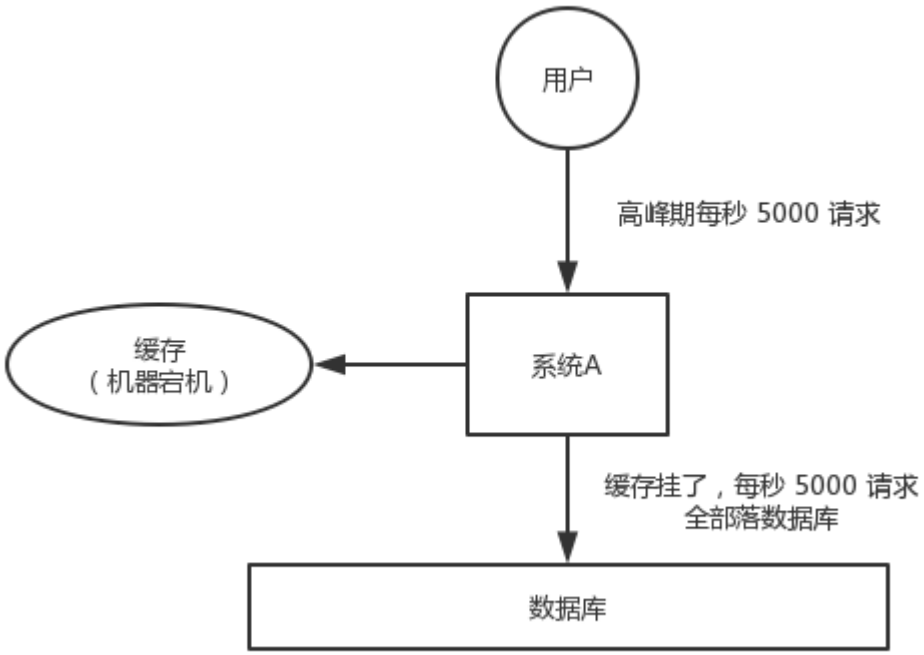
redis 面试

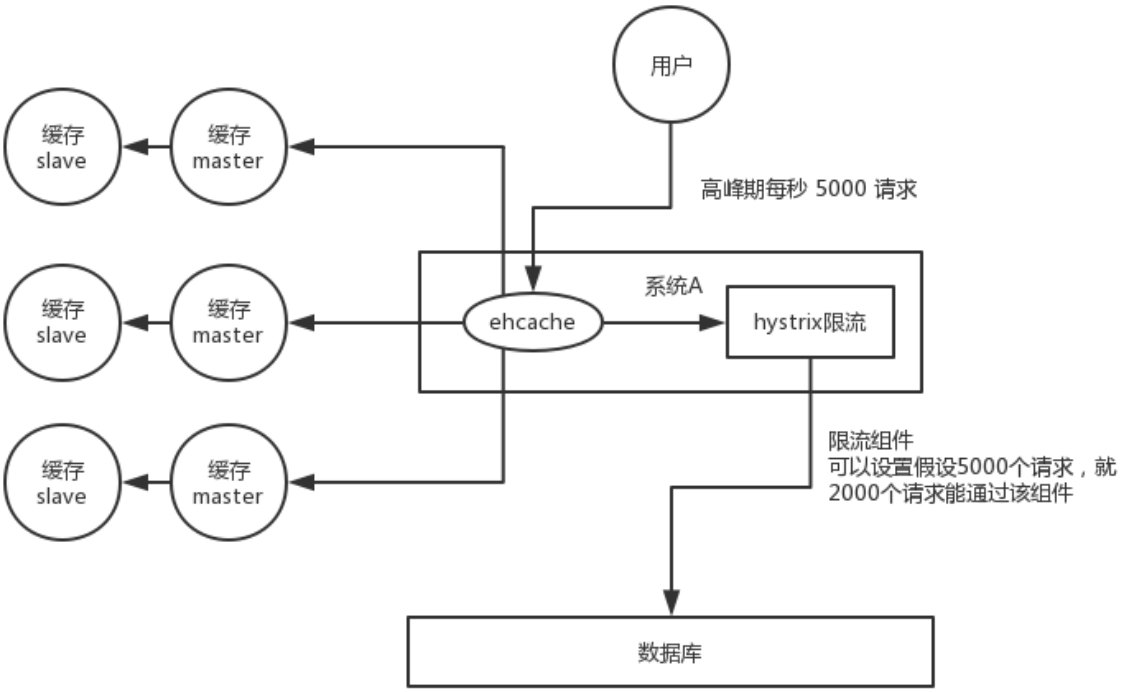
1. redis缓存雪崩、穿透、击穿概念及解决办法

1.1 缓存雪崩

- 对于系统 A，假设每天高峰期每秒 5000 个请求，本来缓存在高峰期可以扛住每秒 4000 个请求，但是缓存机器意外发生了全盘宕机。缓存挂了，此时 1 秒 5000 个请求全部落数据库，数据库必然扛不住，它会报一下警，然后就挂了。此时，如果没有采用什么特别的方案来处理这个故障，DBA 很着急，重启数据库，但是数据库立马又被新的流量给打死了。



