23 | Kafka副本机制详解

2019-07-25 胡夕

你好，我是胡夕。今天我要和你分享的主题是：Apache Kafka的副本机制。

所谓的副本机制（Replication），也可以称之为备份机制，通常是指分布式系统在多台网络互联的机器上保存有相同的数据拷贝。副本机制有什么好处呢？

1. **提供数据冗余**。即使系统部分组件失效，系统依然能够继续运转，因而增加了整体可用性以及数据持久性。
2. **提供高伸缩性**。支持横向扩展，能够通过增加机器的方式来提升读性能，进而提高读操作吞吐量。
3. **改善数据局部性**。允许将数据放入与用户地理位置相近的地方，从而降低系统延时。

这些优点都是在分布式系统教科书中最常被提及的，但是有些遗憾的是，对于Apache Kafka而言，目前只能享受到副本机制带来的第1个好处，也就是提供数据冗余实现高可用性和高持久性。我会在这一讲后面的内容中，详细解释Kafka没能提供第2点和第3点好处的原因。

不过即便如此，副本机制依然是Kafka设计架构的核心所在，它也是Kafka确保系统高可用和消息高持久性的重要基石。

副本定义

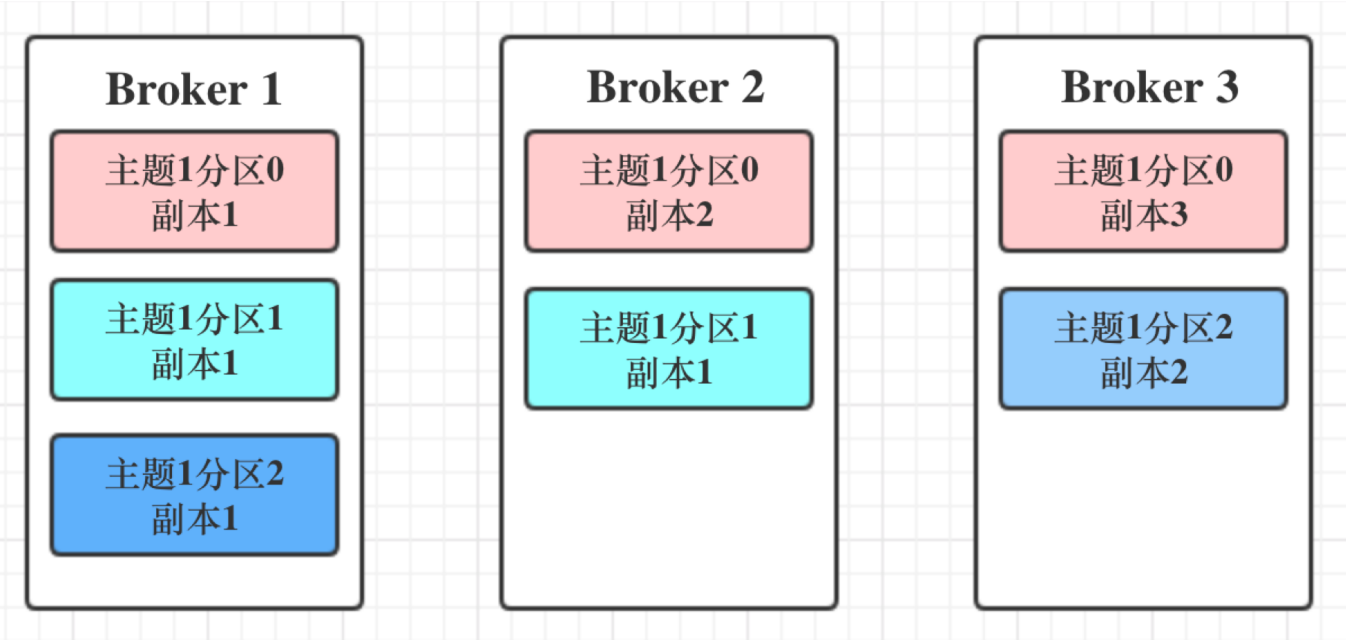
在讨论具体的副本机制之前，我们先花一点时间明确一下副本的含义。

我们之前谈到过，Kafka是有主题概念的，而每个主题又进一步划分成若干个分区。副本的概念实际上是在分区层级下定义的，每个分区配置有若干个副本。

所谓副本（Replica），本质就是一个只能追加写消息的提交日志。根据Kafka副本机制的定义，同一个分区下的所有副本保存有相同的消息序列，这些副本分散保存在不同的Broker上，从而能够对抗部分Broker宕机带来的数据不可用。

在实际生产环境中，每台Broker都可能保存有各个主题下不同分区的不同副本，因此，单个Broker上存有成百上千个副本的现象是非常正常的。

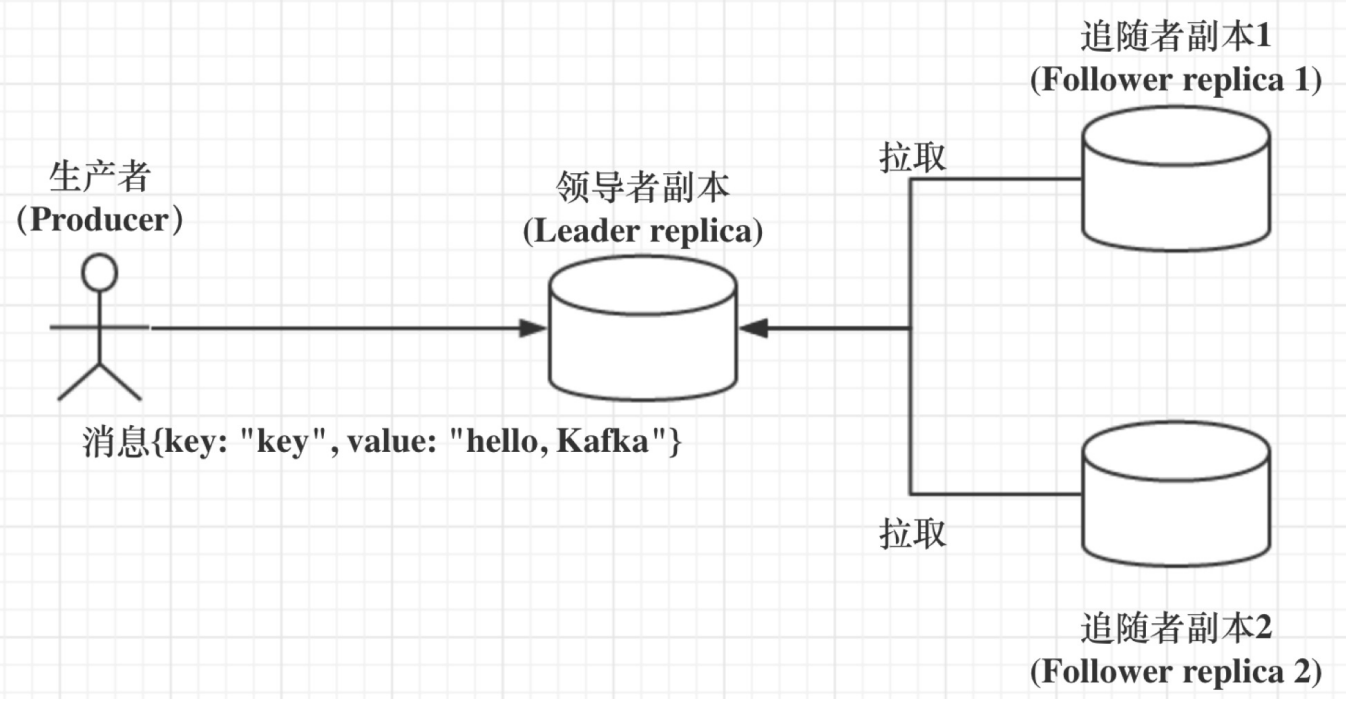
接下来我们来看一张图，它展示的是一个有3台Broker的Kafka集群上的副本分布情况。从这张图中，我们可以看到，主题1分区0的3个副本分散在3台Broker上，其他主题分区的副本也都散落在不同的Broker上，从而实现数据冗余。



