DBMS

Module 5
part 1

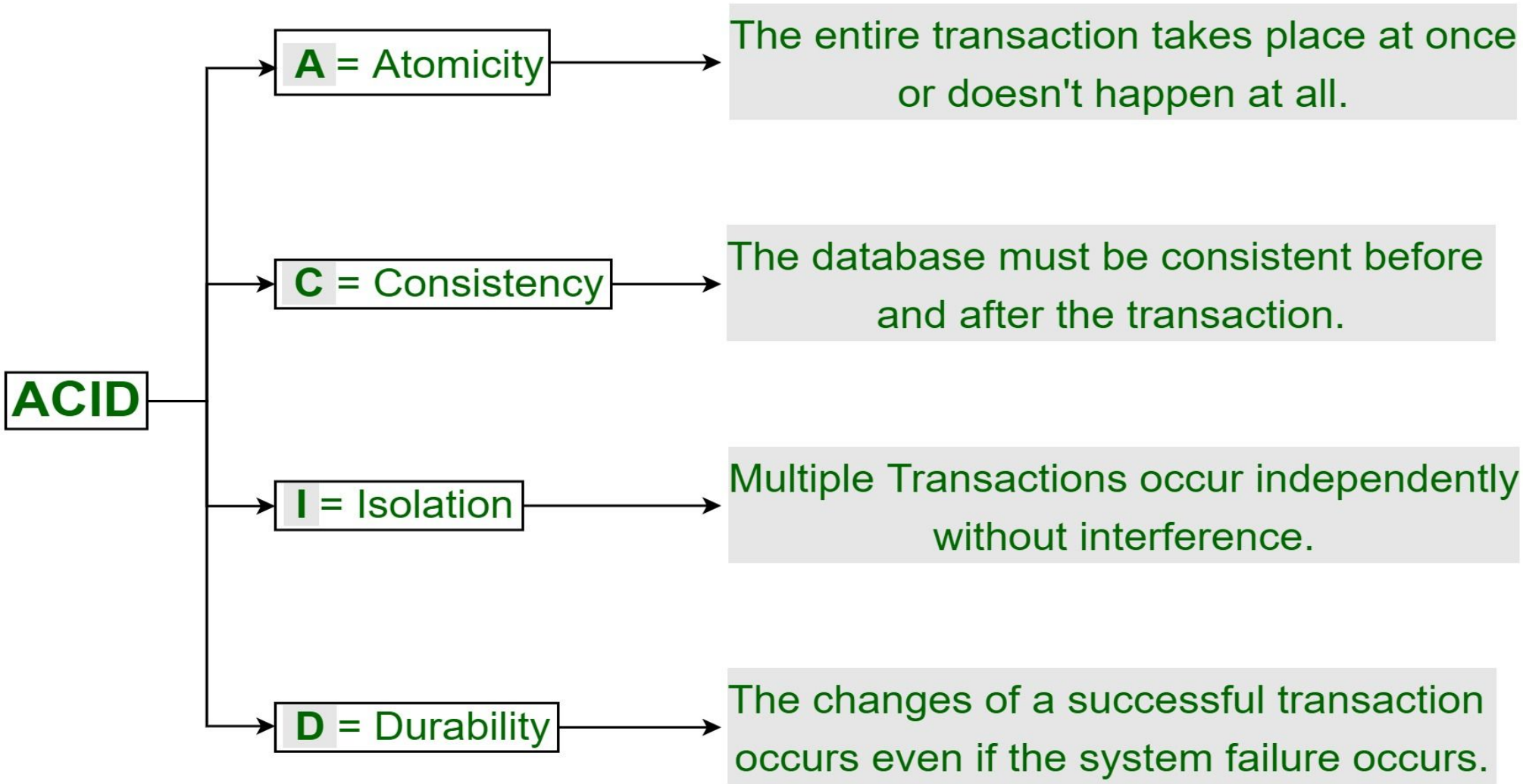
INDEX

- Transaction
  - ACID Properties
  - Concurrency control
  - Transaction Model
  - Transaction States
  - System Log

TRANSACTION

- **Transactions** group a set of tasks into a single execution unit.
- Each transaction begins with a specific task and ends when all the tasks in the group successfully complete.
- If any of the tasks fail, the transaction fails. Therefore, a transaction has only two results: **success or failure.**
- Incomplete steps result in the failure of the transaction.
- A database transaction, by definition, **must be atomic, consistent, isolated and durable**. These are popularly known as **ACID properties**.

