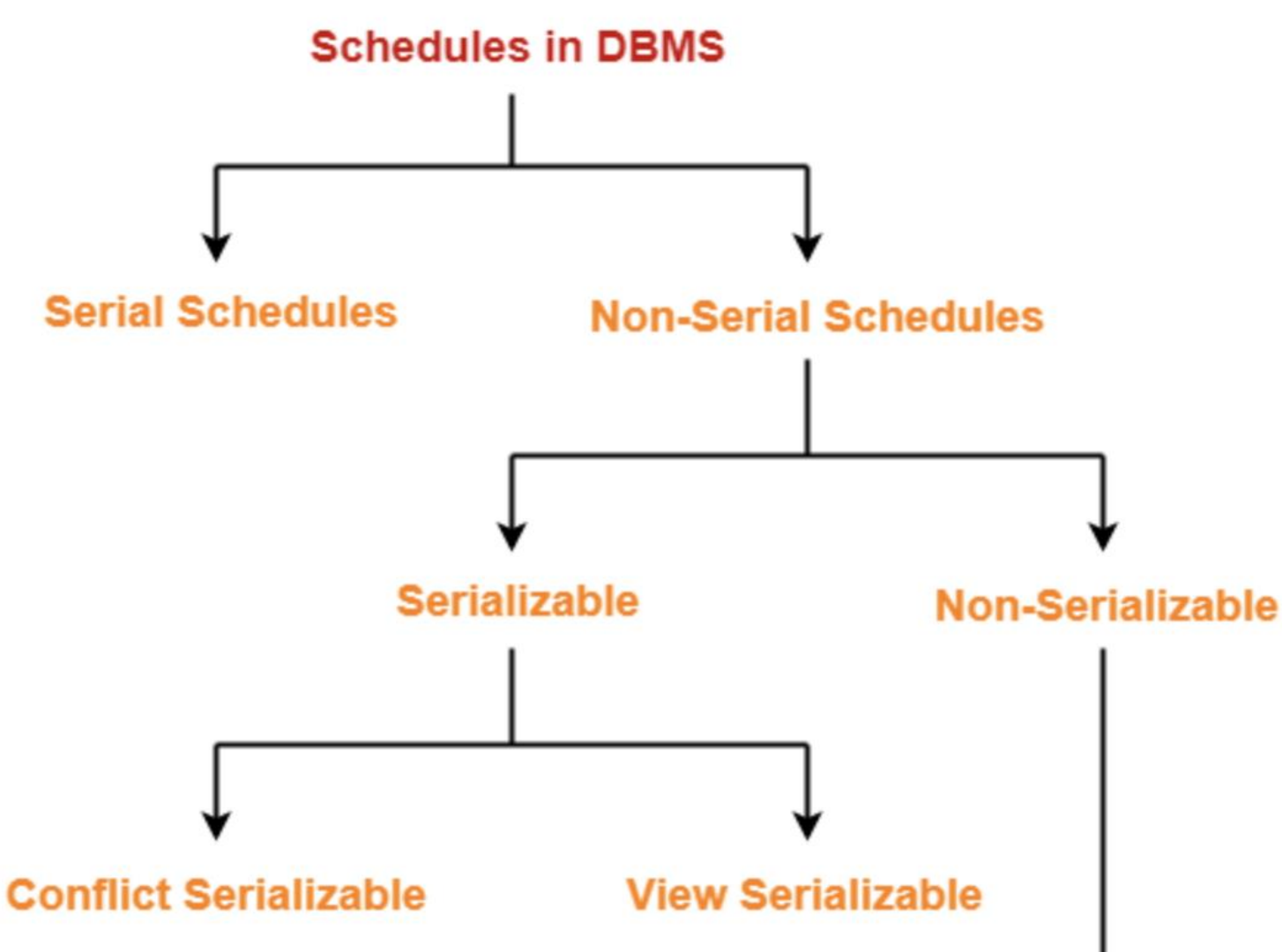
DBMS

PANA ACADEMY


Schedules in DBMS
Serial Schedules
Non-Serial Schedules
Serializable
Non-Serializable
Conflict Serializable
View Serializable

Serial Schedule

Serial Schedules-

In serial schedules,

- All the transactions execute serially one after the other.
- When one transaction executes, no other transaction is allowed to execute.

