HW14

16.2

Checkpointing is done with log-bassed recovery schemes to reduce the time required for recovery after a crash. If there is no checkpointing, then the entire log must be searched after a crash, and all transactions undone/redone from the log. If checkpointing had been performed, then most of the log-records prior to the checkpoint can be ignored at the time of recovery.

Another reason to perform checkpoints is to clear log-records from stable storage as it gets full.

Since checkpoints cause some loss in performance while they are being taken, their frequency should be reduced if fast recovery is not critical. If we need fast recovery checkpointing frequency should be increased. If the amount of stable storage available is less, frequent checkpointing is unavoidable.

Checkpoints have no effect on recovery from a disk crash; archival dumps are the equivalent of checkpoints for recovery from disk crashes.

16.5

a.

The old-value part of an update log record is not required. If the transaction has committed, then the old value is no longer necessary as there would be no need to undo the transaction. And if the transaction was active when the system crashed, the old values are still safe in the stable storage as they haven’t been modified yet.

b.

During the redo phase, the undo list need not be maintained any more, since the stable storage does not reflect updates due to any uncommitted transaction.

c.

A data item read will first issue a read request on the local memory of the transaction. If it is found there, it is returned. Otherwise, the item is loaded from the database buffer into the local memory of the transaction and then returned.

d.

If a single transaction performs a large number of updates, there is a possibility of the transaction running out of memory to store the local copies of the data items.

16.18

Answer: Recovery would happen as follows:

Redo phase:

a. Undo-List = T0, T1

b. Start from the checkpoint entry and perform the redo operation.

c. C = 600

d. T1 is removed from the Undo-list as there is a commit record.

e. T2 is added to the Undo-list on encountering the record.

f. A = 400

g. B = 2000

Undo phase:

a. Undo-list = T0, T2

b. Scan the log backwards from the end.

c. A = 500; output the redo-only record <T2, A, 500>

d. output

e. B = 2000; output the redo-only record <T0, B, 2000>

f. output

At the end of the recovery process, the state of the system is as follows:

A = 500

B = 2000

C = 600

The log records added during recovery are:

<T1, A, 500>

<T0, B, 2000>

Observe that B is set to 2000 by two log records, one created during normal rollback of T0, and the other created during recovery, when the abort of T0 is completed. Clearly the second one is redundant, although not incorrect. Optimizations described in the ARIES algorithm(and equivalent optimizations described in Section 16.7 for the case of logical operations) can help avoid carrying out redundant operations, which create such redundant log records.

16.20

Answer:

There is no check point in the log, so recovery starts from the beginning of the log, and replays each action that is found in the log.

The redo phase would add the following records:

<T0, B, 2050>

<T0, C, 600>

<T1, C, 400>

<T0, C, 500>

At the end of the redo phase, the undo list contains transactions T0 and T1, since their start records are found, but not their end of abort records. During the undo phase, scanning backwards in the log, the following events happen:

<T0, C, 400>

<T1, C, 600>

<T1, O2, operation-abort>

<T1, abort>

<T0, C, 700>

<T0, O1, operation-abort>

<T0, B, 2000>

<T0, abort>

Finally the values of data items B and C would be 2000 and 7000, which, were their original values before T0 or T1 started.

16.22

a.

If a page is not in the checkpoint dirty page table at the beginning of the analysis pass, redo records prior to the checkpoint record need not be applied to it as itmeans that the page has been flushed to disk and been removed from the DirtyPageTable before the checkpoint. However, the page may have been updated after the checkpoint, which means it will appear in the dirty page table at the end of the analysis pass.

For pages that appear in the checkpoint dirty page table, redo records prior to the checkpoint may also need to be applied.

b.

TheRecLSNis an entry in theDirtyPageTable,which reflects the LSN at the end of the log when the page was added to DirtyPageTable. During the redo pass of the ARIES algorithm, if the LSNof the update log record encountered, is less than the RecLSNof the page in DirtyPageTable, then that record is not redone but skipped. Further, the redo pass

starts at RedoLSN, which is the earliest of the RecLSNs among the entries in the checkpoint DirtyPageTable, since earlier log records would certainly not need to be redone. (If there are no dirty pages in the checkpoint, the RedoLSN is set to the LSN of the checkpoint log record.)