

20 | 如何在不停机的情况下，安全地更换数据库？

李玥 · 后端存储实战课



你好，我是李玥。

随着我们的系统规模逐渐增长，总会遇到需要更换数据库的问题。我们来说几种常见的情况。

对 MySQL 做了分库分表之后，需要从原来的单实例数据库迁移到新的数据库集群上。

系统从传统部署方式向云上迁移的时候，也需要从自建的数据库迁移到云数据库上。

一些在线分析类的系统，MySQL 性能不够用的时候，就需要更换成一些专门的分析类数据库，比如说 HBase。

更换数据库这个事儿，是一个非常大的技术挑战，因为我们需要保证整个迁移过程中，既不能长时间停服，也不能丢数据。

那么，今天这节课我们就来说一下，如何在不停机的情况下，安全地迁移数据更换数据库。

如何实现不停机更换数据库？

我们都知道墨菲定律：“如果事情有变坏的可能，不管这种可能性有多小，它总会发生。”放到这里呢，也就是说，我们在更换数据库的过程中，只要有一点儿可能会出问题的地方，哪怕是出现问题的概率非常小，它总会出问题。

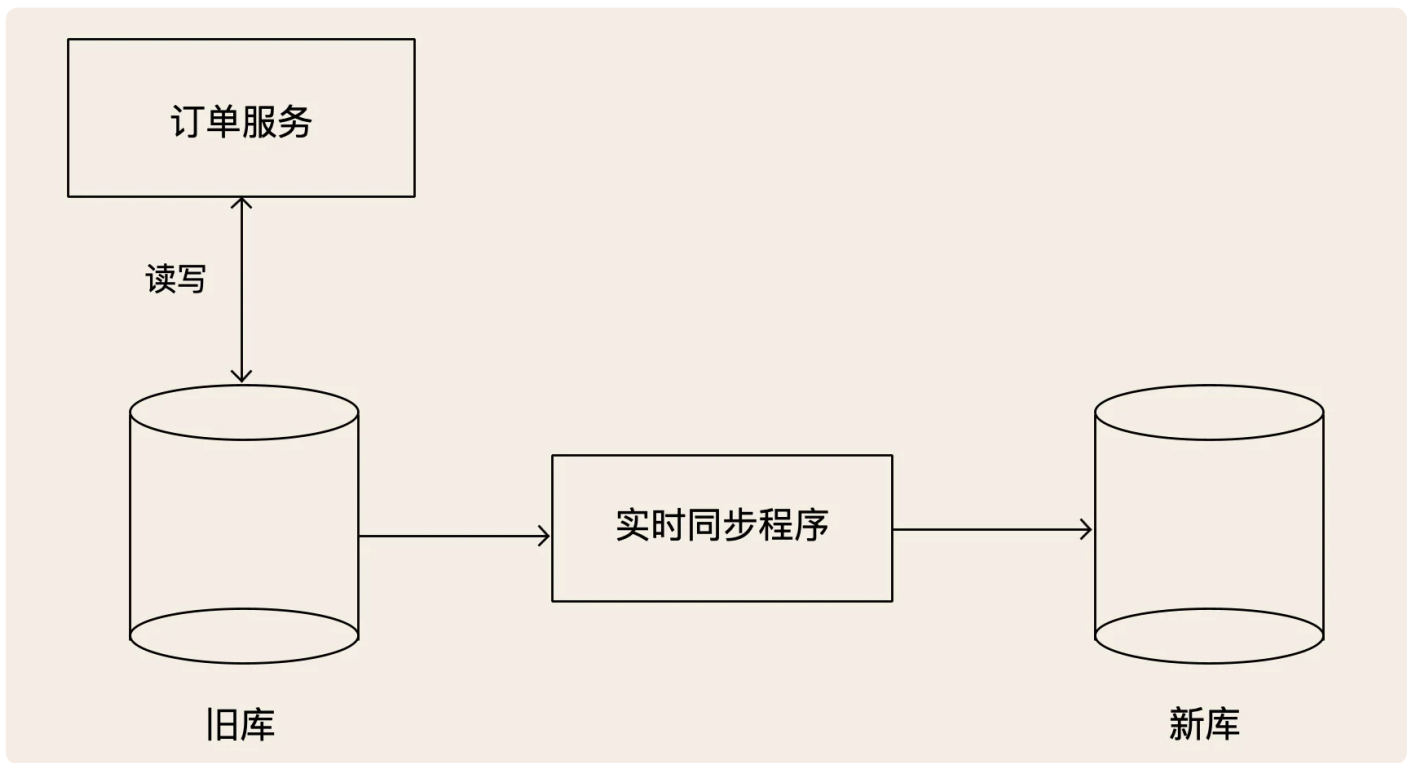
实际上，无论是新版本的程序，还是新的数据库，即使我们做了严格的验证测试，做了高可用方案，刚刚上线的系统，它的稳定性总是没有那么好的，需要一个磨合的过程，才能逐步达到一个稳定的状态，这是一个客观规律。这个过程中一旦出现故障，如果不能及时恢复，造成的损失往往是我们承担不起的。

