UNIT-6

TRANSACTION MANAGEMENT

- **Transaction:** It is a program unit or logical unit whose execution may change the content of database.
- A Transaction can be accessed by using two Operations that are read and write Operations.
- For Example,In a Bank database initially ‘A’ has Rs.15000/- and ‘B’ has Rs.11000/- suppose ‘A’ can be transferred Rs.2000/- to ‘B’,the transaction can be done in following way. i.e.,

A=15000	B=11000
R(A)	R(B)
A=A-2000	B=B+2000
W(A)	W(B)
13000/-	13000/-

- Before and after Transaction database must be consistent.

A Properties Of Transaction Management (OR)

ACID Properties:

- Inorder to maintain consistency in a database before and after the exection certain properties are followed that properties are called as ‘ACID Properties’.
- A--- Atomicity
  C---Consistency
  I----Isolation
  D---Durability

1.Atomicity:

- It is the fundamental property of Transaction Management.
- The Transaction must be treated as Atomic Unit i.e., either all of its Operations are executed (or) None.
- Atomicity is also Known as All (or) Nothing Rule.
- There is no Midway i.e., Transaction don’t Occur Partially.

- Each Trasaction is considered as 1 unit and either run to completion (or) not executed at all..

Example:

A=15000	B=11000
R(A)	R(B)
A=A-2000	B=B+2000
W(A)	W(B)
13000/-	13000/-

- If Transaction fails after completion of transaction T1 But before completion of transaction T2 then amount has been deducted from T1 but not added to B.This result is called as “Inconsistent Database State”.

2.Consistency:

- This means that Integrity Constraiant must be maintained.So that database is Consistent before and after the transaction.
- Consistency also called as “Correctness Of Database”.
- For Example,The total amount before and after the transaction must be maintained in consistent Mode. i.e.,

Example:

A=15000	B=11000
R(A)	R(B)
A=A-2000	B=B+2000
W(A)	W(B)
13000/-	13000/-

Total amount before transaction (A+B)=15000+11000=26,000/-
Total amount after transaction (A+B)=13000+13000=26,000/-

3.Isolation:

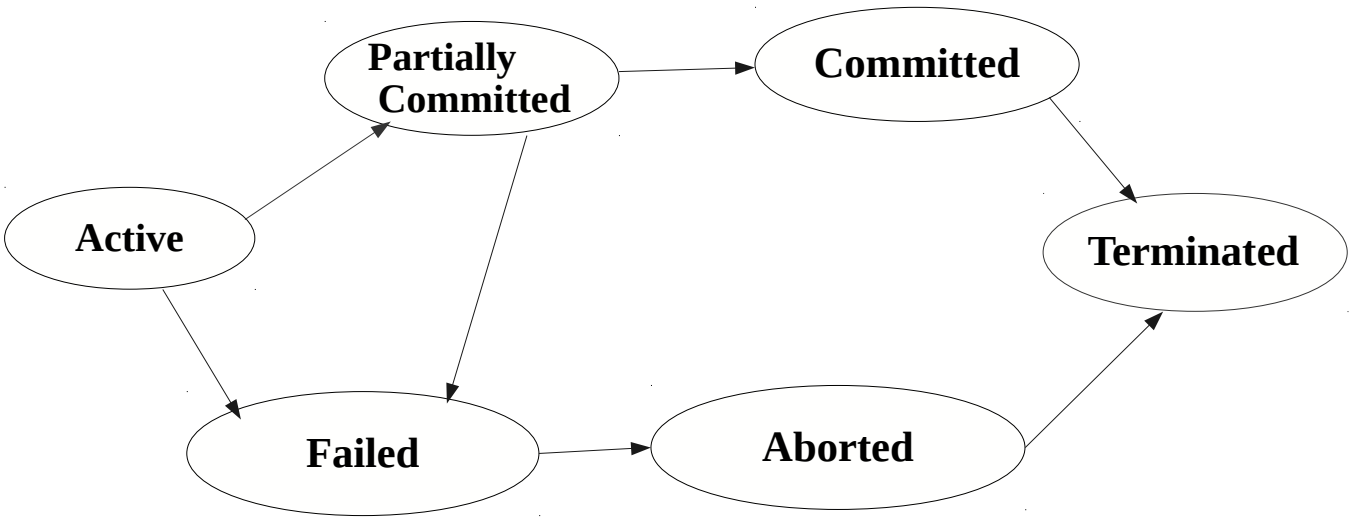
- This property describe that multiple Transaction can occur concurrently without reading to inconsistency of database state.
- Here One transaction can be occur independently with other transaction.
- The Changes occur in a particular transaction will not affect any other transaction.
- For every pair of Transaction,One transaction should start execution only when other finish the execution.