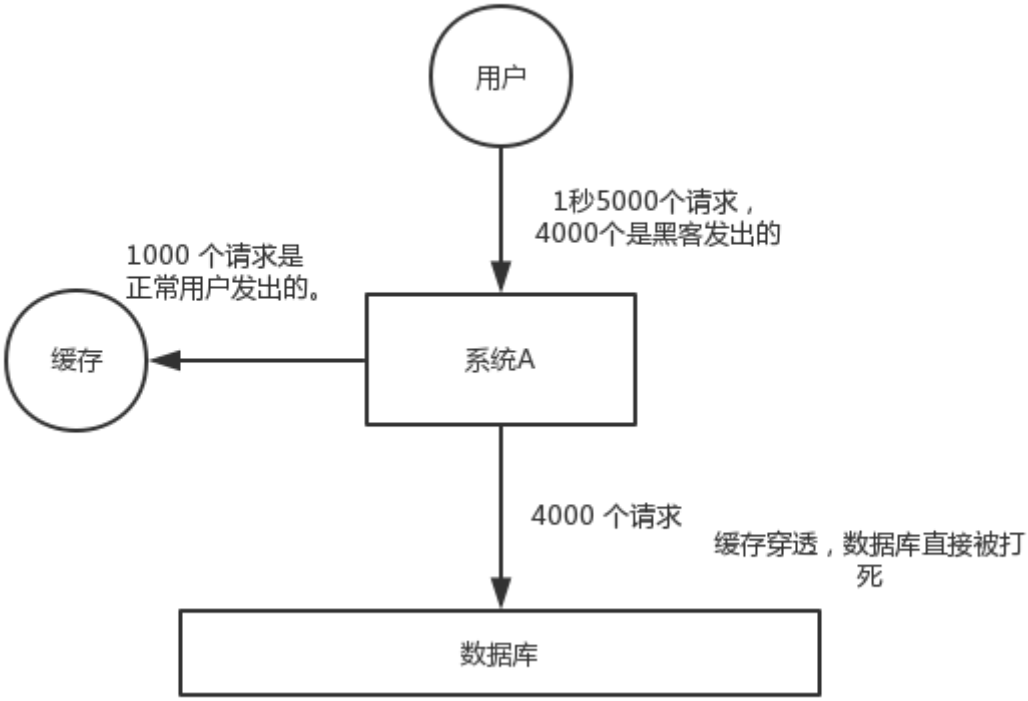
- 大约在 3 年前，国内比较知名的一个互联网公司，曾因为缓存事故，导致雪崩，后台系统全部崩溃，事故从当天下午持续到晚上凌晨 3~4 点，公司损失了几千万。
- 缓存雪崩的事前事中事后的解决方案如下。
    - 事前：redis 高可用，主从+哨兵，redis cluster，避免全盘崩溃。
    - 事中：本地 ehcache 缓存 + hystrix 限流&降级，避免 MySQL 被打死。
    - 事后：redis 持久化，一旦重启，自动从磁盘上加载数据，快速恢复缓存数据。



- 用户发送一个请求，系统 A 收到请求后，先查本地 ehcache 缓存，如果没查到在查 redis。如果 ehcache 和 redis 都没有，再查数据库，将数据库中的结果，写入 ehcache 和 redis 中。
- 限流组件，可以设置每秒的请求，有多少能通过组件，剩余的未通过的请求，怎么办？走降级！可以返回一些默认值，或者友情提示，或者空白的值。
- 好处：
    - 数据库绝对不会死，限流组件确保了每秒只有多少个请求能通过。
    - 只要数据库不死，就是说，对用户来说，$\frac{2}{5}$ 的请求都是可以被处理的。
    - 只要有 $\frac{2}{5}$ 的请求可以被处理，就意味着你的系统没死，对用户来说，可能就是点击几次刷不出来页面，但是多点几次，就可以刷出来一次。

1.2 缓存穿透

- 对于系统 A，假设一秒 5000 个请求，结果其中 4000 个请求是黑客发出的恶意攻击。
- 黑客发出的那 4000 个攻击，缓存中查不到，每次去数据库里查，也查不到。
- 举个例子。数据库 id 是从 1 开始的，结果黑客发过来的请求 id 全部都是负数。这样的话，缓存中不会有，请求每次都“视缓存于无物”，直接查询数据库。这种恶意攻击场景的缓存穿透就会直接把数据库给



打死。

解决方式很简单，每次系统 A 从数据库中只要没查到，就写一个空值到缓存里去，比如 **set -999 UNKNOWN**。然后设置一个过期时间，这样的话，下次有相同的 key 来访问的时候，在缓存失效之前，都可以直接从缓存中取数据。

1.3 缓存击穿

- 缓存击穿，就是说某个 key 非常热点，访问非常频繁，处于集中式高并发访问的情况，当这个 key 在失效的瞬间，大量的请求就击穿了缓存，直接请求数据库，就像是在一道屏障上凿开了一个洞。
- 解决方式也很简单，可以将热点数据设置为永远不过期；或者基于 redis or zookeeper 实现互斥锁，等待第一个请求构建完缓存之后，再释放锁，进而其它请求才能通过该 key 访问数据。