Chapter 15

Disaster Recovery

Recovery Management

- Device characteristics and failure types
- Recovery tools
- Recovery processes

Storage Device Basics

- Volatile: loses state after a shutdown (RAM)
- Nonvolatile: retains state after a shutdown (Disk, tape, optical media)
- Nonvolatile is more reliable than volatile but failures can cause loss of data
- Use multiple levels and redundant levels of nonvolatile storage for valuable data

Failure Types

- Local
  - Error occurs in an individual application
  - Affects only a single transaction
- Operating System Crash
  - Affects all active transactions
  - Less common than local failures
- Device Failure
  - Affects all active and past transactions
  - Least common

Recovery Tools

- Database Backups
- Transaction Log

Database Backups

- Full
  - A complete backup of entire database
  - Database can be recovered in its entirety from a full backup
- Incremental
  - A backup of changes to the database since the most recent full or incremental backup
  - Database cannot be recovered from an incremental backup alone

Recovery using Incremental Backups

1. Restore most recent full backup
2. Restore incremental backups in sequence

What is the data loss exposure?

8:00 am	Full Backup
9:00 am	Incremental Backup 1
10:00 am	Incremental Backup 2
11:00 am	Full Backup
12:00 pm	Incremental Backup 1
1:00 pm	Incremental Backup 2
1:30 pm	SYSTEM CRASH

Transaction Log

- History of database changes
- Every insert, delete, or update is recorded in two places:
  - Database table
  - Transaction log record
- Log used in failure recovery to restore database to a consistent state at a given point in time

Transaction Log Example

- START TRANSACTION;
- UPDATE Acct Set AcctBal = 200 WHERE AcctID = 50;
- UPDATE Acct SET AcctBal = 400 WHERE AcctID = 30;
- INSERT INTO Hist Values(1002, 500, 32);
- COMMIT;

LSN	TransNo	Action	Time	Table	Row	Column	Old	New
1	101001	START	10:29					
2	101001	UPDATE	10:30	Acct	10001	AcctBal	100	200
3	101001	UPDATE	10:30	Acct	15147	AcctBal	500	400
4	101001	INSERT	10:32	Hist	25045	*		<1002, 500, …>
5	101001	COMMIT	10:33					

Media Failure Recovery Procedure

- Steps:
  1. Restore database from the most recent full and incremental backups
  2. Use transaction log to redo all committed transactions since the most recent backup
- What does this procedure assume?
- How can we increase likelihood that backups and logs are available?

Other Recovery Scenarios

- Log is used to
  - Perform individual transaction rollback
  - Recover from OS or DBMS process crash
- Recovery from individual transaction crash / rollback
  - Search log backwards
  - Undo each transaction update using "old" values stored in log

Recovery from OS/DBMS Crash

- We could use the logs alone (how?)
- Want to avoid lengthy recovery time

Recovery Problem

- General Recovery Problem:
  - Start with database in inconsistent state
  - Use log to recover database to consistent state (no partly completed transactions) at a given point in time
- Complication:
  - Order of disk writes

