# DATA AND WEB DEVELOPMENT

## CC6012NI

## WEEK - 03

## Database Recovery II

# Definition

Database Recovery is the process of RESTORING the database to the MOST RECENT CONSISTENT STATE that EXISTED just BEFORE the FAILURE

# Types of Failures Contd..

- Transaction Failure
- Media Failure
- System Failure

# Recovery – System Failure

Will affect ALL TRANSACTIONS currently in PROGRESS but NOT database itself.

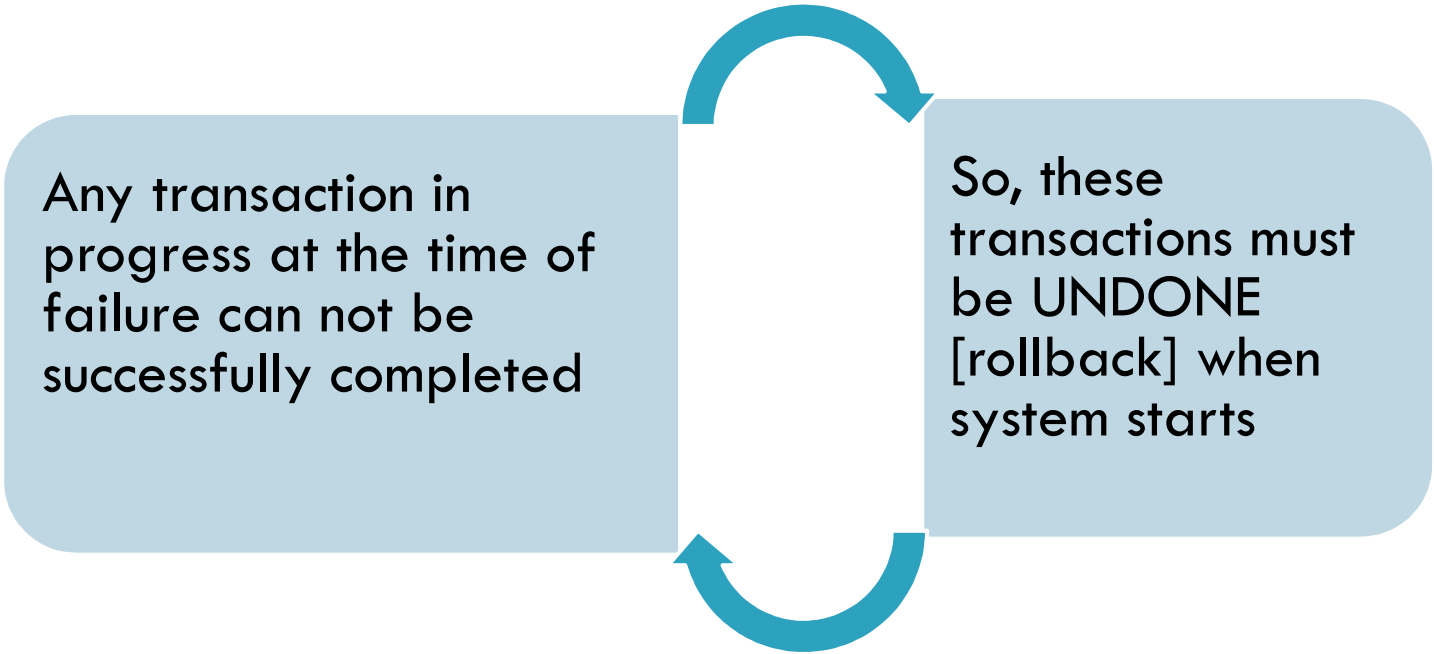
System recovery is carried out as part of system's restart procedure.

