**为什么使用MQ？MQ的优点**

应用解耦 - 系统间通过消息通信，不用关心其他系统的处理。

异步处理 – 不需要等待结果返回，就能继续运行

流量削峰：将突发的业务请求放入MQ中，作为缓冲区。后端的业务根据自己的处理能力从MQ中获取消息，逐个处理任务。可以缓解短时间内的高并发请

求。

延迟队列：可以实现消息发送后，延迟接收的效果。

**消息队列有什么优缺点？**

缺点：系统可用性降低，消息队列挂了，系统也就不行了。系统复杂度提高。

**什么是RabbitMQ？**

RabbitMQ是一款开源的，Erlang编写的，基于AMQP协议的消息中间件

**rabbitmq 的使用场景**

服务间异步通信、定时任务、请求削峰、顺序消费。

用rabbitMQ同步数据库和es之间的数据，当数据库表增加、修改、删除数据，发消息给mq，搜索服务监听消息，并同步最新数据到es

**RabbitMQ基本概念**

Exchange：交换机用来接收和转发消息

Queue： 队列用来存放消息

Binding：绑定，把exchange和queue按照路由规则绑定起来

Routing Key：用于标记消息的路由规则

Binding Key：用于匹配Routing Key

Producer： 生产者，就是投递消息的程序

Consumer： 消费者，就是接收消息的程序

Channel： 消息通道，在客户端的每个连接里，可建立多个channel，每个

channel代表一个会话任务

**死信队列？**

DLX（Dead-Letter-Exchange），死信交换机。当消息在一个队列中变成死信 (dead message) 之后，它能被重新发送到死信交换机，绑定死信交换机的队列就称为死信队列。

**导致的死信的几种原因？**

消息被拒（Basic.Reject /Basic.Nack) 且 requeue = false

消息TTL过期

队列满了，无法再添加

**延迟队列？**

消费者等待特定时间后才能拿到这个消息进行消费。

**RabbitMQ的工作模式**

简单模式：生产者将消息发给队列，消费者监听队列

工作模式：生产者将消息发给队列，多个消费者监听同一个队列

发布订阅：生产者将消息发给交换机，由交换机将消息发送到绑定的每个队列中, 每个消费者监听自己的队列

路由模式：生产者将消息发给交换机，由交换机根据routingkey来转发消息到指定的队列。每个消费者监听自己的队列，并且设置bindingkey

主题模式：生产者将消息发给交换机，由交换机根据routingkey来转发消息到指定的队列。每个消费者监听自己的队列，并且设置带统配符的bindingkey。

**消息如何分发？**

轮询分发。

预取值分发，预取值就是定义通道上允许的未确认消息的最大数量，一旦达到上限，停止分发消息

**消息怎么路由？**

fanout：交换机将消息广播到所有绑定的队列上

direct：如果路由键完全匹配，消息就被投递到相应的队列

topic：基于路由键，可以使用通配符。

**消息基于什么传输？**

由于 TCP 连接的创建和销毁开销较大，且并发数受系统资源限制，会造成性能瓶颈。RabbitMQ 使用信道的方式来传输数据。信道是建立在TCP 连接内的虚拟连接，且每条 TCP 连接上的信道数量没有限制。

