Redis高可用实现原理

由 程利（chengli）-智能平台部创建于九月 02, 2019

- 1. 一主多从（Master-Slave）集群
  - 1.1 集群架构
  - 1.2 Sentinel介绍
  - 1.3 主库高可用
  - 1.4 两地三中心的主库高可用
  - 1.5 从库高可用
- 2. Cluster分片集群
- 3. 总结

目前，云服务主要提供两种模式的Redis集群：一主多从（Master-Slave）集群、Cluster分片集群。其中，一主多从（Master-Slave）集群内存大小20G以内，适用于数据量不大、流量较高、读多写少的场景，当读流量非常高时，可以通过扩容从库分担读压力。Cluster分片集群由多个一主一从的分片组成，每个分片内存大小20G以内，集群总内存等于各个分片内存相加，适用于数据量较大、流量较高、写多读少（或读写相当）的场景，当内存不够用时，可以扩容分片数量。同集群的所有实例分布在不同机器上。

常见的故障场景包含单机故障、机房故障。其中，单机故障包含单个实例进程down、单台机器down、单个实例被慢查询阻塞等场景。机房故障包含单个机房掉电、机房内部交换机故障、机房间专线故障等场景。

下面将分别针对两种故障场景，介绍两种模式的Redis集群是如何实现高可用的。

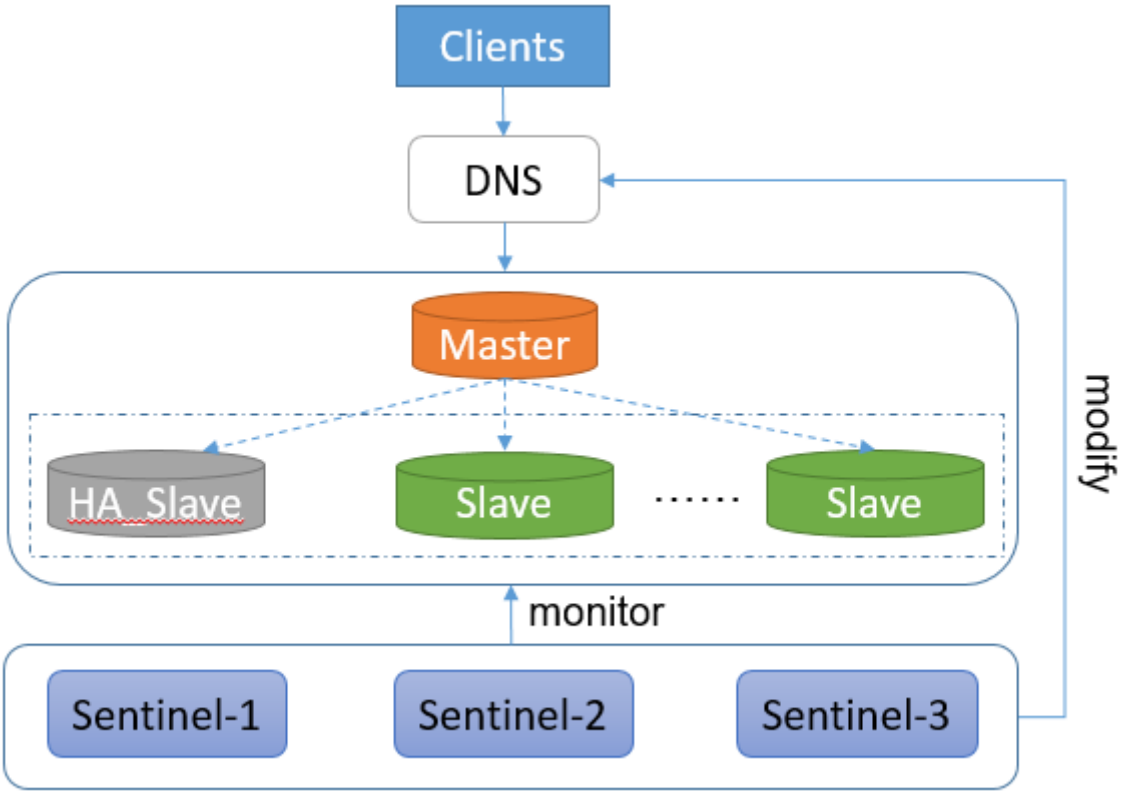
1. 一主多从（Master-Slave）集群

1.1 集群架构

通常一主多从（Master-Slave）集群包含的实例有1个Master、1个HA_Slave（与Master同机房，本质也是Slave，作为Master的灾备）、若干个Slave（每机房不少于2个）。用户在申请新集群时，若只选择了主库，默认会提供一个同机房的HA_Slave。一般Master上绑定写域名、HA_Slave仅作为Master的备库不绑定域名、Slave上绑定只读域名提供使用。

> ⚠ **申请资源注意事项**
> 用户在申请Redis资源时，需要合理选择主从库机房，考虑是否需要跨机房灾备的情况。若一个Redis集群只有一个机房的实例，则在该机房故障的场景下，主库不能发生跨机房的切换；若Redis客户端及上层服务全部都和Redis主库在一个机房，Redis集群却选择了多个机房，则当主库所在机房故障时，Redis主库会发生跨机房切换，可能会导致客户端无法访问主库。

Master-Slave集群架构如下：



