什么是Redis持久化？Redis有哪几种持久化方式？优缺点是什么？

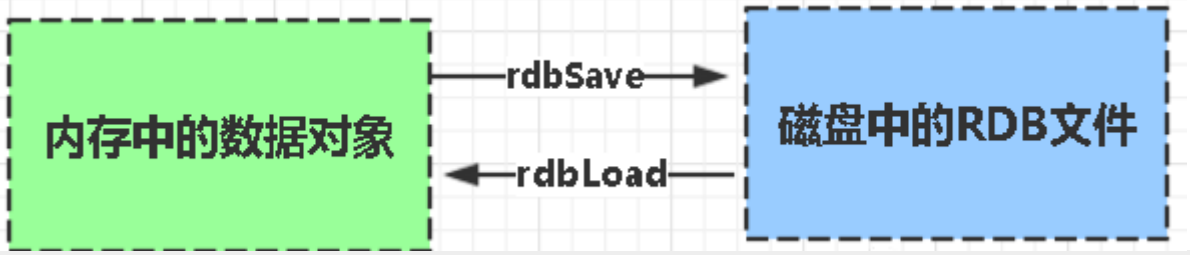
持久化就是把内存的数据写到磁盘中去，防止服务宕机了内存数据丢失。

Redis 提供了两种持久化方式:RDB（默认） 和AOF

RDB:

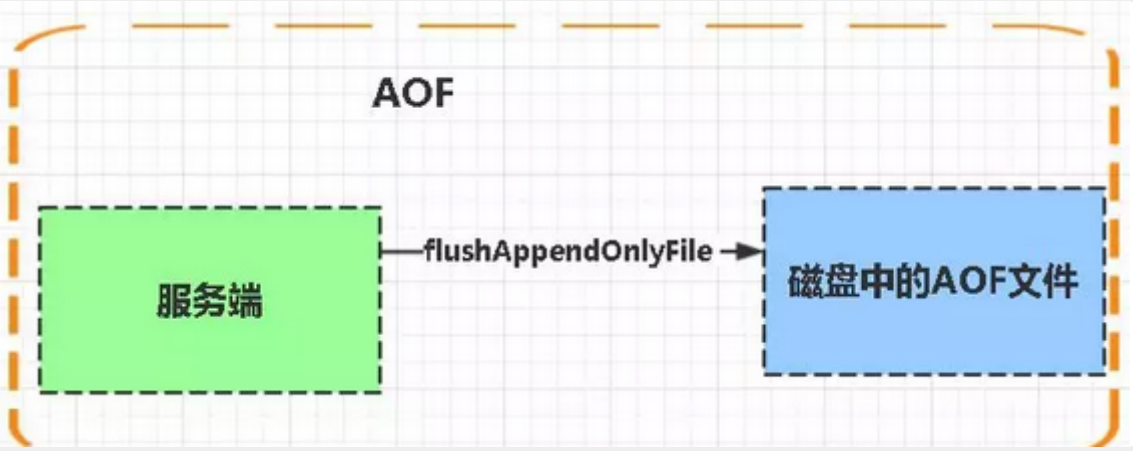
rdb是Redis DataBase缩写

功能核心函数rdbSave(生成RDB文件)和rdbLoad（从文件加载内存）两个函数



AOF:

Aof是Append-only file缩写



每当执行服务器(定时)任务或者函数时flushAppendOnlyFile 函数都会被调用， 这个函数执行以下两个工作

aof写入保存:

WRITE：根据条件，将 aof_buf 中的缓存写入到 AOF 文件

SAVE：根据条件，调用 fsync 或 fdatasync 函数，将 AOF 文件保存到磁盘中。

存储结构:

内容是redis通讯协议(RESP)格式的命令文本存储。

比较:

1、aof文件比rdb更新频率高，优先使用aof还原数据。

2、aof比rdb更安全也更大

3、rdb性能比aof好

4、如果两个都配了优先加载AOF

刚刚上面你有提到redis通讯协议(RESP)，能解释下什么是RESP？有什么特点？ （可以看到很多面试其实都是连环炮，面试官其实在等着你回答到这个点，如果你答上了对你的评

