Chapter 22

Transaction Management (con't)

Granularity of Data Items

- Size of data items chosen as unit of protection by concurrency control protocol.
- Ranging from coarse to fine:
  - The entire database.
  - A file.
  - A page (or area or database spaced).
  - A record.
  - A field value of a record.

Granularity of Data Items

- **Tradeoff:**
  - **coarser, the lower the degree of concurrency;**
  - **finer, more locking information that is needed to be stored.**
- **Best item size depends on the types of transactions.**

Hierarchy of Granularity

- Could represent granularity of locks in a hierarchical structure.
- Root node represents entire database, level 1s represent files, etc.
- When node is locked, all its descendants are also locked.
- DBMS should check hierarchical path before granting lock.

Hierarchy of Granularity

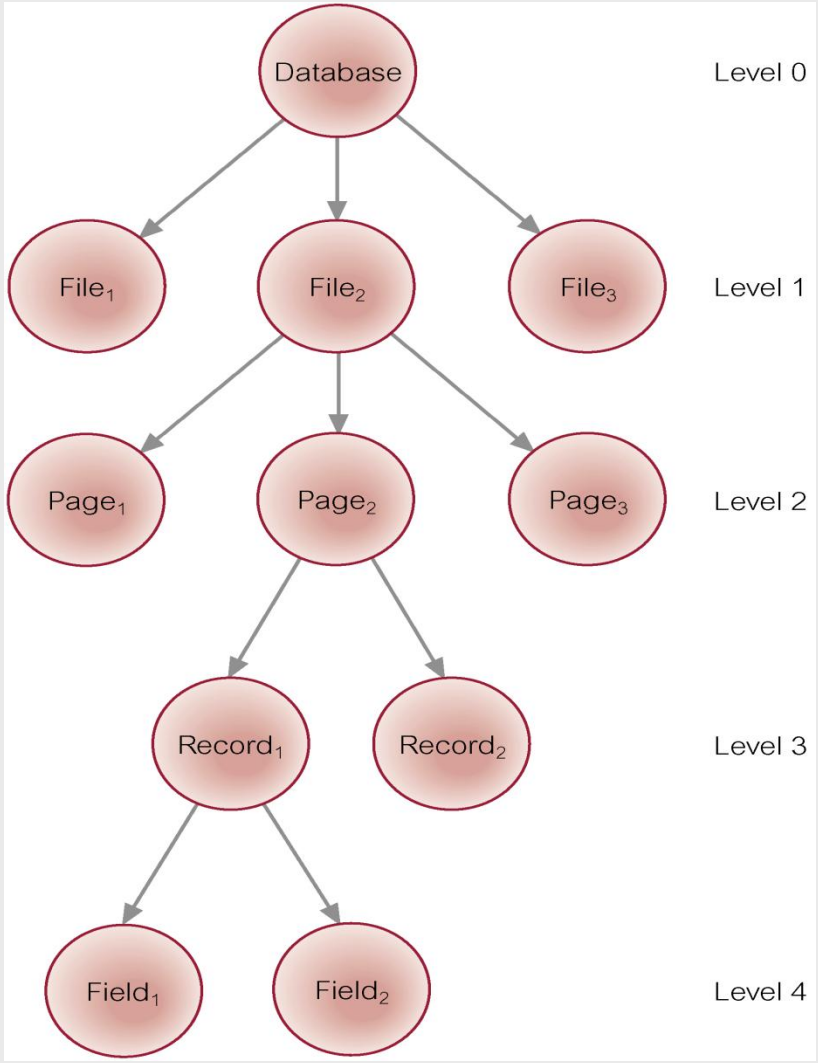
- ***Intention lock*** **could be used to lock all ancestors of a locked node.**
- **Intention locks can be read or write. Applied top-down, released bottom-up.**

Table 20.1 Lock compatibility table for multiple-granularity locking.

	IS	IX	S	SIX	X
IS	✓	✓	✓	✓	✗
IX	✓	✓	✗	✗	✗
S	✓	✗	✓	✗	✗
SIX	✓	✗	✗	✗	✗
X	✗	✗	✗	✗	✗

✓ = compatible, ✗ = incompatible

Levels of Locking



Database Recovery

Process of restoring database to a correct state in the event of a failure.

- **Need for Recovery Control**
  - **Two types of storage: volatile (main memory) and nonvolatile.**
  - **Volatile storage does not survive system crashes.**
  - **Stable storage represents information that has been replicated in several nonvolatile storage media with independent failure modes.**

Types of Failures

- System crashes, resulting in loss of main memory.
- Media failures, resulting in loss of parts of secondary storage.
- Application software errors.
- Natural physical disasters.
- Carelessness or unintentional destruction of data or facilities.
- Sabotage.

