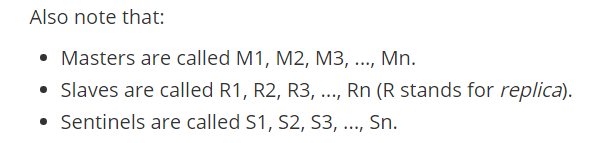
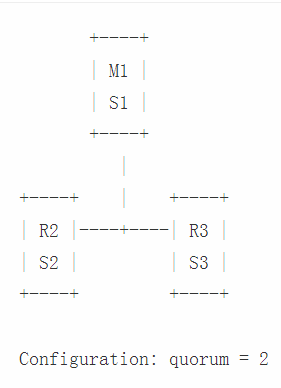
# Redis集群治理方案选型分析

## 官方HA方案Redis Sentinel

**哨兵主从模式集群，典型部署方案如下：**



至少需要三个节点来部署Sentinel集群，一个Master实例，一个或多个Slave实例。

针对Redis Sentinel的扩展，主要是两个方面：

* 根据系统对读性能的要求增加slave节点，但是单机master写瓶颈无法解决。
* 通过设置实例的maxmemory进行垂直扩容。

**所以Redis Sentinel主要是redis的高可用实现，适用于读操作远大于写操作，并且单机足够支撑应用所有的写操作的场景。**

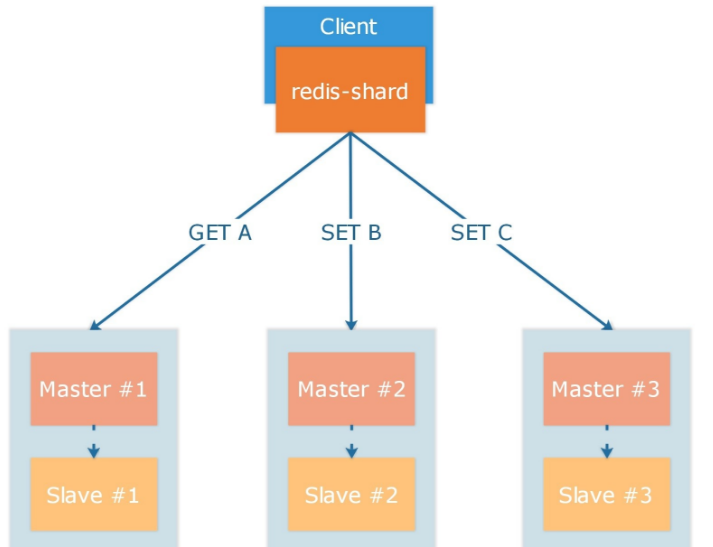
## Redis集群（高可用+分片）

### 客户端分片

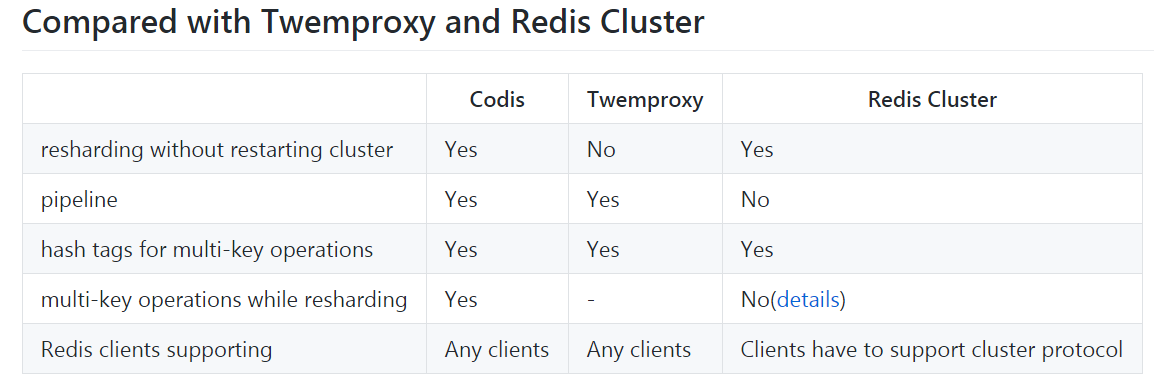
早期会选择客户端通过hash算法去访问相应的集群

优点：最简单的集群方案，没有中间件不需要经过代理，速度最快。

缺点：客户端代码维护困难，升级、扩容困难，需要停机迁移等。



### Proxy分片



好像阿里云也是通过代理的这种方式实现redis集群

#### Twitter开源的Twemproxy

Github地址：https://github.com/twitter/twemproxy

参考文章：https://www.wengbi.com/thread\_93142\_1.html

这里知乎在实际运用中提到了两种策略，一种当做存储（HA的要求会高一些），一种当做缓存（直接动态剔除节点，不做主从复制），也会使用自研的工具来实现平滑的扩容和单核模型的性能瓶颈问题。

