Redis的配置文件详解及info XXX命令

# Redis的运行模式分类

Redis的运行模式分成3类：

### **standalone单机模式**

### **sentinel哨兵模式**

### **cluster集群模式**

每一个Redis服务器都看作是一个Redis节点，sentinel节点、cluster节点都是特殊的Redis节点。通过info server的redis\_mode获取当前Redis的运行模式。Redis的配置文件根据这三种运行模式分成三大部分。

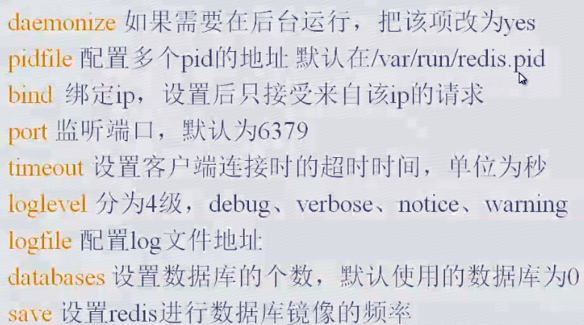
# Redis的配置文件详解

### standalone配置说明与分析:单机模式

### sentinel配置说明与分析

### cluster 配置说明与分析

* 配置文件中大部分参数都是可以通过**config set 配置热生效**的，但是只有小部分是不支持的。如：port、daemonize、bind等不支持。
* 配置文件中的形式都是 ： **param value** #中间不要添加=



