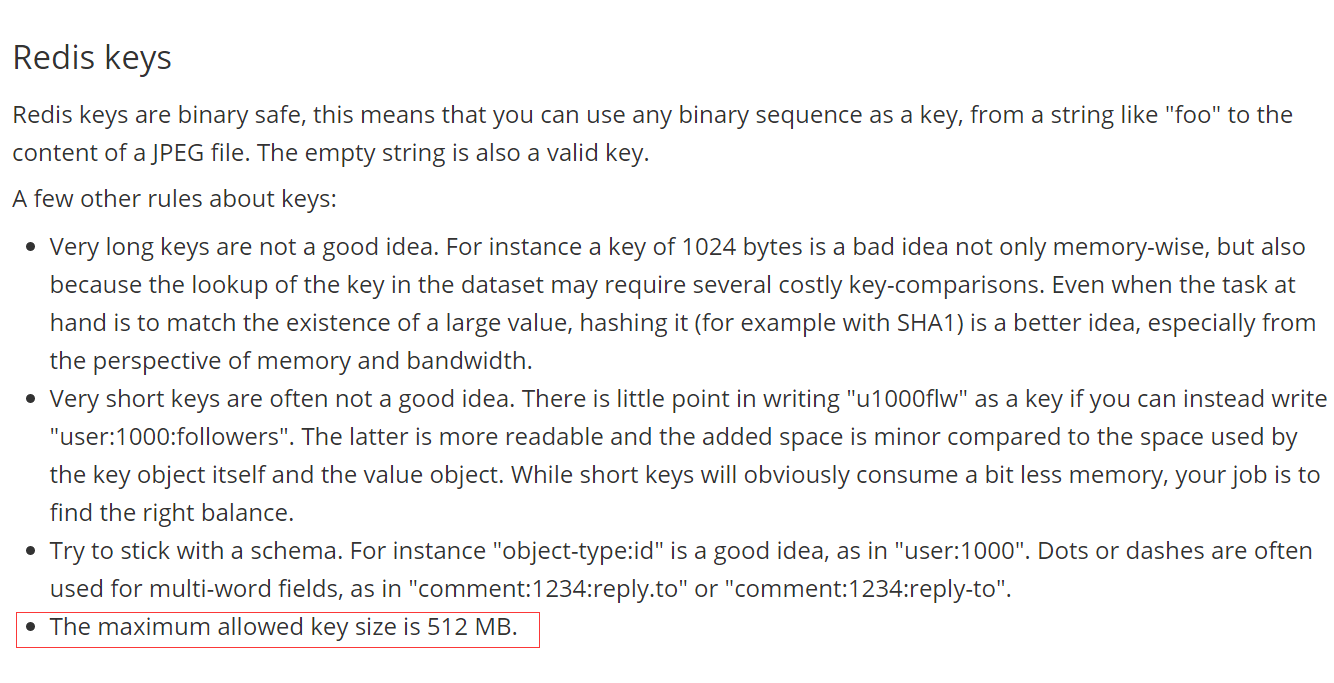
单线程并不代表运行速度就慢，比如，Nginx 和 NodeJs 都是单线程高性能的代表。

#### 8.为什么 Redis 需要把所有数据放到内存中？

答：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，这样 Redis 就拥有了快速查询和数据持久化等特征。

#### 9.在 Redis 中 key 的最大容量是多少？

答：最大容量 512 MB，官方说明如下图所示：



1

#### 10.Jedis 和 Redisson 有什么区别？

答：Jedis 和 Redisson 的区别如下：

* Jedis 是 Redis 的 Java 实现客户端，其 API 提供了比较全面的 Redis 命令的支持；
* Redisson 实现了分布式和可扩展的 Java 数据结构，和 Jedis 相比，功能较为简单，不支持字符串操作，不支持排序、事务、管道、分区等 Redis 特性。Redisson 的宗旨是促进使用者对 Redis 的关注分离，从而让使用者能够将精力更集中地放在处理业务逻辑上。

#### 11.Redis 如何设置过期时间？如何设置永久有效？

答：Redis 通过 expire() 方法设置过期时间，语法：redis.expire(key, expiration)。当 expire 的过期时间设置为 -1 时，表示永不过期。

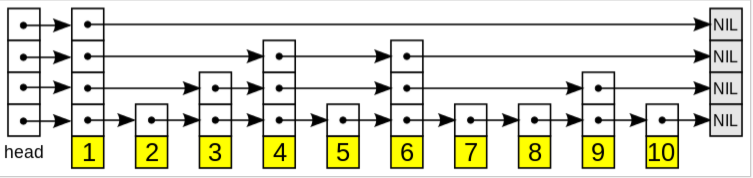
#### 12.如何保证 Redis 的数据一致性？

答：可使用以下方法来保证 Redis 的数据一致性：

* 合理设置缓存的过期时间；
* 新增、更改、删除数据库操作时同步更新 Redis，可以使用事物机制来保证数据的一致性。

#### 13.Redis 的数据结构是什么？它有什么优缺点？

答：Redis 的数据结构是跳跃表，跳跃表是一种基于链表的扩展，跳跃表还是一个链表，是一个有序的链表，在遍历的时候基于比较，但普通链表只能遍历，跳跃表加入了一个层的概念，层级越高元素越少，每次先从高层查找，直到找到合适的位置，从图中可以看到高层的节点远远少于底层的节点数，从而实现了跳跃式查找。



