

Redis中偶数版本号是稳定的版本，奇数是不稳定的。

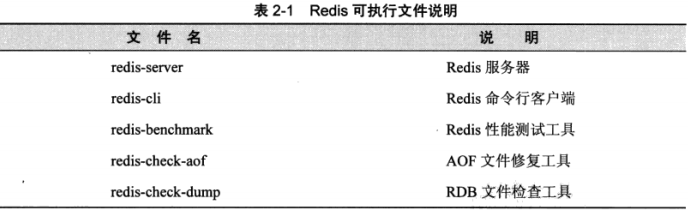
Linux下安装：

wget <http://download.redis.io/redis-stable.tar.gz>

tar xzf redis-stable.tar.gz

cd redis-stable

make

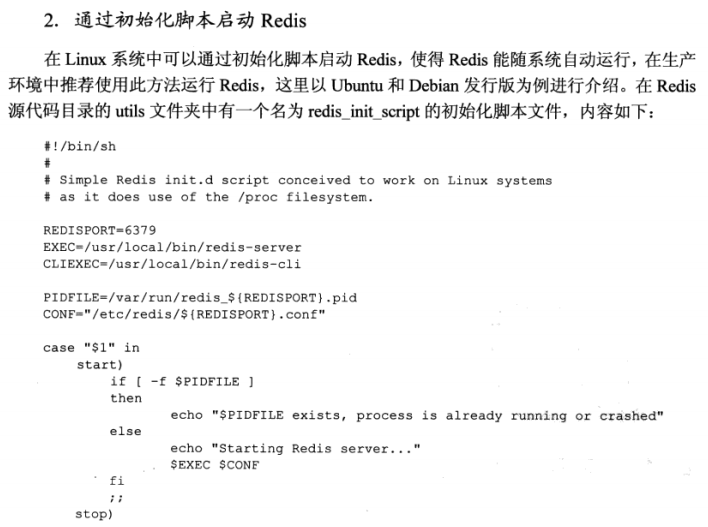


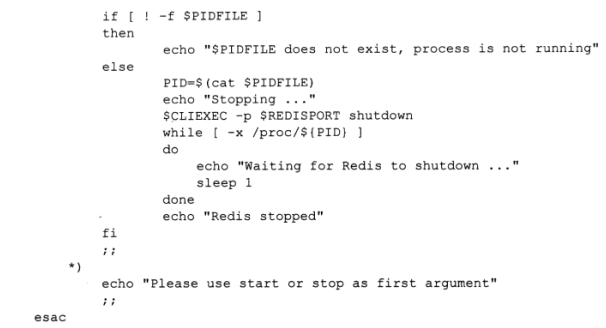
我们最常用的2个程序是redis-server和redis-cli,其中redis-server是redis的服务器，启动redis即运行redis-server;而redis-cli是Redis自带的Redis命令客户端。

