**Producer客户端配置**

1. producer的配置：producer.type=sync

Kafka的发送模式由producer端的配置参数producer.type来设置，这个参数指定了在后台线程中消息的发送方式是同步的还是异步的，默认是同步的方式，即producer.type=sync。如果设置成异步的模式，即producer.type=async，可以是producer以batch的形式push数据，这样会极大的提高broker的性能，但是这样会增加丢失数据的风险。如果需要确保消息的可靠性，必须要将producer.type设置为sync。

1. producer的配置：request.required.acks=-1

1（默认）：这意味着producer在ISR中的leader已成功收到的数据并得到确认后发送下一条message。如果leader宕机了，则会丢失数据。

0：这意味着producer无需等待来自broker的确认而继续发送下一批消息。这种情况下数据传输效率最高，但是数据可靠性确是最低的。

-1：producer需要等待ISR中的所有follower都确认接收到数据后才算一次发送完成，可靠性最高。但是这样也不能保证数据不丢失，比如当ISR中只有leader时（前面ISR那一节讲到，ISR中的成员由于某些情况会增加也会减少，最少就只剩一个leader），这样就变成了acks=1的情况。

**Topic配置**

1. topic的配置：replication.factor>=3,即副本数至少是3个；2<=min.insync.replicas<=replication.factor

min.insync.replicas这个参数设定ISR中的最小副本数是多少，默认值为1，当且仅当request.required.acks参数设置为-1时，此参数才生效。如果ISR中的副本数少于min.insync.replicas配置的数量时，producer客户端会返回异常。

min.insync.replicas需大于等于2，如果=1的话，当如果leader宕机且无法再启动起来，那么未同步的数据将丢失。

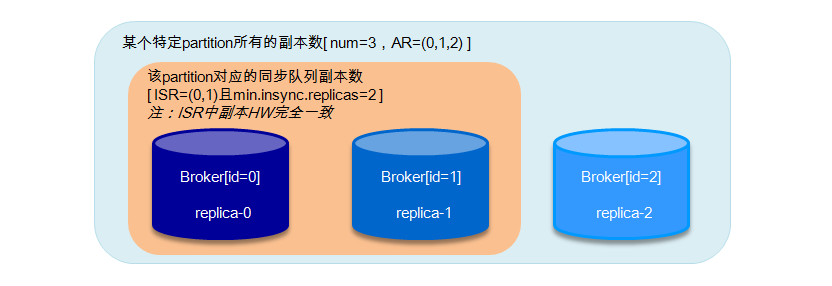
所有的副本（replicas）统称为Assigned Replicas，即AR。ISR（In-Sync Replicas）是AR中的一个子集，由leader维护ISR列表，是指副本同步队列。ISR中包括：leader和follower. Follower在replica.lag.time.max.ms时间内未向Leader发送Fetch请求（也即数据复制请求），则Leader会将其从ISR中移除。如果某Follower持续向Leader发送Fetch请求，但是它与Leader的数据差距在replica.lag.max.messages以上，也会被Leader从ISR中移除。

**Brocker配置**

1. broker的配置：leader的选举条件unclean.leader.election.enable=false

unclean.leader.election.enable=false：等待**ISR**中的replica“活”过来，并且选它作为leader

unclean.leader.election.enable这个参数对于leader的选举、系统的可用性以及数据的可靠性都有至关重要的影响。下面我们来分析下几种典型的场景。

[](https://res.infoq.com/articles/depth-interpretation-of-kafka-data-reliability/zh/resources/9.jpg)

如果上图所示，假设某个partition中的副本数为3，replica-0, replica-1, replica-2分别存放在broker0, broker1和broker2中。AR=(0,1,2)，ISR=(0,1)。设置request.required.acks=-1, min.insync.replicas=2，unclean.leader.election.enable=false。这里讲broker0中的副本也称之为broker0起初broker0为leader，broker1为follower。

* 当ISR中的replica-0出现crash的情况时，broker1选举为新的leader[ISR=(1)]，因为受min.insync.replicas=2影响，write不能服务，但是read能继续正常服务。此种情况恢复方案：
  1. 尝试恢复(重启)replica-0，如果能起来，系统正常；
  2. 如果replica-0不能恢复，需要将min.insync.replicas设置为1，恢复write功能。
* 当ISR中的replica-0出现crash，紧接着replica-1也出现了crash, 此时[ISR=(1),leader=-1],不能对外提供服务，此种情况恢复方案：
  1. 尝试恢复replica-0和replica-1，如果都能起来，则系统恢复正常；
  2. 如果replica-0起来，而replica-1不能起来，这时候仍然不能选出leader，因为当设置unclean.leader.election.enable=false时，leader只能从ISR中选举，当ISR中所有副本都失效之后，需要ISR中最后失效的那个副本能恢复之后才能选举leader, 即replica-0先失效，replica-1后失效，需要replica-1恢复后才能选举leader。保守的方案建议把unclean.leader.election.enable设置为true,但是这样会有丢失数据的情况发生，这样可以恢复read服务。同样需要将min.insync.replicas设置为1，恢复write功能；
  3. replica-1恢复，replica-0不能恢复，这个情况上面遇到过，read服务可用，需要将min.insync.replicas设置为1，恢复write功能；
  4. replica-0和replica-1都不能恢复，这种情况可以参考情形2.
* 当ISR中的replica-0, replica-1同时宕机,此时[ISR=(0,1)],不能对外提供服务，此种情况恢复方案：尝试恢复replica-0和replica-1，当其中任意一个副本恢复正常时，对外可以提供read服务。直到2个副本恢复正常，write功能才能恢复，或者将将min.insync.replicas设置为1。