参考：<http://www.redis.cn/commands/scan.html>

命令模式：

**SCAN cursor [MATCH pattern] [COUNT count]**

**起始版本：2.8.0**

**时间复杂度：**O(1) for every call. O(N) for a complete iteration, including enough command calls for the cursor to return back to 0. N is the number of elements inside the collection.

[SCAN](http://www.redis.cn/commands/scan.html) 命令及其相关的 [SSCAN](http://www.redis.cn/commands/sscan.html), [HSCAN](http://www.redis.cn/commands/hscan.html) 和 [ZSCAN](http://www.redis.cn/commands/zscan.html) 命令都用于***增量迭代一个集合元素***。

* [SCAN](http://www.redis.cn/commands/scan.html) 命令用于迭代当前数据库中的key集合。
* [SSCAN](http://www.redis.cn/commands/sscan.html) 命令用于迭代SET集合中的元素。
* [HSCAN](http://www.redis.cn/commands/hscan.html) 命令用于迭代Hash类型中的键值对。
* [ZSCAN](http://www.redis.cn/commands/zscan.html) 命令用于迭代SortSet集合中的元素和元素对应的分值

以上列出的四个命令都支持增量式迭代，它们每次执行都只会返回少量元素，所以这些命令可以用于生产环境，而不会出现像 [KEYS](http://www.redis.cn/commands/keys.html) 或者 [SMEMBERS](http://www.redis.cn/commands/smembers.html)（返回集合set中所有的成员及）命令带来的可能会阻塞服务器的问题。

不过，[SMEMBERS](http://www.redis.cn/commands/smembers.html) 命令可以返回集合（set）键当前包含的所有元素， 但是对于SCAN这类增量式迭代命令来说，有可能在增量迭代过程中，集合元素被修改，对返回值无法提供完全准确的保证。

因为 [SCAN](http://www.redis.cn/commands/scan.html), [SSCAN](http://www.redis.cn/commands/sscan.html), [HSCAN](http://www.redis.cn/commands/hscan.html) 和 [ZSCAN](http://www.redis.cn/commands/zscan.html) 四个命令的工作方式都非常相似， 所以这个文档会一并介绍这四个命令，***需要注意的是***[***SSCAN***](http://www.redis.cn/commands/sscan.html)***,***[***HSCAN***](http://www.redis.cn/commands/hscan.html)***,***[***ZSCAN***](http://www.redis.cn/commands/zscan.html)***命令的第一个参数总是一个key；***[***SCAN***](http://www.redis.cn/commands/scan.html)***命令则不需要在第一个参数提供任何key，因为它迭代的是当前数据库中的所有key***。

SCAN命令的基本用法

[SCAN](http://www.redis.cn/commands/scan.html)命令是一个基于游标的迭代器。这意味着命令每次被调用都需要使用上一次这个调用返回的游标作为该次调用的游标参数，以此来延续之前的迭代过程

当[SCAN](http://www.redis.cn/commands/scan.html)命令的*游标参数被设置为 0 时， 服务器将开始一次新的迭代*， 而当服务器向用户返回值为 0 的游标时， 表示迭代已结束。

以下是一个 SCAN 命令的迭代过程示例 :

localhost:6379> scan 0

1) "1408"

2) 1) "A0TM:1330:QC\_\xe5\xb9\xbb"

2) "A0TM:1330:QC\_\xe7\xae\x80"

3) "A0TM:1330:QC\_\xe9\x86\x92"

4) "A0TM:1330:QC\_\xe9\xbb\x91"

5) "A0TM:1330:QC\_\xe5\x8f\x8c"

6) "A0TM:1330:QC\_\xe8\x96\x9b"

7) "redis0.molitv.cn:6379:CommandCount:1478201017"

8) "A0TM:1330:QC\_\xe5\xbf\x86"

9) "A0TM:1330:QC\_\xe7\xba\xa2"

10) "A0TM:1330:QC\_\xe6\x89\x92"

localhost:6379>

localhost:6379> scan 1408

1) "1472"

2) 1) "A0TM:1330:QC\_\xe5\xab\x81"

2) "A0TM:1330:QC\_\xe8\x87\xa3"

3) "A0TM:1330:QC\_\xe8\xb5\x9b"

4) "A0TM:1330:QC\_\xe0\xb8\xad"

5) "A0TM:1330:QC\_\xe9\x93\xa0"

6) "A0TM:1330:QC\_\xe6\x97\xa9"

7) "A0TM:1330:QC\_\xe5\xb1\x80"

8) "A0TM:1330:QC\_\xe5\xb7\xb4"

9) "A0TM:1330:QC\_\xe6\xb0\xb4"

10) "A0TM:1330:QC\_\xe5\xb8\xa6"

localhost:6379>

在上面这个例子中， 第一次迭代使用 0 作为游标， 表示开始一次新的迭代。第二次迭代使用的是第一次迭代时返回的游标 1408，作为新的迭代参数 。

显而易见，SCAN命令的返回值 是一个包含两个元素的数组， **第一个数组元素是用于进行下一次迭代的新游标， 而第二个数组元素则是一个数组， 这个数组中包含了所有被迭代的元素**。