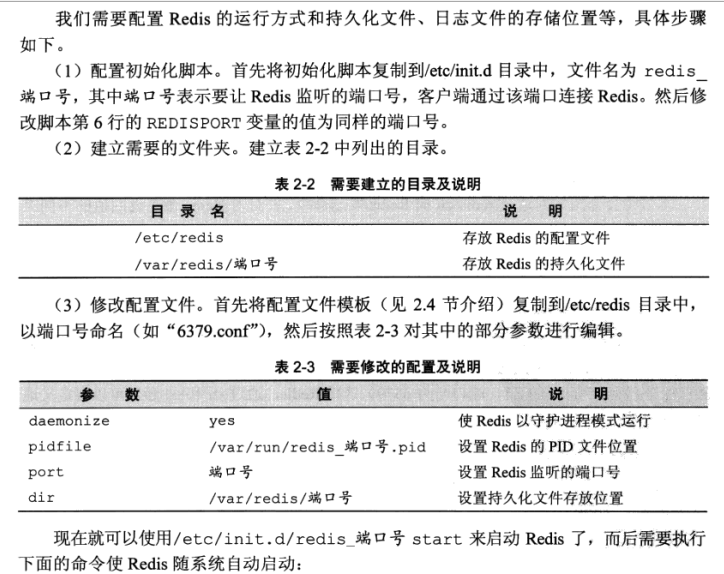
直接启动，直接运行redis-server即可:$ redis-server

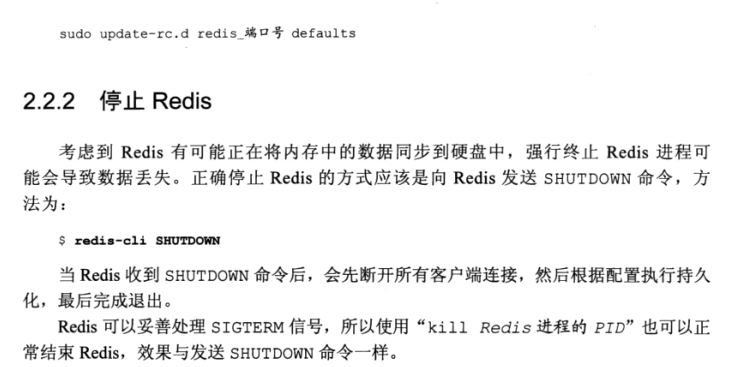
启动默认的端口是6379,可以通过--port参数自定义端口

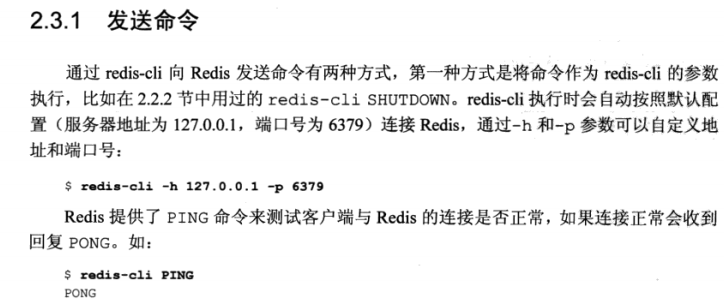
$ redis-server --port 6380

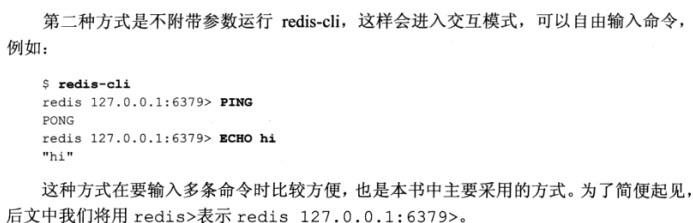












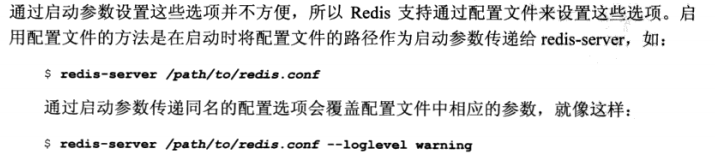
发送相关的命令后会有相应的几种返回值：

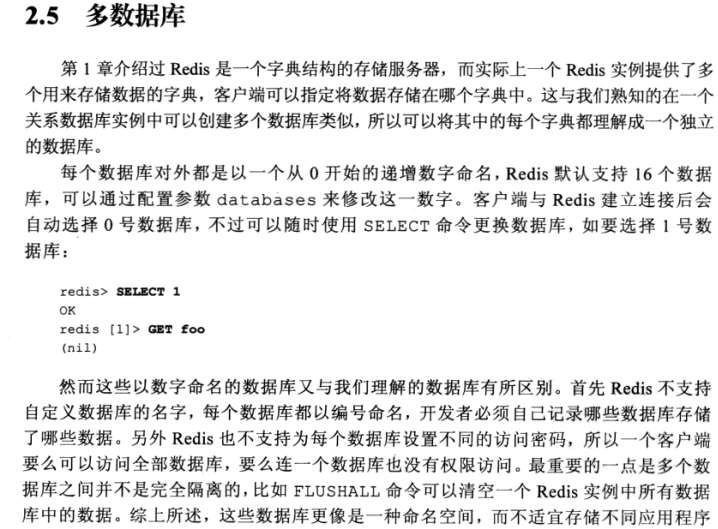
1.状态回复

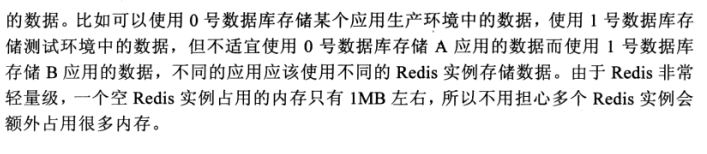
2.错误回复

3.整数回复(执行的结果数，如hget返回1，说明返回一条)