**如何避免消息重复投递或重复消费？**

场景：生产者发送消息后，可能因为网络问题没有收到broker的确认，导致重复发送，或者消费者处理消息后，在发送确认之前崩溃了，导致消息重新入队，再次被消费

生产者发送消息时，每条消息携带全局唯一ID(如UUID)，作为去重的依据，确保即使消息被重复投递，业务逻辑也不会被重复执行

消费者实现幂等性处理：

幂等性设计：业务逻辑保证多次处理同一消息的结果一致，例如，使用数据库唯一索引避免重复插入

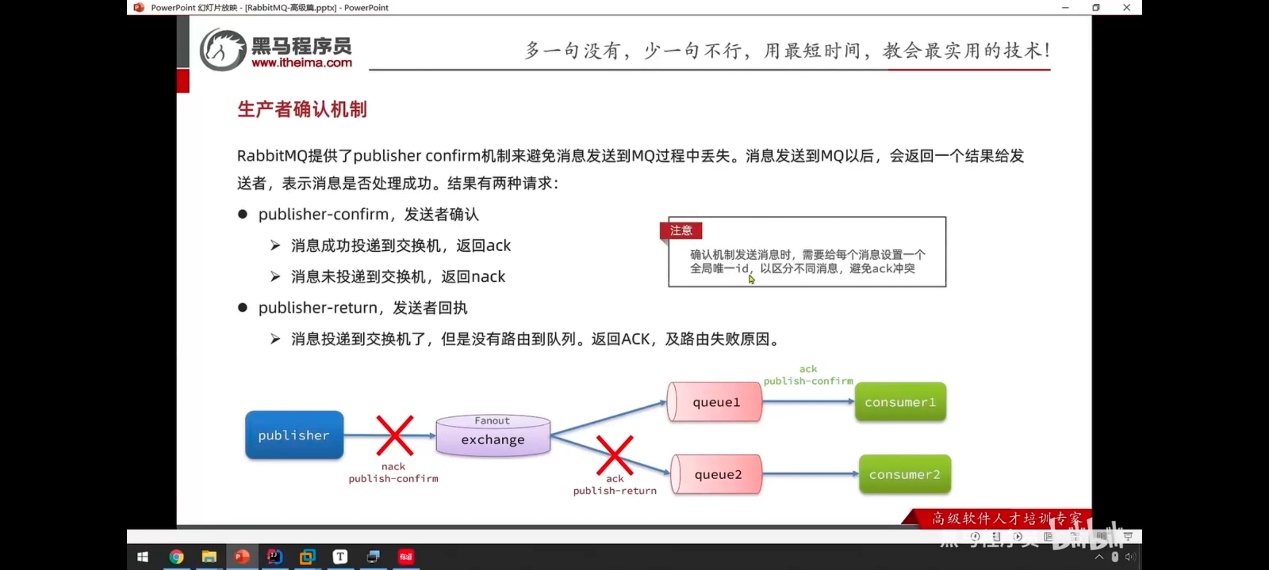
记录已处理消息ID：在数据库或缓存中记录已处理的消息 ID，处理前校验是否已存在；

**手动ACK和自动ACK**

自动ACK：消息一旦被接收，消费者自动发送ACK

手动ACK：消息接收后，不会发送ACK，需要手动调用

**如何保证RabbitMQ消息的可靠传输？**



生产者确认机制：

publisher-confirm，确认模式，RabbitMQ会调用ConfirmCallback函数通知发送者；消息成功投递到交换机，返回ack；消息未投递到交换机，返回nack

publisher-return，退回模式，消息投递到交换机了，但是没有路由到队列。RabbitMQ会调用ReturnCallback函数通知发送者，告知失败原因

消息持久化、交换机持久化、队列持久化

消费者确认机制：

消费者处理消息成功，Spring返回ACK给MQ，消息才被移除。

消费者处理消息失败，抛出异常，Spring返回NACK或者不返回结果，消息不被移除。

消费者重试机制：

默认情况下，消费者处理失败时，消息会再次回到MQ队列，然后投递给其它消费者。Spring提供消费者重试机制，在处理失败后不返回NACK，直接在消费者本地重试。多次重试失败后将消息投递到异常交换机，交由人工处理。

开启消费者确认机制为auto，由spring处理，消息处理成功返回ack，失败返回nack，mq收到ack才会删除消息。开启消费者失败本地重试机制，并设置messagerecoverer，多次重试失败后将消息投递到异常交换机，交由人工处理。

**RabbitMQ的配置**

用户名和密码、ip和端口、虚拟主机；

队列、交换机、Binding 、Routing Key、Binding Key；

**如何避免消息堆积**

提高消费者处理速度

增加更多消费者

增加队列的存储上限

**RabbitMQ如何保证消息的有序性？**

保证消息发送的有序性

保证一个队列只包含一个消费者

**RabbitMQ处理函数**

@RabbitListener定义：@RabbitListener 注解用于标记一个方法，使其成为消息队列的监听器，即这个方法负责接收来自RabbitMQ的消息。



RabbitTemplate.convertAndSend方法实现发送消息