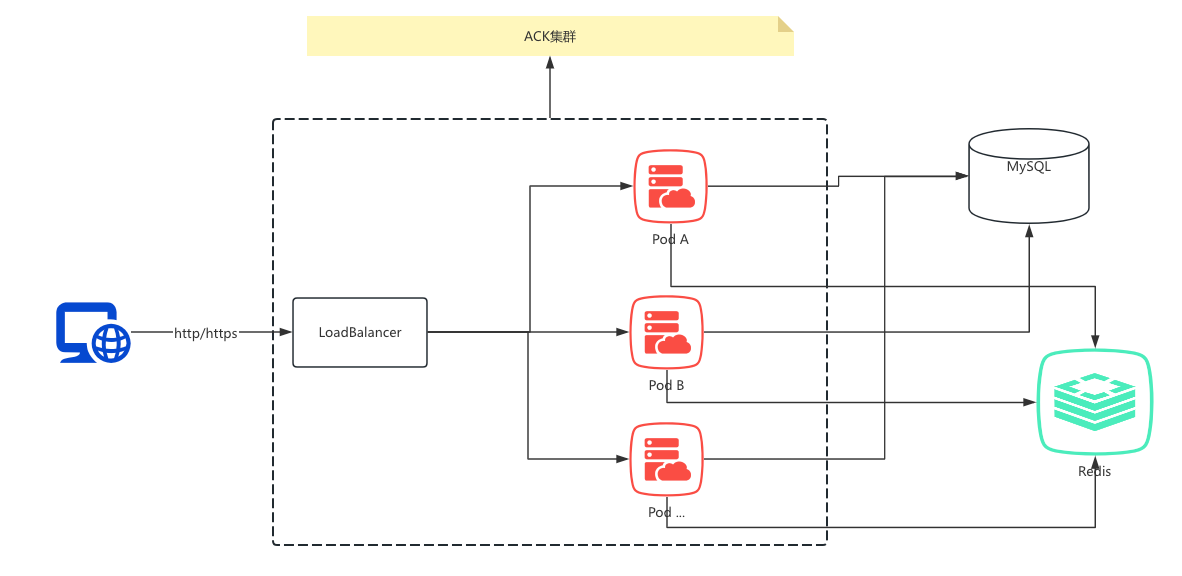
系统部署架构

# 整体架构



# 数据库的高可用性

当前数据库配置：



在真实的线上环境建议：

* 使用阿里云 RDS MySQL，开启 高可用架构（如双机热备）。
* 配置主从同步（如开启异步/半同步复制）并启用自动故障转移。

# 缓存Redis的高可用性

当前配置：



在真实的线上环境建议：

* Redis 部署模式选择 主从架构 或 集群模式，避免单点故障。
* 使用阿里云的 Redis 服务（云数据库 Redis 版），自带高可用和持久化。

# ACK集群架构【确保高可用性和可靠性】

SpringBoot应用程序部署在阿里云ACK集群上。







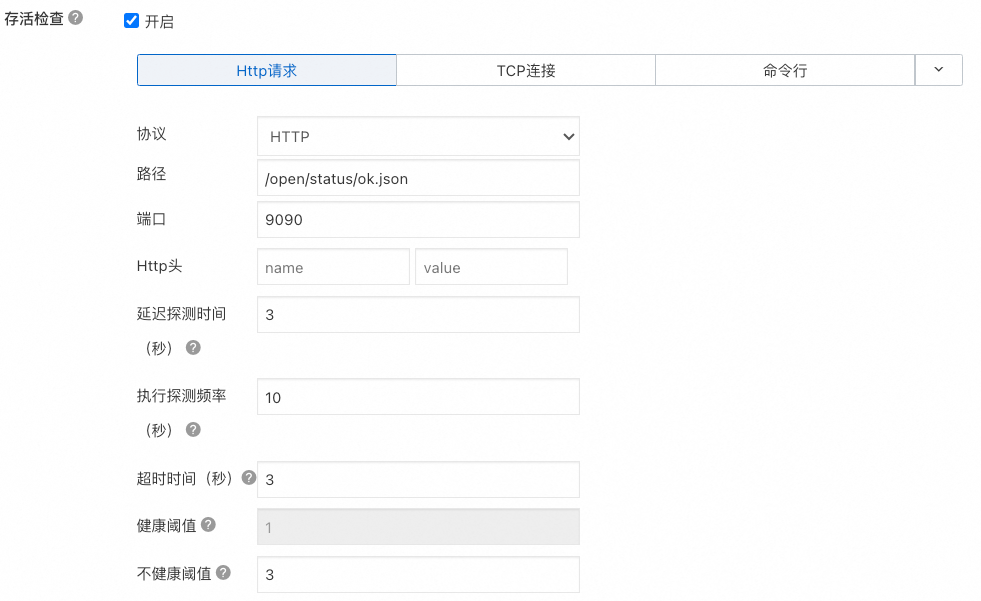
## LoadBalancer【负载均衡】

所有的流量请求通过LoadBalancer进行分发。



### 存活检查

在实际生产中，Java Pod 可能被重启、或者其他的故障、或者网络的因素导致Java Pod无法正常的提供服务，因此需要LoadBalancer可以探测到Java Pod是否在正常工作。