云服务提供的一主多从（Master-Slave）结构的Redis集群，主要使用Sentinel+DNS的方式来保障高可用。Sentinel是Redis官方提供的原生高可用解决方案，通过独立运行的Sentinel进程，每秒对集群中的Redis实例进行1次ping操作，可以实时监控Redis实例进程的存活状态。DNS即域名，将域名和Redis实例所在机器的IP进行绑定，客户端即可通过域名连接到对应的Redis实例。

对于每一个Master-Slave集群，为了避免Sentinel监控单点问题，云服务会使用3个Sentinel节点组成一个分布式的Sentinel集群来监控Redis集群所有实例，通过预先设定的配置，监控故障的发生，并通过调整DNS和实例IP的绑定关系，实现自动故障转移。

1.2 Sentinel介绍

Sentinel的主要功能有：

- 实时监控Redis实例是否按照预期良好地运行
- 如果发现某个Redis实例运行出现状况，能够通知另外一个进程（例如它的客户端）
- 能够进行自动切换。当一个Master实例不可用时，能够选举出Master多个Slave（如果有超过一个Slave的话）中的一个成为新的Master，并切换其它Slave实例为新Master实例的从库。若老的Master实例后期恢复，也会成为新Master的从库

进一步了解Sentinel可参考官方文档。

为了避免和线上Redis集群产生干扰，云服务将每一个Sentinel节点部署在独立的服务器上。由于每个业务需求不同，有的Redis集群全部实例分布在一个机房，有的Redis集群实例会分布在多个机房，因此用于监控每个Redis集群的3个Sentinel节点所在机房分布不能完全一致。关于Sentinel节点的选择，我们制定了一些选取规则：

- 若只申请1个机房的资源，即所有主从实例均在一个机房
  - 若是小机房（考虑跨机房网络稳定性不高），则在主库机房选择3个Sentinel
  - 若是大机房（考虑环路保护，跨机房网络稳定性较高），则主库机房1个Sentinel，其他大机房任选2个机房各1个Sentinel
    ps：当前大机房主要指在北京、上海、济阳联通、武汉等地区的机房
- 若申请2个或以上机房的资源
  - 当申请了2个机房资源时，则主库机房1个Sentinel，从库机房1个Sentinel，其他大机房任选1个Sentinel
  - 当申请了3个及以上机房资源时
    - 若申请了两地三中心的资源配置，则从3个AZ各选择1个Sentinel
    - 其他情况，则主库机房1个Sentinel，再从剩下的从库机房随机选择2个机房各1个Sentinel

