124 主从复制架构中的数据延迟问题，应该如何解决？

之前大家都已经了解过主从复制架构是如何搭建的了，其实他并不难，但是这里比较关键的是，主从复制可能会有较大的延迟。这个延迟是什么意思呢？就是说主库可能你都写入了100条数据了，结果从库才复制过去了50条数据，那么从库就比主库落后了50条数据。

这就是所谓的主从延迟的问题。

可是为什么会产生这个主从延迟的问题呢？也很简单，其实你主库是多线程并发写入的，这个大家都知道的，所以主库写入数据的速度可能是很快的，但是从库是单个线程缓慢拉取数据的，所以才会导致从库复制数据的速度是比较慢的。

那自然会导致主从之间的延迟问题了，大家想，是不是？

那么这个主从之间到底延迟了多少时间呢？这个可以用一个工具来进行监控，比较推荐的是percona-toolkit工具集里的pt-heartbeat工具，他会在主库里创建一个heartbeat表，然后会有一个线程定时更新这个表里的时间戳字段，从库上就有一个monitor线程会负责检查从库同步过来的heartbeat表里的时间戳。

把时间戳跟当前时间戳比较一下，其实就知道主从之间同步落后了多长时间了，关于这个工具的使用，大家可以自行搜索一下，我们这里就不展开了，总之，主从之间延迟了多长时间，我们这里实际上是可以看到的。

那么这个主从同步延迟的问题，会导致一些什么样的不良情况呢？

其实大家可以思考一下，如果你做了读写分离架构，写都往主库写，读都从从库读，那么会不会你的系统刚写入一条数据到主库，接着代码里立即就在从库里读取，可能此时从库复制有延迟，你会读不到刚写入进去的数据！

没错，就是这个问题，这是我们之前也经常会遇到的一个问题。另外就是有可能你的从库同步数据太慢了，导致你从库读取的数据都是落后和过期的，也可能会导致你的系统产生一定的业务上的bug。

所以针对这个问题，首先你应该做的，是尽可能缩小主从同步的延迟时间，那么怎么做呢？其实就是让从库也用多线程并行复制数据就可以了，这样从库复制数据的速度快了，延迟就会很低了。

MySQL 5.7就已经支持并行复制了，可以在从库里设置slave_parallel_workers>0，然后把slave_parallel_type设置为LOGICAL_CLOCK，就ok了。

另外，如果你觉得还是要求刚写入的数据你立马强制必须一定可以读到，那么此时你可以使用一个办法，就是在类似MyCat或者Sharding-Sphere之类的中间件里设置强制读写都从主库走，这样你写入主库的数据，强制从主库里读取，一定立即可以读到的。

总体而言就是这样了，大家在落实读写分离架构的时候，要注意一下复制方式，是异步还是半同步？如果说你对数据丢失并不是强要求不能丢失的话，可以用异步模式来复制，再配合一下从库的并行复制机制。

如果说你要对MySQL做高可用保证数据绝对不丢失的话，建议还是用半同步机制比较好一些，同理最好是配合从库的并行复制机制。

接下来配合这个主从复制架构，我们可以来讲解一下数据库的高可用架构了。

End