# Lec 21 Database Recovery

Big picture
- Steal & no-force
- atomicity: all txns either abort or finish
- durability: committed changes safe

## # ARIES Protocol

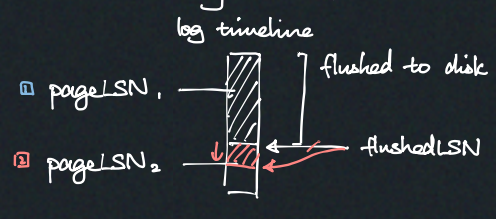
Algorithm for Recovery and Isolation Exploiting Semantics

Principle of Repeating History
└ Reconstruct DB state before crash by redoing changes
└ Rebuilds exact state before crash

Backward in history when undoing
└ Trace log in backward direction

## # Log Sequence Number (LSN)

- Variable length for recording changes
- Fixed-length, unique log sequence number monotonically increasing (use atomic counter)
- Pages also have page LSNs
- DRAM holds flushedLSNs
  └ tracked boundary btwn flushed and unflushed page LSN




1 can directly evict this page
2 when evict, need to first flush log

Full picture:

| | | |
|---|---|---|
| flushedLSN | mem | last logged LSN on disk |
| pageLSN | page x | newest update to page x |
| recLSN | DPT (Dirty Page Table) | oldest update to page x since last flush |
| lastLSN | ATT (Active Txn Table) | latest record of txn $T_i$ |
| MasterRecord | disk | LSN of last ckpt |

Assume for now:
- All log records fit within page
- Disk write atomic
- Single version, SS2PL
- Steal + no-force

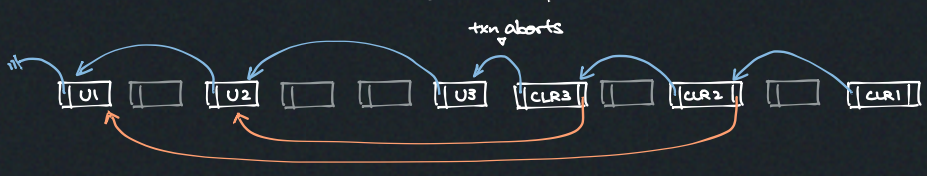
## # Committing

When commit:
1. Write log and ensure flushed
2. Write TXN-END to log to indicate no more logging for this txn
   └ This can be delayed and not flushed immediately, just for future clean up

## # Aborting

New log record field: prevLSN
tracks to prev LSN on curr txn
└ allows backward traversal

New log record: compensating log record (CLR) with additional field undoNextLSN




## # Checkpoint

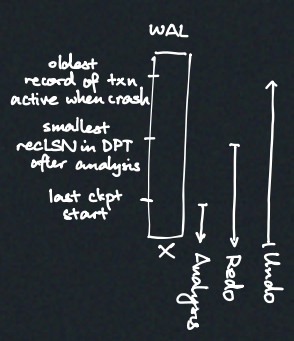
Track when at which point to start recovery from

- Naïve — pause starting new txns,
  wait for active txns to finish
  flush all pages

- Slightly better — pause write txns,
  save ATT and DPT at start of ckpt
  - but lots of copying and flushing

- Fuzzy — put in CHECKPOINT-BEGIN
  memcopy ATT and DPT
  let execution continue while preparing ckpt
  when done, put in CHECKPOINT-END + ATT + DPT
  (start recovery at CHECKPOINT-BEGIN)

## # Recovery




log disk LSNs | Disk pageLSNs | DRAM flushedLSN

Page1 pageLSN1 | Page2 pageLSN2

1. Start at CHECKPOINT-BEGIN
2. Run analysis for ATT and DPT at CHECKPOINT-END
3. Do analysis to identify txns' commit status
4. Redo: repeat actions
5. Undo: failed txns
   - Undo txns in reverse order, making sure CLRs are in linear time order

Weird cases:
- Crash during recovery: just recover again
- Crash during undo: same

Optimisation:
- Flush stuff to disk in background
- Lazy rollback
- Rewrite app for less long-running txns