



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

第三版：Redis 35 道

什么是 Redis?

Redis 是一个基于内存的高性能 key-value 数据库。

Redis 的特点

Redis 本质上是一个 Key-Value 类型的内存数据库，很像 Memcached，整个数据库统统加载在内存当中进行操作，定期通过异步操作把数据库数据 flush 到硬盘上进行保存。

因为是纯内存操作，Redis 的性能非常出色，每秒可以处理超过 10 万次读写操作，是已知性能最快的 Key-Value DB。

Redis 的出色之处不仅仅是性能，Redis 最大的魅力是支持保存多种数据结构，此外单个 value 的最大限制是 1GB，不像 Memcached 只能保存 1MB 的数据，因此 Redis 可以用来实现很多有用的功能。

比方说用他的 List 来做 FIFO 双向链表，实现一个轻量级的高性能消息队列服务，用他的 Set 可以做高性能的 tag 系统等等。另外 Redis 也可以对存入的 Key-Value 设置 expire 时间，因此也可以被当作一个功能加强版的 Memcached 来用。

Redis 的主要缺点是数据库容量受到物理内存的限制，不能用作海量数据的高性能读写，因此 Redis 适合的场景主要局限在较小数据量的高性能操作和运算上。

使用 Redis 有哪些好处?

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

- 1、速度快，因为数据存在内存中，类似于 HashMap，HashMap 的优势就是查找和操作的时间复杂度都是 $O(1)$
- 2、支持丰富数据类型，支持 string, list, set, sorted set, hash
- 3、支持事务，操作都是原子性，所谓的原子性就是对数据的更改要么全部执行，要么全部不执行
- 4、丰富的特性：可用于缓存，消息，按 key 设置过期时间，过期后将会自动删除

Redis 相比 Memcached 有哪些优势？

- 1、Memcached 所有的值均是简单的字符串，Redis 作为其替代者，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

Memcache 与 Redis 的区别都有哪些？

- 1、存储方式 Memecache 把数据全部存在内存之中，断电后会挂掉，数据不能超过内存大小。Redis 有部份存在硬盘上，这样能保证数据的持久性。
- 2、数据支持类型 Memcache 对数据类型支持相对简单。Redis 有复杂的数据类型。

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

3、使用底层模型不同 它们之间底层实现方式 以及与客户端之间通信的应用协议不一样。 Redis 直接自己构建了 VM 机制，因为一般的系统调用系统函数的话，会浪费一定的时间去移动和请求。

Redis 常见性能问题和解决方案：

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。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。

3、Master 调用 BGREWRITEAOF 重写 AOF 文件，AOF 在重写的时候会占大量的 CPU 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，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

MySQL 里有 2000w 数据，Redis 中只存 20w 的数据，如何保

证 Redis 中的数据都是热点数据

相关知识：Redis 内存数据集大小上升到一定大小的时候，就会施行数据淘汰策略（回收策略）。

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

Redis 提供 6 种数据淘汰策略：

- 1、 volatile-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
- 2、 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
- 3、 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
- 4、 allkeys-lru：从数据集（server.db[i].dict）中挑选最近最少使用的数据淘汰
- 5、 allkeys-random：从数据集（server.db[i].dict）中任意选择数据淘汰
- 6、 no- eviction（驱逐）：禁止驱逐数据

请用 Redis 和任意语言实现一段恶意登录保护的代码，

限制 1 小时内每用户 Id 最多只能登录 5 次。具体登录函数或功能用空函数即可，不用详细写出。

用列表实现：列表中每个元素代表登陆时间，只要最后的第 5 次登陆时间和现在时间差不超过 1 小时就禁止登陆。用 Python 写的代码如下：

```
#!/usr/bin/env python3
```

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

```
import Redis
import sys
import time

r = Redis.StrictRedis(host='127.0.0.1', port=6379, db=0)
try:
    id = sys.argv[1]
except:
    print( 'input argument error' )
    sys.exit(0)
if r.llen(id) >= 5 and time.time() - float(r.lindex(id, 4)) <= 3600:
    print( "you are forbidden logging" )
else:
    print( 'you are allowed to login' )
    r.lpush(id, time.time())
    # login_func()
```

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入内存中，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征。如果不将数据放在内存中，磁盘 I/O 速度为严重影响 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不能继续插入新值。

