Database Recovery
Project #3 is due Sun Nov 14\textsuperscript{nd} @ 11:59pm.  
Additional office hour on Saturday.

Homework #4 is due Wed Nov 10\textsuperscript{th} @ 11:59pm.
UPCOMING DATABASE TALK

Fluree - Cloud-Native Ledger Graph Database
→ Mon Nov 15\textsuperscript{th} @ 4:30pm ET
Recovery algorithms are techniques to ensure database consistency, transaction atomicity, and durability despite failures.

Recovery algorithms have two parts:
→ Actions during normal txn processing to ensure that the DBMS can recover from a failure.
→ Actions after a failure to recover the database to a state that ensures atomicity, consistency, and durability.
CRASH RECOVERY

Recovery algorithms are techniques to ensure database consistency, transaction atomicity, and durability despite failures.

Recovery algorithms have two parts:

→ Actions during normal txn processing to ensure that the DBMS can recover from a failure.
→ Actions after a failure to recover the database to a state that ensures atomicity, consistency, and durability.
CHECKPOINTS

The WAL will grow forever.

After a crash, the DBMS must replay the entire log, which will take a long time.

The DBMS periodically takes a checkpoint where it flushes all buffers out to disk.
CHECKPOINTS

Output onto stable storage all log records currently residing in main memory.

Output to the disk all modified blocks.

Write a `<CHECKPOINT>` entry to the log and flush to stable storage.
CHECKPOINTS

WAL

<T1 BEGIN>
<T1, A, 1, 2>
<T1 COMMIT>
<T2 BEGIN>
<T2, A, 2, 3>
<T3 BEGIN>
<CHECKPOINT>
<T2 COMMIT>
<T3, A, 3, 4>

⋯

CRASH!
CHECKPOINTS

WAL

<T1 BEGIN>
<T1, A, 1, 2>
<T1 COMMIT>
<T2 BEGIN>
<T2, A, 2, 3>
<T3 BEGIN>
<CHECKPOINT>
<T2 COMMIT>
<T3, A, 3, 4>

CRASH!
CHECKPOINTS

Any txn that committed before the checkpoint is ignored ($T_1$).
CHECKPOINTS

Any txn that committed before the checkpoint is ignored ($T_1$).

$T_2 + T_3$ did not commit before the last checkpoint.
CHECKPOINTS

Anytxnthatcommittedbeforethecheckpointisignored(T₁).

T₂ + T₃ did not commit before the last checkpoint.
→ Need to redo T₂ because it committed after checkpoint.
→ Need to undo T₃ because it did not commit before the crash.
CHECKPOINTS – CHALLENGES

The DBMS must stall txns when it takes a checkpoint to ensure a consistent snapshot.

Scanning the log to find uncommitted txns can take a long time.

Not obvious how often the DBMS should take a checkpoint...
Checkpoints – Frequency

Checkpointing too often causes the runtime performance to degrade.
→ System spends too much time flushing buffers.

But waiting a long time is just as bad:
→ The checkpoint will be large and slow.
→ Makes recovery time much longer.
Recovery algorithms are techniques to ensure database consistency, transaction atomicity, and durability despite failures.

Recovery algorithms have two parts:
→ Actions during normal txn processing to ensure that the DBMS can recover from a failure.
→ Actions after a failure to recover the database to a state that ensures atomicity, consistency, and durability.
ARIES

Algorithms for Recovery and Isolation Exploiting Semantics

Developed at IBM Research in early 1990s for the DB2 DBMS.

Not all systems implement ARIES exactly as defined in this paper but they're close enough.
ARIES – MAIN IDEAS

Write-Ahead Logging:
→ Any change is recorded in log on stable storage before the database change is written to disk.
→ Must use STEAL + NO-FORCE buffer pool policies.

Repeating History During Redo:
→ On restart, retrace actions and restore database to exact state before crash.

Logging Changes During Undo:
→ Record undo actions to log to ensure action is not repeated in the event of repeated failures.
TODAY’S AGENDA

Log Sequence Numbers
Normal Commit & Abort Operations
Fuzzy Checkpointing
Recovery Algorithm
WAL RECORDS

We need to extend our log record format from last class to include additional info.

Every log record now includes a globally unique log sequence number (LSN).

Various components in the system keep track of LSNs that pertain to them...
# Log Sequence Numbers

<table>
<thead>
<tr>
<th>Name</th>
<th>Where</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>flushedLSN</td>
<td>Memory</td>
<td>Last LSN in log on disk</td>
</tr>
<tr>
<td>pageLSN</td>
<td>page_x</td>
<td>Newest update to page_x</td>
</tr>
<tr>
<td>recLSN</td>
<td>page_x</td>
<td>Oldest update to page_x since it was last flushed</td>
</tr>
<tr>
<td>lastLSN</td>
<td>T_i</td>
<td>Latest record of txn T_i</td>
</tr>
<tr>
<td>MasterRecord</td>
<td>Disk</td>
<td>LSN of latest checkpoint</td>
</tr>
</tbody>
</table>
WRITING LOG RECORDS

Each data page contains a pageLSN.
→ The LSN of the most recent update to that page.

System keeps track of flushedLSN.
→ The max LSN flushed so far.

Before page \( x \) can be written to disk, we must flush log at least to the point where:
→ \( \text{pageLSN}_x \leq \text{flushedLSN} \)
WAL (Tail)
017: <T₅ BEGIN>
018: <T₅, A, 9, 8>
019: <T₅, B, 5, 1>
020: <T₅ COMMIT>

Buffer Pool

current flushedLSN

A=9 B=5 C=2

WAL
001: <T₁, BEGIN>
002: <T₁, A, 1, 2>
003: <T₁, COMMIT>
004: <T₂, BEGIN>
005: <T₂, A, 2, 3>
006: <T₃, BEGIN>
007: <CHECKPOINT>
008: <T₂, COMMIT>
009: <T₃, A, 3, 4>
010: <T₄, BEGIN>
011: <T₄, X, 5, 6>
012: <T₄, Y, 9, 7>
013: <T₃, B, 4, 2>
014: <T₃, COMMIT>
015: <T₄, B, 2, 3>
016: <T₄, C, 1, 2>

Database

current pageLSN recLSN

A=9 B=5 C=2

MasterRecord

COMMEND LOG RECORDS
WRITING LOG RECORDS

Log Sequence Numbers

WAL (Tail)

<table>
<thead>
<tr>
<th>No.</th>
<th>Record</th>
</tr>
</thead>
<tbody>
<tr>
<td>017</td>
<td>&lt;T_5 BEGIN&gt;</td>
</tr>
<tr>
<td>018</td>
<td>&lt;T_5, A, 9, 8&gt;</td>
</tr>
<tr>
<td>019</td>
<td>&lt;T_5, B, 5, 1&gt;</td>
</tr>
<tr>
<td>020</td>
<td>&lt;T_5 COMMIT&gt;</td>
</tr>
</tbody>
</table>

Buffer Pool

<table>
<thead>
<tr>
<th>Page LSN</th>
<th>Record LSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>A=9</td>
<td>B=5</td>
</tr>
<tr>
<td></td>
<td>C=2</td>
</tr>
</tbody>
</table>

flushedLSN

Log Sequence Numbers

WAL

<table>
<thead>
<tr>
<th>No.</th>
<th>Record</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>&lt;T_1 BEGIN&gt;</td>
</tr>
<tr>
<td>002</td>
<td>&lt;T_1, A, 1, 2&gt;</td>
</tr>
<tr>
<td>003</td>
<td>&lt;T_1 COMMIT&gt;</td>
</tr>
<tr>
<td>004</td>
<td>&lt;T_2 BEGIN&gt;</td>
</tr>
<tr>
<td>005</td>
<td>&lt;T_2, A, 2, 3&gt;</td>
</tr>
<tr>
<td>006</td>
<td>&lt;T_3 BEGIN&gt;</td>
</tr>
<tr>
<td>007</td>
<td>&lt;CHECKPOINT&gt;</td>
</tr>
<tr>
<td>008</td>
<td>&lt;T_2 COMMIT&gt;</td>
</tr>
<tr>
<td>009</td>
<td>&lt;T_3, A, 3, 4&gt;</td>
</tr>
<tr>
<td>010</td>
<td>&lt;T_4 BEGIN&gt;</td>
</tr>
<tr>
<td>011</td>
<td>&lt;T_4, X, 5, 6&gt;</td>
</tr>
<tr>
<td>012</td>
<td>&lt;T_4, Y, 9, 7&gt;</td>
</tr>
<tr>
<td>013</td>
<td>&lt;T_3, B, 4, 2&gt;</td>
</tr>
<tr>
<td>014</td>
<td>&lt;T_3 COMMIT&gt;</td>
</tr>
<tr>
<td>015</td>
<td>&lt;T_4, B, 2, 3&gt;</td>
</tr>
<tr>
<td>016</td>
<td>&lt;T_4, C, 1, 2&gt;</td>
</tr>
</tbody>
</table>

Database

<table>
<thead>
<tr>
<th>Page LSN</th>
<th>Record LSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>A=9</td>
<td>B=5</td>
</tr>
<tr>
<td></td>
<td>C=2</td>
</tr>
</tbody>
</table>

MasterRecord

flushedLSN
WRITING LOG RECORDS

WAL (Tail)

