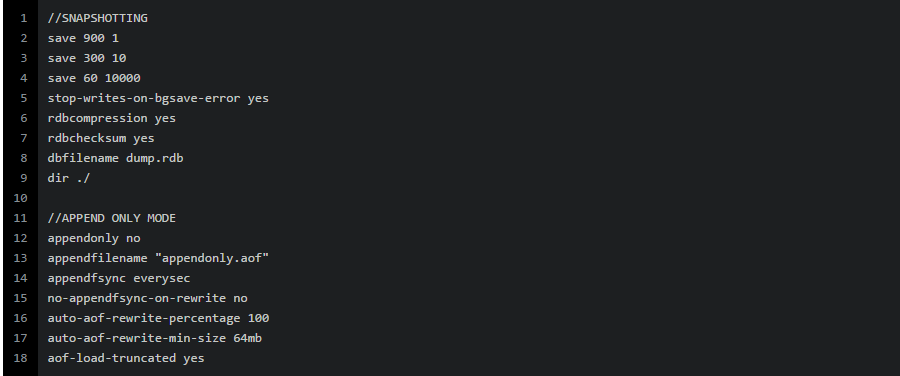
Redis数据持久化

Redis提供了两种数据持久化机制，一种是基于snapshotting快照机制，他可以将某一时刻内存中全部数据都存储到硬盘里。

另外一种是AOF(append only file)机制，对硬盘追加写命令。

两种方式可以根据实际需求自由配置，配置文件redis.conf如下所示、



AOF是默认关闭的。

**快照持久化**

用户可以将快照复制到其他服务器已达到备份的效果，也可以就将快照留在原地以便重启服务器时使用，快照文件存在dbfilename选项指定的文件中，并存储在dir指定的路径，根据默认配置即./dump.rdb，如果在新的快照文件创建之前，Redis、操作系统或者硬件三者任意一个崩溃了，Redis会丢失最近一次创建快照之后写入的所有数据。创建快照有以下几种方式：

客户端主动发送BGSAVE命令来创建一个快照，BGSAVE命令返回Background saving started，Redis是通过调用fork来创建子进程完成快照写入硬盘的，父进程可以继续响应命令请求。

客户端主动发送SAVE命令来创建一个快照，接到SAVE命令的Redis服务器在快照创建完毕之前将不再响应任何其他请求，此命令不常用。

配置save选项，例如save 60 10000，那么从Redis最近一次创建快照之后开始算起，当”60秒之内有10000次写入”这个条件被满足时，Redis就会自动出发BGSAVE命令，如果设置了多个save配置选项，当任意一个满足时，Redis就会出发BGSAVE。默认行为配置了三个阙值。

当Redis通过SHUTDOWN命令接受到关闭服务器的请求时，或者接受到标准TERM信号时，会执行一个SAVE命令，阻塞所有客户端，并在SAVE命令执行完毕之后关闭服务器。

当一个Redis服务器连接另一个Redis服务器，并向对方发送SYNC命令开始一次复制操作的时候，如果主服务器没有或者并非刚刚执行BGSAVE操作，那么主服务器就会执行BGSAVE命令。

由于快照持久化会会在系统发生崩溃时丢失数据，因此只适用于那些即使丢失一部分数据叶不会造成问题的应用程序，如果不能接受这样的损失，可以参考后面AOF持久化。下面列举一些使用于快照持久化的场景：

**1. 个人开发**

个人开发服务器上，考虑到降低快照持久化带来的资源消耗，可以只设置sava 900 1，意思是距离上一次成功生成快照已经超过900秒，并且在此期间至少执行了一次写入操作，Redis就会自动开始一次新的BGSAVE操作。

**2. 对日志进行聚合计算**

在处理日志的同时，记录被处理日志的文件以及偏移量，如果Redis奔溃了而未能生成新的快照，可以从最后一次生成快照开始重新处理日志文件即可。

**3. 大数据**

当Redis存储数据量只有几个GB的时候，使用快照来保存数据是没有问题的，生成快照的时间叶非常短。但随着Redis占用的内存越来越多时，BGSAVE在创建子进程时耗费的时间也越来越多，所以选择合适的创建快照方式以及妥善地处理可能出现的数据丢失，对快照持久化数据来说相当重要。

**AOF持久化**

简单来说，AOF持久化会将被执行的写命令写到AOF文件的末尾，以此来记录数据的变化。因此，Redis只要从头到位重新执行一次AOF文件包含的所有命令，就可以恢复AOF文件所记录的数据集。三种AOF策略。

Always : 每个命令都要同步写入硬盘

Everysec: 每秒执行一次同步，显式的将命令同步到硬盘

No : 让操作系统来决定应该何时进行同步

向硬盘持久化写入数据时，首先会将数据写入系统缓冲区，再有操作系统决定何时写入硬盘。Sync操作就是操作系统将系统文件写入硬盘的过程。同步操作会一直处于阻塞状态，在整个同步写入硬盘操作中，当同步操作完成时，只有硬盘不坏，就不会对数据造成影响。

**appendfsync always选项是最安全同时也是最慢的**，某些情况下还可能会影响固态硬盘的使用寿命，所以慎用！为了兼顾数据安全和写入性能，可以考虑使用appendfsync everysec，也是Redis默认行为。Redis每秒同步一次AOF文件的性能和不适用任何持久化特性时的性能相差无几，而每秒一次的同步，当系统出现故障时，也最多只会丢失一秒内产生的数据。

**appendfsync no**选项是将写入硬盘的决定权交给操作系统，如果硬盘的写入速度不够快，缓冲区被填满时，Redis的写入操作将被阻塞，从而导致Redis处理命令请求的速度变慢，所以appendfsync no也不推荐使用。

AOF文件在Redis重启以后会重新加载到内存中。