

消息队列 RocketMQ

控制台使用指南

控制台使用指南

消息查询

消息队列 RocketMQ 控制台提供几种消息查询方式用于问题排查。

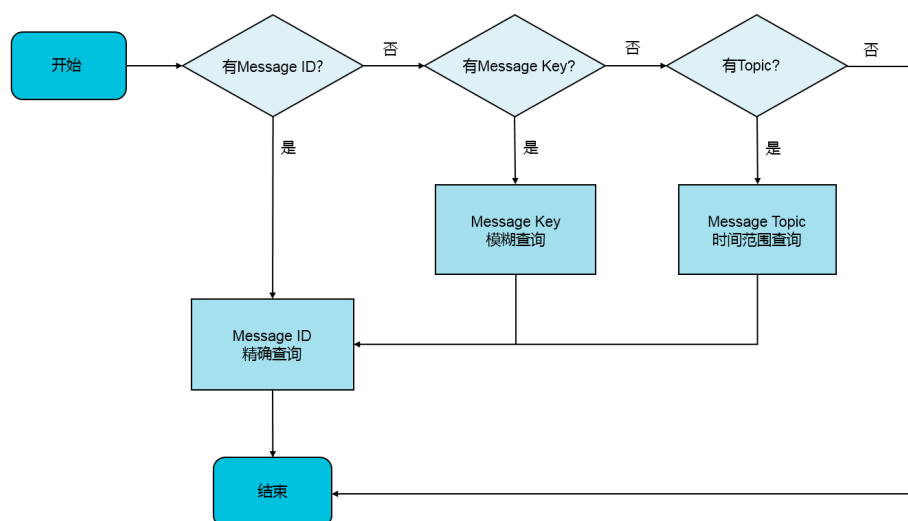
适用场景

消息队列 RocketMQ 提供了三种消息查询的方式，分别是按 Message ID，Message Key 以及 Topic 查询。

三种方式的特点和对比如下。

查询方式	查询条件	查询类别	说明
按 Message ID 查询	Topic+Message ID	精确查询	根据 Topic 和 Message ID 可以精确定位任意一条消息，获取消息的属性
按 Message Key 查询	Topic+Message Key	模糊查询	根据 Topic 和 Message Key 可以匹配到包含指定 Key 的最近 64 条消息,建议消息生产方为每条消息设置尽可能唯一的 Key，以确保相同的 Key 的消息不会超过 64 条，否则消息会漏查
按 Topic 查询	Topic+时间段	范围查询	根据 Topic 和时间范围，批量获取符合条件的所有消息，查询量大，不易匹配

建议查询过程：



按 Message ID 查询

按 Message ID 查询消息属于精确查询，您输入 Topic 和 Message ID 即可精确查询到任意一条消息。因此，为了尽可能精确地查询，建议发送方调用 Producer 发送消息时，在发送消息成功后将 Message ID 信息打印到日志中，方便问题排查。

以 Java SDK 为例，获取 Message ID 的方法如下：

```
SendResult sendResult = producer.send(msg);
String msgId = sendResult.getMessageId();
```

查询结果说明

查询到消息后，您可以在消息详情页面看到每条消息的基本属性，消息体下载链接以及消息的消费状态。

其中，投递状态是消息队列 RocketMQ 根据各个 Group ID 的消费进度计算出的结果，投递状态的信息参考以下表格。

注意：投递状态仅仅是根据消费进度估算的结果，如果需要详细的消费信息，请使用消息轨迹工具查询。消息轨迹可以展示该条消息的完整链路，具体请参见[查询消息轨迹](#)。

投递状态表

投递状态	可能的原因
已订阅，并且消息至少已被消费一次	该 Group ID 已经正常消费过这条消息。
已订阅，但消息被过滤表达式过滤，请查看 Tag	该消息的 Tag 不符合消费方的订阅关系，消息被过滤。您可以进入控制台中的 消费者管理 ，然后单击该 Group ID 的操作列的 消费者状态 查询其订阅关系。
已订阅，但消息未被消费	该 Group ID 订阅了该消息，但还未消费。有可能

	是消费过慢，或者消费出现异常导致阻塞。
已订阅，但 Group ID 当前不在线，请通过消息轨迹功能进行精确查询	该 Group ID 订阅了该消息，但是不在线。请检查消费者端应用不在线的原因。
未知异常	出现未收录的异常情况。请提工单进行解决。

消费验证

消息队列 RocketMQ 提供了消费验证功能，该功能可以将指定消息推送给指定的在线客户端，以检测客户端消费该消息的逻辑和结果是否符合预期。

注意：消费验证功能仅仅是用于验证客户端的消费逻辑是否正常，并不会影响正常的收消息流程，因此消息的消费状态等信息在消费验证后并不会改变。

按 Message Key 查询

按 Message Key 查询消息的原理是，消息队列 RocketMQ 根据您设置的 Message Key 建立消息的索引信息，当您输入 Key 进行查询时，消息队列 RocketMQ 根据该索引即可匹配相关的消息返回。

注意：

- 按 Message Key 查询的条件是用户设置的 Message Key 属性。
- 按 Message Key 查询仅仅返回符合条件的最近的 64 条消息，因此建议您尽可能保证设置的 Key 是唯一的，并具有业务区分度。