Redis 是单进程单线程的

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

Redis 利用队列技术将并发访问变为串行访问，消除了传统数据库串行控制的开销

Redis 的并发竞争问题如何解决？

Redis 为单进程单线程模式，采用队列模式将并发访问变为串行访问。Redis 本身没有锁的概念，Redis 对于多个客户端连接并不存在竞争，但是在 Redis 客户端对 Redis 进行并发访问时会发生连接超时、数据转换错误、阻塞、客户端关闭连接等问题，这些问题均是由于客户端连接混乱造成。

对此有 2 种解决方法：

- 1、客户端角度，为保证每个客户端间正常有序与 Redis 进行通信，对连接进行池化，同时对客户端读写 Redis 操作采用内部锁 `synchronized`。
- 2、服务器角度，利用 `setnx` 实现锁。

注：对于第一种，需要应用程序自己处理资源的同步，可以使用的方法比较通俗，可以使用 `synchronized` 也可以使用 `lock`；第二种需要用到 Redis 的 `setnx` 命令，但是需要注意一些问题。

Redis 事物的了解 CAS(check-and-set 操作实现乐观锁)？

和众多其它数据库一样，Redis 作为 NoSQL 数据库也同样提供了事务机制。在 Redis 中，`MULTI/EXEC/DISCARD/WATCH` 这四个命令是我们实现事务的基石。

相信对有关系型数据库开发经验的开发者而言这一概念并不陌生，即便如此，我们还是会简要的列出 Redis 中事务的实现特征：

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

- 1、在事务中的所有命令都将会被串行化的顺序执行，事务执行期间，Redis 不会再为其它客户端的请求提供任何服务，从而保证了事物中的所有命令被原子的执行。
- 2、和关系型数据库中的事务相比，在 Redis 事务中如果有某一条命令执行失败，其后的命令仍然会被继续执行。
- 3、我们可以通过 MULTI 命令开启一个事务，有关系型数据库开发经验的人可以将其理解为 "BEGIN TRANSACTION" 语句。在该语句之后执行的命令都将被视为事务之内的操作，最后我们可以通过执行 EXEC/DISCARD 命令来提交/回滚该事务内的所有操作。这两个 Redis 命令可被视为等同于关系型数据库中的 COMMIT/ROLLBACK 语句。
- 4、在事务开启之前，如果客户端与服务器之间出现通讯故障并导致网络断开，其后所有待执行的语句都不会被服务器执行。然而如果网络中断事件是发生在客户端执行 EXEC 命令之后，那么该事务中的所有命令都会被服务器执行。
- 5、当使用 Append-Only 模式时，Redis 会通过调用系统函数 write 将该事务内的所有写操作在本次调用中全部写入磁盘。然而如果在写入的过程中出现系统崩溃，如电源故障导致的宕机，那么此时也许只有部分数据被写入到磁盘，而另一部分数据却已经丢失。

Redis 服务器会在重新启动时执行一系列必要的一致性检测，一旦发现类似问题，就会立即退出并给出相应的错误提示。

此时，我们就要充分利用 Redis 工具包中提供的 Redis-check-aof 工具，该工具可以帮助我们定位到数据不一致的错误，并将已经写入的部分数据进行回滚。修复之后我们就可以再次重新启动 Redis 服务器了。

WATCH 命令和基于 CAS 的乐观锁：

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

在 Redis 的事务中，WATCH 命令可用于提供 CAS(check-and-set)功能。假设我们通过 WATCH 命令在事务执行之前监控了多个 Keys，倘若在 WATCH 之后有任何 Key 的值发生了变化，EXEC 命令执行的事务都将被放弃，同时返回 Null multi-bulk 应答以通知调用者事务

执行失败。例如，我们再次假设 Redis 中并未提供 incr 命令来完成键值的原子性递增，如果要实现该功能，我们只能自行编写相应的代码。其伪码如下：

```
val = GET mykey
val = val + 1
SET mykey $val
```

以上代码只有在单连接的情况下才可以保证执行结果是正确的，因为如果在同一时刻有多个客户端在同时执行该段代码，那么就会出现多线程程序中经常出现的一种错误场景--竞态争用(race condition)。

