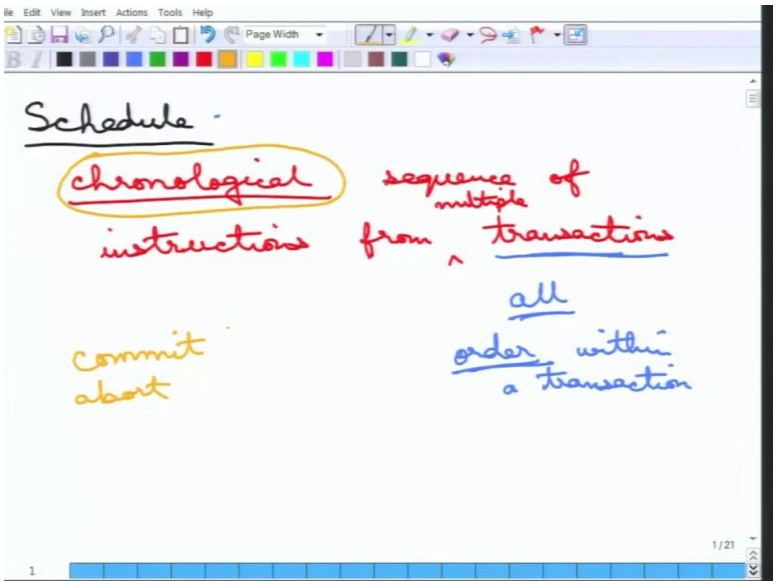
Fundamentals of Database Systems
Prof. Arnab Bhattacharya
Department of Computer Science and Engineering
Indian Institute of Technology, Kanpur

Lecture - 33
Schedules: Introduction

Welcome, we will continue with our study on the database transactions. We have seen the basics of what the database transaction is and what are its properties and we have also studied the recovery management systems. So, today we will start on something which is called Schedules.

(Refer Slide Time: 00:29)



So, schedule, so what is first of all a schedule, a schedule is essentially a chronological sequence of instructions from multiple transactions. So, the important point it is the chronological sequence of instructions from transactions, so that the number of transactions may be one or more and generally, so it may be multiple transactions. Now, the important point is that, this is chronological; that means all the instructions from multiple transactions are written one after another.

So, it may be saying that a schedule may be in the form of transaction one writes to A, then transaction to writes to B, then transaction T reads from A and so on and so forth. It is a chronological sequence, so it is a time order sequence. Now, there are certain rules of

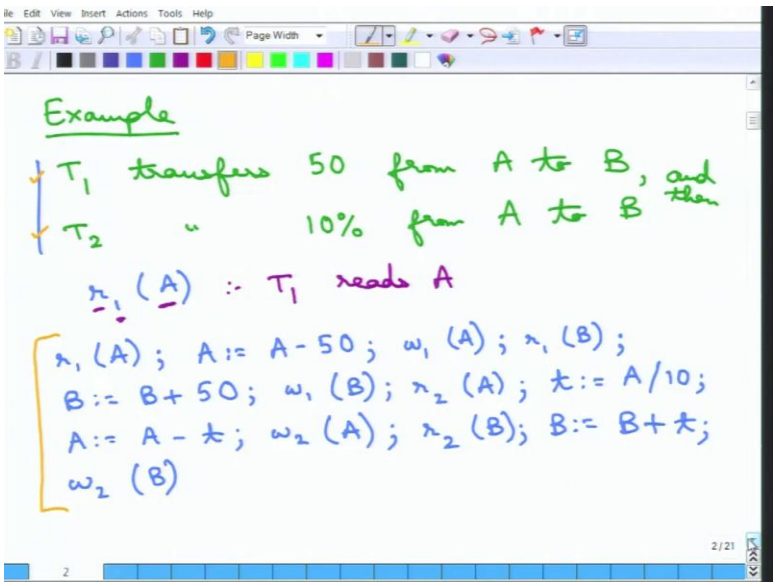
what can constitute schedules, if a transaction appear, if one instruction of transaction appears in a schedule, all the instructions must appear from the transactions.

So, all instructions of the transaction must appear to the schedule or none of the instructions of the transaction appear, so that is the first property. The other is the order of instructions with in a transaction, so order with in a transaction must be maintained. So, if the particular transaction let us concentrate the transaction 1, if transaction 1, first writes to A and then, reads A.

Then, in that schedule that contains transaction 1, it must be the transaction 1 first writes to A and then, reads from A, it cannot be swap. So, the order of instructions within a transaction must be respected in the schedule that contains the transaction. Then, there is a same rule as earlier, so if the transaction commits, if a transaction finishes successfully, then it is writes the commit.

So, that is also part of the schedule or otherwise, you say abort and sometimes in some schedules, when we study the schedule committed abort may be committed, because it may not be needed for that particular study.

