19-跨系统实时同步数据，分布式事务是唯一的解决方案吗？

你好，我是李玥。

我们在《15 | MySQL存储海量数据的最后一招：分库分表》这节课中讲过，数据量太大的时候，单个存储节点存不下，那就只能把数据分片存储。

数据分片之后，我们对数据的查询就没那么自由了。比如订单表如果按照用户ID作为Sharding Key来分片，那就只能按照用户维度来查询。如果我是一个商家，我想查我店铺的订单，对不起，做不到了。（当然，强行查也不是不行，在所有分片上都查一遍，再把结果聚合起来，又慢又麻烦，实际意义不大。）

对于这样的需求，普遍的解决办法是用空间换时间，毕竟现在存储越来越便宜。再存一份订单数据到商家订单库，然后以店铺ID作为Sharding Key分片，专门供商家查询订单。

另外，之前我们在《06 | 如何用Elasticsearch构建商品搜索系统》这节课也讲到过，同样一份商品数据，如果我们是按照关键字搜索，放在ES里就比放在MySQL快了几个数量级。原因是，数据组织方式、物理存储结构和查询方式，对查询性能的影响是巨大的，而且海量数据还会指数级地放大这个性能差距。

所以，在大厂中，对于海量数据的处理原则，都是根据业务对数据查询的需求，反过来确定选择什么数据库、如何组织数据结构、如何分片数据，这样才能达到最优的查询性能。同样一份订单数据，除了在订单库保存一份用于在线交易以外，还会在各种数据库中，以各种各样的组织方式存储，用于满足不同业务系统的查询需求。像BAT这种大厂，它的核心业务数据，存个几十上百份是非常正常的。

那么问题来了，如何能够做到让这么多份数据实时地保持同步呢？

我们之前讲过分布式事务，可以解决数据一致性的问题。比如说，你可以用本地消息表，把一份数据实时同步给另外两、三个数据库，这样还可以接受，太多的话也是不行的，并且对在线交易业务还有侵入性，所以分布式事务是解决不了这个问题的。

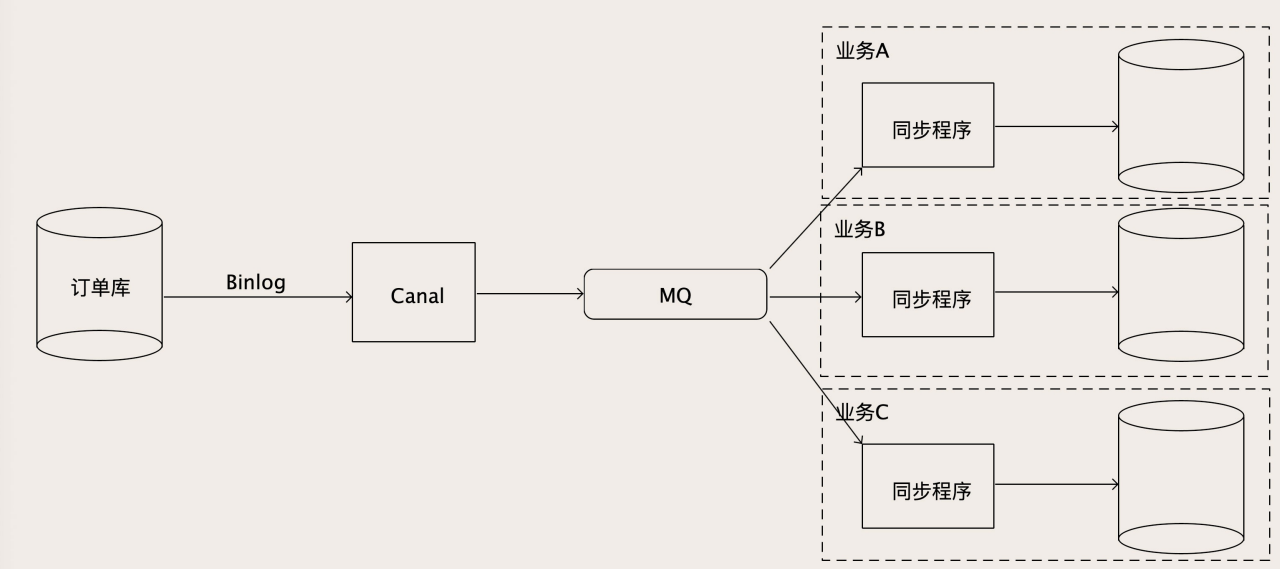
今天这节课我们就来说一下，如何把订单数据实时、准确无误地同步到这么多异构的数据中去。