副本角色

既然分区下能够配置多个副本，而且这些副本的内容还要一致，那么很自然的一个问题就是：我们该如何确保副本中所有的数据都是一致的呢？特别是对Kafka而言，当生产者发送消息到某个主题后，消息是如何同步到对应的所有副本中的呢？针对这个问题，最常见的解决方案就是采用**基于领导者（Leader-based）的副本机制**。Apache Kafka就是这样的设计。

基于领导者的副本机制的工作原理如下图所示，我来简单解释一下这张图里面的内容。



第一，在Kafka中，副本分成两类：领导者副本（Leader Replica）和追随者副本（Follower Replica）。每个分区在创建时都要选举一个副本，称为领导者副本，其余的副本自动称为追随者副本。

第二，Kafka的副本机制比其他分布式系统要更严格一些。在Kafka中，追随者副本是不对外提供服务的。这就是说，任何一个追随者副本都不能响应消费者和生产者的读写请求。所有的请求都必须由领导者副本来处理，或者说，所有的读写请求都必须发往领导者副本所在的Broker，由该Broker负责处理。追随者副本不处理客户端请求，它唯一的任务就是从领导者副本**异步拉取**消息，并写入到自己的提交日志中，从而实现与领导者副本的同步。

第三，当领导者副本挂掉了，或者说领导者副本所在的Broker宕机时，Kafka依托于ZooKeeper提供的监控功能能够实时感知到，并立即开启新一轮的领导者选举，从追随者副本中选一个作为新的领导者。老Leader副本重启回来后，只能作为追随者副本加入到集群中。

你一定要特别注意上面的第二点，即**追随者副本是不对外提供服务的**。还记得刚刚我们谈到副本机制的好处时，说过Kafka没能提供读操作横向扩展以及改善局部性吗？具体的原因就在于此。

对于客户端用户而言，Kafka的追随者副本没有任何作用，它既不能像MySQL那样帮助领导者副本“抗读”，也不能实现将某些副本放到离客户端近的地方来改善数据局部性。

既然如此，Kafka为什么要这样设计呢？其实这种副本机制有两个方面的好处。

1.方便实现“Read-your-writes”。

所谓Read-your-writes，顾名思义就是，当你使用生产者API向Kafka成功写入消息后，马上使用消费者API去读取刚才生产的消息。

举个例子，比如你平时发微博时，你发完一条微博，肯定是希望能立即看到的，这就是典型的Read-your-writes场景。如果允许追随者副本对外提供服务，由于副本同步是异步的，因此有可能出现追随者副本还没有从领导者副本那里拉取到最新的消息，从而使得客户端看不到最新写入的消息。

2.方便实现单调读（Monotonic Reads）。

什么是单调读呢？就是对于一个消费者用户而言，在多次消费消息时，它不会看到某条消息一会儿存在一会儿不存在。

如果允许追随者副本提供读服务，那么假设当前有2个追随者副本F1和F2，它们异步地拉取领导者副本数据。倘若F1拉取了Leader的最新消息而F2还未及时拉取，那么，此时如果有一个消费者先从F1读取消息之后又从F2拉取消息，它可能会看到这样的现象：第一次消费时看到的最新消息在第二次消费时不见了，这就不是单调读一致性。但是，如果所有的读请求都是由Leader来处理，那么Kafka就很容易实现单调读一致性。

