SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING-IOT Including CS&BCT

COURSE NAME : 19SB504 DATABASE MANAGEMENT SYSTEMS

III YEAR / V SEMESTER

Unit V- CONCURRENCY CONTROL AND RECOVERY SYSTEM

Topic : **MULTIPLE GRANULARITY. RECOVERY SYSTEM - FAILURE CLASSIFICATION**

Objective:

Multiple granularity is **hierarchically breaking up our database into smaller blocks that can be locked.**

This locking technique allows the **locking of various data sizes and sets.**

This way of breaking the **data into blocks** that can be locked decreases lock overhead and increases the concurrency in our database.

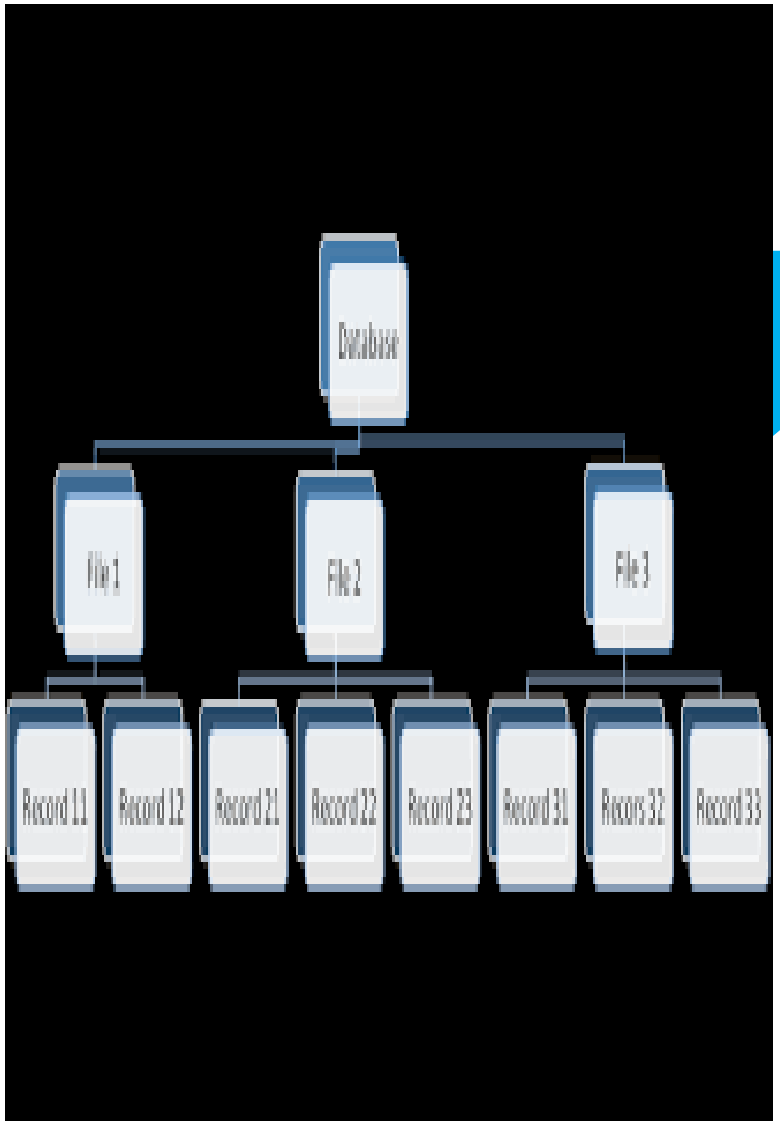
It refers to the **ability of a system to handle data at different levels of detail** or granularity.

It is the **size of data item allowed to lock.**

It can be defined as **hierarchically breaking** up the database **into blocks** which can be locked.

It makes easy to decide either to **lock a data** item or to **unlock a data** item.

This type of hierarchy can be **graphically represented as a tree.**



For example: Consider a tree which has four levels of nodes.

- The **first level** or higher level shows the **entire database.**

- The **second level** represents a **node of type area**. The higher level database consists of exactly these areas.

- The **area consists of children nodes** which are known as **files**. No file can be present in more than one area.

Finally, each **file contains child nodes** known as **records**. The file has exactly those records that are its child nodes. No records represent in more than one file.

