Course code : **CSE2007**
Course title : **Database Management System**
Module : **5**
Topic : **3**

Characterizing Schedules based on Recoverability

Objectives

This session will give the knowledge about

- Characterizing Schedules based on Recoverability
- Recoverable schedule
- Non recoverable schedule
- Cascadeless schedule
- Cascade rollback

Transaction schedule or history

When transactions are executing concurrently in an interleaved fashion, the order of execution of operations from the various transactions forms is known as transaction schedule (or history).

A schedule (or history) S of n transactions T1, T2, …,Tn:

- It is an ordering of the operations of the transactions subject to the constraint that, for each transaction Ti that participates in S, the operations of T1 in S must appear in the same order in which they occur in T1.
- Note, however, that operations from other transactions Tj can be interleaved with the operations of Ti in S.

Notations in Transaction schedule

T1	T2
Begin	
read(a)	Begin
	read(c)
A:=a-100	
	c:=c-200
write(a)	
	write (c)
abort	
	commit

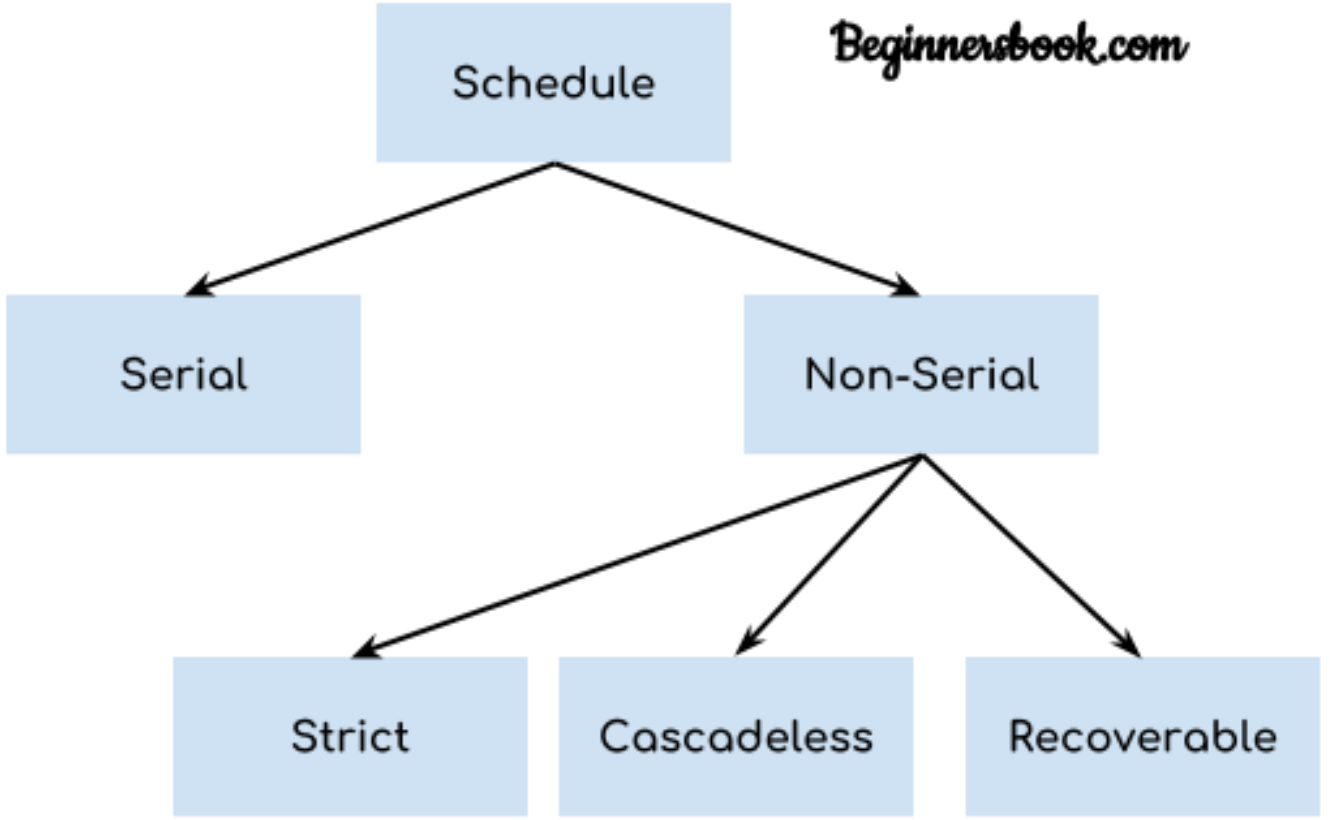
T1	T2
read(a)	
	read(c)
write(a)	
	write (c)
abort	
	commit

T1	T2
r1(a)	
	r2(c)
w1(a)	
	w2 (c)
a1	
	c2

Can be rewritten as:

Sc : r1(a);r2(c);w1(a);w2(c);a1;c2

Categories of schedules based on Recoverability



Serial schedule

In Serial schedule, a transaction is executed completely before starting the execution of another transaction.

