bobby · 更新于 2022-11-16

上一节 12. rocketmq的... 14. go操作rock... 下一节

按照发送的特点分：

1. 同步发送

1. 同步发送，线程阻塞，投递completes阻塞结束
2. 如果发送失败，会在默认的超时时间3秒内进行重试，最多重试2次
3. 投递completes不代表投递成功，要check SendResult.sendStatus来判断是否投递成功
4. SendResult里面有发送状态的枚举：SendStatus，同步的消息投递有一个状态返回值的

代码块

```
    public enum SendStatus {
        SEND_OK,
        FLUSH_DISK_TIMEOUT,
        FLUSH_SLAVE_TIMEOUT,
        SLAVE_NOT_AVAILABLE,
    }
```

5. retry的实现原理:只有ack的SendStatus=SEND_OK才会停止retry

注意事项：发送同步消息且Ack为SEND_OK，只代表该消息成功的写入了MQ当中，并不代表该消息成功的被Consumer消费了

2. 异步发送

1. 异步调用的话，当前线程一定要等待异步线程回调结束再关闭producer啊，因为是异步的，不会阻塞,提前关闭producer会导致未回调链接就断开了
2. 异步消息不retry，投递失败回调onException()方法，只有同步消息才会retry,源码参考DefaultMQProducerImpl.class
3. 异步发送一般用于链路耗时较长，对 RT 响应时间较为敏感的业务场景，例如用户视频上传后通知启动转码服务，转码完成后通知推送转码结果等。

3. 单向发送

1. 消息不可靠，性能高，只负责往服务器发送一条消息，不会重试也不关心是否发送成功
2. 此方式发送消息的过程耗时非常短，一般在微秒级别

下表概括了三者的特点和主要区别。

发送方式	发送 TPS	发送结果反馈	可靠性
同步发送	快	有	不丢失
异步发送	快	有	不丢失
单向发送	最快	无	可能丢失

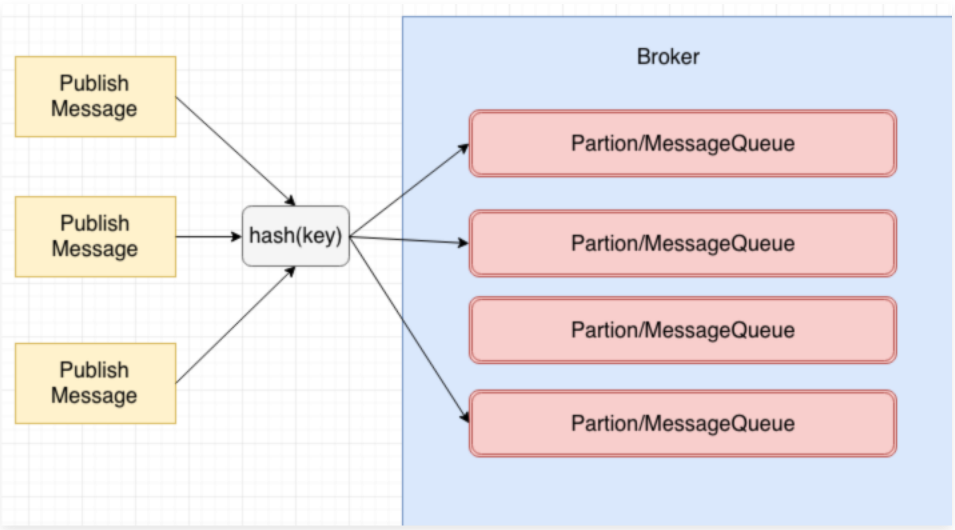
按照使用功能特点分：

1. 普通消息（订阅）

普通消息是我们在业务开发中用到的最多的消息类型，生产者需要关注消息发送成功即可，消费者消费到消息即可，不需要保证消息的顺序，所以消息可以大规模并发地发送和消费，吞吐量很高，适合大部分场景。

2. 顺序消息

顺序消息分为分区顺序消息和全局顺序消息，全局顺序消息比较容易理解，也就是哪条消息先进入，哪条消息就会先被消费，符合我们的FIFO，很多时候全局消息的实现代价很大，所以就出现了分区顺序消息。分区顺序消息的概念可以如下图所示：



我们通过对消息的key，进行hash，相同hash的消息会被分配到同一个分区里面，当然如果要做全局顺序消息，我们的分区只需要一个即可，所以全局顺序消息的代价是比较大的。

3. 延时消息 - 订单超时库存归还

延迟的机制是在 服务端实现的，也就是Broker收到了消息，但是经过一段时间以后才发送
服务器按照1-N定义了如下级别： “1s 5s 10s 30s 1m 2m 3m 4m 5m 6m 7m 8m 9m 10m 20m 30m 1h 2h”；若要发送定时消息，在应用层初始化Message消息对象之后，调用Message.setDelayTimeLevel(int level)方法来设置延迟级别，按照序列取相应的延迟级别，例如level=2，则延迟为5s

代码块
```
msg.setDelayTimeLevel(2);
 SendResult sendResult = producer.send(msg);
```

实现原理：

1. 发送消息的时候如果消息设置了DelayTimeLevel,那么该消息会被丢到ScheduleMessageService.SCHEDULE_TOPIC这个Topic里面
2. 根据DelayTimeLevel选择对应的queue
3. 再把真实的topic和queue信息封装起来，set到msg里面
4. 然后每个SCHEDULE_TOPIC_XXXX的每个DelayTimeLevelQueue，有定时任务去刷新，是否有待投递的消息
5. 每 10s 定时持久化发送进度

4. 事务消息

https://help.aliyun.com/document_detail/43348.html?spm=a2c4g.11186623.2.16.78ee6192siK1qV#concept-2047067

消息队列RocketMQ版提供的分布式事务消息适用于所有对数据最终一致性有强需求的场景。本文介绍消息队列RocketMQ版事务消息的概念、优势、典型场景、交互流程以及使用过程中的注意事项。

概念介绍

- 事务消息：消息队列RocketMQ版提供类似X或Open XA的分布式事务功能，通过消息队列RocketMQ版事务消息能达到分布式事务的最终一致。
- 半事务消息：暂不能投递的消息，发送方已经成功地将消息发送到了消息队列RocketMQ版服务端，但是服务端未收到生产者对该消息的二次确认，此时该消息被标记成“暂不能投递”状态，处于该种状态下的消息即半事务消息。
- 消息回查：由于网络闪断、生产者应用重启等原因，导致某条事务消息的二次确认丢失，消息队列RocketMQ版服务端通过扫描发现某条消息长期处于“半事务消息”时，需要主动向消息生产者询问该消息的最终状态（Commit或是Rollback），该询问过程即消息回查。

分布式事务消息的优势

消息队列RocketMQ版分布式事务消息不仅可以实现应用之间的解耦，又能保证数据的最终一致性。同时，传统的大事务可以被拆分为小事务，不仅能提升效率，还不会因为某一个关联应用的不可用导致整体回滚，从而最大限度保证核心系统的可用性。在极端情况下，如果关联的某一个应用始终无法处理成功，也只需对当前应用进行补偿或数据订正处理，而无需对整体业务进行回滚。

12. rocketmq的基本概念 ◂ 上一节　　下一节 ▸ 14. go操作rocketmq开发环境搭建

我要提出意见反馈

企业服务　网站地图　网站首页　关于我们　联系我们　讲师招募　帮助中心　意见反馈　代码托管

意见反馈　收藏教程　标记书签