**redis还原dump.rdb**

先停止redis服务，然后将新的dump.rdb覆盖掉旧的，再启动redis服务即可

# redis文件中的dump.rdb文件是怎么生成的

默认情况下 每隔一段时间redis服务器程序会自动对数据库做一次遍历，把内存快照写在一个叫做“dump.rdb”的文件里，这个持久化机制叫做SNAPSHOT。

有了SNAPSHOT后，如果服务器宕机，重新启动redis服务器程序时redis会自动加载dump.rdb，将数据库状态恢复到上一次做SNAPSHOT时的状态。

**rdb是redis支持的一种持久化方式，在进行rdb dump时，redis直接将内存中的所有数据写到磁盘的文件上，以实现持久化。以下三种情况会使用rdb dump：**

通过save/bgsave命令保存数据

满足配置文件中的save指令条件触发持久化操作

Slave初始连接上Master时，Master会dump一份数据，传输给从库。

rdb有个隐蔽的问题需要注意，**就是Redis的工作路径和配置的rdb dump文件的路径必须在同一个硬盘分区上。**如果处在不同的分区，所有这些需要使用rdb dump的操作都会失败。

rdb方式的持久化是通过快照完成的，当符合一定条件时redsi会自动将内存中的所有数据进行快照并存储到硬盘上。默认存储在redis根目录的dump.rdb文件中。  
rdb是redis默认采用的持久化方式，配置信息在配置文件redis.conf中  
定期将内存数据生成快照(即某个时间点上数据的备份) 然后存储在硬盘上

快照执行时机：  
save 900 1：表示900秒内至少一个键被更改则进行快照。  
save 300 10  
save 60 10000

手动执行save或者bgsave命令让redis执行快照。  
两个命令的区别在于，save是由主进程进行快照操作，会阻塞其它请求。bgsave会通过fork子进程进行快照操作。  
redis实现快照的过程：  
1：redis使用fork函数复制一份当前进程的副本(子进程)  
2：父进程继续接收并处理客户端发来的命令，而子进程开始将内存中的数据写入硬盘中的临时文件  
3：当子进程写入完所有数据后会用该临时文件替换旧的RDB文件，至此，一次快照操作完成。

注意：redis在进行快照的过程中不会修改RDB文件，只有快照结束后才会将旧的文件替换成新的，也就是说任何时候RDB文件都是完整的。  
这就使得我们可以通过定时备份RDB文件来实现redis数据库的备份  
RDB文件是经过压缩的二进制文件，占用的空间会小于内存中的数据，更加利于传输。

文件修复：redis-check-dump  在启动redis失败时，用于修复dump文件

rdb的优缺点  
优点：由于存储的有数据快照文件，恢复数据很方便。  
缺点：会丢失最后一次快照以后更改的所有数据。

AOF 保存的数据方案是最完整的，如果同时开启了rdb和aof下，会采用aof方式  
   
aof方式的持久化是通过日志文件的方式。默认情况下redis没有开启aof，可以通过参数appendonly参数开启。  
appendonly yes  
aof文件的保存位置和rdb文件的位置相同，都是dir参数设置的，默认的文件名是appendonly.aof，可以通过appendfilename参数修改  
appendfilename appendonly.aof  
aof日志文件重写  
auto-aof-rewrite-percentage 100  
auto-aof-rewrite-min-size 64mb  
手动执行bgrewriteaof进行重写  
redis写命令同步的时机  
appendfsync always 每次都会执行  
appendfsync everysec 默认 每秒执行一次同步操作（推荐）  
appendfsync no不主动进行同步，由操作系统来做，30秒一次  
文件修复：redis-check-aof  
动态切换redis持久方式，从 RDB 切换到 AOF（支持Redis 2.2及以上）  
CONFIG SET appendonly yes  
CONFIG SET save ""（可选）

redis支持两种存储方式：snapshot(快照)和aof(append only mode)，快照是定时将内存快照持久化到硬盘（crash会丢失数据），aof是在写入数据的同时将操作命令保存到日志（不会丢失数据，但操作日志管理维护成本高）