CSE 444: Database Internals

Section 9:

Transactions – Recovery with ARIES

Review in this section

Recovery for ARIES

Follows homework closely

ARIES

- A popular protocol for UNDO-REDO logging
- **Steal (like UNDO)**
  - Changes by uncommitted transactions can be written to disk when a dirty page is flushed
- **No-force (like REDO)**
  - Changes by committed transactions may not have been written to disk
- **Write-ahead logging:**
  - Any changes to a database object is first recorded in the log, and the log is written to disk, before the change to database object is written to disk
  - A record of every change to the database is available while recovering from a crash

ARIES Data Structures

Dirty page table

pageID	recLSN

Transaction table

transID	lastLSN	status

LSN

101

Log

prevLSN	tID	pID	Log entry	Type	undoNextLSN

Log Record “Types”

- **Update**: easy
- **Commit**: log-tail forced-written to disk, up to & including commit
  (note that still no-force, the actual modified pages may not be written, and much smaller cost)
- **Abort**: abort type log record is written + undo is initiated for this transaction
- **End**: when a transaction is aborted or committed, some additional actions are performed, after that an end record is written
- **CLR:**
  Undoing updates (during abort or recovery from crash),
  for every update record undone, write a CLR (Compensation Log Record)

Example.

1. T_{1000} changes the value of **A from "abc" to "def"** on **page P500**
2. T_{2000} changes the value of **B from "hij" to "klm"** on **page P600**
3. T_{2000} changes the value of **D from "mnp" to "qrs"** on **page P500**
4. T_{1000} changes the value of **C from "tuv" to "wxy"** on **page P505**
5. T_{2000} commits and the end log record is written
6. T_{1000} changes the value of **E from "pq" to "rs"** on **page P700**
7. **P600** is flushed to disk
8. **Crash!!**

Same as in Section 6

Example is adopted from Ramakrishnan-Gehrke book

ARIES Data Structures

Dirty page table

pageID	recLSN

Transaction table

transID	lastLSN	status

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101						

Buffer Pool

PageLSN= -

A = abc D = mnp

B = hij

P505
PageLSN= -

C = tuv

P700
PageLSN= -

E = pq

Disk

PageLSN= -

A = abc D = mnp

B = hij

P505
PageLSN= -

C = tuv

P700
PageLSN= -

E = pq

First operation:

1. T_{1000} changes the value of **A** **from "abc" to "def"** on **page P500?**

Dirty page table

pageID	recLSN

LSN

101

Log

prevLSN	tID	pID	Log entry	Type	undoNextLSN

Transaction table

transID	lastLSN	status

Buffer Pool

PageLSN= -

A = abc D = mnp

B = hij

P505
PageLSN= -

C = tuv

P700
PageLSN= -

E = pq

Disk

PageLSN= -

A = abc D = mnp

B = hij

P505
PageLSN= -

C = tuv

P700
PageLSN= -

E = pq

Changes

1. T_{1000} changes the value of **A** from "abc" to "def" on page P500

Dirty page table

pageID	recLSN
P500	101

Transaction table

transID	lastLSN	status
T_{1000}	101	Running

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A "abc" -> "def"	Update	-

Buffer Pool

PageLSN= 101

A = def D = mnp B = hij

P505
PageLSN= -
C = tuv

P700
PageLSN= -
E = pq

Disk

PageLSN= -

A = abc D = mnp B = hij

P505
PageLSN= -
C = tuv

P700
PageLSN= -
E = pq

Next:

2. T_{2000} changes the value of B from "hij" to "klm" on page P600 ?

Dirty page table

pageID	recLSN
P500	101

Transaction table

transID	lastLSN	status
T_{1000}	101	Running

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A "abc" -> "def"	Update	-

Buffer Pool

PageLSN= 101

A = def D = mnp

B = hij

P505
PageLSN= -

C = tuv

P700
PageLSN= -

E = pq

Disk

PageLSN= -

A = abc D = mnp

B = hij

P505
PageLSN= -

C = tuv

P700
PageLSN= -

E = pq

Changes:

2. T_{2000} changes the value of B from "hij" to "klm" on page P600 ?

Dirty page table

pageID	recLSN
P500	101
P600	**102**

Transaction table

transID	lastLSN	status
T_{1000}	101	Running
T_{2000}	**102**	**Running**

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A "abc" -> "def"	Update	-
102	-	T_{2000}	P600	Write B "hij" -> "klm"		

Buffer Pool

PageLSN= 101

A = def D = mnp

P505
PageLSN= -
C = tuv

PageLSN= 102

B = klm

P700
PageLSN= -
E = pq

Disk

PageLSN= -

A = abc D = mnp

P505
PageLSN= -
C = tuv

B = hij

P700
PageLSN= -
E = pq

Next:

3. T_{2000} changes the value of D from "mnp" to "qrs" on page P500?

Dirty page table

pageID	recLSN
P500	101
P600	**102**

Transaction table

transID	lastLSN	status
T_{1000}	101	Running
T_{2000}	**102**	**Running**

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A "abc" -> "def"	Update	-
102	-	T_{2000}	P600	Write B "hij" -> "klm"		