Sentinel如何实现Redis的自动故障转移，是由一些预先设定的配置决定的。云服务的Sentinel配置如下：

```
Sentinel monitor myMaster host port 2  #有2个或2个以上Sentinel认为Master实例down了，即触
Sentinel down-after-milliseconds myMaster 40000  #Sentinel每秒对所有实例进行ping心跳检测
```

```
Sentinel failover-timeout myMaster 900000  #Sentinel触发主从failover切换的间隔时间，单位
Sentinel parallel-syncs myMaster 2  #指定最多2个slave同时对新的Master进行同步。数字越小，完
Sentinel notification-script myMaster $pwd/dns_unbind  #Sentinel监控到的告警事件触发执行
Sentinel client-reconfig-script myMaster $pwd/reconfig_dns  #当发生主从failover切换后，
```

1.3 主库高可用

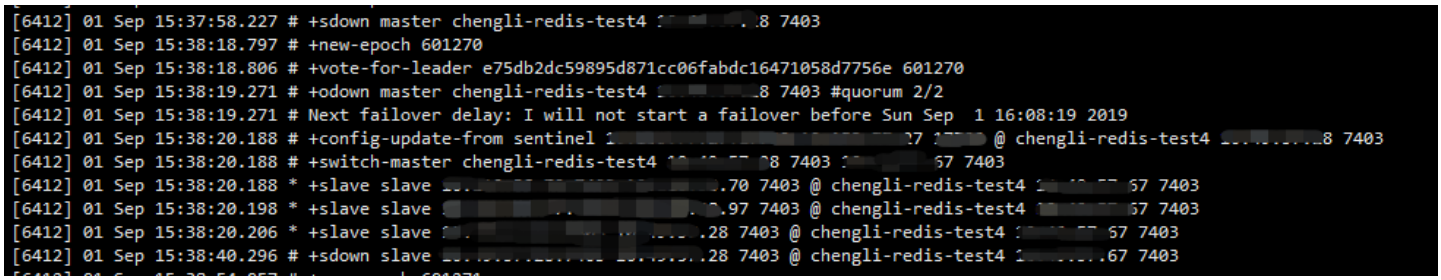
云服务在部署Redis服务时，会通过设置所有从库实例的priority参数（值越小优先级越高，默认是100）来确定主库故障时从库被提升为新主库的优先级。一般设置HA_Slave优先级为10，具有最高优先级，其他Slave的优先级为100。

a.当主库所在机器故障或主库被阻塞时，HA_Slave通常是可用的，Sentinel会优先提升HA_Slave为新的主库。整个切换流程如下：

- Sentinel-1对Master执行ping操作，发现回复时间超设定值20s。Sentinel-1标记Master为主观下线状态
- Sentinel-2对Master执行ping操作，发现回复时间超过设定值40s。Sentinel-2标记Master为主观下线状态
- 超过2个Sentinel认为Master下线，Master被标记为客观下线
- 投票选举出1个Sentinel，授权去真正执行主从failover切换
- failover切换时选择优先级较高的HA_Slave为新的Master（若HA_Slave不可用，则选择其他合适的Slave）。
- 主从failover切换后，执行切换的Sentinel将Master的最新配置通过广播形式通知其它Sentinel，其它的Sentinel则更新对应Master的配置（确保Master的配置版本信息唯一）。同时触发reconfig_dns脚本，将新的Master信息通知给相关客户端，reconfig_dns脚本将主库域名绑定到新的Master IP上，并从老Master IP上解绑
- Sentinel每次指定2个slave slaveof到新的Master上，等待sync数据，直到所有从库都指向新的Master同步数据