(Refer Slide Time: 02:53)



But anyway, let us go to an example to understand, what exactly is the schedule? So, here is an example. Suppose there are two transactions. So, T 1 transaction T 1transfers, let us say some 50 rupees from account of A to that of B and then, transaction T 2

transfer 10 percentage of money from A to B. So, these are the two transactions and both of them can be represented in a single schedule and this is how, it can be represented.

So, the first one transaction T 1 transfer 50 from A to B, what can be done, it is a read 1 of A. Now, please note the notation here read 1 of A, this is a notation that will be using again and again for schedules. So, that first thing is that, this is the read operation; that is why, this is an r, then this is 1; that means, this is what transaction T 1 is doing. So, this read is done by transaction T 1 and A is being rate. So, these essentially means that the form meaning is that transaction T 1 reads A. So, that is the found; that is the notation.

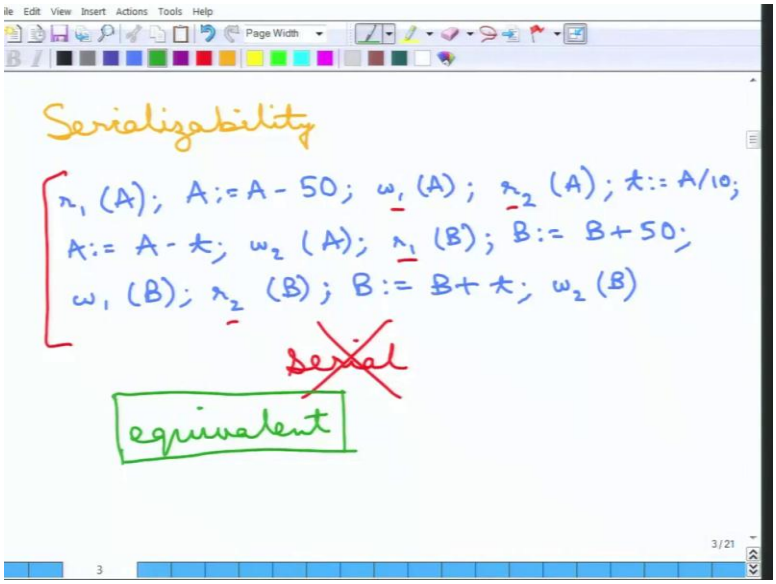
And if it is read it is r, otherwise read it W, which is for write, anyway coming back the point of this schedule, what it can be return in the following manner is that, r 1 A and then, we will write it sequentially. So, in a chronological order and we will write it from left to right as a natural reading. So, A is equal to A minus 50, then it is writes to A, then what happens it reads B, it is still its everything is still going on for transactions 1.

Then, plus 50, then there is transaction 1, then writes to B, this is complete transaction 1 and if you remember this first transaction T 1 transfer and then, this transactions happens. So, this will then the write at it is the following manner read now transaction two start reading and there is a temporary variable that is make use of. Let us say which is 10 present of A, which is essentially A by 10.

Then, A is equal to A minus T, then write W 2 of A, then it is similar to what the first transaction has been doing. So, this only it is being transfer B, this entire thing is the schedule, which is consist of two transactions T 1 and T 2, so T 1 and then T 2. So, now, this is the one schedule, now there may be another schedule; that is return in the following manner.

So, there may be little bit of swiping order transposition of the instructions. Now, one important thing is that, this transaction is called a serial schedule. Why it is called as serial, because you see transaction one first complete all it is operations, this finishes everything here and then, transaction 2 starts. So, this is why this is called a serial. Now, also known that, we have not mentioned here the commit and abort session, because what we are going to study here is about serializability.

(Refer Slide Time: 06:33)



So, the study that will be doing, the first thing that will be studying about this is called a serializability and that is that they will not require as to note whether the transaction has aborted or committed. Now, coming back to this example, so this is the serial schedule. Now, if quantise the entire schedule and do starten transposition that have been saying, then it an it an equivalent schedule can be produced. Now, this is the equivalent schedule, so let me just right down the equivalent schedule.

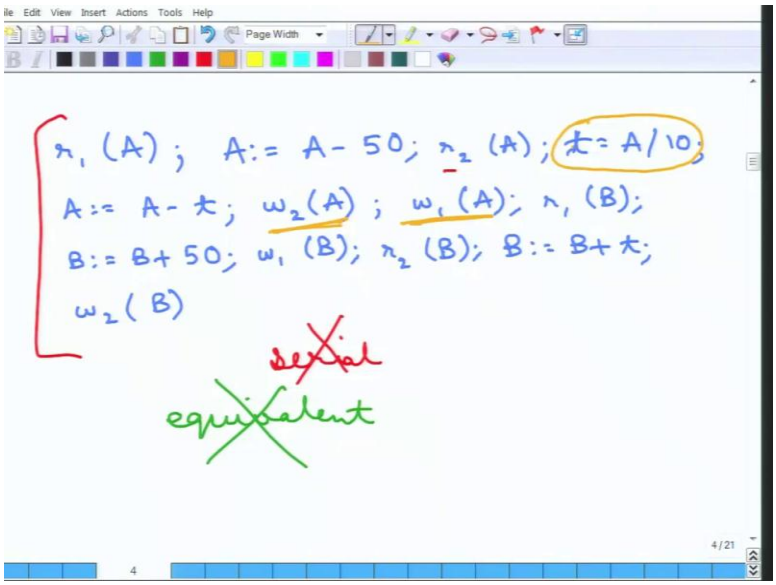
Note, very importantly, what I am doing here is before transaction 1, have transaction return to the B etcetera, transaction to start reading A, which is fine, it should not be any problem. And we will argue our whether there is the problem or not etcetera, but this is essentially, what is being done. So, now, one can do this, so this is not serial, because the instructions of 1 and 2 are inter leave.

