细说 Kafka Partition 分区 原创

Partition（分区）是 Kafka 的核心角色，对于 Kafka 的存储结构、消息的生产消费方式都至关重要。

掌握好 Partition 就可以更快的理解 Kafka。本文会讲解 Partition 的概念、结构，以及行为方式。

一、Events, Streams, Topics

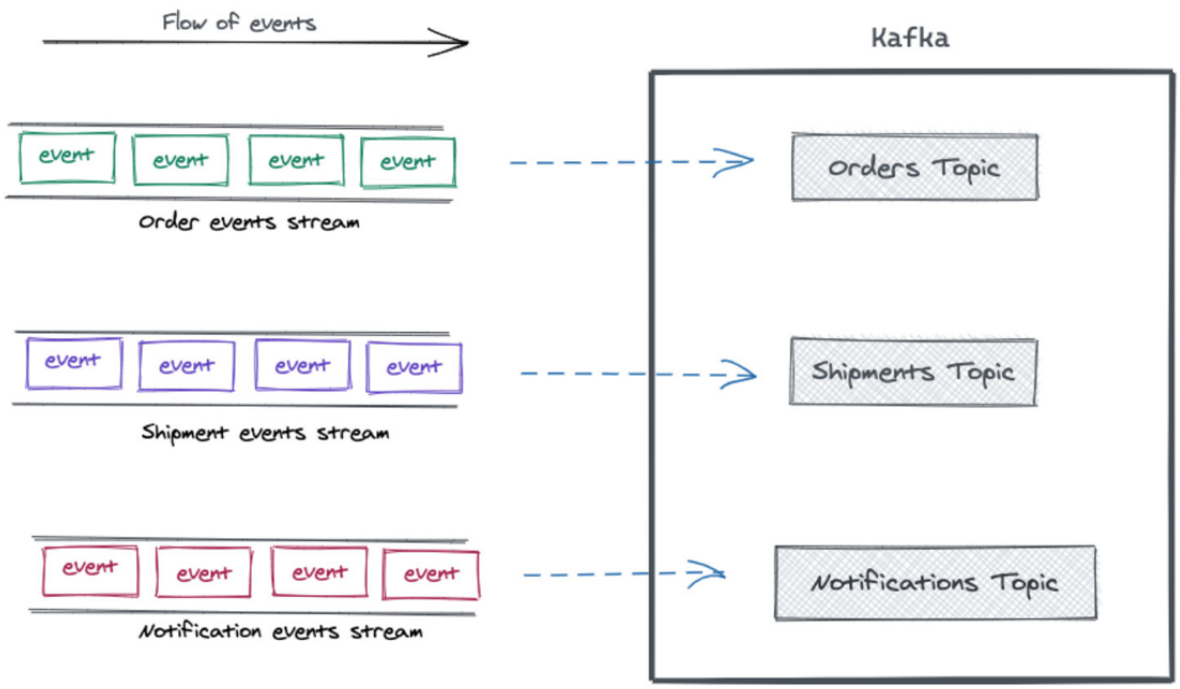
在深入 Partition 之前，我们先看几个更高层次的概念，以及它们与 Partition 的联系。