价就又加了一分）

RESP 是redis客户端和服务端之前使用的一种通讯协议；

RESP 的特点：实现简单、快速解析、可读性好

For Simple Strings the first byte of the reply is "+" 回复

For Errors the first byte of the reply is "-" 错误

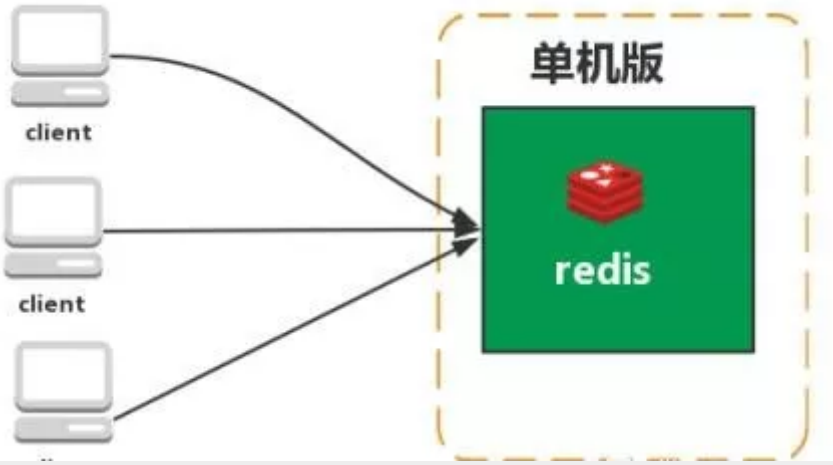
For Integers the first byte of the reply is ":" 整数

For Bulk Strings the first byte of the reply is "$" 字符串

For Arrays the first byte of the reply is "*" 数组

Redis 有哪些架构模式？讲讲各自的特点

单机版

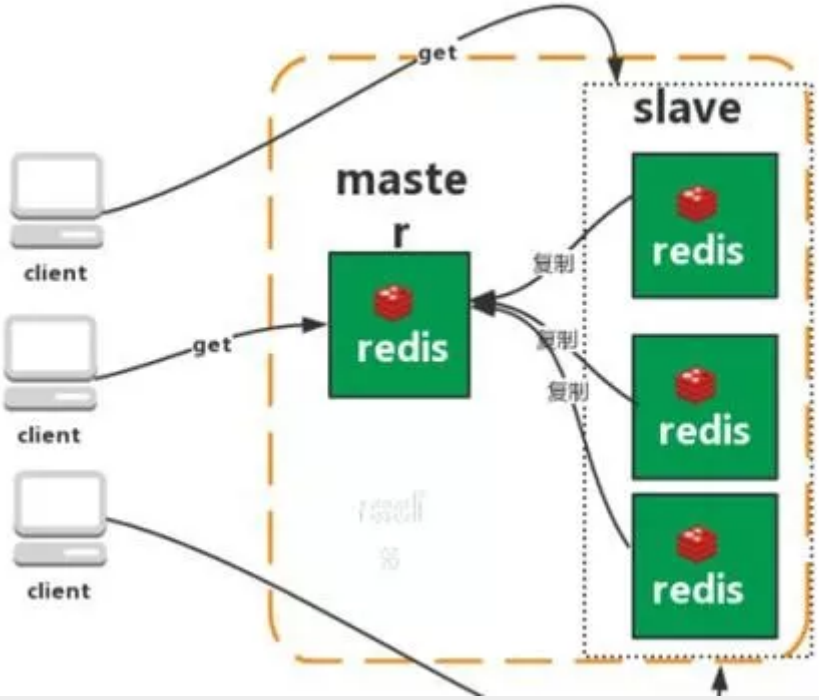


特点：简单

问题：

1、内存容量有限 2、处理能力有限 3、无法高可用。

主从复制



Redis 的复制（replication）功能允许用户根据一个 Redis 服务器来创建任意多个该服务器的复制品，其中被复制的服务器为主服务器（master），而通过复制创建出来的服务器复制品则为从服务器（slave）。 只要主从服务器之间的网络连接正常，主从服务器两者会具有相同的数据，主服务器就会一直将发生在自己身上的数据更新同步 给从服务器，从而一直保证主从服务器的数据相同。