# Key Point about System Failure

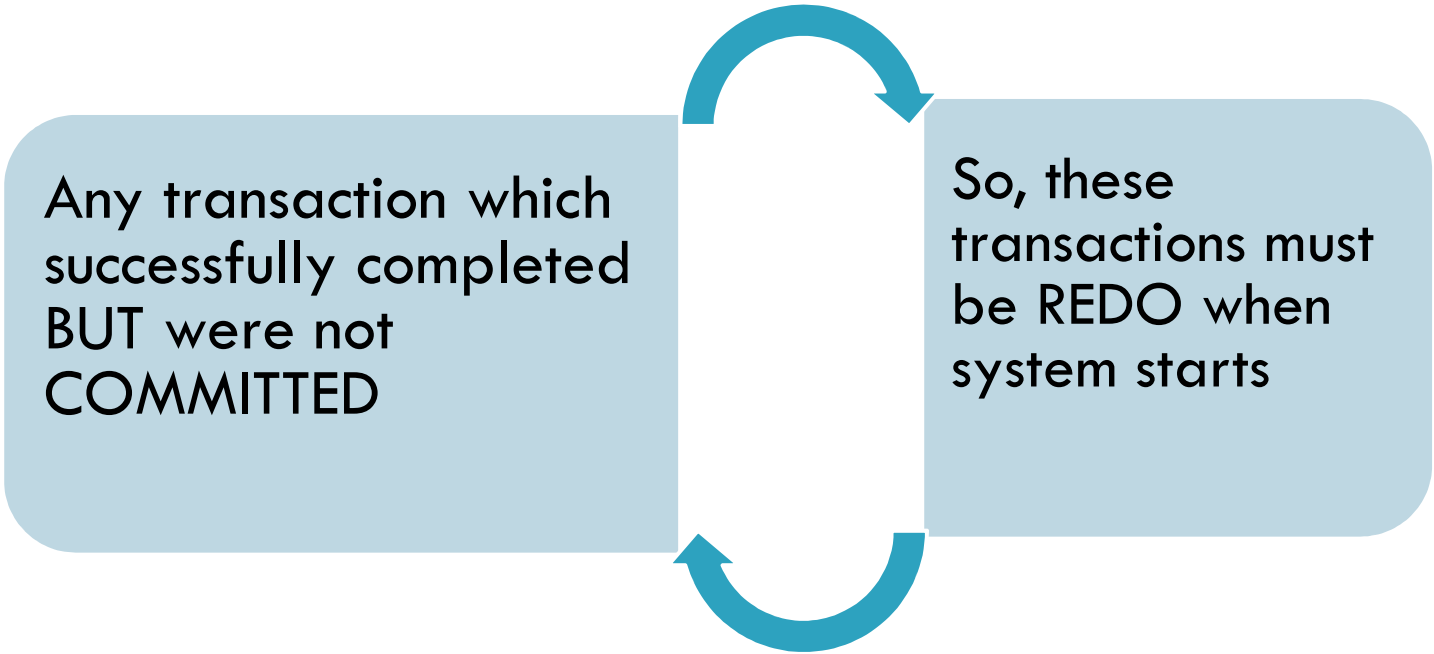
System Failure results in loss of content of Main Memory

[Database Buffer]

# Key Points


Any transaction in progress at the time of failure can not be successfully completed
So, these transactions must be UNDONE [rollback] when system starts

# Key Points


Any transaction which successfully completed BUT were not COMMITTED
So, these transactions must be REDO when system starts