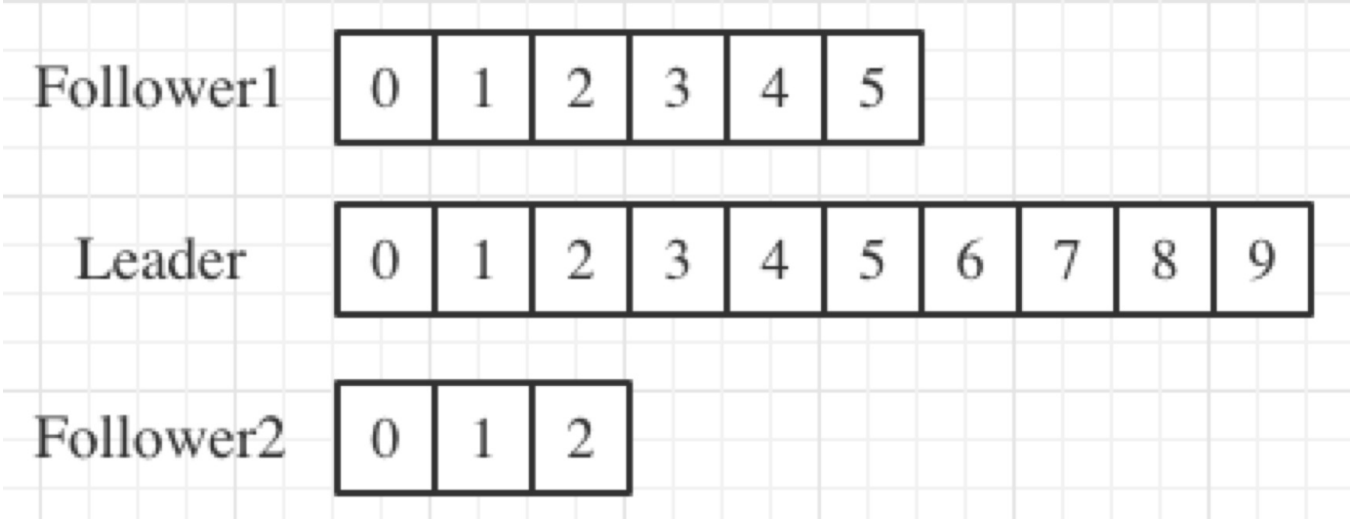
In-sync Replicas（ISR）

我们刚刚反复说过，追随者副本不提供服务，只是定期地异步拉取领导者副本中的数据而已。既然是异步的，就存在着不可能与Leader实时同步的风险。在探讨如何正确应对这种风险之前，我们必须要精确地知道同步的含义是什么。或者说，Kafka要明确地告诉我们，追随者副本到底在什么条件下才算与Leader同步。

基于这个想法，Kafka引入了In-sync Replicas，也就是所谓的ISR副本集合。ISR中的副本都是与Leader同步的副本，相反，不在ISR中的追随者副本就被认为是与Leader不同步的。那么，到底什么副本能够进入到ISR中呢？

我们首先要明确的是，Leader副本天然就在ISR中。也就是说，**ISR不只是追随者副本集合，它必然包括Leader副本。甚至在某些情况下，ISR只有Leader这一个副本**。

另外，能够进入到ISR的追随者副本要满足一定的条件。至于是什么条件，我先卖个关子，我们先来一起看看下面这张图。



图中有3个副本：1个领导者副本和2个追随者副本。Leader副本当前写入了10条消息，Follower1副本同步了其中的6条消息，而Follower2副本只同步了其中的3条消息。现在，请你思考一下，对于这2个追随者副本，你觉得哪个追随者副本与Leader不同步？

答案是，要根据具体情况来定。换成英文，就是那句著名的“It depends”。看上去好像Follower2的消息数比Leader少了很多，它是最有可能与Leader不同步的。的确是这样的，但仅仅是可能。

事实上，这张图中的2个Follower副本都有可能与Leader不同步，但也都有可能与Leader同步。也就是说，Kafka判断Follower是否与Leader同步的标准，不是看相差的消息数，而是另有“玄机”。