特点：

1、master/slave 角色

2、master/slave 数据相同

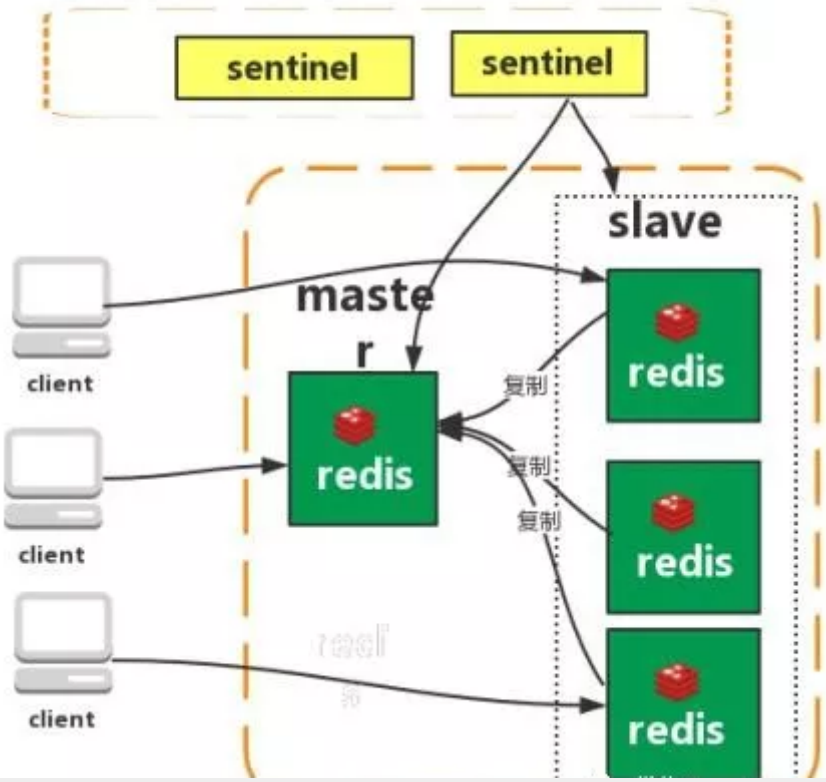
3、降低 master 读压力在转交从库

问题：

无法保证高可用

没有解决 master 写的压力

哨兵



Redis sentinel 是一个分布式系统中监控 redis 主从服务器，并在主服务器下线时自动进行故障转移。其中三个特性：

监控（Monitoring）： Sentinel 会不断地检查你的主服务器和从服务器是否运作正常。

提醒（Notification）： 当被监控的某个 Redis 服务器出现问题时， Sentinel 可以通过 API 向管理员或者其他应用程序发送通知。

自动故障迁移（Automatic failover）： 当一个主服务器不能正常工作时， Sentinel 会开始一次自动故障迁移操作。

特点：

1、保证高可用

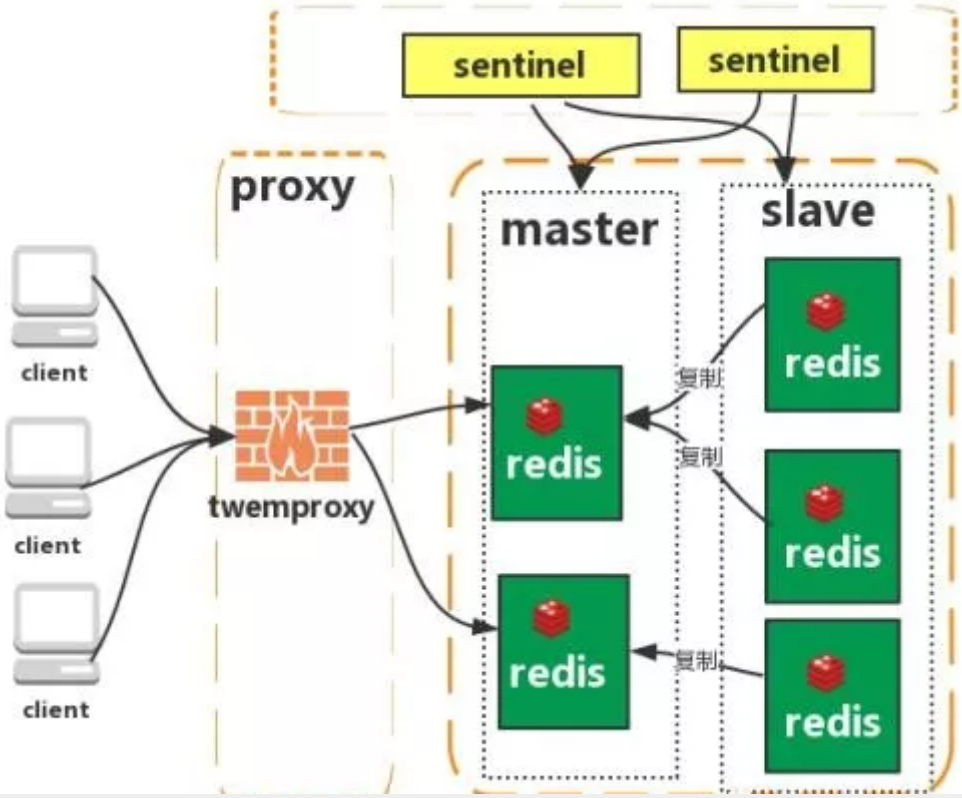
2、监控各个节点

3、自动故障迁移

缺点：主从模式，切换需要时间丢数据

