# Redis安装和部署：

1. 使用redis最新版3.2.3进行安装。
2. **使用1个主节点，3个从节点**。主节点提供读写操作，从节点只提供读操作。主节点Master安装在dbp模块，提供大量的写操作服务； 3个从节点。
3. **配置3个哨兵**，**主节点dbp安装1个哨兵**，另外3台从服务器选其中两台各安装一个。作为HA高可用方案，防止主节点单点失败，通过重新选举主节点实现故障快速转移。
4. 安装步骤：
5. 解压
6. 安装gcc
7. 进入redis的bin目录，先执行 make MALLOC=libc； 再执行make install
8. 配置文件：先拷贝redis目录下的配置文件redis.conf和sentinel.conf到/usr/local/etc(或其他任意目录)，再修改

1. Redis节点：

主要的参数：

**bind 主机ip**  #主机ip

**protected-mode no**  #保护模式关闭，否则不能通过远程连接，哨兵机制也不起作用，下面使用密码进行安全保证

**port 端口**  #端口

**daemonize yes** #守护进程

**pidfile /var/run/redis\_端口.pid** #进程号，命名规则redis\_端口号.pid

**logfile /usr/local/logs/redis/redis\_端口.log** #日志文件

**dir /usr/local/data/redis/端口** #持久化文件夹，必须是**空文件夹**

requirepass 密码 #认证密码

masterauth 密码 #和认证密码一致

**maxmemory 最大内存** #eg:10g

**maxmemory-policy allkeys-lru**  #lru算法回收

从节点需要额外配置：

**slaveof 主机 ip #例如slaveof 172.16.21.127 6379**

2.Sentinel哨兵节点:

主要参数：

**port 端口** #命名规则： 本机redis端口前加个2,比如redis:6379: 则sentinel：26379

**sentinel announce-ip 主机ip**

**protected-mode no** #需要手动添加这条。

**dir /usr/local/data/sentinel\_端口** **#空文件夹**

**logfile /usr/local/logs/redis/sentinel\_端口.log**

**sentinel monitor 主节点名称 主节点ip 主节点端口 仲裁至少需要的哨兵数（2）**

#eg：sentinel monitor mymaster 172.16.21.127 6379 2

**sentinel auth-pass 主节点名称 密码** #认证

1. 进入redis的src目录启动redis和sentinel:

**reids-server redis配置文件地址**

**redis-sentinel sentinel配置文件地址 &**

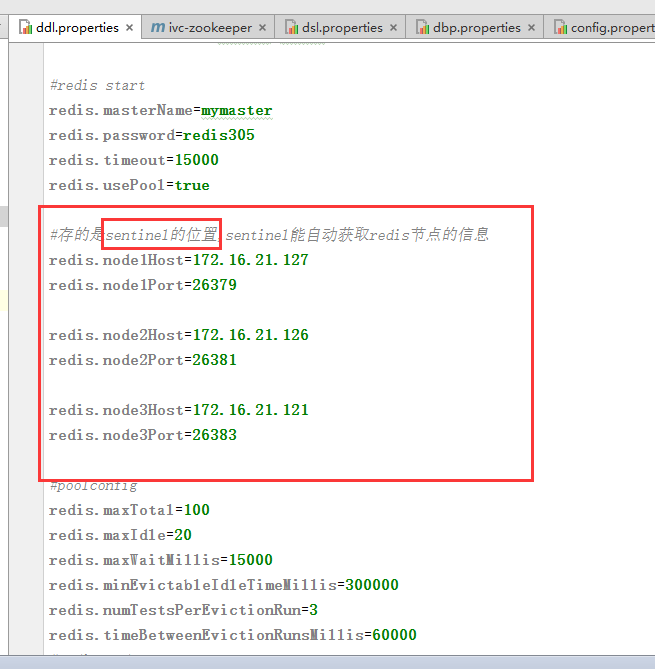
1. 依次启动主节点和从节点后，使用redis-cli连接

reids-cli -h ip地址 -p 端口 -a 密码

使用命令查看部署情况，info replication可查看集群状态

# 缓存技术方案：

1. 使用Redis的Java客户端实现Jedis作为底层的连接工具。配置jedis连接池，使用Sentinel机制作为HA高可用方案，支持单点故障转移。
2. 配合spring-data-redis最新版本1.7.4，对jedis接口进行更高层次的封装，而底层使用的仍然是Jedis，不影响效率。（1.7.4需要spring版本适配，故升级spring版本为4.3.3）
3. 对spring-data-redis进行进一步封装，根据数据结构不同，封装成RedisStringUtil，RedisSetUtil，RedisMapUtil，RedisListUtil四种。利用redis的管道pipeline机制，提供批量操作，减少网络开销，大大提高了性能。
4. ddl.properities: 设置的是sentinel的ip和端口，而不是redis的端口



# 缓存设计方案：

1. **当前登录用户缓存：lu**

使用hash结构，<key, hashkey, value>对应的分别是<lu, username, login times>，即hashkey使用用户名，value使用登录次数。这样好处是能够一次性取出所有的当前登录用户。

1. **用户视野缓存：v:**

使用string结构，简单得存储key-value, <v:username, 视野西南东北经纬度+地图级别>

1. **管辖的所有用户: ru:**

使用Set结构，<ru:terminalSim, set<username>> ，记录托管关系

1. **区域运营限定：lts**

使用hash结构， <lts, terminalSim, areaid>

1. **车库区域限定：gts**

使用hash结构，<gts, terminalSim, areaid>

1. **终端快照缓存：tss:**

使用hash结构， <tss:terminalSim, field, fieldValue>, 一个终端快照使用一个hash结构存储，这样能够快速方便的对单个字段进行修改。

1. **上线状态终端：tso**

使用Set结构，<tso, set<terminalSim>

1. **所有可用服务站，asst**

使用hash结构， <asst, orgName, Organization>，使用json序列化方式存储Organization对象为value。

1. **区域： a:**

使用string结构，结合json序列化存储。<a:areaId, AreaSnapShot>

1. **站内超时告警缓存： atoa**

使用Hash结构，<atoa, terminalSim, TimeOutAlert>