Weekly exercises IN3020 and IN4020
Week 12

April 24, 2022

1 Review

1. What do we mean with recovery?

   Solution: Restore a database to the most recent consistent state before failure; in the ACID settings, this includes ensure all effects of committed transactions and unroll all effects of aborted transactions

2. Why would we want to recover?

   Solution: We want to recover from failures, avoiding breaking any of the ACID properties of the database.

3. Discuss at least two types of failures?

   Solution: Many possible answers. Two major groups of failures:

   (a) Disk is significantly damaged

   (b) Disk is not damaged

   - Computer failure (hardware, software, network)
   - Transaction failure (division by zero, constraint violation)
   - Local transaction errors (no data found)
   - Concurrency control enforced (serialisability violation, deadlock)
   - Disk failure (errors with disk read/write)
   - Physical problems (power cut, fire)

4. What is do we mean by a catastrophic failure? How can we recover from catastrophic failures?

   Solution: This can be disk failure, but also flood, storm, earhquake, fire damage and terrorist attacks. We must restore the entire database

from the latest backup. We can use a back up of the system log to avoid loosing all transactions performed since the last backup.

5. How is the system log stored? What is it used for?

   Solution: The system log is a sequential append-only file. It is used to keep track of executed operations of transactions.

6. In the system log, we have several different common entry types. Which?

   Solution:

   (a) Redo entry

   (b) Undo entry

   (c) Reado/Undo entry

   (d) Read entry

7. What do we mean with the cache when speaking about recovery?

8. Why do we say that the cache complicates things?

9. In the cache, what is:

   (a) Dirty bit

   (b) Pin bit (hot bit)

10. What flushing strategies have we seen for the cache?

    Solution:

    (a) Inplace flushing

    (b) Shadowing

    (c) Steal

    (d) No-steal

    (e) Force

    (f) Steal/no-force

11. What is and why do we need write-ahead logging (give an example)?

    Solution: Write-ahead logging means that a transaction cannot commit until all log entries for that transaction have been flushed to disk. This is necessary, for example, to be able to redo transactions that were committed before a failure when we use a deferred-update recovery protocol.

12. What do we mean by checkpoints for recovery? Why is this important?

    Solution: The checkpoint is a recovery subsystem operation. It flushes relevant buffers and keeps a list of currently active transactions. It is important because it makes it possible, in case of a fail, to safely return to a consistent state.

13. What is the difference between a basic (rigid) checkpoint and fuzzy checkpoints?

    Solution: The basic checkpoint suspend execution of all transactions and flush all main non-pinned dirty memory buffers. Fuzzzy checkpoints allows us to continue transactions while flashing. begin checkpoint, flush all while allowing transactions to continue and then end checkpiont with the list of active transactions.

14. Describe the idea of the shadow recovery protocol. Discuss advantages/disadvantages with this protocol.

    Solution:

    - Flush each modified buffer immediately to a new place on disk, the old (shadow) version is kept on disk
    - If a transaction commits, the current version is used
    - If a transaction aborts, the shadow version is used

    One advantage of this system is that it can be used without having a log (may be needed for concurrency control). Disadvantages include lot of overhead, complicated garbage collection.

15. Describe the deferred-update recovery protocol. Discuss advantages/disadvantages with this protocol.

    Solution: The idea is to postpone any actual updates to the database until the transaction reaches the commit point. It can generally not be used in practice because we may run out of buffer space. Only possible if transactions are short with few changes.

16. Describe the immediate-update recovery protocol. Discuss advantages/disadvantages with this protocol.

    Solution: The idea is that we allow updates to be applied to the database before the transaction commits (not enforced). Advantages include not needing so much buffer space as deferred-update. Disadvantages include that we need to store both redo-type and undo-type entries in the log. More frequent I/O operations. It is more complex.

17. What is the ARIES recovery protocol? Discuss advantages/disadvantages with this protocol.

Solution: Similar to the immediate-update protocol, but the checkpoint remembers the table of currently dirty buffers. Advantages include that transactions can keep executing during checkpoint.

