# **[Redis的五种数据结构的内部编码](https://www.cnblogs.com/yangmingxianshen/p/8054094.html)**

type命令实际返回的就是当前键的数据结构类型，它们分别是：string（字符串）、hash（哈希）、

list（列表）、set（集合）、zset（有序集合），但这些只是Redis对外的数据结构。



实际上每种数据结构都有自己底层的内部编码实现，而且是多种实现，这样Redis会在合适的场景选择合适的内部编码。

Redis会根据数据量自适应地选择较优化的内部编码格式。

可以看到**每种数据结构都有两种以上的内部编码实现**，例如string数据结构就包含了raw、int和embstr三种内部编码。

同时，**有些内部编码可以作为多种外部数据结构的内部实现**，例如ziplist就是hash、list和zset共有的内部编码。

我们可以通过object encoding命令查询内部编码：

[IMG_256](https://www.cnblogs.com/yangmingxianshen/p/javascript:void(0);)

127.0.0.1:6379> set set:1 hello

OK

127.0.0.1:6379> object encoding set:1

"embstr"

127.0.0.1:6379> hset user:1 name kebi

(integer) 1

127.0.0.1:6379> object encoding user:1

"ziplist"

[IMG_257](https://www.cnblogs.com/yangmingxianshen/p/javascript:void(0);)

可以看到键set:1对应值的内部编码是“embstr”，键user:1对应值的内部编码是“ziplist”。

Redis这样设计有两个好处：

第一，**可以改进内部编码**，而对外的数据结构和命令没有影响，这样一旦开发开发出优秀的内部编码，无需改动外部数据结构和命令。

第二，**多种内部编码实现可以在不同场景下发挥各自的优势**。例如ziplist比较节省内存，但是在列表元素比较多的情况下，性能会有所下降，

　　这时候Redis会根据配置选项将列表类型的内部实现转换为linkedlist。

Redis可以支撑每秒数万的读写操作，但是这指的是Redis服务端的处理

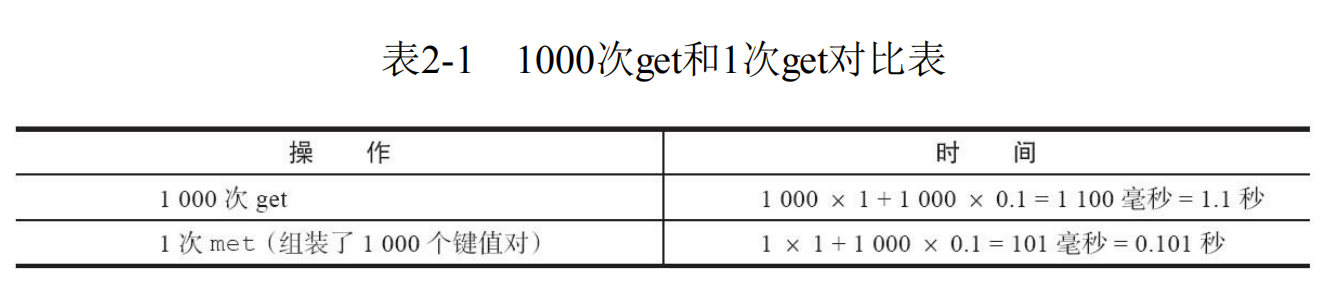
能力，对于客户端来说，一次命令除了命令时间还是有网络时间，假设网络

时间为1毫秒，命令时间为0.1毫秒（按照每秒处理1万条命令算），那么执

行1000次get命令和1次mget命令的区别如表2-1，因为Redis的处理能力已经

足够高，对于开发人员来说，网络可能会成为性能的瓶颈。

表2-1 1000次get和1次get对比表



学会使用批量操作，有助于提高业务处理效率，但是要注意的是每次批

量操作所发送的命令数不是无节制的，如果数量过多可能造成Redis阻塞或

者网络拥塞。