比如，客户端 A 和 B 都在同一时刻读取了 mykey 的原有值，假设该值为 10，此后两个客户端又均将该值加一后 set 回 Redis 服务器，这样就会导致 mykey 的结果为 11，而不是我们认为是 12。为了解决类似的问题，我们需要借助 WATCH 命令的帮助，见如下代码：

```
WATCH mykey
val = GET mykey
val = val + 1
MULTI
SET mykey $val
EXEC
```

和此前代码不同的是，新代码在获取 mykey 的值之前先通过 WATCH 命令监控了该键，此后又将 set 命令包围在事务中，这样就可以有效的保证每个连接在执行

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

EXEC 之前，如果当前连接获取的 mykey 的值被其它连接的客户端修改，那么当前连接的 EXEC 命令将执行失败。这样调用者在判断返回值后就可以获悉 val 是否被重新设置成功。

Redis 持久化的几种方式

1、快照 (snapshots)

缺省情况情况下，Redis 把数据快照存放在磁盘上的二进制文件中，文件名为 dump.rdb。你可以配置 Redis 的持久化策略，例如数据集中每 N 秒钟有超过 M 次更新，就将数据写入磁盘；或者你可以手工调用命令 SAVE 或 BGSAVE。

工作原理

1. Redis forks.
2. 子进程开始将数据写到临时 RDB 文件中。
3. 当子进程完成写 RDB 文件，用新文件替换老文件。
4. 这种方式可以使 Redis 使用 copy-on-write 技术。

2、AOF

快照模式并不十分健壮，当系统停止，或者无意中 Redis 被 kill 掉，最后写入 Redis 的数据就会丢失。

这对某些应用也许不是大问题，但对于要求高可靠性的应用来说，Redis 就不是一个合适的选择。Append-only 文件模式是另一种选择。你可以在配置文件中打开 AOF 模式

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

3、 虚拟内存方式

当你的 key 很小而 value 很大时,使用 VM 的效果会比较好.因为这样节约的内存比较大.

当你的 key 不小时,可以考虑使用一些非常方法将很大的 key 变成很大的 value,比如你可以考虑将 key,value 组合成一个新的 value.

vm-max-threads 这个参数,可以设置访问 swap 文件的线程数,设置最好不要超过机器的核数,如果设置为 0,那么所有对 swap 文件的操作都是串行的.可能会造成比较长时间的延迟,但是对数据完整性有很好的保证.

自己测试的时候发现用虚拟内存性能也不错。如果数据量很大，可以考虑分布式或者其他数据库。

Redis 的缓存失效策略和主键失效机制

作为缓存系统都要定期清理无效数据，就需要一个主键失效和淘汰策略。

在 Redis 当中，有生存期的 key 被称为 volatile。在创建缓存时，要为给定的 key 设置生存期，当 key 过期的时候（生存期为 0），它可能会被删除。

1、 影响生存时间的一些操作

生存时间可以通过使用 DEL 命令来删除整个 key 来移除，或者被 SET 和 GETSET 命令覆盖原来的数据，也就是说，修改 key 对应的 value 和使用另外相同的 key 和 value 来覆盖以后，当前数据的生存时间不同。

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

比如说，对一个 key 执行 INCR 命令，对一个列表进行 LPUSH 命令，或者对一个哈希表执行 HSET 命令，这类操作都不会修改 key 本身的生存时间。另一方面，如果使用 RENAME 对一个 key 进行改名，那么改名后的 key 的生存时间和改名前一样。

RENAME 命令的另一种可能是，尝试将一个带生存时间的 key 改名成另一个带生存时间的 another_key，这时旧的 another_key (以及它的生存时间)会被删除，然后旧的 key 会改名为 another_key，因此，新的 another_key 的生存时间也和原本的 key 一样。使用 PERSIST 命令可以在不删除 key 的情况下，移除 key 的生存时间，让 key 重新成为一个 persistent key。

2、如何更新生存时间

可以对一个已经带有生存时间的 key 执行 EXPIRE 命令，新指定的生存时间会取代旧的生存时间。过期时间的精度已经被控制在 1ms 之内，主键失效的时间复杂度是 $O(1)$ ，EXPIRE 和 TTL 命令搭配使用，TTL 可以查看 key 的当前生存时间。设置成功返回 1；当 key 不存在或者不能为 key 设置生存时间时，返回 0。