## standalone配置说明与分析单机模式(P436)

详细的请参看书本：P436。这里主要是把重要的介绍一下。

### 总体配置：如端口、日志、数据库等

**daemonize**：是否是守护进程，默认no，可选值yes|no；

**port**：端口号，默认为6379；

**databases**：可用的数据库数，默认16个；

tcp-backlog：默认511,；

**dir：工作目录，AOF、RDB、log文件都存放在此目录下，默认./**

**日志**：

**loglevel：日志级别**，默认值为notice，可选值debug|verbose|notice|warning；

**logfile：日志文件名**，默认值空，自定义，建议以端口号为名；

日志的说明：

# Specify the server verbosity level.

# This can be one of:

# debug (a lot of information, useful for development/testing)

# verbose (many rarely useful info, but not a mess like the debug level)

# notice (moderately verbose, **what you want in production probably**)

# warning (only very **important / critical** messages are logged)

**loglevel notice**

# Specify the log file name. Also the empty string can be used to force

# Redis to log on **the standard output**. Note that if you use standard

# output for logging but daemonize, logs will be sent to /dev/null

默认logfile "" 日志打印到标准输出，就是显示屏上，一般建议利用端口号命名如

**logfile "6380.log"**

**log文件存放在dir指定的目录下，默认为./。**

### 最大内存及策略

maxmemory：最大可用内存，单位字节，默认值0，表示没有限制；支持热配置；

maxmemory-policy：内存不够时，淘汰策略，默认值noeviction(不删除键)，详见P437；

maxmemory：检测LRU采样数，默认值5。

### RDB相关配置

**save：RDB保存文件，自动触发bgsave，默认值见下面，如果没有该配置，代表不使用自动RDB策略**；

**save 900 1**

**save 300 10**

**save 60 10000**

# Save the DB on disk:

# **save <seconds> <changes>**

# Will save the DB if both the given number of seconds and the given

# number of **write operations** against the DB occurred.

#

# In the example below the behaviour will be to save:

# after 900 sec (15 min) if at least 1 key changed

# after 300 sec (5 min) if at least 10 keys changed

# after 60 sec if at least 10000 keys changed

#15分钟内至少1次写操作；5分钟内至少10次写操作；1分钟内至少10000次操作。

# Note: you can disable saving completely by commenting out all "save" lines.

#

# It is also possible to remove all the previously configured save

# points by adding a save directive with a single empty string argument

# like in the following example:

#

# save ""

save 900 1

save 300 10

save 60 10000

**dbfilename**：RDB文件名，默认dump.rdb,建议dump-{port}.rdb;

**rebcompression**：RDB文件是否压缩，默认值yes，可选值yes|no；

**rdbchecksum**：RDB文件是否使用校验和，默认值yes，可选值yes|no；

**stop-writes-on-bgsave-error**：bgsave执行错误，是否停止Redis接收写请求，默认值yes，可选值yes|no；

### AOF相关配置

**appendonly**：是否开启AOF持久化模式，默认值no，可选值yes|no;

**appendfsync**：AOF同步磁盘频率，默认值everysec，可选值always|everysec|no;

**appendfilename**：AOF文件名，默认值appendonly.aof，可选值appendonly-{port}.aof;

**aof-load-truncated**：加载AOF文件时，是否忽略AOF文件不完整的情况，让Redis正常启动，默认值yes，可选值yes|no；

**no-appendfsync-on-rewrite**：设置为yes表示**rewrite期间对新写操作不fsync，暂时缓存在缓冲区中，等rewrite完成后再写入**，默认值no，可选值no|yes；

**auto-aof-rewrite-min-size**：触发rewrite的AOF文件最小阀值(单位M)，默认值100,；

**aof-rewrite-incremental-fsync**：AOF重写过程中，是否采取增量文件同步策略，默认为yes，可选值yes|no；

### 慢查询配置

见P438；

### 数据结构优化配置

见P439；

### 复制相关配置replication

**salveof ：配置主节点；**

**slave-read-only**：从节点只读模式；集群架构下从节点默认读写都不可用，需要调用readonly命令开启只读模式；默认值yes，可选值yes|no；

其他见P439；

### 客户端相关配置

**maxclients：最大客户端连接数，默认值10000；**

**client-output-buffer-limit：客户端输出缓冲区限制；**

**timeout：客户端闲置多少秒后关闭连接，默认为0，表示永不关闭；**

**tcp-keepalive：检测TCP连接活性的周期(秒)，默认为0不检测；**

### 安全相关配置

**requirepass**：密码，默认为空；

**bind**：绑定IP，默认为空，绑定IP地址。

# By default, if no "bind" configuration directive is specified, **Redis listens**

**# for connections from all the network interfaces available on the server**.

# It is possible to listen to just one or multiple selected interfaces using

# the "bind" configuration directive, followed by one or more IP addresses.

#

# Examples:

# bind 192.168.1.100 10.0.0.1

# bind 127.0.0.1 ::1

**masterauth**：从节点需要配置的主节点密码。

说明：如果主节点配置了密码，从节点没有配置masterauth或配置密码错误，则一直循环尝试连接。

**rename-command oldcommand newcommand** 对敏感命令重命名，以防误操作。

如rename-command keys dfsdklgdasd

## sentinel配置说明与分析(P440)

## sentinel配置说明与分析(P441)

# 系统状态信息的获取：info [section] (P430)

## info [section] 命令

通过info XXX命令获取Redis的系统状态信息，使用方法有三种：

### **info**：部分系统状态统计信息；

### **info all**：全部系统状态统计信息；

### **info [section]：特定的系统状态统计信息**，其中section忽略大小写。

一般都是有目的地获取关于Redis的某部分信息，所以哪些模块包括哪些统计信息很重要：setion主要分成10部分。

## section主要包括以下10部分：

### **server**：Redis服务器信息;(常用)

### **clients**：Redis客户端信息;(常用)

注意：由于当前连接的Redis服务器是一个，故server后不加s；一个Redis服务器可以连接多个客户端，故是加s的clients。

### **replication**：主从复制信息;(常用)

### memory：内存信息；

### persistence：持久化信息；

### stats：全局统计信息；

### CPU：CPU消耗信息；

### commandstats：命令统计信息；

### **cluster**：集群信息；

### keyspace：数据库键统计信息；

# info [section] 各个部分详解(P431)

## info server (见P431)

info server获取Redis服务器的状态统计信息，包括Redis版本号、TCP端口号、run\_id、运行模式redis\_mode、运行的天数和秒数、配置文件、操作系统的的版本等信息。