Hence, the levels of the tree starting from the top level are as follows:

- **Database**
- **Area**
- **File**
- **Record**

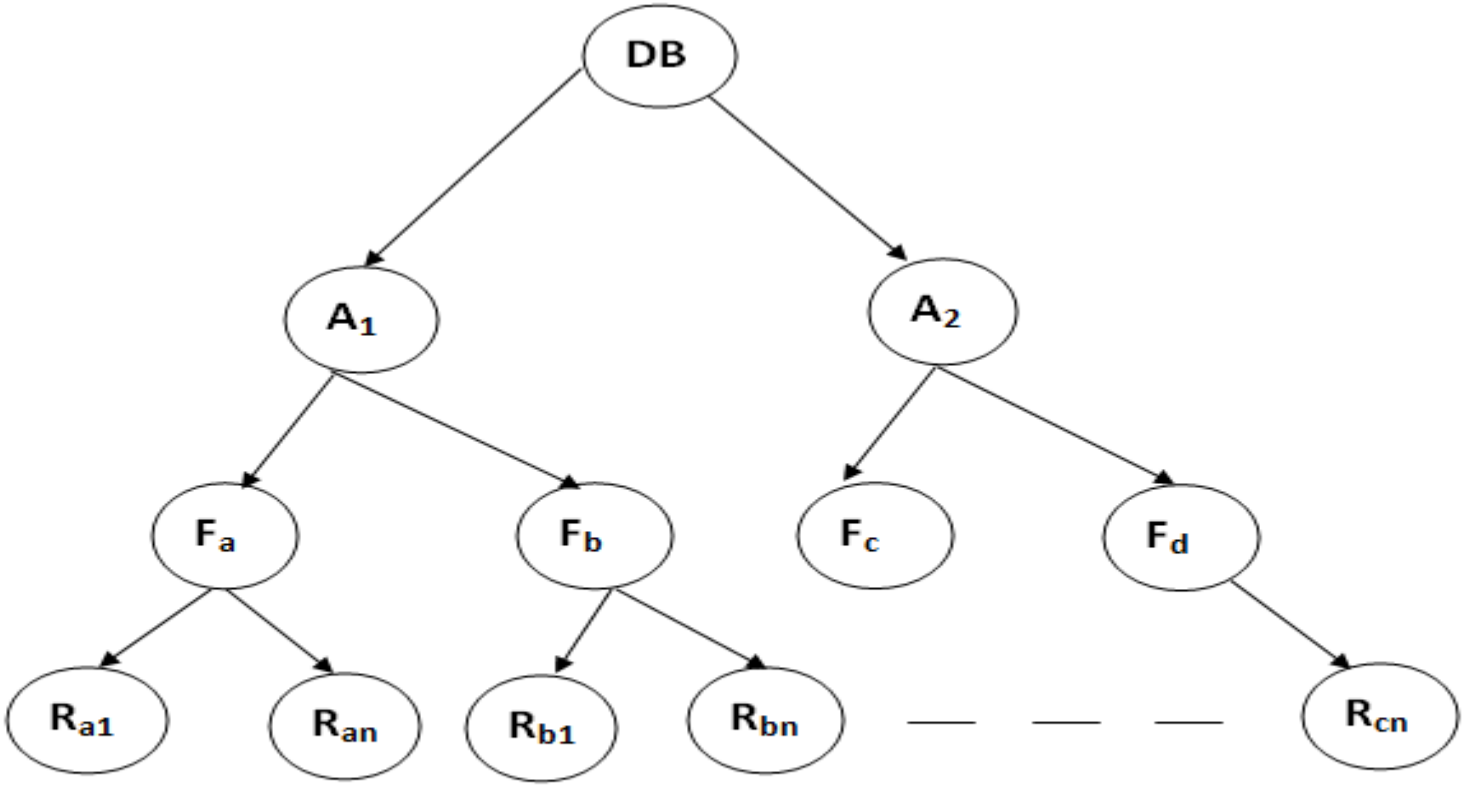


Figure: Multi Granularity tree Hierarchy

In this example, the highest level shows the entire database. The levels below are file, record, and fields.

Here are **three additional lock modes** with multiple granularity:

Intention Mode Lock

Intention-shared (IS): It contains explicit locking at a lower level of the tree but only with shared locks.

Intention-Exclusive (IX): It contains explicit locking at a lower level with exclusive or shared locks.

Shared & Intention-Exclusive (SIX): In this lock, the node is locked in shared mode, and some node is locked in exclusive mode by the same transaction.

Recovery system

Objective:

Database recovery techniques are used in database management systems (DBMS) to **restore a database** to a consistent state **after a failure** or error has occurred.

The **main goal** of recovery techniques is to ensure data **integrity and consistency** and prevent **data loss**