这个标准就是Broker端参数replica.lag.time.max.ms参数值。这个参数的含义是Follower副本能够落后Leader副本的最长时间间隔，当前默认值是10秒。这就是说，只要一个Follower副本落后Leader副本的时间不连续超过10秒，那么Kafka就认为该Follower副本与Leader是同步的，即使此时Follower副本中保存的消息明显少于Leader副本中的消息。

我们在前面说过，Follower副本唯一的工作就是不断地从Leader副本拉取消息，然后写入到自己的提交日志中。如果这个同步过程的速度持续慢于Leader副本的消息写入速度，那么在replica.lag.time.max.ms时间后，此Follower副本就会被认为是与Leader副本不同步的，因此不能再放入ISR中。此时，Kafka会自动收缩ISR集合，将该副本“踢出”ISR。

值得注意的是，倘若该副本后面慢慢地追上了Leader的进度，那么它是能够重新被加回ISR的。这也表明，ISR是一个动态调整的集合，而非静态不变的。

Unclean领导者选举（Unclean Leader Election）

既然ISR是可以动态调整的，那么自然就可以出现这样的情形：ISR为空。因为Leader副本天然就在ISR中，如果ISR为空了，就说明Leader副本也“挂掉”了，Kafka需要重新选举一个新的Leader。可是ISR是空，此时该怎么选举新Leader呢？

Kafka把所有不在ISR中的存活副本都称为非同步副本。通常来说，非同步副本落后Leader太多，因此，如果选择这些副本作为新Leader，就可能出现数据的丢失。毕竟，这些副本中保存的消息远远落后于老Leader中的消息。在Kafka中，选举这种副本的过程称为Unclean领导者选举。**Broker端参数unclean.leader.election.enable控制是否允许Unclean领导者选举**。

开启Unclean领导者选举可能会造成数据丢失，但好处是，它使得分区Leader副本一直存在，不至于停止对外提供服务，因此提升了高可用性。反之，禁止Unclean领导者选举的好处在于维护了数据的一致性，避免了消息丢失，但牺牲了高可用性。

如果你听说过CAP理论的话，你一定知道，一个分布式系统通常只能同时满足一致性（Consistency）、可用性（Availability）、分区容错性（Partition tolerance）中的两个。显然，在这个问题上，Kafka赋予你选择C或A的权利。

你可以根据你的实际业务场景决定是否开启Unclean领导者选举。不过，我强烈建议你**不要**开启它，毕竟我们还可以通过其他的方式来提升高可用性。如果为了这点儿高可用性的改善，牺牲了数据一致性，那就非常不值当了。

小结

今天，我主要跟你分享了Apache Kafka的副本机制以及它们实现的原理。坦率地说，我觉得有些地方可能讲浅了，如果要百分之百地了解Replication，你还是要熟读一下Kafka相应的源代码。不过你也不用担心，在专栏后面的内容中，我会专门从源码角度分析副本机制，特别是Follower副本从Leader副本拉取消息的全过程。从技术深度上来说，那一讲应该算是本专栏中最贴近技术内幕的分析了，你一定不要错过。

开放讨论

到目前为止，我反复强调了Follower副本不对外提供服务这件事情。有意思的是，社区最近正在考虑是否要打破这个限制，即允许Follower副本处理客户端消费者发来的请求。社区主要的考量是，这能够用于改善云上数据的局部性，更好地服务地理位置相近的客户。如果允许Follower副本对外提供读服务，你觉得应该如何避免或缓解因Follower副本与Leader副本不同步而导致的数据不一致的情形？

欢迎写下你的思考和答案，我们一起讨论。如果你觉得有所收获，也欢迎把文章分享给你的朋友。

精选留言

趙衍

13

老师讲的很好，我做一些补充吧。

Kafka在启动的时候会开启两个任务，一个任务用来定期地检查是否需要缩减或者扩大ISR集合，这个周期是replica.lag.time.max.ms的一半，默认5000ms。当检测到ISR集合中有失效副本时，就会收缩ISR集合，当检查到有Follower的HighWatermark追赶上Leader时，就会扩充ISR。