Event（事件）代表过去发生的一个事实。简单理解就是一条消息、一条记录。

Event 是不可变的，但是很活跃，经常从一个地方流向另一个地方。

Stream 事件流表示运动中的相关事件。

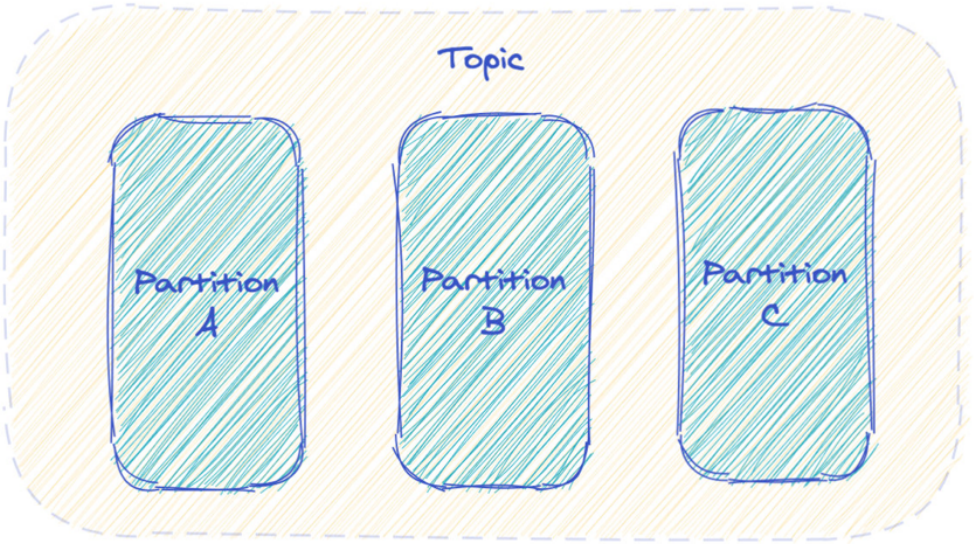
当一个事件流进入 Kafka 之后，它就成为了一个 **Topic** 主题。



所以，Topic 就是具体的事件流，也可以理解为一个 Topic 就是一个静止的 Stream。

Topic 把相关的 Event 组织在一起，并且保存。一个 Topic 就像数据库中的一张表。

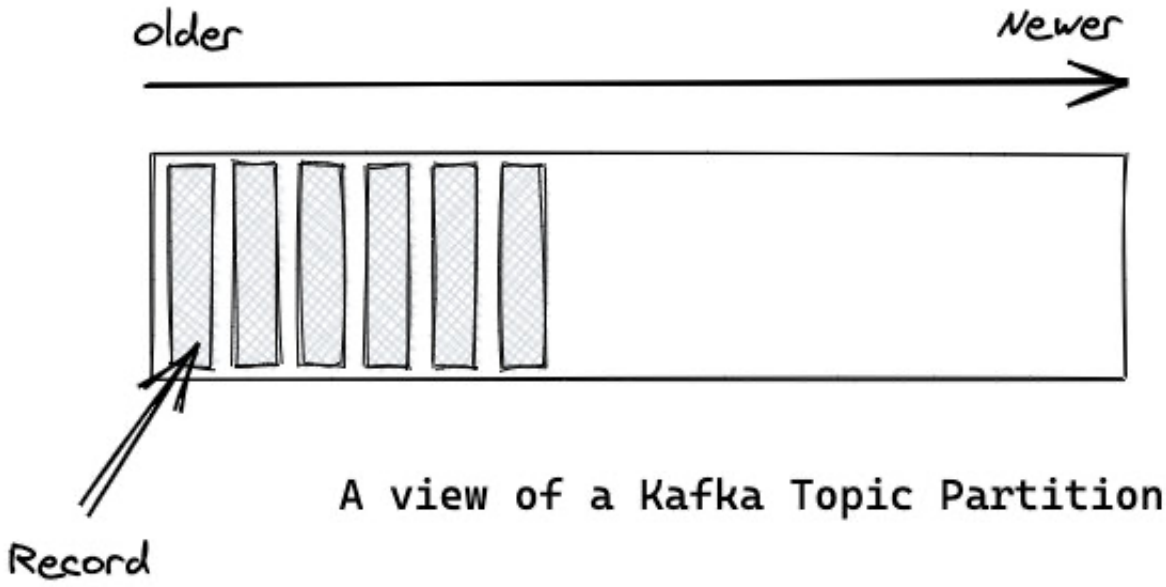
二、Partition 分区



Kafka 中 Topic 被分成多个 Partition 分区。

Topic 是一个逻辑概念，Partition 是最小的存储单元，掌握着一个 Topic 的部分数据。

每个 Partition 都是一个单独的 log 文件，每条记录都以追加的形式写入。



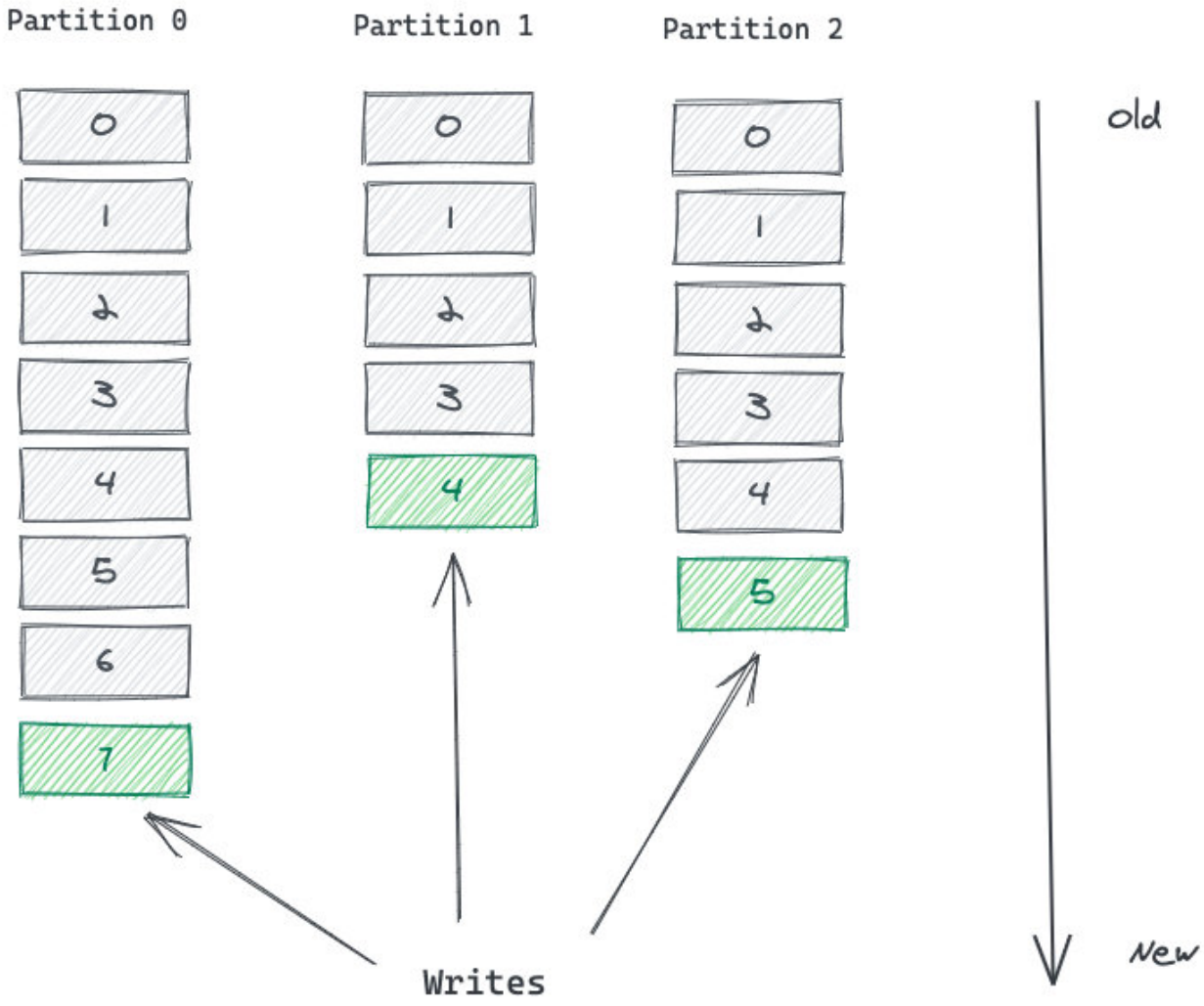
