Course code : **CSE2007**
Course title : **Database Management System**
Module : **5**
Topic : **2**

Properties of Transaction

Objectives

This session will give the knowledge about

- Transaction and System Concepts
- Properties of Transaction

Transaction and System Concepts

A transaction is an atomic unit of work that is either completed in its entirety or not done at all.

For recovery purposes, the system needs to keep track of when the transaction starts, terminates, and commits or aborts.

Transaction states:

- Active state
- Partially committed state
- Committed state
- Failed state
- Terminated State

Transaction and System Concepts

Recovery manager keeps track of the following operations:

- begin_transaction: This marks the beginning of transaction execution.
- read or write: These specify read or write operations on the database items that are executed as part of a transaction.
- end_transaction: This specifies that read and write transaction operations have ended and marks the end limit of transaction execution.

  At this point it may be necessary to check whether the changes introduced by the transaction can be permanently applied to the database or whether the transaction has to be aborted because it violates concurrency control or for some other reason.

Transaction and System Concepts

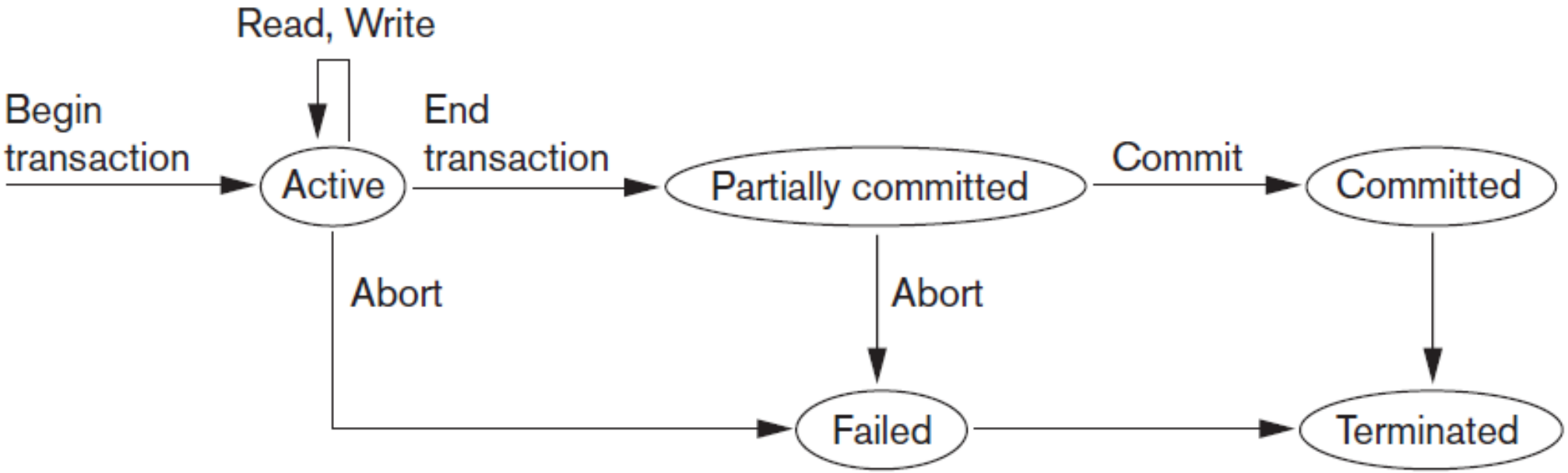
- commit_transaction: This signals a successful end of the transaction so that any changes (updates) executed by the transaction can be safely committed to the database and will not be undone.
- rollback (or abort): This signals that the transaction has ended unsuccessfully, so that any changes or effects that the transaction may have applied to the database must be undone.

Transaction and System Concepts

Recovery techniques use the following operators:

- undo: Similar to rollback except that it applies to a single operation rather than to a whole transaction.
- redo: This specifies that certain transaction operations must be redone to ensure that all the operations of a committed transaction have been applied successfully to the database.

State transition diagram for transaction execution



The System Log

Log or Journal: The log keeps track of all transaction operations that affect the values of database items.

- This information may be needed to permit recovery from transaction failures.
- The log is kept on disk, so it is not affected by any type of failure except for disk or catastrophic failure.
- In addition, the log is periodically backed up to archival storage (tape) to guard against such catastrophic failures.

The System Log

T in the following discussion refers to a unique transaction-id that is generated automatically by the system and is used to identify each transaction:

Types of log record:

- [start_transaction,T]: Records that transaction T has started execution.
- [write_item,T,X,old_value,new_value]: Records that transaction T has changed the value of database item X from old_value to new_value.

The System Log

- [read_item,T,X]: Records that transaction T has read the value of database item X.
- [commit,T]: Records that transaction T has completed successfully, and affirms that its effect can be committed (recorded permanently) to the database.
- [abort,T]: Records that transaction T has been aborted.

The System Log

Protocols for recovery that avoid cascading rollbacks do not require that read operations be written to the system log, whereas other protocols require these entries for recovery.