017: <T₅ BEGIN>
018: <T₅, A, 9, 8>
019: <T₅, B, 5, 1>
020: <T₅ COMMIT>

Buffer Pool

pageSN  recSN
A=9  B=5  C=2
flushedLSN

WAL

001: <T₁ BEGIN>
002: <T₁, A, 1, 2>
003: <T₁ COMMIT>
004: <T₂ BEGIN>
005: <T₂, A, 2, 3>
006: <T₂ COMMIT>
007: <CHECKPOINT>
008: <T₂ COMMIT>
009: <T₃, A, 3, 4>
010: <T₄ BEGIN>
011: <T₄, X, 5, 6>
012: <T₄, Y, 9, 7>
013: <T₃, B, 4, 2>
014: <T₃ COMMIT>
015: <T₄, B, 2, 3>
016: <T₄, C, 1, 2>

Database

pageSN  recSN
A=9  B=5  C=2
MasterRecord
flushedLSN
WRITING LOG RECORDS

WAL (Tail)

017: <T_5 BEGIN>
018: <T_5, A, 9, 8>
019: <T_5, B, 5, 1>
020: <T_5 COMMIT>

Buffer Pool

WAL

001: <T_1 BEGIN>
002: <T_1, A, 1, 2>
003: <T_1 COMMIT>
004: <T_2 BEGIN>
005: <T_2, A, 2, 3>
006: <T_3 BEGIN>
007: <CHECKPOINT>
008: <T_2 COMMIT>
009: <T_3, A, 3, 4>
010: <T_4 BEGIN>
011: <T_4, X, 5, 6>
012: <T_4, Y, 9, 7>
013: <T_3, B, 4, 2>
014: <T_3 COMMIT>
015: <T_4, B, 2, 3>
016: <T_4, C, 1, 2>

Database

MasterRecord

flushedLSN

pageLSN recLSN
A=9 B=5 C=2

pageLSN recLSN
A=9 B=5 C=2
WRITING LOG RECORDS

WAL (Tail)

017: <T_5 BEGIN>
018: <T_5, A, 9, 8>
019: <T_5, B, 5, 1>
020: <T_5 COMMIT>

Buffer Pool

<table>
<thead>
<tr>
<th>pageLSN</th>
<th>recLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>A=9</td>
<td>B=5</td>
</tr>
</tbody>
</table>

flushedLSN

MasterRecord

<table>
<thead>
<tr>
<th>pageLSN</th>
<th>recLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>A=9</td>
<td>B=5</td>
</tr>
</tbody>
</table>

Database

WAL

001: <T_1 BEGIN>
002: <T_1, A, 1, 2>
003: <T_1 COMMIT>
004: <T_2 BEGIN>
005: <T_2, A, 2, 3>
006: <T_3 BEGIN>
007: <CHECKPOINT>
008: <T_2 COMMIT>
009: <T_3, A, 3, 4>
100: <T_4 BEGIN>
101: <T_4, X, 5, 6>
102: <T_4, Y, 9, 7>
103: <T_3, B, 4, 2>
104: <T_3 COMMIT>
105: <T_4, B, 2, 3>
106: <T_4, C, 1, 2>
WAL (Tail)

017: <T₅ BEGIN>
018: <T₅, A, 9, 8>
019: <T₅, B, 5, 1>
020: <T₅ COMMIT>
...

Buffer Pool

WAL

001: <T₁, BEGIN>
002: <T₁, A, 1, 2>
003: <T₁ COMMIT>
004: <T₂, BEGIN>
005: <T₂, A, 2, 3>
006: <T₃ BEGIN>
007: <CHECKPOINT>
008: <T₂ COMMIT>
009: <T₃, A, 3, 4>
010: <T₄ BEGIN>
011: <T₄, X, 5, 6>
012: <T₄, Y, 9, 7>
013: <T₃, B, 4, 2>
014: <T₃ COMMIT>
015: <T₄, B, 2, 3>
016: <T₄, C, 1, 2>

Database

MasterRecord

Page LSN  Rec LSN
A=9  B=5  C=2

Page LSN  Rec LSN
flushedLSN

Page LSN  Rec LSN
flushedLSN

WAL (Tail)

017: <T₅ BEGIN>
018: <T₅, A, 9, 8>
019: <T₅, B, 5, 1>
020: <T₅ COMMIT>
...
WRITING LOG RECORDS

WAL (Tail)

017: <T_5 BEGIN>
018: <T_5, A, 9, 8>
019: <T_5, B, 5, 1>
020: <T_5 COMMIT>

Buffer Pool

A=9  B=5  C=2

flushedLSN

Database

A=9  B=5  C=2

MasterRecord

WAL

001: <T_1, BEGIN>
002: <T_1, A, 1, 2>
003: <T_1, COMMIT>
004: <T_2, BEGIN>
005: <T_2, A, 2, 3>
006: <T_2, COMMIT>
007: <CHECKPOINT>
008: <T_2, COMMIT>
009: <T_3, A, 3, 4>
100: <T_4, BEGIN>
101: <T_4, X, 5, 6>
102: <T_4, Y, 9, 7>
103: <T_3, B, 4, 2>
104: <T_3, COMMIT>
105: <T_4, B, 2, 3>
106: <T_4, C, 1, 2>

pageLSN  recLSN
WAL (Tail)

017: `<T₅ BEGIN>`
018: `<T₅, A, 9, 8>`
019: `<T₅, B, 5, 1>`
020: `<T₅ COMMIT>`

Buffer Pool

Safe to unpin because pageLSN ≤ flushedLSN

WAL

001: `<T₁ BEGIN>`
002: `<T₁, A, 1, 2>`
003: `<T₁ COMMIT>`
004: `<T₂ BEGIN>`
005: `<T₂, A, 2, 3>`
006: `<T₃ BEGIN>`
007: `<CHECKPOINT>`
008: `<T₂ COMMIT>`
009: `<T₃, A, 3, 4>`
010: `<T₄ BEGIN>`
011: `<T₄, X, 5, 6>`
012: `<T₄, Y, 9, 7>`
013: `<T₃, B, 4, 2>`
014: `<T₃ COMMIT>`
015: `<T₄, B, 2, 3>`
016: `<T₄, C, 1, 2>`

WAL (Tail) and WAL diagrams with annotations.
WAL (Tail)

017: <T₅ BEGIN>
018: <T₅, A, 9, 8>
019: <T₅, B, 5, 1>
020: <T₅ COMMIT>

Buffer Pool

Database

WAL

001: <T₁, BEGIN>
002: <T₁, A, 1, 2>
003: <T₁, COMMIT>
004: <T₂, BEGIN>
005: <T₂, A, 2, 3>
006: <T₂, COMMIT>
007: <CHECKPOINT>
008: <T₃, BEGIN>
009: <T₃, A, 3, 4>
010: <T₄, BEGIN>
011: <T₄, X, 5, 6>
012: <T₄, Y, 9, 7>
013: <T₃, B, 4, 2>
014: <T₃, COMMIT>
015: <T₄, B, 2, 3>
016: <T₄, C, 1, 2>

Writing Log Records
WAL (Tail)

017: <T₅, BEGIN>
018: <T₅, A, 9, 8>
019: <T₅, B, 5, 1>
020: <T₅, COMMIT>

Buffer Pool

WAL

001:<T₁, BEGIN>
002:<T₁, A, 1, 2>
003:<T₁, COMMIT>
004:<T₂, BEGIN>
005:<T₂, A, 2, 3>
006:<T₂, COMMIT>
007:<CHECKPOINT>
008:<T₂, COMMIT>
009:<T₃, A, 3, 4>
010:<T₄, BEGIN>
011:<T₄, X, 5, 6>
012:<T₄, Y, 9, 7>
013:<T₃, B, 4, 2>
014:<T₃, COMMIT>
015:<T₄, B, 2, 3>
016:<T₄, C, 1, 2>

Not safe to unpin because pageLSN > flushedLSN

W R I T I N G  L O G  R E C O R D S

pageLSN  recLSN
A=9  B=5  C=2

MasterRecord

Database
WRITING LOG RECORDS

All log records have an **LSN**.

Update the **pageLSN** every time a txn modifies a record in the page.

Update the **flushedLSN** in memory every time the DBMS writes out the WAL buffer to disk.
NORMAL EXECUTION

Each txn invokes a sequence of reads and writes, followed by commit or abort.

Assumptions in this lecture:
→ All log records fit within a single page.
→ Disk writes are atomic.
→ Single-versioned tuples with Strict 2PL.
→ STEAL + NO-FORCE buffer management with WAL.
TRANSACTION COMMIT

Write **COMMIT** record to log.

All log records up to txn's **COMMIT** record are flushed to disk.
→ Log flushes are sequential, synchronous writes to disk.
→ Many log records per log page.

