

目录 Contents

- ◆ 集群简介
- ◆ Redis集群结构设计
- ◆ cluster集群结构搭建

Redis 集群主要是为了应对并发写的压力

集群

现状问题

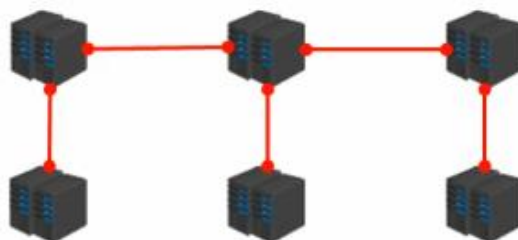


业务发展过程中遇到的峰值瓶颈

- redis提供的服务OPS可以达到10万/秒，当前业务OPS已经达到20万/秒
- 内存单机容量达到256G，当前业务需求内存容量1T
- 使用集群的方式可以快速解决上述问题

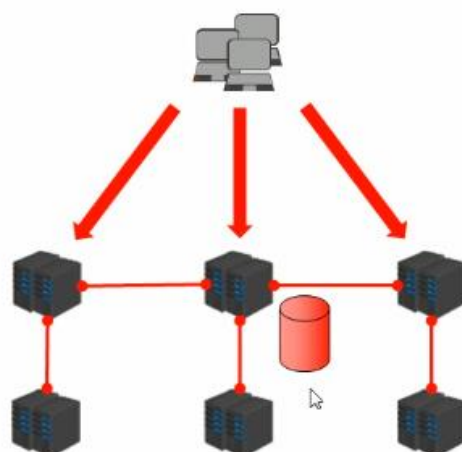
集群架构

- 集群就是使用网络将若干台计算机联通起来，并提供统一的管理方式，使其对外呈现单机的服务效果



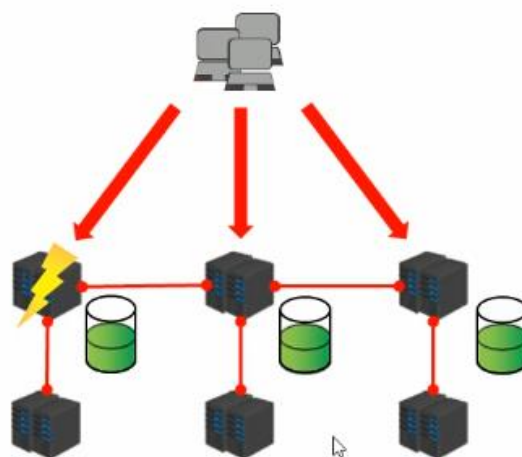
集群作用

- 分散单台服务器的访问压力，实现负载均衡



集群作用