使用Binlog和MQ构建实时数据同步系统

早期大数据刚刚兴起的时候，大多数系统还做不到异构数据库实时同步，那个时候普遍的做法是，使用ETL工具定时同步数据，在T+1时刻去同步上一个周期的数据，然后再做后续的计算和分析。定时ETL对于一些需要实时查询数据的业务需求就无能为力了。所以，这种定时同步的方式，基本上都被实时同步的方式给取代了。

怎么来做这么大数据量、这么多个异构数据库的实时同步呢？你还记得我在《17 | 大厂都是怎么做MySQL to Redis同步的》这节课中讲到的方法吧？利用Canal把自己伪装成一个MySQL的从库，从MySQL实时接收Binlog然后写入Redis中。把这个方法稍微改进一下，就可以用来做异构数据库的同步了。

为了能够支撑下游众多的数据库，从Canal出来的Binlog数据肯定不能直接去写下游那么多数据库，一是写不过来，二是对于每个下游数据库，它可能还有一些数据转换和过滤的工作要做。所以需要增加一个MQ来解耦上下游。



Canal从MySQL收到Binlog并解析成结构化数据之后，直接写入到MQ的一个订单Binlog主题中，然后每一个需要同步订单数据的业务方，都去订阅这个MQ中的订单Binlog主题，消费解析后的Binlog数据。在每个消费者自己的同步程序中，它既可以直接入库，也可以做一些数据转换、过滤或者计算之后再入库，这样就比较灵活了。

如何保证数据同步的实时性

这个方法看起来不难，但是非常容易出现性能问题。有些接收Binlog消息的下游业务，对数据的实时性要求比较高，不能容忍太高的同步时延。比如说，每个电商在大促的时候，都会有一个大屏幕，实时显示现在有多少笔交易，交易额是多少。这个东西都是给老板们看的，如果说大促的时候，你让老板们半小时之后才看到数字，那估计你就得走人了。

大促的时候，数据量大、并发高、数据库中的数据变动频繁，同步的Binlog流量也非常大。为了保证这个同步的实时性，整个数据同步链条上的任何一个环节，它的处理速度都必须得跟得上才行。我们一步一步分析可能会出现性能瓶颈的环节。

源头的订单库，如果它出现繁忙，对业务的影响就不只是大屏延迟了，那就影响到用户下单了，这个问题是数据库本身要解决的，这里我们不考虑。再顺着数据流向往下看，Canal和MQ这两个环节，由于没什么业务逻辑，性能都非常好。所以，**一般容易成为性能瓶颈的就是消费MQ的同步程序**，因为这些同步程序里面一般都会有一些业务逻辑，而且如果下游的数据库写性能跟不上，表象也是这个同步程序处理性能上不来，消息积压在MQ里面。

那我们能不能多加一些同步程序的实例数，或者增加线程数，通过增加并发来提升处理能力呢？这个地方的并发数，还真不是随便说扩容就可以就扩容的，我来跟你讲一下为什么。

我们知道，MySQL主从同步Binlog，是一个单线程的同步过程。为什么是单线程？原因很简单，在从库执行Binlog的时候，必须按顺序执行，才能保证数据和主库是一样的。**为了确保数据一致性，Binlog的顺序很重要，是绝对不能乱序的。** 严格来说，对于每一个MySQL实例，整个处理链条都必须是单线程串行执行，MQ的主题也必须设置为只有1个分区（队列），这样才能保证数据同步过程中的Binlog是严格有序的，写到目标数据库的数据才能是正确的。

那单线程处理速度上不去，消息越积压越多，这不无解了吗？其实办法还是有的，但是必须得和业务结合起来解决。

还是拿订单库来说啊，其实我们并不需要对订单库所有的更新操作都严格有序地执行，比如说A和B两个订单号不同的订单，这两个订单谁先更新谁后更新并不影响数据的一致性，因为这两个订单完全没有任何关系。但是同一个订单，如果更新的Binlog执行顺序错了，那同步出来的订单数据真的就错了。

也就是说，我们只要保证每个订单的更新操作日志的顺序别乱就可以了。这种一致性要求称为**因果一致性（Causal Consistency）**，有因果关系的数据之间必须要严格地保证顺序，没有因果关系的数据之间的顺序是无所谓的。