# Checkpoint

Log Files may also contain CHECKPOINT records

CHECKPOINT is a certain prescribed or scheduled time interval system takes

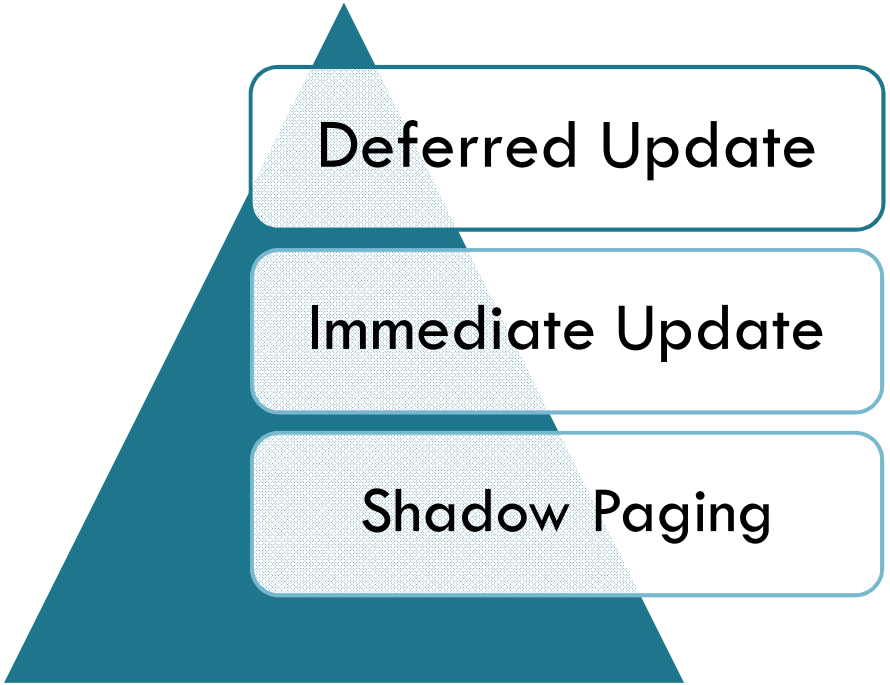
# At savepoint

Physically writes the content of database buffer to physical database

Physically writes a special checkpoint record to the log

Checkpoint record gives a list of transactions in progress at the time the savepoint is taken

# Techniques for Recovery



# Deferred Update

- Under this recovery protocol, UPDATES are not WRITTEN to database until TRANSACTION has reached its COMMIT point
- If TRANSACTION fails before reaching its COMMIT point, it doesn't modify database, hence NO action is needed.
- If SYSTEM FAILURE occurs after COMMIT point, REDO the updates of COMMITTED TRANSACTION

# Intermediate Update

- Under this recovery protocol, UPDATES are WRITTEN to database immediately without waiting to reach COMMIT point
- If TRANSACTION fails, the system should UNDO the updates made by TRANSACTION not COMMITTED at the time of failure.
- May be necessary to REDO the updates of COMMITTED TRANSACTIONS