> Record（记录） 和 Message（消息）是一个概念。

三、Offsets（偏移量）和消息的顺序

Partition 中的每条记录都会被分配一个唯一的序号，称为 **Offset（偏移量）**。

Offset 是一个递增的、不可变的数字，由 Kafka 自动维护。

当一条记录写入 Partition 的时候，它就被追加到 log 文件的末尾，并被分配一个序号，作为 Offset。



如上图，这个 Topic 有 3 个 Partition 分区，向 Topic 发送消息的时候，实际上是被写入某一个 Partition，并赋予 Offset。

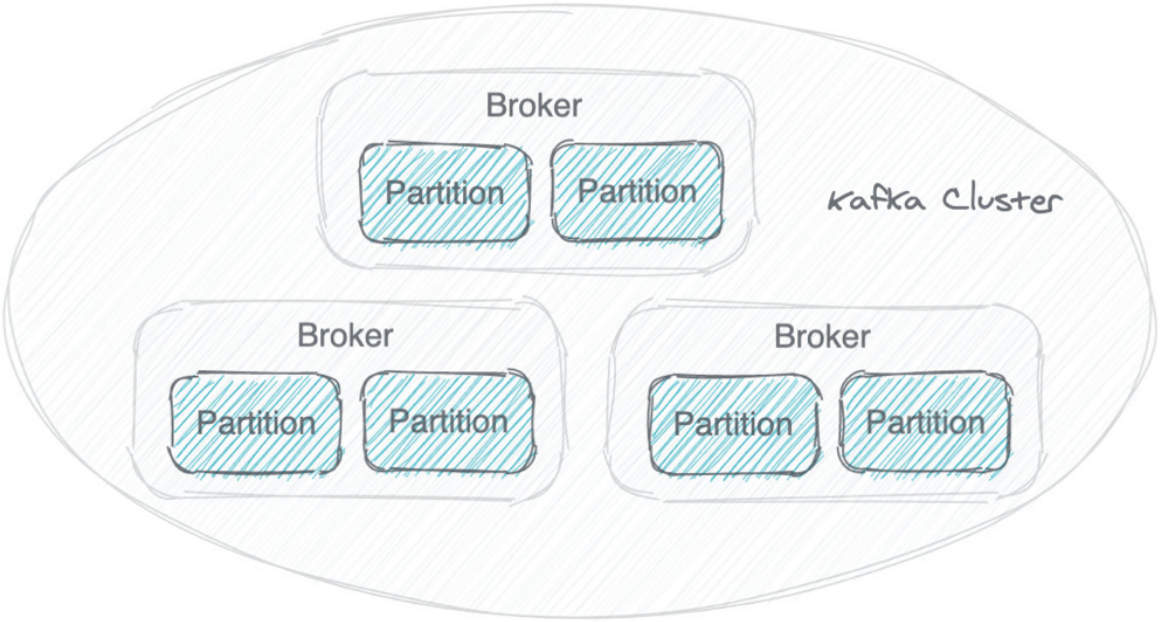
消息的顺序性需要注意，一个 Topic 如果有多个 Partition 的话，那么从 Topic 这个层面来看，消息是无序的。

但单独看 Partition 的话，Partition 内部消息是有序的。

所以，一个 Partition 内部消息有序，一个 Topic 跨 Partition 是无序的。

如果强制要求 Topic 整体有序，就只能让 Topic 只有一个 Partition。

四、Partition 为 Kafka 提供了扩展能力



一个 Kafka 集群由多个 Broker（就是 Server） 构成，每个 Broker 中含有集群的部分数据。

Kafka 把 Topic 的多个 Partition 分布在多个 Broker 中。

这样会有多种好处：

- 如果把 Topic 的所有 Partition 都放在一个 Broker 上，那么这个 Topic 的可扩展性就大大降低了，会受限于这个 Broker 的 IO 能力。把 Partition 分散开之后，Topic 就可以水平扩展 。
- 一个 Topic 可以被多个 Consumer 并行消费。如果 Topic 的所有 Partition 都在一个 Broker，那么支持的 Consumer 数量就有限，而分散之后，可以支持更多的 Consumer。
- 一个 Consumer 可以有多个实例，Partition 分布在多个 Broker 的话，Consumer 的多个实例就可以连接不同的 Broker，大大提升了消息处理能力。可以让一个 Consumer 实例负责一个 Partition，这样消息处理既清晰又高效。

