1、项目中为什么要用MQ：

答：解耦，假设在A系统中修改的某个数据，需要把这个数据也同步到BCD系统中，现在E系统也需要更改这个数据，但是C系统不需要，使用MQ的话，可以通过配置解决，不实用的话，需要在代码中删除调用C的代码，增加调用E的代码；

异步：A系统修改数据需要一秒，BCD系统每个也需要一秒，加起来就需要四秒，但是异步的话，只需要一秒；

消峰：在系统中，可能有一个时间段请求特别多，但是系统处理请求的速度比发送的请求要慢，这样会导致系统挂掉，但是使用MQ可以先将请求放在消息队列中，等峰值过去之后，系统依旧以最高速度处理积压的请求；

2、MQ的缺点：

答：系统可用性降低、系统复杂度提高、一致性问题；

3、MQ消息丢失：

答：1、生产者丢数据，是指生产者的消息没有投递到MQ，RabbitMQ提供了transaction（事务）和confirm来确保生产者不丢失消息。transaction（事务）机制就是说，在发送消息前，开始事务，然后发送消息，发送过程中出现了什么异常，事务就会回滚，发送成功则提交事务。缺点是吞吐量下降了；MQ开启confirm模式，所有在该信道上面发布的消息都将会被指派一个唯一的ID（从1开始），一旦消息被投递所有匹配的队列之后，RabbitMQ就会发送一个ack给生产者，如果RabbitMQ没有收到处理消息，就会发送一个Nack消息给生产者；

2、消息队列丢数据：处理消息队列丢数据的情况，一般是开启持久化磁盘的配置，这个持久化磁盘配置可以和confirm机制配置使用，可以在消息持久化磁盘后再给生产者发送一个ack信号；怎么设置MQ持久化配置：将queue的持久化标识durable设置为true，发送消息的将deliveryMode设置为2

3、消费者丢数据：启用手动确认模式，需要注意的是，启动手动确认模式，一定不能忘记应答消息，不然会造成消息队列阻塞，影响业务执行。

4、RabbitMQ重复消费：

答：产生的原因有两个：一是生产者重试，为了保证消息成功到达MQ，可以选择开启异步确认模式（confirm），MQ在收到消息后会回调应用程序ack或者nack，什么时候回调是由MQ决定的，当负载较高时，程序没有收到ack或nack消息时会进行重试；二是消费者确认异常，自动ack并不存在这种问题，因为消息离开MQ就已经被移除了；手动有可能业务代码报错了导致ack没执行，ack超时了，ack MQ收到了，但是MQ出问题了，RabbltMQ会在消费者连接断开后将unacked的消息重新投递给消费者，当然也会发给原来的消费者。

解决方案是生产者不重试，消费者做幂等；消费者做幂等一般由 业务天然是幂等的、乐观锁、消息表、redis等几种