4.字符串回复   5.多行字符串回复







命令：

赋值与取值

SET key value

GET key

redis> SET key hello

redis> GET key

redis> KEYS \* //keys命令需要遍历Redis中所有键，当键的数量较多的时候会影响性能，不建议在生产环境中使用。

redis命令不区分大小写。

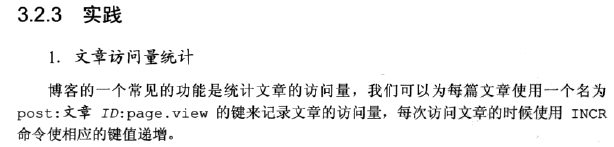
redis> EXISTS key //判断一个键是否存在

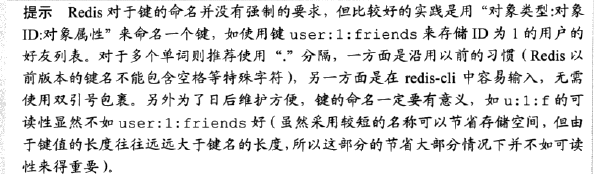
redis> DEL key [key ...] //可以删除一个或多个键

redis> TYPE key 获得键值的数据类型

redis> LPUSH bar 1 //LPUSH命令的作用是向指定的列表类型键中增加一个元素，如果不存在则创建它。

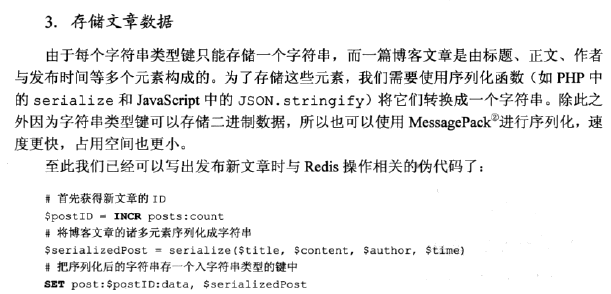
redis> INCR key //如果key对应的值是数字类似则它自增

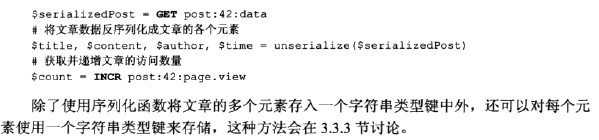




自增ID：

在关系数据库中我们通过设置字段属性为auto\_increment来实现.Redis中可以通过另一种模式来实现 ：对于每一类对象使用名为对象类型(复数形式):count的键(如users:count)来存储当前类型对象的数量，每增加一个对象时都 使用INCR命令递增该键的值。由于使用incr命令建立的键的初始键值是1，所以可以很容易得知,INCR命令的返回值既是加入该对象后的当前类型的对象总数，又是该新增对象的ID。





INCRBY key increment  增加指定的整数，incrby命令与incr命令基本一样，只不过前者可以通过increment参数指定一次增加的数值。 redis > incrby bar 2 (在bar上增加2)

DECR key decrement 减少指定的整数 ,DECR命令与INCR命令用法相同，只不过是让键值递减。 redis>DECR bar , redis > decrby key 5 就是把key的值减去5

增加浮点数:INCRBYFLOAT key increment

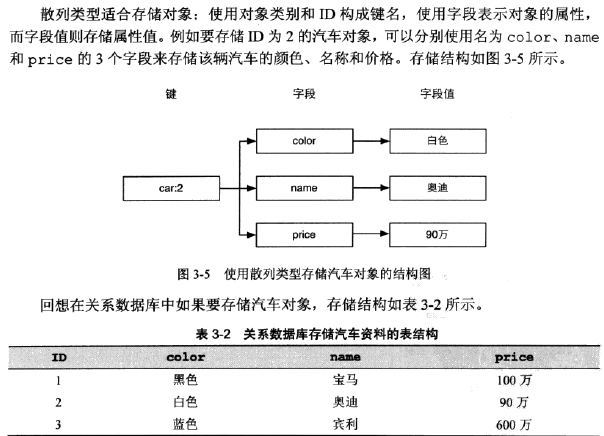
向尾部追加值: APPEND key valule

获取字符串长度: STRLEN key

同时获得/设置多个键值: MGET key [key ...]  ; redis > MGET key1 key3

位操作:GETBIT key offset ;

