COS 425:
Database and Information
Management Systems

Crash Recovery:
Recovery Phase

1

ARIES algorithm: review of preliminaries

- Transactions do concurrently (mixed):
  - Commit
  - Abort (those not part of restart after crash)
  - Checkpoint
  - Update
    - *Pin* data page in buffer and write change
    - Write log entry (LSN=#)
    - Update translation table (lastLSN = #)
    - Update dirty page table
    - Write pageLSN= # to page and *unpin* page

2

ARIES algorithm: review of preliminaries cont.

- Crash recovery manager does alone:
  - All actions during restore of database during restart after crash

3

Review: Writing to disk

- Write log pages from buffer:
  - on checkpoint
  - on commit of transaction
  - When want to write data page but pageLSN > flushedLSN
- Write data pages from buffer:
  - At discretion of buffer manager
- Writing fewer log pages and sequentially: cheaper

4

Crash recovery Phase I: Analysis

- Get log from disk
- Get most recently checkpointed transaction table and dirty page table
  - use *master record*
- Read log forward from checkpoint and update tables
  - For END log entries, remove transaction from transaction table
  - For other log entries, add or update transaction table entry

5

Crash recovery Phase II: Redo

- REDO all actions in log starting at earliest point when a change not on disk
  - Want earliest recLSN of all recLSNs in dirty pg table
  - Includes redo of UNDOs and ABORTs
    - See Phase III
- When redo action
  - Write new pageLSN
  - Do NOT write new Log entry

6

At end phase II Redo

- DB now in state was as recorded by *log on disk* at crash
- To finish phase II
  - write END log records for transactions in transaction table that were committed
  - Remove committed transactions from transaction table

7

Crash recovery Phase III: Undo

- UNDO actions of all transactions not committed by the end of phase II
- Work backwards through log
  - Follow pointer chain from each still-active transaction

    lastLSN →prevLSN →prevLSN → … → prevLSN
  - To process, interleave chains in LSN order from all active transactions
    - Event queue

8

Phase III UNDO Actions

- For UPDATE
  1. Write CLR record to log *NEW*
     - Records change done to undo UPDATE
     - Records ***undoNextLSN*** storing prevLSN of this UPDATE
       - Records next record to undo
     - Think of as ABORT log record like UPDATE log record
  2. Undo change in UPDATE
  3. If prevLSN for UPDATE == NULL, write END record for transaction

     Else queue prevLSN for processing

UNDO makes new DB changes =>

Need step 1 to deal with another crash as undoing

9

Phase III UNDO Actions

- For CLR

  If undoNextLSN == NULL, write END record for transaction
  - Undo/abort of transaction done

  Else queue undoNextLSN for processing
  - Re-establishes prevLSN chain for undoing/aborting transaction

- If are undoing a CLR, were in the process of undoing/aborting a transaction when crashed
- The redo of the CLR in phase II did the actual undoing
- Don't undo the UNDO represented by CLR record!

10

Effects of recovery

- REDO does “clean-up”
  - ends committed transactions
  - Writes ENDs to log
- UNDO does *new work* to undo/abort
  - Changes data pages, which may be on disk
  - Writes log entries for its actions

11

Abort as part of a transaction

- Write ABORT log record
  - Analogous to COMMIT but more to do before END
- Execute UNDO phase for

  lastLSN →prevLSN →prevLSN → … → prevLSN

  of the aborting transaction
- When UNDO phase writes END to log, is end of ABORT of transaction
  - Must remove from transaction table

12