CS186 Discussion 08

(Recovery)

Matthew Deng

Recovery

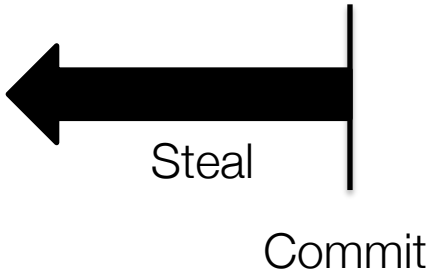
Acid

- Atomicity: All or none
- Consistency: Stay consistent
- Isolation: Isolated from other transactions
- Durability: Commit effects persist

Atomicity: No Steal/Steal

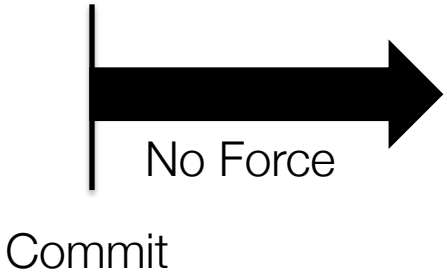
- No Steal
  - Do not allow disk to steal dirty pages from buffer

- Steal:
  - Allow dirty pages to be evicted from buffer before commit
  - Requires UNDO for atomicity



Durability: Force/No Force

- Force:
  - Force dirty pages to be written to disk before committing
- No Force:
  - Allow dirty pages to be in buffer when committing
  - Requires REDO for durability



Buffer Management

- No Steal + Force:
  - Slow
- Steal + No Force:
  - Fast
  - Requires REDO and UNDO

Write-Ahead Logging (WAL)

- Force log record before updated data written to disk
  - UNDO
  - atomicity

- Force all log records for transaction before commit
  - REDO
  - durability

Log Records

<LSN, prevLSN, XID, type, [pageID, length, offset, old, new]>

- LSN: log sequence number
- prevLSN: previous log sequence number
- XID: transaction ID
- Type: update, commit, abort, checkpoint, CLR, end

Transaction Table

- One entry per active transaction

<XID, status, lastLSN>

- XID: transaction ID
- Status: running, committing, aborting
- lastLSN: most recent LSN written by transaction

Dirty Page Table

- One entry per dirty page in buffer pool

<pageID, recLSN>

- pageID: page ID

- recLSN: LSN of log record which first dirtied the page

Committing

- Write commit record to log
- Flush all log records up to commit record to disk
- Write end record to log

Aborting

- Get lastLSN of transaction from transaction table
- Write abort record to log
- Write CLR for each transaction undone
  - Follow prevLSN
- CLR: compensation log record

Checkpoints

- begin_checkpoint: indicate beginning of checkpoint
- end_checkpoint: transaction and dirty page table
  - Accurate as of begin_checkpoint
- LSN of most recent checkpoint record

Recovery

- Start at checkpoint
- Analysis: find which transactions committed
  and which failed
- Redo: redo all transactions
- Undo: undo failed transactions

Phase 1: Analysis

Transaction Table

- Start from checkpoint table
- Update: add transaction
  - lastLSN = LSN
- Commit: change status
- Abort: change status
- END: remove transaction

Transactions active at last log flush before crash

Dirty Page Table

- Start from checkpoint DPT
- Update: add transaction if not in DPT
  - recLSN = LSN

Dirty pages that might not have made it to disk

Phase 2: Redo

```
for LSN >= smallest recLSN in DPT:
    if Page in DPT
    and recLSN <= LSN
    and pageLSN < LSN:
        Reapply log record or CLR
        pageLSN = LSN
```

Phase 3: Undo

```
ToUndo = {lastLSN of all Xact in Xact Table}
while ToUndo not empty:
    LSN = remove largest LSN from ToUndo
    if LSN is CLR and undoNextLSN != null:
        Add undoNextLSN to ToUndo
    if LSN is CLR and undoNextLSN == null:
        write END
    if LSN is update:
        UNDO
        write CLR
        add prevLSN to ToUndo
```