Strict protocols require simpler write entries that do not include new_value

Recovery using log records

If the system crashes, we can recover to a consistent database state by examining the log and using one of the techniques described in next session.

1. Because the log contains a record of every write operation that changes the value of some database item, it is possible to undo the effect of these write operations of a transaction T by tracing backward through the log and resetting all items changed by a write operation of T to their old_values.

2. We can also redo the effect of the write operations of a transaction T by tracing forward through the log and setting all items changed by a write operation of T (that did not get done permanently) to their new_values.

Commit Point of a Transaction

Definition a Commit Point:

- A transaction T reaches its commit point when all its operations that access the database have been executed successfully and the effect of all the transaction operations on the database has been recorded in the log.
- Beyond the commit point, the transaction is said to be committed, and its effect is assumed to be permanently recorded in the database.
- The transaction then writes an entry [commit,T] into the log.

Roll Back of transactions:

- Needed for transactions that have a [start_transaction,T] entry into the log but no commit entry [commit,T] into the log.

Commit Point of a Transaction

Redoing transactions:

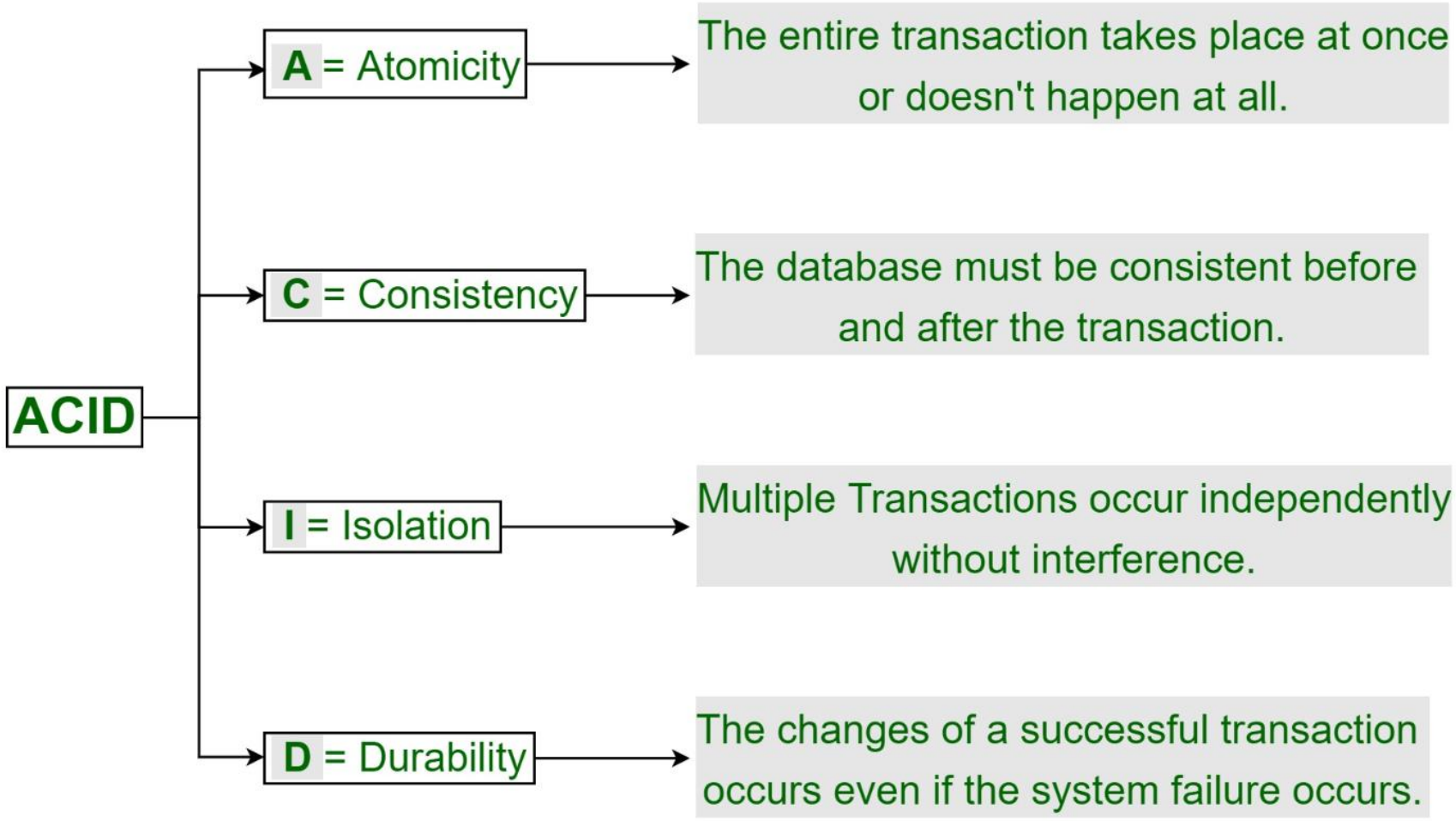
- Transactions that have written their commit entry in the log must also have recorded all their write operations in the log; otherwise they would not be committed, so their effect on the database can be redone from the log entries. (Notice that the log file must be kept on disk.
- At the time of a system crash, only the log entries that have been written back to disk are considered in the recovery process because the contents of main memory may be lost.)