所以我们在设计迁移方案的时候，一定要做到，每一步都是可逆的。**要保证，每执行一个步骤后，一旦出现问题，能快速地回滚到上一个步骤。**这是很多同学在设计这种升级类技术方案的时候，容易忽略的问题。

接下来我们还是以订单库为例子，说一下这个迁移方案应该如何来设计。

首先要做的就是，把旧库的数据复制到新库中。因为旧库还在服务线上业务，所以不断会有订单数据写入旧库，我们不仅要往新库复制数据，还要保证新旧两个库的数据是实时同步的。所以，我们需要用一个同步程序来实现新旧两个数据库实时同步。

怎么来实现两个异构数据库之间的数据实时同步，这个方法我们上节课刚刚讲过，我们可以使用 Binlog 实时同步数据。如果源库不是 MySQL 的话，就麻烦一点儿，但也可以参考我们讲过的，复制状态机理论来实现。这一步不需要回滚，原因是，只增加了一个新库和一个同步程序，对系统的旧库和程序都没有任何改变。即使新上线的同步程序影响到了旧库，只要停掉同步程序就可以了。



然后，我们需要改造一下订单服务，业务逻辑部分不需要变，DAO 层需要做如下改造：

1. 支持双写新旧两个库，并且预留热切换开关，能通过开关控制三种写状态：只写旧库、只写新库和同步双写。
2. 支持读新旧两个库，同样预留热切换开关，控制读旧库还是新库。