部分切换日志如下：

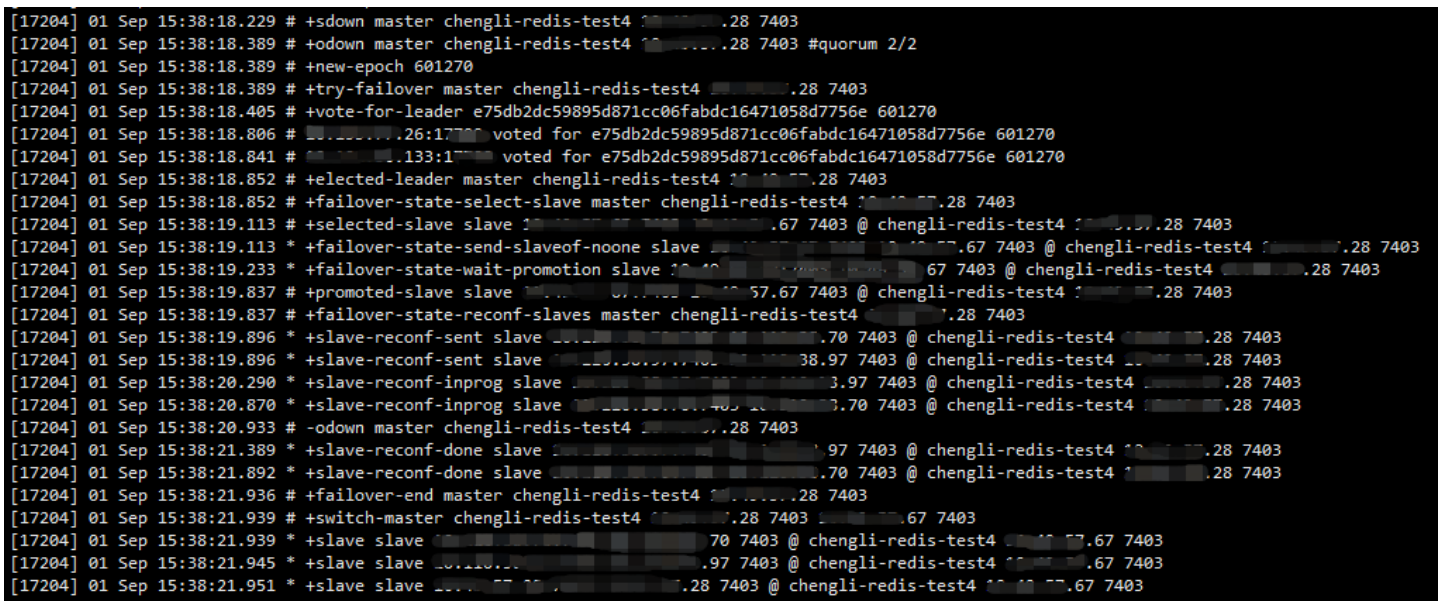
Sentinel-1发现Master ping超过20s未回复，标记为主观下线+sdown



Sentinel-2发现Master ping超过40s未回复，标记为主观下线+sdown

超过2个Sentinel认为Master下线，Master被标记为客观下线+odown

后续选举Sentinel，并触发主从切换



触发reconfig_dns脚本，进行域名绑定解绑过程

```
[2019-09-01 15:38:20.272310] --- : Starting domain rebind @ chengli-redis-test4
[2019-09-01 15:38:20.285281] --- : parameters: chengli-redis-test4 leader start  .28 7403  .67 7403
[2019-09-01 15:38:20.287258] --- : Binding domain curl http://  ./add/zjy.chengliredistest4.qiyi.redis/  .67/
[2019-09-01 15:38:21.228457] --- : +domain {"ip": ".67", "request_ip": "", "name": "zjy.chengliredistest4.qiyi.redis", "result": true}
[2019-09-01 15:38:21.234566] --- : domain zjy.chengliredistest4.qiyi.redis successfully bound to  .67 @ chengli-redis-test4
[2019-09-01 15:38:21.237859] --- : Unbinding domain curl http://  /del/zjy.chengliredistest4.qiyi.redis/  .28/
[2019-09-01 15:38:21.477965] --- : -domain {"ip": ".28", "request_ip": "", "name": "zjy.chengliredistest4.qiyi.redis", "result": true}
[2019-09-01 15:38:21.484929] --- : domain zjy.chengliredistest4.qiyi.redis successfully unbound from  .28 @ chengli-redis-test4
```

