Database Management System (DBMS)

Lecture-41

Dharmendra Kumar

December 18, 2020

Recoverability

If a transaction T_i fails, for whatever reason, we need to undo the effect of this transaction to ensure the atomicity property of the transaction. In a system that allows concurrent execution, it is necessary also to ensure that any transaction T_j that is dependent on T_i (that is, T_j has read data written by T_i) is also aborted. To achieve this surety, we need to place restrictions on the type of schedules permitted in the system.

In the following sections, we will study two schedules which are acceptable from the viewpoint of recovery from transaction failure.

Recoverable Schedules

A schedule is said to be recoverable schedule if for each pair of transactions T_i and T_j such that T_j reads a data item previously written by T_i, the commit operation of T_i appears before the commit operation of T_j.

Example: Is the following schedule recoverable?

T_8	T_9
read(A)	
write(A)	
	read(A)
read(B)	

Solution:

Clearly, in this schedule transaction T_9 reads the value of A written by transaction T_8, but T_9 commits before T_8. Therefore this schedule is not recoverable schedule.

Transaction

A schedule is said to be cascadeless schedule if for each pair of transactions T_i and T_j such that T_j reads a data item previously written by T_i, the commit operation of T_i appears before the read operation of T_j.

Example: Is the following schedule cascadeless?

T_{10}	T_{11}	T_{12}
read(A)		
read(B)		
write(A)		
	read(A)	
	write(A)	
		read(A)

Solution:

Clearly, in this schedule transaction T_{11} reads the value of A written by transaction T_{10} and T_{10} commits before the read operation of T_{11}.

Similarly, transaction T_{12} reads the value of A written by transaction T_{11} and T_{11} commits before the read operation of T_{12}.

Therefore this schedule is cascadeless schedule.

Note: Every cascadeless schedule is also recoverable schedule. But converse need not be true.

Exercise:

1. Consider the following two transactions:

T_1: read(A);
read(B);
if A = 0 then B := B + 1;
write(B)

T_2: read(B);
read(A);
if B = 0 then A := A + 1;
write(A)

Let the consistency requirement be $A = 0 \vee B = 0$, with $A = B = 0$ the initial values.

(a) Show that every serial execution involving these two transactions preserves the consistency of the database.

(b) Show a concurrent execution of T_1 and T_2 that produces a non-serializable schedule.

(c) Is there a concurrent execution of T_1 and T_2 that produces a serializable schedule?

Transaction

2. Which of the following schedules is (conflict) serializable? For each serializable schedule, determine the equivalent serial schedules.

1. $r_1(X); r_3(X); w_1(X); r_2(X); w_3(X);$
2. $r_1(X); r_3(X); w_3(X); w_1(X); r_2(X);$
3. $r_3(X); r_2(X); w_3(X); r_1(X); w_1(X);$
4. $r_3(X); r_2(X); r_1(X); w_3(X); w_1(X);$

Transaction

3. Consider the three transactions T_1, T_2, and T_3, and the schedules 51 and 52 given below. Draw the serializability (precedence) graphs for 51 and 52' and state whether each schedule is serializable or not. If a schedule is serializable, write down the equivalent serial schedule(s).

T_1: $r_1(X); r_1(Z); w_1(X);$

T_2: $r_2(Z); r_2(Y); w_2(Z); w_2(Y);$

T_3: $r_3(X); r_3(Y); w_3(Y);$

S_1: $r_1(X); r_2(Z); r_1(Z); r_3(X); r_3(Y); w_1(X); w_3(Y); r_2(Y); w_2(Z); w_2(Y);$

S_2: $r_1(X); r_2(Z); r_3(X); r_1(Z); r_2(Y); r_3(Y); w_1(X); w_2(Z); w_3(Y); w_2(Y);$

AKTU previous year questions

1. What do you mean by Conflict Serializable Schedule?
2. What do you understand by ACID properties of transaction ? Explain in details.
3. Define Transaction and explain its properties with suitable example.
4. What is schedule? What are its types? Explain view serializable and cascadeless schedule with suitable example of each.
5. Which of the following schedules are conflicts serializable? For each serializable schedule find the equivalent serial schedule.

S_1: $r_1(x); r_3(x); w_3(x); w_1(x); r_2(x)$

S_2: $r_3(x); r_2(x); w_3(x); r_1(x); w_1(x)$

S_3: $r_1(x); r_2(x); r_3(y); w_1(x); r_2(z); r_2(y); w_2(y)$

Transaction

6. Explain I in ACID Property.
7. Define schedule.
8. What do you mean by serializability? Discuss the conflict and view serialzability with example. Discuss the testing of serializability also.
9. What do you mean by Transaction? Explain transaction property with detail and suitable example.
10. What is serializability? How it is tested?
11. State the properties of transaction.
12. What is transaction? Draw a state diagram of a transaction showing its state. Explain ACID properties of a transaction with suitable examples.
13. What are the schedules? What are the differences between conflict serialzability and view serialzability ? Explain with suitable examples what are cascadeless and recoverable schedules?

Thank You.