- 分散单台服务器的访问压力，实现负载均衡
- 分散单台服务器的存储压力，实现可扩展性
- 降低单台服务器宕机带来的业务灾难



目录 Contents

- ◆ 集群简介
- ◆ Redis集群结构设计
- ◆ cluster集群结构搭建

Redis集群结构设计



数据存储设计



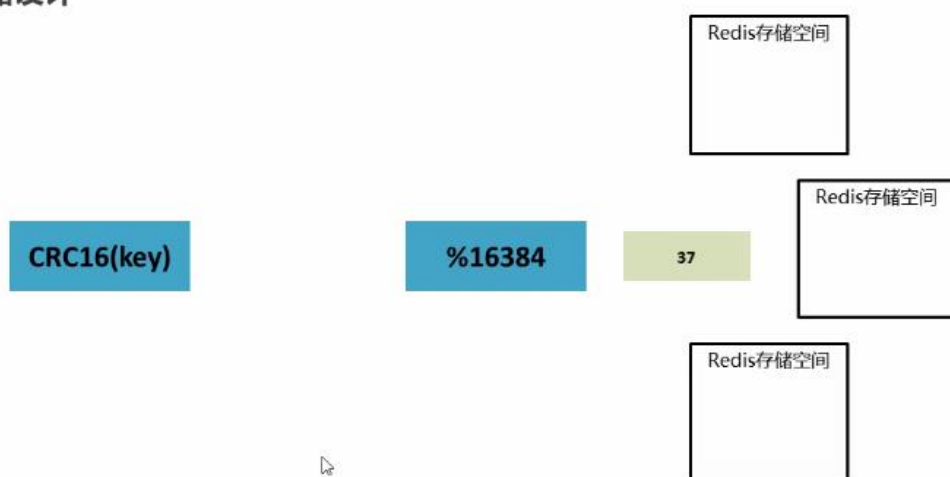
数据存储设计



数据存储设计

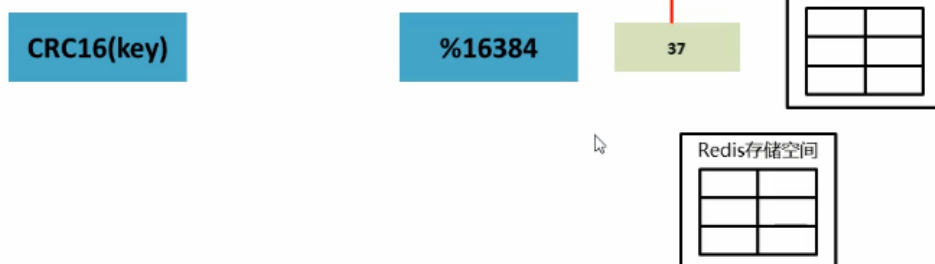


数据存储设计



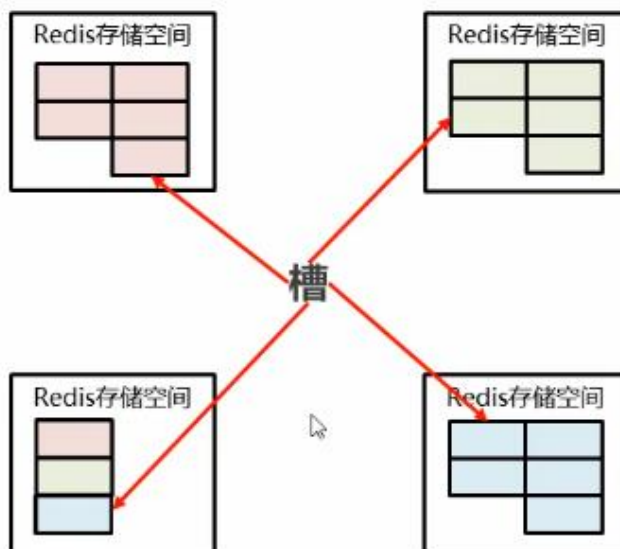
数据存储设计

- 通过算法设计，计算出key应该保存的位置
- 将所有的存储空间计划切割成16384份，每台主机保存一部分
每份代表的是一个存储空间，不是一个key的保存空间
- 将key按照计算出的结果放到对应的存储空间