除此之外，当ISR集合发生变更的时候还会将变更后的记录缓存到isrChangeSet中，另外一个任务会周期性地检查这个Set,如果发现这个Set中有ISR集合的变更记录，那么它会在zk中持久化一个节点。然后因为Controllr在这个节点的路径上注册了一个Watcher，所以它就能够感知到ISR的变化，并向它所管理的broker发送更新元数据的请求。最后删除该路径下已经处理过的节点。

此外，在0.9X版本之前，Kafka中还有另外一个参数replica.lag.max.messages，它也是用来判定失效副本的，当一个副本滞后leader副本的消息数超过这个参数的大小时，则判定它处于同步失效的状态。它与replica.lag.time.max.ms参数判定出的失效副本取并集组成一个失效副本集合。

不过这个参数本身很难给出一个合适的值。以默认的值4000为例，对于消息流入速度很低的主题（比如TPS为10），这个参数就没什么用；对于消息流入速度很高的主题（比如TPS为2000），这个参数的取值又会引入ISR的频繁变动。所以从0.9x版本开始，Kafka就彻底移除了这一

个参数。

2019-07-27

刘彬 1

老师好，想请教您两个问题，如下：
如果某个follower副本同步持续慢于leader副本写入速度，repkica.lag.time.max.ms 是对于二者的同步时间做的判断，我理解就是如果一直检查10s follower都赶不上leader副本的进度！
但是，这个同步进度是用哪一块进行判别的呢？是通过index值吗？
另外，如果某个follower不在ISR中了，kafka如果维持副本数均衡呢？比如设置了副本数为3，其中一个副本不在ISR集合中了，那么就一直少了一个副本吗？前提是这个副本一直没有跟上leader的同步进度！
谢谢！

2019-07-25

作者回复

1. 通过比较follower和leader的最新消息位移或末端消息位移（Log End Offset, LEO）
2. 嗯，就一直少一个副本了

2019-07-25

凯 0

请问一下，producer生产消息ack=all的时候，消息是怎么保证到follower的，因为看到follower是异步拉取数据的，难道是看leader和follower上面的offset吗？

2019-08-01

曹伟雄 0

老师你好，请教个问题，在不同场景下，应该如何设计副本数? 有什么要注意的地方?

2019-07-30

作者回复

其实也没什么规则。如果你觉得你的消息很值钱，不能丢，副本数就多第一点，比如3或4；否则一般的消息设置成2~3就可以了。

2019-07-30

摇山樵客™ 0

谢谢老师的讲解！
这里有两个问题想请教一下老师：
1.kafka使用replica.lag.max.time.ms来判断是否保留replica在ISR里，那么问题来了，在吞吐量较高的场景下，replica满足这个时间限制，但是LEO相差比较大，leader这时候挂掉，这个replica被选举为新leader，这个时候是不是有一部分数据丢失了？
2.如果问题1确实存在，目前是怎样处理的呢？

2019-07-29

作者回复

是有这种可能，如果你在意这种情况producer端设置acks=all就可以避免了

2019-07-30

外星人 0

请问，关闭unclean后，有哪些方法可以保证available啊？谢谢

2019-07-28

作者回复

增加副本数：）

2019-07-29

电光火石 0

如果我创建的时候指定有三个副本

1. 如果某一个副本所在的broker挂了，kafka会在另一个broker上面新创建一个partition来补充吗？

2. 如果这三个副本所在的broker都挂了，那kafka会不会在一个新的broker上面重新创建一个新的partition来支持读写，还是说，这个partition就不在工作了？

2019-07-27

作者回复

1. 不会

2. 不工作了

2019-07-29

张学磊 0

个人觉得只所以使用消息队列进行异步处理就不会太关心消息处理的及时性，那当允许Follower副本对外提供读请求时，第一消费者可以降低读取消息的频率，给予Follower副本一定同步的时间，第二消费者在读取消息是优先读取Follower副本中的信息，如果读取不到再转到Leader副本中进行读取。

