1   Two Phase Commit with Logging

1. In the two-phase commit protocol, suppose that the coordinator sends PREPARE messages and crashes before receiving any votes. Answer the following questions.

   (a) What is the sequence of operations at the coordinator after it recovers?

   After the coordinator restarts, the recovery process will find that a transaction was executing at the time of the crash and that no commit protocol log record had been written (remember that the coordinator does not write any log records before sending PREPARE messages).

   The recovery process will abort the transaction locally (not sending an ABORT message) by undoing its actions, if any, using the UNDO log records, and writing an abort record.

   (b) What is the sequence of operations at a participant that received the message and replied No before the coordinator crashed?

   If the participant sent a NO vote, it knows that the transaction will be aborted because a NO vote acts like a veto. The participant does not care if the coordinator crashes or not. It aborts the local effects of the transaction.

   (c) What is the sequence of operations at a participant that received the message and replied Yes before the coordinator crashed?

   If the participant sent a YES vote, it cannot make any unilateral decisions. If the participant notices the failure of the coordinator (for example by using a timeout), it hands the transaction over to the recovery process. The recovery process will find that it is in the prepared state for the transaction. It will periodically try to contact the coordinator site to find out how the transaction should be resolved.

   As we discussed above, after the coordinator recovers, it will abort the transaction and will answer "abort" upon receiving an inquiry message. The participant will then abort the transaction.

2. Now in the two-phase commit protocol, describe what happens if a subordinate receives a PREPARE message, replies with a YES vote, crashes, and restarts (All other participants also voted YES and didn't crash).

In this scenario, upon restarting, the recovery process will find that the participant is in the prepared state for the transaction. It will periodically try to contact the coordinator site to find out how the transaction should be resolved. In our scenario, the final outcome of the transaction is a commit. Because the coordinator cannot end a committed transaction until it receives final ACKS from all nodes, it will correctly respond with a COMMIT message to the inquiry. The participant will then be able to properly commit the transaction.

3. Suppose we have one coordinator and three participants. It takes 30ms for a coordinator to send messages to all participants; 5, 10, and 15ms for participant 1, 2, and 3 to send a message to the coordinator respectively; and 10ms for each machine to generate and flush a record. Assume for the same message, each participant receives it from the coordinator at the same time.

Under proper 2PC and logging protocols, how long does the whole 2PC process (from the beginning to the coordinator's final log flush) take for a successful commit in the best case?

130ms

Explanation:

Phase 1: 30 + 10 + 15 + 10 = 65ms
Coordinator sends prepare message + Participant generates and flushes prepare record + max time it takes a participant to send a Yes message + Coordinator generates and flushes commit record

Phase 2: 30 + 10 + 15 + 10 = 65ms
Coordinator sends commit message + Participant generates and flushes commit record + max time it takes a participant to send an Ack message + Coordinator generates and flushes end record

Total: 130ms