This type of execution of transaction is also known as non-interleaved execution.

t1	t2
----	----
r1(a)	
w1(a)	
c1	
	r2(a)
	w2(b)
	c2

Recoverable schedule

In Recoverable schedule, if a transaction is reading a value which has been updated by some other transaction then this transaction can commit only after the commit of other transaction which is updating value.

Only reads are allowed before write operation on same data.

t1	t2
----	----
r1(a)	
a=a+10	
w1(a)	
	r2(b)
	b=b+20
	w2(b)
	c2
a1	

Recoverable schedule

Check whether the given schedule is recoverable or not.

Sc: r1(x); w1(x); r2(x);w2(x);r2(x);c1;c2;

t1	t2
r1(x)	
w1(x)	
	r2(x)
	w2(x)
	r2(x)
c1	
	c2

Non Recoverable schedule

Transaction T2 also should be aborted as value read by T2 becomes invalid when T1 aborts. There is a need for T2 to be aborted after it has committed. Hence the schedule is Non-Recoverable.

Check whether the given schedule is recoverable or not.

Sc: r1(x); w1(x); r2(x);w2(x);c2;c1;

t1	t2
----	----
r1(x)	
w1(x)	
	r2(x)
	w2(x)
	c2
c1	

Find the type schedule

Check whether the given schedule is recoverable or not.

S1: r1(x); w1(x); r2(x); r1(y); r2(y); w2(x); w1(y); c1; c2;

t1	t2
r1(x)	
w1(x)	
	r2(x)
r1(y)	
	r2(y)
	w2(x)
w1(y)	
c1	
	c2

Find the type schedule

Check whether the given schedule is recoverable or not.

S2: r1(x); r2(x); r1(z); r3(x); r3(y); w1(x); w3(y); r2(y); w2(z); w2(y); c1; c2; c3;

t1	t2	t3
----	----	----
r1(x)		
	r2(x)	
r1(z)		
		r3(y)
w1(x)		
		w3(y)
	r2(y)	
	w2(z)	
	w2(y)	
c1		
	c2	
		c3

Cascadeless schedule

For any transactions Ti and Tj: if Tj reads data written by Ti, then Ti commits before read operation of Tj.

Dirty Read not allowed, means reading the data written by an uncommitted transaction is not allowed. Lost Update problem may occur.

t1	t2	t3
----	----	----
r1(x)		
		r3(z)
w1(x)		
	r2(y)	
c1		
	r2(x)	
	w2(x)	
	c2	
		r3(x)
		r3(x)
		c3

Cascade Rollback schedule

If in a schedule, failure of one transaction causes several other dependent transactions to rollback or abort, then such a schedule is called as a Cascading Rollback or Cascading Abort or Cascading Schedule.

t1	t2	t3
----	----	----
r1(x)		
		r3(z)
w1(x)		
	r2(y)	
	r2(x)	
	w2(x)	
		r3(x)
		r3(x)
c1		
	c2	
		c3

Strict Schedule

A schedule in which a transaction can neither read or write an item X until the last transaction that wrote X has committed.

Simply, if there is any write(x), the next should be commit(c) before it is read by other transaction.

t1	t2
----	----
r1(a)	
	r2(b)
w1(a)	
	w2(b)
c1	
	r2(a)
	c2

Casacadless VS Strict Schedule

Cascadeless schedule: T2 can read a only after commit action from T1. But some transaction write and read. T2 can write a before commit action from T1

Strict schedule: T2 can read and write a only after commit action from T1.

