Qu’est-ce que le contrôle de la concurrence?

Conccurency Control is the process of managin simultaneous operations on the database without having them interfere with one another. Database recovery is the process of restoring the database to a correct state after a failure. Both protect the database from inconsistencies an data loss.

Qu’est-ce qu’une transaction?

A transaction is an action. or series of actions, carried out by a single user or application program that acesses or changes the contents of the database. A transaction is a logical unit of work that takes the database from one consistant state to another.

Transaction can terminate sucessfully (commit)

or unsucessfully (abort).

Aborted transactions must be undone or rolled back. Tre transaction is also the unit of conccurrency and the unit of recovery.

A transaction should posess the four basic or so called ACID properties :

atomicity, consistency, isolation and durability.

Atomicity and durability are the responsability of the recevory subsystem :

isolation and to some extent, consistency are the responsibility of the concurency control subsystem.

Concurrency control is needed when multiple users are allowed to acess the database simultaneously. Without it, problems of lost update, uncommitted dependecy, and inconsistent analysis can arise.

Serial execution means executing one transaction at a time. with no interleaving of operations. A schedules show the sequence of the operations of transactions.

A schedule is serializable if it produces the same results as some serial schedule.

Two methods that guarantee serializability are two-phase locking (2PL) and timestamping. Locks may be shared(read) or exclusive (write). In two-phase locking. a transaction acquires all its locks before releasing any. In timestamping. transactions are ordered in such a way that older transactions get priority in the event of conflicts.

Deadlocks

Deadlock occurs when two or more transactions are waiting to acess data the other transaction has locked. The onyl way to break deadlock one it has occured is to abort one or more of the transactions.

A tree may be used to represent the granuliarity of locks in a system that allows locking of data items of different sizes. When an item is locked, all its descendants are also locked. When a new transaction requests a lock, it is easy to check all the ancestors of the object to determine wheter they are already locked. To show whether any of the node’s descendants are locked, an intention lock is placed on all the ancestors of any node being locked.

Some cause of failure are system crashes, media failure, application software errors, carelessness, natural physical disasters and sabotage. These failures can result in the loss of main memory and or the disk copy of the dababase. Recov ery tech niques minimize these effects.

To facilitate recovery, one method is for the system to maintain a log file : containg transaction records that identify the start end of transactions and the before – and -after image of the write operations.

Using deferred updates, write are done initially to the log only and the log records are used to perform actual updates to the database if the system fails, it examines the log to dtermine which transactions it needs to redo. but there is no need to undo any writes. Using immediates updates, an update may be made to the database itself any time after a log record is written to disk. If a failure occurs, the checkpoint record identifies which transactions need to be redone.

Advanced transactions models :

include :

1 - (hierarchies de transactions , on la vu déjà)

2 – sagas : consiste en une série de transactions appliquées les unes après les autres qui crée des transactions compensatoires pour chaque transaction appliquée. On utilise les transactions compensatoires pour retourner à l’état initial en cas de problème.

Par exemple si j’ai une transaction qui ajoute 10 dollars sur un compte : la transaction compensatoire est d’enlever 10 dollars dans le compte. Bien sur là on parle d’une transaction simple, une saga comporte une série de transaction et de transaction compensatoires (T1, T2, T3 et C1, C2, C3 etc).

3 - transactions multi-niveaux: <http://www-poleia.lip6.fr/~doucet/CoursBDRA/BDRA2-transactions.pdf>

Si j’Ai bien compris c’Est un modèle qui divise des transactions en sous-transaction dépendamment des opérations faites. Les opérations cumulatives seront mises ensemble, par exemple s’il y a addition et soustraction sur la même donnée. Au final si les opérations se font dans n’importe quel sens cela va donner le même résultat. On peut pratiquement dire qu’il y a combinaison de transaction.

4 - dynamically restructuring transactions,

Transaction avec Structure dynamique.

ce que j’ai comprit : C’est un modèle qui divise les transactions en sous-transaction pour diviser les ressources utilisés (ex verrous) Le resultat de ce modèle doit donner des transactions qui sont indépendante les unes des autres. Cela permet de partager des résultats partielle tout en conservant la cohérence.

At the start of this section we discussed some of the characteristics of design applications; for example, uncertain duration from hour to months interactions with other concurrent activities and uncertain developments, so that some actions cannot be foreseen at the beginning. To address the constraints imposed by the ACID properties of flat transactions, two new operations were proposed: split transaction and join transaction the principle behind split-transactions is to split an active transaction in two serializable transaction and divide its actions and resource (for example, locked data items) between the new transactions. The resulting transactions can proceed independently from that point perhaps controlled by different users, and behave as though they had always been independent. This allows the partial results of a transaction to be shared with other transactions while preserving its semantics; that’s is if the original transaction conformed to the ACID properties then so will the new transactions.

The split transaction operation can be applied only when it is possible to generate two transactions that are serializable with each other and with all other concurrently executing transactions. The conditions that permit a transaction T to be split into transactions A and B are defined as follows:

A write Bwriteset Bwrite Last This conditions states that if Both A and B write to the same object B write operations must follow A write operations

AReadSet BwriteSet = O This conditions states that A cannot see any of the results from B

BReadSet A write Set = ShareSet this conditions tsates that B may see the results of A

These three conditions guarantee that A is serialized before B However, if A aborts, B must also abort because it has read data written by A. If both BWriteLast and ShareSet are empty then A and B can be serialized in any order and both can be commited independently.

The join-transaction performs the reverse operation of the split-transaction, merging the ongoing work of two or more independent transactions as though these transactions had always been a single transaction. A split transaction followed by a join-transaction on one of the newly created transactions can be used to transfer resources among particular transactions without having to make the resource available o other transactions.

The main advantage of dynamic restructuring methods are:

Adaptive recovery which allows part of the work done by a transaction to be committed so that it will not be affected by subsequent failures.

Reducing isolation which allows resource to be released by committing part of the transaction.

5 - and workflow model (modèle de flux de travaux, on a vu en cours)

On en a besoin lorsque on doit faire des execution coordonée de taches impliquant plusieurs ressources différentes (humain, informatique, plusieurs BD qui se parlent)

En gros sa implique de faire des traductions entre les systèmes