

问题收集: https://shimo.im/docs/qWdtj63QcYJ9HHyT/《Java1期第二十三课问题收集》

### 目录

- 1. Redis集群与高可用
- 2. Redisson介绍
- 3. Hazelcast介绍
- 4. 总结回顾与作业实践

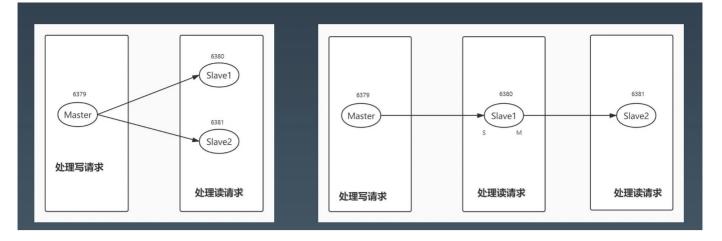
# 1. Redis的集群与高可用

## Redis 主从复制:从单机到多节点

极简的风格, 从节点执行:

> SLAVEOF 127.0.0.1 6379 也可以在配置文件中设置。

注意:从节点只读、异步复制。



备注: 第一: 当从比较多的时候,每个都会从主库拉去信息 会占用主库的带宽;

第二:链比较长的时候,延迟比较大

Redis Sentinel[ˈsentɪnl](哨兵) 主从切换:走向高可用 failover自动的故 障转移

高可用,就是系统可用的时间比较长,实现方式一般是 redis sentinel;

可用性: 点单位时间内 可用的时间长;

稳定性:单位时间内, 宕机次数少;

## Redis 主从复制:从单机到多节点

可以做到监控主从节点的在线状态,并做切换(基于raft协议)。

两种启动方式:

2

```
1 > redis-sentinel sentinel.conf
```

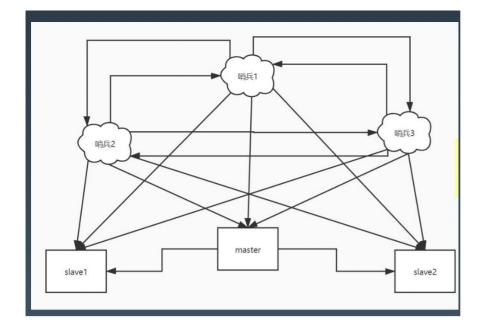
> redis-server redis.conf --sentinel

#### sentinel.conf配置:

```
1 sentinel monitor mymaster 127.0.0.1 6379 2
```

- 2 sentinel down-after-milliseconds mymaster 60000
- 3 sentinel failover-timeout mymaster 180000
- 4 sentinel parallel-syncs mymaster 1

不需要配置从节点,也不需要配置其他sentinel信息
redis sentinel原理介绍:http://www.redis.cn/topics/sentinel.html
redis复制与高可用配置:https://www.cnblogs.com/itzhouq/p/redis5.html



## Redis Cluster:走向分片

主从复制从容量角度来说、还是单机。

Redis Cluster通过一致性hash的方式,将数据分散到多个服务器节点:先设计 16384 个哈希槽,分配到多台redis-server。当需要在 Redis Cluster中存取一个 key时, Redis 客户端先对 key 使用 crc16 算法计算一个数值,然后对 16384 取模,这样每个 key 都会对应一个编号在 0-16383 之间的哈希槽,然后在 此槽对应的节点上操作。

### > cluster-enabled yes

注意: 1、节点间使用gossip通信,规模<1000

- 2、默认所有槽位可用,才提供服务
- 3、一般会配合主从模式使用

redis cluster介绍:http://redisdoc.com/topic/cluster-spec.html

redis cluster原理:https://www.cnblogs.com/williamjie/p/11132211.html

redis cluster详细配置:https://www.cnblogs.com/renpingsheng/p/9813959.html

## Java中配置使用Redis Sentinel

代码示例 && 作业

### Java中配置使用Redis Cluster

代码示例 && 作业

# 第 23 课 2. Redission介绍

Redis 的Java分布式组件库-Redission

基于Netty NIO, API线程安全。

亮点:大量丰富的分布式功能特性,比如JUC的线程安全集合和工具的分布式版本,分 布式的基本数据类型和锁等。

官网:https://github.com/redisson/redisson

Redission

示例1: 分布式锁, RLock

示例2: 分布式的Map, RMap 代码演示。

第 23 课 3. Hazelcast介绍

内存网格 - Hazelcast

Hazelcast IMGD(in-memory data grid) 是一个标准的内存网格系统;它具有以下的一 些基本特性:

