redis数据淘汰策略

本文讲的是当redis设定了最大内存之后，缓存中的数据集大小超过了一定比例，实施的淘汰策略，不是删除过期键的策略，虽然两者非常相似。

概述

在redis中，允许用户设置最大使用内存大小。通过配置redis.conf中的maxmemory这个值来开启内存淘汰功能，在内存限定的情况下是很有用的。设置最大内存大小可以保证redis对外提供稳健服务。

redis内存数据集大小上升到一定大小的时候，就会施行数据淘汰策略。redis提供6种数据淘汰策略，通过maxmemory-policy设置：

1. volatile-lru

从已设置过期时间的数据集（server.db[i].expires）中挑选最近最少使用的数据淘汰。

1. volatile-ttl

从已设置过期时间的数据集（server.db[i].expires）中挑选将要过期的数据淘汰。

1. volatile-random

从已设置过期时间的数据集（server.db[i].expires）中任意选择数据淘汰。

1. allkeys-lru

从数据集（server.db[i].dict）中挑选最近最少使用的数据淘汰。

1. allkeys-random

从数据集（server.db[i].dict）中任意选择数据淘汰。

1. no-enviction

禁止驱逐数据。

redis确定驱逐某个键值对后，会删除这个数据并将这个数据变更消息发布到本地（AOF持久化）和从机（主从连接）。

LRU数据淘汰机制

在服务器配置中保存了lru计数器server.lrulock，会定时（redis定时程序serverCorn()）更新，server.lrulock的值是根据server.unixtime计算出来的。

另外，从struct redisObject中可以发现，每一个redis对象都会设置相应的lru。可以想象的是，每一次访问数据的时候，会更新redisObject.lru。

LRU数据淘汰机制是这样的：在数据集中随机挑选几个键值对，取出其中lru最大的键值对淘汰。所以，你会发现，redis并不是保证取得所有数据集中最近最少使用（LRU）的键值对，而只是随机挑选的几个键值对中。

TTL数据淘汰机制

redis数据集数据结构中保存了键值对过期时间表，即redisDb.expires。和LRU数据淘汰机制类似，TTL数据淘汰机制是这样的：从过期时间表中随机挑选几个键值对，取出其中ttl最大的键值对淘汰。同样你会发现，redis并不是保证取得所有过期时间的表中最快过期的键值对，而只是随机挑选的几个键值对中的。

总结，redis每服务客户端执行一个命令的时候，会检测使用的内存是否超额。如果超额，即进行数据淘汰。

适用场景

下面看看几种策略的适用场景：

* allkeys-lru

如果我们的应用对缓存的访问符合幂律分布（也就是存在相对热点数据），或者我们不太清楚我们应用的缓存访问分布状况，我们可以选择allkeys-lru策略。

* allkeys-random

如果我们的应用对于缓存key的访问概率相等，则可以使用这个策略。

* volatile-ttl

这种策略使得我们可以向redis提示哪些key更适合被eviction。

另外，volatile-lru策略和volatile-random策略适合我们将一个Redis实例既应用于缓存和又应用于持久化存储的时候，然而我们也可以通过使用两个redis实例来达到相同的效果，值得一提的是将key设置过期时间实际上会消耗更多的内存，因此我们建议使用allkeys-lru策略从而更有效率地使用内存。