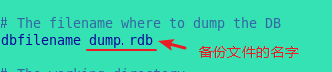
# RBM备份与恢复

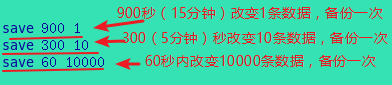
在redis读写数据时，会根据redis.conf配置文件进行自动的备份，在打开redis服务的目录下，创建一个dump.rdb文件，打开redis的时候，会把备份到这个文件的内容加载到内存中，redis信息恢复，在备份的时候，会开一个新的进程进行io操作，主进程无需io

1. 配置详解

备份文件的名字



更新备份的时间



如果设置save “””就可以不使用自动备份

1. 操作详解

①根据配置文件的备份规则，redis进行备份，每次的备份会把之前的备份覆盖掉

②需要写一个脚本或其他，把每次本备份的文件保存起来，一般会把备份文件（dump.rdb）放置到专门的备份机器上，防止操作redis的机器出现物理故障，无法还原备份。

③如果进行误操作，如误删除了所有的key，可以找到删除之前备份的dump.rdb文件，复制到目录下，重新打开redis服务即可恢复。

注意：

1. 每次redis进行shutdown操作时，就都会进行一次备份
2. 可以执行save指令进行手动备份，但是会占据cpu资源，备份完成之前无法进行其他操作，bgsave会新开一个线程备份，可以继续执行redis操作
3. 可以通过lastsave指令获取最后备份的时间

缺点：

1. 每次执行快照备份的时候，会把数据库所有资源全部form到内存中（内存中的复制），然后写入到磁盘，太消耗内存了
2. 可能存在操作完数据，还没有进行备份，服务器突然出现故障，最后的数据没有被保存

优点：

1. 自动备份，减少人工成本，备份到一个文件中去，方便管理
2. 速度相对于AOF会更快

适合大型数据的恢复和备份，对精准度要求不是很高的

# AOF备份与恢复

介绍：

以日志的形式记录每一个写操作，将redis所有的写操作记录下来，只许追加到文件但不可修改文件，redis在启动之初会读取文件重新构建数据，redis重启的话，就会根据日志文件将写的操作从头到尾执行一遍，完成恢复数据，需要持续的进行io操作

1. 配置修改



1. 备份详解

每次执行语句就会备份到开启redis服务目录下的appendonly.aof文件中。

如果误操作删除数据，就可以把有数据的appendonly.aof放到执行目录下

config get dir ，就可以恢复数据了。

如果文件被损坏，用redis的redis-check-aof --fix appendonly.aof指令修复（bin目录下）

补充：当指令越来越多的时候，appendonly.aof 文件越来越大，redis会在aof文件超出设定标准的时候进行优化压缩。



优点：

1. 更准确的恢复数据，最多有1秒的数据丢失
2. 文件容易分析，能够很轻易的看懂

缺点：

1. 随着指令的增多，文件会越来越大，要比rdb文件大很多
2. 恢复速度要慢于rdb

appendonly.aof和dump.rdb可以同时存在，若同时存在会优先找appendonly.aof