- 1. 分布式的:数据按照某种策略尽可能均匀的分布在集群的所有节点上。
- 2. 高可用:集群的每个节点都是 active 模式,可以提供业务查询和数据修改事务;部 分节点不可用,集群依然可以提供业务服务。
- 3. 可扩展的:能按照业务需求增加或者减少服务节点。
- 4. 面向对象的:数据模型是面向对象和非关系型的。在 java 语言应用程序中引入 hazelcast client api是相当简单的。
- 5. 低延迟:基于内存的,可以使用堆外内存。

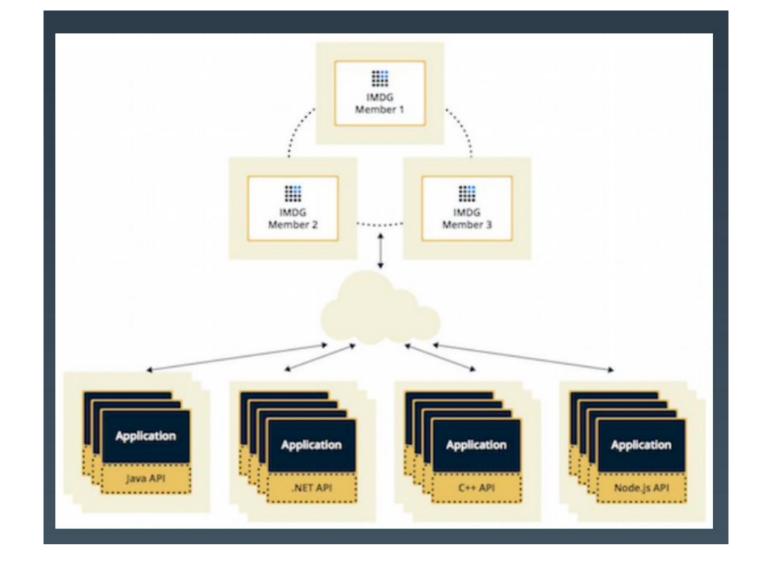
文档:https://docs.hazelcast.org/docs/4.1.1/manual/html-single/index.html

内存网格 - Hazelcast 架构

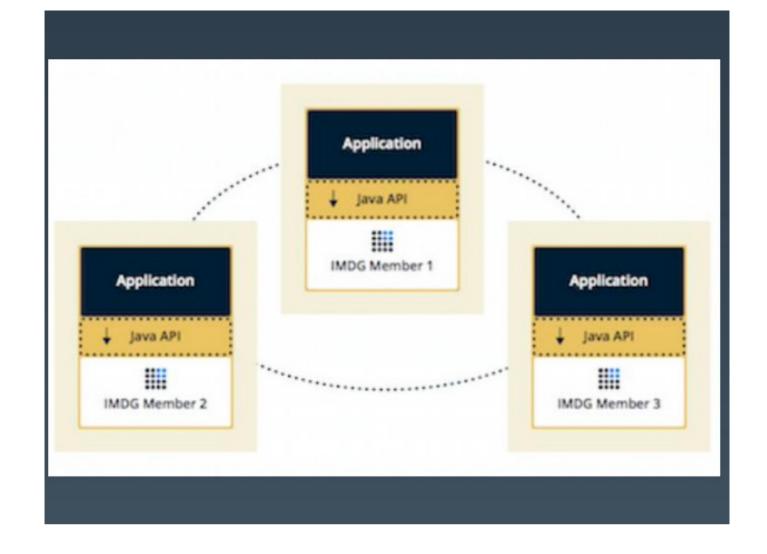
Java C++ .NET	
Java C++ .NET	
	<u> </u>
Memcached REST Open Client Network Protocol	(Connection,
Serialization (Serializable, Externalizable, DataSerializable, IdentifiedDataSerializable, Portable, Custom)	ction, I
Web Sessions Continuous Hibernate 2 <sup>nd</sup> Level JCache Journal Java.util.concurrent	Security Encryption, Authentication, Authorization, LoginModule) WAN
Penlicated Ping Lock/ Atomic	tion, A
Map MultiMap Map Set Buffer Sem. Long UUID Queue Topic  SQL Query Predicate Entry Processor Executor Service Map / Reduce Aggregation  Low-level Services API  Node Engine (Threads, Instances, Eventing, Wait/Notify, Invocation)	Se uthent
Low-level Services API	ecurity nticatio
Node Engine (Threads, Instances, Eventing, Wait/Notify, Invocation)	n, Aut
Partition Management (Members, Lite Members, Master Partition, Replicas, Migrations, Partition Groups, Partition Aware)	horiza
Cluster Management with Cloud Discovery SPI (Multicast, IP List, AWS, JClouds, Kubernetes, Consul)	ition, L
Networking (IPv4, IPv6)	-ogin\
On-heap Store HD Memory Store	Module
Hot Restart Store	
■ Open Source ■ Enterprise Edition ■ Enterprise HD Edition ■ HD Enabled Feature	