五、Partition 为 Kafka 提供了数据冗余

Kafka 为一个 Partition 生成多个副本，并且把它们分散在不同的

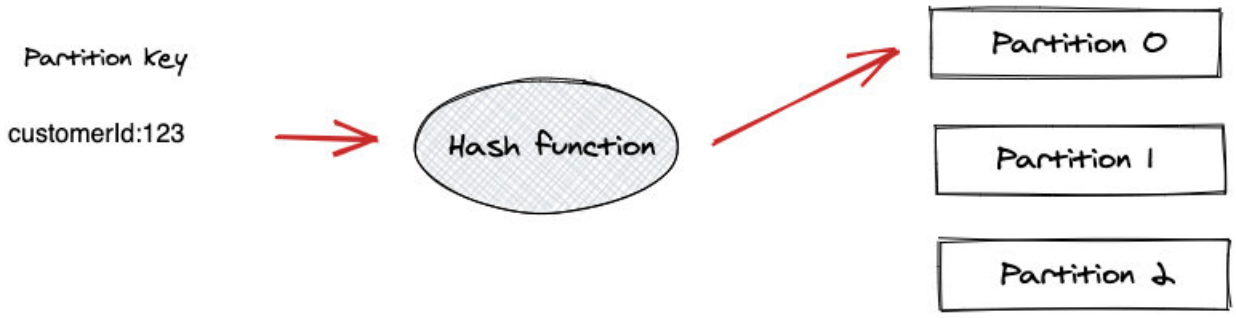
Broker。

如果一个 Broker 故障了，Consumer 可以在其他 Broker 上找到 Partition 的副本，继续获取消息。

六、写入 Partition

一个 Topic 有多个 Partition，那么，向一个 Topic 中发送消息的时候，具体是写入哪个 Partition 呢？有3种写入方式。

1. 使用 Partition Key 写入特定 Partition



Producer 发送消息的时候，可以指定一个 Partition Key，这样就可以写入特定 Partition 了。

Partition Key 可以使用任意值，例如设备ID、User ID。

Partition Key 会传递给一个 Hash 函数，由计算结果决定写入哪个 Partition。

所以，有相同 Partition Key 的消息，会被放到相同的 Partition。

例如使用 User ID 作为 Partition Key，那么此 ID 的消息就都在同一个 Partition，这样可以保证此类消息的有序性。