When the commit succeeds, write a special **TXN-END** record to log.
→ This does **not** need to be flushed immediately.
TRANSACTION COMMIT

WAL (Tail)

012: <T₄, BEGIN>
013: <T₄, A, 9, 8>
014: <T₄, B, 5, 1>
015: <T₄, COMMIT>

Buffer Pool

pageSN  recLSN
A=9  B=5  C=2
flushedLSN

WAL

001: <T₁, BEGIN>
002: <T₁, A, 1, 2>
003: <T₁, COMMIT>
004: <T₂, BEGIN>
005: <T₂, A, 2, 3>
006: <T₂, COMMIT>
007: <CHECKPOINT>
008: <T₂, COMMIT>
009: <T₃, A, 3, 4>
010: <T₃, B, 4, 2>
011: <T₃, COMMIT>

Database

pageSN  recSN
A=9  B=5  C=2
MasterRecord

Database MasterRecord
flushedLSN


TRANSACTION COMMIT

WAL (Tail)

012: <T₄ BEGIN>
013: <T₄, A, 9, 8>
014: <T₄, B, 5, 1>
015: <T₄ COMMIT>

Buffer Pool

<table>
<thead>
<tr>
<th>pageLSN</th>
<th>recLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>A=9</td>
<td>B=5</td>
</tr>
<tr>
<td>C=2</td>
<td></td>
</tr>
</tbody>
</table>

flushedLSN

Database

<table>
<thead>
<tr>
<th>pageLSN</th>
<th>recLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>A=9</td>
<td>B=5</td>
</tr>
<tr>
<td>C=2</td>
<td></td>
</tr>
</tbody>
</table>

WAL

001: <T₁, BEGIN>
002: <T₁, A, 1, 2>
003: <T₁, COMMIT>
004: <T₂, BEGIN>
005: <T₂, A, 2, 3>
006: <T₂, BEGIN>
007: <CHECKPOINT>
008: <T₂, COMMIT>
009: <T₃, A, 3, 4>
010: <T₃, B, 4, 2>
011: <T₃, COMMIT>

MasterRecord
TRANSACTION COMMIT

WAL (Tail)

012: \langle T_4, \text{BEGIN}\rangle
013: \langle T_4, A, 9, 8\rangle
014: \langle T_4, B, 5, 1\rangle
015: \langle T_4, \text{COMMIT}\rangle

WAL

001: \langle T_1, \text{BEGIN}\rangle
002: \langle T_1, A, 1, 2\rangle
003: \langle T_1, \text{COMMIT}\rangle
004: \langle T_2, \text{BEGIN}\rangle
005: \langle T_2, A, 2, 3\rangle
006: \langle T_3, \text{BEGIN}\rangle
007: \langle \text{CHECKPOINT}\rangle
008: \langle T_2, \text{COMMIT}\rangle
009: \langle T_3, A, 3, 4\rangle
010: \langle T_3, B, 4, 2\rangle
011: \langle T_3, \text{COMMIT}\rangle
012: \langle T_4, \text{BEGIN}\rangle
013: \langle T_4, A, 9, 8\rangle
014: \langle T_4, B, 5, 1\rangle
015: \langle T_4, \text{COMMIT}\rangle

Buffer Pool

flushedLSN = 015

Database

flushedLSN

A=9 B=5 C=2
**TRANSACTION COMMIT**

**WAL (Tail)**
- 012: `<T₄ BEGIN>`
- 013: `<T₄, A, 9, 8>`
- 014: `<T₄, B, 5, 1>`
- 015: `<T₄ COMMIT>`

**Buffer Pool**
- `pageLSN` `recLSN`
  - A=9  B=5  C=2

**WAL**
- 001: `<T₁ BEGIN>`
- 002: `<T₁, A, 1, 2>`
- 003: `<T₁ COMMIT>`
- 004: `<T₂ BEGIN>`
- 005: `<T₂, A, 2, 3>`
- 006: `<T₂ COMMIT>`
- 007: `<CHECKPOINT>`
- 008: `<T₃ BEGIN>`
- 009: `<T₃, A, 3, 4>`
- 010: `<T₃, B, 4, 2>`
- 011: `<T₃, COMMIT>`
- 012: `<T₄ BEGIN>`
- 013: `<T₄, A, 9, 8>`
- 014: `<T₄, B, 5, 1>`
- 015: `<T₄ COMMIT>`

**Database**
- `pageLSN` `recLSN`
  - A=9  B=5  C=2

**flushedLSN**

**MasterRecord**
TRANSACTION COMMIT

WAL (Tail)

012: <T₄, BEGIN>
013: <T₄, A, 9, 8>
014: <T₄, B, 5, 1>
015: <T₄, COMMIT>

099: <T₄, TXN-END>

Buffer Pool

A=9, B=5, C=2

flushedLSN

Database

001: <T₁, BEGIN>
002: <T₁, A, 1, 2>
003: <T₁, COMMIT>
004: <T₂, BEGIN>
005: <T₂, A, 2, 3>
006: <T₂, COMMIT>
007: <CHECKPOINT>
008: <T₃, BEGIN>
009: <T₃, A, 3, 4>
010: <T₃, B, 4, 2>
011: <T₃, COMMIT>
012: <T₄, BEGIN>
013: <T₄, A, 9, 8>
014: <T₄, B, 5, 1>
015: <T₄, COMMIT>

pageLSN, recLSN

A=9, B=5, C=2

MasterRecord
We can trim the in-memory log up to flushedLSN.

Transaction Commit

```
012: <T₄ BEGIN>
013: <T₄, A, 9, 8>
014: <T₄, B, 5, 1>
015: <T₄ COMMIT>
...
099: <T₄ TXN-END>
```
We can trim the in-memory log up to flushedLSN.
TRANSACTION ABORT

Aborting a txn is a special case of the ARIES undo operation applied to only one txn.

We need to add another field to our log records:

→ prevLSN: The previous LSN for the txn.
→ This maintains a linked-list for each txn that makes it easy to walk through its records.
**TRANSACTION ABORT**

**WAL (Tail)**

- 012|nil:<T₄ BEGIN>
- 013|012:<T₄, A, 9, 8>
- 014|013:<T₄, B, 5, 1>

**Buffer Pool**

- pagerSN | recLSN
- A=9 | B=5 | C=2

- flushedLSN

**WAL**

- pagerSN | recLSN
- A=9 | B=5 | C=2

**Database**

**MasterRecord**
TRANSACTION ABORT

- WAL (Tail)
  - LSN  | prevLSN
  - 012|nil
  - 013|012
  - 014|013

- Database
  - MasterRecord
    - pageSN | recSN
    - A=9  B=5  C=2

- Buffer Pool
  - pageSN | recSN
  - A=9  B=5  C=2
  - flushedLSN
 TRANSACTION ABORT

WAL (Tail)

012|nil:<T4, BEGIN>
013|012:<T4, A, 9, 8>
014|013:<T4, B, 5, 1>

Buffer Pool

<table>
<thead>
<tr>
<th>pageSN</th>
<th>recSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>A=9</td>
<td>B=5</td>
</tr>
</tbody>
</table>

flushedLSN

WAL

<table>
<thead>
<tr>
<th>pageSN</th>
<th>recSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>A=9</td>
<td>B=5</td>
</tr>
</tbody>
</table>

MasterRecord

Database
TRANSACTION ABORT

WAL (Tail)

012|nil:<T₄ BEGIN>
013|012:<T₄, A, 9, 8>
014|013:<T₄, B, 5, 1>
015|014:<T₄ ABORT>

Buffer Pool

pageLSN reclSN
A=9 B=5 C=2

flushedLSN

Database

pageLSN reclSN
A=9 B=5 C=2

MasterRecord
**Transaction Abort**

**WAL (Tail)**

012|nil:<T₄ BEGIN>
013|012:<T₄, A, 9, 8>
014|013:<T₄, B, 5, 1>
015|014:<T₄ ABORT>
???
099|098:<T₄ TXN-END>

**Buffer Pool**

<table>
<thead>
<tr>
<th>pagelSN</th>
<th>recLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>A=9</td>
<td>B=5</td>
</tr>
<tr>
<td>C=2</td>
<td></td>
</tr>
</tbody>
</table>

**FlushedLSN**

**WAL**

<table>
<thead>
<tr>
<th>pagelSN</th>
<th>recLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>A=9</td>
<td>B=5</td>
</tr>
<tr>
<td>C=2</td>
<td></td>
</tr>
</tbody>
</table>

**MasterRecord**
**TRANSACTION ABORT**

**WAL (Tail)**

| 012 | nil: <T₄, BEGIN |
| 013 | 012: <T₄, A, 9 |
| 014 | 013: <T₄, B, 5 |
| 015 | 014: <T₄, ABORT |
| 099 | 098: <T₄, TXN-END |

**Buffer Pool**

- pageSN
- recSN
- A=9
- B=5
- C=2
- flushedLSN

Important: Need to record what steps we took to undo the txn.
COMPENSATION LOG RECORDS

A **CLR** describes the actions taken to undo the actions of a previous update record.

It has all the fields of an update log record plus the **undoNext** pointer (the next-to-be-undone LSN).

**CLRs** are added to log records but the DBMS does not wait for them to be flushed before notifying the application that the txn aborted.
## TRANSACTION ABORT – CLR EXAMPLE

<table>
<thead>
<tr>
<th>LSN</th>
<th>prevLSN</th>
<th>TxnId</th>
<th>Type</th>
<th>Object</th>
<th>Before</th>
<th>After</th>
<th>UndoNext</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>nil</td>
<td>T₁</td>
<td>BEGIN</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>002</td>
<td>001</td>
<td>T₁</td>
<td>UPDATE</td>
<td>A</td>
<td>30</td>
<td>40</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>011</td>
<td>002</td>
<td>T₁</td>
<td>ABORT</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
## TRANSACTION ABORT – CLR EXAMPLE

<table>
<thead>
<tr>
<th>LSN</th>
<th>prevLSN</th>
<th>TxnId</th>
<th>Type</th>
<th>Object</th>
<th>Before</th>
<th>After</th>
<th>UndoNext</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>nil</td>
<td>T₁</td>
<td>BEGIN</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>002</td>
<td>001</td>
<td>T₁</td>
<td>UPDATE</td>
<td>A</td>
<td>30</td>
<td>40</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>011</td>
<td>002</td>
<td>T₁</td>
<td>ABORT</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>026</td>
<td>011</td>
<td>T₁</td>
<td>CLR-002</td>
<td>A</td>
<td>40</td>
<td>30</td>
<td>001</td>
</tr>
</tbody>
</table>