散列类型的键值也是一种字典结构，其存储了字段(field)和字段值的映射，伸出字段值只能是字符串，不支持其他数据类型。散列类型不能嵌套数据类型。



Reids完全可以自由的为任何键增减字段而不影响其他键.

散列的赋值与取值:

HSET key field value

HGET key field

HMSET key field value [field value ...]

HMGET key field [field ...]

HGETALL key

HSET命令用来给字段赋值，而HGET命令用来获得字段的值.

HSET操作不区分插入和更新操作,这意味着修改数据时不用事先判断字段是否存在来决定要执行的是插入还是更新操作。当执行的是插入操作时(即之前字段不存在)HSET命令会返回1,当执行的是更新操作时(即之前字段已经存在)HSET命令返回0.更进一步，当键本身不存在时，HSET命令还自动建立它。

当需要同时设置多个字段的值时，可以使用HMSET命令:

HSET key field1 value1

HSET key field2 value2

可以用HMSET命令改成: HMSET key field1 value1 field2 value2

相应的HMGET命令可以同时获得多个字段的值:  redis > HMGET car price name

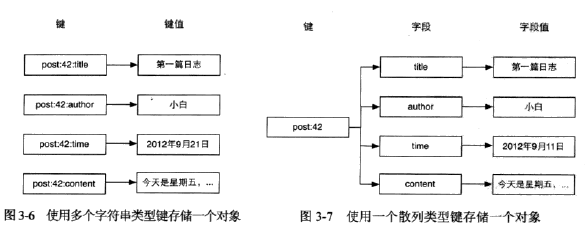
2.判断字段是否存在: HEXISTS key field, HEXISTS命令用来判断一个字段是否存在，存在返回1，否则0.

HSETNX key field value ;当字段不存在时赋值;

3.增加数字: HINCRBY key field increment; 如果不存在，则建立默认值为0.

4.删除字段  HDEL key field [field ...]; HDEL命令可以删除一个或多个字段，返回值是被删除的字段个数。

redis> HDEL car price;



只获取字段名或字段值： HKEYS key , HVALS key

获取字段数量：HLEN key

列表类型：

列表类型可以存储一个有序的字符串列表，常用的操作是向列表两端添加元素，或者获得列表的某一个片段。列表类型内部是使用双向链表实现的，所以向列表两端添加元素的时间复杂度为O(1),获取越接近两端的元素速度就越快。不过使用链表的代价是通过索引访问元素比较慢。

1.向列表两端增加元素：

LPUSH key value [value ...]

RPUSH key value [value ...]

2.从列表两端弹出元素

LPOP key ;  RPOP key;

3.获取列表中元素的个数: LLEN key; 当键不存在时就会返回0.

4.获取列表片段: LRANGE key start stop ; LRANGE number 0 2;获取key为number的索引从0到2.

LRANGE也支持负索引，表示从右边开始计算序数，如"-1"表示最右边第一个元素，“-2”表示最右边第二个元素，依次类推。

5.删除列表中指定的值: LREM key count value ;LREM命令会删除列表中前count个值为value的元素,返回值是实际删除的元素个数。count>0时LREM命令会从列表左边开始删除前count个值为value的元素. count<0时LREM会从列表右边开始删除前count个值为value的元素。

如果要将列表类型当作数组来用，LINDEX命令是必不可少的。LINDEX命令用来返回指定索引的元素，索引从0开始。

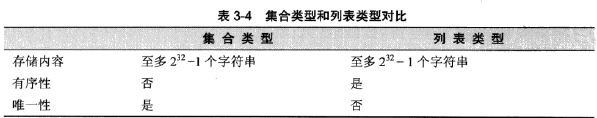
1.获得/设置指定索引的元素值:  LINDEX key index; LSET key index value;