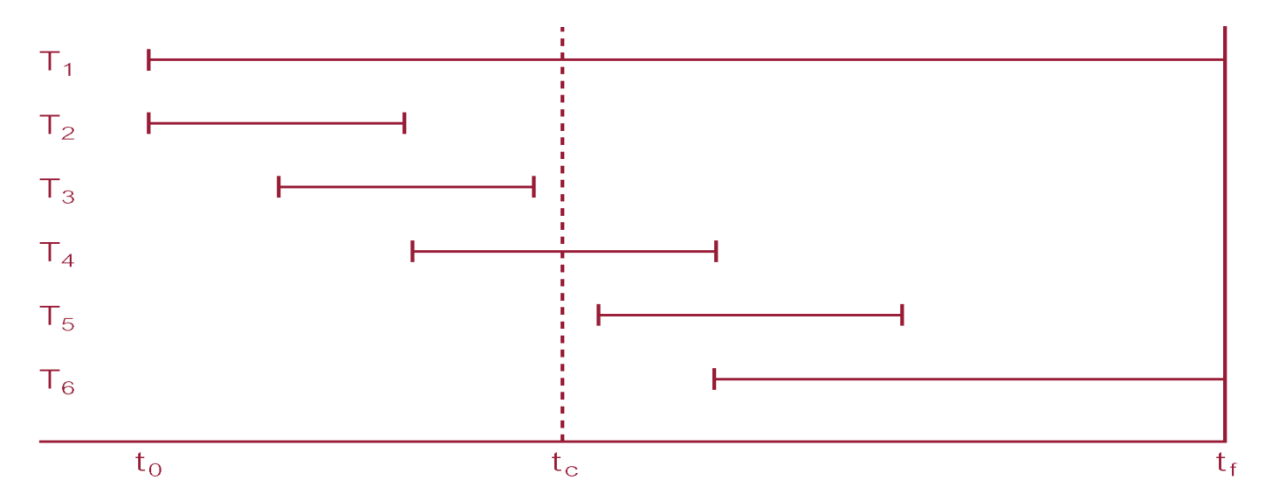
Transactions and Recovery

- **Transactions represent basic unit of recovery.**
- **Recovery manager responsible for atomicity and durability.**
- **If failure occurs between commit and database buffers being flushed to secondary storage then, to ensure durability, recovery manager has to *redo* (*rollforward*) transaction's updates.**

Transactions and Recovery

- **If transaction had not committed at failure time, recovery manager has to *undo* (*rollback*) any effects of that transaction for atomicity.**
- **Partial undo - only one transaction has to be undone.**
- **Global undo - all transactions have to be undone.**

Example



- DBMS starts at time t_0, but fails at time t_f. Assume data for transactions T_2 and T_3 have been written to secondary storage.
- T_1 and T_6 have to be undone. In absence of any other information, recovery manager has to redo T_2, T_3, T_4, and T_5.

Recovery Facilities

- **DBMS should provide following facilities to assist with recovery:**
  - **Backup mechanism, which makes periodic backup copies of database.**
  - **Logging facilities, which keep track of current state of transactions and database changes.**
  - **Checkpoint facility, which enables updates to database in progress to be made permanent.**
  - **Recovery manager, which allows DBMS to restore database to consistent state following a failure.**

Log File

- Contains information about all updates to database:
  - Transaction records.
  - Checkpoint records.
- Often used for other purposes (for example, auditing).

Log File

- **Transaction records contain:**
  - **Transaction identifier.**
  - **Type of log record, (transaction start, insert, update, delete, abort, commit).**
  - **Identifier of data item affected by database action (insert, delete, and update operations).**
  - **Before-image of data item.**
  - **After-image of data item.**
  - **Log management information.**

Sample Log File (This log file not related to Slide 11)

Tid	Time	Operation	Object	Before image	After image	pPtr	nPtr
T1	10:12	START				0	2
T1	10:13	UPDATE	STAFF SL21	(old value)	(new value)	1	8
T2	10:14	START				0	4
T2	10:16	INSERT	STAFF SG37		(new value)	3	5
T2	10:17	DELETE	STAFF SA9	(old value)		4	6
T2	10:17	UPDATE	PROPERTY PG16	(old value)	(new value)	5	9
T3	10:18	START				0	11
T1	10:18	COMMIT				2	0
	10:19	CHECKPOINT	T2, T3				
T2	10:19	COMMIT				6	0
T3	10:20	INSERT	PROPERTY PG4		(new value)	7	12
T3	10:21	COMMIT				11	0

Log File

- Log file may be duplexed or triplexed.
- Log file sometimes split into two separate random-access files.
- Potential bottleneck; critical in determining overall performance.

Checkpointing

Checkpoint

Point of synchronization between database and log file. All buffers are force-written to secondary storage.

- **Checkpoint record is created containing identifiers of all active transactions.**
- **When failure occurs, redo all transactions that committed since the checkpoint and undo all transactions active at time of crash.**

Checkpointing

- In previous example, with checkpoint at time t_c, changes made by T_2 and T_3 have been written to secondary storage.
- Thus:
  - only redo T_4 and T_5,
  - undo transactions T_1 and T6.

Recovery Techniques

- If database has been damaged:
  - Need to restore last backup copy of database and reapply updates of committed transactions using log file.
- If database is only inconsistent:
  - Need to undo changes that caused inconsistency. May also need to redo some transactions to ensure updates reach secondary storage.
  - Do not need backup, but can restore database using before- and after-images in the log file.

Main Recovery Techniques

- **Three main recovery techniques:**
  - **Deferred Update**
  - **Immediate Update**
  - **Shadow Paging**

Deferred Update

- Updates are not written to the database until after a transaction has reached its commit point.
- If transaction fails before commit, it will not have modified database and so no undoing of changes required.
- May be necessary to redo updates of committed transactions as their effect may not have reached database.

Immediate Update

- **Updates are applied to database as they occur.**
- **Need to redo updates of committed transactions following a failure.**
- **May need to undo effects of transactions that had not committed at time of failure.**
- **Essential that log records are written before write to database. *Write-ahead log protocol*.**

Immediate Update

- **If no "*transaction commit*" record in log, then that transaction was active at failure and must be undone.**
- **Undo operations are performed *in reverse order in which they were written to log*.**

Shadow Paging

- Maintain two page tables during life of a transaction: *current* page and *shadow* page table.
- When transaction starts, two pages are the same.
- Shadow page table is never changed thereafter and is used to restore database in event of failure.
- During transaction, current page table records all updates to database.
- When transaction completes, current page table becomes shadow page table.