容积表诊断平台：

需求：采集加油站中传感器设备数据进行处理展示。

1. 设备会上传大量基础数据，比如说罐存数据（每个站一秒一条，可能连接几十到几百个站）。
2. 针对基础数据会进行大量计算，比如针对罐存数据会形成液位轨迹数据。计算过程可能长达数秒

原设计：数据采集和处理在同一个服务内，部署和维护效率较低。原设计相当于每个加油站的数据是独立，无法进行汇总。

优化设计：亮点在哪？

1. 业务上将数据的采集和处理分离，加油站只部署数据采集模块，数据处理模块单独部署在一台服务器上。
2. 技术上增加RabbitMQ将数据采集和处理进行解耦，对不同的数据通过MQ发布订阅的来进行数据分发（有些数据需要大量计算，有些数据则可以直接保存）
3. 增加redis来对基础数据进行缓存
4. 使用shardingJDBC对部分数据进行分表操作（后期可以设置主从数据库，但是需要考虑如何实现数据一致性） 涉及到Mysql读写锁表，索引，事务隔离

考虑：以下只是想法，实际项目中并没有完成开发

1. 如果连进来几百个站，TPC可能会到达数百，如何处理，仅通过MQ能否完成数据的限流并且不遗漏数据。如果服务端处理速度小于MQ的消息传递，造成MQ消息堆积该如何处理。 MQ与服务端部署在一起，或者考虑做成MQ的服务器
2. 减少请求数：将采集到的数据以日志文件的形式进行保存（logback日志组件），每次发送一秒的日志，部分对实时性不高的数据可以延长日志堆积时间。（项目中以实现）
3. 这个如果请求量依旧很大，配置多个服务（分布式）：

由Nginx根据请求进行路由分配？iphash：定向流量分发（会不会造成流量倾斜问题，每个站的数据量不一定相同：大小站）~~（油站编码hash值%服务节点数）~~

这里就改成微服务吧，如果做服务高可用，就不能让服务器带着特定状态，这样当特定状态的服务器挂掉之后，会影响这一部分的功能，将服务设置为无状态的，做客户端的负载均衡

将服务端做成微服务，根据请求做负载均衡，设置负载均衡策略，用Ribbon或者事sentinal做负载均衡，随机或者轮询

1. 分布式id：使用雪花算法，按照服务节点划片
2. 数据库：分库分表，主从分离，主写从读。//只用shadingjdbc做过分库分表，没有实现过主从数据库
3. 如果服务端做分布式，如何实现用户会话管理：~~Session共享->session复制(服务节点不会太多，session复制的成本可控)~~ 
   1. 还是选用无状态的会话管理，配置jwt和oauth2.0
4. Redis只用作缓存基础数据(string、list)？是否可以使用其他功能：使用bitmap来实现油站在线情况诊断，每个站占据一位，缓存中可以存一个变量A，每半个小时将变量A与0做&操作（半小时未上传过数据视为离线），加油站有数据上传则将该变量A与油站编号做位操作（每个加油站的id自增，作为偏移位数,用1做完右移再做|操作），该位设为1，查看油站在线情况时，可直接通过遍历该缓存数据来得知由多少站在线，哪些站不在线。
   1. 这个考虑每个油站设置一个标记flag，然后设置过期时间为半小时，每次来数据就更新标记重新设置过期时间。

Redis考虑做**分布式锁/配置中心做配置文件的发布订阅？**

1. 针对计算问题，使用专门的线程池，来执行计算任务：计算密集型任务 参数如何设置
   1. 核心线程数=CPU数+1

线程个数=CPU个数× 期望CPU利用率 ×(1+线程等待时间/线程计算时间);

* 1. 最大线程数=核心线程数
  2. 空闲存活时间：0
  3. 时间单位：TimeUint.SENCONDE
  4. 阻塞队列：ArrayBlockingQueue 长度
  5. 线程工厂: 命名: XXX-pool-%d
  6. 拒绝策略：自定义，继承RejectedExecutionHandler,

重写rejecetedExecution(Runnable,ThreadPoolExecutor)

做数据回池，重新放到MQ中

拒绝策略：修改，等待？了解一下RabbitMq的模型，推拉模式，内部实现