Characteristics-

Serial schedules are always-

- Consistent
- Recoverable
- Cascadeless
- Strict

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

Non-serial Schedule

In non-serial schedules,

- Multiple transactions execute concurrently.
- Operations of all the transactions are interleaved or mixed with each other.

Characteristics-

Non-serial schedules are **NOT** always-

- Consistent
- Recoverable
- Cascadeless
- Strict

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

Serializability

Serializability is a property of a schedule (sequence of operations from multiple transactions) that ensures the outcome of the execution of the transactions is the same as if the transactions were executed in some serial order (one after the other, without overlapping in time).

If a given non-serial schedule of ‘n’ transactions is equivalent to some serial schedule of ‘n’ transactions, then it is called as a **serializable schedule**.

Types of Serializability

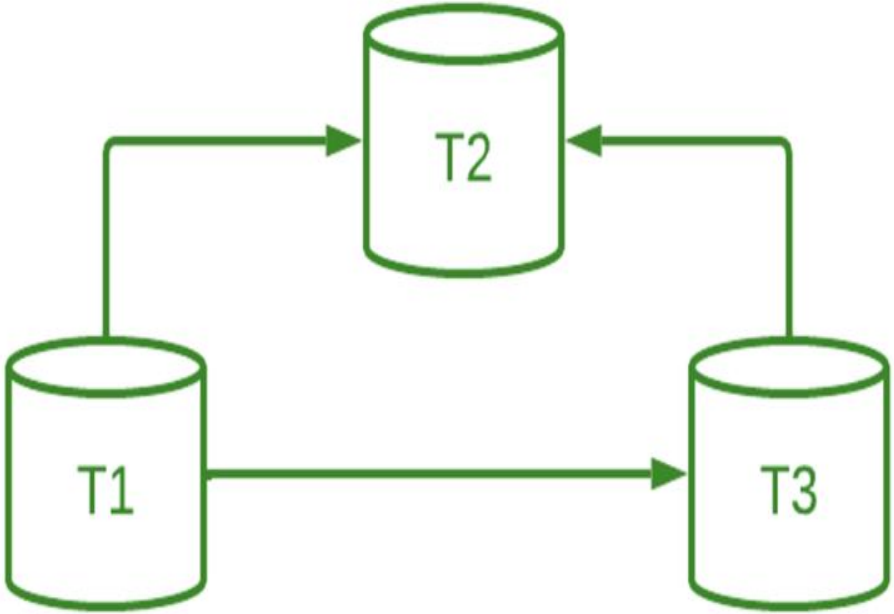
1. **Conflict Serializability**:
    - If a given non-serial schedule can be converted into a serial schedule by swapping its non-conflicting operations, then it is called as a **conflict serializable schedule**.
2. **View Serializability**:

    If a given schedule is found to be view equivalent to some serial schedule, then it is called as a view serializable schedule.

    Tips

- All conflict serializable schedules are view serializable.
- All view serializable schedules may or may not be conflict serializable.

It is a conflict serializable schedule as well as a serial schedule because the graph (a DAG) has no loops. We can also determine the order of transactions because it is a serial schedule.



T1	T2
READ1(A)	
WRITE1(A)	
READ1(B)	
C1	
	READ2(B)
	WRITE2(B)
	READ2(B)
	C2

T1	T2
READ1(A)	
WRITE1(A)	
	READ2(B)
	WRITE2(B)
READ1(B)	
WRITE1(B)	
READ1(B)	

Conflicting Operations and Non-conflicting operations

- Readi(x) readj(x) - non conflict read-read operation
- Readi(x) writej(x) - conflict read-write operation.
- Writei(x) readj(x) - conflict write-read operation.
- Writei(x) writej(x) - conflict write-write operation.

