# Redis sharding方案

实现数据分片基本有三种方式

1. 客户端实现数据分片 （jedis API）
2. 服务端实现数据分片 (redis 官方cluster方案)
3. 通过代理服务器实现数据分片（twemproxy， codis）

## 客户端实现数据分片

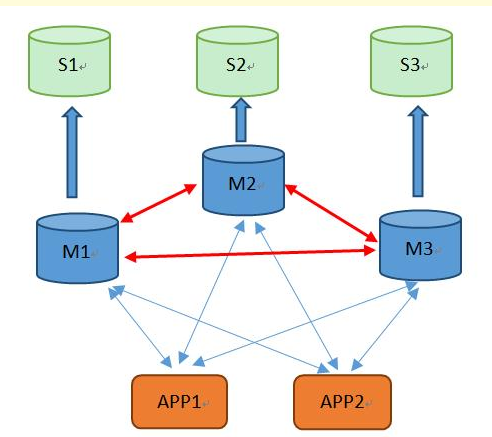
客户端数据分片与服务器端是强耦合，服务端增/减节点都必须修改客户端代码

有一种presharding的提法能够解决服务端扩容而不需要修改客户端代码，但还是解决不了强耦合的问题

若采用这种方式，最佳实践是提供统一的服务、统一访问接口，对开发者隐藏连接参数

## 服务器端实现分片

Redis cluster是一种服务器sharding技术，架构图如下



M1、M2、M3是三个主节点

S1、S2、S3是三个从节点，分别为M1、M2、M3的备份和故障切换节点

APP1、APP2是客户端应用

采用slot（槽）概念的概念来分片，一共分为16384个槽，对每个进入的key进行散列

使用的hash算法是CRC16后16384取模

Redis集群教程

<http://www.redis.cn/topics/cluster-tutorial.html>

Redis集群规范

<http://www.redis.cn/topics/cluster-spec.html>

数据扩容redis cluster也有实现 如下命令

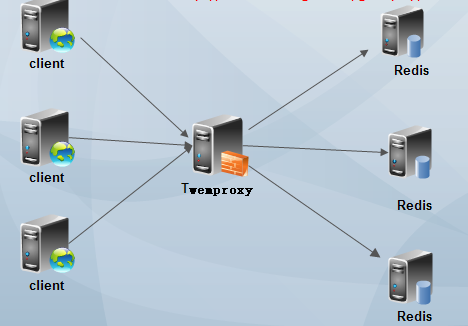
Redis-trib.rb add-node 192.168.1.1:6386 192.168.1.2:6386

特点：

* 配置复杂
* 通过连接任何一个主节点可以获读写所有节点上的数据
* 无法实现集群级别的主从，即一个集群是主，另一个是从

## 通过代理服务器实现数据分片

Twemproxy是一种代理分片机制，其架构图如下



当然上图可以看出twemproxy是单点，需要用keepalived做高可用方案

特点：

* 架构简单
* 对于客户端来说，redis集群是透明的，客户端简单，便于动态扩容
* Redis挂掉后能自动摘除，恢复后能自动恢复
* 新增redis node 需要重启twemproxy；并且数据不会自动reblance，需要人工单独写脚本
* 高性能，cpu密集型，正好与redis互补，可部署在redis机器上
* 会有约20%的性能损失
* 可设计成读写分离架构

读写分离架构图