Redis服务器的**运行模式redis\_mode**有3种，分别为**Standalone、Sentinel、Cluster**。

run\_id在主从复制中有所应用，根据run\_id唯一标识一个主从Redis节点，而不是根据host:port。

127.0.0.1:6379> **info server**

**# Server**

**redis\_version**:3.2.12 #Redis服务器版本号

redis\_git\_sha1:00000000 #Git SHA 1

redis\_git\_dirty:0 #Git dirty flag

redis\_build\_id:cf8f4582898b66c #Redis build id

**redis\_mode**:standalone #Redis服务器的运行模式

**os**:Linux 3.10.0-514.26.2.el7.x86\_64 x86\_64 #Redis所在机器的操作系统

arch\_bits:64 #架构(32位或64位)

multiplexing\_api:epoll #Redis所使用的事件处理机制

gcc\_version:4.8.5 #编译Redis时所使用的GCC版本

**process\_id**:2925 #Redis服务进程的PID

**run\_id**:c20fdc83f643da30ebce5add7ba1a0e32eb57a73 #Redis服务的标识符

**tcp\_port**:6379 #Redis服务的监听端口

**uptime\_in\_seconds**:67468 #自Redis服务启动以来，运行的秒数

**uptime\_in\_days**:0 #自Redis服务启动以来，运行的天数

hz:10 #**serverCron每秒运行次数**

lru\_clock:6037503 #以分钟为单位进行自增的时钟，用于LRU管理

**executable**:**/usr/src/redis-3.2.12/redis-server**

**config\_file**:/usr/src/redis-3.2.12/./redis-6379.conf #**Redis的配置文件**

## info clients 与 client list (见P434)

### info clients

**info clients**获取客户端的统计信息，包括连接客户端数量、阻塞命令连接数、输入输出缓冲区等。

127.0.0.1:6380> info clients

# Clients

**connected\_clients**:2 #当前客户端连接数

client\_longest\_output\_list:0 #**当前所有输出缓冲区中队列对象个数的最大值**

client\_biggest\_input\_buf:0 #**当前所有输入缓冲区中占用的最大容量**

**blocked\_clients**:0 #正在等待阻塞命令的客户端数量，**阻塞命令如blpop**

### client list

**client list**用于详细列出当前客户端的信息：查看该Redis服务器的所有的客户端。

主从复制的Redis服务器，彼此把对方也模拟成一个客户端，**flags=M、S**。flags=N表示普通的连接客户端。

下面看一个**树状拓扑复制结构**的中间复制层的Redis节点：

127.0.0.1:6380> **client list**

id=11 addr=127.0.0.1:36872 fd=8 name= age=22553 idle=0 **flags=N** db=0 sub=0 psub=0 multi=-1 qbuf=0 qbuf-free=32768 obl=0 oll=0 omem=0 events=r cmd=client#通过redis-cli连接的

id=189 addr=127.0.0.1:6379 fd=7 name= age=4605 idle=6 **flags=M** db=0 sub=0 psub=0 multi=-1 qbuf=0 qbuf-free=0 obl=0 oll=0 omem=0 events=r cmd=ping

id=190 addr=127.0.0.1:37300 fd=6 name= age=2154 idle=1 **flags=S** db=0 sub=0 psub=0 multi=-1 qbuf=0 qbuf-free=0 obl=0 oll=0 omem=0 events=r cmd=replconf

## info replication (见P434)

查看**当前Redis服务器的主从复制配置信息及使用情况**。

主从复制中关键组件：**复制偏移量、复制积压缓冲区、运行时id**。

主节点和从节点都存储着复制偏移量master\_repl\_offset和slave\_repl\_offset；复制积压缓冲区信息对应repl\_backlog\_xxx用于部分复制出现故障时的备份。

(run\_id通过info server获取)