2.只保留列表指定片段: LTRIM key start end; LTRIM命令可以删除指定索引范围之外的所有元素，其指定列表范围的方法和LRANGE命令相同。

3.向列表中插入元素: LINSERT key BEFORE|AFTER pivot value ; LINSERT命令首先会在列表 中从左到右查找值为pivot的元素，然后根据第二个参数是BEFORE还是AFTER来决定将value插入到该元素的前面还是后面。

4.将元素从一个列表转到另一个列表: PROPLPUSH source destination

集合类型：



集合类型的常用操作是向集合中加入或删除元素、判断某个元素是否存在等，由于集合类型在Redis内部是使用值为空的散列表(hash table)实现的，所以这些操作的时间复杂度是O(1).最方便的是多个集合类型键之间还可以进行并集，交集和差集运算。

命令：

1. 增加/删除元素; SADD key member [member ...]; SREM key member [member ...];

2. 获得集合中的所有元素;SMEMBERS key;

3. 判断元素是否在集合中; SISMEMBER key member;

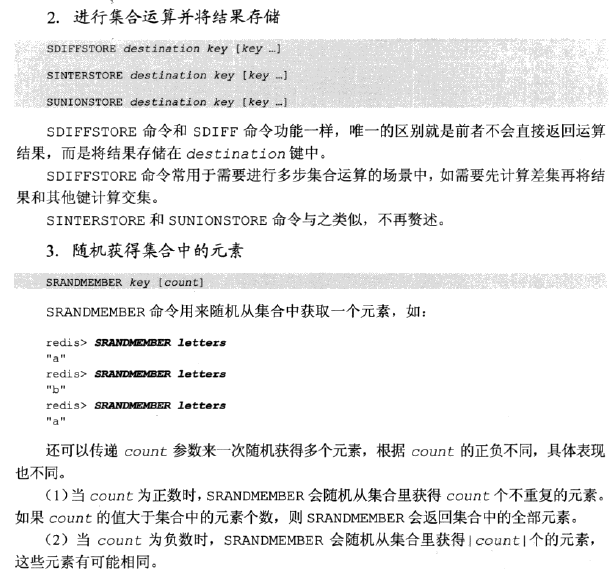
4.集合间运算

SDIFF key [key ...]; 差集

SINTER key [key ...]; 交集

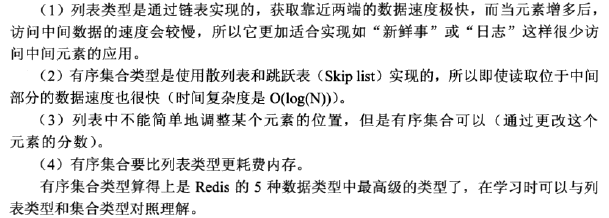
SUNION key [key ...]; 并集

5.获得集合中元素个数: SCARD key;



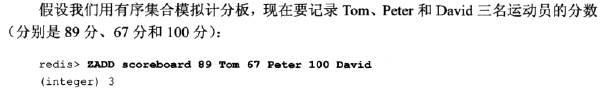
从集合中弹出一个元素：SPOP key;

**有序集合类型：**



   命令：

1.增加元素： ZADD key score member [score member ...]



2.获得元素的分数：ZSCORE key member;

3.获得排名在某个范围的元素列表:

ZRANGE key start stop [WITHSCORES]

ZREVRANGE key start stop [WITHSCORES]

ZRANGE命令会按照元素分数从小到大的顺序返回索引从start 到 stop之间的所有元素，包含两端的元素。ZREVRANGE命令是按照元素分数从大到小的顺序给出结果。

4.获得指定分数范围的元素： ZRANGEBYSCORE key min max [WITHSCORES] [LIMIT offset count]

ZRANGEBYSCORE 命令参数虽然多，但是都很好理解。该命令按照元素分数从小到大的顺序返回分数在min和max之间（包括min和max).

5.增加某个元素的分数: ZINCRBY key increment member; ZINCRBY命令可以增加一个元素的分数，返回值是更改后的分数。例如，想给Jerry加4分:redis> ZINCRBY scoreboard 4 Jerry

