（1）CanCommit 阶段

3PC 的 CanCommit 阶段其实和 2PC 的准备阶段很像。协调者向参与者发送 commit 请求，参与者如果可以提交就返回 Yes 响应，否则返回 No 响应。

- 事务询问：协调者向参与者发送 CanCommit 请求。询问是否可以执行事务提交操作。然后开始等待参与者的响应。
- 响应反馈： 参与者接到 CanCommit 请求之后，正常情况下，如果其自身认为可以顺利执行事务，则返回 Yes 响应，并进入预备状态。否则反馈 No

（2）PreCommit 阶段

协调者根据参与者的反应情况来决定是否可以记性事务的 PreCommit 操作。根据响应情况，有以下两种可能。

- 假如协调者从所有的参与者获得的反馈都是 Yes 响应，那么就会执行事务的预执行。
  - 发送预提交请求 协调者向参与者发送 PreCommit 请求，并进入 Prepared 阶段。
  - 事务预提交 参与者接收到 PreCommit 请求后，会执行事务操作，并将 undo 和 redo 信息记录到事务日志中。
  - 响应反馈 如果参与者成功的执行了事务操作，则返回 ACK 响应，同时开始等待最终指令。
- 假如有任何一个参与者向协调者发送了 No 响应，或者等待超时之后，协调者都没有接到参与者的响应，那么就执行事务的中断。
  - 发送中断请求：协调者向所有参与者发送 abort 请求。
  - 中断事务：参与者收到来自协调者的 abort 请求之后（或超时之后，仍未收到协调者的请求），执行事务的中断。

（3）doCommit 阶段

该阶段进行真正的事务提交，也可以分为以下两种情况：

- 执行提交
  - 发送提交请求：协调接收到参与者发送的 ACK 响应，那么他将从预提交状态进入到提交状态。并向所有参与者发送 doCommit 请求。
  - 事务提交：参与者接收到 doCommit 请求之后，执行正式的事务提交。并在完成事务提交之后释放所有事务资源。
  - 响应反馈：事务提交完之后，向协调者发送 Ack 响应。
  - 完成事务：协调者接收到所有参与者的 ack 响应之后，完成事务。
- 中断事务

协调者没有接收到参与者发送的 ACK 响应（可能是接受者发送的不是 ACK 响应，也可能响应超时），那么就会执行中断事务。

- 发送中断请求：协调者向所有参与者发送 abort 请求

✧事务回滚：参与者接收到 abort 请求之后，利用其在阶段二记录的 undo 信息来执行事务的回滚操作，并在完成回滚之后释放所有的事务资源。

✧反馈结果：参与者完成事务回滚之后，向协调者发送 ACK 消息

✧中断事务：协调者接收到参与者反馈的 ACK 消息之后，执行事务的中断。

在 doCommit 阶段，如果参与者无法及时接收到来自协调者的 doCommit 或者 rebort 请求时，会在等待超时之后，会继续进行事务的提交。（其实这个应该是基于概率来决定的，当进入第三阶段时，说明参与者在第二阶段已经收到了 PreCommit 请求，那么协调者产生 PreCommit 请求的前提条件是他在第二阶段开始之前，收到所有参与者的 CanCommit 响应都是 Yes。（一旦参与者收到了 PreCommit，意味他知道大家其实都同意修改了）所以，一句话概括就是，当进入第三阶段时，由于网络超时等原因，虽然参与者没有收到 commit 或者 abort 响应，但是他有理由相信：成功提交的概率很大。 ）