Database System (SW5)

11. Recovery

Tiantian Liu
Department of Computer Science
Aalborg University

Fall 2025

Motivation and Learning Goals

- Motivation
  - We want to preserve consistency and availability even in the case of failures.
- Learning Goals
  - Understanding basic logging algorithms
  - Understanding the importance of atomicity and durability

Recovery

- "Problems" with transactions
  - Atomicity: Transactions may abort (rollback)
  - Durability: What if a DBMS crashes?
- The DBMS ensures that a transaction
  - Either completes and has a permanent result (committed transaction)
  - Or has no effect at all on the database (aborted transaction).
- The role of the recovery component is to ensure atomicity and durability of transactions in the presence of system failures.

Failure Classification

- Transaction failure (failure of a not yet committed transaction)
  - Undo the changes of the transaction
- System crash (failure with main memory loss)
  - Changes of committed transactions must be preserved
  - Changes of all non-committed transactions need to be undone
- Disk failure (failure with hard disk loss)
  - Recovery based on archives/dumps

Data Access

- **Physical blocks** are those blocks residing on the disk.
- **Buffer blocks** are the blocks residing temporarily in main memory.
- Block movements between disk and main memory are initiated through the following two operations:
  - Input (B) transfers the physical block B to main memory.
  - Output (B) transfers the buffer block B to the disk, and replaces the appropriate physical block there.
- We assume, for simplicity, that each data item fits in, and is stored inside, a single block.

The WAL rule for log-based recovery

- WAL (Write Ahead Logging)
  - Before a transaction enters the commit state, "all its" log entries have to be written to stable storage, incl. the commit log entry
  - Before a modified page (or block) in main memory can be written to the database (non-volatile storage), "all its" log entries have to be written to stable storage

Structure of a log entry (log record)

- [TID, DID, old, new]
  - TID identifier of the transaction that caused the update
  - DID data item identifier: location on disk (page, block)
  - old value of the data item before the update
  - new value of the data item after the update
- Additional entries
  - start Transaction TID has started [TID start]
  - commit Transaction TID has committed [TID commit]
  - abort Transaction TID has aborted [TID abort]

Log entry example

T_1	T_2
begin	
read(B, b)	
b ← b + 100	
write(B, b)	
commit	
	begin
	read(D, d)
	d ← d - 10
	write(D, d)
	read(E, e)
	e ← e - 20
	write(E, e)
	commit

[TID, DID, old, new]

[T1 start]

[T1, B, 300, 400]

[T1 commit]

[T2 start]

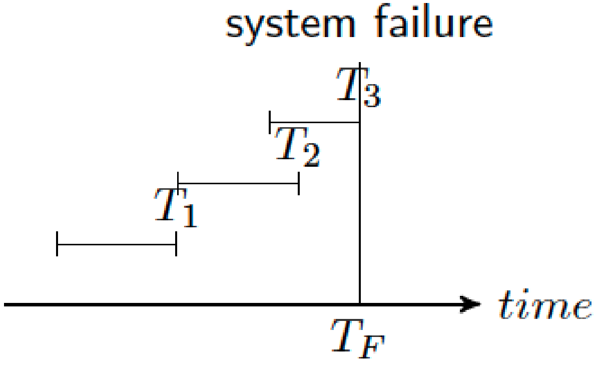
[T2, D, 530, 520]

[T2, E, 70, 50]

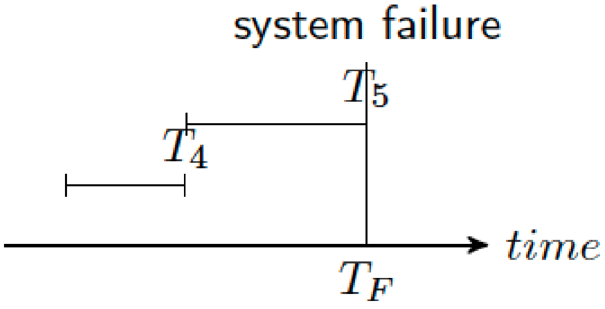
[T2 commit]

Log-based recovery

- Operations to recover from failures
  - Redo: perform the changes to the database again
  - Undo: restore database to state prior to execution
- To recover from a failure
  - Redo results for committed transactions
  - Undo changes of transactions that did not commit



Redo T_1 and T_2
Undo T_3



Redo T_4
Undo T_5

The phases of recovery

- Redo (repeat history)
  - Forward scan through the log
  - Repeat all updates in the same order as in the log file
  - Determine "undo" transactions
    - [TID start] add TID to the "undo list"
    - [TID abort] or [TID commit] remove TID from the "undo list"
- Undo (rollback) all transactions in the "undo list"
  - Backward scan through the log
  - Undo all updates of transactions in the "undo list": create a compensating log record
  - For a [TID start] record of a transaction TID in the "undo list", add a [TID abort] record to the log file, remove TID from the "undo list"
  - Stop, when "undo list" is empty

Compensation log records

- [TID, DID, old]
    - Created to undo (compensate) the changes of [TID, DID, old, new]
    - Redo-only log record
    - Can also be used to rollback a transaction during normal operation

Example

AALBORG UNIVERSITY

Database

A	100
B	200
C	50
D	60
E	120

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

Example

Phase 1 (redo)

Database

A	100
B	200
C	50
D	60
E	120

undo list
{}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

Example

Phase 1 (redo)

Database

A	100
B	200
C	50
D	60
E	120

undo list

{T1}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

Example

Phase 1 (redo)

Database

A	100
B	200
C	50
D	60
E	120

undo list
{T1}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

Example

Phase 1 (redo)

Database

A	100
B	400
C	50
D	60
E	120

undo list
{T1}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

Example

Phase 1 (redo)

