WAL recovery

Options for the log:

Can use a physical log. Whenever the bits change, write a log record.

Insert could cause logging of 4K worth of data

Can use a logical log – record the SQL command (nobody does this – too slow -- and won't work for undo)

Can use something in between – e.g. insert (record) on page-X. We will assume for now

(insert, page#, slot#, bytes)
(delete, P#, slot#, bytes)
update, p#, slot#, old bytes, new bytes)

Can log B-tree updates
Physical: means 8K bytes for a page splitter
Logical: do nothing – side effect of SQL
In between: insert (key, block#)

We will assume for now – no logging of B-trees

One simple scheme.

Periodically take a checkpoint.
    Force all dirty blocks to disk
    Write a log record containing a list of active (uncommitted xacts)

Do not log any information on B-trees
Logical (in between) data log -- per above

On a crash.
Start at the end of the log.

Look for commit and abort records; keep a list of finished xacts. Any log record that corresponds to an uncommitted or aborted xact, perform undo, by logically undoing the operation, but only if the after image matches the bytes on the page. modifying any affected B-trees, by searching the B-tree for the correct index key and fixing it, if necessary.

When you reach a checkpoint, compare checkpoint list of active xacts, with commit/abort of finishers list. If checkpoint list in commit/abort list, then "far enough". Otherwise, keep going until you find such a checkpoint.

Turn around and go forward, redoing the effects of all committed xacts.

If you crash during recovery, do it all again.

Example done in class

Problems: have to force buffer pool at a checkpoint -- expensive
All operations logical. Recovery may be slow -- have to do B-tree inserts and deletes
Won't work for escrow xacts – do example

Aries: more sophisticated, faster and way more complex.

We will simplify it somewhat – It is very complex.

Aries in a nutshell:

Cheap checkpoints
Physical redo
Logical undo
Redo-done first
Deals with escrow xacts (but we won't)

Now the details

Every log record has a LSN (sequence#)
Dirty page table – block#, current-LSN (dirtiers log record)
Xact table (1st-log record, last log record)

When you write a log record, you set the current in the xact table to it. You also store in Each log record the previous current one – i.e. one way linked list of log records.

On a crash:

Find the dirty page table and the xact table.

Start at the min (LSN in dirty page table)
Physical redo to the end of the log -- data plus B-trees – all xacts – whether committed or not – brings data base to the state at the time of the crash

Look at xact table. Find max LSN. Get that record and undo it. Replace current in xact table by it. Go backwards to the end – doing logical undo.

Crash during recovery – do it again.

CLRs are used for escrow xacts – cannot be undone multiple times.