在主从复制中，Redis的角色role分为**master和slave**;在client list中也通过flags标识出来M和S。

### 当role:master

连接的slave个数及详细信息，其中offset为从节点发给主节点的复制偏移量。

**主节点的复制偏移量**：master\_repl\_offset

主节点的复制积压缓冲区的使用信息:**repl\_backlog\_xxx**。

127.0.0.1:6379> info replication

**# Replication**

**role:master**

**connected\_slaves:3**

**slave0:ip=127.0.0.1,port=6380,state=online,offset=27749,lag=0**

**slave1:ip=127.0.0.1,port=6382,state=online,offset=27749,lag=1**

**slave2:ip=127.0.0.1,port=6381,state=online,offset=27749,lag=0**

**master\_repl\_offset:27749**

repl\_backlog\_active:1 #开启了复制积压缓冲区

repl\_backlog\_size:1048576 #复制缓冲区的最大长度，默认1M=220

repl\_backlog\_first\_byte\_offset:2 #起始偏移量

repl\_backlog\_histlen:27748 #已保存数据的有效长度

复制积压缓冲区内的可用偏移量范围为：

[**repl\_backlog\_first\_byte\_offset**, **repl\_backlog\_first\_byte\_offset+ repl\_backlog\_histlen**]

### 当role:slave

role:slave的Redis可能只作为从节点，也有可能作为树状拓扑结构的中间复制层，也有自己的slave。

仅做slave的时候，一般复制积压缓冲区是关闭的。

127.0.0.1:6380> info replication

**# Replication**

**role:slave**

master\_host:127.0.0.1

master\_port:6379

**master\_link\_status**:up

master\_last\_io\_seconds\_ago:7

master\_sync\_in\_progress:0

**slave\_repl\_offset:28883 #从节点复制偏移量**

**slave\_priority:100**

**slave\_read\_only:1 #默认只读模式**

connected\_slaves:0

**master\_repl\_offset:0**

repl\_backlog\_active:0 #未开启复制积压缓冲区

repl\_backlog\_size:1048576 #默认大小1M

repl\_backlog\_first\_byte\_offset:2 #起始偏移量

repl\_backlog\_histlen:154

当主从复制拓扑为树状结构时，充当中间复制层的Redis节点，角色以slave为主，其次也会显示连接到该节点的从节点个数及信息：

127.0.0.1:6380> info replication

# Replication

**role:slave**

master\_host:127.0.0.1

master\_port:6379

master\_link\_status:up

master\_last\_io\_seconds\_ago:0

master\_sync\_in\_progress:0

slave\_repl\_offset:29051

slave\_priority:100

slave\_read\_only:1

**connected\_slaves:1**

slave0:ip=127.0.0.1,port=6382,state=online,**offset=15**,lag=0

**master\_repl\_offset:15**

repl\_backlog\_active:1

repl\_backlog\_size:1048576

repl\_backlog\_first\_byte\_offset:2

repl\_backlog\_histlen:14

## info cluster

info cluster获取Redis集群模块的统计信息，目前只有一个统计信息，标识当前Redis是否为Cluster模式。

127.0.0.1:6380> **info cluster**

# Cluster

**cluster\_enabled:0** #当前Redis节点是否为cluster模式，1是，0否。

## info memory

info memory命令获取Redis的内存使用、系统内存使用、碎片率、内存分配器等相关统计信息。

后面带有human的，表示把字节转换成人类可读的形式，如K、M或G。

**mem\_fragmentation\_ratio**:0.09 #内存碎片率；也就是used\_memory\_rss/ **used\_memory的比值。**

127.0.0.1:6379[1]> **info memory**

**# Memory**

**used\_memory:118283272 #Redis分配器分配的内存总量**

**used\_memory\_human:112.80M**

used\_memory\_rss:10371072 #从操作系统角度，Redis进程占用的物理内存总量

used\_memory\_rss\_human:9.89M

**used\_memory\_peak:118325000 #内存使用的最大值**

used\_memory\_peak\_human:112.84M

total\_system\_memory:1928933376

total\_system\_memory\_human:1.80G

used\_memory\_lua:37888

used\_memory\_lua\_human:37.00K

maxmemory:0

maxmemory\_human:0B

maxmemory\_policy:noeviction

**mem\_fragmentation\_ratio:0.09 #内存碎片率**

**mem\_allocator:jemalloc-4.0.3 #Redis所使用的内存分配器，默认为jemallo。**

## info persistence

info persistence 获取Redis持久化的统计信息，包括RDB和AOF两种持久化的统计信息。

127.0.0.1:6379[1]> **info persistence**

**# Persistence**

loading:0 #是否正在加载持久化文件

**rdb\_changes\_since\_last\_save**:0

rdb\_bgsave\_in\_progress:0 #RDB的bgsave操作是否在进行中

rdb\_last\_save\_time:1532766679

rdb\_last\_bgsave\_status:ok

rdb\_last\_bgsave\_time\_sec:0 #上次basave操作使用的时间(秒数)