最大缓存配置：

在 Redis 中，允许用户设置最大使用内存大小，server.maxmemory 默认为 0，没有指定最大缓存，如果有新的数据添加，超过最大内存，则会使 Redis 崩溃，所以一定要设置。Redis 内存数据集大小上升到一定大小的时候，就会实行数据淘汰策略。

Redis 提供 6 种数据淘汰策略：

1、volatile-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

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

2、 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

3、 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

4、 allkeys-lru: 从数据集 (server.db[i].dict) 中挑选最近最少使用的数据淘汰

5、 allkeys-random: 从数据集 (server.db[i].dict) 中任意选择数据淘汰

6、 no- eviction (驱逐): 禁止驱逐数据

注意这里的 6 种机制, volatile 和 allkeys 规定了是对已设置过期时间的数据集淘汰数据还是从全部数据集淘汰数据, 后面的 lru、ttl 以及 random 是三种不同的淘汰策略, 再加上一种 no- eviction 永不回收的策略。

使用策略规则:

1、 如果数据呈现寡律分布, 也就是一部分数据访问频率高, 一部分数据访问频率低, 则使用 allkeys-lru

2、 如果数据呈现平等分布, 也就是所有的数据访问频率都相同, 则使用 allkeys-random

三种数据淘汰策略:

ttl 和 random 比较容易理解, 实现也会比较简单。主要是 Lru 最近最少使用淘汰策略, 设计上会对 key 按失效时间排序, 然后取最先失效的 key 进行淘汰

Redis 最适合的场景

关注公众号: 磊哥聊编程, 回复: 面试题, 获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

Redis 最适合所有数据 in-memory 的场景，虽然 Redis 也提供持久化功能，但实际上更多的是一个 disk-backed 的功能，跟传统意义上的持久化有比较大的差别，那么可能大家就会有疑问，似乎 Redis 更像一个加强版的 Memcached，那么何时使用 Memcached，何时使用 Redis 呢？

如果简单地比较 Redis 与 Memcached 的区别，大多数都会得到以下观点：

- 1、 Redis 不仅仅支持简单的 k/v 类型的数据，同时还提供 list, set, zset, hash 等数据结构的存储。
- 2、 Redis 支持数据的备份，即 master-slave 模式的数据备份。
- 3、 Redis 支持数据的持久化，可以将内存中的数据保持在磁盘中，重启的时候可以再次加载进行使用。

1、 会话缓存 (Session Cache)

最常用的一种使用 Redis 的情景是会话缓存 (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的，现在，他们还会这样吗？

幸运的是，随着 Redis 这些年的改进，很容易找到怎么恰当的使用 Redis 来缓存会话的文档。甚至广为人知的商业平台 Magento 也提供 Redis 的插件。

2、 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。回到一致性问题，即使重启了 Redis 实例，因为有磁盘的持久化，用户也不会看到页面加载速度的下降，这是一个极大改进，类似 PHP 本地 FPC。

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

再次以 Magento 为例，Magento 提供一个插件来使用 Redis 作为全页缓存后端。

此外，对 WordPress 的用户来说，Pantheon 有一个非常好的插件 wp-Redis，这个插件能帮助你以最快速度加载你曾浏览过的页面。

3、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（如 Python）对 list 的 push/pop 操作。

如果你快速的在 Google 中搜索“Redis queues”，你马上就能找到大量的开源项目，这些项目的目的就是利用 Redis 创建非常好的后端工具，以满足各种队列需求。例如，Celery 有一个后台就是使用 Redis 作为 broker，你可以从这里去查看。

4、排行榜/计数器

Redis 在内存中对数字进行递增或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的非常简单，Redis 正好提供了这两种数据结构。

所以，我们要从排序集合中获取到排名最靠前的 10 个用户-我们称之为“user_scores”，我们只需要像下面一样执行即可：

当然，这是假定你是根据你用户的分数做递增的排序。如果你想返回用户及用户的分数，你需要这样执行：

```
ZRANGE user_scores 0 10 WITHSCORES
```

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

Agora Games 就是一个很好的例子，用 Ruby 实现的，它的排行榜就是使用 Redis 来存储数据的，你可以在这里看到。

5、/订阅

最后（但肯定不是最不重要的）是 Redis 的 /订阅功能。/订阅的使用场景确实非常多。我已看见人们在社交网络连接中使用，还可作为基于 /订阅的脚本触发器，甚至用 Redis 的 /订阅功能来建立聊天系统！（不，这是真的，你可以去核实）。