ACID Properties in DBMS



CONCURRENCY CONTROL

Concurrency Control in Database Management System is a procedure of **managing simultaneous operations** **without conflicting with each other**

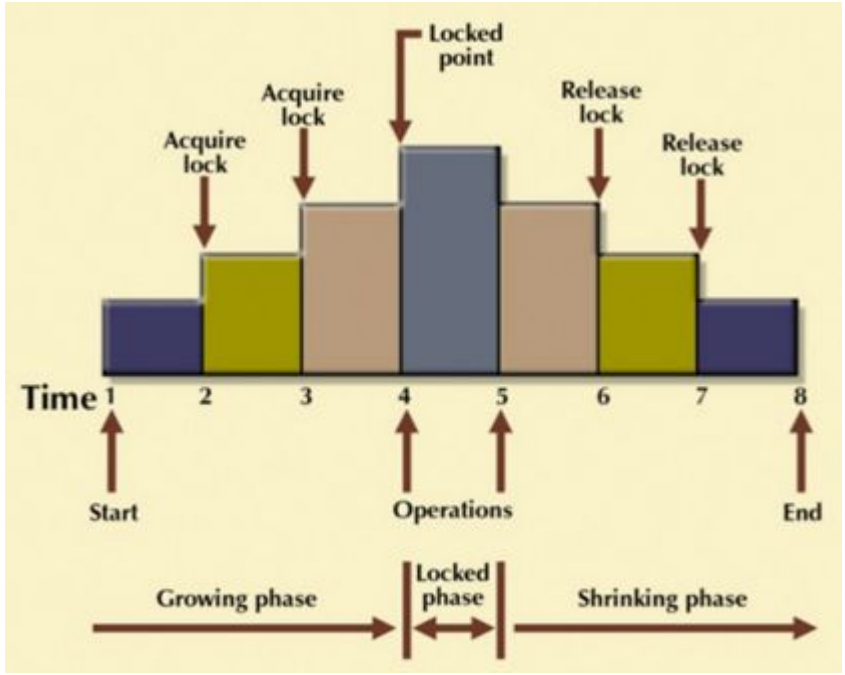
- Concurrency Control Protocols
  - Lock-Based Protocols
  - Two Phase Locking Protocol
  - Timestamp-Based Protocols
  - Validation-Based Protocols

1.LOCK-BASED PROTOCOLS

- Lock Based Protocols in DBMS is a mechanism in which a **transaction cannot Read or Write the data until it acquires an appropriate lock**.
- Lock based protocols help to eliminate the concurrency problem in DBMS for simultaneous **transactions** by locking or **isolating a particular transaction to a single user.**
- All **lock requests** are made **to** the **concurrency-control manager**. Transactions proceed only once the lock request is granted.

TWO PHASE LOCKING PROTOCOL (2 PL Protocol)

- It is a method of concurrency control in DBMS that **ensures serializability** by applying a **lock to the transaction data which blocks other transactions to access the same data simultaneously**



Growing Phase: In this phase transaction may obtain locks but may not release any locks.

Shrinking Phase: In this phase, a transaction may release locks but not obtain any new lock

TIMESTAMP-BASED PROTOCOLS

- It is an algorithm which uses the **System Time or Logical Counter as a timestamp to serialize the execution** of concurrent transactions.
- It ensures that every conflicting read and write operations are executed in a timestamp order.
- The **older transaction** is always **given priority** in this method.
- It **uses system time to determine the time stamp** of the transaction.
- This is the most **commonly used** concurrency protocol.