设置 Message Key 的方法如下：

```
Message msg = new Message("Topic", "*", "Hello MQ".getBytes());  
/**  
 * 对每条消息设置其检索的 Key，该 Key 值代表消息的业务关键属性，请尽可能全局唯一。  
 * 以方便您在无法正常收到消息情况下，可通过消息队列 RocketMQ 控制台查询消息。不设置也不会影响消息正常收发。  
 */  
msg.setKey("TestKey"+System.currentTimeMillis());
```

按 Topic 查询

按 Topic 查询一般用在前两种查询条件都无法获得的情况下，根据 Topic 和消息的发送时间范围，批量获取该时间范围内的所有消息，然后再找到关心的数据。

注意：

- 按 Topic 查询属于范围查询，获取 Topic 下符合时间条件的所有消息，消息量大，建议尽量缩短查询区间。
- 按 Topic 查询数据量大，采用分页展示。

消息轨迹

消息轨迹简介

本文档介绍消息队列 RocketMQ 的消息轨迹的基本原理、使用场景、以及使用案例。

注意：目前消息队列 RocketMQ 支持 TCP 协议下的消息轨迹查询。

基本原理

定义：消息轨迹指的是一条消息从生产方发出到消费方消费处理，整个过程中的各个相关节点的时间地点等数据汇聚而成的完整链路信息。

原理：消息队列 RocketMQ 系统中，一条消息的完整链路包含生产方、服务方、消费方三个角色，每个角色处理消息的过程中都会在轨迹链路中增加相关的信息，将这些信息汇聚即可获取任意消息当前的状态，从而为生产环境中的问题排查提供强有力的数据支持。

消息轨迹的数据包含：

生产方信息	消费方信息	服务方信息
生产客户端信息	消费客户端信息	消息所属的 Topic
发送时间	投递时间，投递轮次	消息存储的地域位置
发送成功与否	消费成功与否	消息的 Key
发送耗时	消费耗时	消息的 Tag

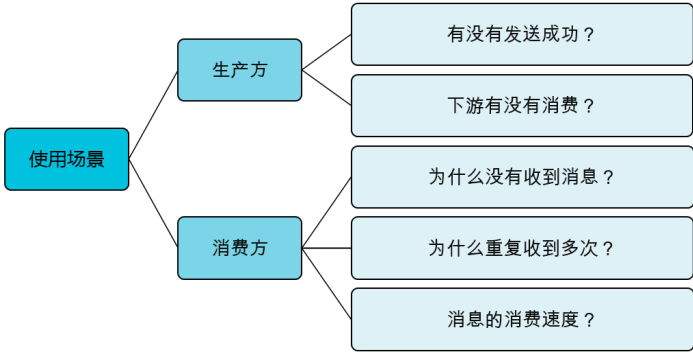
消息轨迹查询规则：

消息类型	可以查询的时间	查询说明
普通消息	消息发送后	消息发送之后有发送轨迹，没消费前显示“尚未消费”。消费后会展示投递和消费信息。
顺序消息	消息发送后	消息发送之后有发送轨迹，没消费前显示“尚未消费”。消费后会展示投递和消费信息。
定时/延时消息	当前系统时间到达消息指定消费的时间后	当前系统时间没有到达指定消费的时间，轨迹可以查询到，但是消息查询不到。
事务消息	消息发送后	消息发送之后有发送轨迹。事务未提交之前，轨迹可以查询到

		，但是消息查询不到。
--	--	------------

使用场景

在生产环境的消息收发不符合预期时可以使用消息轨迹工具排查问题。通过消息的属性（ Message ID、 Message Key、 Topic ）搜索相关的消息轨迹，找到消息的实际收发状态，帮助诊断问题。



示例：

假设您根据业务日志里的信息判断某条消息一直没有收到，则可以参考以下步骤，利用消息轨迹来排查消息队列 RocketMQ 的问题。

收集怀疑的消息的信息， Message ID ， Message Key ， Topic 以及大概的发送时间。

进入消息队列 RocketMQ 控制台，根据已有的信息创建查询任务，查询相关的消息的轨迹。

查看结果并分析判断原因。如果轨迹显示尚未消费，则可以去**消费者管理**页面查询，确认是否有堆积导致消息尚未消费。

如果发现已经消费，请根据消费端的信息，找到对应的客户端机器和时间，登录查看相关日志。

查询消息轨迹

消息轨迹的使用不会增加额外的接入成本，仅仅需要确保客户端 SDK 版本支持该特性。正常收发消息后，根据消息的属性在消息队列 RocketMQ 控制台上查询即可。

接入条件

消息轨迹功能目前支持 Java 客户端（1.2.7 版本及以上），C++ 和 .NET 客户端。详细的客户端信息，请参考控制台给出的升级提示，获取最新的 SDK。