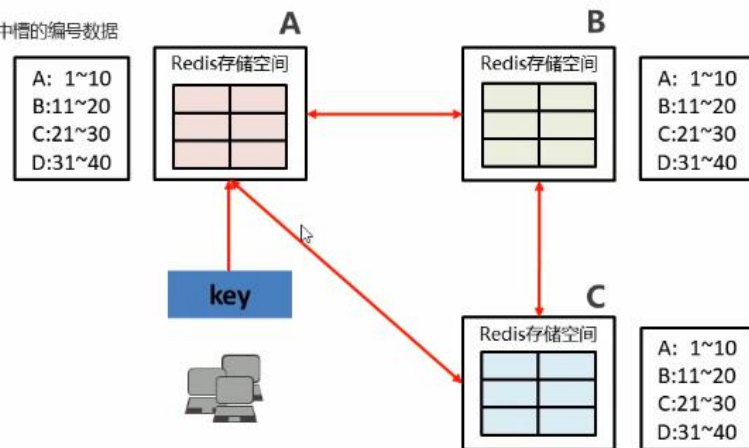
数据存储设计

- 增强可扩展性



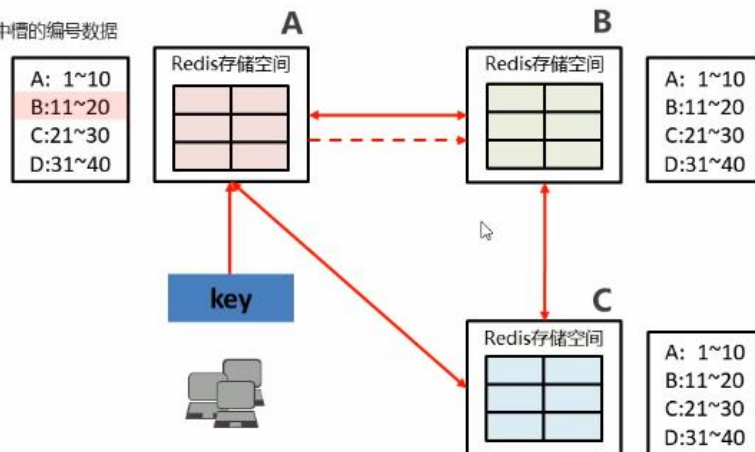
集群内部通讯设计

- 各个数据库相互通信，保存各个库中槽的编号数据
- 一次命中，直接返回



集群内部通讯设计

- 各个数据库相互通信，保存各个库中槽的编号数据
- 一次命中，直接返回
- 一次未命中，告知具体位置



要将结点连接需要用到 `redis-trib.rb`。注意：要是用这个命令需要先下载 Ruby 和 gem

```
[root@localhost src]# ll | grep redis-
-rwxr-xr-x. 1 root root 2451856 Oct 23 18:31 redis-benchmark
-rw-rw-r--. 1 root root 29558 Jul 14 2017 redis-benchmark.c
-rw-r--r--. 1 root root 108992 Oct 23 18:31 redis-benchmark.o
-rwxr-xr-x. 1 root root 5739680 Oct 23 18:31 redis-check-aof
-rw-rw-r--. 1 root root 7142 Jul 14 2017 redis-check-aof.c
-rw-r--r--. 1 root root 28632 Oct 23 18:31 redis-check-aof.o
-rwxr-xr-x. 1 root root 5739680 Oct 23 18:31 redis-check-rdb
-rw-rw-r--. 1 root root 13840 Jul 14 2017 redis-check-rdb.c
-rw-r--r--. 1 root root 62328 Oct 23 18:31 redis-check-rdb.o
-rwxr-xr-x. 1 root root 2606120 Oct 23 18:31 redis-cli
-rw-rw-r--. 1 root root 92355 Jul 14 2017 redis-cli.c
-rw-r--r--. 1 root root 371432 Oct 23 18:31 redis-cli.o
-rwxr-xr-x. 1 root root 5739680 Oct 23 18:31 redis-sentinel
-rwxr-xr-x. 1 root root 5739680 Oct 23 18:31 redis-server
-rwxrwxr-x. 1 root root 60843 Jul 14 2017 redis-trib.rb
[root@localhost src]#
-rwxr-xr-x. 1 root root 5739680 Oct 23 18:31 redis-server
-rwxrwxr-x. 1 root root 60843 Jul 14 2017 redis-trib.rb
[root@localhost src]# ruby -v
ruby 2.3.1p112 (2016-04-26 revision 54768) [x86_64-linux]
[root@localhost src]# gem -v
2.7.7
[root@localhost src]#
```

```
[root@localhost src]# redis-trib.rb
bash: redis-trib.rb: command not found...
[root@localhost src]# ./redis-trib.rb create --replicas 1 127.0.0.1:6379 127.0.0.1:6380 127.0.0.1:6381 127.0.0.1:6382 127.0.0.1:6383 127.0.0.1:6384
>>> Creating cluster
>>> Performing hash slots allocation on 6 nodes...
Using 3 masters:
127.0.0.1:6379
127.0.0.1:6380
127.0.0.1:6381
Adding replica 127.0.0.1:6382 to 127.0.0.1:6379
Adding replica 127.0.0.1:6383 to 127.0.0.1:6380
Adding replica 127.0.0.1:6384 to 127.0.0.1:6381
M: 8bd87b4d5fc269f286d95c55dfa676b74720a9bd 127.0.0.1:6379
slots:0-5460 (5461 slots) master
M: 91f0f76da398e38b38ab0df4e49248dee2756794 127.0.0.1:6380
slots:5461-10922 (5462 slots) master
M: 446b6f0abdb3743ca349cf3a628e4df5309cecf 127.0.0.1:6381
slots:10923-16383 (5461 slots) master
S: 715ea09a37c333e6d3a3acbf97f95e3e951de6e7 127.0.0.1:6382
replicas: 8bd87b4d5fc269f286d95c55dfa676b74720a9bd
S: 948abfc95e0c37b2726bd0450d3f2e90b78ca089 127.0.0.1:6383
replicas: 91f0f76da398e38b38ab0df4e49248dee2756794
S: 2814234930c1df302dd1eaa6d94578f935a4287 127.0.0.1:6384
replicas: 446b6f0abdb3743ca349cf3a628e4df5309cecf
Can I set the above configuration? (type 'yes' to accept): yes
```