TIME
## TRANSACTION ABORT – CLR EXAMPLE

<table>
<thead>
<tr>
<th>LSN</th>
<th>prevLSN</th>
<th>TxnId</th>
<th>Type</th>
<th>Object</th>
<th>Before</th>
<th>After</th>
<th>Undo</th>
<th>Next</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>nil</td>
<td>T₁</td>
<td>BEGIN</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>002</td>
<td>001</td>
<td>T₁</td>
<td>UPDATE</td>
<td>A</td>
<td>30</td>
<td>40</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>011</td>
<td>002</td>
<td>T₁</td>
<td>ABORT</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>026</td>
<td>011</td>
<td>T₁</td>
<td>CLR-002</td>
<td>A</td>
<td>40</td>
<td>30</td>
<td>001</td>
<td></td>
</tr>
</tbody>
</table>

**TIME**
## TRANSACTION ABORT – CLR EXAMPLE

<table>
<thead>
<tr>
<th>LSN</th>
<th>prevLSN</th>
<th>TxnId</th>
<th>Type</th>
<th>Object</th>
<th>Before</th>
<th>After</th>
<th>UndoNext</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>nil</td>
<td>T₁</td>
<td>BEGIN</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>002</td>
<td>001</td>
<td>T₁</td>
<td>UPDATE</td>
<td>A</td>
<td>30</td>
<td>40</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>011</td>
<td>002</td>
<td>T₁</td>
<td>ABORT</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>026</td>
<td>011</td>
<td>T₁</td>
<td>CLR-002</td>
<td>A</td>
<td>40</td>
<td>30</td>
<td>001</td>
</tr>
</tbody>
</table>
## TRANSACTION ABORT – CLR EXAMPLE

<table>
<thead>
<tr>
<th>LSN</th>
<th>prevLSN</th>
<th>TxnId</th>
<th>Type</th>
<th>Object</th>
<th>Before</th>
<th>After</th>
<th>UndoNext</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>nil</td>
<td>T₁</td>
<td>BEGIN</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>002</td>
<td>001</td>
<td>T₁</td>
<td>UPDATE</td>
<td>A</td>
<td>30</td>
<td>40</td>
<td>-</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>011</td>
<td>002</td>
<td>T₁</td>
<td>ABORT</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>026</td>
<td>011</td>
<td>T₁</td>
<td>CLR-002</td>
<td>A</td>
<td>40</td>
<td>30</td>
<td>001</td>
</tr>
</tbody>
</table>
### TRANSACTION ABORT – CLR EXAMPLE

<table>
<thead>
<tr>
<th>LSN</th>
<th>prevLSN</th>
<th>TxnId</th>
<th>Type</th>
<th>Object</th>
<th>Before</th>
<th>After</th>
<th>UndoNext</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>nil</td>
<td>T₁</td>
<td>BEGIN</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>002</td>
<td>001</td>
<td>T₁</td>
<td>UPDATE</td>
<td>A</td>
<td>30</td>
<td>40</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>011</td>
<td>002</td>
<td>T₁</td>
<td>ABORT</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>026</td>
<td>011</td>
<td>T₁</td>
<td>CLR-002</td>
<td>A</td>
<td>40</td>
<td>30</td>
<td>001</td>
</tr>
</tbody>
</table>

The LSN of the next log record to be undone.
## TRANSACTION ABORT – CLR EXAMPLE

<table>
<thead>
<tr>
<th>LSN</th>
<th>prevLSN</th>
<th>TxnId</th>
<th>Type</th>
<th>Object</th>
<th>Before</th>
<th>After</th>
<th>UndoNext</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>nil</td>
<td>T₁</td>
<td>BEGIN</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>002</td>
<td>001</td>
<td>T₁</td>
<td>UPDATE</td>
<td>A</td>
<td>30</td>
<td>40</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>011</td>
<td>002</td>
<td>T₁</td>
<td>ABORT</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>026</td>
<td>011</td>
<td>T₁</td>
<td>CLR-002</td>
<td>A</td>
<td>40</td>
<td>30</td>
<td>001</td>
</tr>
<tr>
<td>027</td>
<td>026</td>
<td>T₁</td>
<td>TXN-End</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>nil</td>
</tr>
</tbody>
</table>
ABORT ALGORITHM

First write an **ABORT** record to log for the txn.

Then play back the txn's updates in reverse order. For each update record:

→ Write a **CLR** entry to the log.
→ Restore old value.

At end, write a **TXN-END** log record.

Notice: **CLRs** never need to be undone.
TODAY’S AGENDA

- Log Sequence Numbers
- Normal Commit & Abort Operations
- Fuzzy Checkpointing
- Recovery Algorithm
NON-FUZZY CHECKPOINTS

The DBMS halts everything when it takes a checkpoint to ensure a consistent snapshot:

→ Halt the start of any new txns.
→ Wait until all active txns finish executing.
→ Flushes dirty pages on disk.

This is bad for runtime performance but makes recovery easy.
SLIGHTLY BETTER CHECKPOINTS

Pause modifying txns while the DBMS takes the checkpoint.

→ Prevent queries from acquiring write latch on table/index pages.
→ Don't have to wait until all txns finish before taking the checkpoint.
Pause modifying txns while the DBMS takes the checkpoint.
→ Prevent queries from acquiring write latch on table/index pages.
→ Don't have to wait until all txns finish before taking the checkpoint.
SLIGHTLY BETTER CHECKPOINTS

Pause modifying txns while the DBMS takes the checkpoint.

→ Prevent queries from acquiring write latch on table/index pages.
→ Don't have to wait until all txns finish before taking the checkpoint.
Pause modifying txns while the DBMS takes the checkpoint.

→ Prevent queries from acquiring write latch on table/index pages.
→ Don't have to wait until all txns finish before taking the checkpoint.
SLIGHTLY BETTER CHECKPOINTS

Pause modifying txns while the DBMS takes the checkpoint.

→ Prevent queries from acquiring write latch on table/index pages.
→ Don't have to wait until all txns finish before taking the checkpoint.
SLIGHTLY BETTER CHECKPOINTS

Pause modifying txns while the DBMS takes the checkpoint.

→ Prevent queries from acquiring write latch on table/index pages.
→ Don't have to wait until all txns finish before taking the checkpoint.
Pause modifying txns while the DBMS takes the checkpoint.

→ Prevent queries from acquiring write latch on table/index pages.
→ Don't have to wait until all txns finish before taking the checkpoint.
Pause modifying txns while the DBMS takes the checkpoint.

→ Prevent queries from acquiring write latch on table/index pages.
→ Don't have to wait until all txns finish before taking the checkpoint.
Pause modifying txns while the DBMS takes the checkpoint.
→ Prevent queries from acquiring write latch on table/index pages.
→ Don't have to wait until all txns finish before taking the checkpoint.

We must record internal state as of the beginning of the checkpoint.
→ Active Transaction Table (ATT)
→ Dirty Page Table (DPT)
ACTIVE TRANSACTION TABLE

One entry per currently active txn.

→ **txnId**: Unique txn identifier.
→ **status**: The current "mode" of the txn.
→ **lastLSN**: Most recent *LSN* created by txn.

Entry removed after the txn commits or aborts.

Txn Status Codes:

→ **R** → Running
→ **C** → Committing
→ **U** → Candidate for Undo
DIRTY PAGE TABLE

Keep track of which pages in the buffer pool contain changes from uncommitted transactions.

One entry per dirty page in the buffer pool:
-> recLSN: The LSN of the log record that first caused the page to be dirty.
At the first checkpoint, $T_2$ is still running and there is one dirty page ($P_{22}$).
At the first checkpoint, $T_2$ is still running and there is one dirty page ($P_{22}$).
At the first checkpoint, $T_2$ is still running and there is one dirty page ($P_{22}$).
At the first checkpoint, $T_2$ is still running and there is one dirty page ($P_{22}$).
At the first checkpoint, $T_2$ is still running and there is one dirty page ($P_{22}$).
At the first checkpoint, $T_2$ is still running and there is one dirty page ($P_{22}$).
SLIGHTLY BETTER CHECKPOINTS

At the first checkpoint, $T_2$ is still running and there is one dirty page ($P_{22}$).

At the second checkpoint, $T_3$ is active and there is one dirty page ($P_{33}$).
SLIGHTLY BETTER CHECKPOINTS

At the first checkpoint, $T_2$ is still running and there is one dirty page ($P_{22}$).

At the second checkpoint, $T_3$ is active and there is one dirty page ($P_{33}$).
At the first checkpoint, \( T_2 \) is still running and there is one dirty page \( P_{22} \).

At the second checkpoint, \( T_3 \) is active and there is one dirty page \( P_{33} \).
SLIGHTLY BETTER CHECKPOINTS

At the first checkpoint, $T_2$ is still running and there is one dirty page ($P_{22}$).

At the second checkpoint, $T_3$ is active and there is one dirty pages ($P_{33}$).

This still is not ideal because the DBMS must stall txns during checkpoint...
**FUZZY CHECKPOINTS**

A *fuzzy checkpoint* is where the DBMS allows active txns to continue the run while the system flushes dirty pages to disk.

New log records to track checkpoint boundaries:

- → **CHECKPOINT-BEGIN**: Indicates start of checkpoint
- → **CHECKPOINT-END**: Contains **ATT + DPT**.
FUZZY CHECKPOINTS

The **LSN** of the **CHECKPOINT-BEGIN** record is written to the database's **MasterRecord** entry on disk when the checkpoint successfully completes.

