11/03/16

UNIT - 4

Transaction management

A transection is set of changes that must be made all together.

→ Transaction management
→ concurrency control.
→ Recovery

ACID Properties :-

A - Atomicity
C - consistency
I - ~~Integrity~~ / Isolation
D - Durability.

Atomicity :- A transaction is said to be atomic if the transa-ction always executes all of its action in one step or not execute any action at all.

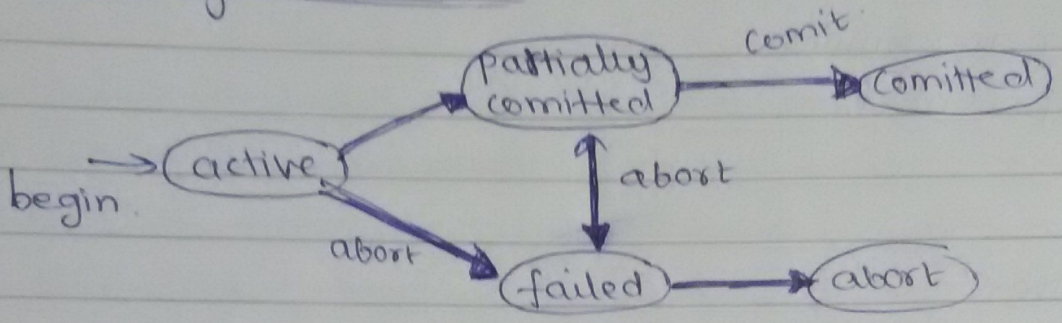
Consistency :- A transaction must preserve consistency of its database after its execution, It means that execution of transaction must ~~live~~ leave in database either in prior stable state or in new stable state.

Isolation :- It means that a data used during execution of a transaction cannot be used by the second transaction untill the first one is completed.

Durability :- Transaction are durable that means the effect of completed or comitted transaction should persist even after a system failure. It means once a transaction

commit the system must gaurentee that result of transaction will not be lost.

State of transaction:-



~~eran~~ active state:- It is the initial state, and the transaction stays in this state while it is executing.

Partially comitted:- When the final statement has been executed but comit command is not executed we said the command is partially comitted.

failed:- when the normal execution can no longer proceed we say that transaction is in failed state.

Aborted:- A transaction is aborted if the ~~trans~~ tramaction has been rolled back and ~~d~~ database is restored ~~to~~ in the state prior to starting tramaction.

Comitted:- It mean the tramaction has been succesfully comitted.

Component execution:- when more than one tramaction is executing at the same time.

20/03/16

Concurrent execution:-

It means executing more than one transaction at a time

T_1	T_2	T_3
Read(A)		
	write(A)	
		Read(A)

Schedule

A schedule is a list of actions from a set of transaction which must consist of all the instruction of a transaction We have different type of schedule:-

① Complete Schedule:- A schedule that contain either an abort or comit for each transaction is called as a complete schedule.

② Serial schedule:- Each serial schedule consist of sequence of instructions from various transaction which are executed serially.

③ Non-serial schedule:-

T_1	T_2	T_3
Read(A)		
	write(A)	
		Read(A)
Write(A)	~~Read~~	
	Read(A)	
		write(A)

A non-serial schedule in which operation from set of component transactions are interleave is called as non-~~as~~ serial schedule

4) Equivalent schedule:-
Two schedules are said to be equivalent if they produce same result on database

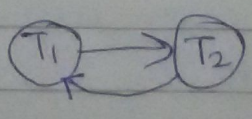
(5) Serializable schedule:- A non-serial schedule that is equivalent to some serial execution of transaction is known as serializable schedule.

Advantages of component execution:→
(1) Increased CPU utilization.
(2) Better average turn around time.
(3) Better average waiting time.
(4) Better disk utilization.

Conflicting operations :→
rule-1 → If two perform only read operation on same data item or different data item then there is no conflict
rule 2: If two transaction perform write operation on different data then also there is no conflict.
rule 3: If atleast one of the operation is write operation then these operations are called as conflicting operations.