在第二次调用 SCAN 命令时， 命令返回了游标 0 ， 这表示迭代已经结束， 整个数据集已经被完整遍历过了。

full iteration ：以 0 作为游标开始一次新的迭代， 一直调用 SCAN 命令， 直到命令返回游标 0 ， 我们称这个过程为一次完整遍历。

**Scan命令的保证**

SCAN命令以及其他增量式迭代命令（sscan zscan hscan）， 在进行完整遍历的情况下可以为用户带来以下保证 ：

1、从完整遍历开始直到完整遍历结束期间， 一直存在于数据集内的所有元素都会被完整遍历返回； 这意味着， 如果有一个元素， 它从遍历开始直到遍历结束期间都存在于被遍历的数据集当中， 那么 SCAN 命令总会在某次迭代中将这个元素返回给用户。

2、同样，如果一个元素在开始遍历之前被移出集合，并且在遍历开始直到遍历结束期间都没有再加入，那么在遍历返回的元素集中就不会出现该元素。

然而因为增量式命令仅仅使用游标来记录迭代状态， 所以这些命令**带有以下缺点**：

1、同一个元素可能会被返回多次。 处理重复元素的工作交由应用程序负责， 比如说， 可以考虑将迭代返回的元素仅仅用于可以安全地重复执行多次的操作上。

2、如果一个元素是在迭代过程中被添加到数据集的， 又或者是在迭代过程中从数据集中被删除的， 那么这个元素可能会被返回， 也可能不会。所以在同步的时候要锁定acquirekey

**SCAN命令每次执行返回的元素数量**

SCAN增量式迭代命令并不保证每次执行都返回某个给定数量的元素,甚至可能会返回零个元素， 但只要命令返回的游标不是 0 ， 应用程序就不应该将迭代视作结束。

不过命令返回的元素数量总是符合一定规则的， 对于一个大数据集来说， 增量式迭代命令每次最多可能会返回数十个元素；而对于一个足够小的数据集来说， 如果这个数据集的底层表示为编码数据结构（小的sets, hashes and sorted sets）， 那么增量迭代命令将在一次调用中返回数据集中的所有元素。

如果需要的话，用户可以通过增量式迭代命令提供的***COUNT选项来指定每次迭代返回元素的最大值***（即一次返回多少个元素）。

**COUNT选项**

对于增量式迭代命令不保证每次迭代所返回的元素数量，我们可以使用COUNT选项， 对命令的行为进行一定程度上的调整。COUNT 选项的作用就是让用户告知迭代命令， 在每次迭代中应该从数据集里返回多少元素。使用COUNT 选项对于对增量式迭代命令相当于一种提示， 大多数情况下这种提示都比较有效的控制了返回值的数量。

1、COUNT 参数的默认值为 10 。

2、数据集比较大时，如果没有使用MATCH 选项, 那么命令返回的元素数量通常和 COUNT 选项指定的一样， 或者比 COUNT 选项指定的数量稍多一些。

3、在迭代一个编码为整数集合（intset，一个只由整数值构成的小集合）、 或者编码为压缩列表（ziplist，由不同值构成的一个小哈希或者一个小有序集合）时， 增量式迭代命令通常会无视 COUNT 选项指定的值， 在第一次迭代就将数据集包含的所有元素都返回给用户。

注意: \*\*并非每次迭代都要使用相同的 COUNT 值 \*\*，用户可以在每次迭代中按自己的需要随意改变 COUNT 值， 只要记得将上次迭代返回的游标用到下次迭代里面就可以了。

localhost:6379> SCAN 0 COUNT 20 ##自定一次返回多少个数据，只对当前命令有效

1) "1472"

2) 1) "A0TM:1330:QC\_\xe5\xb9\xbb"

2) "A0TM:1330:QC\_\xe7\xae\x80"

3) "A0TM:1330:QC\_\xe9\x86\x92"

…

**MATCH 选项**

类似于KEYS 命令，增量式迭代命令通过给定 MATCH 参数的方式实现了通过提供一个 glob 风格的模式参数， 让命令只返回和给定模式相匹配的元素。

以下是一个使用 MATCH 选项进行迭代的示例:

localhost:6379> scan 0 match \*33\* count 2 ##指定匹配key的模式并返回两个匹配项

1) "256"

2) 1) "A0TM:1330:QC\_\xe5\xb9\xbb"

2) "A0TM:1330:QC\_\xe7\xae\x80"

localhost:6379>

**MATCH功能对元素的模式匹配工作是在命令从数据集中取出元素后和向客户端返回元素前的这段时间内进行的（类似于管道操作符）**， 所以如果被迭代的数据集中只有少量元素和模式相匹配， 那么迭代命令或许会在多次执行中都不返回任何元素。

以下是这种情况的一个例子:

localhost:6379> scan 0 match \*33\* count 2 ##开始一次迭代

1) "256"