There are mainly **two types** of recovery techniques used in DBMS

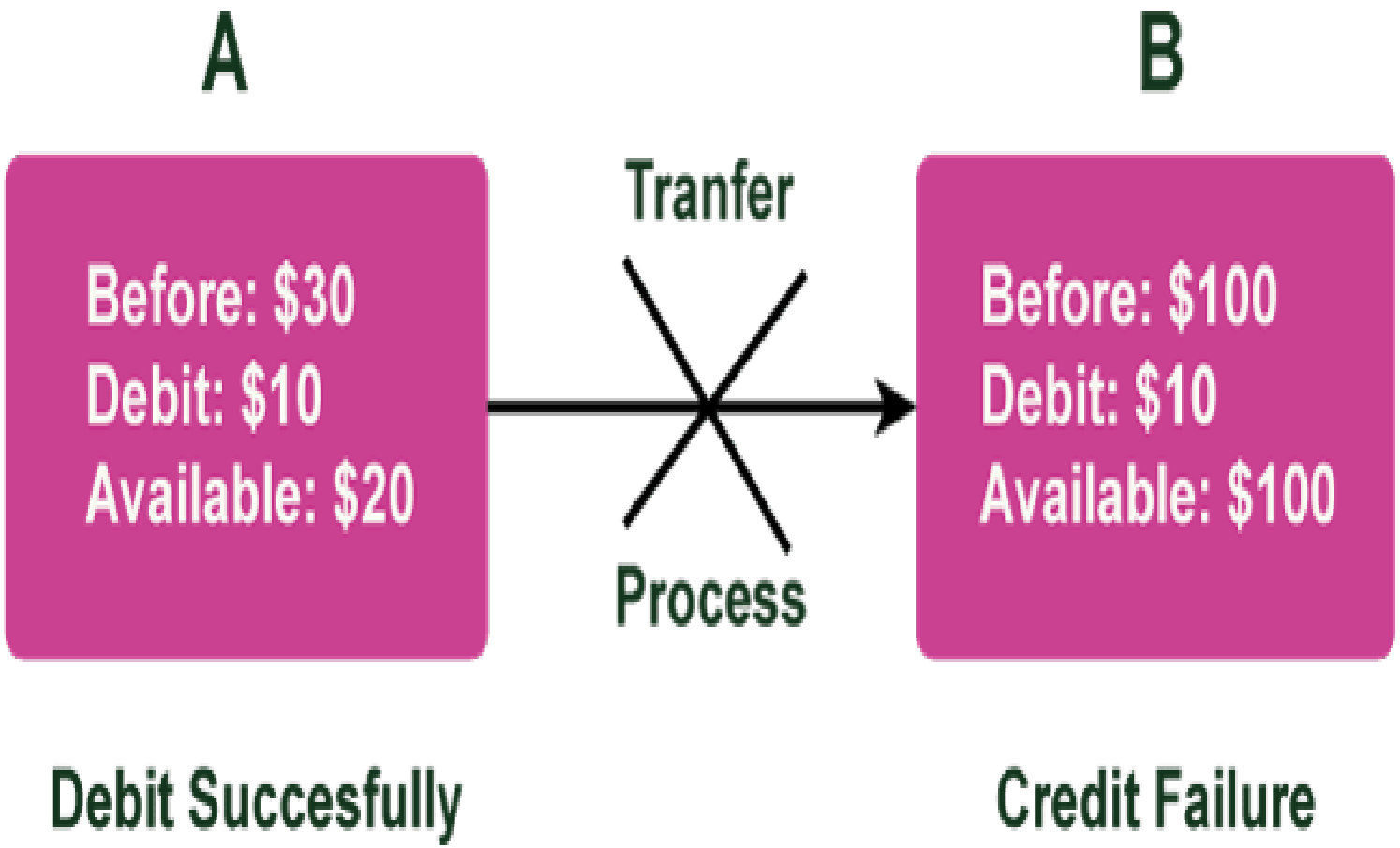
1. Rollback/Undo Recovery Technique
2. Commit/Redo Recovery Technique

1.Rollback/Undo Recovery Technique

The rollback/undo recovery technique is based on the principle of **backing out or undoing the effects** of a transaction that has not been completed successfully **due to a system failure or error.**

The system uses the **log records** to **undo the changes made by the failed transaction and restore the database to its previous state.**


CODING NINJAS
RECOVERY TECHNIQUES
IN DBMS


A
B
Tranfer
Before: $30
Debit: $10
Available: $20
Process
Before: $100
Debit: $10
Available: $100
Debit Succesfully
Credit Failure

2. Commit/Redo Recovery Technique

The commit/redo recovery technique is based on the principle of **reapplying the changes** made by a transaction that has been completed successfully to the database.

The system uses the log records to reapply the changes made by the transaction and restore the database to its most recent consistent state.

This technique is accomplished by using the log records stored in the transaction log to redo the changes made by the transaction that was in progress at the time of the failure or error.