2019-07-27

Geek_986289 0

老师请问下，即使再怎么同步，也一定是有延迟的，在leader 挂掉切到一个ISR后怎么保证这部分延迟同步的数据不会丢呢？

2019-07-26

rm -rf 0

这就跟mysql的主从读写差不多，主写入速度过快导致从产生不一致情况。如果真要保证更好的一致性来让follower也能读，只能牺牲可用性了，比如生产的时候设置acks=all或者min.insync.replicas == 副本数，或者有其他更好办法？老师指导一下

2019-07-26

ban 0

老师，可以讲下可用性、分区容错性的区别吗。搞不懂两个的区别，有点模糊

2019-07-25

作者回复

可用性就不讲了估计大家理解的差不多。分区容错性指的是由于网络故障部分网络被切断，与其他网络断开了连接

2019-07-26

ban

0

老师，Kafka 会自动收缩 ISR 集合，将该副本“踢出”ISR。踢出去后副本还会做同步操作吗

2019-07-25

作者回复

会的

2019-07-26

ban

0

老师，你说“对于客户端用户而言，Kafka 的追随者副本没有任何作用，它既不能像 MySQL 那样帮助领导者副本“抗读”，也不能实现将某些副本放到离客户端近的地方来改善数据局部性。”

副本按地区分布只有领导者能读。那如果从分区的角度来说，分区按地区来分布，是不是就可以说改善数据的局部性，也可以做到横向扩展。

2019-07-25

z.l

0

我觉得应该是，一个 consumer实例固定从某一个副本上拉消息。

2019-07-25

QQ怪

0

老师讲的真好，期待源码讲解

2019-07-25

振超

0

老师，如果仅仅以 replica.lag.time.max.ms 决定 follower 是否应该加入 ISR 集合，是否存在这样一种情况，一个 follower 离线了一段时间之后重新上线，然后开始与 leader 同步，此时同步时间间隔在 replica.lag.time.max.ms 内，该 follower 成功加入到 ISR 集合，但是因为长时间失效，导致该 follower 的 LEO 值小于 leader 的 HW，如果此时 leader 宕机，并正好选择该 follower 作为新的 leader，这样是不是就丢消息了？

2019-07-25

作者回复

不会。ISR比较的是follower和leader的LEO，不是和leader的HW比较。

2019-07-25

丘壑

0

老师，producer的config参数中的acks参数，应该就是用来保证消息是否被写入了多个副本的机制之一把，如果开启acks，除非这几个副本的broker都挂掉了，这样是否就会保证ISR不会为空呢？

2019-07-25

作者回复

不能。acks的设置与否与ISR中有几个副本没有关系

2019-07-25

☆appleう

0

老师，这里说的unclean选举新的领导者方式，会判断哪个追求者副本与挂掉的老leader差异最小吗？

如果我们不选择最小差异的副本作为新领导者，那么这一段时间broker就不能用了，有什么方式可以支持一致性数据的方案吗

2019-07-25

作者回复

不会判断。后面的问题好像没太明白。。。

2019-07-25

cgddw 0

问下老师，kafka新增一块磁盘，如何做到数据迁移到新盘上。是这样的，之前磁盘只有一块，用久了磁盘忙了，修改保留数据时间已经不能支撑业务了。于是在不重启机器的情况下，新增了一块盘，然后重启kafka，发现新盘没有数据进来。请问要如何做，可以把数据迁移到新盘上

2019-07-25

作者回复

使用kafka-reassign-partitions脚本可以，具体用法看一下--reassignment-json-file参数

2019-07-25

lmtoo 0

在“无消息丢失配置怎么实现？”中有两个配置项，acks=all这里的all是指所有的isr的副本还是所有的副本,如果是所有的副本，那副本同步都是通过拉取实现的，那这个等待时间就是最慢的那个副本了，也就是消费者端等待时间>replica.lag.time.max.ms；min.insync.replicas这个参数是表示isr的最小副本数？

2019-07-25

作者回复

指的是ISR中的所有副本

2019-07-25