基于这个理论基础，我们就可以并行地来进行数据同步，具体的做法是这样的。

首先根据下游同步程序的消费能力，计算出需要多少并发；然后设置MQ中主题的分区（队列）数量和并发数一致。因为MQ是可以保证同一分区内，消息是不会乱序的，所以我们需要把具有因果关系的Binlog都放到相同的分区中去，就可以保证同步数据的因果一致性。对应到订单库就是，相同订单号的Binlog必须发到同一个分区上。

这是不是和之前讲过的数据库分片有点儿像呢？那分片算法就可以拿过来复用了，比如我们可以用最简单的哈希算法，Binlog中订单号除以MQ分区总数，余数就是这条Binlog消息发往的分区号。

Canal自带的分区策略就支持按照指定的Key，把Binlog哈希到下游的MQ中去，具体的配置可以看一下Canal接入MQ的文档。

小结

对于海量数据，必须要按照查询方式选择数据库类型和数据的组织方式，才能达到理想的查询性能。这就需要把同一份数据，按照不同的业务需求，以不同的组织方式存放到各种异构数据库中。因为数据的来源大多都是在线交易系统的MySQL数据库，所以我们可以利用MySQL的Binlog来实现异构数据库之间的实时数据同步。

为了能够支撑众多下游数据库实时同步的需求，可以通过MQ解耦上下游，Binlog先发送到MQ中，下游各业务方可以消费MQ中的消息再写入各自的数据库。

如果下游处理能力不能满足要求，可以增加MQ中的分区数量实现并发同步，但需要结合同步的业务数据特点，把具有因果关系的数据哈希到相同分区上，才能避免因为并发乱序而出现数据同步错误的问题。

思考题

在我们这种数据同步架构下，如果说下游的某个同步程序或数据库出了问题，需要把Binlog回退到某个时间点然后重新同步，这个问题该怎么解决？欢迎你在留言区与我讨论。

感谢你的阅读，如果你觉得今天的内容对你有帮助，也欢迎把它分享给你的朋友。

精选留言：

- 李玥 2020-04-09 10:37:33
  Hi，我是李玥。

  这里回顾一下上节课的思考题：

对象存储并不是基于日志来进行主从复制的。假设我们的对象存储是一主二从三个副本，采用半同步方式复制数据，也就是主副本和任意一个从副本更新成功后，就给客户端返回成功响应。主副本所在节点宕机之后，这两个从副本中，至少有一个副本上的数据是和宕机的主副本上一样的，我们需要找到这个副本作为新的主副本，才能保证宕机不丢数据。但是没有了日志，如果这两个从副本上的数据不一样，我们如何确定哪个上面的数据是和主副本一样新呢？

这个问题有些同学已经在留言区给出了答案，一般都是基于版本号来解决，在Leader上，KEY每更新一次，KEY的版本号就加1，版本号作为KV的一个属性，一并复制到从节点上，通过比较版本号就可以知道哪个节点上的数据是最新的。

另外，有的同学提出用比较时间戳的方式来解决这个问题。这个方法理论上可行，但实际上非常难实现，因为它要求集群上的每个节点的时钟都必须时刻保持同步，这个要求往往非常难达到。

- 豆腐居士 2020-04-09 12:07:26
  如果预估了分区（队列）数量之后 随着业务数据的增长 需要增加分区 提高并发 怎么去做扩容？
  因为统一笔订单需要打到同一个分区上 [1赞]

  作者回复2020-04-09 18:05:37
  1. 停掉Canel；
  2. 等MQ中所有的消息都消费完了。
  3. 扩容MQ分区数，增加消费者实例数量。
  4. 重新启动Canel。

- 丁小明 2020-04-10 08:16:00
  mysql本身的多线程主从同步也是基于这个原理吧

- Simon 2020-04-09 20:49:27
  老师，请问下都用mq了还能是实时同步数据嘛？

- 此方彼方Francis 2020-04-09 11:42:30
  这节课感觉可以和MySQL同步数据到redis那节合起来。

- 一步 2020-04-09 11:24:32
  确定出现问题的时间点，然后把这个时间点之后的数据，全部再重新处理对应的 binlog ，然后再发送到对应的MQ中

- 那时刻 2020-04-09 10:10:28
  使用MQ的消息回放功能？

- yasin 2020-04-09 09:30:05
  订单号的id应该是时间序列递增的，将回退那个时间点的id找到，删除下游表里大于这个id的记录，然后再进行同步。

- 饭饭 2020-04-09 08:29:05
  非mysql 同步呢，mysqlsever oracle 是否有对应功能或同步工具？

  作者回复2020-04-09 18:42:56
  oracle 也有类似的归档日志。