INFO

INFO [section]

以一种易于解析（parse）且易于阅读的格式，返回关于 Redis 服务器的各种信息和统计数值。

通过给定可选的参数 section ，可以让命令只返回某一部分的信息：

- server 部分记录了 Redis 服务器的信息，它包含以下域：

  使用 单元测试 来保证所有服务器的配置是一样的

  - redis_version ：Redis 服务器版本
  - redis_git_sha1 ：Git SHA1
  - redis_git_dirty ：Git dirty flag
  - os ：Redis 服务器的宿主操作系统
  - arch_bits ：架构（32 或 64 位）
  - multiplexing_api ：Redis 所使用的事件处理机制
  - gcc_version ：编译 Redis 时所使用的 GCC 版本
  - process_id ：服务器进程的 PID　　验证Redis进程是否还活着
  - run_id ：Redis 服务器的随机标识符（用于 Sentinel 和集群）
  - tcp_port ：TCP/IP 监听端口
  - uptime_in_seconds ：自 Redis 服务器启动以来，经过的秒数
  - uptime_in_days ：自 Redis 服务器启动以来，经过的天数　　验证Redis服务器中途是否重启过
  - lru_clock ：以分钟为单位进行自增的时钟，用于 LRU 管理

- clients 部分记录了已连接客户端的信息，它包含以下域：

  查看当前服务器的压力

  - connected_clients ：已连接客户端的数量（不包括通过从属服务器连接的客户端）
  - client_longest_output_list ：当前连接的客户端当中，最长的输出列表
  - client_longest_input_buf ：当前连接的客户端当中，最大输入缓存
  - blocked_clients ：正在等待阻塞命令（BLPOP、BRPOP、BRPOPLPUSH）的客户端的数量

- memory 部分记录了服务器的内存信息，它包含以下域：

  - used_memory ：由 Redis 分配器分配的内存总量，以字节（byte）为单位
  - used_memory_human ：以人类可读的格式返回 Redis 分配的内存总量　　这个值和 top 、 ps 等命令的输出一致。
  - used_memory_rss ：从操作系统的角度，返回 Redis 已分配的内存总量（俗称常驻集大小）。这个值和 top 、
  - used_memory_peak ：Redis 的内存消耗峰值（以字节为单位）
  - used_memory_peak_human ：以人类可读的格式返回 Redis 的内存消耗峰值
  - used_memory_lua ：Lua 引擎所使用的内存大小（以字节为单位）
  - mem_fragmentation_ratio ：used_memory_rss 和 used_memory 之间的比率
  - mem_allocator ：在编译时指定的， Redis 所使用的内存分配器。可以是 libc 、 jemalloc 或者 tcmalloc 。

  在理想情况下， used_memory_rss 的值应该只比 used_memory 稍微高一点儿。
  当 rss > used ，且两者的值相差较大时，表示存在（内部或外部的）内存碎片。
  内存碎片的比率可以通过 mem_fragmentation_ratio 的值看出。
  当 used > rss 时，表示 Redis 的部分内存被操作系统换出到交换空间了，在这种情况下，操作可能会产生明显的延迟。

  Because Redis does not have control over how its allocations are mapped to memory pages, high used_memor
  in memory usage.

  当 Redis 释放内存时，分配器可能会，也可能不会，将内存返还给操作系统。
  如果 Redis 释放了内存，却没有将内存返还给操作系统，那么 used_memory 的值可能和操作系统显示的 Redis 内存占
  查看 used_memory_peak 的值可以验证这种情况是否发生。