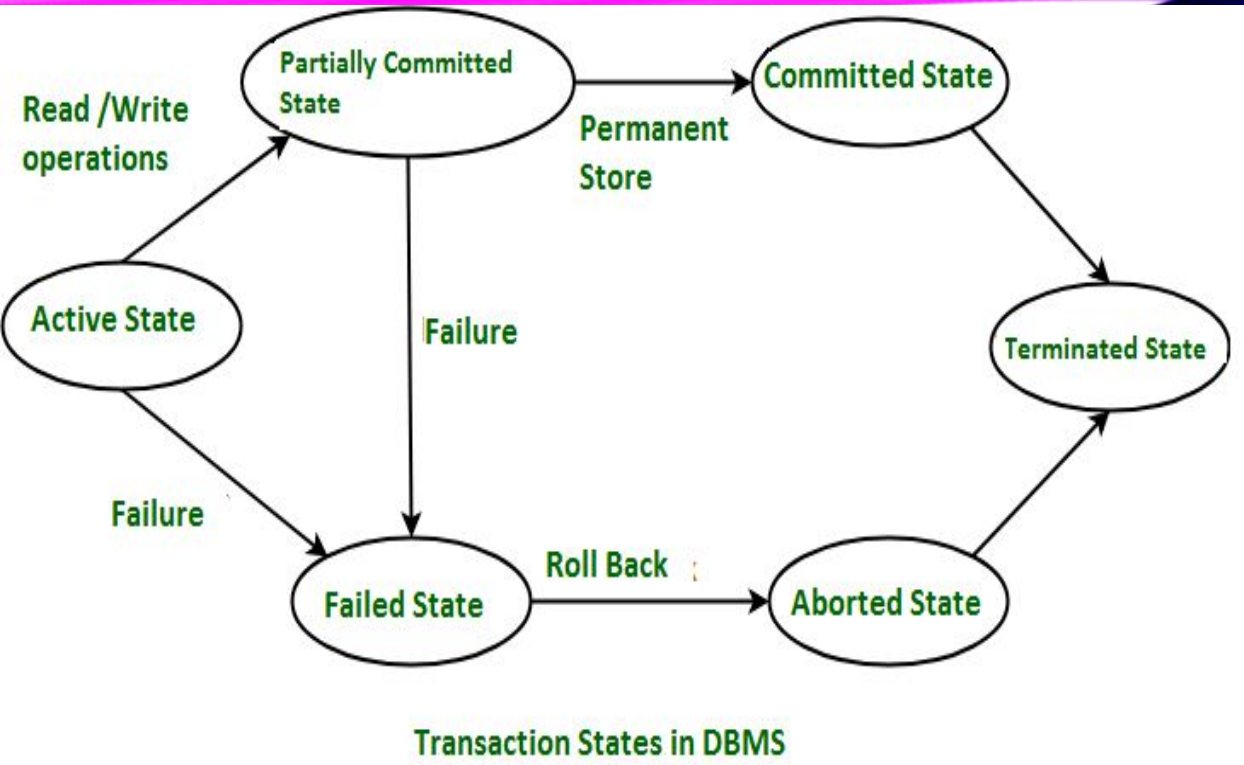
VALIDATION BASED PROTOCOL

- It is also called **Optimistic Concurrency Control Technique**.
- It is called optimistic **because of the assumption it makes**, i.e. very less interference occurs, therefore, there is no need for checking while the transaction is executed.
- Until the transaction end is reached updates in the transaction are not applied directly to the database. All updates are applied to local copies of data items kept for the transaction. **At the end of transaction** execution, while execution of the transaction, a validation phase checks whether any of transaction updates violate serializability. If there is **no violation** of serializability the **transaction is committed** and the **database is updated**

TRANSACTION MODEL

Transactions access data using two operations:

- **read(X)**, which transfers the data item X from the database to a variable, also called X, in a buffer in main memory belonging to the transaction that executed the read operation.
- **write(X)**, which transfers the value in the variable X in the main-memory buffer of the transaction that executed the write to the data item X in the database.