Any txn that starts **after** the checkpoint is excluded from the **ATT** in the **CHECKPOINT-END** record.
FUZZY CHECKPOINTS

The $LSN$ of the **CHECKPOINT-BEGIN** record is written to the database's **MasterRecord** entry on disk when the checkpoint successfully completes.

Any txn that starts **after** the checkpoint is excluded from the ATT in the **CHECKPOINT-END** record.
The **LSN** of the **CHECKPOINT-BEGIN** record is written to the database's **MasterRecord** entry on disk when the checkpoint successfully completes.

Any txn that starts **after** the checkpoint is excluded from the **ATT** in the **CHECKPOINT-END** record.
FUZZY CHECKPOINTS

The **LSN** of the **CHECKPOINT-BEGIN** record is written to the database's **MasterRecord** entry on disk when the checkpoint successfully completes.

Any txn that starts **after** the checkpoint is excluded from the ATT in the **CHECKPOINT-END** record.
The \textit{LSN} of the \texttt{CHECKPOINT-BEGIN} record is written to the database's \texttt{MasterRecord} entry on disk when the checkpoint successfully completes.

Anytxn that starts \underline{after} the checkpoint is excluded from the ATT in the \texttt{CHECKPOINT-END} record.
FUZZY CHECKPOINTS

The LSN of the CHECKPOINT-BEGIN record is written to the database's MasterRecord entry on disk when the checkpoint successfully completes.

Any txn that starts after the checkpoint is excluded from the ATT in the CHECKPOINT-END record.
The **LSN** of the **CHECKPOINT-BEGIN** record is written to the database's **MasterRecord** entry on disk when the checkpoint successfully completes.

Any txn that starts **after** the checkpoint is excluded from the **ATT** in the **CHECKPOINT-END** record.
The **LSN** of the CHECKPOINT-BEGIN record is written to the database's MasterRecord entry on disk when the checkpoint successfully completes.

Any txn that starts **after** the checkpoint is excluded from the ATT in the CHECKPOINT-END record.
ARIES – RECOVERY PHASES

Phase #1 – Analysis
→ Read WAL from last MasterRecord to identify dirty pages in the buffer pool and active txns at the time of the crash.

Phase #2 – Redo
→ Repeat all actions starting from an appropriate point in the log (even txns that will abort).

Phase #3 – Undo
→ Reverse the actions of txns that did not commit before the crash.
ARIES – OVERVIEW

Start from last **BEGIN-CHECKPOINT** found via **MasterRecord**.

**Analysis:** Figure out which txns committed or failed since checkpoint.

**Redo:** Repeat all actions.

**Undo:** Reverse effects of failed txns.
Start from last **BEGIN-CHECKPOINT** found via **MasterRecord**.

**Analysis:** Figure out which txns committed or failed since checkpoint.

**Redo:** Repeat all actions.

**Undo:** Reverse effects of failed txns.
Start from last **BEGIN-CHECKPOINT** found via **MasterRecord**.

**Analysis:** Figure out which txns committed or failed since checkpoint.

**Redo:** Repeat **all** actions.

**Undo:** Reverse effects of failed txns.
Start from last **BEGIN-CHECKPOINT** found via **MasterRecord**.

**Analysis:** Figure out which txns committed or failed since checkpoint.

**Redo:** Repeat all actions.

**Undo:** Reverse effects of failed txns.
Start from last **BEGIN-CHECKPOINT** found via **MasterRecord**.

**Analysis**: Figure out which txns committed or failed since checkpoint.

**Redo**: Repeat **all** actions.

**Undo**: Reverse effects of failed txns.
Start from last **BEGIN-CHECKPOINT** found via **MasterRecord**.

**Analysis:** Figure out which txns committed or failed since checkpoint.

**Redo:** Repeat all actions.

**Undo:** Reverse effects of failed txns.
Start from last `BEGIN-CHECKPOINT` found via `MasterRecord`.

**Analysis:** Figure out which txns committed or failed since checkpoint.

**Redo:** Repeat all actions.

**Undo:** Reverse effects of failed txns.
ARIES – OVERVIEW

Start from last **BEGIN-CHECKPOINT** found via **MasterRecord**.

**Analysis**: Figure out which txns committed or failed since checkpoint.

**Redo**: Repeat all actions.

**Undo**: Reverse effects of failed txns.
ARIES – OVERVIEW

Start from last **BEGIN-CHECKPOINT** found via **MasterRecord**.

**Analysis:** Figure out which txns committed or failed since checkpoint.

**Redo:** Repeat all actions.

**Undo:** Reverse effects of failed txns.
Start from last **BEGIN-CHECKPOINT** found via **MasterRecord**.

**Analysis:** Figure out which txns committed or failed since checkpoint.

**Redo:** Repeat all actions.

**Undo:** Reverse effects of failed txns.
ANALYSIS PHASE

Scan log forward from last successful checkpoint.
If you find a **TXN-END** record, remove its corresponding txn from **ATT**.

All other records:
→ Add txn to **ATT** with status **UNDO**.
→ On commit, change txn status to **COMMIT**.

For **UPDATE** records:
→ If page **P** not in **DPT**, add **P** to **DPT**, set its **recLSN=LSN**.
ANALYSIS PHASE

At end of the Analysis Phase:
→ **ATT** identifies which txns were active at time of crash.
→ **DPT** identifies which dirty pages might not have made it to disk.
ANALYSIS PHASE EXAMPLE

WAL

010: <CHECKPOINT-BEGIN>

020: <T_{96}, A\rightarrow P_{33}, 10, 15>

030: <CHECKPOINT-END
ATT=\{T_{96}, T_{97}\},
DPT=\{P_{20}, P_{33}\}>

040: <T_{96} COMMIT>

050: <T_{96} TXN-END>

CRASH!

<table>
<thead>
<tr>
<th>LSN</th>
<th>ATT</th>
<th>DPT</th>
</tr>
</thead>
<tbody>
<tr>
<td>010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>020</td>
<td></td>
<td></td>
</tr>
<tr>
<td>030</td>
<td></td>
<td></td>
</tr>
<tr>
<td>040</td>
<td></td>
<td></td>
</tr>
<tr>
<td>050</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**ANALYSIS PHASE EXAMPLE**

**WAL**

```
010: <CHECKPOINT-BEGIN>
020: <T96, A→P33, 10, 15>
030: <CHECKPOINT-END
    ATT={T96, T97},
    DPT={P20, P33}>
040: <T96 COMMIT>
050: <T96 TXN-END>
CRASH!
```

<table>
<thead>
<tr>
<th>LSN</th>
<th>ATT</th>
<th>DPT</th>
</tr>
</thead>
<tbody>
<tr>
<td>010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>020</td>
<td></td>
<td></td>
</tr>
<tr>
<td>030</td>
<td></td>
<td></td>
</tr>
<tr>
<td>040</td>
<td></td>
<td></td>
</tr>
<tr>
<td>050</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Analysis Phase Example**

### WAL

<table>
<thead>
<tr>
<th>LSN</th>
<th>ATT</th>
<th>DPT</th>
</tr>
</thead>
<tbody>
<tr>
<td>010</td>
<td>[CHECKPOINT-BEGIN]</td>
<td></td>
</tr>
<tr>
<td>020</td>
<td>&lt;T_{96}, A\rightarrow P_{33}, 10, 15&gt;</td>
<td></td>
</tr>
<tr>
<td>030</td>
<td>&lt;CHECKPOINT-END ATT={{T_{96}, T_{97}}, DPT={P_{20}, P_{33}}}</td>
<td></td>
</tr>
<tr>
<td>040</td>
<td>&lt;T_{96} COMMIT&gt;</td>
<td></td>
</tr>
<tr>
<td>050</td>
<td>&lt;T_{96} TXN-END&gt;</td>
<td></td>
</tr>
</tbody>
</table>

**CRASH!**
ANALYSIS PHASE EXAMPLE

WAL

010: <CHECKPOINT-BEGIN>

020: <T₉₆, A→P₃₃, 10, 15>

030: <CHECKPOINT-END
   ATT={T₉₆, T₉₇},
   DPT={P₂₀, P₃₃}>

040: <T₉₆ COMMIT>

050: <T₉₆ TXN-END>

CRASH!

<table>
<thead>
<tr>
<th>LSN</th>
<th>ATT</th>
<th>DPT</th>
</tr>
</thead>
<tbody>
<tr>
<td>010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>020</td>
<td></td>
<td></td>
</tr>
<tr>
<td>030</td>
<td></td>
<td></td>
</tr>
<tr>
<td>040</td>
<td></td>
<td></td>
</tr>
<tr>
<td>050</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(TxnId, Status)
### Analysis Phase Example

#### WAL Log

<table>
<thead>
<tr>
<th>Log Entry</th>
<th>Action</th>
<th>Timestamps</th>
</tr>
</thead>
<tbody>
<tr>
<td>010: &lt;CHECKPOINT-BEGIN&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>020: &lt;T96, A→P33, 10, 15&gt;</td>
<td>Modify A in page P33</td>
<td></td>
</tr>
<tr>
<td>030: &lt;CHECKPOINT-END&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ATT={T96, T97}, DPT={P20, P33}&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>040: &lt;T96 COMMIT&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>050: &lt;T96 TXN-END&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRASH!</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### ATT and DPT

<table>
<thead>
<tr>
<th></th>
<th>ATT</th>
<th>DPT</th>
</tr>
</thead>
<tbody>
<tr>
<td>010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>020</td>
<td>(T96, U)</td>
<td></td>
</tr>
<tr>
<td>030</td>
<td></td>
<td></td>
</tr>
<tr>
<td>040</td>
<td></td>
<td></td>
</tr>
<tr>
<td>050</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
ANALYSIS PHASE EXAMPLE

Modify A in page P_{33}
**ANALYSIS PHASE EXAMPLE**

**WAL**

010: <CHECKPOINT-BEGIN>

...  

