



## 第二版：RabbitMQ 12 道

### 目录

第二版：RabbitMQ 12 道	1
1、什么是 rabbitmq	1
2、为什么要使用 rabbitmq	1
3、使用 rabbitmq 的场景	2
4、如何确保消息正确地发送至 RabbitMQ？ 如何确保消息接收方消费了消息？	2
发送方确认模式	2
接收方确认机制	2
5.如何避免消息重复投递或重复消费？	3
6、消息基于什么传输？	3
7、消息如何分发？	3
8、消息怎么路由？	4
9、如何确保消息不丢失？	4
10、使用 RabbitMQ 有什么好处？	4
11、rabbitmq 的集群	5
12、mq 的缺点	5
系统可用性降低	5
系统复杂性提高	5
一致性问题	5

我们的网站：<https://tech.souyunku.com>

**关注我们的公众号：搜云库技术团队，回复以下关键字**

回复：**【进群】** 邀请您进 **「技术架构分享群」**

回复：**【内推】** 即可进：北京，上海，广周，深圳，杭州，成都，武汉，南京，

郑州，西安，长沙 **「程序员工作内推群」**

微信搜一搜

搜云库技术团队



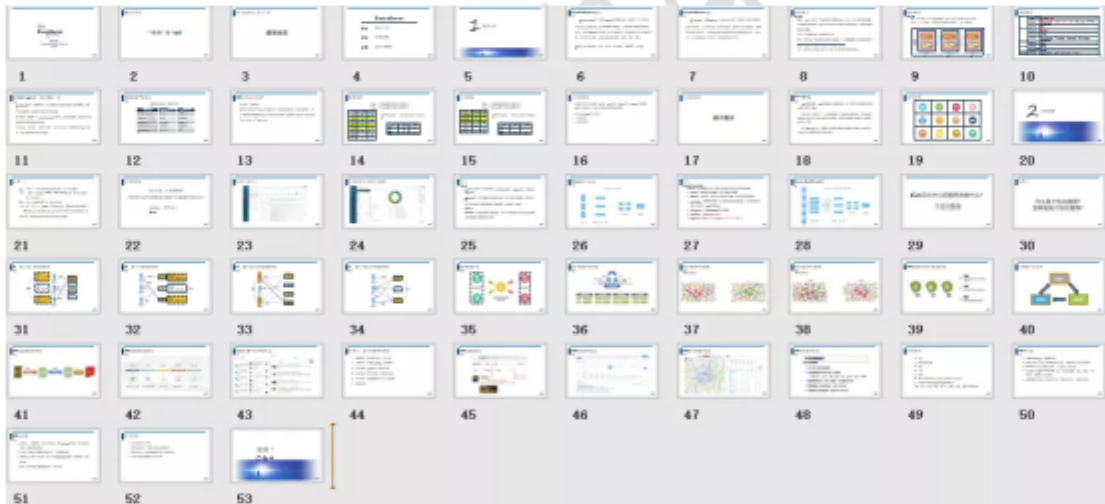
回复 **【1024】** 送 4000G 最新架构师视频

回复 **【PPT】** 即可无套路获取，以下最新整理调优 PPT!