创建轨迹查询任务

在消息队列 RocketMQ 控制台的左侧菜单栏选择**消息轨迹**，然后单击右上角**创建查询任务**按钮。

消息轨迹查询功能支持三种查询方式。请选择一种查询方式，然后指定相应的查询条件进行查询。关于三种查询方式的具体说明如下：

- 按 Message ID 查询：需要输入消息的唯一 Message ID，Topic 名称以及消息的大致发送时间（一个时间范围）。
- 按 Message Key 查询：需要输入消息的 Message Key 和 Topic 以及大致发送时间（一个时间范围）。该方式适用于没有记录 Message ID，但记录了 Message Key 的场景。
- 按 Topic 查询：仅需要输入 Topic 和消息大致发送时间（一个时间范围），进行批量查询。该方式适用于没有上述 Message ID 和 Message Key，而且消息量比较小的场景。

注意：

- 查询时，尽可能设置最为精确的时间区间，以便缩小查询范围，提高速度。
- 按 Message ID 查询属于精确查询，速度快，精确匹配，推荐使用。
- 按 Message Key 查询属于模糊查询，仅适用于您没有记录 Message ID 但是设置了 Message Key，同时 Message Key 具有区分度的情况。按 Message Key 查询最多查询 1000 条轨迹。
- 按 Topic 查询属于范围查询，不推荐使用，因为时间范围内消息很多，不具备区分度。

管理查询任务

您创建查询任务后，消息队列 RocketMQ 后台会异步执行这个查询任务，并将任务状态反馈到**查询任务**页面。查询结束时，**任务状态**会显示**查询完成**，否则显示**查询中**。

您可以根据任务的状态可以选择查看轨迹，或者删除查询任务。

查看消息轨迹

完成查询后，单击**详情**可查看消息轨迹。如果发现没有结果，请参考弹窗中的链接，排查原因。

如果查询到轨迹信息，可以看到轨迹的简要信息，主要是消息本身的属性以及接收状态的信息。

查看轨迹链路图

单击**详情**后，再在展开的页面单击**查看轨迹**即可查看完整的链路图。

消息链路图包含3个部分：

- 生产者信息
- Topic 信息

消费者信息

对于 Message Key 和 Topic 查询方式，如果匹配到多条轨迹，可以进行上下翻页，查看比对轨迹数据。

消息轨迹名词解释

消息轨迹链路图中的部分关键字段的解释如下。

链路部分	字段	说明
生产者	发送时间	消息从发送端发送时的客户端时间戳
生产者	发送耗时	发送端调用 send 方法发送消息的毫秒耗时
Topic	Region	消息存储的地域
消费者	耗时	消息推送到客户端之后执行 consumeMessage 方法的耗时
消费者	投递时间	客户端执行 consumeMessage 方法开始消费消息时的时间戳

状态信息说明

在消息轨迹的任务列表页，点开某任务的详情会显示消息的发送状态和消费状态；而消息轨迹链路图中的生产者和消费者信息区域也会分别显示消息的发送和消费状态。这些状态字段及其解释如下。

发送/消费状态	状态字段	说明
发送状态	发送成功	消息发送成功，服务端已经存储成功
发送状态	发送失败	消息发送失败，服务端没有存储消息，需要重试
发送状态	消息定时中	该消息是定时或者延时消息，且尚未到达投递时间
发送状态	事务未提交	该消息是事务消息，且尚未提交状态
发送状态	事务回滚	该消息是事务消息，并且已经回滚
消费状态	全部成功	该消息的所有投递都已成功消费

消费状态	部分成功	该消息投递中存在消费失败的情况，或消费失败并重试成功的情况
消费状态	全部失败	该消息的所有投递都消费失败
消费状态	尚未消费	该消息尚未投递给任何消费方
消费状态	消费结果未返回	消费消息的方法尚未返回结果，或者被中断，导致本次消费结果未传回服务端
消费状态	消费成功	该消息已被成功消费
消费状态	消费失败	消费消息的方法主动返回失败标志，或者是消费方法抛异常

如果对消息轨迹的查询结果有疑问，请查看《常见问题》中的消息轨迹一节。

查看消费者状态

简介

消息队列 RocketMQ 控制台为您提供了查看消费者状态的功能，帮助您在消息消费出现异常时进行问题排查。通过该功能，您可以查看每个 Group ID 代表的消费者实例群组的综合信息及其底下单个消费者实例的信息，包括在线状态、订阅关系、消费 TPS、消息堆积量和线程堆栈信息等，用以初步检查消息消费异常的原因。