020: <T_{96}, A\rightarrow P_{33}, 10, 15>

...  

030: <CHECKPOINT-END
    ATT={T_{96}, T_{97}},
    DPT={P_{20}, P_{33}}>

...  

040: <T_{96} COMMIT>

...  

050: <T_{96} TXN-END>

CRASH!

---

<table>
<thead>
<tr>
<th>LSN</th>
<th>ATT</th>
<th>DPT</th>
</tr>
</thead>
<tbody>
<tr>
<td>010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>020</td>
<td>(T_{96}, U)</td>
<td>(P_{33}, 020)</td>
</tr>
<tr>
<td>030</td>
<td></td>
<td></td>
</tr>
<tr>
<td>040</td>
<td></td>
<td></td>
</tr>
<tr>
<td>050</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### ANALYSIS PHASE EXAMPLE

#### WAL

010: `<CHECKPOINT-BEGIN>`

020: `<T_{96}, A \rightarrow P_{33}, 10, 15>`

030: `<CHECKPOINT-END`
   
   ATT={T_{96}, T_{97}},
   
   DPT={P_{20}, P_{33}}`

040: `<T_{96} COMMIT>`

050: `<T_{96} TXN-END>`

CRASH!

#### LSN  | ATT       | DPT
---------|------------|------------------
010      |            |                  |
020      | (T_{96}, U)| (P_{33}, 020)   |
030      | (T_{96}, U), (T_{97}, U) | (P_{33}, 020), (P_{20}, 022) |
040      |            |                  |
050      |            |                  |
ANALYSIS PHASE EXAMPLE

WAL

010: <CHECKPOINT-BEGIN>

020: <T96, A→P33, 10, 15>

030: <CHECKPOINT-END
ATT={T96, T97},
DPT={P20, P33}>

040: <T96 COMMIT>

050: <T96 TXN-END>

CRASH!

<table>
<thead>
<tr>
<th>LSN</th>
<th>ATT</th>
<th>DPT</th>
</tr>
</thead>
<tbody>
<tr>
<td>010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>020</td>
<td>(T96, U)</td>
<td>(P33, 020)</td>
</tr>
<tr>
<td>030</td>
<td>(T96, U), (T97, U)</td>
<td>(P33, 020), (P20, 022)</td>
</tr>
<tr>
<td>040</td>
<td>(T96, C), (T97, U)</td>
<td>(P33, 020), (P20, 022)</td>
</tr>
<tr>
<td>050</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
ANALYSIS PHASE EXAMPLE

WAL

010: <CHECKPOINT-BEGIN>

020: <T96, A→P33, 10, 15>

030: <CHECKPOINT-END
ATT={T96, T97},
DPT={P20, P33}>

040: <T96 COMMIT>

050: <T96 TXN-END>

CRASH!

LSN | ATT          | DPT          |
--- | ------------ | ------------ |
010 |              |              |
020 | (T96, U)    | (P33, 020)  |
030 | (T96, U),  (T97, U) | (P33, 020), (P20, 022) |
040 | (T96, C),  (T97, U) | (P33, 020), (P20, 022) |
050 | (T97, U)    | (P33, 020), (P20, 022) |
REDO PHASE

The goal is to repeat history to reconstruct state at the moment of the crash:
→ Reapply all updates (even aborted txns!) and redo CLR.

There are techniques that allow the DBMS to avoid unnecessary reads/writes, but we will ignore that in this lecture...
REDO PHASE

Scan forward from the log record containing smallest recLSN in DPT.

For each update log record or CLR with a given LSN, redo the action unless:
→ Affected page is not in DPT, or
→ Affected page is in DPT but that record's LSN is less than the page's recLSN.
REDO PHASE

To redo an action:
→ Reapply logged action.
→ Set pageLSN to log record's LSN.
→ No additional logging, no forcedflushes!

At the end of Redo Phase, write TXN-END log records for all txns with status C and remove them from the ATT.
**UNDO PHASE**

Undo all txns that were active at the time of crash and therefore will never commit.

→ These are all the txns with U status in the ATT after the Analysis Phase.

Process them in reverse LSN order using the lastLSN to speed up traversal.

Write a CLR for every modification.
FULL EXAMPLE

<table>
<thead>
<tr>
<th>TIME</th>
<th>LSN</th>
<th>LOG</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td></td>
<td>&lt;CHECKPOINT-BEGIN&gt;</td>
</tr>
<tr>
<td>05</td>
<td></td>
<td>&lt;CHECKPOINT-END&gt;</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>&lt;T₁, A→P₅, 1, 2&gt;</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>&lt;T₂, B→P₃, 2, 3&gt;</td>
</tr>
<tr>
<td>30</td>
<td></td>
<td>&lt;T₁ ABORT&gt;</td>
</tr>
<tr>
<td>40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>45</td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>60</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
FULL EXAMPLE

<table>
<thead>
<tr>
<th>TIME</th>
<th>LSN</th>
<th>LOG</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td></td>
<td>&lt;CHECKPOINT-BEGIN&gt;</td>
</tr>
<tr>
<td>05</td>
<td></td>
<td>&lt;CHECKPOINT-END&gt;</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>&lt;T₁, A→P₅, 1, 2&gt;</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>&lt;T₂, B→P₃, 2, 3&gt;</td>
</tr>
<tr>
<td>30</td>
<td></td>
<td>&lt;T₁ ABORT&gt;</td>
</tr>
<tr>
<td>40</td>
<td></td>
<td>&lt;CLR: Undo T₁ LSN 10&gt;</td>
</tr>
<tr>
<td>45</td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>60</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
FULL EXAMPLE

<table>
<thead>
<tr>
<th>LSN</th>
<th>LOG</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>&lt;CHECKPOINT-BEGIN&gt;</td>
</tr>
<tr>
<td>05</td>
<td>&lt;CHECKPOINT-END&gt;</td>
</tr>
<tr>
<td>10</td>
<td>&lt;T₁, A→P₅, 1, 2&gt;</td>
</tr>
<tr>
<td>20</td>
<td>&lt;T₂, B→P₃, 2, 3&gt;</td>
</tr>
<tr>
<td>30</td>
<td>&lt;T₁ ABORT&gt;</td>
</tr>
<tr>
<td>40</td>
<td>&lt;CLR: Undo T₁ LSN 10&gt;</td>
</tr>
<tr>
<td>45</td>
<td>&lt;T₁ TXN-END&gt;</td>
</tr>
<tr>
<td>50</td>
<td></td>
</tr>
<tr>
<td>60</td>
<td></td>
</tr>
</tbody>
</table>
FULL EXAMPLE

<table>
<thead>
<tr>
<th>LSN</th>
<th>LOG</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>&lt;CHECKPOINT-BEGIN&gt;</td>
</tr>
<tr>
<td>05</td>
<td>&lt;CHECKPOINT-END&gt;</td>
</tr>
<tr>
<td>10</td>
<td>&lt;T₁, A→P₅, 1, 2&gt;</td>
</tr>
<tr>
<td>20</td>
<td>&lt;T₂, B→P₃, 2, 3&gt;</td>
</tr>
<tr>
<td>30</td>
<td>&lt;T₁ ABORT&gt;</td>
</tr>
<tr>
<td>40</td>
<td>&lt;CLR: Undo T₁ LSN 10&gt;</td>
</tr>
<tr>
<td>45</td>
<td>&lt;T₁ TXN-END&gt;</td>
</tr>
</tbody>
</table>

prevLSNs
FULL EXAMPLE

<table>
<thead>
<tr>
<th>TIME</th>
<th>LSN</th>
<th>LOG</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td></td>
<td>&lt;CHECKPOINT-BEGIN&gt;</td>
</tr>
<tr>
<td>05</td>
<td></td>
<td>&lt;CHECKPOINT-END&gt;</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>&lt;T₁, A→P₅, 1, 2&gt;</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>&lt;T₂, B→P₃, 2, 3&gt;</td>
</tr>
<tr>
<td>30</td>
<td></td>
<td>&lt;T₁ ABORT&gt;</td>
</tr>
<tr>
<td>40</td>
<td></td>
<td>&lt;CLR: Undo T₁ LSN 10&gt;</td>
</tr>
<tr>
<td>45</td>
<td></td>
<td>&lt;T₁ TXN-END&gt;</td>
</tr>
<tr>
<td>50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>60</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
FULL EXAMPLE

<table>
<thead>
<tr>
<th>TIME</th>
<th>LSN</th>
<th>LOG</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td></td>
<td>&lt;CHECKPOINT-BEGIN&gt;</td>
</tr>
<tr>
<td>05</td>
<td></td>
<td>&lt;CHECKPOINT-END&gt;</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>&lt;T₁, A→P₅, 1, 2&gt;</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>&lt;T₂, B→P₃, 2, 3&gt;</td>
</tr>
<tr>
<td>30</td>
<td></td>
<td>&lt;T₁ ABORT&gt;</td>
</tr>
<tr>
<td>40</td>
<td></td>
<td>&lt;CLR: Undo T₁ LSN 10&gt;</td>
</tr>
<tr>
<td>45</td>
<td></td>
<td>&lt;T₁ TXN-END&gt;</td>
</tr>
<tr>
<td>50</td>
<td></td>
<td>&lt;T₃, C→P₁, 4, 5&gt;</td>
</tr>
<tr>
<td>60</td>
<td></td>
<td>&lt;T₂, D→P₅, 6, 7&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X CRASH!</td>
</tr>
</tbody>
</table>
FULL EXAMPLE