2) 1) "A0TM:1330:QC\_\xe5\xb9\xbb"

2) "A0TM:1330:QC\_\xe7\xae\x80"

localhost:6379> scan 0 match \*33\* count 5

1) "128"

2) 1) "A0TM:1330:QC\_\xe5\xb9\xbb"

2) "A0TM:1330:QC\_\xe7\xae\x80"

3) "A0TM:1330:QC\_\xe9\x86\x92"

4) "A0TM:1330:QC\_\xe9\xbb\x91"

5) "A0TM:1330:QC\_\xe5\x8f\x8c"

localhost:6379> scan 128 match \*33\* count 10

1) "704"

2) 1) "A0TM:1330:QC\_\xe8\x96\x9b"

2) "A0TM:1330:QC\_\xe5\xbf\x86"

3) "A0TM:1330:QC\_\xe7\xba\xa2"

4) "A0TM:1330:QC\_\xe6\x89\x92"

5) "A0TM:1330:QC\_\xe5\xab\x81"

6) "A0TM:1330:QC\_\xe8\x87\xa3"

7) "A0TM:1330:QC\_\xe8\xb5\x9b"

8) "A0TM:1330:QC\_\xe0\xb8\xad"

9) "A0TM:1330:QC\_\xe9\x93\xa0"

10) "A0TM:1330:QC\_\xe6\x97\xa9"

localhost:6379> scan 704 match \*33\* count 3

1) "448"

2) 1) "A0TM:1330:QC\_\xe5\xb1\x80"

2) "A0TM:1330:QC\_\xe5\xb7\xb4"

3) "A0TM:1330:QC\_\xe6\xb0\xb4"

1) "1520"

2) (empty list or set)

localhost:6379> scan 1520 match \*3ccc\* count 100

1) "1572"

2) (empty list or set)

localhost:6379> scan 1572 match \*3ccc\* count 100

1) "28"

2) (empty list or set)

localhost:6379> scan 28 match \*3ccc\* count 1000

1) "1903"

2) (empty list or set)

localhost:6379> scan 1903 match \*3ccc\* count 1000

1) "0" ##结束一次迭代

2) (empty list or set)

localhost:6379>

可以看出，以上的大部分迭代都不返回任何元素（因为没有匹配元素）。在最后一次迭代， 我们通过将 COUNT 选项的参数设置为 1000 ， 强制命令为本次迭代扫描更多元素， 从而使得命令返回的元素也变多了。

**并发执行多个迭代**

在同一时间， 可以有任意多个客户端对同一数据集进行迭代， 客户端每次执行迭代都需要传入一个游标， 并在迭代执行之后获得一个新的游标， 而这个游标就包含了迭代的所有状态， 因此， 服务器无须为迭代记录任何状态。

**中止迭代**

因为迭代的所有状态都保存在游标里面， 而服务器无须为迭代保存任何状态， 所以客户端可以在中途停止一个迭代， 而无须对服务器进行任何通知。即使有任意数量的迭代在中途停止， 也不会产生任何问题。

**使用错误的游标**

使用SCAN 命令传入间断的（broken）、负数、超出范围或者其他非正常的游标来执行增量式迭代并不会造成服务器崩溃， 但可能会让命令产生未定义的行为。未定义行为指的是， 增量式命令对返回值所做的保证可能会不再为真。 只有两种游标是合法的:

1、在开始一个新的迭代时， 游标必须为 0 。

2、增量式迭代命令在执行之后返回的， 用于延续迭代过程的游标。

**返回值**

[SCAN](http://www.redis.cn/commands/scan.html), [SSCAN](http://www.redis.cn/commands/sscan.html), [HSCAN](http://www.redis.cn/commands/hscan.html) 和 [ZSCAN](http://www.redis.cn/commands/zscan.html) 命令都返回一个包含两个元素的 multi-bulk （多块）回复： 回复的第一个元素是字符串表示的无符号 64 位整数（游标）， 回复的第二个元素是另一个 multi-bulk 回复， 包含了本次被迭代的元素。

1、[SCAN](http://www.redis.cn/commands/scan.html) 命令返回的每个元素都是一个key。

2、[SSCAN](http://www.redis.cn/commands/sscan.html) 命令返回的每个元素都是一个集合成员。

3、[HSCAN](http://www.redis.cn/commands/hscan.html) 命令返回的每个元素都是一个键值对，一个键值对由一个键和一个值组成。

4、[ZSCAN](http://www.redis.cn/commands/zscan.html)命令返回的每个元素都是一个有序集合元素，一个有序集合元素由一个成员（member）和一个分值（score）组成。

另外的例子

迭代hash中的键值对：

192.168.100.249:6379> hscan A1TM:6861:VideoIntro 0 COUNT 1 ##0表示开始一次迭代，1表示返回1个元素

1) "32768"

2) 1) "25495"

2) "{\"guid\":\"467f8199-8e77-4542-8114-d3f0b5c8a501\",