Conflicting Operations

- Both the operations belong to different transactions
- Both the operations are on the same data item
- At least one of the two operations is a write operation

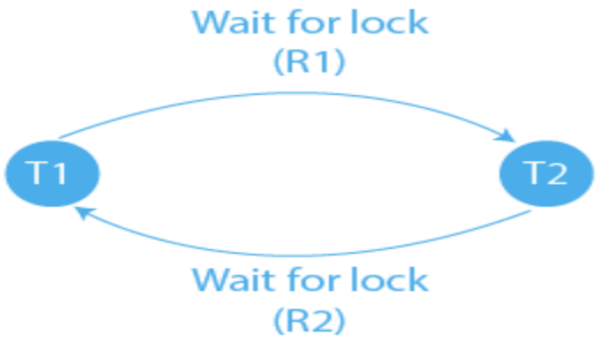
Important Points Regarding Serializability

1. A schedule is view serializable if it is view equivalent to a serial schedule.
2. Every conflict serializable schedule is view serializable, but not every view serializable schedule is conflict serializable.

Deadlock in database

A deadlock is a condition where two or more transactions are waiting indefinitely for one another to give up locks.

Wait-for Graph: This method uses a graph-based model to represent the allocation of resources. The graph is used to detect the existence of cycles, which indicate a deadlock has occurred. The wait-for graph is updated whenever a resource is requested or released.



Lock in database

1. Shared lock: It is also known as a Read-only lock. In a shared lock, the data item can only read by the transaction.

2. Exclusive lock: In the exclusive lock, the data item can be both reads as well as written by the transaction.

Lock Protocols in DBMS:

Two-phase locking (2PL)

Strict Two-phase locking (Strict-2PL)

2 phase locking protocol

Growing phase: In the growing phase, a new lock on the data item may be acquired by the transaction, but none can be released.

Shrinking phase: In the shrinking phase, existing lock held by the transaction may be released, but no new locks can be acquired.

Strict 2 phase locking protocol

- The only difference between 2PL and strict 2PL is that Strict-2PL does not release a lock after using it.
- Strict-2PL waits until the whole transaction to commit, and then it releases all the locks at a time.

Log based recovery

- The log is a sequence of records. Log of each transaction is maintained in some stable storage so that if any failure occurs, then it can be recovered from there.
- If any operation is performed on the database, then it will be recorded in the log.

Let's assume there is a transaction to modify the City of a student. The following logs are written for this transaction.

- When the transaction is initiated, then it writes 'start' log.
    1. <Tn, Start>
- When the transaction modifies the City from 'Noida' to 'Bangalore', then another log is written to the file.
    1. <Tn, City, 'Noida', 'Bangalore' >
- When the transaction is finished, then it writes another log to indicate the end of the transaction.
    1. <Tn, Commit>

Approach of modification

There are two approaches to modify the database:

1. Deferred database modification:

- The deferred modification technique occurs if the transaction does not modify the database until it has committed.
- In this method, all the logs are created and stored in the stable storage, and the database is updated when a transaction commits.

2. Immediate database modification:

- The Immediate modification technique occurs if database modification occurs while the transaction is still active.
- In this technique, the database is modified immediately after every operation. It follows an actual database modification.

Recovery mechanism on crash

Recovery using Log records

When the system is crashed, then the system consults the log to find which transactions need to be undone and which need to be redone.

1. If the log contains the record <Ti, Start> and <Ti, Commit> or <Ti, Commit>, then the Transaction Ti needs to be redone.
2. If log contains record<T_n, Start> but does not contain the record either <Ti, commit> or <Ti, abort>, then the Transaction Ti needs to be undone.

MCQ

https://www.sanfoundry.com/rdbms-questions-answers-lock-based-protocols/

https://www.sanfoundry.com/rdbms-questions-answers-serializability/

https://www.sanfoundry.com/database-mcqs-deadlocks/