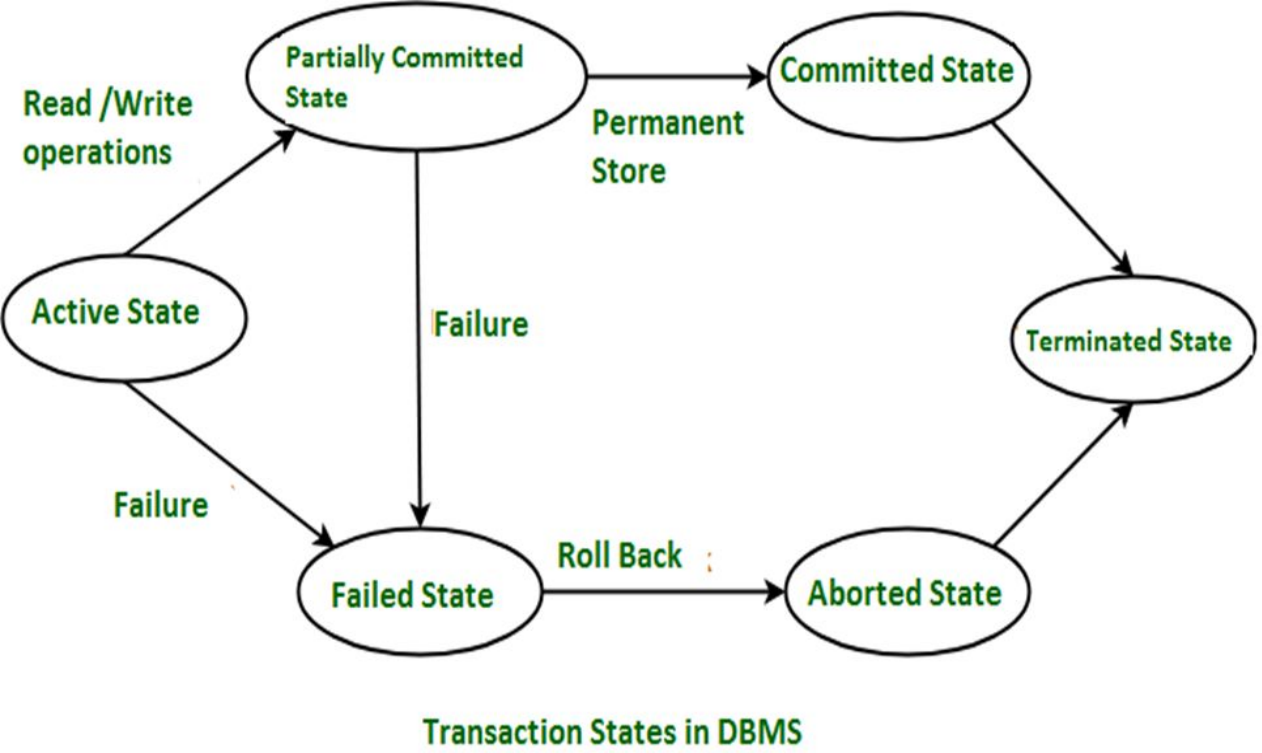


Transaction States in DBMS

TRANSACTION STATES

- **Active State:When the instructions of the transaction are running** then the transaction is in active state. If all the '**read and write**' operations are performed without any error then it goes to the "**partially committed state**"; if any instruction **fails**, it goes to the "**failed state**".

- **Partially Committed:After completion of all the read and write** operation the changes are made in main memory or local buffer. If the the **changes are made permanent** on the Data Base then the state will change to "**committed state**" and in case of **failure** it will go to the "**failed state**".
- **Failed State**:When **any instruction of the transaction fails**, it goes to the "failed state"



Transaction States in DBMS

- **Aborted State** :**After having any type of failure** the transaction goes from "**failed state**" **to** "**aborted state**"

- **Committed State**:**It is the state when the changes are made permanent on the Data Base** and the transaction is complete and therefore terminated in the "**terminated state**".
- **Terminated State** :the **transaction comes** from the "**committed state**" goes to this state, then the **system is consistent and ready for new transaction** and the old transaction is terminated.

SYSTEM LOG

- **Log is a sequence of records**, which maintains the records of actions performed by a transaction.
- It is **important that the logs are written prior to the actual modification and stored on a stable storage media**, which is failsafe.

Log-based recovery works as follows –

- The log file is kept on a stable storage media.
- When a transaction enters the system and starts execution, it writes a log about it.
- **<Tn, Start>**
- When the transaction modifies an item X, it write logs as **<Tn, X, V1, V2>.**It reads Tn has changed the value of X, from V1 to V2.
- When the transaction finishes, it logs **<Tn, commit>**

THANK YOU