More about transaction management

Interaction between crash recovery and concurrency control

- Crash recovery: recover from system errors by means of logging
- Concurrency control: prevent non-serializable schedules
- Combination?

More about transaction management

Dirty reads

T_1	T_2	A	B
$l_1(A); r_1(A);$		25	25
$A := A + 100;$			
$w_1(A); l_1(B); u_1(A);$		125	
	$l_2(A); r_2(A);$		
	$A := A * 2;$		
	$w_2(A);$	250	
	Commit		
$r_1(B);$			
Crash			

- Recovery problem: T_2 has committed, and can hence not be rolled back. T_1, on the other hand, requires a rollback. But T_2 depends on T_1!

More about transaction management

Dirty reads

T_1	T_2	A	B
$l_1(A); r_1(A);$		25	25
$A := A + 100;$			
$w_1(A); l_1(B); u_1(A);$		125	
	$l_2(A); r_2(A);$		
	$A := A * 2;$		
	$w_2(A);$	250	
	$l_2(B)$**Denied**		
$r_1(B);$			
Abort$; u_1(B);$			
	$l_2(B); u_2(A); r_2(B);$		
	$B := B * 2;$		
	$w_2(B); u_2(B);$		50

- Implies cascading rollbacks in lock-based schedulers.

More about transaction management

Recoverable schedules

A schedule is called recoverable when every transaction in the schedule commits only when every other transaction from which it has read data, have already committed.

Example

- Recoverable and serial: $S_1 = w_1(A); w_1(B); w_2(A); r_2(B); c_1; c_2;$
- Recoverable, but not serializable: $S_2 = w_2(A); w_1(B); w_1(A); r_2(B); c_1; c_2;$
- Not recoverable, but serializable: $S_3 = w_1(A); w_1(B); w_2(A); r_2(B); c_2; c_1;$

Notice that

Recoverable schedules like S_1 may still require a cascading rollback!

More about transaction management

Avoid Cascading Rollback (ACR) schedules

A schedule is said to avoid cascading rollbacks if transactions in the schedule only read data from transaction that have already committed. In other words: the transaction can never ready "dirty data".

Example

- ACR and serial: $S_4 = w_1(A); w_1(B); w_2(A); c_1; r_2(B); c_2;$

Observe:

Every ACR schedule is recoverable.

More about transaction management

Strict schedules

A lock-based schedule is strict when every transaction only releases its exclusive locks when it has committed or aborted, and the commit or abort log record has been written to disk.

Observe:

- Every strict schedule is ACR.
- Every strict schedule is serializable.

Simplification

We need not wait until the commit or abort log record has been written to disk, provided that we are guaranteed that log records are written in the same order as they are created (group commit).