高可用方案

客户端方案

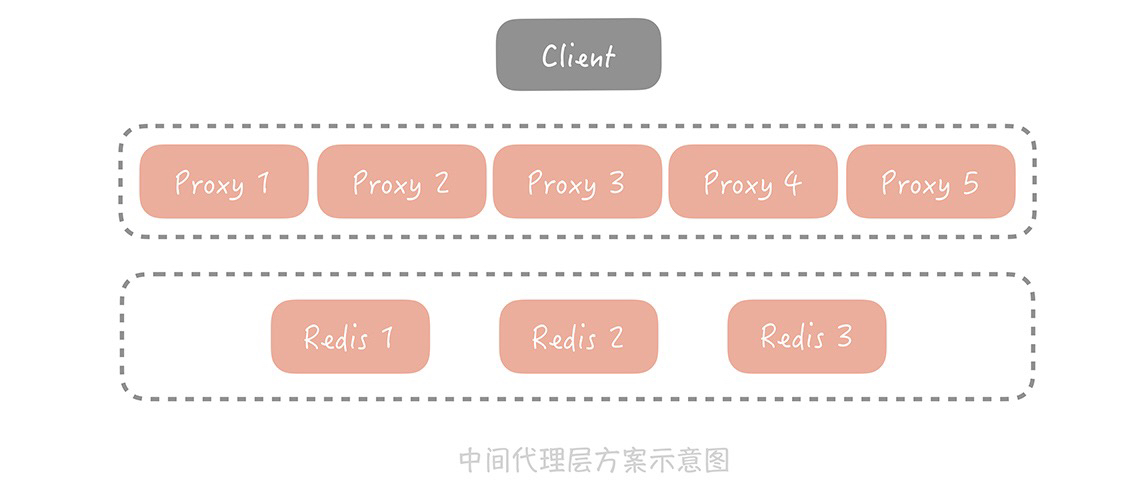
在客户端配置多个缓存的节点，通过缓存写入和读取算法策略来实现分布式，从而提高缓存的可用性。

写入数据时，需要把被写入缓存的数据分散到多个节点中，即进行数据分片；

读数据时，可以利用多组的缓存来做容错，提升缓存系统的可用性。关于读数据，这里可以使用主从和多副本两种策略，两种策略是为了解决不同的问题而提出的。

中间代理层方案

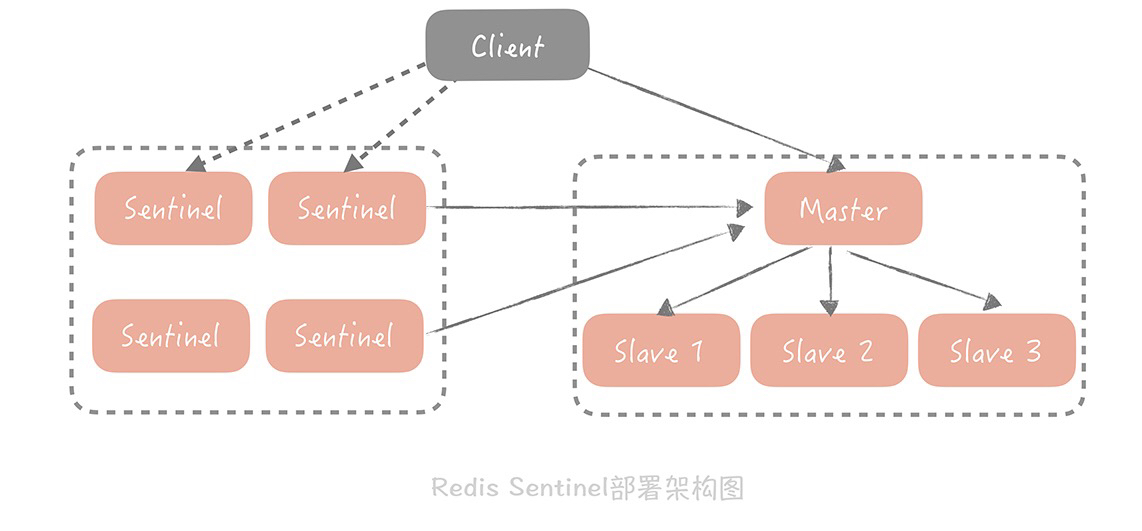
在应用代码和缓存节点之间增加代理层，客户端所有的写入和读取的请求都通过代理层，而代理层中会内置高可用策略，帮助提升缓存系统的高可用。



所有缓存的读写请求都是经过代理层完成的。代理层是无状态的，主要负责读写请求的路由功能，并且在其中内置了一些高可用的逻辑，不同的开源中间代理层方案中使用的高可用策略各有不同。

服务端方案

Redis 2.4 版本后提出的 Redis Sentinel 方案。



Redis Sentinel 也是集群部署的，这样可以避免 Sentinel 节点挂掉造成无法自动故障恢复的问题，每一个 Sentinel 节点都是无状态的。在 Sentinel 中会配置 Master 的地址，Sentinel 会时刻监控 Master 的状态，当发现 Master 在配置的时间间隔内无响应，就认为 Master 已经挂了，Sentinel 会从从节点中选取一个提升为主节点，并且把所有其他的从节点作为新主的从节点。Sentinel 集群内部在仲裁的时候，会根据配置的值来决定当有几个 Sentinel 节点认为主挂掉可以做主从切换的操作，也就是集群内部需要对缓存节点的状态达成一致才行。