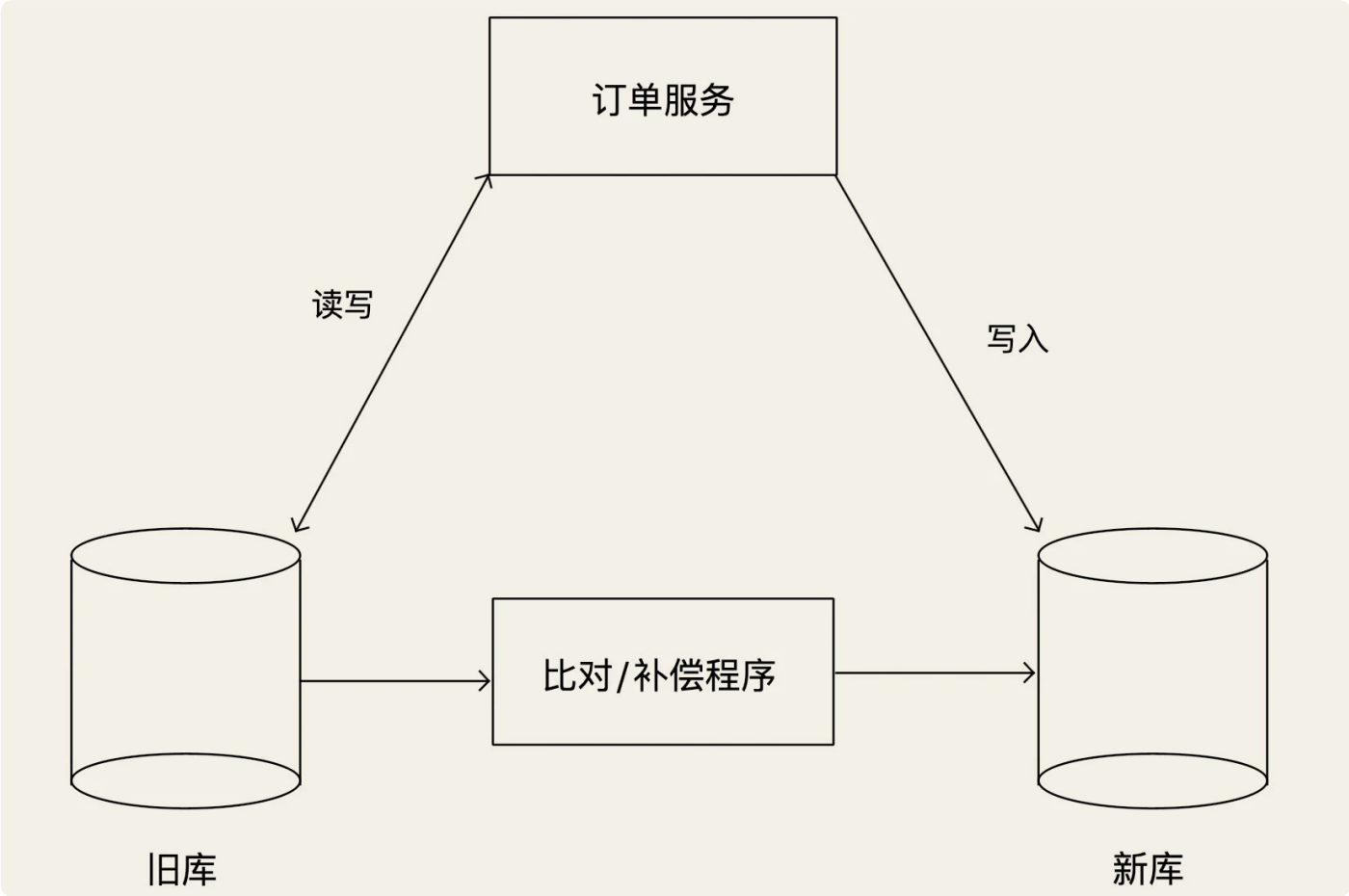
然后上线新版的订单服务，这个时候订单服务仍然是只读写旧库，不读写新库。让这个新版的订单服务需要稳定运行至少一到二周的时间，期间除了验证新版订单服务的稳定性以外，还要验证新旧两个订单库中的数据是否是一致的。这个过程中，如果新版订单服务有问题，可以立即下线新版订单服务，回滚到旧版本的订单服务。

稳定一段时间之后，就可以开启订单服务的双写开关了。开启双写开关的同时，需要停掉同步程序。这里面有一个问题需要注意一下，就是**这个双写的业务逻辑，一定是先写旧库，再写新库，并且以写旧库的结果为准。**

旧库写成功，新库写失败，返回写成功，但这个时候要记录日志，后续我们会用到这个日志来验证新库是否还有问题。旧库写失败，直接返回失败，就不写新库了。这么做的原因是，不能

让新库影响到现有业务的可用性和数据准确性。上面这个过程如果出现问题，可以关闭双写，回滚到只读写旧库的状态。

切换到双写之后，新库与旧库的数据可能会存在不一致的情况，原因有两个：一是停止同步程序和开启双写，这两个过程很难做到无缝衔接，二是双写的策略也不保证新旧库强一致，这时候我们需要上线一个对比和补偿的程序，这个程序对比旧库最近的数据变更，然后检查新库中的数据是否一致，如果不一致，还要进行补偿。



开启双写后，还需要至少稳定运行至少几周的时间，并且期间我们要不断地检查，确保不能有旧库写成功，新库写失败的情况出现。对比程序也没有发现新旧两个库的数据有不一致的情况，这个时候，我们就可以认为，新旧两个库的数据是一直保持同步的。

接下来就可以用类似灰度发布的方式，把读请求一点儿一点儿地切到新库上。同样，期间如果出问题的话，可以再切回旧库。全部读请求都切换到新库上之后，这个时候其实读写请求就已经都切换到新库上了，实际的切换已经完成了，但还有后续的收尾步骤。