Failure Classification in DBMS

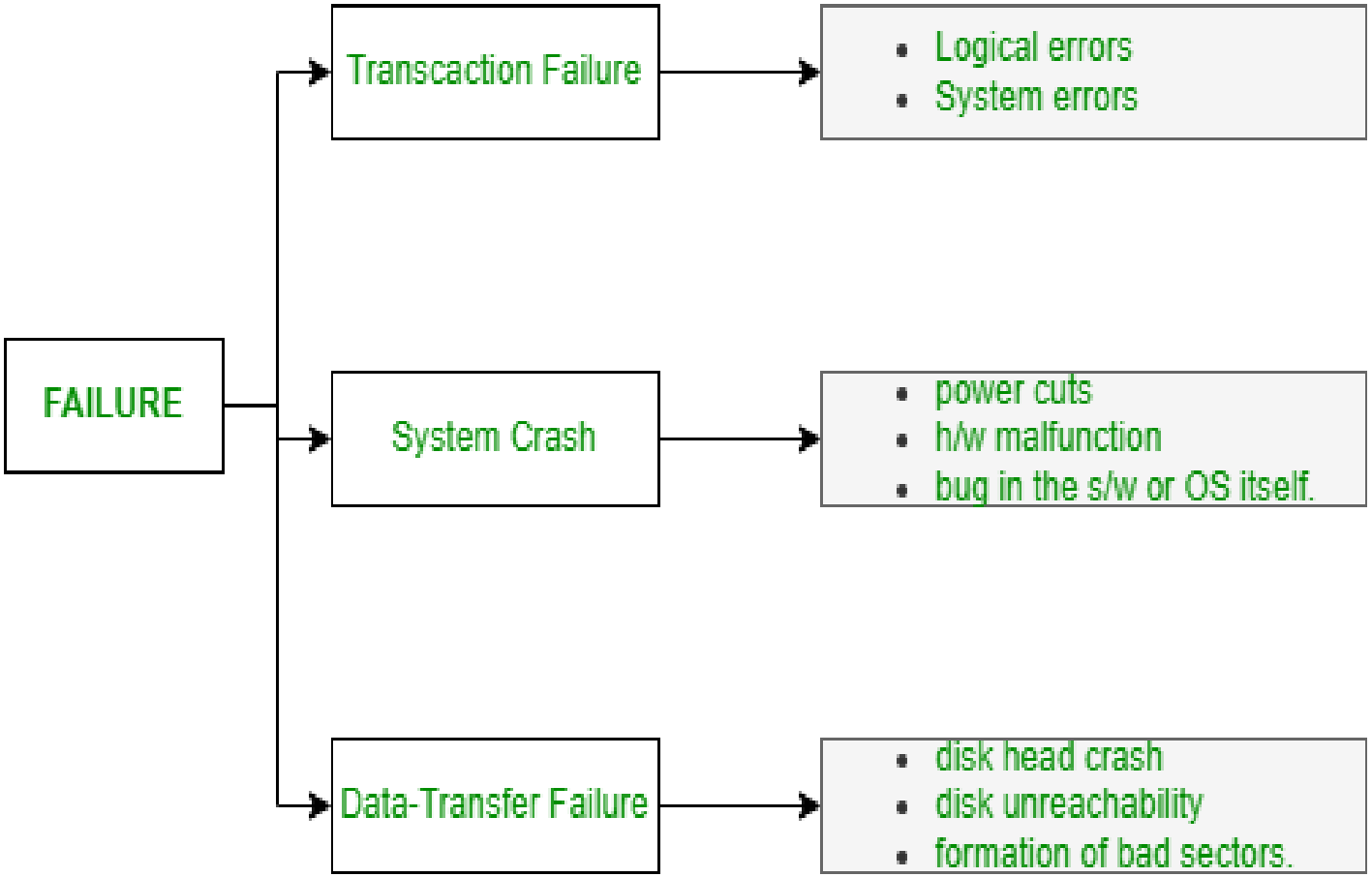
Objective:

Failure in terms of a database can be defined as its **inability to execute the specified transaction** or loss of data from the database.

A DBMS is vulnerable to several kinds of failures and each of these failures needs to be managed differently.

There are **many reasons that can cause database failures** such as **network failure, system crash, natural disasters, carelessness, sabotage(corrupting the data intentionally), software errors, etc.**

Failure Classification in DBMS



Transaction Failure:

If a transaction is **not able to execute** or it comes to a point from where the transaction becomes incapable of executing further then it is termed as a failure in a transaction.

Logical error:

A logical error occurs if a transaction is unable to execute because of **some mistakes** in the **code** or due to the presence of some **internal faults.**

System error:

Where the termination of an active transaction is done by the database system itself **due to some system issue** or because the database management system is unable to proceed with the transaction.

For example– The system **ends an operating transaction if it reaches a deadlock condition** or if there is an unavailability of resources.

System failure can occur due to **power failure** or other **hardware or software failure.**

Example: Operating system error.

Fail-stop assumption: In the system crash, non-volatile storage is assumed not to be corrupted.

Disk Failure

It occurs where **hard-disk drives or storage drives** used to fail frequently.

It was a common problem in the early days of technology evolution.

Disk failure occurs due to the **formation of bad sectors, disk head crash, and unreachability** to the disk or any other failure, which destroy all or part of disk storage.

Thank you .....