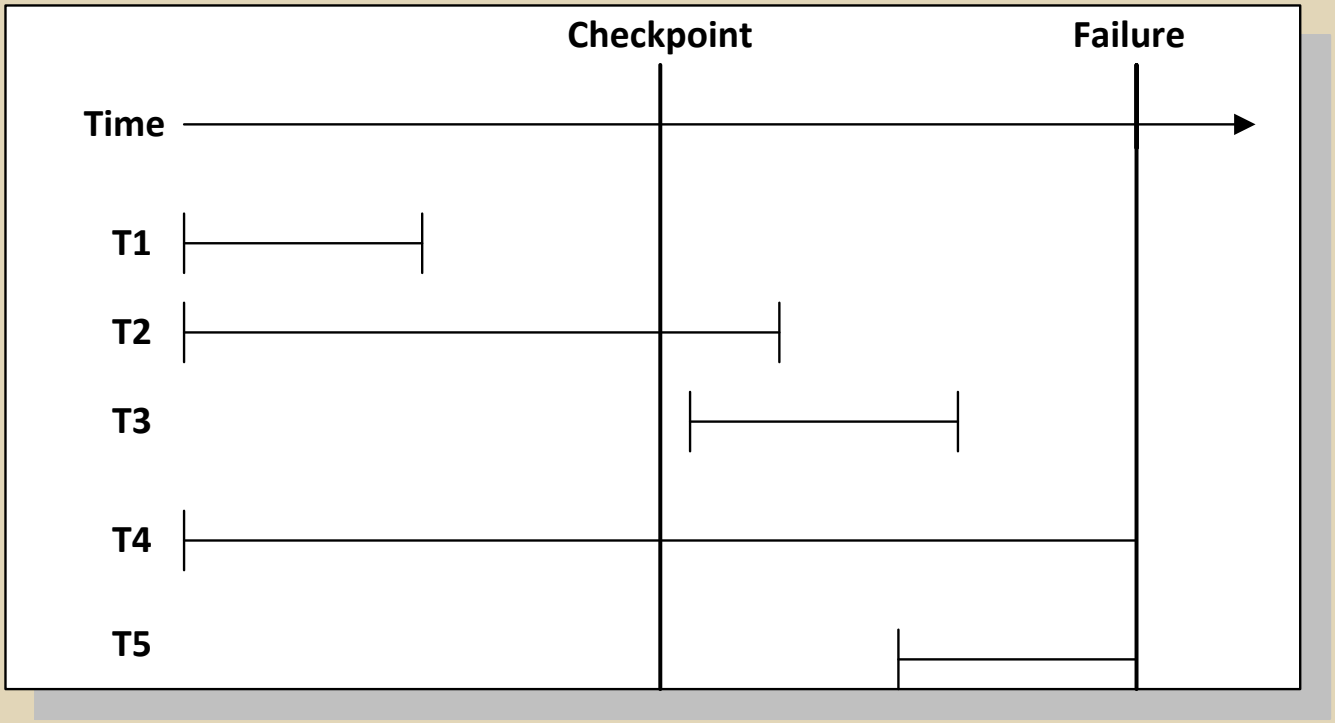
Disk Write Ordering

- Consider a series of UPDATE/INSERT/DELETE statements issued by various transactions
  - DBMS processes statements and requests disk writes
  - OS controls timing / ordering of disk writes to optimize system performance

Checkpoints

- Checkpoints are events that occur at regular time intervals
- At checkpoint:
  - 1. DBMS flushes ("force writes") to disk:
    - All log records
    - All "dirty" database pages (database updates)
  - 2. DBMS adds checkpoint record to log
- DB transaction activity pauses during checkpoint

Recovery Timeline



Recovery Processing

Class	Description	Restart Work
T1	Finished before CP	None
T2	Started before CP; finished before failure	Redo forward from the most recent checkpoint
T3	Started after CP; finished before failure	Redo forward from the most recent checkpoint
T4	Started before CP; not yet finished	Undo backwards from most recent log record
T5	Started after CP; not yet finished	Undo backwards from most recent log record

- Summary:
  - Undo uncommitted transactions
  - Redo committed transactions that were active at the most recent Checkpoint

LSN	TransNo	Action	Time	Table	Row	Column	Old	New
1	1	START	10:29					
2	1	UPDATE	10:30	Acct	10	Bal	100	200
3		CKPT(1)	10:31					
4	1	UPDATE	10:32	Acct	25	Bal	500	400
5	2	START	10:33					
6	2	UPDATE	10:34	Acct	11	Bal	105	205
7	1	INSERT	10:35	Hist	101	*		<1, 400,…>
8	2	UPDATE	10:36	Acct	26	Bal	100	200
9	2	INSERT	10:37	Hist	102	*		<2, 200,…>
10	3	START	10:38					
11	3	UPDATE	10:39	Acct	10	Bal	100	200
12		CKPT(1,2,3)	10:40					
13	3	UPDATE	10:41	Acct	25	Bal	500	400
14	1	COMMIT	10:42					
15	2	UPDATE	10:43	Acct	29	Bal	200	300
16	2	COMMIT	10:44					
17	3	INSERT	10:45	Hist	103	*		<3, 400,…>
18	4	START	10:46					
19	4	UPDATE	10:47	Acct	10	Bal	100	200
20	4	INSERT	10:48	Hist	104	*		<3, 200,…>

Recovery Algorithm

1. Define two lists, scanning forwards from most recent checkpoint:
   - Uncommitted/aborted transactions
   - Committed transactions
2. Process log in reverse order:
   - Undo uncommitted/aborted transactions
3. Process log forwards, redoing committed transactions
   - To find beginning of committed transactions, scan backwards from most recent checkpoint

Disaster Recovery Plan

- Begins with planning filesystem locations of database data files, log files, backup files
- Includes backup schedule, backup media handling, offsite backup storage
- Includes recovery procedures

Summary

- Database failures will occur
- Backup and logging mechanisms are available to assist recovery
- A disaster recovery plan is an important tool to help reduce data loss and downtime