2PC

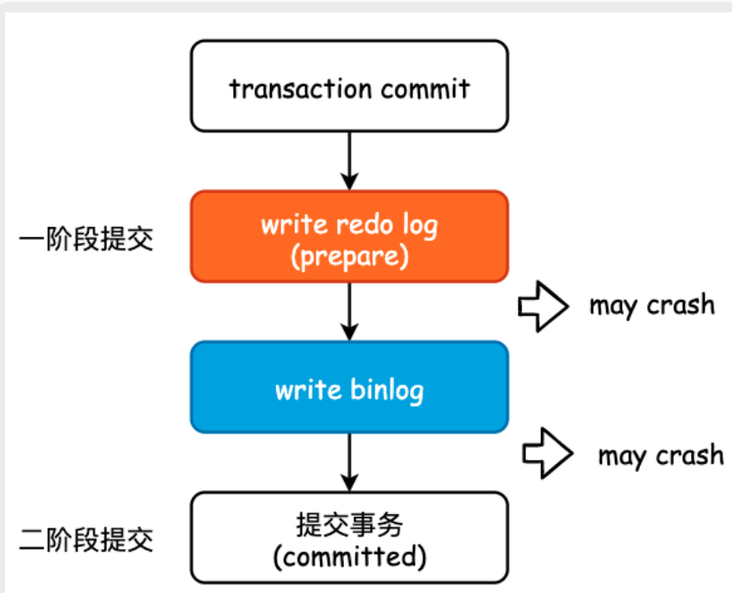
MySQL 中的 2PC

2PC，即 Two Phase Commit，翻译为两阶段提交协议，将事务流程分成准备阶段（Prepare）和提交阶段（Commit），用于保证事务执行时数据的强一致性

一个最被人熟知的两阶段提交就是 MySQL redo log 和 binlog

由于 InnoDB 存储引擎使用 B+Tree 作为底层索引结构，在写入数据时无法保证顺序写入，因此引入了 redo log 作为 WAL，用于提高数据的写入效率。同时 MySQL 使用 binlog 这一逻辑日志实现数据的复制与灾难恢复

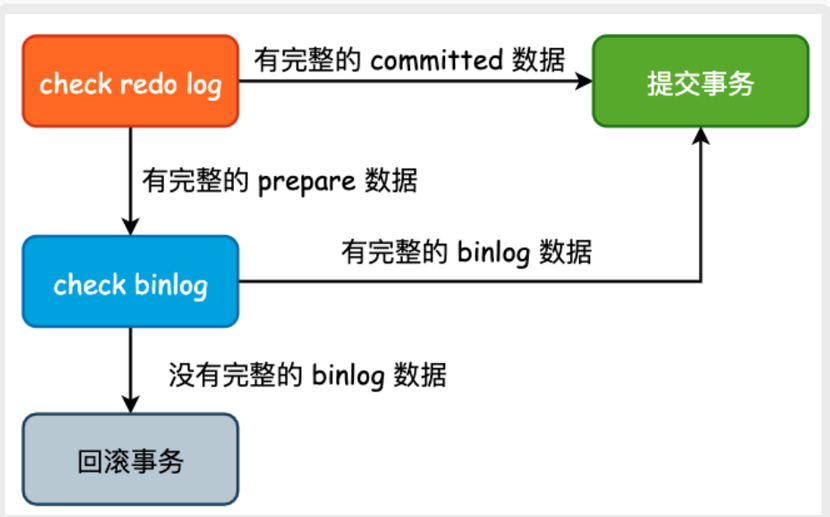
redo log 和 binlog 属于不同的文件，在事务执行时，既要写 redo log，又要写 binlog，那么如何保证这两个文件的一致性就是 2PC 需要解决的问题



如左图所示，假设在没有 Group Commit 的前提下，一次事务的提交需要两次 fsync() 系统调用

现在来具体分析一下 2PC 为什么能保证 redo log 和 binlog 的强一致性。在上图中标注了两个 crash 的时刻，一个是写完 redo log 之后、写 binlog 之前，另一个则是写完 binlog 之后、提交事务之前