## 46 页《JVM 深度调优，演讲 PPT》



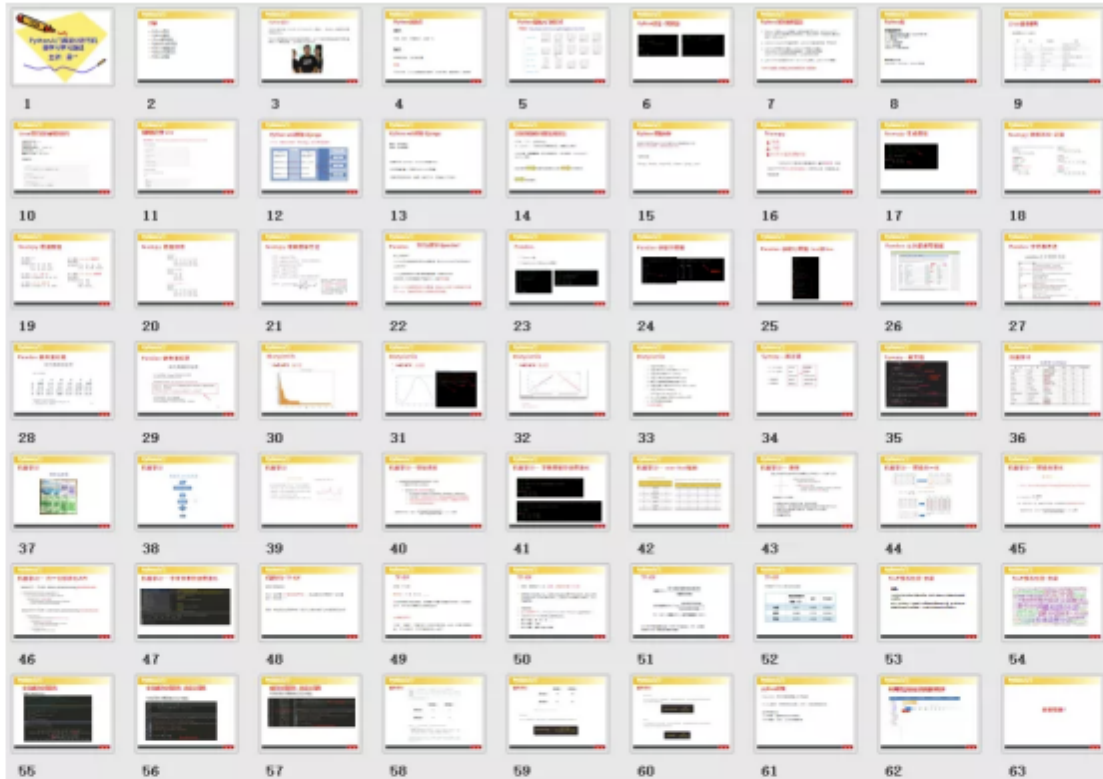
## 53 页《Elasticsearch 调优演讲 PPT》



## 63 页《Python 数据分析入门 PPT》

微信搜一搜

搜云库技术团队



微信扫一扫

<https://tech.souyunku.com>

技术、架构、资料、工作、内推  
专注于分享最有价值的互联网技术干货文章

## 1、什么是 rabbitmq



采用 AMQP 高级消息队列协议的一种消息队列技术,最大的特点就是消费并不需要确保提供方存在,实现了服务之间的高度解耦

## 2、为什么要使用 rabbitmq

- 1、在分布式系统下具备异步,削峰,负载均衡等一系列高级功能;
- 2、拥有持久化的机制, 进程消息, 队列中的信息也可以保存下来。
- 3、实现消费者和生产者之间的解耦。
- 4、对于高并发场景下, 利用消息队列可以使得同步访问变为串行访问达到一定量的限流, 利于数据库的操作。
- 5.可以使用消息队列达到异步下单的效果, 排队中, 后台进行逻辑下单。

## 3、使用 rabbitmq 的场景

- 1、服务间异步通信
- 2、顺序消费
- 3、定时任务
- 4、请求削峰

## 4、如何确保消息正确地发送至 RabbitMQ? 如何确保消息接收方消费了消息?

### 发送方确认模式

将信道设置成 confirm 模式 (发送方确认模式), 则所有在信道上发布的消息都会被指派一个唯一的 ID。



一旦消息被投递到目的队列后，或者消息被写入磁盘后（可持久化的消息），信道会发送一个确认给生产者（包含消息唯一 ID）。

如果 RabbitMQ 发生内部错误从而导致消息丢失，会发送一条 `nack`（not acknowledged，未确认）消息。

发送方确认模式是异步的，生产者应用程序在等待确认的同时，可以继续发送消息。当确认消息到达生产者应用程序，生产者应用程序的回调方法就会被触发来处理确认消息。

### 接收方确认机制

#### 接收方消息确认机制

消费者接收每一条消息后都必须进行确认（消息接收和消息确认是两个不同操作）。只有消费者确认了消息，RabbitMQ 才能安全地把消息从队列中删除。

这里并没有用到超时机制，RabbitMQ 仅通过 Consumer 的连接中断来确认是否需要重新发送消息。也就是说，只要连接不中断，RabbitMQ 给了 Consumer 足够长的时间来处理消息。保证数据的最终一致性；

#### 下面罗列几种特殊情况

如果消费者接收到消息，在确认之前断开了连接或取消订阅，RabbitMQ 会认为消息没有被分发，然后重新分发给下一个订阅的消费者。（可能存在消息重复消费的隐患，需要去重）