2 Recovery protocols

1. The following is a sequence of an immediate-update log written by two transactions T and U:

$[Checkpoint]$
$[START, U]$
$[write_item, U, A, 10, 11]$
$[START, T]$
$[write_item, T, B, 20, 21]$
$[write_item, U, C, 30, 31]$
$[write_item, T, D, 40, 41]$
$[commit, T]$
$[write_item, U, E, 50, 51]$
$[commit, U]$

   Describe the action of the recovery manager if there is a crash and the last log record to appear on the disk is:

   (a) $[START, T]$

   (b) $[commit, T]$

   (c) $[commit, U]$

   Solution:

   (a) If the last log record is $[START, T]$, the recovery manager must undo the write action of transaction U. That is write 10 to A.

   (b) If the last log record is $[commit, T]$, the actions of T must be redone. That is write 21 to T and 41 to D. The actions (until the crash) of U must be undone, so write 30 to C and 10 to A.

   (c) If the last log is $[commit, U]$ all actions of both transactions must be redone.

2. For each of the following recovery protocols, discuss how the log must look like and discuss what the system would have to do to recover after the system crash in the schedule in Table 1 using:

   (a) A deferred-update recovery protocol

   (b) An immediate-update UNDO/REDO recovery protocol

   Solution:

Table 1: Schedule

T_1	T_2	T_3	T_4
s_1			
$r_1(A)$			
$w_1(A)$			
	s_2		
	$r_2(B)$		
	$r_2(C)$		
	$w_2(B)$		
c_1			
Checkpoint t_1			
		s_3	
		$r_3(D)$	
			s_4
			$r_4(E)$
			$w_4(F)$
	c_2		
		c_3	
System crash t_2			

- **Deferred-update:**

$[start_transaction, T_1]$
$[write_item, T_1, A, new_value]$
$[start_transaction, T_2]$
$[write_item, T_2, B, new_value]$
$[checkpoint]$
$[start_transaction, T_3]$
$[start_transaction, T_4]$
$[write_item, T_4, F, new_value]$
$[commit, T_2]$
$[commit, T_3]$

  Before t_2, T_2 and T_3 had committed and T_4 had not. There is no need to deal with the operations of T_1 because it had committed before the checkpoint. The write operations of T_2 and T_3 must be redone. This is possible because the log is always force written before committing. T_4 is ignored. Since nothing is written to the disk before committing, this will effectively cancel (roll back) T_4.

- **Immediate-update:**

$[start_transaction, T_1]$
$[write_item, T_1, A, old_value, new_value]$
$[start_transaction, T_2]$
$[write_item, T_2, B, old_value, new_value]$
$[checkpoint]$
$[start_transaction, T_3]$
$[start_transaction, T_4]$
$[write_item, T_4, F, old_value, new_value]$
$[commit, T_2]$
$[commit, T_3]$

We don't have to care about T_1 since it is before the checkpoint. The committed transaction are T_2 and T_3. These have to be redone with the log entries on the disk. Transaction T_4 is not committed, so we have to undo the known operations of this transaction. There may be some entries not on the disk, but these changes have not been flushed.

3. Assume a single-user environment and a shadowing recovery protocol. Describe how the log should be like and how this system can recover from a system crash.

   Solution: No log required. We have both the old versions and the new versions on the disk. We have two directories, the new one pointing to the new, modified block, and a shadow directory with pointers to the unmodified blocks. To recover after the system failure, we free the modified database pages and discard the new directory and use the shadow directory as the new one.

4. Let us consider the ARIES recovery protocol. When do we not have to do a redo?

   Solution:

   - If the data page of the logpost is not in the Dirty Page Table
   - If a change recorded in the log (with LSN = N, for example) pertains to page P and the Dirty Page Table contains an entry for P with LSN greater than N, then that change must already be presient.