b.当主库所在机房故障时，同机房的HA_Slave和Slave将不可用，Sentinel会选择提升其他机房的Slave为新的主库，切换流程同上。

> ⚠ **注意**
> 云服务当前使用的Redis版本（Redis 3.2 及以下），若集群发生了主从failover切换，所有Slave在slaveof到新的Master上后，均需要做一次全量数据同步。在同步数据的过程中有一段时间Slave会不支持访问，这个时间的长短和数据量大小以及网络延迟有关。

1.4 两地三中心的主库高可用

目前两地三中心主要有北京4个AZ、武汉1个AZ，满足两地三中心高可用的基本资源配置为北京2个AZ、武汉1个AZ均有Redis实例。

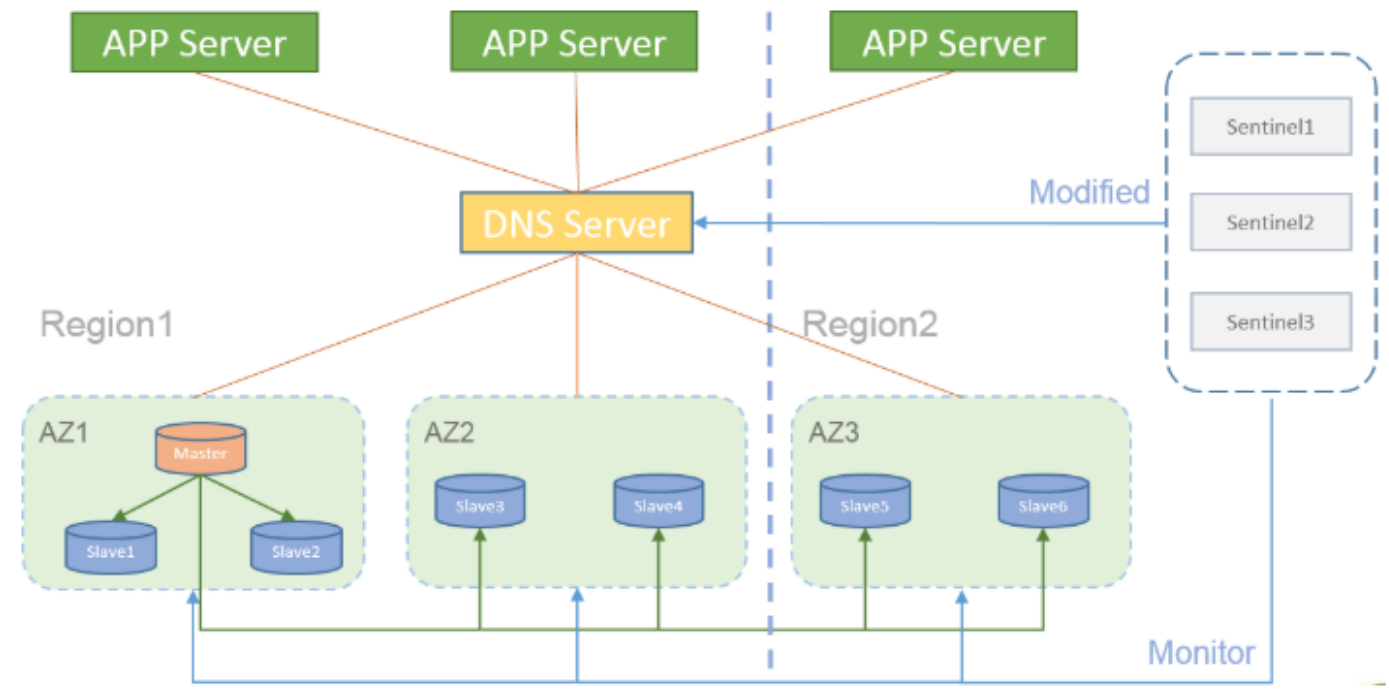
AZ（Availability Zones）即为可用区，具有以下特点：

- 是1个大型的数据中心组织，或者由地理位置相近的几个数据中心组成
- 拥有独立的包括电力和网络在内的基础设施
- 同一个AZ内的数据中心间，TOR至TOR时延≤1ms