没有解决 master 写的压力

集群（proxy 型）：



Twemproxy 是一个 Twitter 开源的一个 redis 和 memcache 快速/轻量级代理服务器；Twemproxy 是一个快速的单线程代理程序，支持 Memcached ASCII 协议和 redis 协议。

特点：1、多种 hash 算法：MD5、CRC16、CRC32、CRC32a、hsieh、murmur、Jenkins

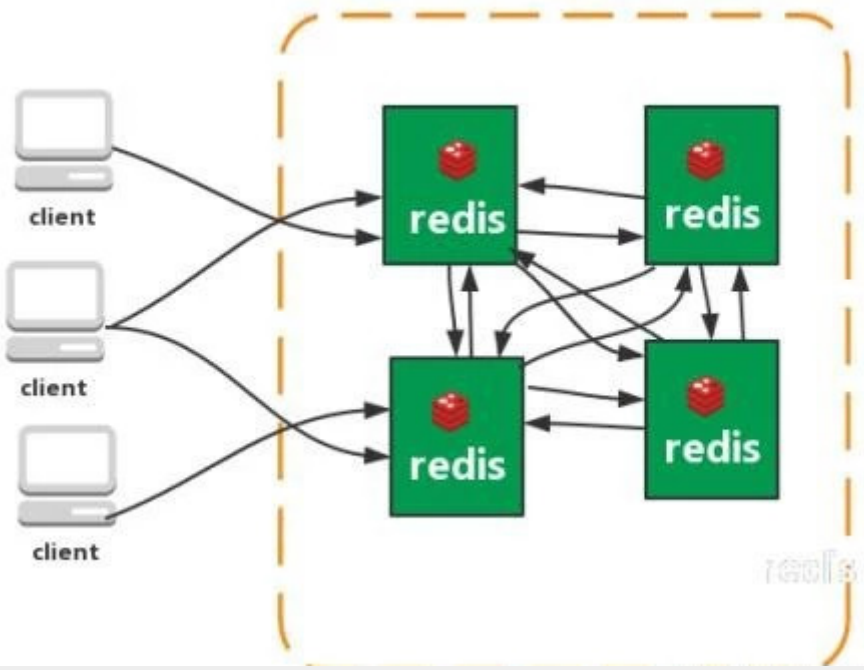
2、支持失败节点自动删除

3、后端 Sharding 分片逻辑对业务透明，业务方的读写方式和操作单个 Redis 一致

缺点：增加了新的 proxy，需要维护其高可用。

failover 逻辑需要自己实现，其本身不能支持故障的自动转移可扩展性差，进行扩缩容都需要手动干预

集群（直连型）：



从redis 3.0之后版本支持redis-cluster集群，Redis-Cluster采用无中心结构，每个节点保存数据和整个集群状态,每个节点都和其他所有节点连接。