Imp
Testing for serializability:-
→ T_i executes write(A) before T_j execute read(A).
→ T_i execute read(A) " T_j " write(A)
→ " " write(A) " T_j " "

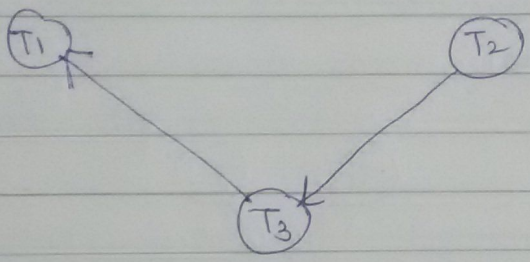
T_1	T_2
Read(A)	
	write(A)
write(A)	



cycle
non-serializable

⇒ $r_3(X)\ r_2(X)\ w_3(X)\ r_1(X)\ w_1(X)$

T_1	T_2	T_3
		read(X)
	read(X)	
		write(X)
read(X)		
write(X)		



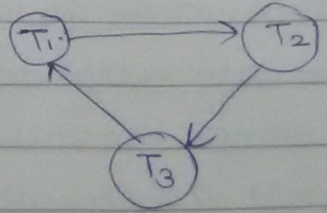
no cycle hence Serializable

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

can be written in serial order.

31/08/16 ⇒ $r_1(X)\ r_3(X)\ w_1(X)\ R_2(X)\ w_3(X)$

T_1	T_2	T_3
$r_1(X)$		
		$r_3(X)$
$w_1(X)$		
	$r_2(X)$	
		$w_3(X)$

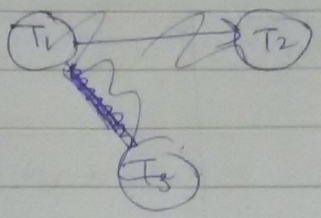


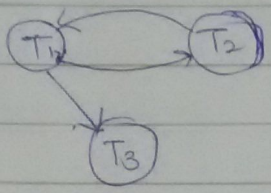
non-serializable

DS DBMS

⇒ $r_1(x)\ w_2(x)\ w_1(x)\ w_3(x)$

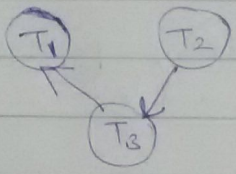
T_1	T_2	T_3
$r_1(x)$		
	$w_2(x)$	
$w_1(x)$		
		$w_3(x)$





⇒ $r_3(x)\ r_2(x)\ w_3(x)\ r_1(x)\ w_1(x)$

T_1	T_2	T_3
		r_3
	r_2	
		w_3
r_1		
w_1		



Imp

⇒ 3-problems of concurrency control

(i) lost-update problem

(ii) Dirty read problem

(iii) unrepatable Read.

(i) A lost update problm occurs when 2 transaction that access same data base item have their operations in a way that makes the value of some database item incorrect

for eg:- If 2 transaction T_1 and T_2 both read a record and then update it, the effect of Ist update will be overwritten by second transaction.

2) Dirty read problem or uncomitted data :- This problm occurs when one transaction updates a data base item and then the transaction fails for some reason and the updated data base item is accened by another transaction before it is changed back to original value.

3) Unrepeatable read :- This problem occurs when a transaction reads several values from a database while a second transaction updates some of them.

for eg :- If Trans T_1 read the record and does some other processing during which the transaction T_2 updates the record.

Concurrency control :- ~~Locking~~ Techniques

The Concurrency control property of DBMS allow many transaction to access same data base at the same time without intefering with each other to achieve independence we use lock-based protocals which is also called as locking. A lock is a variable associated with data that describes the status of the item wrt possible operations that can be applied to it.

(i) Binary lock
(ii) shared/exclusive lock (Read/write lock)

Binary lock :- A binary lock has 2 states locked or unlocked

If lock(A) = 1 locked state
lock(A) = 0 unlocked State

Rule I :- A trans say t ~~first~~ must issue the operation lock item A before any Read or write operation to be performed by t

Rule 2:- The trans T must issue unlock item A after all read and write operation on A are completed.

Advantages of Binary lock:-

→ Binary lock is easy to impliment but it is restrictive. It will not allow to trans to read to same database object even if niether of the transaction update the data base.