<table>
<thead>
<tr>
<th>LSN</th>
<th>LOG</th>
</tr>
</thead>
<tbody>
<tr>
<td>00,05</td>
<td>&lt;CHECKPOINT-BEGIN&gt;, &lt;CHECKPOINT-END&gt;</td>
</tr>
<tr>
<td>10</td>
<td>&lt;T₁, A→P₅, 1, 2&gt;</td>
</tr>
<tr>
<td>20</td>
<td>&lt;T₂, B→P₃, 2, 3&gt;</td>
</tr>
<tr>
<td>30</td>
<td>&lt;T₁ ABORT&gt;</td>
</tr>
<tr>
<td>40,45</td>
<td>&lt;CLR: Undo T₁ LSN 10&gt;, &lt;T₁ TXN-END&gt;</td>
</tr>
<tr>
<td>50</td>
<td>&lt;T₃, C→P₁, 4, 5&gt;</td>
</tr>
<tr>
<td>60</td>
<td>&lt;T₂, D→P₅, 6, 7&gt;</td>
</tr>
<tr>
<td></td>
<td><strong>X CRASH! RESTART!</strong></td>
</tr>
</tbody>
</table>

ATT

<table>
<thead>
<tr>
<th>TxnId</th>
<th>Status</th>
<th>lastLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₂</td>
<td>U</td>
<td>60</td>
</tr>
<tr>
<td>T₃</td>
<td>U</td>
<td>50</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

DPT

<table>
<thead>
<tr>
<th>PageId</th>
<th>recLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>P₁</td>
<td>50</td>
</tr>
<tr>
<td>P₃</td>
<td>08</td>
</tr>
<tr>
<td>P₅</td>
<td>10</td>
</tr>
</tbody>
</table>

flushedLSN
FULL EXAMPLE

**ATT**

<table>
<thead>
<tr>
<th>TxnId</th>
<th>Status</th>
<th>lastLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₂</td>
<td>U</td>
<td>60</td>
</tr>
<tr>
<td>T₃</td>
<td>U</td>
<td>50</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

**DPT**

<table>
<thead>
<tr>
<th>PageId</th>
<th>recLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>P₁</td>
<td>50</td>
</tr>
<tr>
<td>P₃</td>
<td>08</td>
</tr>
<tr>
<td>P₅</td>
<td>10</td>
</tr>
</tbody>
</table>

**LSN**

- 00,05 <CHECKPOINT-BEGIN>, <CHECKPOINT-END>
- 10 <T₁, A→P₅, 1, 2>
- 20 <T₂, B→P₃, 2, 3>
- 30 <T₁ ABORT>
- 40,45 <CLR: Undo T₁ LSN 10>, <T₁ TXN-END>
- 50 <T₃, C→P₁, 4, 5>
- 60 <T₂, D→P₅, 6, 7>

**LOG**

CRASH! RESTART!
**FULL EXAMPLE**

### LSN

<table>
<thead>
<tr>
<th>LSN</th>
<th>LOG</th>
</tr>
</thead>
<tbody>
<tr>
<td>00,05</td>
<td>&lt;CHECKPOINT-BEGIN&gt;, &lt;CHECKPOINT-END&gt;</td>
</tr>
<tr>
<td>10</td>
<td>&lt;T₁, A→P₅, 1, 2&gt;</td>
</tr>
<tr>
<td>20</td>
<td>&lt;T₂, B→P₃, 2, 3&gt;</td>
</tr>
<tr>
<td>30</td>
<td>&lt;T₁ ABORT&gt;</td>
</tr>
<tr>
<td>40,45</td>
<td>&lt;CLR: Undo T₁ LSN 10&gt;, &lt;T₁ TXN-END&gt;</td>
</tr>
<tr>
<td>50</td>
<td>&lt;T₃, C→P₁, 4, 5&gt;</td>
</tr>
<tr>
<td>60</td>
<td>&lt;T₂, D→P₅, 6, 7&gt;</td>
</tr>
<tr>
<td>70</td>
<td>&lt;CLR: Undo T₂ LSN 60, UndoNext 20&gt;</td>
</tr>
</tbody>
</table>

### ATT

<table>
<thead>
<tr>
<th>TxnId</th>
<th>Status</th>
<th>lastLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₂</td>
<td>U</td>
<td>60</td>
</tr>
<tr>
<td>T₃</td>
<td>U</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### DPT

<table>
<thead>
<tr>
<th>PageId</th>
<th>recLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>P₁</td>
<td>50</td>
</tr>
<tr>
<td>P₃</td>
<td>08</td>
</tr>
<tr>
<td>P₅</td>
<td>10</td>
</tr>
</tbody>
</table>

**CRASH! RESTART!**

flushedLSN
FULL EXAMPLE

ATT

<table>
<thead>
<tr>
<th>TxnId</th>
<th>Status</th>
<th>lastLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₂</td>
<td>U</td>
<td>60</td>
</tr>
<tr>
<td>T₃</td>
<td>U</td>
<td>50</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

DPT

<table>
<thead>
<tr>
<th>PageId</th>
<th>recLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>P₁</td>
<td>50</td>
</tr>
<tr>
<td>P₃</td>
<td>08</td>
</tr>
<tr>
<td>P₅</td>
<td>10</td>
</tr>
</tbody>
</table>

LSN

00,05  <CHECKPOINT-BEGIN>, <CHECKPOINT-END>
10     <T₁, A→P₅, 1, 2>
20     <T₂, B→P₃, 2, 3>
30     <T₁ ABORT>
40,45  <CLR: Undo T₁ LSN 10>, <T₁ TXN-END>
50     <T₃, C→P₁, 4, 5>
60     <T₂, D→P₅, 6, 7>
70     <CLR: Undo T₂ LSN 60, UndoNext 20>

LOG

<CLR> Undo Next 20

CRASH! RESTART!
**FULL EXAMPLE**

<table>
<thead>
<tr>
<th>ATT</th>
<th>DPT</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TxnId</strong></td>
<td><strong>Status</strong></td>
</tr>
<tr>
<td>T₂</td>
<td>U</td>
</tr>
<tr>
<td>T₃</td>
<td>U</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

**LSN**

00,05  <CHECKPOINT-BEGIN>, <CHECKPOINT-END>
10  <T₁, A→P₅, 1, 2>
20  <T₂, B→P₃, 2, 3>
30  <T₁ ABORT>
40,45  <CLR: Undo T₁ LSN 10>, <T₁ TXN-END>
50  <T₃, C→P₁, 4, 5>
60  <T₂, D→P₅, 6, 7>

**CRASH! RESTART!**

70  <CLR: Undo T₂ LSN 60, UndoNext 20>
80,85  <CLR: Undo T₃ LSN 50>, <T₃ TXN-END>
FULL EXAMPLE

LSN  LOG

00,05  \langle CHECKPOINT-BEGIN\rangle, \langle CHECKPOINT-END\rangle

10  \langle T_1, A\rightarrow P_5, 1, 2\rangle

20  \langle T_2, B\rightarrow P_3, 2, 3\rangle

30  \langle T_1 \text{ ABORT} \rangle

40,45  \langle CLR: \text{ Undo } T_1 \text{ LSN } 10\rangle, \langle T_1 \text{ TXN-END} \rangle

50  \langle T_3, C\rightarrow P_1, 4, 5\rangle

60  \langle T_2, D\rightarrow P_5, 6, 7\rangle

\text{ CRASH! RESTART!}

Flush dirty pages + WAL to disk!

70  \langle CLR: \text{ Undo } T_2 \text{ LSN } 60, \text{ UndoNext} \rangle

80,85  \langle CLR: \text{ Undo } T_3 \text{ LSN } 50\rangle, \langle T_3 \text{ TXN-END} \rangle
FULL EXAMPLE

ATT

<table>
<thead>
<tr>
<th>TxnId</th>
<th>Status</th>
<th>lastLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>T2</td>
<td>U</td>
<td>60</td>
</tr>
<tr>
<td>T3</td>
<td>U</td>
<td>50</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

DPT

<table>
<thead>
<tr>
<th>PageId</th>
<th>recLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>50</td>
</tr>
<tr>
<td>P3</td>
<td>08</td>
</tr>
<tr>
<td>P5</td>
<td>10</td>
</tr>
</tbody>
</table>

LSN  LOG

00,05  <CHECKPOINT-BEGIN>, <CHECKPOINT-END>
10      <T₁, A→P₅, 1, 2>
20      <T₂, B→P₃, 2, 3>
30      <T₁ ABORT>
40,45   <CLR: Undo T₁ LSN 10>, <T₁ TXN-END>
50      <T₃, C→P₁, 4, 5>
60      <T₂, D→P₅, 6, 7>

Flush dirty pages + WAL to disk!

CRASH! RESTART!