**rdb\_current\_bgsave\_time\_sec:-1**

**aof\_enabled:0 #是否开启了AOF功能：0否，1是。**

aof\_rewrite\_in\_progress:0 # AOF的rewrite操作是否在进行中

aof\_rewrite\_scheduled:0 #是否在RDB的bgsave操作结束后执行AOF的rewrite操作

aof\_last\_rewrite\_time\_sec:-1

aof\_current\_rewrite\_time\_sec:-1

aof\_last\_bgrewrite\_status:ok

aof\_last\_write\_status:ok

## keyspace

info keyspace 获取Redis每个数据库的所有键的统计信息。

127.0.0.1:6379[1]> **info keyspace**

# Keyspace

**db0**:keys=10,expires=0,avg\_ttl=0

**db1**:keys=1,expires=0,avg\_ttl=0

keys表示当前数据库的key总数，**expires表示设有过期时间的key总数**，avg-ttl平均存活时间。

## info stats

info stats获取Redis的基础统计信息，包括：连接、命令、网络、过期、同步等统计信息。

127.0.0.1:6379[1]> info stats

# Stats

**total\_connections\_received**:18 #连接过的客户端的数量

**total\_commands\_processed**:70871 #执行过的命令总数

**instantaneous\_ops\_per\_sec**:0 #每秒处理命令条数

**total\_net\_input\_bytes:2684995**

**total\_net\_output\_bytes:129955**

instantaneous\_input\_kbps:0.02

instantaneous\_output\_kbps:0.00

**rejected\_connections:0**

sync\_full:1

sync\_partial\_ok:0

sync\_partial\_err:0

**expired\_keys:2**

evicted\_keys:0

keyspace\_hits:7

keyspace\_misses:3

pubsub\_channels:0

pubsub\_patterns:0

**latest\_fork\_usec:813 #最近一次fork操作消耗的时间(微秒)**

migrate\_cached\_sockets:0

## info CPU

**info CPU**获取Redis进程和子进程对于CPU消耗的一些统计信息。

127.0.0.1:6379[1]> info cpu

# CPU

**used\_cpu\_sys:22.75**

**used\_cpu\_user:18.67**

used\_cpu\_sys\_children:0.01

used\_cpu\_user\_children:0.00

## info commandstats

**info commandstats**获取Redis命令的统计信息，包括**各个命令的命令名、总次数、总耗时、平均耗时**等。

**calls总次数、usec总耗时、平均耗时usec\_per\_call**。(单位:微秒)

**127.0.0.1:6379[1]> info commandstats**

# Commandstats

cmdstat\_get:**calls=5,usec=30,usec\_per\_call=6.00**

cmdstat\_set:calls=1,usec=6,usec\_per\_call=6.00

cmdstat\_setnx:calls=3,usec=19,usec\_per\_call=6.33

cmdstat\_setex:calls=2,usec=13,usec\_per\_call=6.50

cmdstat\_lpop:calls=1,usec=16,usec\_per\_call=16.00

cmdstat\_spop:calls=1,usec=24,usec\_per\_call=24.00

cmdstat\_select:calls=1,usec=9,usec\_per\_call=9.00

cmdstat\_keys:calls=4,usec=75,usec\_per\_call=18.75

cmdstat\_auth:calls=6,usec=14,usec\_per\_call=2.33

cmdstat\_psync:calls=1,usec=1550,usec\_per\_call=1550.00

cmdstat\_replconf:calls=71062,usec=95173,usec\_per\_call=1.34

cmdstat\_info:calls=43,usec=1161,usec\_per\_call=27.00

cmdstat\_ttl:calls=5,usec=15,usec\_per\_call=3.00

cmdstat\_command:calls=1,usec=387,usec\_per\_call=387.00