# 1 缓存

## 1.1 基于Redis的设计与实践

1.1.1 秒杀抢购

**背景**

秒杀抢购不仅仅是数据的读取集中并发，同时也是数据写入的集中并发。抢购那一下太耗费服务器资源，在同一时间段内涌入的人数大大超过了服务器的负载，根本承受不了，CPU占用率很多时候都接近了100%，请求的积压也很严重，从请求接入到数据的读取都有问题，尤以数据的读取更为严重。

问题是发现了，表面上看起来解决没那么简单。应用层的问题解决起来相对容易，实在不行多加点机器也能解决；但数据的问题就不是那么简单了，靠增加机器来解决是不行的。大部分关系型数据库没有真正的分布式解决方案，最多做一个主从分离或多加从库分担读取的压力，但因为秒杀抢购是数据集中式超高并发的读，所以一般的关系型数据库因为它本身局限性很难支撑这样瞬间突发的高并发，就算勉强顶上，也会因为秒杀抢购还有写的高并发，影响到读节点的数据同步问题。当然也可以拼命提升一下服务器的硬件性能，比如换最好的CPU，把硬盘换成SSD等等，但效果应该是不会太显著，没有解决本质的问题，还比较费钱。

为什么呢？先看那个主从+Keepalived的方案，这本来是个很好的方案，但是忽略了主数据节点挂掉的情况。我们在前面说过，Redis的单进程、单线程设计是其简单和稳定的基石，只要不是服务器发生了故障，在一般情况下是不会挂的。但同时，单进程、单线程的设计会导致Redis接收到复杂指令时会忙于计算而停止响应，可能就因为一个zset或者keys之类的指令，Redis计算时间稍长，Keepalived就认为其停止了响应，直接更改虚IP的指向，然后做一次主从切换。过不了多久，zset和keys之类的指令又会从客户端发送过来，于是从机上又开始堵塞，Keepalived就一直在主从机之间不断地切换IP。终于主节点和从节点都堵了，Keepalived发现后，居然直接将虚IP释放了，然后所有的客户端都无法连接Redis了，只能等运维到线上手工绑定才行。

数据落盘也引起了很大的问题，RDB属于非阻塞式的持久化，它会创建一个子进程来专门把内存中的数据写入RDB文件里，同时主进程可以处理来自客户端的命令请求。但子进程内的数据相当于是父进程的一个拷贝，这相当于两个相同大小的Redis进程在系统上运行，会造成内存使用率的大幅增加。如果在服务器内存本身就比较紧张的情况下再进行RDB配置，内存占用率就会很容易达到100%，继而开启虚拟内存和进行磁盘交换，然后整个Redis的服务性能就直线下降了。

另外，像Zset、发布订阅、消息队列、Redis的各种功能不断被介绍，开发者们也在利用这些特性，开发各种应用，但从来没想过这么一个小小的Redis这么多新奇的功能，它缺点在什么地方，什么样的场景是不合适用的。这时Redis在大部分的开发者手上就是像是一把锤子，看什么都是钉子，随时都一锤了事。同时也会渐渐地淡忘了开发的一些细节点和规范，因为用它解决性能的问题是那么轻松简单，于是一些基于Redis的新奇功能就接连不断地出现了：基于Redis的分布式锁、日志系统、消息队列、数据清洗，等等，各种各样的功能不断上线使用，从而引发了各种各样的问题。这时候原来那个救火神器就会变成四处点火的神器，到是Redis堵塞、网卡打爆、连接数爆表等等问题层出不穷，经过这么多折腾，Redis终于也变成了大家的噩梦了。