1. redo log 之后、binlog 之前崩溃。此时未写入 binlog， redo log 也没有提交，因此崩溃恢复之后 InnoDB 会回滚该事务，并且由于 binlog 中没有这部分的数据，所以从库也不会有该事务的数据
2. binlog 之后、committed 之前崩溃。此时 binlog 数据完整，那么在崩溃恢复之后 InnoDB 会提交该事务。此时不管是使用异步复制还是半同步复制，从库都会收到该事务的数据并重放，主从之间的数据是一致的

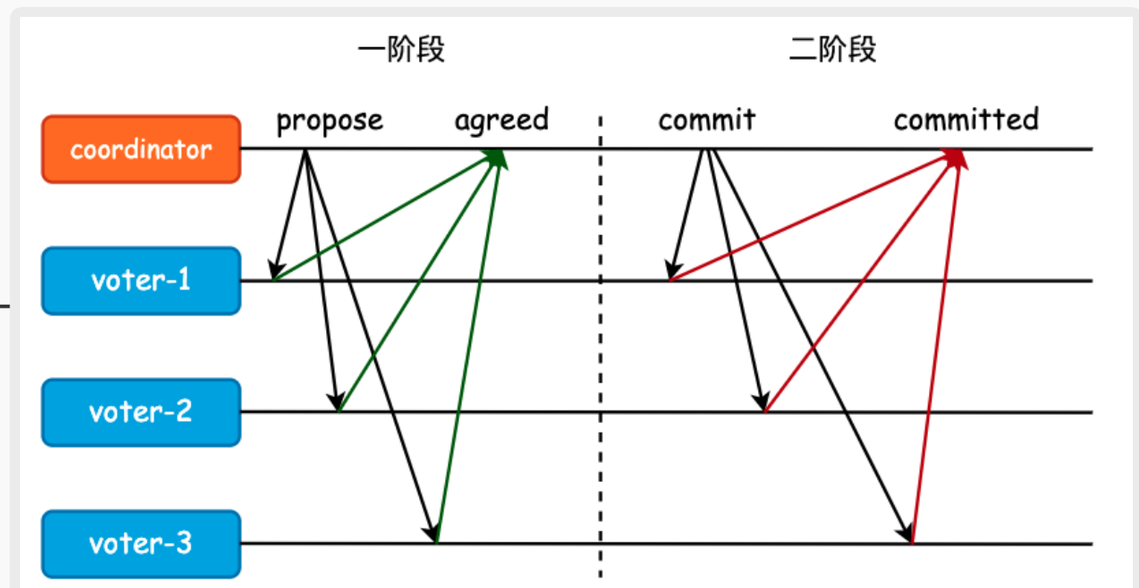


左图为 InnoDB 在崩溃恢复时的判定规则

可以看到，2PC 保证的并不是 redo log 和 binlog 这两个文件的一致性性，而是数据的一致性。就算这两个文件在某些时刻并不一致，InnoDB 依然能够保证事务的一致性

2PC 是分布式系统中非常重要的协议，不管是在 paxos 协议还是在 Raft 协议中都会使用 2PC 来保证数据的一致性

分布式系统中的 2PC



上图是一个通用的 2PC 模型，将图中的 voter 替换成集群中对应的节点即可。在 Paxos、Raft 等共识算法中，一阶段和二阶段只需要得到超过半数节点的成功即可。而在分布式事务中，则必须得到全部节点的同意，否则就会出现数据的不一致

Reference

KLEPPMANN MARTIN, 数据密集型应用系统设计[M]. 赵军平，李三平，吕云松，耿煜，译. 北京：中国电力出版社，2018

Gregor Hohpe: "Your Coffee Shop Doesn't Use Two-Phase Commit," IEEE Software, volume 22, number 2, pages 64–66, March 2005. doi:10.1109/MS.2005.52

姜承尧，MySQL技术内幕：InnoDB存储引擎[M]. 北京：机械工业出版社，2013

Wikipedia: Two-phase commit protocol, https://en.wikipedia.org/wiki/Two-phase_commit_protocol

Wikipedia: Redo log, https://en.wikipedia.org/wiki/Redo_log