再稳定一段时间之后，就可以停掉对比程序，把订单服务的写状态改为只写新库。到这里，旧库就可以下线了。注意，整个迁移过程中，只有这个步骤是不可逆的。但是，这步的主要操作就是摘掉已经不再使用的旧库，对于在用的新库并没有什么改变，实际出问题的可能性已经非常小了。

到这里，我们就完成了在线更换数据库的全部流程。双写版本的订单服务也就完成了它的历史使命，可以在下一次升级订单服务版本的时候，下线双写功能。

如何实现对比和补偿程序？

在上面的整个切换过程中，如何实现这个对比和补偿程序，是整个这个切换设计方案中的一个难点。这个对比和补偿程序的难度在于，我们要对比的是两个都在随时变换的数据库中的数据。这种情况下，我们没有类似复制状态机这样理论上严谨实际操作还很简单的方法，来实现对比和补偿。但还是可以根据业务数据的实际情况，来针对性地实现对比和补偿，经过一段时间，把新旧两个数据库的差异，逐渐收敛到一致。

像订单这类时效性强的数据，是比较好对比和补偿的。因为订单一旦完成之后，就几乎不会再变了，那我们的对比和补偿程序，就可以依据订单完成时间，每次只对比这个时间窗口内完成的订单。补偿的逻辑也很简单，发现不一致的情况后，直接用旧库的订单数据覆盖新库的订单数据就可以了。

这样，切换双写期间，少量不一致的订单数据，等到订单完成之后，会被补偿程序修正。后续只要不是双写的时候，新库频繁写入失败，就可以保证两个库的数据完全一致。

比较麻烦的是更一般的情况，比如像商品信息这类数据，随时都有可能会变化。如果说数据上有更新时间，那我们的对比程序可以利用这个更新时间，每次在旧库取一个更新时间窗口内的数据，去新库上找相同主键的数据进行对比，发现数据不一致，还要对比一下更新时间。如果新库数据的更新时间晚于旧库数据，那可能是对比期间数据发生了变化，这种情况暂时不要补偿，放到下个时间窗口去继续对比。另外，时间窗口的结束时间，不要选取当前时间，而是要比当前时间早一点儿，比如 1 分钟前，避免去对比正在写入的数据。

如果数据连时间戳也没有，那只能去旧库读取 Binlog，获取数据变化，然后去新库对比和补偿。

有一点需要说明的是，上面这些方法，如果严格推敲，都不是百分之百严谨的，都不能保证在任何情况下，经过对比和补偿后，新库的数据和旧库就是完全一样的。但是，在大多数情况下，这些实践方法还是可以有效地收敛新旧两个库的数据差异，你可以酌情采用。

小结

设计在线切换数据库的技术方案，首先要保证安全性，确保每一个步骤一旦失败，都可以快速回滚。此外，还要确保迁移过程中不丢数据，这主要是依靠实时同步程序和对比补偿程序来实现。

我把这个复杂的切换过程的要点，按照顺序总结成下面这个列表，供你参考：

1. 上线同步程序，从旧库中复制数据到新库中，并实时保持同步；
2. 上线双写订单服务，只读写旧库；
3. 开启双写，同时停止同步程序；
4. 开启对比和补偿程序，确保新旧数据库数据完全一样；
5. 逐步切量读请求到新库上；
6. 下线对比补偿程序，关闭双写，读写都切换到新库上；
7. 下线旧库和订单服务的双写功能。

思考题

我们整个切换的方案中，只有一个步骤是不可逆的，就是由双写切换为单写新库这一步。如果说不计成本，如何修改我们的迁移方案，让这一步也能做到快速回滚？你可以思考一下这个问题，欢迎你在留言区与我交流讨论。

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。

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

精选留言 (32)



李玥 置顶

2020-04-13

Hi, 我是李玥。

这里回顾一下上节课的思考题：

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？

这个问题的解决方案是这样的。如果说，下游只有一个同步程序，那直接按照时间重置Canal实例的位点就可以了。但是，如果MQ的下游有多个消费者，这个时候就不能重置Canal里的位点了，否则会影响到其它的消费者。正确的做法是，在MQ的消费订阅上按照时间重置位点，这样只影响出问题的那个订阅。所以，这种架构下，MQ中的消息，最好将保存时间设置得长一些，比如保留3天。