特点：

1、无中心架构（不存在哪个节点影响性能瓶颈），少了 proxy 层。

2、数据按照 slot 存储分布在多个节点，节点间数据共享，可动态调整数据分布。

3、可扩展性，可线性扩展到 1000 个节点，节点可动态添加或删除。

4、高可用性，部分节点不可用时，集群仍可用。通过增加 Slave 做备份数据副本

5、实现故障自动 failover，节点之间通过 gossip 协议交换状态信息，用投票机制完成 Slave到 Master 的角色提升。

缺点：

1、资源隔离性较差，容易出现相互影响的情况。

2、数据通过异步复制,不保证数据的强一致性

什么是一致性哈希算法？什么是哈希槽?

这两个问题篇幅过长 网上找了两个解锁的不错的文章

https://www.cnblogs.com/lpfuture/p/5796398.html

https://blog.csdn.net/z15732621582/article/details/79121213

08

使用过Redis分布式锁么，它是怎么实现的?

先拿setnx来争抢锁，抢到之后，再用expire给锁加一个过期时间防止锁忘记了释放。

如果在setnx之后执行expire之前进程意外crash或者要重启维护了，那会怎么样?

set指令有非常复杂的参数，这个应该是可以同时把setnx和expire合成一条指令来用的!

09

使用过Redis做异步队列么，你是怎么用的？有什么缺点?

一般使用list结构作为队列，rpush生产消息，lpop消费消息。当lpop没有消息的时候，要适当sleep一会再重试。

缺点:

在消费者下线的情况下，生产的消息会丢失，得使用专业的消息队列如rabbitmq等。

能不能生产一次消费多次呢?

使用pub/sub主题订阅者模式，可以实现1:N的消息队列。

10

什么是缓存穿透？如何避免？什么是缓存雪崩？何如避免?

缓存穿透

一般的缓存系统，都是按照key去缓存查询，如果不存在对应的value，就应该去后端系统查找（比如DB）。一些恶意的请求会故意查询不存在的key,请求量很大，就会对后端系统造成很大的压力。这就叫做缓存穿透。

如何避免?

1：对查询结果为空的情况也进行缓存，缓存时间设置短一点，或者该key对应的数据insert了之后清理缓存。

2：对一定不存在的key进行过滤。可以把所有的可能存在的key放到一个大的Bitmap中，查询时通过该bitmap过滤。

缓存雪崩

当缓存服务器重启或者大量缓存集中在某一个时间段失效，这样在失效的时候，会给后端系统带来很大压力。导致系统崩溃。

如何避免?

1：在缓存失效后，通过加锁或者队列来控制读数据库写缓存的线程数量。比如对某个key只允许一个线程查询数据和写缓存，其他线程等待。

2：做二级缓存，A1为原始缓存，A2为拷贝缓存，A1失效时，可以访问A2，A1缓存失效时间设置为短期，A2设置为长期

3：不同的key，设置不同的过期时间，让缓存失效的时间点尽量均匀。

11.分布式的环境下， MySQL和Redis如何保持数据的一致性？

，增删改都是操作mysql，对于读是保存到redis，这样就涉及到数据同步操作，同步操作分为两大块，我们的叫法是，一个是全量(将全部数据一次写入到redis，时间几小时不等)，一个是增量（实时更新）。这里说的是增量，主要问题是即时性，因为增删改都是直接操作mysql变更都在MySQL（这里高并发的问题是用分库分表加外层的负载均衡） 所以我们的方

向是读取binlog然后分析 ，利用消息推送到某服务器A，再进行分析，然后更新各台redis，消息推送工具用的是rabbitMQ，可设定某表的变更推送(分三类update insert delate 包含变更前后的数据)，这里有个问题是：mysql数据操作太频繁产生的推送可能会很多，所以分析处理脚本处理速度一定要跟得上（我用Python写，前期多线程（坑），后来改成多进程）

MySQL binlog增量订阅消费+消息队列+处理并把数据更新到redis