- persistence 部分记录了跟 RDB 持久化和 AOF 持久化有关的信息，它包含以下域：

  - loading ：一个标志值，记录了服务器是否正在载入持久化文件。

- rdb_changes_since_last_save ： 距离最近一次成功创建持久化文件之后，经过了多少秒。
- rdb_bgsave_in_progress ： 一个标志值，记录了服务器是否正在创建 RDB 文件。
- rdb_last_save_time ： 最近一次成功创建 RDB 文件的 UNIX 时间戳。
- rdb_last_bgsave_status ： 一个标志值，记录了最近一次创建 RDB 文件的结果是成功还是失败。
- rdb_last_bgsave_time_sec ： 记录了最近一次创建 RDB 文件耗费的秒数。
- rdb_current_bgsave_time_sec ： 如果服务器正在创建 RDB 文件，那么这个域记录的就是当前的创建操作已经
- aof_enabled ： 一个标志值，记录了 AOF 是否处于打开状态。
- aof_rewrite_in_progress ： 一个标志值，记录了服务器是否正在创建 AOF 文件。
- aof_rewrite_scheduled ： 一个标志值，记录了在 RDB 文件创建完毕之后，是否需要执行预约的 AOF 重写操作。
- aof_last_rewrite_time_sec ： 最近一次创建 AOF 文件耗费的时长。
- aof_current_rewrite_time_sec ： 如果服务器正在创建 AOF 文件，那么这个域记录的就是当前的创建操作已经
- aof_last_bgrewrite_status ： 一个标志值，记录了最近一次创建 AOF 文件的结果是成功还是失败。

如果 AOF 持久化功能处于开启状态，那么这个部分还会加上以下域：

- aof_current_size ： AOF 文件目前的大小。
- aof_base_size ： 服务器启动时或者 AOF 重写最近一次执行之后，AOF 文件的大小。
- aof_pending_rewrite ： 一个标志值，记录了是否有 AOF 重写操作在等待 RDB 文件创建完毕之后执行。
- aof_buffer_length ： AOF 缓冲区的大小。
- aof_rewrite_buffer_length ： AOF 重写缓冲区的大小。
- aof_pending_bio_fsync ： 后台 I/O 队列里面，等待执行的 fsync 调用数量。
- aof_delayed_fsync ： 被延迟的 fsync 调用数量。

- stats 部分记录了一般统计信息，它包含以下域：
  - total_connections_received ： 服务器已接受的连接请求数量。 可判断出Client是否使用"长链接"技术
  - total_commands_processed ： 服务器已执行的命令数量。
  - instantaneous_ops_per_sec ： 服务器每秒钟执行的命令数量。 查看服务器的压力
  - rejected_connections ： 因为最大客户端数量限制而被拒绝的连接请求数量。 是否需要调大'ulimit -n'值
  - expired_keys ： 因为过期而被自动删除的数据库键数量。
  - evicted_keys ： 因为最大内存容量限制而被驱逐（evict）的键数量。
  - keyspace_hits ： 查找数据库键成功的次数。
  - keyspace_misses ： 查找数据库键失败的次数。
  - pubsub_channels ： 目前被订阅的频道数量。
  - pubsub_patterns ： 目前被订阅的模式数量。
  - latest_fork_usec ： 最近一次 fork() 操作耗费的毫秒数。

- replication ： 主/从复制信息
  - role ： 如果当前服务器没有在复制任何其他服务器，那么这个域的值就是 master ； 否则的话，这个域的值就
    候，一个从服务器也可能是另一个服务器的主服务器。

如果当前服务器是一个从服务器的话，那么这个部分还会加上以下域：

- master_host ： 主服务器的 IP 地址。
- master_port ： 主服务器的 TCP 监听端口号。
- master_link_status ： 复制连接当前的状态， up 表示连接正常， down 表示连接断开。
- master_last_io_seconds_ago ： 距离最近一次与主服务器进行通信已经过去了多少秒钟。
- master_sync_in_progress ： 一个标志值，记录了主服务器是否正在与这个从服务器进行同步。

如果同步操作正在进行，那么这个部分还会加上以下域：

- master_sync_left_bytes ： 距离同步完成还缺少多少字节数据。
- master_sync_last_io_seconds_ago ： 距离最近一次因为 SYNC 操作而进行 I/O 已经过去了多少秒。

