Granularity of Locks and Degrees of Consistency in a Shared Data Base

--- J.N. Gray, R.A. Lorie, G.B. Putzolu, I.L. Traiqer

Summary

The main problems addressed in this paper are

- Database management systems that allow for concurrent transactions on the entities within the database must ensure a level of consistency of the data.
- One way to ensure consistency is to use locking mechanisms to limit data access by transactions that can lead to inconsistencies.
- Different concurrencies that can exist in a database system and the locking protocols of database entities at different granularities.
- Degree of consistency: Degree 0,Degree 1, Degree 2, and Degree 3. The relation between degrees and locking protocols, concurrency, overhead, recovery etc.

The Main Ideas discussed are

- **Granularity of Locks**: The Lockable units (units that lock automatically) represents trade off between concurrency and overhead. The size of lock units depends on the complexity of transaction. So, we need to consider lockable units of different granularities co exist in same system.
- **Hierarchical Locks**: we organize the set of resources to be locked in a hierarchy and each level is called as node. When the Exclusive & Shared access request is granted to a particular node, it implicitly locks each of its descendants and forms a lock to entire sub tree. We use a new access mode called Intention mode to lock all ancestors of a node in intention mode and to lock node in shared or exclusive mode to lock a sub tree rooted at that node.

- **Access Modes & Compatibility**: Access modes determine the compatibility of concurrent requests of two transactions. Share Lock allows only reading but not modification, for requester and other transactions too. Exclusive Lock allows reading and modifying the resource only by requester. Intention

lock is not compatible with exclusive or shared locks but compatible with itself. Intention mode is refined into :Intention Share mode, Intention Exclusive mode. One more mode is Share & Intention exclusive mode, which implicitly locks all descendants of node in share mode and allows requestor to explicitly lock descendant nodes in Exclusive, Shared Intention Exclusive Mode, or Intention Exclusive mode. Null mode gives no access to resources.

- **Rules for Requesting Nodes**: Before requesting Shared or Intention Shared on node, all ancestor nodes must be held in Intention Exclusive or Intention Shared mode. Before requesting Exclusive, Shared Intention Exclusive, or Intention Exclusive on node, all ancestors must be held in Shared Intention Exclusive, or Intention Exclusive on mode. Locks should be released either at end of transaction (in any order), or in leaf to root order in the middle of a transaction.
- **Directed Acyclic Graphs**: To lock a node implicitly or explicitly, we should lock all the parents of node in Graph. Before requesting Shared or Intention Exclusive on a node, request at least one parent (and by induction a path to root) in Intention Shared or greater mode. Before requesting Intention Exclusive, Shared Intention Exclusive or Exclusive, request all parents of the node in Intention Exclusive or greater mode. Locks should be release either at end of transaction in any order, or in leaf to root order.

- **Dynamic lock graphs**: if hierarchy/Graphs is dynamically updated, we need rule for changing parents of a node. Before moving a node in the lock graph, the node must be implicitly or explicitly granted Exclusive mode in both its old and its new position in the graph. Further, the node must not be moved in a way to introduce cycles.
- **Degrees of Consistency:** Database consists of entities that are structured in ways. Structure is assertions on data. Database is consistent if it satisfies all assertions. Database needs to be temporarily inconsistent in order to transform to a new consistent state - need transactions to hide inconsistency. Transaction is committed when transaction abdicates the right to undo the write; its output is available to other transactions. Output is uncommitted or dirty if not yet committed. Crux of concurrency is preventing the reading or writing of other transactions' dirty data. Degree 3: Transaction does not overwrite dirty data of other transactions. Transaction does not commit any writes until it completes all its writes. Transaction does not read dirty data from other transactions. Other transactions do not dirty any data read by Transaction before Transaction completes. Degree 2: Transaction does not

overwrite dirty data of other transactions. Transaction doss not commit any writes before POT. Transaction does not read dirty data of other transactions. Degree 1: Transaction does not overwrite dirty data of other transactions. Transaction does not commit any writes before EOT. Degree 0: Transaction does not overwrite dirty data of other transactions. If each transaction observes the lock protocol definition of consistency, any legal schedule is assured of that degree of consistency. Unless a transaction sets the locks, it can be constructed in a schedule to run at lower degree of consistency. Each transaction can choose its degree of consistency if all transactions observe at least degree 0 protocols.

- **Dependencies among transactions**: If Transaction T performs **a** then Transaction T' performs **a'**. Then, T <<< T' if **a** is a write and **a'** is a write, or if **a** is a write and **a'** is a read, or if **a** is a read and **a'** is a write. T << T' if **a** is a write and **a'** is a write, or if **a** is a write and **a'** is a read. T < T' if **a** is a write and **a'** is a write.

- **Relationship to Transaction Backup and System Recovery:** system with degree 1 consistency allows transaction backup and system recovery without lost updates. If the system is of degree 2 then no transaction reads uncommitted data. So if the completed transactions are re-done in log order but in the absence of some undone (incomplete) transactions they will give exactly the sane results as were obtained in the presence of the undone transactions.

Question: Which degree of consistency to choose will depend on the nature of the transactions?

Submitted By

Lakshman Madhav Kollipara

SID- 932655504

Submitted To

Arash Termehchy

CS540- Database Management Systems