共 1 条评论 >

👍 34



hlllllllll

2020-05-05

在16年去o时设计的一个方案和这个很相似。需要将56亿oracle的财务数据迁移到mysql中并保证一直提供稳定的线上服务。方案主要涉及以下几点：

- 1.设计同步worker以6亿/天的速度同步至mysql，预先按照1000的间隔生成560w个任务，用于保证数据同步不丢失。
- 2.mysql业务数据库库为了防止和oracle中id碰撞，设计id生成器 其中，前三位为库编号；
- 3.因为高达100w/分钟的事务量，设计了前置防重表+任务表（按照uuid分库），用于防重、规避热点和异步增加吞吐量；
- 4.业务数据库通过在前置库生成的id分库，同时会异步同步至es；
- 5.原接口服务改造，写老库数据和发送mq在一个事务（mq挂了会影响服务）。提供新接口服务写新库并发mq；
- 6.接收端消费mq并根据标示同步至老库或新库；
- 7.提前定义好一个阈值单号，比如当前数据已经累积到57亿，那我们可以根据增速估算一个阈值58亿。在上线后只有58亿以后的数据会通过mq同步至老库的。以前的数据都由worker同步（在适当时候：如目前单号已超过58亿，生成另外20w个任务，并开始执行另外两亿数据的同步，期间新库会缺少这两亿数据）；

8.在mq同步稳定一段时间后，用新接口灰度替代老接口，在整体稳定后全量切换至新接口，并用mq同步老库。

共 4 条评论 >

👍 60



菠萝吹雪—Code

2020-04-15

无知者总觉得这样做麻烦，智者满满都是细节

共 2 条评论 >

👍 23



靠人品去赢

2020-04-13

然而我们小公司，或者比较传统的行业直接还是一个维护系统的公告，然后大晚上，凌晨加班。

共 4 条评论 >

👍 20



myrfy

2020-04-13

不计成本的话，搭建一套全新的影子服务，在整个系统接入层做流量双写分发，切换时影子系统变为主系统，原主系统变为影子系统

共 1 条评论 >

👍 11



等风来👂

2020-05-31

老师，这个热切开关具体什么方式实现呢？

作者回复: 这个一般都是通过编码来实现的。具体触发的方法，可以对外暴露一个可供调用的接口，或者通过动态配置下发等方式来触发。

共 2 条评论 >

👍 10



nfx

2020-05-25

抱歉上个留言没说说清楚，请问老师线上大表怎么在不影响业务的情况下增加字段？我能想到的一个办法是在从库增加字段，等从库同步追上来的时候切换主从。 请问还有没有其他办法？

另外线上扩容怎么做？是不是和这节课更换数据库的方法一样？加个从库，同步追上来后分库分表？

作者回复: 大表加字段过程中会锁表，期间所有写操作都会阻塞。如果能接受的话，直接加还是最方便的。

你提的主从切换的方式，我没有试过，理论上也是可行的。但需要特别注意切换过程中的数据一致性。

线上扩容，相当于更换数据库，建议参照这节课中的方法操作。

共 3 条评论 >

👍 6



锐

2020-04-11

借楼请教一下李老师，无限层级(每条记录都对应一个父id)怎样设计能够快速查询，之前设计是存一个path字段，用like查询，总感觉这样设计不优雅。

作者回复: 可以再把问题具体话一下么？我们可以一起来讨论一下。

共 4 条评论 >

👍 4



林铭铭

2021-07-29

半夜停机升级下，会舒服点



👍 2



趁早

2020-08-20

开启双写和停止数据同步服务这一步需要停服或者锁库，不然会有数据不一致风险

共 1 条评论 >

👍 2



百威

2020-04-21

有个问题，既然有比对和补偿程序，可不可以不使用数据实时同步。首先上线观察双写和补偿程序，没问题后先进行数据从旧到新的快照复制，然后开启双写，因有缝衔接而丢失的数据通过补偿程序来做.....求教～

作者回复: 考虑到很难实现一个完美的对比补偿程序, 还是建议不要这么做。



👍 2



Jxin

2020-04-11

回答:

1.主要得解决, 断开同步双写后, 只写在新库的这部分数据如何同步到旧库, 要严谨其实还得保证同步(这也是要用同步双写的主要原因)。讲真, 水平有限, 放弃同步双写后还要支持实时的同步数据真没招。那么让切换开关同时开启新库增量数据异步同步到旧库。感觉得加锁保切开关和开启增量同步两个操作原子性。严格来说还是有一会儿的停服现象(db不可用)。而且异步数据是不实时的, 切回旧库还是可能因为数据未同步完, 导致数据异常。比如退款一次成功了, 切回来时数据更新还没做, 那就还能发起一次退款申请。

疑问:

1.双写时, 对账系统校对两个数据库数据是否一致时, 要么比对一段时间内单表新增的数据行数, 要么比对最新的订单是否一致。但不会去比对每一条数据是否一致。所以如果是数据更新还是可能漏了。

2.感觉如果不停服, 其实方案都很难严谨。只能做好自动校对自动补偿的系统, 在切换后, 尽快回复数据一致的现象。 所以午夜干稳妥。



👍 2



lcf枫

2020-08-17

老师, 自增ID 在新旧库不同是否会出现问题? 新库应该还要一个唯一的ID生成器?

共 1 条评论 >

👍 1



小水

2020-07-16

老师, 那在迁移数据库的时候, 前置条件就得有个全局业务流水号发号器来做新旧库的主键, 如果用自带的MySQL的自增主键就可能导致新旧两个库插入数据时主键不一致的情况, 我想到的就是用全局发号器来做插入的id值, 不知道我的想法是否正确



👍 1



nfx

2020-05-22

请问老师，修改线上数据库表结构怎么处理？

我原来方案是建个新表，把旧数据倒过去，然后新旧表分别改名。

但数据一致问题比较麻烦，改名过程中也容易出错

作者回复: 这个问题是非常难处理的，所以，一般来说很少去删除表的字段，只增加字段。这样，新表能够兼容旧表，也就不用迁移数据了。

共 2 条评论>



1



me不是一个人战斗

2020-04-28

对于双写阶段，会不会存在两份数据id不一致的情况（比如mysql的自增ID），如果下游有依赖这个表的ID，一旦切换就没办法回退了

作者回复: 这种建议以一边的自增ID为准。



2



ifelse

2022-12-14 来自浙江

要保证，每执行一个步骤后，一旦出现问题，能快速地回滚到上一个步骤。--记下来



千锤百炼领悟之极限

2022-03-04

- 1.创建新库，全量复制一次旧库数据，上线新旧同步程序，开启同步。
- 2.上线支持随时切换双写，单写的程序版本，暂时只读写旧库。
- 3.开启双写，停止新旧同步程序。
- 4.开启新旧对比与补偿程序。
- 5.逐步切换读请求到新库。
- 6.关闭双写，只写新库。这时读写都是新库。关闭新旧对比与补偿程序。
- 7.下线旧库。



请叫我和尚

2022-03-02

监听新库的 Binlog 同步到旧库里！

也可以把对比/补偿程序以新库为准去补偿到旧库里



凯文小猪

2021-12-24

二刷说下自己的看法：在切库这一步和老师不同

- 1.上线写程序 上线时开关模式为双写，如果此时新库有问题 立刻通过开关将其只写老库
- 2.记录新库第一条写入的时间 通知DBA 或者自己写程序，以新库第一条时间为准开始同步
- 3.上线对账程序 自动比对数据 && 修复数据
- 4.切读流量至新库 待读流程切换完毕 以为新库开始运行于线上
- 5.观察无误后（可能需要1至2个礼拜） 切换写库

这里与老师方案的区别在于：

上线新库时 因为是先直接上线双线模式，所以同步新库剩余的数据是可反复幂等操作的，且因为数据的写入点是定格的 故同步旧数据可保证不丢 不多不少

而老师的方案 是先上线canal等同步程序，打开双写后再关闭同步程序，但同步程序或是binlog默认并不是从头开始，而是从当下开始对新库累计数据，开始同步之前的旧数据如果同步过来呢？