# Intermediate Update – Key Operations

## REDO

- Transaction which completed successfully before the crash/failure

## UNDO

- Transaction which started but did not complete before the crash/failure

# Intermediate Update – Process

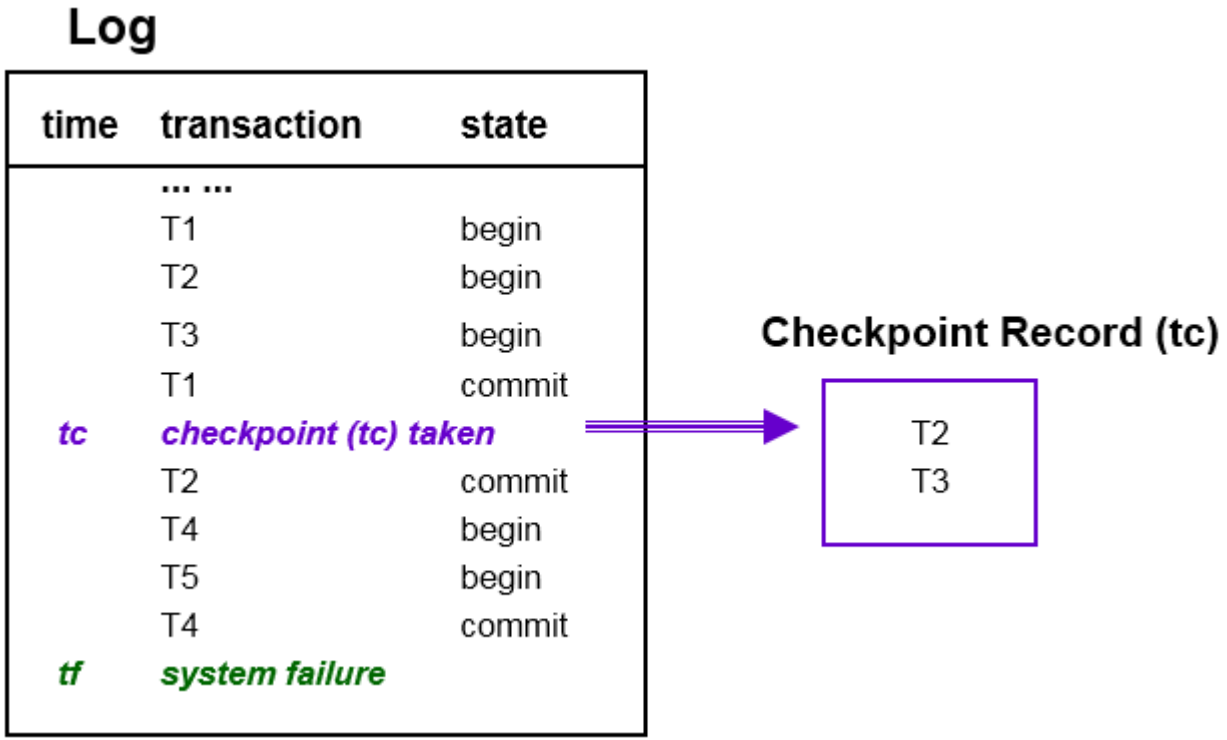
