**4.X文档 RocketMQ**

**消息构成：** Topic, 必填项。在RocketMQ-JMS中，createQueue是不支持的，此外messageSelector也不支持。body ，properties ，transactionId 。

Tag：Topic 是一级分类，而 Tag 可以理解为是二级分类。使用 Tag 可以实现对 Topic 中的消息进行过滤。Tag 可以用来区分同一个 Topic 下相互关联的消息，例如全集和子集的关系、流程先后的关系。

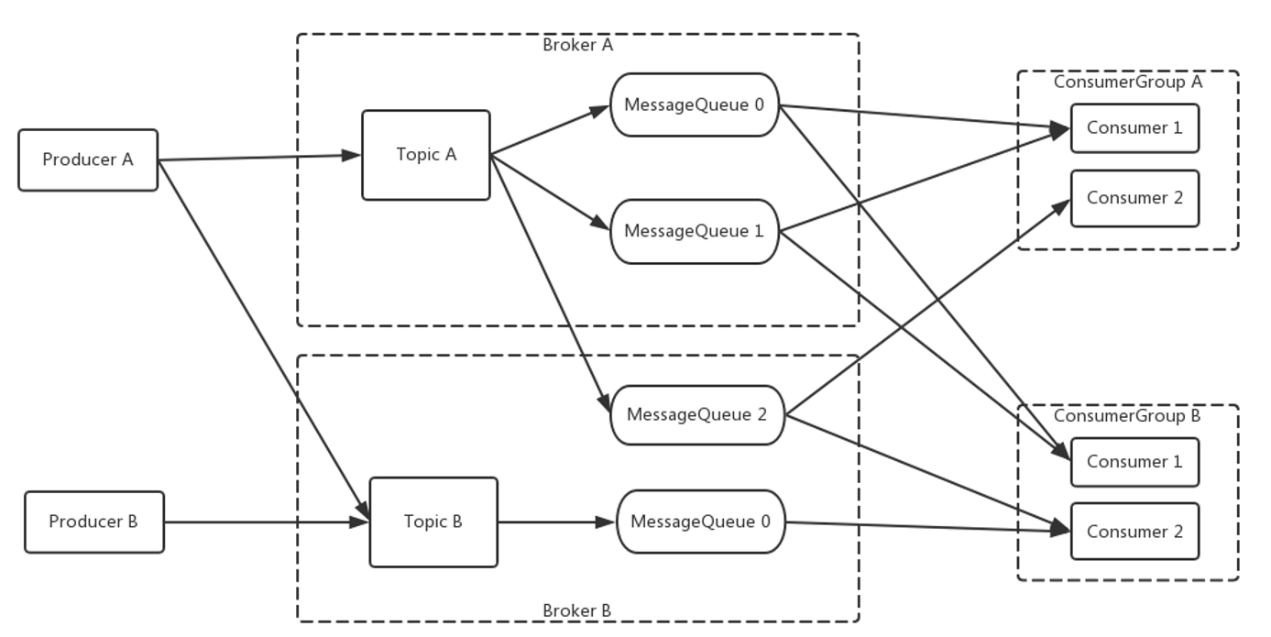
Keys：在业务层面的设置唯一标识码 keys 字段，方便将来定位消息丢失问题**A screen shot of a computer program