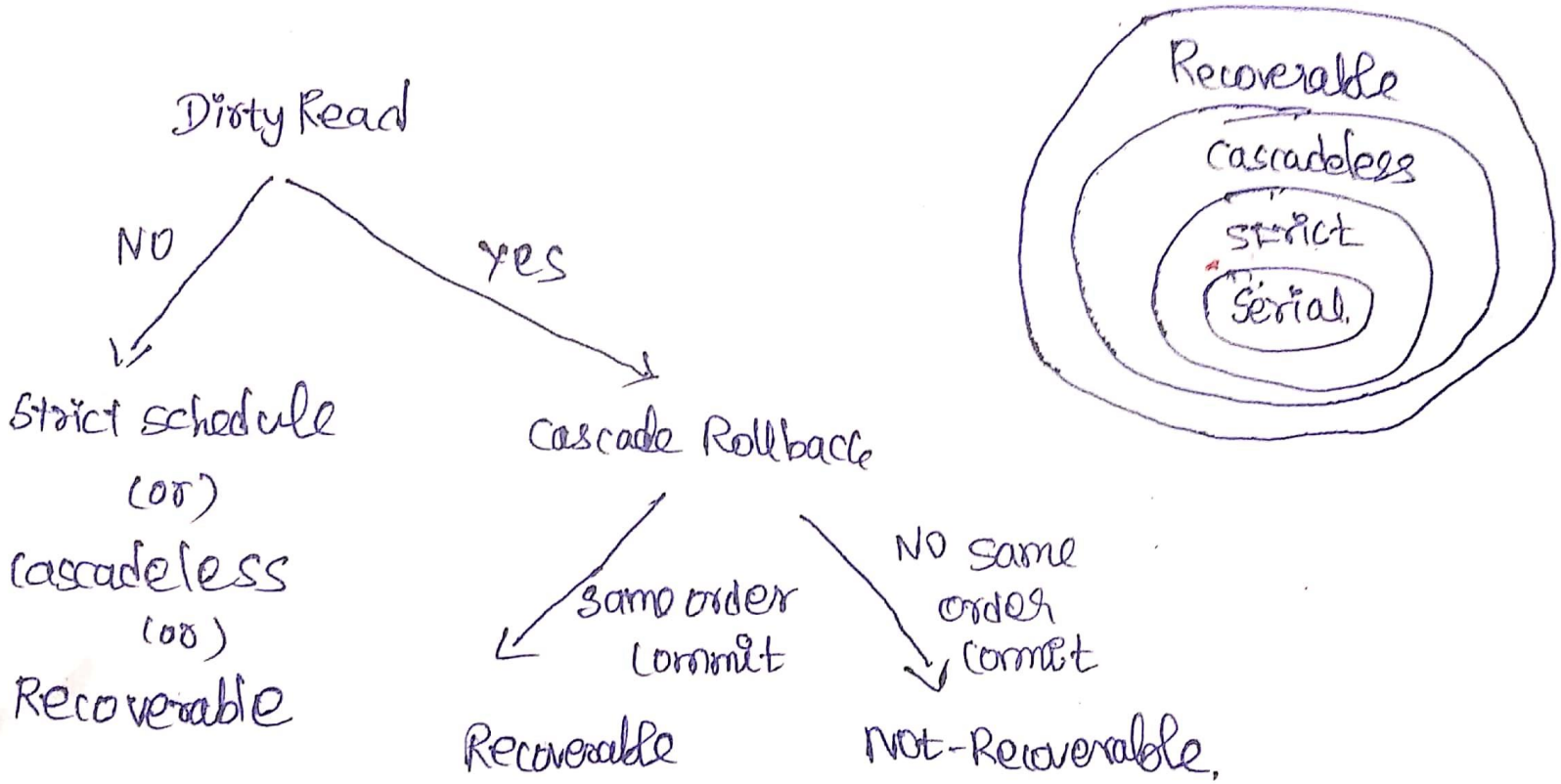
T1	T2
R(A)	
W(A)	
	W(A)
Commit	
	R(A)

Cascadless

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

Strict

Characterizing Schedules



Conclusion

- Recoverable schedules: Recovery is possible
- Non-recoverable schedules should not be permitted by the DBMS
- No committed transaction ever needs to be rolled back
- Cascading rollback may occur in some recoverable schedules
- Uncommitted transaction may need to be rolled back
- Cascadeless schedule: Avoids cascading rollback
- Strict schedule: More strict nature of parallel processing

Problems

Categorize the schedule given below:

s1: r1(x);w1(x);r2(x);r1(y);w2(x);c2;a1;

s2: r1(x);w1(x);r2(x);r1(y);w2(x);w1(y);c1;c2;

s4: r1(x); r2(x); r1(z); r3(x); r3(y); w1(x); c1; w3(y); c3; r2(y); w2(z); w2(y); c2;

s3: r1(x); r2(z); r3(x); r1(z); r2(y); r3(y); w1(x); c1; w2(z); w3(y); w2(y); c3; c2;

Problems

Consider schedules S_3, S_4, and S_5 below. Determine whether each schedule is strict, cascadeless, recoverable, or nonrecoverable. (Determine the strictest recoverability condition that each schedule satisfies.)

S_3: $r_1(X)$; $r_2(Z)$; $r_1(Z)$; $r_3(X)$; $r_3(Y)$; $w_1(X)$; c_1; $w_3(Y)$; c_3; $r_2(Y)$; $w_2(Z)$; $w_2(Y)$; c_2;

S_4: $r_1(X)$; $r_2(Z)$; $r_1(Z)$; $r_3(X)$; $r_3(Y)$; $w_1(X)$; $w_3(Y)$; $r_2(Y)$; $w_2(Z)$; $w_2(Y)$; c_1; c_2; c_3;

S_5: $r_1(X)$; $r_2(Z)$; $r_3(X)$; $r_1(Z)$; $r_2(Y)$; $r_3(Y)$; $w_1(X)$; c_1; $w_2(Z)$; $w_3(Y)$; $w_2(Y)$; c_3; c_2;

Summary

This session will give the knowledge about

- Characterizing Schedules based on Recoverability
- Recoverable schedule
- Non recoverable schedule
- Cascadeless schedule
- Cascade rollback