通常消息消费异常发生的原因会比较复杂，单纯依靠在控制台查看消费者状态还不足以排查问题，这时就需要结合日志分析和业务场景分析等进行进一步排查。

关于与消费者状态查询相关的典型故障排查场景，请参见下文的**典型使用场景**。

典型使用场景

消费者状态查询经常被应用于以下两个故障排查场景。

订阅关系不一致

查询：在**消费者状态**页面，看到 Group ID 的**订阅关系是否一致**一栏显示否。

处理：关于订阅关系不一致的处理方法，请参见**收到订阅关系不一致告警后怎么办？**。

堆积告警

查询：在**消费者状态**页面，看到 Group ID 的**实时消息堆积量**一栏的值较大。

处理：关于消费堆积的告警处理方法，请参见**收到堆积告警后怎么办？**。

操作步骤

查看 Group ID 综合信息

请按照以下步骤查看 Group ID 所代表的消费者实例集群的综合信息：

登录消息队列 RocketMQ 控制台。

在左侧导航栏选择**Group 管理**。

在**Group 管理**页面，找到要查看的 Group ID，单击其右侧**操作**列中的**消费者状态**。

字段说明

该页面的主要字段的说明如下：

字段	说明
在线状态	只要该 Group ID 下有一个消费者实例在线，就显示 在线 ，此时您可以看到页面下方的 连接信息 区域显示了在线的消费者实例信息；若该 Group ID 下所有消费者实例都不在线，则显示 离线 。
订阅关系是否一致	显示是否该 Group ID 下多个消费者实例的订阅关系一致。关于订阅关系一致的概念，请参见 订阅关系一致 。
实时消费速度	该 Group ID 下消费者实例群组接收消息的总 TPS，单位为“条/秒”。
实时消息堆积量	该 Group ID 下消费者实例群组的未消费消息的总量。
最近消费时间	该 Group ID 下消费者实例群组最近一次消费消息的时间。
消息延迟时间	该 Group ID 下消费者实例群组最早的一条未消费消息的生产时间与当前时间的差值。

查看 Group ID 下单个消费端信息

若此时该 Group ID 的在线状态为**在线**，则在**连接信息**区域您可以查看此时在线的每个消费者实例的具体信息

，包括客户端 ID、宿主机 IP/公网 IP、当前进程 ID 和消息堆积量等。

查看更多详细信息

除了上述的信息，若您还想查看关于某个消费者实例的更多详细信息，可以在其右侧的**详细说明列**，单击**详细信息**。

此时出现的页面如下图所示，您可以查看该消费者实例的消费线程数、启动时间、订阅关系和消息消费的统计数据等。

消费者状态

连接状态

项目	值
消息类型	PUSH
消费线程数	20
消费者启动时间	2018年10月22日 16:16:29
最近消费时间	2018年10月22日 16:16:41
订阅信息	无

订阅关系

序号	Topic	Tag
1	kingpin_test_1	-

消息统计

Topic	业务处理耗时（毫秒级）	消息成功数(条)	消息失败数(条)	消息成功数	消息堆积量
kingpin_test_1	0	0	0	0	0

清除取消

查看线程堆栈信息

若您想要查看某消费者实例当前进程的堆栈信息，在要查看堆栈信息的客户端的右侧，可以在其右侧的**堆栈信息列**，单击**堆栈信息**，展现的信息如下图所示：

消费者状态

Java Stack

Thread	Stack
RebalanceService	TID: 17 STATE: TIMED_WAITING run-misc: Unstable (parkNative Method) java.util.concurrent.locks.LockSupport.parkNanos(LockSupport.java:215) java.util.concurrent.locks.AbstractQueuedSynchronizer.doAcquireSharedName(AbstractQueuedSynchronizer.java:1037) java.util.concurrent.locks.AbstractQueuedSynchronizer.tryAcquireSharedName(AbstractQueuedSynchronizer.java:1128) com.alipay.openplatform.shade.com.alibaba.rocketmq.common.CountDownLatch.await(CountDownLatch.java:114) com.alipay.openplatform.shade.com.alibaba.rocketmq.common.ServiceThread.waitForRunning(ServiceThread.java:115) java.lang.Thread.run(Thread.java:745)
ConsumeMessageThread_1	TID: 45 STATE: WAITING run-misc: Unstable (parkNative Method) java.util.concurrent.locks.LockSupport.park(LockSupport.java:175) java.util.concurrent.locks.AbstractQueuedSynchronizer\$ConditionObject.await(AbstractQueuedSynchronizer.java:2059) java.util.concurrent.ThreadPoolExecutor.getTask(ThreadPoolExecutor.java:1074) java.util.concurrent.ThreadPoolExecutor.run(ThreadPoolExecutor.java:1134) java.util.concurrent.ThreadPoolExecutor\$Worker.run(ThreadPoolExecutor.java:634) java.lang.Thread.run(Thread.java:745)
RebalanceService	TID: 32 STATE: TIMED_WAITING run-misc: Unstable (parkNative Method) java.util.concurrent.locks.LockSupport.parkNanos(LockSupport.java:215) java.util.concurrent.locks.AbstractQueuedSynchronizer.doAcquireSharedName(AbstractQueuedSynchronizer.java:1037) java.util.concurrent.locks.AbstractQueuedSynchronizer.tryAcquireSharedName(AbstractQueuedSynchronizer.java:1128) com.alipay.openplatform.shade.com.alibaba.rocketmq.common.CountDownLatch.await(CountDownLatch.java:114) com.alipay.openplatform.shade.com.alibaba.rocketmq.common.ServiceThread.waitForRunning(ServiceThread.java:115) com.alipay.openplatform.shade.com.alibaba.rocketmq.client.impl.consumer.RebalanceService.run(RebalanceService.java:40) java.lang.Thread.run(Thread.java:745)

