1 消息队列的应用场景: 解耦, 异步, 削峰; 举例项目中说明~

2 消息队列的优点和缺点:

优点: 解耦, 异步, 削峰;

缺点: 系统可用性降低; 系统复杂性提高; 一致性问题(通过最终一致性解决);

3 ActiveMQ/RabbitMQ/RocketMQ/Kafka 对比:

ActiviMQ:基于JMS实现的; 社区活跃度比较低;

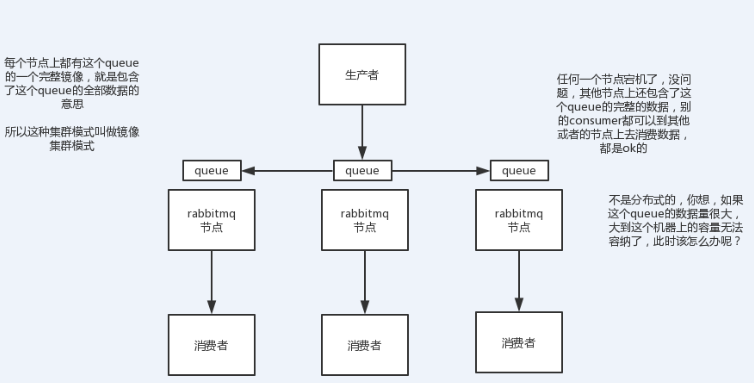
RabbitMQ:基于erlang开发,并发性能好;社区活跃度高;

RocketMQ:基于JMS实现的,阿里开源框架,对技术实力有要求;

Kafka:大数据实时计算和日志采集;

4 如何保证消息队列的高可用性?

镜像集群模式:



5 如何保证消息不被重复消费呢（如何保证消息消费时的幂等性）?

结合自己业务说明;

根据这个id去查询一下,之前消费过吗?没有消费,则处理,如果消费过了,就别处理了,保证别重复处理相同的消息;

实际项目中:先去历史表中查询有没有消费记录, 如果有消费记录,则直接返回,没有的话,做插入操作;

6如何保证消息的可靠性传输（如何处理消息丢失的问题）?

生产者搞丢了:开启confirm机制,每次写的消息会分配一个唯一的id,如果消息写入了mq,则会回传一个ack消息;如果没能处理消息,则回调一个nack接口,告诉你消息接收失败了,可以重试;

MQ搞丢了:开启rabbitmq持久化;1 创建queue时,将其持久化 2发送消息时,将消息的传递模式设置为2(将消息持久化)

消费者搞丢了:

将autoAck关闭;每次自己处理完消息之后,手动发送ack 给rabbitmq;

7 如何保证消息的顺序性？

每个消息创建时,赋予一个全局唯一的/单调递增/连续的序列号;

一个Queue 对应多个消费者;

8 如何设计一个消息队列?

思路: 可伸缩性,扩容; 持久化(落地到磁盘上); 可用性; 支持数据0丢失;