链滴

# XA 与两阶段提交

作者：skyesx

原文链接：https://ld246.com/article/1448896381093

来源网站：链滴

先讲讲两阶段提交把，两阶段提交顾名思义分为两个阶段，一个是准备阶段，一个是提交阶段。
在两阶段提交中有三个参与者，一个是应用程序（向事务管理器提交 commit或者rollback请求，一个是事务管理器（接收应用程序的请求，并协调 多个资源管理器 进行事务提交或者回滚），还一个就是 资源管理器咯（管理单个资源——数据库，JMS等的事务）
有一个经典的对两阶段的比喻，就是结婚的仪式：
牧师（事务管理器）分别询问新娘、新郎（资源管理器）你是否愿意与对方结婚（第一阶段开始，如果两个人都回答 I do（第一阶段结束，第二阶段开始），那么牧师就会向两位新人宣布，你们的姻关系成立了~（第二阶段结束）<br />如果有异常情况，有一个人say no的话，那么，就结不了婚~
两阶段提交的第一阶段存在的意义是 让对应的资源做好一切提交的准备，如果提交不了的话，以尽快暴露出来（如磁盘空间不足什么的），告诉协调器，让大家一起回滚。
 
一些两阶段提交协议的实现里会有 “最后参与者支持（Last Participant Support ” 或者 “最后资源提交优化（Last Resource Commit Optimization）”的支持，这个支持允许有 一个 非XA支持的资源参与到全局事务中来，具体是么实现的~
还是结婚的栗子，不过我们假设中间多了个民政局（非XA资源），在新郎新娘都SAY I DO后（X的第一阶段结束），牧师问民政局你是否愿意给他们颁发结婚证呀（让非XA资源提交），若民政局颁证书了（非资源XA提交成功），牧师则宣布婚姻关系成立，若民政局说，你们是失散多年的亲兄妹！吧，牧师只能说，你们的婚姻关系不成立，散了吧（XA资源回滚）。
上述这个优化项并不建议使用，因为在第一阶段和第二阶段之间必须串行的接入第三阶段，这就加了事务执行的时间，有一句老话大家都知道的，叫夜长梦多，尤其是在这种网络环境，并且 这个实 只是 一些 两阶段提交的实现中支持...请理解 一些 的含义...
 
当然，写两阶段协议实现的人都跟你我一样聪明，当事务参与者只有一个的时候，他会从两阶段交变成一阶段提交，省去询问的提交准备的步骤
 
 
现在我们对两阶段提交有了一个大概整体的了解了，下面我们来分析一下本协议在网络环境中是何保持一致性。
在协调者发出COMMIT前，参与者总是能通过ROLLBACK恢复到一致性状态的，在此，不再描。我们着重描述进入 “第二阶段”， 协调者开始发送COMMIT令的场景
有三个角色 协调者C 部分参与者A 部分参与者B ， A与B加起来就是所有参与者。3个元素任意自组成组合，一个元素可归属多个组合，组合内元素能正常互访，不同组合元素无法互访（对方挂了，者与对方的网络中断），那么有以下的组成：
一、总共只有一个组合的情况：
1、ABC三个组成一个组合，这表示ABC都能互相正常互联，两阶段提交正常运行
二、总共只有两个组合的情况：
1) C,AB——参与者可以询问包括自己在内的所有参与者 可提交状态，从而判定该执行提交还是回滚的结果，从而保证强一致性及可用性。
2) CA,B——A的数据能够明确知道执行结果，能执行回滚或者提交，这部分保持一致性及可用性。而B因为无从获得提交结果，因此只能阻塞。
3) CB,A——与2)类似。
4) CA,CB——AB都能获得最终的提交结果,能保证一致性。由于消息是可以传递，因此这个组合等价于 “一、”中全部可互联的情况
5) AC,AB——A能够从C中获得最终的提交结果，B能够从A中获得最终提交结果因此最终能达到强一致性也保证了可用性。本质上跟4）一致。由于消息是可以传递的，因此这个组等价于 “一、”中全部可互联的情况
6) BC,BA——与4）类似
总共只有三个组合的情况：
1)A,B,C——好吧，肯定都阻塞，没有异议把....
2)其他的组合……——存在三个或以上的组合的话，无论怎么分通过

息传递后都必然等价于上述已经存在的情况，无需再讨论
 
因此，总结一下，两阶段提交在 部分资源管理器 无法与 协调者互联 且无法与 其他资源管理器联时，只能阻塞等待 互联恢复，完全失去了 可用性。但在其他情况下，还是能较好的处理异常，消部分可用性（从其他资源管理器中获得提交结果，增加了耗时）来维持一致性的。
 
 
好吧，到现在两阶段提交的概念以及一些异常的处理应该都讲的差不多了，那么XA与两阶段提是啥关系？其实关系很简单，可能大家都猜出来了，XA是基于两阶段提交设计的一个接口标准，实现这个XA接口的资源管理器那么就能参与到XA管控的全局事务中~好咯，本章完毕 ~
 
 
 
 
 