清除取消

重置消费位点

如果想跳过堆积的消息，即不想消费这部分消息，或者只想消费某个时间点之后的消息（不论之前是否消费过这些消息），您可以使用重置消息的消费位点。具体操作步骤如下：

在消息队列 RocketMQ 控制台的左侧导航栏，选择**Group 管理**。

找到需要重置消费位点的 Group ID 和针对的 Topic，在该行的操作列中单击**更多功能**>**重置消费位点**。

在**重置消费位点**对话框中，根据需求选择“清除所有堆积消息，从最新位点开始消费”或“按时间段进行消费位点重置”。关于这两个选项的具体说明如下：

清除所有堆积消息，从最新位点开始消费：若选择此项，该 Group ID 在消费该 Topic 下的消息时会跳过当前堆积（未被消费）的所有消息，从这之后发送的最新消息开始消费。

注意：对于程序返回“reconsumeLater”，即走重试流程的这部分消息来说，清除无效。

按时间点进行消费位点重置：点击该选项后会出现时间点选择的控件。请选择一个时间点，这个时间点之后发送的消息才会被消费。

注意：

可选时间范围中的起始和终止时间分别是该 Topic 中储存的最早的和最晚的一条消息的生产时间。

若选择的时间点超出标明的可选时间范围，系统会默认按以下逻辑进行处理：

若选择的时间点早于可选时间范围的起始时间：将消费位点重置到可选时间范围的起始时间。

若选择的时间点晚于可选时间范围的终止时间：将消费位点重置到可选时间范围的终止时间。

单击**确定**执行消费位点重置。

说明：目前不支持指定 Message ID，Message Key 和 Tag 来对消息的消费位点进行重置。

死信队列

简介

死信队列用于处理无法被正常消费的消息。当一条消息初次消费失败，消息队列 RocketMQ 会自动进行消息重

试；达到最大重试次数后，若消费依然失败，则表明消费者在正常情况下无法正确地消费该消息，此时，消息队列 RocketMQ 不会立刻将消息丢弃，而是将其发送到该消费者对应的特殊队列中。

消息队列 RocketMQ 将这种正常情况下无法被消费的消息称为**死信消息**（Dead-Letter Message），将存储死信消息的特殊队列称为**死信队列**（Dead-Letter Queue）。

死信消息具有以下特性：

- 不会再被消费者正常消费。
- 有效期与正常消息相同，均为 3 天，3 天后会被自动删除。因此，请在死信产生后的 3 天内及时处理。

死信队列具有以下特性：

- 一个死信队列对应一个 Group ID，而不是对应一个单个的消费者实例。
- 当一个 Group ID 未产生死信时，消息队列 RocketMQ 不会为其创建相应的死信队列。
- 一个死信队列包含了对应 Group ID 产生的所有死信消息，不论该消息属于哪个 Topic。

消息队列 RocketMQ 控制台提供对死信消息的查询、导出和重发的功能。

查询死信消息

消息队列 RocketMQ 提供了以下两种查询死信消息的方式：

- 按 Group ID 查询
- 按 Message ID 查询

这两种查询方式的对比如下表所示：

查询方式	查询条件	查询类别	说明
按 Group ID 查询	Group ID+时间段	范围查询	根据 Group ID 和时间范围，批量获取符合条件的所有消息；查询量大，不易匹配。
按 Message ID 查询	Group ID+Message ID	精确查询	根据 Group ID 和 Message ID 可以精确定位任意一条消息。

按 Consumer ID 查询

您可以根据 Group ID 和死信消息产生的时间范围，批量查询该 Group ID 在某段时间内产生的所有死信消息。

注意：死信消息产生的时间是指一条消息在投递重试达到最大次数后被发送到死信队列的时间。

具体操作步骤如下：

1. 在消息队列 RocketMQ 控制台的左侧导航栏，选择**死信队列**。

2. 在**死信队列**页面，单击**按 Group ID 查询**页签。
3. 在 Group ID 的输入框中，输入关键词搜索想要查找的 Group ID，然后单击选中，或直接在下拉列表中选择想要查找的 Group ID。
4. 单击时间选择框，选择想要查询的时间范围的起止日期和具体时间点。
5. 单击**搜索**按钮。页面中会显示所有符合以上条件的死信消息。
6. 找到想要查看的死信消息，单击其**操作**列中的**查看详情**查看该消息的详细信息，包括其基本属性、消息体的下载链接、消息轨迹和投递状态。

按 Message ID 查询

按 Message ID 查询消息属于精确查询。您可以根据 Group ID 与 Message ID 精确查询到任意一条消息。

具体操作步骤如下：

1. 在消息队列 RocketMQ 控制台的左侧导航栏，选择**死信队列**。
2. 在**死信队列**页面，单击**按 Message ID 查询**页签。
3. 在 Group ID 的输入框中，输入关键词搜索想要查找的 Group ID，然后单击选中，或直接在下拉列表中选择想要查找的 Group ID。
4. 在 Message ID 的输入框中输入想要查找的消息的 Message ID。
5. 单击**搜索**按钮。页面中会显示所有符合以上条件的死信消息。
6. 找到想要查看的死信消息，单击其**操作**列中的**查看详情**查看该消息的详细信息，包括其基本属性、消息体的下载链接、消息轨迹和投递状态。