70      <CLR: Undo T₂ LSN 60, UndoNext>
80,85   <CLR: Undo T₃ LSN 50>, <T₃ TXN-END>

CRASH! RESTART!
FULL EXAMPLE

<table>
<thead>
<tr>
<th>LSN</th>
<th>LOG</th>
</tr>
</thead>
<tbody>
<tr>
<td>00, 05</td>
<td>&lt;CHECKPOINT-BEGIN&gt;, &lt;CHECKPOINT-END&gt;</td>
</tr>
<tr>
<td>10</td>
<td>&lt;T₁, A→P₅, 1, 2&gt;</td>
</tr>
<tr>
<td>20</td>
<td>&lt;T₂, B→P₃, 2, 3&gt;</td>
</tr>
<tr>
<td>30</td>
<td>&lt;T₁ abort&gt;</td>
</tr>
<tr>
<td>40, 45</td>
<td>&lt;CLR: Undo T₁ LSN 10&gt;, &lt;T₁ TXN-END&gt;</td>
</tr>
<tr>
<td>50</td>
<td>&lt;T₃, C→P₁, 4, 5&gt;</td>
</tr>
<tr>
<td>60</td>
<td>&lt;T₂, D→P₅, 6, 7&gt;</td>
</tr>
<tr>
<td>70</td>
<td>&lt;CLR: Undo T₂ LSN 60, UndoNext</td>
</tr>
<tr>
<td>80, 85</td>
<td>&lt;CLR: Undo T₃ LSN 50&gt;, &lt;T₃ TXN-END&gt;</td>
</tr>
</tbody>
</table>

Flush dirty pages + WAL to disk!
**FULL EXAMPLE**

**ATT**

<table>
<thead>
<tr>
<th>TxnId</th>
<th>Status</th>
<th>lastLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₂</td>
<td>U</td>
<td>70</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

**DPT**

<table>
<thead>
<tr>
<th>PageId</th>
<th>recLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>P₁</td>
<td>50</td>
</tr>
<tr>
<td>P₃</td>
<td>08</td>
</tr>
<tr>
<td>P₅</td>
<td>10</td>
</tr>
</tbody>
</table>

**LSN**

- 00,05: `<CHECKPOINT-BEGIN>, <CHECKPOINT-END>`
- 10: `<T₁, A→P₅, 1, 2>`
- 20: `<T₂, B→P₃, 2, 3>`
- 30: `<T₁ ABORT>`
- 40,45: `<CLR: Undo T₁ LSN 10>, <T₁ TXN-END>`
- 50: `<T₃, C→P₁, 4, 5>`
- 60: `<T₂, D→P₅, 6, 7>`

**LOG**

- CRASH! RESTART!
- 70: `<CLR: Undo T₂ LSN 60, UndoNext 20>`
- 80,85: `<CLR: Undo T₃ LSN 50>, <T₃ TXN-END>`

**flushedLSN**
FULL EXAMPLE

**ATT**

<table>
<thead>
<tr>
<th>TxnId</th>
<th>Status</th>
<th>lastLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>T_2</td>
<td>U</td>
<td>70</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

**DPT**

<table>
<thead>
<tr>
<th>PageId</th>
<th>recLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>P_1</td>
<td>50</td>
</tr>
<tr>
<td>P_3</td>
<td>08</td>
</tr>
<tr>
<td>P_5</td>
<td>10</td>
</tr>
</tbody>
</table>

**LSN**

- 00,05: <CHECKPOINT-BEGIN>, <CHECKPOINT-END>
- 10: <T_1, A→P_5, 1, 2>
- 20: <T_2, B→P_3, 2, 3>
- 30: <T_1 ABORT>
- 40,45: <CLR: Undo T_1 LSN 10>, <T_1 TXN-END>
- 50: <T_3, C→P_1, 4, 5>
- 60: <T_2, D→P_5, 6, 7>

**LOG**

- CRASH! RESTART!
- 70: <CLR: Undo T_2 LSN 60, UndoNext 20>
- 80,85: <CLR: Undo T_3 LSN 50>, <T_3 TXN-END>

**flushedLSN**

- CRASH! RESTART!
**FULL EXAMPLE**

<table>
<thead>
<tr>
<th>LSN</th>
<th>LOG</th>
</tr>
</thead>
<tbody>
<tr>
<td>00,05</td>
<td>&lt;CHECKPOINT-BEGIN&gt;, &lt;CHECKPOINT-END&gt;</td>
</tr>
<tr>
<td>10</td>
<td>&lt;T₁, A→P₅, 1, 2&gt;</td>
</tr>
<tr>
<td>20</td>
<td>&lt;T₂, B→P₃, 2, 3&gt;</td>
</tr>
<tr>
<td>30</td>
<td>&lt;T₁ ABORT&gt;</td>
</tr>
<tr>
<td>40,45</td>
<td>&lt;CLR: Undo T₁ LSN 10&gt;, &lt;T₁ TXN-END&gt;</td>
</tr>
<tr>
<td>50</td>
<td>&lt;T₃, C→P₁, 4, 5&gt;</td>
</tr>
<tr>
<td>60</td>
<td>&lt;T₂, D→P₅, 6, 7&gt;</td>
</tr>
<tr>
<td>70</td>
<td>× CRASH! RESTART!</td>
</tr>
<tr>
<td>80,85</td>
<td>&lt;CLR: Undo T₂ LSN 60, UndoNext 20&gt;</td>
</tr>
<tr>
<td>50</td>
<td>&lt;CLR: Undo T₃ LSN 50&gt;, &lt;T₃ TXN-END&gt;</td>
</tr>
<tr>
<td>70</td>
<td>× CRASH! RESTART!</td>
</tr>
</tbody>
</table>

**ATT**

<table>
<thead>
<tr>
<th>TxnId</th>
<th>Status</th>
<th>lastLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₂</td>
<td>U</td>
<td>70</td>
</tr>
</tbody>
</table>

**DPT**

<table>
<thead>
<tr>
<th>PageId</th>
<th>recLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>P₁</td>
<td>50</td>
</tr>
<tr>
<td>P₃</td>
<td>08</td>
</tr>
<tr>
<td>P₅</td>
<td>10</td>
</tr>
</tbody>
</table>

*flushedLSN*
FULL EXAMPLE

ATT

<table>
<thead>
<tr>
<th>TxnId</th>
<th>Status</th>
<th>lastLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>T2</td>
<td>U</td>
<td>70</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

DPT

<table>
<thead>
<tr>
<th>PageId</th>
<th>recLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>50</td>
</tr>
<tr>
<td>P3</td>
<td>08</td>
</tr>
<tr>
<td>P5</td>
<td>10</td>
</tr>
</tbody>
</table>

LSN

00,05  <CHECKPOINT-BEGIN>, <CHECKPOINT-END>
10     <T1, A→P5, 1, 2>
20     <T2, B→P3, 2, 3>
30     <T1 ABORT>
40,45  <CLR: Undo T1 LSN 10>, <T1 TXN-END>
50     <T3, C→P1, 4, 5>
60     <T2, D→P5, 6, 7>

CRASH! RESTART!

70     <CLR: Undo T2 LSN 60, UndoNext 20>
80,85  <CLR: Undo T3 LSN 50>, <T3 TXN-END>

CRASH! RESTART!
FULL EXAMPLE

ATT

<table>
<thead>
<tr>
<th>TxnId</th>
<th>Status</th>
<th>lastLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₂</td>
<td>U</td>
<td>70</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

DPT

<table>
<thead>
<tr>
<th>PageId</th>
<th>recLSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>P₁</td>
<td>50</td>
</tr>
<tr>
<td>P₃</td>
<td>08</td>
</tr>
<tr>
<td>P₅</td>
<td>10</td>
</tr>
</tbody>
</table>

LSN

00,05  <CHECKPOINT-BEGIN>, <CHECKPOINT-END>

10     <T₁, A→P₅, 1, 2>

20     <T₂, B→P₃, 2, 3>

30     <T₁ ABORT>

40,45  <CLR: Undo T₁ LSN 10>, <T₁ TXN-END>

50     <T₃, C→P₁, 4, 5>

60     <T₂, D→P₅, 6, 7>

CRASH! RESTART!

70     <CLR: Undo T₂ LSN 60, UndoNext 20>

80,85  <CLR: Undo T₃ LSN 50>, <T₃ TXN-END>

90,95  <CLR: Undo T₂ LSN 20>, <T₂ TXN-END>

LOG

X CRASH! RESTART!

flushedLSN
ADDITIONAL CRASH ISSUES (1)

What does the DBMS do if it crashes during recovery in the Analysis Phase?

What does the DBMS do if it crashes during recovery in the Redo Phase?
What does the DBMS do if it crashes during recovery in the Analysis Phase?
→ Nothing. Just run recovery again.

What does the DBMS do if it crashes during recovery in the Redo Phase?
What does the DBMS do if it crashes during recovery in the Analysis Phase?
→ Nothing. Just run recovery again.

What does the DBMS do if it crashes during recovery in the Redo Phase?
→ Again nothing. Redo everything again.
How can the DBMS improve performance during recovery in the Redo Phase?

How can the DBMS improve performance during recovery in the Undo Phase?
How can the DBMS improve performance during recovery in the Redo Phase?
→ Assume that it is not going to crash again and flush all changes to disk asynchronously in the background.

How can the DBMS improve performance during recovery in the Undo Phase?
How can the DBMS improve performance during recovery in the Redo Phase?
→ Assume that it is not going to crash again and flush all changes to disk asynchronously in the background.

How can the DBMS improve performance during recovery in the Undo Phase?
→ Lazily rollback changes before new txns access pages.
→ Rewrite the application to avoid long-running txns.
CONCLUSION

Mains ideas of ARIES:
→ WAL with STEAL/NO-FORCE
→ Fuzzy Checkpoints (snapshot of dirty page ids)
→ Redo everything since the earliest dirty page
→ Undo txns that never commit
→ Write CLRs when undoing, to survive failures during restarts

Log Sequence Numbers:
→ LSNs identify log records; linked into backwards chains per transaction via prevLSN.
→ pageLSN allows comparison of data page and log records.
You now know how to build a single-node DBMS.

So now we can talk about distributed databases!