AI-generated content may be incorrect.**

Producer：消息的发送者，需要注意的是，生产环境中不同消息类型需要使用不同的主题。

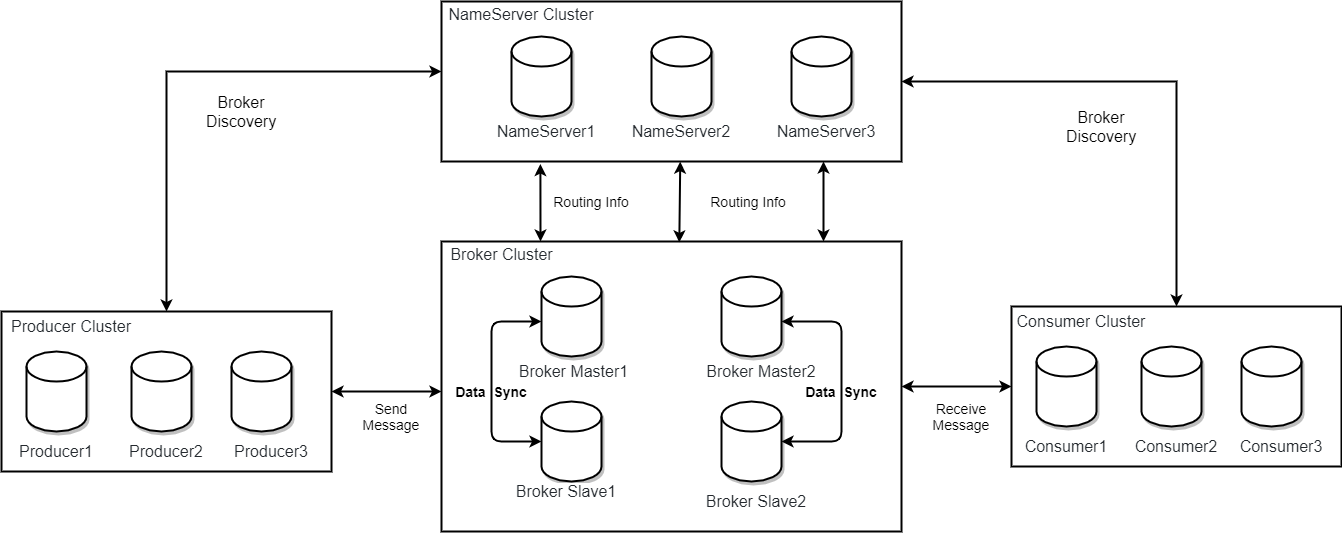
Consumer：

Topic : 每条消息只能属于一个Topic，是RocketMQ进行消息订阅的基本单位。

为了支持高并发和水平扩展，需要对 Topic 进行分区，在 RocketMQ 中这被称为队列。一个 Topic 可能有多个队列，并且可能分布在不同的 Broker 上

ConsumerGroup：两种模式：集群模式, 同一个 ConsumerGroup 中的 Consumer 实例是负载均衡消费 ； 广播模式：每个 Consumer 实例都处理全部的队列。

RocketMQ-JMS连接时，必须设置ProducerID和ConsumerID



NameServer：一个简单的 Topic 路由注册中心，支持 Topic、Broker 的动态注册与发现。无状态，各实例间相互不进行信息通讯，Broker是向每一台NameServer注册自己的路由信息。也可用于客户端查询的队列信息。通过NameServer就可以知道整个Broker集群的路由信息，从而进行消息的投递和消费。

Broker：主要负责消息的存储、投递和查询以及服务高可用保证。Broker 与 NameServer 集群中的所有节点建立长连接，保持心跳。在 Master-Slave 架构中，Broker 分为 Master 与 Slave，一个Master可以对应多个Slave，但是一个Slave只能对应一个Master。

Producer 与 NameServer 集群中的其中一个节点建立长连接，定期从 NameServer 获取Topic路由信息，并向提供 Topic 服务的 Master 建立长连接，且定时向 Master 发送心跳。Producer 完全无状态

Consumer 与 NameServer 集群中的其中一个节点建立长连接，定期从 NameServer 获取 Topic 路由信息，并向提供 Topic 服务的 Master、Slave 建立长连接，且定时向 Master、Slave发送心跳。Consumer 既可以从 Master 订阅消息，也可以从Slave订阅消息。

启动流程：

NameServer

Broker

创建 Topic，需要指定该 Topic 要存储在哪些 Broker 上，也可以在发送消息时自动创建。

发送

消费