导出死信消息

若您暂时无法处理死信消息，可以在消息队列 RocketMQ 控制台上将其导出，以免超过有效期。

消息队列 RocketMQ 控制台提供对死信消息的**单条导出**和**批量导出**功能。导出的文件格式为 CSV。

导出的消息内容如下表所示：

消息字段	字段含义
topic	消息所属的 Topic
msgId	消息的 ID
bornHost	消息产生的地址
bornTimestamp	消息产生的时间
storeTimestamp	死信消息产生的时间
reconsumeTimes	消费失败的次数
properties	消息属性；JSON 格式
body	消息体；base64 编码格式
bodyCRC	消息体 CRC

单条导出

在消息队列 RocketMQ 控制台按任意方式查询到死信消息后，在某条死信消息的**操作列**，单击**导出**，导出该条死信消息。

批量导出

在消息队列 RocketMQ 控制台按 Group ID 查询到死信消息后，勾选目标死信消息，然后单击页面下方的**批量导出**按钮，导出所有勾选的死信消息。

重新发送死信消息

一条消息进入死信队列，意味着某些因素导致消费者无法正常消费该消息，因此，通常需要您对其进行特殊处理。排查可疑因素并解决问题后，您可以在消息队列 RocketMQ 控制台重新发送该消息，让消费者重新消费一次。

注意：死信消息被重新发送后，不会在死信队列中被立即删除。

单条重发

在消息队列 RocketMQ 控制台按任意方式查询到死信消息后，在某条死信消息的**操作列**，单击**重新发送**，重发该条死信消息。

批量重发

在消息队列 RocketMQ 控制台按 Group ID 查询到死信消息后，勾选目标死信消息，然后单击页面下方的**批量重发**按钮，重新发送所有勾选的死信消息。

全球消息路由

简介

全球消息路由常用于不同地域之间的消息同步，保证地域之间的数据一致性。消息队列 RocketMQ 的消息路由功能依托阿里云优质基础设施实现的高速通道专线，可以高效地实现国内外不同地域之间的消息同步复制。

通过消息队列 RocketMQ 消息路由，您可以指定需要进行消息同步的源地域和 Topic，以及目标地域和 Topic，并配置消息过滤规则（例如通过特定的 Tag 过滤）及指定消息同步的起始位点，即可将某个地域的某个 Topic 的消息按条件同步至另一个地域的指定 Topic。

基本概念

消息队列 RocketMQ 消息路由涉及的基本概念如下表所示：

概念	解释
源	需要同步的消息的来源，例如源地域，即消息来源的地域。
目标	消息同步的目的地，例如目标地域，即消息将被同步到的地域。
起始同步位点	起始同步位点是指从源 Topic 中的消息队列的哪个位置开始进行消息同步，即从这个位置之后进入队列的消息都会被同步到目标 Topic。
过滤规则	将需要同步的消息进行过滤的规则（例如通过 Tag 过滤），按照该规则过滤后的消息才会被同步。
延迟	考虑到需要同步的消息量以及网络带宽瓶颈，消息同步可能会产生一定的时间延迟，即消息到达目标 Topic 的时间会晚于该消息进入源 Topic 的时间。

要使用消息队列 RocketMQ 消息路由，您可以登录消息队列 RocketMQ 控制台，根据业务需求创建跨地域的消息同步任务并进行任务的追踪和管理。

创建任务

请按照以下步骤创建消息同步的任务：

登录 消息队列 RocketMQ 控制台。

在左侧导航栏，单击**消息路由**。

在**消息路由**页面，单击**创建任务**按钮。

在**创建任务**页面，填写以下信息：

在**消息源**区域，选择消息的来源地域和 Topic。

在**消息目标**区域，选择消息将被同步到的地域和 Topic。

在**起始同步位点**一栏，选择消息同步的起始位点，即从源 Topic 中的消息队列的哪个位置的消息开始进行同步，该位置之前的消息都会被跳过。具体选项说明如下：

选项	说明	是否支持
最大位点	队列当前最新的一条消	支持

	息	
最小位点	队列当前最老的一条消息	暂不开放
时间戳	队列中指定的某个具体的时间点位置	暂不开放
消费组位点	指定 Group ID 的消费位点，即其消费进度所达到的时间点位置	暂不开放

在**过滤规则**一栏，输入消息的过滤条件，例如通过某些 Tag 进行过滤，具体方法可参考消息过滤。

在**描述**一栏，输入对该同步任务的具体描述或备注等。

单击**确定**按钮。

任务创建完成后，您可以在**消息路由**页面的任务列表中看到刚才创建的任务。

管理任务

您可以在**消息路由**页面对所有创建的任务进行启动、停止、查看和删除的操作。

搜索任务

在**消息路由**页面，在**源**和**目标**两栏分别输入源地域、源 Topic、目标地域、目标 Topic 后，单击**搜索**按钮，即可查询到符合搜索条件的任务。

