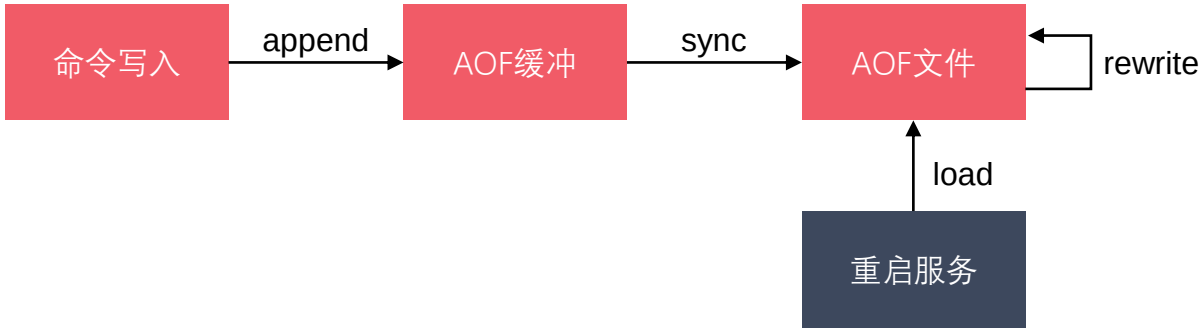
AOF持久化

01 / AOF简介

1. AOF（Append Only File），解决了数据持久化的实时性，是目前Redis持久化的主流方式；
2. AOF以独立日志的方式，记录了每次写入命令，重启时再重新执行AOF文件中的命令来恢复数据；
3. AOF的工作流程包括：命令写入（append）、文件同步（sync）、文件重写（rewrite）、重启加载（load）。



02 / 启用AOF

AOF默认不开启，需要修改配置项来启用它：

```
appendonly yes                    # 启用AOF
appendfilename "appendonly.aof"   # 设置文件名
```

AOF以文本协议格式写入命令，如：

```
*3\r\n$3\r\nset\r\n$5\r\nhello\r\n$5\r\nworld\r\n
```

AOF为什么直接采用文本协议格式？

1．文本协议具有很好的兼容性；

2．直接采用文本协议格式，可以避免二次处理的开销；

3．文本协议具有可读性，方便直接修改和处理。

- 为了提高程序的写入性能，现代操作系统会把针对硬盘的多次写操作优化为一次写操作：
  1．当程序调用write对文件写入时，系统不会直接把书记写入硬盘，而是先将数据写入内存的缓冲区中；
  2．当达到特定的时间周期或缓冲区写满时，系统才会执行flush操作，将缓冲区中的数据冲洗至硬盘中；
- 这种优化机制虽然提高了性能，但也给程序的写入操作带来了不确定性：
  1．对于AOF这样的持久化功能来说，冲洗机制将直接影响AOF持久化的安全性；
  2．为了消除上述机制的不确定性，Redis向用户提供了appendfsync选项，来控制系统冲洗AOF的频率；
  3．Linux的glibc提供了fsync函数，可以将指定文件强制从缓冲区刷到硬盘，上述选项正是基于此函数。

取值	说明
always	每执行一个写入命令，就对AOF文件执行一次冲洗操作； 这种情况下，服务器在停机时最多丢失一个命令，但这种方式会大大降低Redis服务器的性能；
everysec	每隔1秒，就对AOF文件执行一次冲洗操作； 这种情况下，服务器在停机时最多丢失1秒之内的命令，是一种兼顾性能和安全性的折中方案；
no	不主动对AOF文件执行冲洗操作，由操作系统决定何时冲洗； 这种情况下，服务器在停机时将丢失最后一次冲洗后执行的写入命令，数据量取决于系统的冲洗频率；

05 / AOF的优缺点

- 优点
  1. 与RDB持久化可能丢失大量的数据相比，AOF持久化的安全性要高很多；
  2. 通过使用everysec选项，用户可以将数据丢失的时间窗口限制在1秒之内；
- 缺点
  1. AOF文件存储的是协议文本，它的体积要比二进制格式的".rdb"文件大很多；
  2. AOF需要通过执行AOF文件中的命令来恢复数据库，其恢复速度比RDB慢很多；
  3. AOF在进行重写时也需要创建子进程，在数据库体积较大时将占用大量资源，会导致服务器的短暂阻塞。

牛客大学
-专业求职辅导-

THANKS

关注【牛客大学】公众号
回复“牛客大学”获取更多求职资料