若业务方在多个机房均有部署服务，根据AZ内延迟较小的特点可得知，每个AZ内申请一个数据中心（数据中心即为机房）的资源即可。

两地三中心的Redis集群部署架构（假设AZ1和AZ2为同一Region）：

- AZ1部署Master，同AZ1部署1个或以上从库，设置其中1个从库为HA_Slave（优先级10），其他从库为普通Slave（优先级100）
- AZ2部署2个或以上从库，若AZ2和AZ1属于同一地区（Region，同城），则设置其中一个从库为HA_Slave2（优先级50），其他从库为普通Slave（优先级100）
- AZ3部署2个或以上从库，设置所有从库为普通Slave（优先级100）
- 3个AZ各部署1个Sentinel



当Master所在机器故障或主库被阻塞时，Sentinel节点之间、Sentinel和集群其他Redis实例之间无网络连通性问题，3个Sentinel均会判定Master down，并优先提升AZ1的HA_Slave为新的主库。

当Master所在的AZ1机房故障时，同AZ1的Sentinel和其他Sentinel之间会出现网络连通性问题导致判断失败，但其他两个AZ的Sentinel均会判定Master down，且判定HA_Slave down，会优先提升同地区（同城）的AZ2的HA_Slave2为新的主库，尽量避免跨地区切换，减少访问延迟。

1.5 从库高可用

为了避免跨机房延迟，每个只读域名下的从库都是同一个机房的，故障时不会跨机房绑定其他从库。1个只读域名至少绑定2个从库实例，才能保障高可用，因此，申请在新机房扩容从库时，需要在一个机房至少申请2个从库。

当从库所在机器故障或从库被阻塞时，具体的切换流程如下：

- Sentinel通过ping心跳检测，发现从库实例回复时间超设定值，认为从库实例down，event触发dns_unbind 脚本
- dns_unbind 脚本内部进行判断：当3个Sentinel都认为从库实例down后，会先检查从库所在域名下有几个实例
  - 若只有1个实例，则不进行操作
  - 若有2个或以上实例，则触发域名解绑操作
- 当Sentinel能再次ping通故障恢复的从库时，会触发另一个恢复脚本，先等待主从同步恢复正常，再将只读域名绑定至刚恢复的从库上

当从库所在机房故障时，若有同机房的Sentinel监控，则由于同机房的Sentinel与其他机房的Sentinel连通性出现问题，将无法满足3个Sentinel均认为该从库down的条件，不会触发只读域名解绑操作；若无同机房的Sentinel监控，则Sentinel会将先判断为down的从库实例进行域名解绑，保留最后1个实例不进行解绑操作。

部分切换日志如下：

```
[2019-09-01 15:39:20.330358] --- : +sdown .28:7403 @ chengli-redis-test4
[2019-09-01 15:39:20.333011] --- : event description: slave .28 7403 @ chengli-redis-test4 .67 7403
[2019-09-01 15:39:20.579511] --- : 28:7403 odown quorum 3 @ chengli-redis-test4
[2019-09-01 15:39:20.581758] --- : 28:7403 is really down @ chengli-redis-test4
[2019-09-01 15:39:20.631669] --- : start domain unbind zjy.chengliredistest4.r.qiyi.redis .28 @ chengli-redis-test4
[2019-09-01 15:39:21.050323] --- : -domain {"ip": ".28", "request_ip": "", "name": "zjy.chengliredistest4.r.qiyi.redis", "result": true}
[2019-09-01 15:39:21.162669] --- : 28:7403 : zjy.chengliredistest4.r.qiyi.redis was recorded in Redis @ chengli-redis-test4
[2019-09-01 15:39:21.164587] --- : domain zjy.chengliredistest4.r.qiyi.redis was successfully unbound from .28 @ chengli-redis-test4
```