内存网格 - Hazelcast 部署模式

Client-Server模式



嵌入(Embedded)模式



内存网格 - Hazelcast 数据分区

以 Map 结构说明如下:

数据集默认分为 271 个分区;可以通过 hazelcast.partition.count 配置修改。

所有分区均匀分布在集群的所有节点上;同一个节点不会同时包含一个分区的多个副本(副本总是 分散的以保证高可用)。

副本配置: <hazelcast>

内存网格 - Hazelcast 集群与高可用

- 1、AP,集群自动管理,
- 2、扩容和弹性,分区自动rebalance,业务无感知,
- 3、相关问题:

内存网格 - Hazelcast 事务支持

#### 支持事务操作:

TransactionContext context = hazelcastInstance.newTransactionContext(options);
context.beginTransaction(); try {
// do other things
context.commitTransaction(); } catch (Throwable t) {

支持两种事务类型:

context.rollbackTransaction(); }

ONE PHASE: 只有一个提交阶段;在节点宕机等情况下可能导致系统不一致;

TWO\_PHASE: 在提交前增减一个 prepare 阶段;该阶段检查提价冲突,然后将commit log 拷贝到一个本

分 节点;如果本节点宕机,备份节点会完成事务提交动作;

内存网格 - Hazelcast 数据亲密性

确保业务相关的数据在同一个集群节点上,避免操作多个数据的业务事务在执行中通过网络请求数据,从而实现更低的事务延迟。

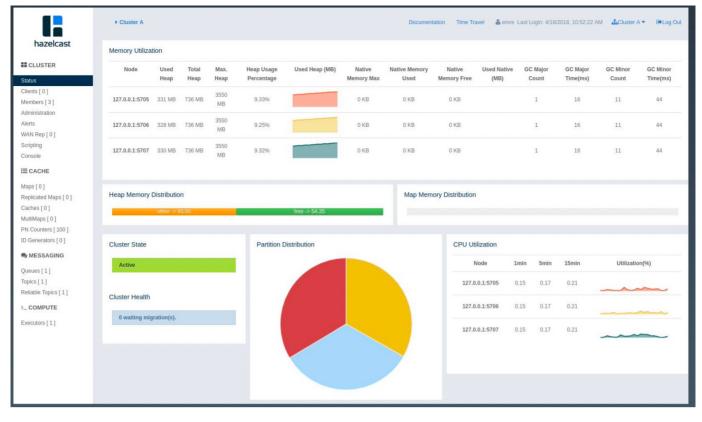
1. 通过 PartitionAware 接口,可以将相关数据定位在相同的节点上; public interface PartitionAware<T>

T getPartitionKey(); }

2. 自定义:PartitioningStrategy <map name="name-of-the-map">

<partition-strategy> com.hazelcast.partition.strategy.StringAndPartitionAwarePartitioningStrategy
</partition-strategy> </map>

内存网格 - Hazelcast 控制台



第23课4.总结回顾与作业实践

### 第 23 课总结回顾

### 第 23 课作业实践

- 1、(必做)配置redis的主从复制, sentinel高可用, Cluster集群。
- 2、(选做)练习示例代码里下列类中的作业题:

08cache/redis/src/main/java/io/kimmking/cache/RedisApplication.java

- 3、(选做☆)练习redission的各种功能;
- 4、(选做☆☆)练习hazelcast的各种功能;
- 5、(选做☆☆☆)搭建hazelcast 3节点集群,写入100万数据到一个map,模拟和演 示高可用;