这种方式需要注意 Partition 热点问题。

例如使用 User ID 作为 Partition Key，如果某一个 User 产生的消息特别多，是一个头部活跃用户，那么此用户的消息都进入同一个

Partition 就会产生热点问题，导致某个 Partition 极其繁忙。

2. 由 kafka 决定

如果没有使用 Partition Key，Kafka 就会使用轮询的方式来决定写入哪个 Partition。

这样，消息会均衡的写入各个 Partition。

但这样无法确保消息的有序性。

3. 自定义规则

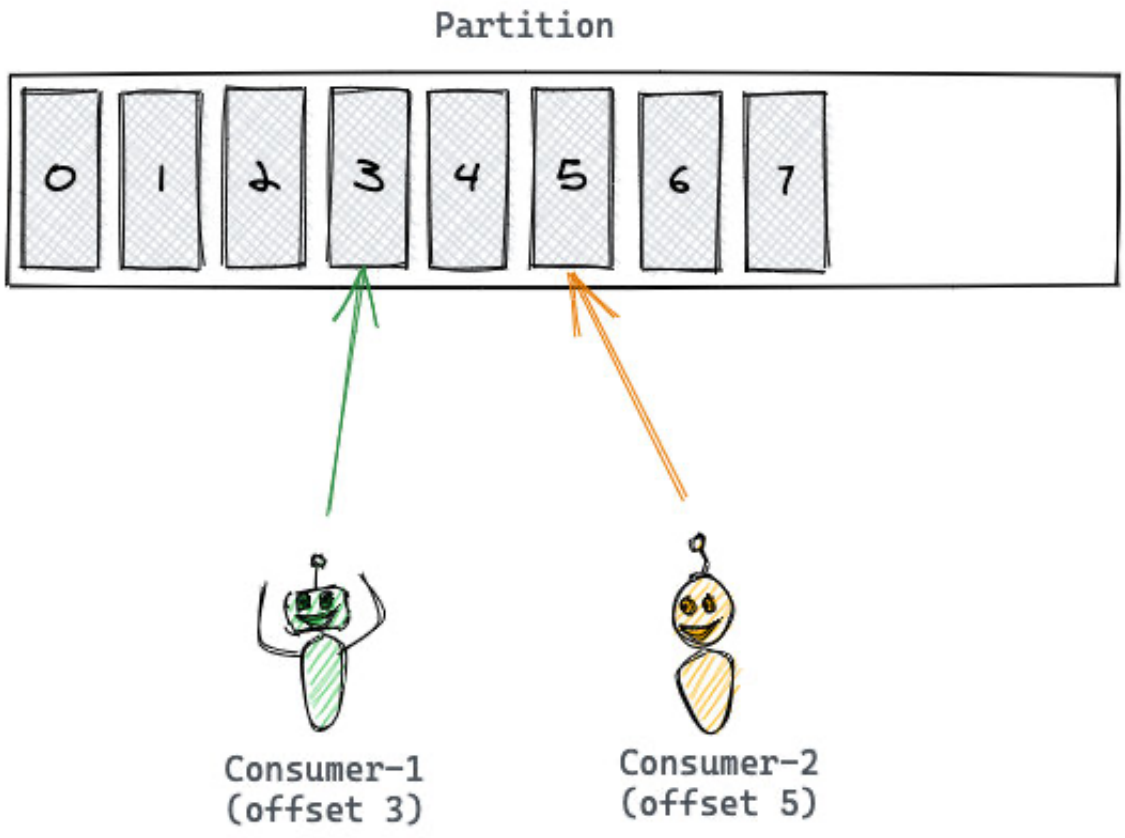
Kafka 支持自定义规则，一个 Producer 可以使用自己的分区指定规则。

七、读取 Partition

Kafka 不像普通消息队列具有发布/订阅功能，Kafka 不会向 Consumer 推送消息。

Consumer 必须自己从 Topic 的 Partition 拉取消息。

一个 Consumer 连接到一个 Broker 的 Partition，从中依次读取消息。

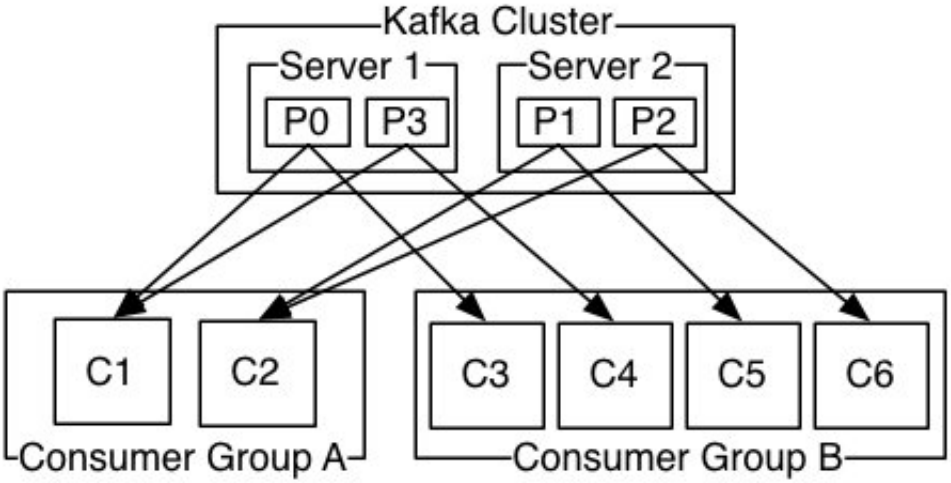


Each consumer has its own view about the partition.