上图的 1 意思是一个 master 对应一个 slaver，如果是 2 的话意思是一个 master 对应两个 slaver，以此类推。之后写主和从的 ip 地址，系统会自动识别。

```
[root@localhost data]# cat nodes-6379.conf
2814234930c1df302dd1eaa6d94578f935a4287 :000 myself,master - 0 0 0 connected
vars currentEpoch 0 lastVoteEpoch 0
[root@localhost data]# cat nodes-6379.conf
446b6f0abdb3743ca349cf3a628e4df5309cecf 127.0.0.1:6381@16381 master - 0 1571887184961 3 connected 10923-16383
715ea09a37c333e6d3a3acbf97f95e3e951de6e7 127.0.0.1:6382@16382 slave 8bd87b4d5fc269f286d95c55dfa676b74720a9bd 0 1571887186000 4 connected
948abfc95e0c37b2726bd0450d3f2e90b78ca089 127.0.0.1:6383@16383 slave 91f0f76da398e38b38ab0df4e49248dee2756794 0 1571887185000 5 connected
2814234930c1df302dd1eaa6d94578f935a4287 127.0.0.1:6384@16384 slave 446b6f0abdb3743ca349cf3a628e4df5309cecf 0 1571887187019 6 connected
91f0f76da398e38b38ab0df4e49248dee2756794 127.0.0.1:6380@16380 master - 0 1571887185000 2 connected 5461-10922
8bd87b4d5fc269f286d95c55dfa676b74720a9bd 127.0.0.1:6379@16379 myself,master - 0 1571887184000 1 connected 0-5460
vars currentEpoch 6 lastVoteEpoch 0
[root@localhost data]# cat nodes-6379.conf
```

```
12895:M 23 Oct 19:57:54.349 # WARNING: The TCP backlog setting of 511 cannot be enforced because /proc/sys/net/core/somaxconn is set to the lower value of 128.
12895:M 23 Oct 19:57:54.349 # Server initialized
12895:M 23 Oct 19:57:54.349 # WARNING overcommit_memory is set to 0! Background save may fail under low memory condition. To fix this issue add 'vm.overcommit_memory = 1' to /etc/sysctl.conf and then reboot or run the command 'sysctl vm.overcommit_memory=1' for this process to take effect.
12895:M 23 Oct 19:57:54.350 # WARNING you have Transparent Huge Pages (THP) support enabled in your kernel. This will create latency and memory usage issues with Redis. To fix this issue run the command 'echo never > /sys/kernel/mm/transparent_hugepage/enabled' as root, and add it to your /etc/rc.local in order to retain the setting after a reboot. Redis must be restarted after THP is disabled.
12895:M 23 Oct 19:57:54.350 * Ready to accept connections
12895:M 23 Oct 20:19:37.947 # configEpoch set to 1 via CLUSTER SET-CONFIG-EPOCH
12895:M 23 Oct 19:57:54.349 # IP address for this node updated to 127.0.0.1
12895:M 23 Oct 20:19:42.909 # Cluster state changed: ok
12895:M 23 Oct 20:19:43.530 * Slave 127.0.0.1:6382 asks for synchronization
12895:M 23 Oct 20:19:43.530 * Partial resynchronization not accepted: Replication ID mismatch (Slave asked for 'f5a2a97ab807c0371d5a01bedd93c71afd28b0fa' my replication IDs are 'daad139829193ea0a96d6eb62794e0924d3b4eaf' and '0000000000000000000000000000000000000000')
12895:M 23 Oct 20:19:43.530 * Starting BGSAVE for SYNC with target: disk
12895:M 23 Oct 20:19:43.531 * Background saving started by pid 13261
13261:C 23 Oct 20:19:43.533 * DB saved on disk
13261:C 23 Oct 20:19:43.534 * RDB: 0 MB of memory used by copy-on-write
12895:M 23 Oct 20:19:43.629 * Background saving terminated with success
12895:M 23 Oct 20:19:43.629 * Synchronization with slave 127.0.0.1:6382 succeeded
```