## UNDO List

- If begin transaction is found
- Backward Direction

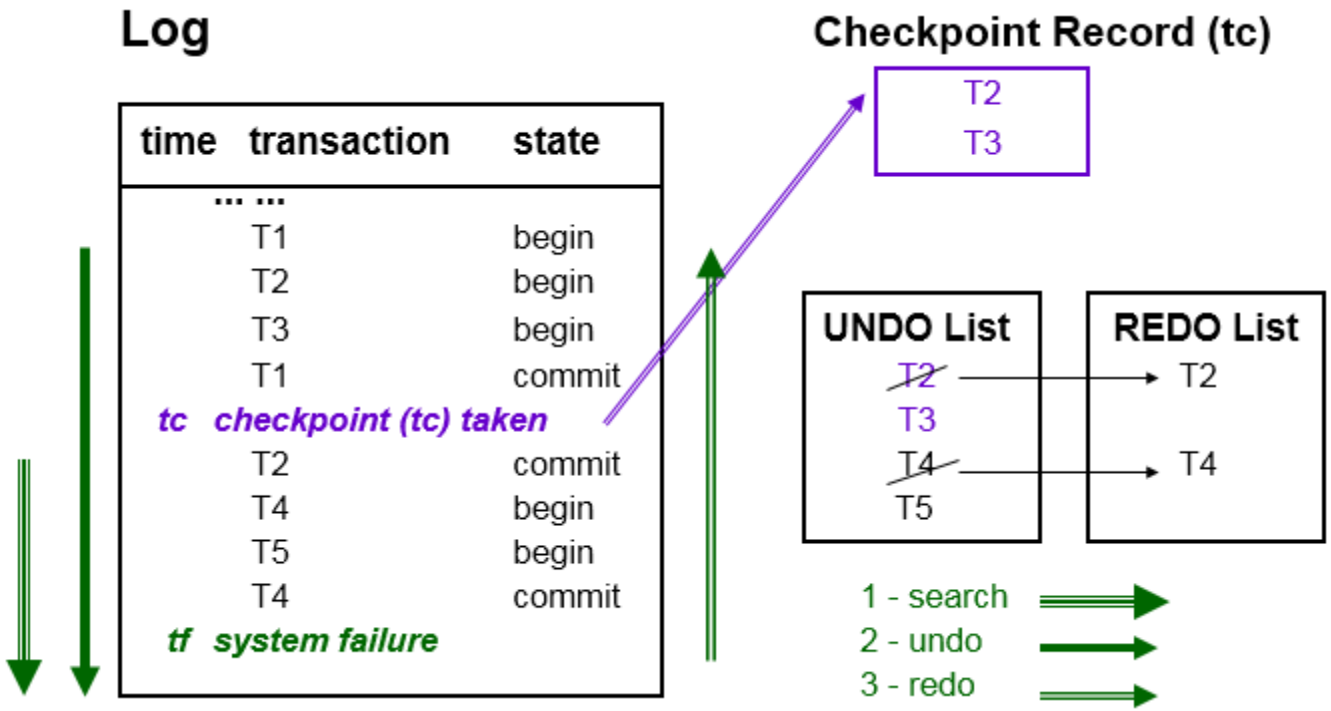
## REDO List

- If commit is found, move from UNDO to REDO
- Forward Direction

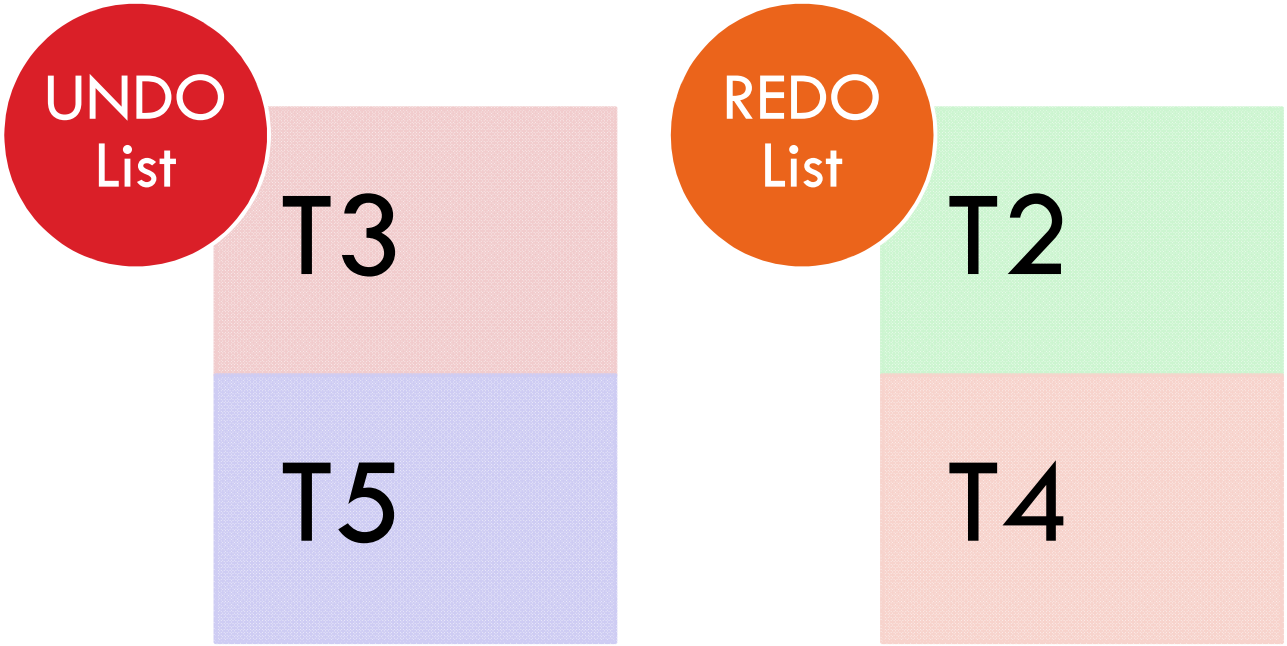
# Intermediate Update - Example



# Intermediate Update - Example

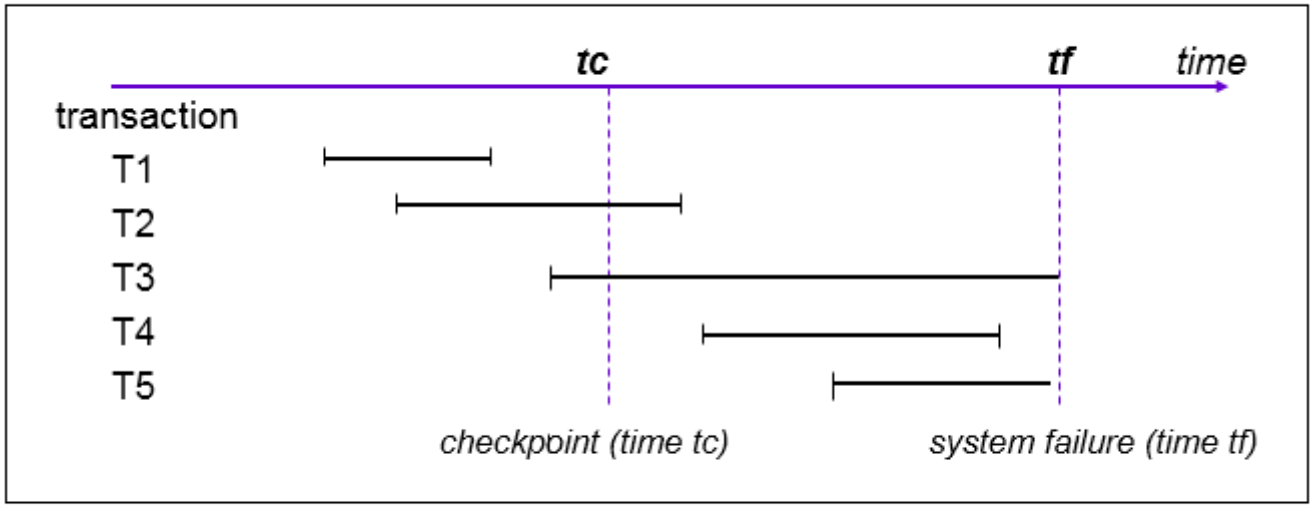