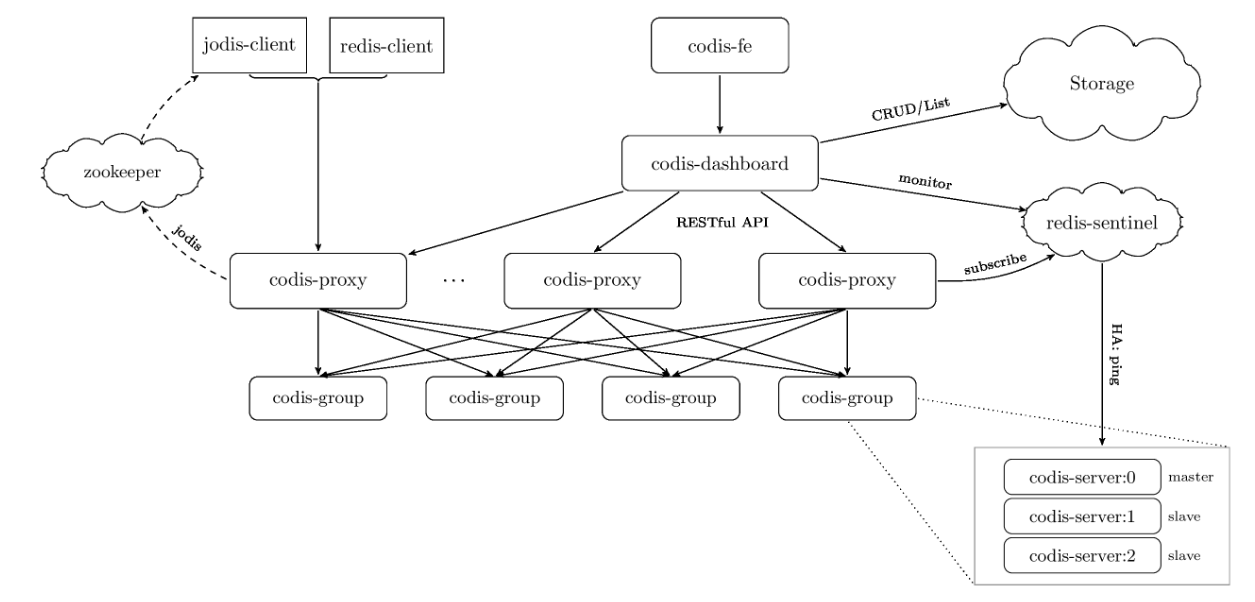
单核模型造成性能瓶颈；

传统扩容模式仅支持停机扩容；

无运维控制台

#### 豌豆荚开源的Codis

Github地址：https://github.com/CodisLabs/codis



优点：

* 生产验证丰富，可靠性较高；
* 自带控制台；
* 支持多键操作，不停机平滑扩容，新版本resharding支持异步数据迁移；
* 支持Pipeline

缺点：

* 通过代理降低性能；
* 一些redis操作不支持；
* 不负责主从复制，不支持跨机房架构
* 部署较复杂

### Redis Cluster

分片采用hash slot

优点：

* 原生redis，实现简单；
* 无中心设计，无代理性能高；
* 支持不停机扩容；
* 搜狐开源Cachecloud，基本符合SEE平台集成的要求，快速实现

缺点：

* 不支持pipeline、不支持多键操作，resharding时使用的是同步阻塞方式；
* 早先是不够成熟，生产验证较少，现在看好像美团的Squirrel和搜狐的cachecloud也都在用；
* 并不能保证强一致性，在特定情况下会丢失（主从异步复制间master宕掉、网络分区等）；

**这里应该也需要稍微了解一下公司目前redis使用的一些情况：读写要求，集群规模，业务一致性要求等等。**

## 多租户

* 小业务通过在key前缀增加业务标识，复用相同集群；
* 大业务使用独立集群，独立机器；
* 但是redis本身是有database的概念，默认是16个db中的0，db间相互隔离，客户端通过选择db来实现多租户。但是目前好像没找到针对db级别的动态扩容方案。