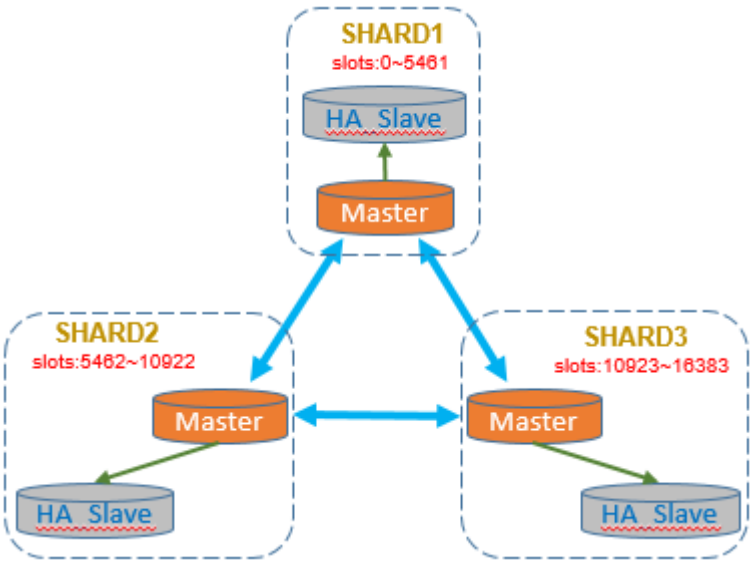
2. Cluster分片集群

云服务采用的是官方原生的Redis Cluster集群方案，Cluster分片集群是一种去中心化架构的集群。每个集群有16384个虚拟的槽（slot），集群里的每个key会根据hash算法（HASH_SLOT = CRC16(key) mod 16384）映射到一个具体的槽上，这些槽均匀的分布在集群的多个分片上，这样每个分片就负责管理一定数量槽上的数据。

一个Cluster分片集群至少由3个分片组成，每个分片一主一从（一Master一HA_slave）。客户端配置Master IP（理论上配置1个Master IP即可正常连接集群，为保障HA，建议配置所有Master IP）连接集群，Master承担读写流量，HA_slave仅作为Master的备库不提供访问。目前云服务提供的Cluster分片集群最小规格为总容量30G，分为3个分片，每个分片10G。随着业务发展，如果集群容量不足，可以申请扩容单个分片最大到20G，也可以申请扩容集群的分片数（扩容分片数需要进行数据rebalance）。当前线上的分片集群数量已有上百个，使用稳定。

注意：一个Cluster分片集群的所有实例均在一个机房，不支持跨机房部署，因此仅支持机器故障自动切换。

Cluster分片集群架构如下：



当1个分片的Master实例down或所在机器故障时，集群中其他超过半数的Master认为该分片Master down，则该分片的从库被自动提升为新的主库。新主库上的slot被激活可访问。

当1个分片的HA_Slave实例down或所在机器故障时，对客户端访问无感知。HA_Slave实例恢复后会自动作为该分片Master的从库同步数据。

3. 总结

一主多从（Master-Slave）集群支持跨机房部署，需要两地三中心建设或跨机房灾备的业务建议选择这种。缺点是单集群数据量大小限制20G以内，有更多存储需求的，可能需要在客户端侧进行分片，分片数量需要预先设定好。如果使用Java语言，客户端分片推荐使用服务云开发的SmartJedis客户端。

Cluster分片集群优点是扩容方便（增加分片，rebalance数据）、多个分片Master实例可以支持更高写入流量。缺点是不支持跨机房部署，如果需要跨机房灾备，目前需要业务方多写数据。

无标签