Recovery Concepts

Chapter 18 (lightly)

Need for Recovery

- Non-catastrophic: need Log only
  - Transaction abort
  - Normal part of many concurrency schemes
- Catastrophic: need Log + Backup
  - Physical loss of disks
  - Pervasive application error
  - System software error (corrupted filesystem, buffer management errors, etc.)
  - Virus, sabotage, etc.

(Review) The Log File Contains:

- Transaction starts/stops
- DB writes: "before" and "after" images
  - *before*s can be used to rollback an aborted transaction
  - *after*s can be used to redo a transaction without reexecuting it
- COMMITs and ABORTs

The log itself is as critical as the DB!

Reliable backups are critical, too!

Strategies Which Anticipate Normal Recovery

- *Deferred update*
  - Writes are not actually applied to DB until after T commits.
  - No UNDO is needed.
  - Implementation: buffers, shadow page table, etc.
- *Immediate update*
  - Writes are actually applied as T executes
  - Aborted transactions: UNDO (rollback)

Catastrophe

- First restore from a full backup
- Rollforward from log
  - REDO all <u>committed</u> transactions
    - Apply all logged WRITEs
  - Could actually REDO changes in reverse chrono order: i.e., only apply latest change
  - T's interrupted by the catastrophe must be restarted or user notified

Disaster Recovery via Redundancy

- A reliable duplicate copy could be used for "instant" recovery
  - copy could be "hot" (in use by applications) or only on standby
- SW-based
  - managed by DBMS or OS
  - could be part of a distributed system
- HW-based
  - RAID: Redundant Array of Inexpensive Disks