## Chapter 6.2 Undo Logging

*Definition:*

Logging is the file that consists of Logging Records, each logging records the situation that records about some things that Transaction has done.

*If logging records has appeared in the Non-Volatile storage, then after crash happens, we can use them to recover Database System into Consistent Status. Our first Logging type is Undo Logging, it can be used to Undo Transaction that happens before crash to repair database status.*

*Other Contents:*

* Introduce the basic thinking of Logging Records, including Commit and its influence on Database Status and Logging.
* Create Logging into the main memory and flush logging to copy the logging to Disk.
* Check undo Logging, learn to use it to recover from the crash.
* In order to avoid checking all logging, so introduce ‘Checkpoint’ and it can be used to throw the old logging part.

### Chapter 6.2.1 Logging Record

Here, logging file is only seen as file which only can be opened by using additional method. However when transaction executes, the Logging Management is responsible for the each event in the Logging.

* *Each time when the logging block is filled with logging records, and each logging record is corresponding to one of these event.*
* *Initially, the logging block is created in Main Memory, and just like other blocks that assigned by Buffer Management, when there is any chance, the logging block would be written back to Non - Volatile Memory.*

There have several types of Logging Records, including:

1. *<START T>*: This recording is used to record the *start of Transaction T*.
2. *<COMMIT T>*: *Transaction T has been executed successfully*, and *there would have no more changes on Database Element*. Any updates that T to the Database would reflect to Disk. However, since we can not control when Buffer Management would write back the block to Main Memory. However we see record <COMMIT T> logging record, we still can not make sure whether all updates have been reflected on the Disk. If we insist to make all update on Disk, then this requirement can be finished by Logging Management, this should be the *task of Buffer Management*.
3. *<ABORT T>*: *Transaction T has not been finished successfully.* Since if Transaction T has been aborted, then any updates from the transaction can not be copied to the Disk. The Transaction Management has the responsibility to ensure that such update can not appear on the Disk, or any updates should be eliminated from Disk.

*For Undo Logging, the only Logging type of record is update Record, the format of Logging Record is just as <T, X, v>.*

* *Meaning: Transaction T change the database element X, and the original value of X is v.*
* *The place where this change happens is normally on the Main Memory but not on the Disk.*
* *This Transaction is only used to record OLD Value but not NEW Value.*

Just as we see before, If we want to use Undo Logging records to recover the Database System, then we just need to reuse OLD Value on the disk to eliminate the influence of this Transaction.

### Chapter 6.2.2 Rule of Logging Record

### Chapter 6.2.3 Using Undo Logging to Recover

### Chapter 6.2.4 Checkpoint

### Chapter 6.2.5 Non - Static Checkpoint