4.Durability:

- Once a Transaction Completed Successfully,the changes made into the database should be permanent if there is a system failure.
- The Modification to the Databse once stored in and written to the disk.if there is a system failure,the recovery management components are recover the data.

TRANSACTION STATES (or) LIFECYCLE OF TRANSACTION:

Transaction is goes through different states in its lifecycle.



(1)ACTIVE STATE:

- It is the first state of the transaction.
- A Transaction is enter into active state when execution process begins.
- During the state read/write operations can be performed.
- For example,You can insert,delete,update record is done here.But all the records are still not saved to the database.

(2)Partially Committed State:

- A Transaction is goes into partially committed state after the end of Transaction.
- After entering into state the Transaction is considered as partially committed and it is not cconsidered as fully committed because changes made by Transactions still stored in Buffer in Main Memory.
- When the Transaction is in Committed state,it has already completed its execution successfully and Moreover all of its changes are recovered to the database permanantly.
- Now the Transaction is considered as fully committed.
- After a Transaction enter into committed state then it is not possible to rollback the Transaction ,it means it is not possible to “UNDO” the changes that has been made by thr Transaction.

(4)FAILED STATE:

- When a Transaction is getting executed in the partially committed (or) active state and some failures occur due to impossible to continue its execution That’s why it enter into “Failed” State.

(5)ABORTED STATE:

- If one of the checks fails and the Transaction reached to failed state and then goes to aborted state.

- If the Transaction fails in the middle of the Transaction then before executing Transaction all the Transactions are rollback to consistency State.
- After Abort the Transaction,the database recovery state can select one of the following operation:
  1.Restart the Transaction
  2.Kill the Transaction

(6)TERMINATED STATE:

- It is Last state of the Life cycle of Transaction.
- After enter into committed state (or) Aborted state .The Transaction finally enter into terminated state..

CONCURRENCY CONTROL IN DBMS:

- It is the process of managing simultaneous execution of Transaction in a shared database,to ensure the serializability at Transaction.
- It may affect the transaction result.
- Here each and every transaction are isolated

CONCURRENCY CONTROL PROBLEMS:

- When multiple Transactions exexute concurrently in an uncontrolled manner,then it might lead to several problems.such problems are called “Concurrency Control Problems”
- There are different Concurrent Control Problems:
  1.Lost Update Problem (Write-Write)
  2.Dirty Read (Write-Read) (Uncommitted Data)
  3.Unrepeated Read

1.Lost Update Problem:

- The lost update problem occur when two transaction occur the same database item and one of the update is lost because Overwritten by the Other Transaction.
- Here update made by one transaction is overwritten by other Transaction.

Example: Let us consider two Transactions T1 and T2 with respective of time and X is data item.

T1	T2
Read(X)	
X=X+10	
	R(X)
	X=X-20
W(X)	
	W(X)

So,Update Transaction T1 is lost because Transaction T2 is Overwritten.so,it is Lost update problem.

2.Dirty Read(Uncommitted Data):

- Reading the data written by an uncommitted Transaction is called as “Dirty Read”
- Dirty Read problem occurred when one Transaction update the database item and the Transaction fail for some reason.
- The Updated Database item is accessed by another Transaction before it rollback to the Original value.

Example:

T1	T2
Read(X)	
X=X+10	
W(X)	
	R(X)
	X=X-20
	W(X)
	(commit)
rollback(Failure)	
	W(X)

- Here uncommitted data occurs when two Transactions T1 and T2 are executed concurrently and first Transaction T1 is rollback after the second Transaction T2 has already accessed the uncommitted data does violating the isolation property of Transaction.

3.Unrepeated Read(Inconsistent Retrivals):

- This Problem occur when the Transaction gets to read unrepeated i.e., different value of same variable in its different read Operations even when it has not updated its value.

Example:

T1	T2
Read(X)	
	R(X)
W(X)	
	W(X)

- Here T1 reads a value of 100 then T2 reads the value X=100,Now T1 updates Values of X in the buffer in main memory , X=110 and T2 again read the value of X=110.
- Here T2 gets to read [to reo] two different values in second reading.