# Intermediate Update - Solution



# Intermediate Update - Question

**Consider the various transactions in the case:**

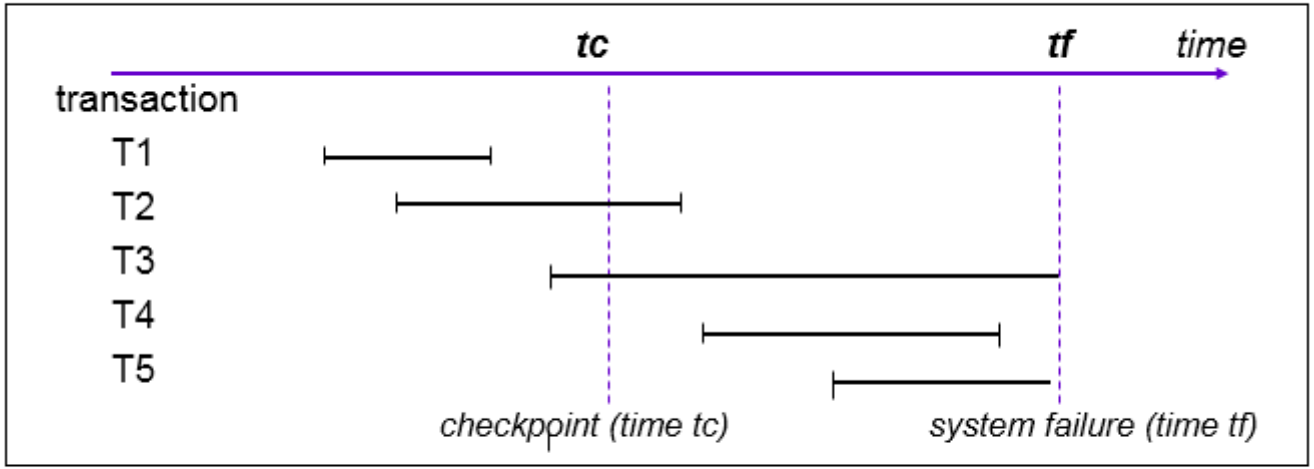



**Question:**

Which transaction(s) need to be undone, redone, or done nothing?