注意，这个时候我们向 master 中放数据应该写 redis-cli -c 表示集群客户端，如果按照原先的方法写会出问题。

```
[root@localhost ~]# redis-cli
127.0.0.1:6379> set name itheima
(error) MOVED 5798 127.0.0.1:6380
127.0.0.1:6379>
[root@localhost ~]# redis-cli -c
127.0.0.1:6379> set name itheima
-> Redirected to slot [5798] located at 127.0.0.1:6380
OK
127.0.0.1:6380>
```

对应的在 slaver 中的取的操作也要改成 redis-cli -c -p 63xx


```
master1 | master2 | master3 | slave1 | slave2 | slave3 | 主命令操作客户端 | master1客户端 | slave1客户端
[root@localhost /]# redis-cli -c -p 6382
127.0.0.1:6382> get name
-> Redirected to slot [5798] located at 127.0.0.1:6380
"itheima"
127.0.0.1:6380>

"itheima"
127.0.0.1:6380> cluster nodes
2814234930c1d1f302dd1eaad094378f935a4287 127.0.0.1:6384@16384 slave 446b6f0abdbe3743ca349cf3a628e4df5309cecf 0 1571888641707 6 connected
91f0f76da398e38b38ab0df4e49248dee2756794 127.0.0.1:6380@16380 myself,master - 0 1571888639000 2 connected 5461-10922
715ea09a37c333e6d3a3achf97f95e3e951de6e7 127.0.0.1:6382@16382 master - 0 1571888642014 7 connected 0-5460
948abfc95e0c37b2726bd0450d3f2e90b78ca089 127.0.0.1:6383@16383 slave 91f0f76da398e38b38ab0df4e49248dee2756794 0 1571888640683 5 connected
8bd87b4d5fc269f286d95c55dfa676b74720a9bd 127.0.0.1:6379@16379 master,fail - 1571888211253 1571888208000 1 disconnected
446b6f0abdbe3743ca349cf3a628e4df5309cecf 127.0.0.1:6381@16381 master - 0 1571888640000 3 connected 10923-16383
127.0.0.1:6380>
```

redis-trib.rb 中不需要使用 cluster nodes 命令，
他有自己的一套命令。

Cluster集群结构搭建



Cluster配置

- 设置加入cluster，成为其中的节点

```
cluster-enabled yes|no
```

- cluster配置文件名，该文件属于自动生成，仅用于快速查找文件并查询文件内容

```
cluster-config-file <filename>
```

- 节点服务响应超时时间，用于判定该节点是否下线或切换为从节点

```
cluster-node-timeout <milliseconds>
```

- master连接的slave最小数量

```
cluster-migration-barrier <count>
```

Cluster节点操作命令

- 查看集群节点信息

```
cluster nodes
```

- 进入一个从节点redis，切换其主节点

```
cluster replicate <master-id>
```

- 发现一个新节点，新增主节点

```
cluster meet ip:port
```

- 忽略一个没有slot的节点

```
cluster forget <id>
```

- 手动故障转移

```
cluster failover
```