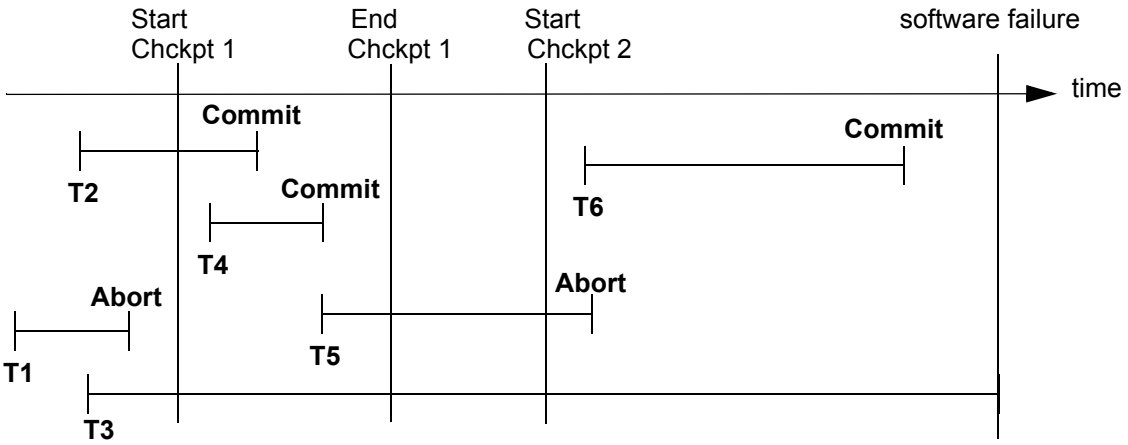
Exercise: Transactions / Recovery

1. Recovery with checkpoints

Consider the following scenario with a log file where fuzzy checkpointing is used:



a)
List which transactions the recovery manager must undo and which ones it must redo. Briefly explain your choice.

b)
Now assume that Checkpoint 2 had successfully finished just before the software failure occurred, after the Commit of T6. What is the answer of problem a) now? Briefly explain.

2. Using Savepoints

Test savepoints in MySQL by defining a transaction with several write actions. Set several savepoints and rollback to different savepoints.
Also test rollback of the entire transaction.

Exercise: Transactions / Recovery

1. Recovery with checkpoints

a)
T1: No recovery needed. Any dirty pages left over from the abort process of T1 were written by the end of checkpoint 1.

T2: must be redone, at least those changes that the transaction performed after the beginning of checkpoint 1, because it is not certain that those changes were transferred to the database disk. Checkpoint 2 would have taken care of that, but it was not finished.

T3: must be undone because it was not finished at the time of the crash.

T4: analogous to T2, the only difference is that for T2 those changes that happened before the start of checkpoint 2 are already safely on disk, whereas for T4, nothing is certain. Any dirty pages left by T4 were collected for flushing by checkpoint 2, but since it did not finish it is not certain, what was in fact written to disk. Hence the writing must be repeated.

T5: needs to be undone. It was aborted, but it is not certain whether the writing of before images has been finished at the time of the crash. The abort process started after the time when checkpoint 2 collected the list of dirty pages. So checkpoint 2 did nothing to flush dirty pages for this abort. Of course, the abort process might have written some or all pages by the time of the crash, but as this is not certain, it must be repeated.

T6: has committed and must be redone. Checkpoint 2 has done nothing for T6, so it is uncertain what has been written of T6‘s change and what hasn‘t.

b)
Now everything that ended before the beginning of checkpoint 2 is safely on disk because checkpoint 2 collected any pending write operations and flushed them. Thus, only T3, T5, and T6 require any recovery at all. As before, T3 must be undone as it was not finished. T5‘s abort happened after checkpoint 2‘s collection of dirty pages so the undo must be repeated. T6 has committed and has performed its writes after the beginning of checkpoint 2, so it must be redone.