So, 1, then 2, then and again 1, then and again this 2, etcetera, so this is not serial. So, this is not serial; however, as one can very clearly see from the event in the transaction that, this is equivalent the serial schedule that we saw earlier. So, this is equivalent, this is the very important term, why is it call equivalent is the following idea is that equivalent, because if instead of the serial schedule, if this serial schedule was run.

So, the transactions are inter leaved and transaction 1 and 2 had taking place consonantly, the effect on the database finally, assuming both the transactions of course, we are assuming both the transactions finish successfully. Then, the there is these

schedule and the serial schedule, the effect of both of the schedule is exactly the same, no matter, what the state of the databases. Thus, if they start from the same state of the database, then the end of in the same state of the database. So, that is why, these are called equivalent.

(Refer Slide Time: 09:19)



Now, we will study on this a little bit more, but before that here is another schedule and which is the following is that. So, we saw one serial schedule and we saw one schedule, which is equivalent to it. Now, let us see another schedule, which is the following, this is another schedule. This is again not serial as we can very clearly see that this is not serial because again this is inter leave, so r 2, then again r 1, then r 2 and so on and so forth, so this is not serial.

But, also this is not equivalent to the serial schedule that we earlier solve, this is not equivalent, why this is not equivalent, a little bit of examination is show that, this is not equivalent, because look at what T is here. So, this is what T is reading is this temporary value, this is the 10 percent before transaction 1 has return to the A. So, this is the 10 percent of this originally.

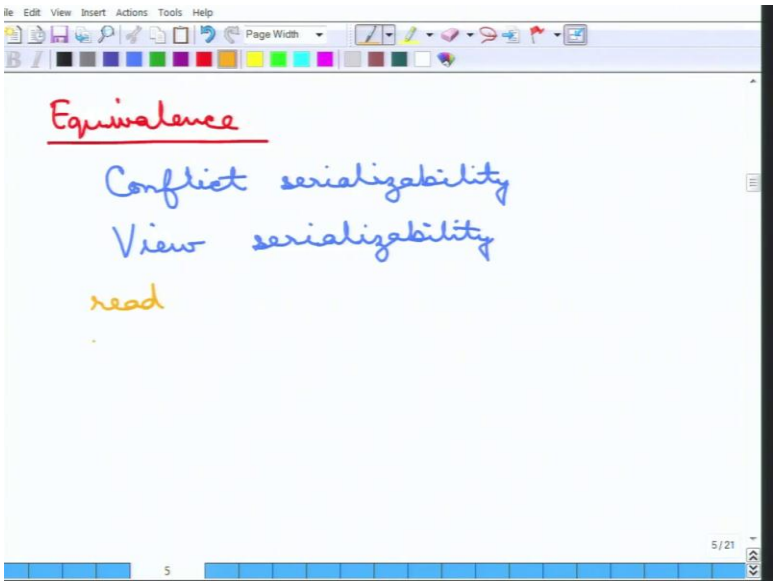
And so this is not correct because this T, when it is reads, it is not going to correct, this W 2 A is first return and then W 1 A is return. So, this is not equivalent this will not have the same effect on the database as the serial schedule. So, essentially the study of

serializability of schedule is the following is that, suppose as schedule is given with two are more transactions and suppose, there is a serial order of the transactions.

So, serial order meaning, so suppose there is a two transactions either T 1 all the operation of T 1 happen and T 2 starts and complete all it is operations or T 2 finishes all it is operations and then, T 1 starts. So, these are the two serial schedules. Now, an another schedule is said to be equivalent to the serial schedule, if the effect of that schedule is the same as the serial schedule as one of the serial schedule.

So, that is what the equivalent and we will try to formally show how equivalences can be shown So, this is the study of serializability that we solve. So, this is essentially just once more, this is this study is called the serializability.

(Refer Slide Time: 12:12)



So, there are different fonts of this equivalence, so the equivalence. So, essentially we are trying to say when have to study equivalence. So, for that there are two motions of equivalence first is that conflict serializability and the second is the view serializability. So, these are the two motions of equivalence serializability that we will study. Now, what essentially is this, so we will see this?

But, there are some assumptions, first of all the transaction ignores every other operations other than read and write. So, read and write are the only two operations that

are consider by a transaction. For that purposes are equivalent, note that, this is only for the purposes are equivalent serializability, whatever you want.