启停任务

在**消息路由**页面的任务列表中，找到需要启停的任务，在其**操作**列单击**开始**或**停止**，启动或停止该任务。

查看任务详情

在**消息路由**页面的任务列表中，找到需要查看的任务，在其**操作**列单击**查看详情**，即可查看该任务的属性信息、执行情况和监控指标等。

消息详情页的关键字段的说明如下：

信息项	说明
消息同步 TPS	每分钟统计一次消息同步的 TPS 值；TPS 即平均每秒传输的消息数量
消息延迟	当前同步到的位点和最新消息位点之间的时间差

消息堆积量	当前还没有来得及同步的消息数量
最新同步时间	最近一次消息同步发生的时间

编辑过滤规则

在**任务详情**区域的**过滤规则**一栏，单击编辑图标，即可对过滤规则进行修改。

注意：若要在此处编辑过滤规则，需先停止任务。

删除任务

在**消息路由**页面的任务列表中，找到需要删除的任务，在其**操作**列单击**删除**，即可删除该任务。

注意：删除任务前需要先停止该任务。任务被删除后将不会再搜索结果中展示。

查看订阅关系

在消息队列 RocketMQ 控制台，您可以实时查看 Group ID 和 Topic 之间的订阅关系，即某个 Topic 被哪些 Group ID 订阅了，以及某个 Group ID 订阅了哪些 Topic。

注意：只有在线状态的 Group ID 才能在订阅关系中显示。

查看 Topic 被哪些 Group ID 订阅

前提条件

订阅该 Topic 的 Group ID 至少有一个处于在线状态。

操作步骤

登录消息队列 RocketMQ 控制台。

在左侧导航栏，选择 **Topic 管理**。

在 **Topic 管理**页面找到目标 Topic，然后在其**操作**列单击**订阅关系**。

结果验证

在弹出的**订阅关系**对话框，除了看到该 Topic 的名称和最新一条消息的发送时间外，您还可以看到所有订阅该 Topic 的在线 Group ID 及其消费模式。

此外，您还可以单击任一 Group ID 的**操作列的消费者状态**，查看该 Group ID 的消息消费详情。

查看 Group ID 订阅的 Topic

前提条件

存在在线状态的 Group ID。

操作步骤

登录消息队列 RocketMQ 控制台。

在左侧导航栏，选择**Group 管理**。

在**Group 管理**页面找到目标 Group ID，在其**操作列**单击**订阅关系**。

结果验证

在弹出的**订阅关系**对话框，除了看到该 Group ID 的名称、在线状态和消费模式外，您还可以看到该 Group ID 订阅的所有 Topic，以及这些 Topic 的消息过滤规则。

资源报表

资源报表是消息队列 RocketMQ 提供的消息生产和消费的统计数据查询功能，用户通过资源报表可以对某个 Topic 在一段时间范围内从 Broker 接收的消息的总量或者 TPS 进行查询，也可以对某个 Topic 在一个时间段内投递给某个消费者 Group ID 的消息的总量或 TPS 进行查询。

查看消息生产数据

可供查看的消息生产数据是某个 Topic 在一个时间段内从 Broker 接收的消息的总量或者TPS。

具体操作步骤如下：

登录消息队列 RocketMQ 控制台，然后在左侧导航栏，选择**资源报表**。

在**资源报表**页面，选择**消息生产**标签页。

在 **Topic** 一栏，选择您要查询的 Topic。

指定其他信息，然后单击**搜索**。

字段说明：

- **采集类型**：分为总量和 TPS。总量提供每个周期内 Topic 接收的消息总量，TPS 提供每个周期内该 Topic 接收消息的平均 TPS。
- **采集周期**：包括 1 分钟、10 分钟、30 分钟、1 小时。采集周期决定了数据采集的时间间隔，周期越短，采集点越密集，消息生产数据越详细。
- **时间范围**：消息队列 RocketMQ 最多可以提供最近三天之内的消息的生产查询。

查询结果以图表的形式返回。

查看消息消费数据

可供查看的消息消费数据是某个 Topic 在一个时间段内投递给某个消费者 Group ID 的消息的总量或 TPS。

具体操作步骤如下：

登录消息队列 RocketMQ 控制台。在消息队列 RocketMQ 控制台左侧导航栏，选择**资源报表**。

在**资源报表**页面，选择**消息消费**标签页。

在 **Group ID** 和 **Topic** 两栏，分别选择您要查询的 Group ID 和 Topic。

指定其他信息，然后单击**搜索**。

字段说明：

- **采集类型**：分为总量和 TPS。总量提供每个周期内该 Topic 投递给该 Group ID 的消息总量，TPS 提供每个周期内该 Topic 投递给该 Group ID 消息的平均 TPS。
- **采集周期**：包括 1 分钟、10 分钟、30 分钟、1 小时。采集周期决定了数据采集的时间间隔，周期越短，采集点越密集，消息消费数据越详细。
- **时间范围**：消息队列 RocketMQ 最多可以提供最近三天之内的消息的消费查询。

查询结果以图表的形式返回。

监控报警

MQ 提供了对您订阅的 Topic 下的消息的消费状态进行监控并且发送短信报警的功能，帮助您实时掌握消息消费状态并及时处理消费异常。

使用监控报警功能，首先需要在通过 Consumer ID 对 Topic 进行订阅，然后对相关监控项进行设置。当消费者在线，并且消息堆积数或者延迟时间超过阈值之后，MQ 会对报警接收人发送短信进行报警。