Database

A	100
B	400
C	50
D	60
E	120

undo list
{T1}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

Example

Phase 1 (redo)

Database

A	100
B	400
C	100
D	60
E	120

undo list
{T1}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

Example

Phase 1 (redo)

Database

A	100
B	400
C	100
D	60
E	120

undo list
{T1}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

Example

Phase 1 (redo)

Database

A	100
B	400
C	100
D	60
E	120

undo list

{T1， T2}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

Example

Phase 1 (redo)

Database

A	100
B	400
C	100
D	60
E	120

undo list
{T1， T2}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

Example

Phase 1 (redo)

Database

A	100
B	400
C	100
D	60
E	480

undo list

{T1， T2}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

Example

Phase 1 (redo)

Database

A	100
B	400
C	100
D	60
E	480

undo list
{T1， T2}

Log records

[T1 start]
[T1, B, 200, 400]
[T1, C, 50, 100]
[T2 start]
[T2, E, 120, 480]
[T1, A, 100, 320]
[T1 commit]
[T2, A, 320, 520]
[T2, D, 60, 530]

Example

Phase 1 (redo)

Database

A	320
B	400
C	100
D	60
E	480

undo list
{T1， T2}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

Example

Phase 1 (redo)

Database

A	320
B	400
C	100
D	60
E	480

undo list

{T1，T2}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

Example

Phase 1 (redo)

Database

A	320
B	400
C	100
D	60
E	480

undo list
{T2}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

Example

Phase 1 (redo)

Database

A	320
B	400
C	100
D	60
E	480

undo list
{T2}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

Example

Phase 1 (redo)

Database

A	520
B	400
C	100
D	60
E	480

undo list
{T2}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

Example

Phase 1 (redo)

Database

A	520
B	400
C	100
D	60
E	480

undo list
{T2}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

Example

Phase 1 (redo)

Database

A	520
B	400
C	100
D	530
E	480

undo list
{T2}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

Example

Phase 2 (undo)

Database

A	520
B	400
C	100
D	530
E	480

undo list
{T2}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

Example

Phase 2 (undo)

Database

A	520
B	400
C	100
D	530
E	480

undo list
{T2}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

Example

Phase 2 (undo)

Database

A	520
B	400
C	100
D	60
E	480

undo list
{T2}

Log records

[T1 start]
[T1, B, 200, 400]
[T1, C, 50, 100]
[T2 start]
[T2, E, 120, 480]
[T1, A, 100, 320]
[T1 commit]
[T2, A, 320, 520]
[T2, D, 60, 530]
[T2, D, 60]

Example

Phase 2 (undo)

Database

A	520
B	400
C	100
D	60
E	480

undo list
{T2}

Log records

[T1 start]
[T1, B, 200, 400]
[T1, C, 50, 100]
[T2 start]
[T2, E, 120, 480]
[T1, A, 100, 320]
[T1 commit]
[T2, A, 320, 520]
[T2, D, 60, 530]
[T2, D, 60]

Example

Phase 2 (undo)

Database

A	320
B	400
C	100
D	60
E	480

undo list
{T2}

Log records

[T1 start]
[T1, B, 200, 400]
[T1, C, 50, 100]
[T2 start]
[T2, E, 120, 480]
[T1, A, 100, 320]
[T1 commit]
[T2, A, 320, 520]
[T2, D, 60, 530]
[T2, D, 60]
[T2, A, 320]

Example

Phase 2 (undo)

Database

A	320
B	400
C	100
D	60
E	480

undo list
{T2}

Log records

[T1 start]
[T1, B, 200, 400]
[T1, C, 50, 100]
[T2 start]
[T2, E, 120, 480]
[T1, A, 100, 320]
[T1 commit]
[T2, A, 320, 520]
[T2, D, 60, 530]
[T2, D, 60]
[T2, A, 320]

Example

Phase 2 (undo)

Database

A	320
B	400
C	100
D	60
E	480

undo list
{T2}

Log records

[T1 start]
[T1, B, 200, 400]
[T1, C, 50, 100]
[T2 start]
[T2, E, 120, 480]
[T1, A, 100, 320]
[T1 commit]
[T2, A, 320, 520]
[T2, D, 60, 530]
[T2, D, 60]
[T2, A, 320]

Example

Phase 2 (undo)

Database

A	320
B	400
C	100
D	60
E	480

undo list

{T2}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

[T2, D, 60]

[T2, A, 320]

Example

Phase 2 (undo)

Database

A	320
B	400
C	100
D	60
E	120

undo list

{T2}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

[T2, D, 60]

[T2, A, 320]

[T2, E, 120]

Example

Phase 2 (undo)

Database

A	320
B	400
C	100
D	60
E	120

undo list

{T2}

Log records

[T1 start]

[T1, B, 200, 400]

[T1, C, 50, 100]

[T2 start]

[T2, E, 120, 480]

[T1, A, 100, 320]

[T1 commit]

[T2, A, 320, 520]

[T2, D, 60, 530]

[T2, D, 60]

[T2, A, 320]

[T2, E, 120]

Example

Phase 2 (undo)

Database

A	320
B	400
C	100
D	60
E	120

undo list
{}

Log records

[T1 start]
[T1, B, 200, 400]
[T1, C, 50, 100]
[T2 start]
[T2, E, 120, 480]
[T1, A, 100, 320]
[T1 commit]
[T2, A, 320, 520]
[T2, D, 60, 530]
[T2, D, 60]
[T2, A, 320]
[T2, E, 120]
[T2, abort]

Summary

- Recovery: ensuring atomicity and durability despite failures and crashes
- WAL rule
- Log-based recovery
  - All changes need to be written into the log file
  - A transaction commits when the commit entry in the log is written