消息的 Offset 就是 Consumer 的游标，根据 Offset 来记录消息的消费情况。

读完一条消息之后，Consumer 会推进到 Partition 中的下一个 Offset，继续读取消息。

Offset 的推进和记录都是 Consumer 的责任，Kafka 是不管的。

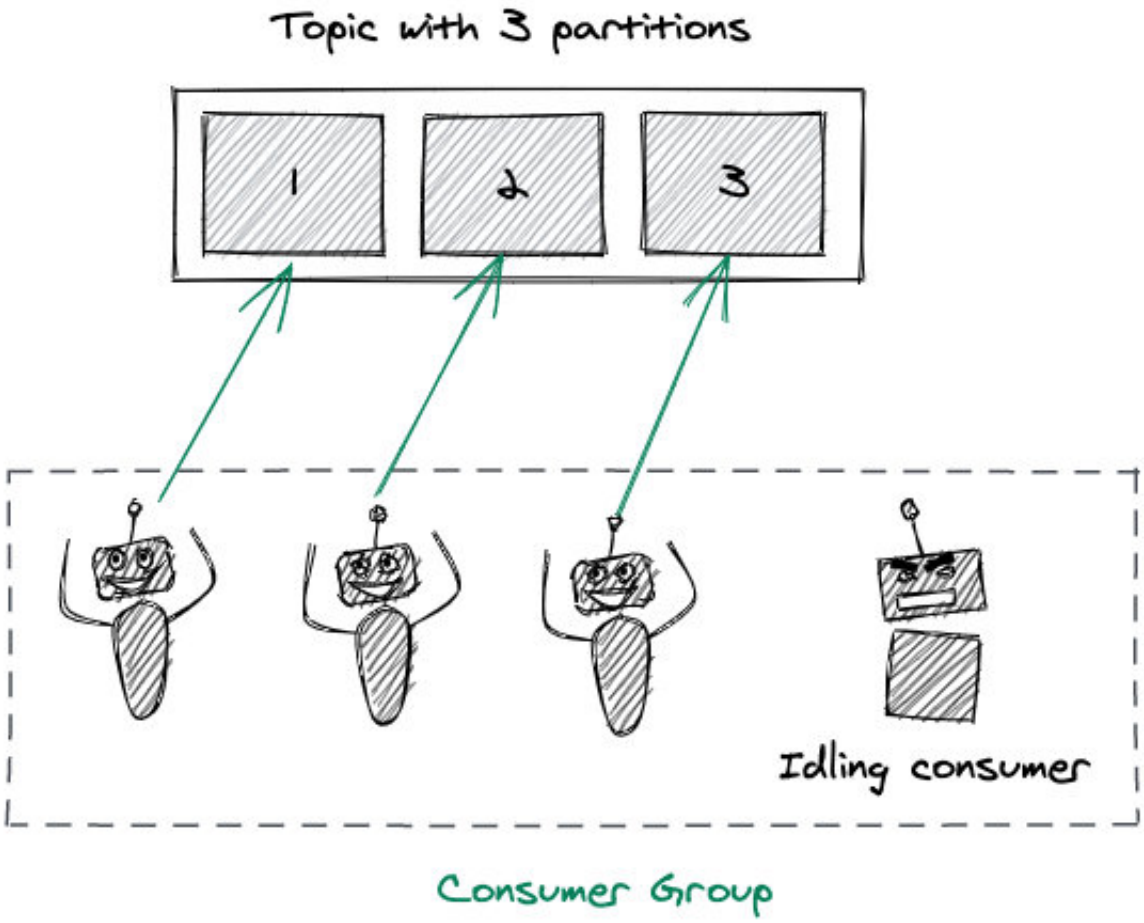


Kafka 中有一个 Consumer Group（消费组）的概念，多个 Consumer 组团去消费一个 Topic。

同组的 Consumer 有相同的 Group ID。

Consumer Group 机制会保障一条消息只被组内唯一一个 Consumer 消费，不会重复消费。

消费组这种方式可以让多个 Partition 并行消费，大大提高了消息的消费能力，最大并行度为 Topic 的 Partition 数量。



例如一个 Topic 有 3 个 Partition，你有 4 个 Consumer 负责这个 Topic，也只会有 Consumer 工作，另一个作为后补队员，当某个 Consumer 故障了，它再补上去，是一种很好的容错机制。

参考资料

https://medium.com/event-driven-utopia/understanding-kafka-topic-partitions-ae40f80552e8