2

跳跃表优点：

* 实现比红黑树简单
* 比红黑树更容易扩展
* 红黑树插入删除时为了平衡高度需要旋转附近节点，高并发需要锁，跳跃表不需要考虑

跳跃表缺点：

* 比红黑树占用更多的内存，每个节点的大小取决于该节点层数
* 空间局部性差导致缓存命中率低，比红黑树略慢

#### 14.Redis 为什么用跳跃表来存储？

答：第一是因为红黑树存储比较复杂，调整涉及到多个节点的并发修改；第二是越接近根节点的地方越容易产生竞争，即使是不同叶子节点的操作由于平衡操作也可能逐级向上涉及到接近根的节点，而跳跃表可以用 CAS（Compare And Swap）来并发操作节点，比较容易实现，且更加局部化。

#### 15.什么是缓存穿透？如何解决？

答：缓存穿透是指查询一个一定不存在的数据，由于缓存中没有，因而每次需要从数据库中查询，但数据库也没有相应的数据，所以不会写入缓存，这就将导致每次请求都会去数据库查询，这种行为就叫缓存穿透。

解决方案是不管查询数据库是否有数据，都缓存起来，只不过把没有数据的缓存结果的过期时间设置为比较短的一个值，比如 3 分钟。

#### 16.什么是缓存雪崩，该如何解决？

答：指缓存由于某些原因，比如，宕机或者缓存大量过期等，从而导致大量请求到达后端数据库，进而导致数据库崩溃的情况。

解决缓存雪崩的方案如下：

* 分析业务功能，尽量让缓存的失效时间点均匀分布；
* 使用 Redis 主备，保证缓存系统的高可用。

#### 17.什么是缓存预热？有几种实现方式？

答：缓存预热是指系统上线后，将相关的缓存数据直接加载到缓存系统，这样就可以避免在用户请求的时候，先查询数据库，然后再将数据缓存的问题。

缓存预热的实现方式，可分为以下两种：

* 数据量不大的时候，工程启动的时候进行加载缓存动作；
* 数据量大的时候，设置一个定时任务脚本，进行缓存的刷新。

#### 18.在 Java 程序中如何使用 Redis？

答：在 Java 程序中可使用 Jedis 来操作 Redis，使用步骤如下：

1）添加 Jedis 引用

<dependency>  
 <groupId>redis.clients</groupId>  
 <artifactId>jedis</artifactId>  
 <version>x.x.x</version>  
</dependency>

2）连接并操作 Redis

Jedis jedis = new Jedis("127.0.0.1",6379);  
// 存值  
jedis.set("hello","world");  
// 取值  
jedis.get("hello");  
// 关闭连接  
jedis.close();

#### 19.什么是 Redis 持久化？如何进行 Redis 持久化？

答：Redis 持久化是指将内存中的数据保存在磁盘中，重启的时候可以再次加载进行使用。 Redis 有以下两种持久化方案：

* RDB（Redis DataBase）：是指在制定的时间间隔内将内存中的数据集快照写入磁盘；
* AOF（Append Only File）：该机制将以日志的形式记录服务器所处理的每一个写操作，在 Redis 服务器启动之初会读取该文件来重新构建数据库，以保证启动后数据库中的数据是完整的。

Redis 默认支持的持久化方式是 RDB 方式。

#### 20.RDB 和 AOF 有什么区别？

答：RDB 和 AOF 的区别如下：

* RDB 可能会引起一定程度的数据丢失，AOF 不会造成数据丢失
* RDB 的启动效率更高
* AOF 占用的空间比 RDB 大，AOF 的同步速度比 RDB 慢

总体来说如果对数据的完整性要求不高，RDB 是最好的解决方案，反之则选择 AOF。

#### 21.Redis 的监控工具都有哪些？

答：常用的 Redis 监控工具如下：

* Redis-stat：采用 Ruby 开发，基于 Redis 的 info 命令来统计，不影响 Redis 的性能；
* RedisLive：采用 Python 开发的可视化及查询分析工具，它是通过监控脚本来 Redis 提供的 MONITOR 命令从被监控 Redis 实例中获取数据，并存储到 Redis 的监控实例中。

#### 22.如何定位 Redis 的慢查询？

答：使用 slowlog get 来定位慢查询操作，如下所示：

127.0.0.1:6379> slowlog get

* 1. (integer) 0

1. (integer) 1565937939
2. (integer) 28003
   1. “lpush”

2) "list"

3) "1"

4) "2"

5) "6"

6) "3"

7) "4"

8) "9"

9) "8"