Redis 提供的所有特性中，我感觉这个是喜欢的人最少的一个，虽然它为用户提供如果此多功能。

Redis 集群方案什么情况下会导致整个集群不可用？

有 A, B, C 三个节点的集群,在没有复制模型的情况下,如果节点 B 失败了,那么整个集群就会以为缺少 5501-11000 这个范围的槽而不可用。

Redis 支持的 Java 客户端都有哪些？官方推荐用哪个？

Redisson、Jedis、lettuce 等等，官方推荐使用 Redisson。

Redis 和 Redisson 有什么关系？

Redisson 是一个高级的分布式协调 Redis 客户端，能帮助用户在分布式环境中轻松实现一些 Java 的对象 (Bloom filter, BitSet, Set, SetMultimap, ScoredSortedSet, SortedSet, Map, ConcurrentMap, List, ListMultimap, Queue, BlockingQueue, Deque, BlockingDeque, Semaphore, Lock,

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

ReadWriteLock, AtomicLong, CountdownLatch, Publish / Subscribe, HyperLogLog)。

Jedis 与 Redisson 对比有什么优缺点？

Jedis 是 Redis 的 Java 实现的客户端，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。

Redis 如何设置密码及验证密码？

设置密码：`config set requirepass 123456`

授权密码：`auth 123456`

说说 Redis 哈希槽的概念？

Redis 集群没有使用一致性 hash，而是引入了哈希槽的概念，Redis 集群有 16384 个哈希槽，每个 key 通过 CRC16 校验后对 16384 取模来决定放置哪个槽，集群的每个节点负责一部分 hash 槽。

Redis 集群的主从复制模型是怎样的？

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

为了使在部分节点失败或者大部分节点无法通信的情况下集群仍然可用，所以集群使用了主从复制模型，每个节点都会有 N-1 个复制品。

Redis 集群会有写操作丢失吗？为什么？

Redis 并不能保证数据的强一致性，这意味着在实际中集群在特定的条件下可能会丢失写操作。

Redis 集群之间是如何复制的？

异步复制

Redis 集群最大节点个数是多少？

16384 个。

Redis 集群如何选择数据库？

Redis 集群目前无法做数据库选择，默认在 0 数据库。

怎么测试 Redis 的连通性？

ping

Redis 中的管道有什么用？

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

一次请求/响应服务器能实现处理新的请求即使旧的请求还未被响应。这样就可以将多个命令发送到服务器，而不用等待回复，最后在一个步骤中读取该答复。

这就是管道（pipelining），是一种几十年来广泛使用的技术。例如许多 POP3 协议已经实现支持这个功能，大大加快了从服务器下载新邮件的过程。

怎么理解 Redis 事务？

事务是一个单独的隔离操作：事务中的所有命令都会序列化、按顺序地执行。事务在执行的过程中，不会被其他客户端发送来的命令请求所打断。

事务是一个原子操作：事务中的命令要么全部被执行，要么全部都不执行。

Redis 事务相关的命令有哪几个？

MULTI、EXEC、DISCARD、WATCH

Redis key 的过期时间和永久有效分别怎么设置？

EXPIRE 和 PERSIST 命令。

Redis 如何做内存优化？

尽可能使用散列表（hashes），散列表（是说散列表里面存储的数少）使用的内存非常小，所以你应该尽可能的将你的数据模型抽象到一个散列表里面。

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

比如你的 web 系统中有一个用户对象，不要为这个用户的名称，姓氏，邮箱，密码设置单独的 key,而是应该把这个用户的所有信息存储到一张散列表里面。

Redis 回收进程如何工作的？

一个客户端运行了新的命令，添加了新的数据。

Redis 检查内存使用情况，如果大于 maxmemory 的限制，则根据设定好的策略进行回收。

一个新的命令被执行，等等。

所以我们不断地穿越内存限制的边界，通过不断达到边界然后不断地回收回到边界以下。

如果一个命令的结果导致大量内存被使用（例如很大的集合的交集保存到一个新的键），不用多久内存限制就会被这个内存使用量超越。

Redis 集群方案应该怎么做？都有哪些方案？

1、codis。

目前用的最多的集群方案，基本和 twemproxy 一致的效果，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。

2、Redis cluster3.0 自带的集群，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。具体看官方文档介绍。

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题

