[MQ介绍及在CDC中应用 1](#_Toc420609324)

[一、 什么是MQ 1](#_Toc420609325)

[二、 什么是JMS 1](#_Toc420609326)

[1. JMS1.1定义了的部分概念： 1](#_Toc420609327)

[2. JMS的两种域模型 2](#_Toc420609328)

[3. 消息可靠性 2](#_Toc420609329)

[4. 消息的通知确认 3](#_Toc420609330)

[5. 同步还是异步 3](#_Toc420609331)

[三、 ActiveMQ在CDC中的应用 4](#_Toc420609332)

MQ介绍及在CDC中应用

# 什么是MQ

* MQ(message queue)即消息队列，
* 消息分类
	+ 消息：人与人之间通过消息进行沟通，邮件、短信都属于消息。
	+ 消息分为同步消息和异步消息，同步消息在接收对方的返回前，需要挂起，直到返回或超时，如java RPC调用，同步调用依赖于被调用方，如果被调用方失败或网络错误，那么程序就没办法继续执行下去。而异步消息只需要发送消息，不需要对方系统的立即反馈，异步消息，如同一个邮箱系统，我们把信件丢入邮桶，邮递员会更具上面的地址，送达到这封信要去的地方，我们继续做其他的事情，而不会等待回信。
* 消息规范
	+ JMS是sun公司对于消息中间件的一个规范。比如我们国家，各个地区有自己的方言，自从规范了普通话，我们的交流成本降低了。这也正如JMS规范在整个java消息领域的作用。
* 队列
	+ 是一种数据结构，先进先出的特点。

# 什么是JMS

JMS即Java消息服务（Java Message Service）

1. JMS1.1定义了的部分概念：
JMS客户端：接收或发送消息的java系统
JMS消息体：系统间发送的消息体
JMS提供商：JMS规范实现厂商
JMS管理对象：预先配置好的用于JMS客户端的JMS对象。如ConnectionFactory跟JMS服务端的连接工厂，Destination 接收和发送消息的目标地址等
2. JMS的两种域模型

2.1. PTP（point-to-point）即点对点消息传输模型

PTP通过一个先进先出的queue实现。所谓点对点不是指生产者和消费者只有一个。PTP

如下图所示，在CDC中就是PTP，CDC中比较特殊，每个消费者有自己对应的队列。



一个或多个生产者发送消息，消息m2先抵达了queue，然后m1也发出了，并一同存在于一个先进先出的queue里面。消费者也存在一个或多个，对queue里的消息进行消费。但消息被随机的一个消费者消费且仅消费一次。

2.2.pub/sub(publish-subscribe)即发布订阅模型

pub/sub消息模型中，消息被广播给所有订阅者。如下图：


每个消息不止消费一次，而是所有的订阅者都可以消费。

1. 消息可靠性
    在上面，谈及消息体格式定义中，有个字段 JMSDeliveryMode用来表示该消息发送后，JMS提供商应该怎么处理消息。PERSISTENT（持久化）的消息在JMS服务器中持久化。接收端如果采用点对点的queue方式或者Durable Subscription（持久订阅者）方式，那么消息可保证只且只有一次被成功接收。NON\_PERSISTENT(非持久化)的消息在JMS服务器关闭或宕机时，消息丢失。根据发送端和接收端采用的方式，列出如下可靠性表格，以作参考。

|  |  |  |
| --- | --- | --- |
| 消息发送端 | 消息接收端 | 可靠性及因素 |
| PERSISTENT | queue receiver/durable subscriber | 消费一次且仅消费一次。可靠性最好，但是占用服务器资源比较多。 |
| PERSISTENT | non-durable subscriber | 最多消费一次。这是由于non-durable subscriber决定的，如果消费端宕机或其他问题导致与JMS服务器断开连接，等下次再联上JMS服务器时的一系列消息，不为之保留。 |
| NON\_PERSISTENT | queue receiver/durable subscriber | 最多消费一次。这是由于服务器的宕机会造成消息丢失 |
| NON\_PERSISTENT | non-durable subscriber | 最多消费一次。这是由于服务器的宕机造成消息丢失，也可能是由于non-durable subscriber的性质所决定 |

1. 消息的通知确认
在客户端接收了消息之后，JMS服务怎样有效确认消息是否已经被客户端接收呢？Session session=connection.createSession(false,Session.AUTO\_ACKNOWLEDGE);这段代码创建一个非事务性的session，并采用auto\_acknowledge方式通知JMS服务器。如果采用事务性session时，通知会伴随session的commit/rollback同时发送通知。在我们采用非事务session时，有三种通知方式。

|  |  |
| --- | --- |
| 通知方式 | 效果 |
| DUPS\_OK\_ACKNOWLEDGE | session延迟通知。如果JMS服务器宕机，会造成重复消息的情况。程序必须保证处理重复消息而不引起程序逻辑的混乱。 |
| AUTO\_ACKNOWLEDGE | 当receive或MessageListener方法成功返回后自动通知。 |
| CLIENT\_ACKNOWLEDGE | 客户端调用消息的acknowledge方法通知 |

1. 同步还是异步

AcitveMQ是以异步模式发送消息。例外的情况：在没有使用事务的情况下，生产者以 PERSISTENT传送模式发送消息。在这种情况下，send方法都是同步的，并且一直阻塞直到 ActiveMQ发回确认消息：消息已经存储在持久性数据存储中。这种确认机制保证消息不会丢失，但会造成生产者阻塞从而影响反应时间。

高性能的程序一般都能容忍在故障情况下丢失少量数据。如果编写这样的程序，可以通过使用异步发送来提高吞吐量（甚至在使用PERSISTENT 传送模式的情况下）。

# ActiveMQ在CDC中的应用

1在CDC中，通过日志挖掘挖掘出源端操作的SQL，通过MQ进行通信，将SQL传送到目标端。



这里yinqing\_blu相当于生产者，yinqing\_blu\_Filter和yinqing\_blu\_Filter是消费者

2.ActiveMQ的启动



3.监控页面的查看 <http://127.0.0.1:8161/admin/queues.jsp>



Number Of Consumers：消费者 这个是消费者端的消费者数量
Number Of Pending Messages：等待消费的消息 这个是当前未出队列的数量。可以理解为总接收数-总出队列数
Messages Enque[ued](http://www.haogongju.net/tag/ued) ：进入队列的消息，进入队列的总数量,包括出队列的。 这个数量只增不减

Messages Dequeued ：出了队列的消息，可以理解为是消费这消费掉的数量

这个要分两种情况理解
在queues里它和进入队列的总数量相等(因为一个消息只会被成功消费一次),如果暂时不等是因为消费者还没来得及消费。
在 topics里 它因为多消费者从而导致数量会比入队列数高。