其中：

1. 表示慢查询记录 id
2. 表示发起命令的时间戳
3. 表示命令耗时，单位为微秒
4. 表示该条记录的命令及参数

#### 23.SAVE 和 BGSAVE 有什么区别？

答：SAVE 和 BGSAVE 都是用于 Redis 持久化的，它们的区别如下：

* SAVE 直接调用 rdbSave 函数（用于 Redis 持久化的函数），阻塞 Redis 主进程，直到保存完成为止，在主进程阻塞期间，服务器不能处理客户端的任何请求；
* BGSAVE 则会创建一个子进程，子进程负责调用 rdbSave 函数，并在保存完成之后向主进程发送完成信号，Redis 服务器在 BGSAVE 执行期间仍然可以继续处理客户端的请求。

#### 24.Redis 是如何实现同步的？

答：Redis 可以实现主从同步和从从同步。当第一次同步时，主节点做一次 BGSAVE，并同时将后续修改操作记录到内存中，待完成后将 RDB 文件全量同步到复制节点，复制节点接受完成后将 RDB 镜像加载到内存，加载完成后再通知主节点将期间修改的操作记录，同步到复制节点进行重放，这样就完成了同步过程。

#### 25.Redis 可以切换数据库吗？如何切换？

答：Redis 不像 MySQL 等关系型数据库那样有数据库的概念，不同的数据存在不同的数据库中，Redis 数据库是由一个整数索引标识，而不是一个数据库名称，默认情况下客户端连接到数据库 0，可以在配置文件中控制数据库总数，默认是 16 个。

可以使用 select index 来切换数据库，如下所示：

127.0.0.1:6379> select 0

OK

#### 26.Redis 有哪些集群策略？

答：Redis 集群策略有以下 3 种：

* 主从策略：1 台机器作为写操作，另外 2 台作为读操作，类似于 MySQL 的主从方式；
* 哨兵策略：增加 1 台机器作为哨兵，监控 3 台主从机器，当主节点挂机的时候，机器内部进行选举，从集群中从节点里指定一台机器升级为主节点，从而实现高可用。当主节点恢复的时候，加入到从节点中继续提供服务；
* 集群策略：Redis 3.0 之后增加了集群的概念，可实现多主多从的结构，实现真正的高可用。

#### 27.Redis 集群方案都有哪些？

答：Redis 集群实现方案如下：

* Twemproxy 是 Twitter 开源的 Redis 代理，它的使用和普通 Redis 完全一致，它会以一个代理的身份接收请求，并使用 hash 算法将请求转接到具体 Redis，将结果再返回 Twemproxy；
* Codis 是开源解决方案，也是目前用的最多的集群方案，基本和 Twemproxy 效果一致，但它支持在节点数量改变情况下，旧节点数据可恢复到新 hash 节点；
* Redis Cluster 是 Redis 3.0 自带的集群方案，特点在于他的分布式算法不是一致性 hash，而是 hash 槽的概念，以及自身支持节点设置从节点；
* 业务代码层实现，创建几个独立的 Redis 实例，在代码层对 key 进行 hash 计算，然后去对应的 Redis 实例操作数据。这种方式对 hash 层代码要求比较高，考虑部分包括，节点失效后的替代算法方案，数据震荡后的自动脚本恢复、实例的监控等。

#### 28.Redis 如何做内存优化？

答：把相关的信息整体存储，而不是把每个信息独立存储，这样就可以有效的减少内存使用。

#### 29.分布式锁设计时需要注意哪些事项？

答：通常分布式锁在设计时，需同时满足以下四个约束条件。

* 互斥性：在任意时刻，只有一个客户端能持有锁。
* 安全性：即不会形成死锁，当一个客户端在持有锁的期间崩溃而没有主动解锁的情况下，其持有的锁也能够被正确释放，并保证后续其他客户端能加锁。
* 可用性：就 Redis 而言，当提供锁服务的 Redis master 节点发生宕机等不可恢复性故障时，slave 节点能够升主并继续提供服务，支持客户端加锁和解锁；对基于分布式一致性算法实现的锁服务，如 ETCD 而言，当 leader 节点宕机时，follow 节点能够选举出新的 leader 继续提供锁服务。
* 对称性：对于任意一个锁，其加锁和解锁必须是同一个客户端，即，客户端 A 不能把客户端 B 加的锁给解了。

#### 30.Redis 集群实现的原理是什么？

答：集群的实现原理和集群的实现方式有关，如下所述：

* Redis Sentinal 着眼于高可用，在 Master 宕机时会自动将 Slave 提升为 Master，继续提供服务；
* Redis Cluster 着眼于扩展性，在单个 Redis 内存不足时，使用 Cluster 进行分片存储。

#### 31.Redis 常见的性能问题有哪些？

答：Redis 常见性能问题如下：

* 主服务器写内存快照，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，因此主服务器最好不要写内存快照；
* Redis 主从复制的性能问题，为了主从复制的速度和连接的稳定性，主从库最好在同一个局域网内。

更多java学习资料如下：

[Java高级架构师课程(总共122门课程，1460GB)](https://www.consultdog.com/index.php/front/44/)

[最强java面试视频课程(21门课程，126GB)](https://www.consultdog.com/index.php/front/78/)