SCHEDULE IN DBMS:

****Schedule**:**

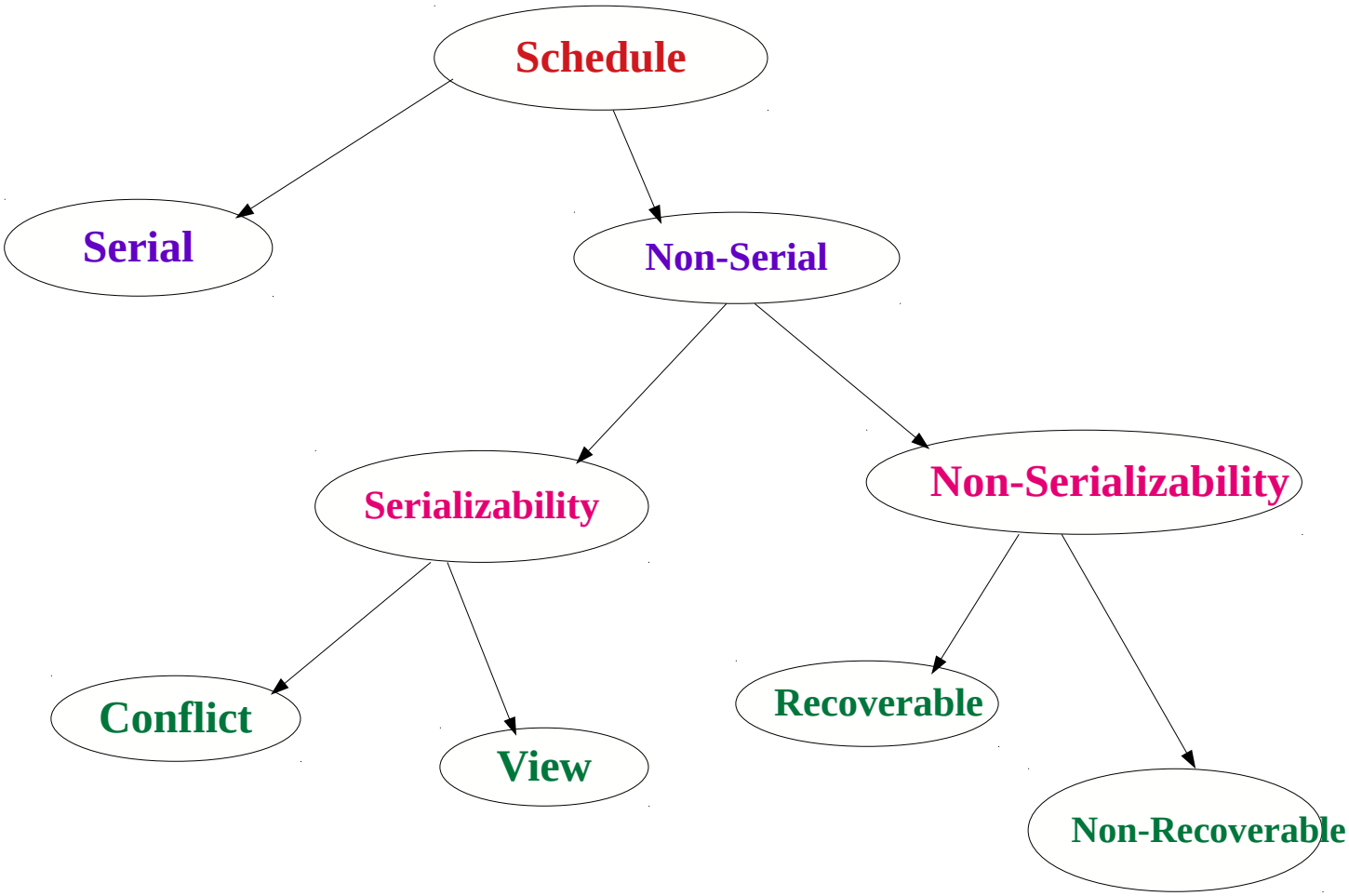
- When Multiple transaction are running concurrently then database goes in inconsistency state.
- Schedule is a process of sequence of operation done one after another.

Example:

T1	T2
R(X)	
W(X)	
	R(X)
	W(X)

- In above example the Transaction T1 and T2 are Running concurrently.Here the transaction T1 is executed first then goes to Transaction T2.

Types Of Schedules:



Serializable Schedule:

- If a given Serial Schedule of ‘n’ transactions is equalent to some Non-serial of ‘n’ transactions then it is called as “Serializable Schedule”.
- It is behave same as “Serial Schedule”
- Concurrency is allowed that means multiple transaction can execute concurrently.But it can improve both resource utilization and high throughput.

Types of Serializability:

There are 2 types: 1.Conflict Serializability
2.View Serializability

1.Conflict Serializability:

A Serializability said to be conflict,it should satisfy following conditions:

(1)Both the Operations belongs to different transaction.
(2)Both the Operations are works on same Data item.
(3)Atleast one of the Operation is write operation.