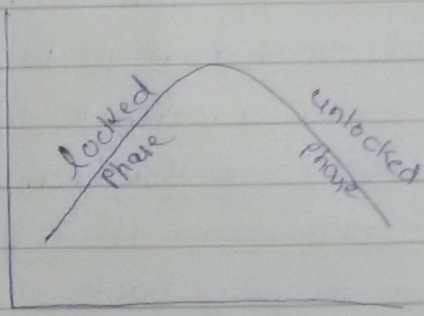
Shared/exclusive lock :-

When several trans try to access the same data item for reading purpose only then we allow the access, but if the transaction is to write an item then it must have a exclusive ~~lock~~ access. It is also called as multiple mode lock.

S_lock(A)

X_lock(A)

Two-phase locking :-



growing phase - A phase during which all the locks are requested.

Shrinking phase - when all the locks are released

Limitations of 2-phase locking:-

1) Dead lock

2) Cascading Roll-back

12/04/16

Functional dependencies :-

Let x and y be the two subset of relational schema ~~are~~ R then if given value of x ~~carries~~ There is only one value of y then y is said to be functionally dependent on x.

The Rules :-

~~1) Relative rule~~

1) Reflexive rule :- if $\beta \subseteq \alpha$ then $\alpha \to \beta$

2) Augmentation rule :- If $a \to b$ then $ax \to bx$ holds.

3) Transitive Rule :- $\alpha \to \beta$ & $\beta \to \gamma$
then $\alpha \to \gamma$

4) Union rule :- $\alpha \to \beta$ and $\alpha \to \beta\gamma$.
$\alpha \to \beta\gamma$.

5) Decomposition rule :-
if $\alpha \to \beta\gamma$ then $\alpha \to \beta$ & $\alpha \to \gamma$ holds.

6) Pseudo transitive
If $\alpha \to \beta$ & $\beta\gamma \to \delta$
then $\alpha\gamma \to \delta$.

Q→ Given the f^n dependencies
1) ~~A~~ $\to$ B
2) ABCD $\to$ E
3) EF $\to$ G
find whether ACDF $\to$ G

ABCDF → EF

Q → $A \to B$ $\quad CG \to I$

$A \to C$ $\quad B \to H$

$CG \to H$

$AG \to I$

Sol: → $A \to C$

$AG \to CG$

$AG \to I$

13/04/16

Closure on attribute set

Q → $A \to B$. $\quad A^+ = \{A, B, H, C\}$

$A \to C$ $\quad CG^+ = \{H, C, G, I$

$CG \to H$ $\quad B^+ = \{B, H\}$.

$CG \to I$

$B \to H$.

Candidate

Q → $ABCD \to E$ $\quad ABCD^+ = \{A, B, C, D, E, G, F, J, I, H\}$

$AB \to G$ $\quad AB^+ = \{A, B, G, F, H\}$

$B \to F$ $\quad B^+ = \{B, F\}$

$C \to J$ $\quad C^+ = \{C, J, I\}$

$CJ \to I$ $\quad CJ^+ = \{I, C, J\}$

$G \to H$ $\quad G^+ = \{G, H\}$

Problems with un-normalized database :-

1) Data Redundancy :- Same info is stored at various places and data redundancy leads to inconsistency & extra storage cost. This data redundancy also leads to several anamalies like insertion, deletion & updation anamolies.

→ Insertion anamaly :- We are not able to insert new tupple in a data bave.

→ deletion anaumaly :- This is the problem which causes loss of data when a particular tupple is deleted from data bau

→ updation anaumaly :- It comes when data is not updated in a database

FIRST NORMAL FORM (1NF) :-

A table is said to be in 1NF iff there are no repeating groups. Each row is atomic

Name	Marks	branch
XYZ	50, 60, 70	CSE
ABC	10, 20, 30	ECE

XYZ	50	CSE
XYZ	60	CSE
XYZ	70	CSE

Name	Branch

All attributes are dependent on a primary key.

2NF :- It is based on the concept of fully fn dependent. There should not be any partial dependency.

3NF :- A table is said to be in 3NF if it is in 2NF and there is not tranitive dependency.