新建监控项

您可以通过新建监控项对一个 Consumer ID 的消息消费状态进行监控：

登录 MQ 控制台，在左侧菜单栏选择**监控报警**。

在**监控项管理**页面，单击右上角**新增监控项**。

在**新增监控项**对话框输入相关信息，单击**确定**。新建的监控项会出现在**监控项管理**页面。

字段说明：

- **Consumer ID**：要监控的 Consumer ID
- **Topic**：要监控的该 Consume ID 订阅的 Topic
- **堆积量阈值**：消费堆积的报警阈值。阈值的可填范围是 1 到 100,000,000。如果您选择的 Consumer ID 在消费对应的 Topic 时产生了消息堆积，并且堆积超过了报警阈值，MQ 就会给报警接收人发送短信通知。
- **消费延迟阈值**：消费延迟的值是指该 Consumer ID 最近一次消费该 Topic 消息的时间和该 Topic 最近一次发送消息的时间之间的差值。可设置的消费延迟阈值的最小值是 1 分钟。
- **报警时间**：时间精确到分钟，最大范围为 00:00-23:59。您只会在设置的接收时间范围内才会收到报警短信。
- **报警频率**：每隔 5 分钟、15 分钟或者 30 分钟。
- **报警接收人**：包括接收人的昵称和手机号码。昵称长度限制在 32 个字符以内。

管理监控项

在**监控项管理**页面，您可以对已经建好的监控项进行编辑、禁用、删除等操作。对于禁用的监控项，也可以在操作列单击**启用**重新启用。

资源管理优化说明

MQ 的资源包括 3 个部分：消息主题（Topic）、生产者（Producer ID）、消费者（Consumer ID），三者之间都是多对多的关系。

对于新用户来说，这三者之间的关系复杂而难以理解；对于中、大型企业客户来说，每创建一个 Topic，都要关联或者更新 Producer ID 和 Consumer ID，流程过于复杂。

因此，为进一步优化用户体验，降低您的接入门槛，将对 MQ 控制台的资源管理进行流程简化。

业务影响

在线业务

- 流程优化不会影响当前在线业务。您无需更改任何代码，继续保持当前的使用方式即可。

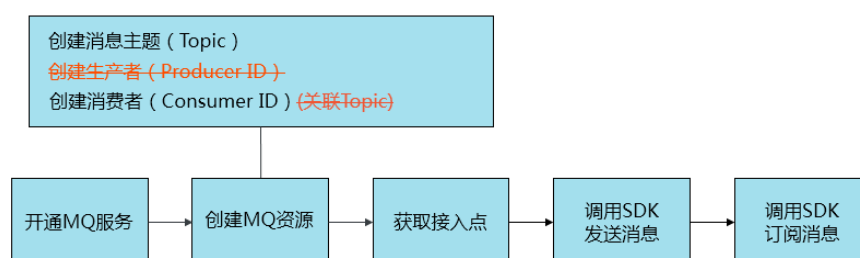
管控操作

- MQ 控制台不再展示**生产者管理**页面；API 也不再提供与 Producer ID 相关的操作。
- MQ 控制台的**消费者管理**页面不再展示 Consumer 与 Topic 之间的关联关系；API 也不再提供两者之间的关联数据。
- MQ 控制台新增查看订阅关系功能；API 提供相关操作。

流程优化

流程优化包括 3 个方面：去除 Producer ID 创建流程、优化 Consumer ID 创建流程、新增订阅关系查看功能。

优化后的操作流程



去除 Producer ID 创建流程

首先，Producer ID 与 Topic 之间是多对多的对应关系，每次使用都要进行大量的创建，创建过程复杂且过于

频繁；其次，Producer ID 在 MQ 中是一个非常弱化的概念，服务端并不会对 Producer ID 的使用进行约束与检查。

因此，MQ 控制台去除 Producer ID 的创建流程，您可以设置任意字符的 Producer ID 来发送消息。

注意：事务消息与其它类型消息（普通消息、顺序消息、定时/延时消息等），建议使用不同的 Producer ID 以示区分。

优化 Consumer ID 创建流程

首先，Consumer ID 与 Topic 之间是多对多的对应关系，每次使用都要进行大量的创建，创建过程复杂且过于频繁；其次，由于创建 Consumer ID 是需要关联指定 Topic，容易出现 MQ 控制台上配置的订阅关系与生产环境代码中的订阅关系不一致的情况。

因此，MQ 控制台简化 Consumer ID 的创建流程，您只需要创建 Consumer ID 而无需关联指定 Topic。

创建消费者

生产地域

服务需部署在与云平台相同地域的ECS上, 提供内网访问, 更加安全、稳定、可靠。

公网

服务可部署在任何地域, 提供公网访问, 可用于本地开发测试。

★ Consumer ID:

0/64

1. 以"CID_"或者"CID-"开头, 只能包含字母、数字、短横线 (-) 和下划线 (_);

2. 长度限制在 7-64 字节之间;

3. Consumer ID一旦创建, 将无法再修改。

★ 描述

请输入描述信息

确认

取消

新增订阅关系查看功能

MQ 控制台新增订阅关系查看功能，具体操作步骤请参见查看订阅关系。