Example:

1. ($R_1(A)$,$W_2(A)$) (True)
2. ($W_1(A)$,$W_2(A)$) (True)
3. ($W_1(A)$,$R_2(A)$) (True)
4. ($R_1(A)$,$W_2(B)$) (False)
5. ($W_1(A)$,$W_2(B)$) (False)
6. ($R_1(A)$,$R_2(A)$) (True)

Precedence Graph Of Conflict Serilizability:

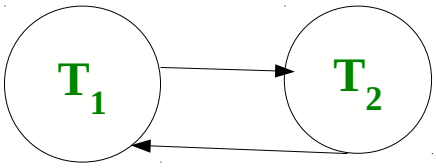
Note: For suppose different Transactions with Read & Write Operations from a cycle then it is not a conflict serializability otherwise it is conflict serializability.

Example:

(1)R1(x),R2(x),W1(x),R1(y),W2(x),W1(y)

Sol:

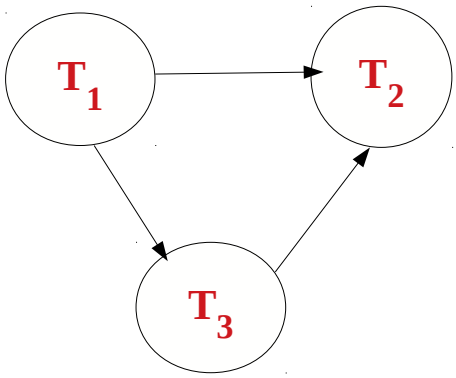
T_1	T_2
R(X)	
	R(X)
W(X)	
R(Y)	
	W(X)
W(Y)	



Therefore,it forms a cycle.So,it is not a Conflict Serializable..

(2)

T_1	T_2	T_3
R(A)		
R(B)		
	R(A)	
	R(C)	
W(B)		
		R(B)
		R(C)
		R(B)
	W(A)	
	W(B)	



Therefore,there is no cycle..so,It is a conflict.

2.View Serializability:

- A Serializability said to be view Serializability if it is equivalent to serial schedule.

- If Schedule is conflict Schedule the it will be view serializable but view serializability is not a conflict serializability.
- Two Schedules S1 and S2 said to be view equivaleent then it can be satisfy the following conditions..
  (1)Initial read
  (2)Updated read
  (3)Finl read

(1)Initial read:

The initial read of both the schedules must be same.Suppose two schedules s1 and s2,in schedule S1 if a transaction it is read the data item x then in S2 the Transaction T1 also read the data item x.

S1:

T_1	T_2
R(X)	
	W(X)

S2:

T_1	T_2
	W(X)
R(X)	

- In above example two schedules are equivalent initial read operation is s1 is done by transaction T1 and S2 it also done by transaction T1.

(2)Updated Read:

In Schedule S1 if Tj is reading x which is updated by Ti then in S2 also.

S1:

T_1	T_2	T_3
W(X)		
	W(X)	
		R(X)

S2:

T_1	T_2	T_3
	W(X)	
W(X)		
		R(X)

- In above example two schedules are not view equivalent because in s1,the Transaction T3 read(x) updated by T2 and in schedule S2 the Transaction T3 read(x) updated by T1.

S1:

T_1	T_2	T_3
W(X)		
	W(X)	
		R(X)

S2:

T_1	T_2	T_3
	W(X)	
W(X)		
	W(X)	
		R(X)

(3)Final Write:

A final Write must be same between both the schedules.In Schedule S1 if a transaction T3 update x at last operation with write,In schedule S2 also.

S1:--

T_1	T_2	T_3
R(A)		
	W(A)	
W(A)		
		W(A)

S2:--

T_1	T_2	T_3
R(A)		
W(A)		
	W(A)	
		W(A)

It is View equivalent..

****Note****: With 'n' Transactions the total no.of schedules are occurred that equal to "n!".

Ex: With 3 transactions the total no.of schedules are occured 3!=6.

S_1: T_1 T_2 T_3
S_2: T_1 T_3 T_2
S_3: T_2 T_1 T_3
S_4: T_2 T_3 T_1
S_5: T_3 T_1 T_2
S_6: T_3 T_2 T_1

Recoverability in Schedule:

If some transactions Tj is reading the values that can be updated (or) written by some other transactions Ti,then commit of Tj must be occur after the commit of Ti it is called “Recoverability”

Ti	Tj
R(A)	
W(A)	
	W(A)
	R(A)
commit	
	commit

In above example the schedule S is recoverable because Tj is commit after the Ti.

Irrecoverability:

The schedule is said to be irrecoverable if Tj read the Updated (or) written by Ti and Tj commits before Ti commit.

Ti	Tj
R(A)	
W(A)	
	W(A)
	R(A)
	commit
commit	