Commit Point of a Transaction

Force writing a log:

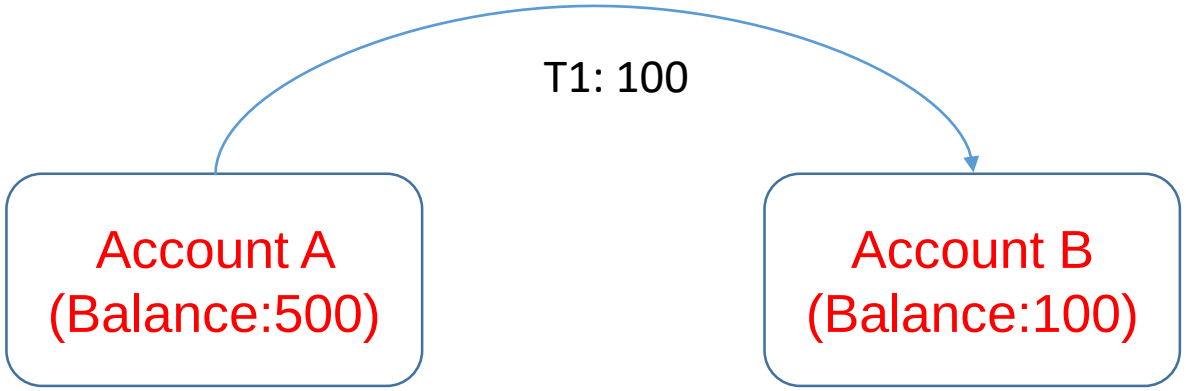
- Before a transaction reaches its commit point, any portion of the log that has not been written to the disk yet must now be written to the disk.
- This process is called force-writing the log file before committing a transaction.

Properties of Transactions



Properties of Transactions

1. Begin Transaction
2. read balance(a)
3. compute balance(a)-100
4. write balance(a)
5. read balance(b)
6. compute balance(b)+100
7. write balance(b)
8. commit



Concurrency Control Techniques

Locking :

Transaction uses locks to deny access to other transactions and so prevent incorrect updates. Two Types of Locks

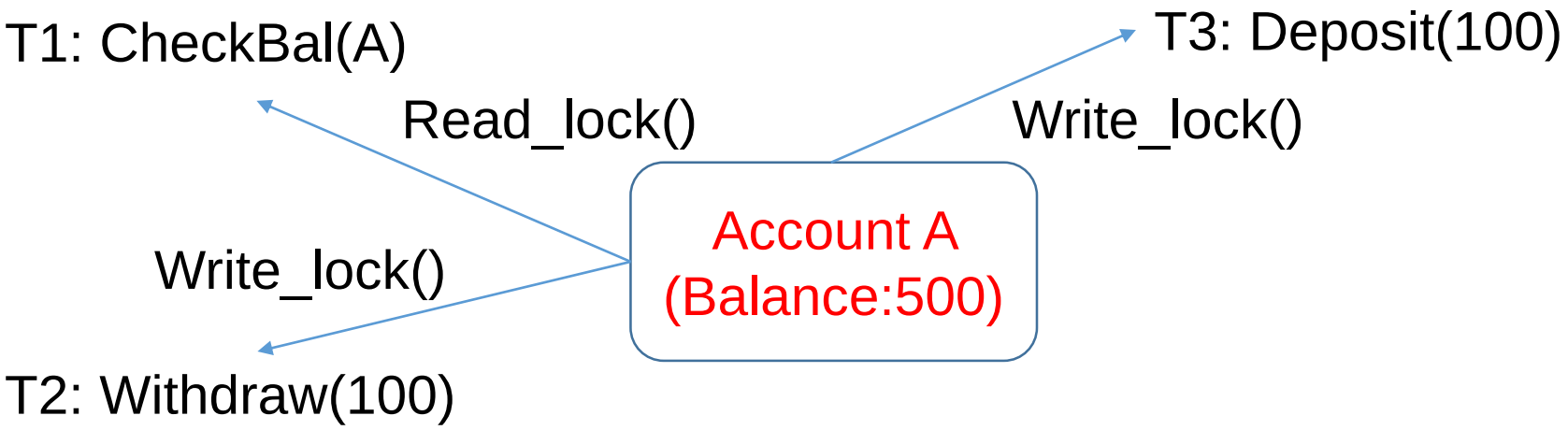
Read_Lock(item)

- Read locked item is also called share-locked because other transactions are allowed to read the item.

Write_Lock(item)

- If a transaction is to write an item X, it must have exclusive access to X.
- Write-locked item is called exclusive-locked because a single transaction exclusively holds the lock on the item.

Concurrency Control Techniques



If write_lock() is implemented unlock is mandatory to be used by other transactions

Summary

This session will give the knowledge about

- Transaction and System Concepts
- Properties of Transaction