如果消费者接收到消息却没有确认消息，连接也未断开，则 RabbitMQ 认为该消费者繁忙，将不会给该消费者分发更多的消息。

## 5. 如何避免消息重复投递或重复消费？

在消息生产时，MQ 内部针对每条生产者发送的消息生成一个 `inner-msg-id`，作为去重的依据（消息投递失败并重传），避免重复的消息进入队列；



在消息消费时，要求消息体中必须要有一个 bizId（对于同一业务全局唯一，如支付 ID、订单 ID、帖子 ID 等）作为去重的依据，避免同一条消息被重复消费。

## 6、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。信道是建立在真实的 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。

## 7、消息如何分发？

若该队列至少有一个消费者订阅，消息将以循环（round-robin）的方式发送给消费者。每条消息只会分发给一个订阅的消费者（前提是消费者能够正常处理消息并进行确认）。

通过路由可实现多消费的功能

## 8、消息怎么路由？

消息提供方 -> 路由 -> 一至多个队列

消息发布到交换器时，消息将拥有一个路由键（routing key），在消息创建时设定。

通过队列路由键，可以把队列绑定到交换器上。

消息到达交换器后，RabbitMQ 会将消息的路由键与队列的路由键进行匹配（针对不同的交换器有不同的路由规则）；

常用的交换器主要分为一下三种





fanout: 如果交换器收到消息, 将会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: 可以使来自不同源头的消息能够到达同一个队列。使用 topic 交换器时, 可以使用通配符

## 9、如何确保消息不丢失?

消息持久化, 当然前提是队列必须持久化

RabbitMQ 确保持久性消息能从服务器重启中恢复的方式是, 将它们写入磁盘上的一个持久化日志文件, 当发布一条持久性消息到持久交换器上时, Rabbit 会在消息提交到日志文件后才发送响应。

一旦消费者从持久队列中消费了一条持久化消息, RabbitMQ 会在持久化日志中把这条消息标记为等待垃圾收集。如果持久化消息在被消费之前 RabbitMQ 重启, 那么 Rabbit 会自动重建交换器和队列 (以及绑定), 并重新发布持久化日志文件中的消息到合适的队列。

## 10、使用 RabbitMQ 有什么好处?

- 1、服务间高度解耦
- 2、异步通信性能高
- 3、流量削峰

## 11、rabbitmq 的集群

镜像集群模式



你创建的 queue，无论元数据还是 queue 里的消息都会存在于多个实例上，然后每次你写消息到 queue 的时候，都会自动把消息到多个实例的 queue 里进行消息同步。

好处在于，你任何一个机器宕机了，没事儿，别的机器都可以用。坏处在于，第一，这个性能开销也太大了吧，消息同步所有机器，导致网络带宽压力和消耗很重！第二，这么玩儿，就没有扩展性可言了，如果某个 queue 负载很重，你加机器，新增的机器也包含了这个 queue 的所有数据，并没有办法线性扩展你的 queue

## 12、mq 的缺点

### 系统可用性降低

系统引入的外部依赖越多，越容易挂掉，本来你就是 A 系统调用 BCD 三个系统的接口就好了，人 ABCD 四个系统好好的，没啥问题，你偏加个 MQ 进来，万一 MQ 挂了咋整？MQ 挂了，整套系统崩溃了，你不就完了么。

### 系统复杂性提高

硬生生加个 MQ 进来，你怎么保证消息没有重复消费？怎么处理消息丢失的情况？怎么保证消息传递的顺序性？头大头大，问题一大堆，痛苦不已

### 一致性问题

A 系统处理完了直接返回成功了，人都以为你这个请求就成功了；但是问题是，要是 BCD 三个系统那里，BD 两个系统写库成功了，结果 C 系统写库失败了，咋整？你这数据就不一致了。

所以消息队列实际是一种非常复杂的架构，你引入它有很多好处，但是也得针对它带来的坏处做各种额外的技术方案和架构来规避掉，最好之后，你会发现，妈



六 微信搜一搜

Q 搜云库技术团队 |



呀，系统复杂度提升了一个数量级，也许是复杂了 10 倍。但是关键时刻，用，还是得用的

公众号：架构师专栏