# Intermediate Update - Question

**Answer:**




**REDO:** T2, T4

**UNDO:** T3, T5

**Do nothing:** T1

# Shadow Paging

**No UNDO**

**No REDO**

**approach to**

**RECOVERY**

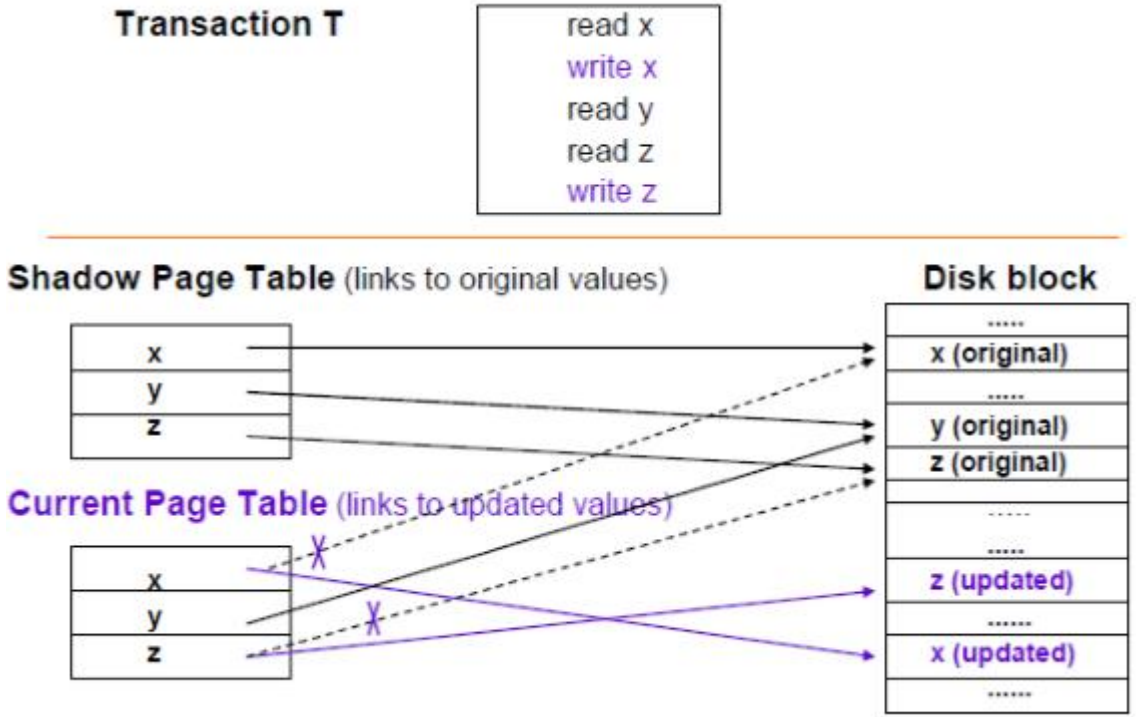
# Shadow Paging - Idea

- Database is partitioned into fixed-length blocks referred to as PAGES.
- Page table contains n entries
- Maintain 2 pages tables during life of transaction
  - The current page table
  - The shadow page table

# Shadow Paging - Process

- When transaction starts BOTH page tables are IDENTICAL
- The SHADOW page is NEVER changed over the duration of transaction
- The CURRENT page table may be changed when transaction performs WRITE operation

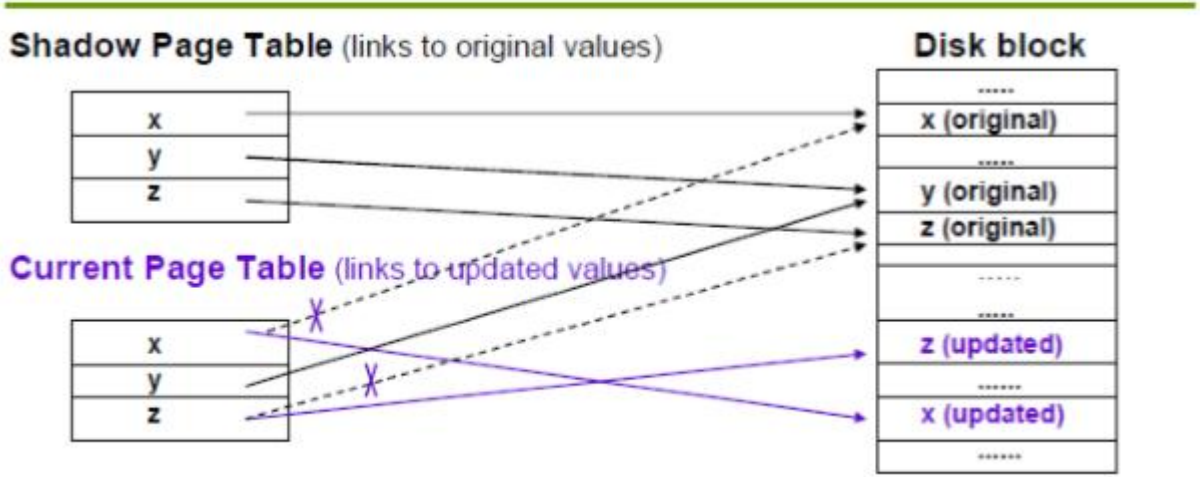
# Shadow Paging - Example



# Shadow Paging - Example

**At commit:-** delete shadow page table
current page table becomes new 'shadow' page table

**At failure:-** delete current page table
shadow page table provides the original data values



# SUMMARY

- Importance of Database Recovery
- Principles of Database Recovery
- Concepts of Transaction
- Role of LOG File
- Three Types of Failures
- Recovery Techniques for System Failure

# Thank You