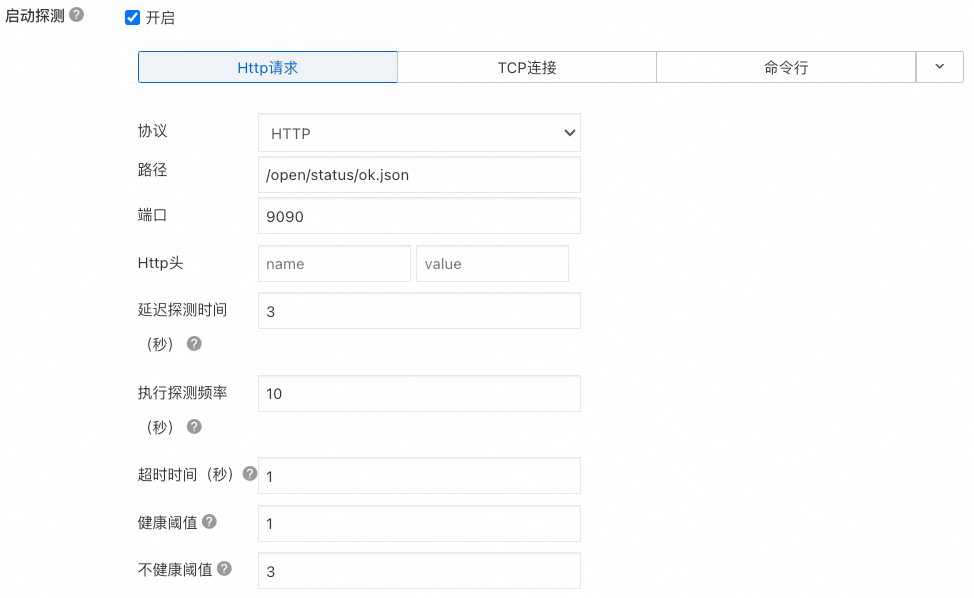
### 就绪检查

在Java Pod发布之后，需要确认是否可以接受流量。



### 启动探测

Java Pod是否启动成功

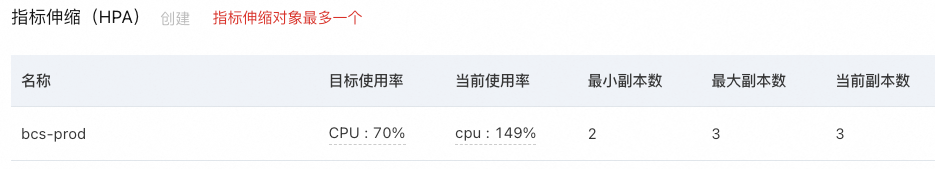


## Java Pod的自动扩容和缩容

在常规部署时，至少部署2台Pod，确保其中一台Pod故障而无法提供服务。

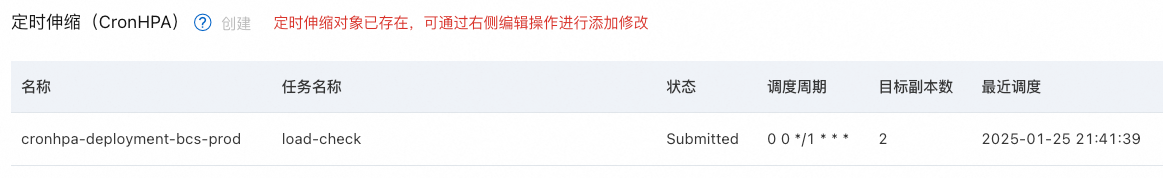
### 自动扩容

自动扩容，当流量过大时，每台Java Pod负载会变高，负载过高不利用正常的提供服务，因此集群需要可以自动扩容。



### 定时缩容

流量变小时，就不需要大规模的集群提供服务了，需要减少Java Pod的数量。需要定时监测当前集群的整体状态，适时进行缩容。



### 应用的反亲和性

当前应用需要监听9090端口，因此不同的Java Pod需要部署在不同的ECS节点上。