如果主从服务器之间的连接处于断线状态，那么这个部分还会加上以下域：

- master_link_down_since_seconds ： 主从服务器连接断开了多少秒。

以下是一些总会出现的域：

- connected_slaves ： 已连接的从服务器数量。

对于每个从服务器，都会添加以下一行信息：

- slaveXXX ： ID、IP 地址、端口号、连接状态

- cpu 部分记录了 CPU 的计算量统计信息，它包含以下域：

  - used_cpu_sys ： Redis 服务器耗费的系统 CPU 。
  - used_cpu_user ： Redis 服务器耗费的用户 CPU 。
  - used_cpu_sys_children ： 后台进程耗费的系统 CPU 。
  - used_cpu_user_children ： 后台进程耗费的用户 CPU 。

累加值，无多少参考价值

- commandstats 部分记录了各种不同类型的命令的执行统计信息，比如命令执行的次数、命令耗费的 CPU 时间、执行每个命令耗
  令，这个部分都会添加一行以下格式的信息：

  - cmdstat_XXX:calls=XXX,usec=XXX,usecpercall=XXX

执行每条命令耗费的平均 CPU 时间

- cluster 部分记录了和集群有关的信息，它包含以下域：

  - cluster_enabled ： 一个标志值，记录集群功能是否已经开启。

- keyspace 部分记录了数据库相关的统计信息，比如数据库的键数量、数据库已经被删除的过期键数量等。对于每个数据库，这

  - dbXXX:keys=XXX,expires=XXX

除上面给出的这些值以外， section 参数的值还可以是下面这两个：

- all ： 返回所有信息
- default ： 返回默认选择的信息

当不带参数直接调用 INFO 命令时，使用 default 作为默认参数。

> 不同版本的 Redis 可能对返回的一些域进行了增加或删减。
>
> 因此，一个健壮的客户端程序在对 INFO 命令的输出进行分析时，应该能够跳过不认识的域，并且妥善地处理丢失不见的域。

可用版本：

>= 1.0.0

时间复杂度：

O(1)

返回值：

具体请参见下面的测试代码。

```
redis> INFO
# Server
redis_version:2.9.11
redis_git_sha1:937384d0
redis_git_dirty:0
redis_build_id:8e9509442863f22
redis_mode:standalone
os:Linux 3.13.0-35-generic x86_64
arch_bits:64
multiplexing_api:epoll
gcc_version:4.8.2
process_id:4716
run_id:26186aac3f2380aaee9eef21cc50aecd542d97dc
tcp_port:6379
uptime_in_seconds:362
uptime_in_days:0
hz:10
lru_clock:1725349
```

```
config_file:

# Clients
connected_clients:1
client_longest_output_list:0
client_biggest_input_buf:0
blocked_clients:0

# Memory
used_memory:508536
used_memory_human:496.62K
used_memory_rss:7974912
used_memory_peak:508536
used_memory_peak_human:496.62K
used_memory_lua:33792
mem_fragmentation_ratio:15.68
mem_allocator:jemalloc-3.2.0

# Persistence
loading:0
rdb_changes_since_last_save:6
rdb_bgsave_in_progress:0
rdb_last_save_time:1411011131
rdb_last_bgsave_status:ok
rdb_last_bgsave_time_sec:-1
rdb_current_bgsave_time_sec:-1
aof_enabled:0
aof_rewrite_in_progress:0
aof_rewrite_scheduled:0
aof_last_rewrite_time_sec:-1
aof_current_rewrite_time_sec:-1
aof_last_bgrewrite_status:ok
aof_last_write_status:ok

# Stats
total_connections_received:2
total_commands_processed:4
instantaneous_ops_per_sec:0
rejected_connections:0
sync_full:0
sync_partial_ok:0
sync_partial_err:0
expired_keys:0
evicted_keys:0
keyspace_hits:0
keyspace_misses:0
pubsub_channels:0
pubsub_patterns:0
latest_fork_usec:0
migrate_cached_sockets:0

# Replication
role:master
connected_slaves:0
master_repl_offset:0
repl_backlog_active:0
repl_backlog_size:1048576
repl_backlog_first_byte_offset:0
repl_backlog_histlen:0

# CPU
used_cpu_sys:0.21
used_cpu_user:0.17
used_cpu_sys_children:0.00
```

```
used_cpu_user_children:0.00

# Cluster
cluster_enabled:0

# Keyspace
db0:keys=2,expires=0,avg_ttl=0
```