Assignment 2

1) Discuss Thomas Write Rule in detail.

Ans In optimistic concurrency control, a timestamp ordering is imposed on transactions, and validation checks that all conflicting actions occured in the same order.

Timestamp can also be used in another way: each transaction can be assigned a timestamp at startup and we can ensure, at execution time that if action ai of transaction Ti conflicts with action aj of transaction Tj, ai occurs before aj if $TS(Ti) < TS(Tj)$. If an action violates this ordering the transaction is aborted and restarted.

To implement this concurrency control scheme, every database object O is given a "read timestamp RTS(O)" and "write timestamp WTS(O).

Let us consider what happens when transaction T wants to write object O.

1. If $TS(T) < RTS(O)$, the write action conflicts with the most recent read action of O, and T is therefor

aborted and restarted.

2. If TS(T) < WTS(O), a naive approach would be to abort T because its write action conflicts with the most recent write of O and is out of timestamp order. It turns out that we can safely ignore such writes and continue ignoring outdated write is called the Thomas write Rule.

3. Otherwise, T write O and WTS(O) is set to TS(T)

* The Thomas write Rule.

- We now consider the justification for the thomas write Rule. If TS(T) < WTS(O), the current write action has, in effect, been made absolute by the most recent write of O, which follows the current write according to the timestamp ordering on transactions.

We can think of T's write action as if it had occured immediately before the most recent write of O and was never read by anyone.

If the Thomas Write Rule is not used, that is T, is aborted in case (2) above, the time stamp Protocol like 2PL allows only Conflict Serializable schedules are Permitted that are not Conflict Serializable.

T1	T2
R(A)	
	W(A)
	Commit
W(A)	
Commit	

A Serializable schedule that is not Conflict serializable.

T1's Read are precedes T1's write of the same object, this schedule is not Conflict Serializable. The Thomas Write Rule relies on the observation that T2's write is never seen by any transaction and the schedule is therefore equivalent to the serializable schedule obtained by deleting this write action, which is shown in figure.

T_1	T_2
R(A)	commit
W(A)	
commit	

A conflict serializable schedule.

2) Discuss three Phase of ARIES Recovery algorithm.

Ans

ARIES is a recovery algorithm that is designed to work with a steal, no force approach. When the recovery manager is invoked after a crash, restart proceeds in three Phases.

1) Analysis :-

Identifies dirty pages in the buffer Pool and active transactions at the time of the crash.

2) Redo :-

Reports all actions, starting from an appropriate point in the log, and restores the database state to what it was at the time of the crash.

3) Undo :-

Undoes the actions of transaction then did not commit so that database reflects only the actions of commited transactions.

3) Short note on CLR.

Ans A compensation log record (CLR) is written just before the change recorded in an update log record is undone.

A compensation log record (also contains a field called Undo Next LSN, which is the LSN of the next log record that is to be undone for the transaction that wrote update record U. This field in C is set to the value of prevLSN in U.

Unlike an update log record, a CLR describes an action that will never be ~~described an action that will never be~~ undone, that is we never undo an undo action. The reason is simple an update log record describes changes made by a transaction during normal execution and the transaction may subsequently be aborted an action taken to roll back a transaction for which the decision to abort has already been made

Thus, the transaction action must be rollback and the undo action describe by the CLR is definetly required.

This observation is very useful because it bounds the amount of space needed for the log during restart from a crash. The number of CLRs that can be written during undo is no more than the number of update log records for active transaction at the time of the crash.

It may well happen that CLR is written to stable storage but that the undo action that it describe is not yet written to disk when the system crashes again.

4) Define transaction with references to MySQL.

Ans A transaction is a set of one or more SQL statements that are logically grouped together and that must be either applied to the database in their entirely or not applied at all.

Consider the commonly cited example of a funds transfer from one account to another. In its most simple form, this transfer will involve two update statements one to reduce the account balance in the "from" account, and another to increase the account balance in "to" account. Suppose that "from" account has been updated but then the change to the "to" account cannot be completed. We must be sure to undo that first update; or the money that was to be transferred will have in effect "disappeared".

We expect database transactions to conform to the ACID principle which means the transaction should be,

Atomic :- The transaction is indivisible either all the statements in the

transaction are applied to the database or none are,

Consistent :- The database remains in a consistent state before and after transaction execution.

Isolated :- while multiple transaction can be executed by one or more users simultaneously, one transaction should not see the effects of other concurrent transactions.

Durable :- once a transaction is saved to the database its changes are expected to persist. Even if the user turns off her computer or the database server goes down the changes will be saved. This is usually means that the result of the transaction must be written to a non volatile form of storage such a harddisk. Stored program provide an excellent mechanism for defining encapsulating and managing transactions. Without these features available in Stored programs the calling provide the logic to control locking and handle transaction failure.

With MySQL stored program support we can now encapsulate the multiple interdependent SQL statement of the transaction into single stored program.

There are two most popular storage engines used with MySQL are MyISAM and InnoDB.

5) Discuss four properties of transaction to ensure integrity of data (ACID).

Ans ① Atomic:

- User should be able to regard the execution of each transaction as atomic, either all actions are carried out or non are user should not have to worry about the effect of incomplete transaction.

② Consistency:

- Each transaction, run by itself with no concurrent execution of other transactions must preserve the consistency of the database. This property is called consistency and the DBMS assumes that it holds for each transaction. Ensuring this property of a transaction is the responsibility of the user.

③ Isolation:

User should be able to understand a transaction without considering the effect of other concurrency executing transaction, even if the DBMS

interleaves the actions of several transaction for performance reasons. This property is sometimes referred to as isolation. Transaction are isolated or protected from the effects of concurrency scheduling other transactions.

④ Durability:

once the DBMS informs the user that a transaction has been successfully completed its effects should persist even if the system crashes before all its changes are reflected on disk. This property is called durability

⇒ The acronym ACID is sometimes used to refer to the four properties of transactions.

Page No.
Date :

6) Discuss the role of DBA in security with reference to Database.

Ans The database administrator (DBA) plays an important role in enforcing the security-related aspects of Database design.

In conjunction with the owners of the data, the DBA will probably also contribute to developing a security policy. The DBA has a special account, which we will call the system account and is responsible for the overall security of the system.

→ In particular the DBA deals with the following.

① Creating new accounts:

Each new user or group must be assigned an authorization id and a password. Note that application program that access the database have the same authorization id as the user execution the program.

② Mandatory control issues:

If the DBMS supports mandatory

control some customized systems for applications with very high security requirements provide such the DBA must assign security classes to each database object and assign security clearances to each authorization id accordance with the chosen security policy.

The DBA is also responsible for maintaining the audit trail, which is essentially the log of updates with the authorization id added to each log entry. This log is just a minor extension of the log mechanism used to recover from crashes.

Additionally, the DBA may choose to maintain a log of all actions including reads, performed by a user, Analyzing such histories of how the DBMS is accessed can help prevent security violations by identifying suspicious patterns before an intruder finally succeeds in breaking in or it can help track down an intruder after a violation has been detected.

7) Discuss two important assumptions with respect to transactions.

Ans

① Transactions interact with each other only via database read and write operations. For example, they are not allowed to exchange messages.

② A database is a field collection of independent objects. When objects are added to or deleted from a database or there are relationships between database objects what we want to exploit for performance, some additional issue arise.