CS 240: Database Transactions Transcript

This video shows a split screen of Professor Wilkerson on the right and a PowerPoint screen on the left. Visual descriptions are not needed.

[00:00:00] **JEROD WILKERSON:** The next thing we need to learn is how to do database transactions.

[00:00:03] By default, each SQL statement will be executed in its own transaction, or in other words, it'll be executed independently.

[00:00:11] It will execute, it will either succeed or fail, and then we'll do the next statement.

[00:00:16] But sometimes, we need to consider statements as a whole.

[00:00:20] Let me give you an example of that.

[00:00:22] One example would be if you go to the ATM to transfer money.

[00:00:27] Let's say that you're going to make a purchase and you're going to write a check.

[00:00:31] I know nobody writes checks anymore, but let's just pretend that you are.

[00:00:34] You go to your ATM and you have some money in your savings account and you transfer it to your checking account so you can write a check.

[00:00:41] We're going to transfer a $100.

[00:00:43] If we really think about that, if we have this represented in some accounts we could use any of the ways that I showed you for representing accounts.

[00:00:52] But we really need to do two database operations, we need to do two updates.

[00:00:57] We need to do an update to our savings account where we reduce the balance by $100, and we need to do an update to our checking account where we increase the balance by $100.

[00:01:07] Neither of those should happen by itself.

[00:01:10] Let's say that we go to the ATM and we're going to transfer the money and we start the transaction and it does the update to the checking account first.

[00:01:20] We say we want to transfer $100.

[00:01:22] It updates our checking account and puts an extra $100 in there and then the ATM crashes.

[00:01:28] Now we're really happy because we just gained $100 magically somehow.

[00:01:32] The problem with that is the bank will find out and they will hunt you down.

[00:01:36] They'll eventually look and see that you've got an extra $100 and then hopefully, you're not going to feel good about that anyway.

[00:01:42] and so you'll go to the bank and you'll tell him, "Hey, I've got this extra $100, I want to give it back." The reverse could happen where if the statements happen in the opposite order, if we decrement or reduce the savings account first and the system crashes, now I've just lost a $100, so you're not very happy about that.

[00:02:00] Let's say that you want to make the purchase.

[00:02:03] It's Friday afternoon or it's a banking holiday, and so you try to make the transfer, it crashes in the middle, and you notice that now I have $100 less in my savings account and I don't have the money in my checking account.

[00:02:16] You want to go to the bank and fix that.

[00:02:18] When the bank is finally open again, you go talk to a teller and they say, “We’ll look into that.

[00:02:23] Give us three days and you’ll have your money.” Obviously, that’s not very convenient.

[00:02:27] What we really want is we want a situation where if there’s a crash, all the statements get undone.

[00:02:35] We don’t leave these two operations in a partially completed state.

[00:02:40] That’s exactly what we have with database transactions.

[00:02:43] We use database transactions if we have multiple statements that either need to all succeed together or all fail together.

[00:02:51] If you think about that, if you go to the ATM and it’s starting to process your transfer, and it does in one of the statements, and then it crashes, we want it to just undo the one that it did.

[00:03:01] We want it to say, “Well, I didn’t really get to finish the transfer, therefore, no transfer happened.” That’s what we would want and that’s what you get with transaction.

[00:03:07] It’s really simple.

[00:03:10] In SQL, you just say “BEGIN TRANSACTION;”, you submit that as an SQL statement.

[00:03:15] Then you specify all the statements that should be part of that transaction and then you either “COMMIT TRANSACTION;”.

[00:03:22] If everything was good, you commit it.

[00:03:23] If not, you “ROLLBACK TRANSACTION;”.

[00:03:25] When you learn how to do this in JDBC and your Java program, you'll use catch blocks for that and you will make it so you roll back if you've got some error, and you'll commit if you didn't.

[00:03:38] That's a database transaction.

[00:03:40] They're really important in SQL and they're really easy to do with SQL.