Buffer Pool

PageLSN= 101

A = def D = mnp

P505
PageLSN= -

C = tuv

PageLSN= 102

B = klm

P700
PageLSN= -

E = pq

Disk

PageLSN= -

A = abc D = mnp

B = hij

P505
PageLSN= -

C = tuv

P700
PageLSN= -

E = pq

Changes:

3. T_{2000} changes the value of D from "mnp" to "qrs" on page P500

Dirty page table

pageID	recLSN
P500	101
P600	102

Transaction table

transID	lastLSN	status
T_{1000}	101	Running
T_{2000}	**103**	**Running**

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A "abc" -> "def"	Update	-
102	-	T_{2000}	P600	Write B "hij" -> "klm"	Update	-
103	102	T_{2000}	P500	Write D "mnp" -> "qrs"	Update	-

Buffer Pool

PageLSN= 103

A = def **D = qrs**

P505
PageLSN= -

C = tuv

PageLSN= 102

B = klm

P700
PageLSN= -

E = pq

Disk

PageLSN= -

A = abc D = mnp

P505
PageLSN= -

C = tuv

B = hij

P700
PageLSN= -

E = pq

Next:

4. T_{1000} changes the value of **C from "tuv" to "wxy"** on **page P505?**

Dirty page table

pageID	recLSN
P500	101
P600	102

Transaction table

transID	lastLSN	status
T_{1000}	101	Running
T_{2000}	**103**	**Running**

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A "abc" -> "def"	Update	-
102	-	T_{2000}	P600	Write B "hij" -> "klm"	Update	-
103	102	T_{2000}	P500	Write D "mnp" -> "qrs"	Update	-

Buffer Pool

PageLSN= 103 PageLSN= 102

A = def **D = qrs** B = klm

P505 PageLSN= -

C = tuv

P700 PageLSN= -

E = pq

Disk

PageLSN= -

A = abc D = mnp B = hij

P505 PageLSN= -

C = tuv

P700 PageLSN= -

E = pq

Changes:

4. T_{1000} changes the value of C from "tuv" to "wxy" on page P505?

Dirty page table

pageID	recLSN
P500	101
P600	102
P505	**104**

Transaction table

transID	lastLSN	status
T_{1000}	**104**	**Running**
T_{2000}	103	Running

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A "abc" -> "def"	Update	-
102	-	T_{2000}	P600	Write B "hij" -> "klm"	Update	-
103	102	T_{2000}	P500	Write D "mnp" -> "qrs"	Update	-
104	101	T_{1000}	P505	Write C "tuv" -> "wxy"	Update	-

Buffer Pool

PageLSN= 103 PageLSN= 102

A = def D = qrs B = klm

P505
PageLSN= 104
C = wxy

P700
PageLSN= -
E = pq

Disk

PageLSN= -

A = abc D = mnp B = hij

P505
PageLSN= -
C = tuv

P700
PageLSN= -
E = pq

Next:

5. T_{2000} commits and the end log record is written

Dirty page table

pageID	recLSN
P500	101
P600	102
P505	**104**

Transaction table

transID	lastLSN	status
T_{1000}	104	Running
T_{2000}	103	Running

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102	-	T_{2000}	P600	Write B “hij” -> “klm”	Update	-
103	102	T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-

Buffer Pool

PageLSN= 103 PageLSN= 102

A = def D = qrs B = klm

P505
PageLSN= 104
C = wxy

P700
PageLSN= -
E = pq

Disk

PageLSN= -

A = abc D = mnp B = hij

P505
PageLSN= -
C = tuv

P700
PageLSN= -
E = pq

Changes:

5. T_{2000} commits and the end log record is written --- step 1

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102	-	T_{2000}	P600	Write B “hij” -> “klm”	Update	-
103	102	T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	

Dirty page table

pageID	recLSN
P500	101
P600	102
P505	104

Transaction table

transID	lastLSN	status
T_{1000}	104	Running
T_{2000}	103	Committed

Buffer Pool

PageLSN= 103

A = def D = qrs

P505
PageLSN= 104

C = wxy

PageLSN= 102

B = klm

P700
PageLSN= -

E = pq

Disk

PageLSN= -

A = abc D = mnp

P505
PageLSN= -

C = tuv

B = hij

P700
PageLSN= -

E = pq

Changes:

5. T_{2000} commits and the end log record is written --- step 2

Log

Dirty page table

pageID	recLSN
P500	101
P600	102
P505	104

T2000 removed from transaction table

Transaction tabl

transID	lastLSN	status
T_{1000}	104	Running
~~T_{2000}~~	~~103~~	~~Committed~~

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102			P600	Write B “hij” -> “klm”	Update	-
103		T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	

Buffer Pool

PageLSN= 103 PageLSN= 102

A = def D = qrs B = klm

P505
PageLSN= 104

C = wxy

P700
PageLSN= -

E = pq

Disk

PageLSN= -

A = abc D = mnp B = hij

P505
PageLSN= -

C = tuv

P700
PageLSN= -

E = pq

Changes:

5. T_{2000} commits and the end log record is written --- step 2

Dirty page table

pageID	recLSN
P500	101
P600	102
P505	104

Transaction table

transID	lastLSN	status
T_{1000}	104	Running
T_{2000}	103	Committed

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A "abc" -> "def"	Update	-
102	-	T_{2000}	P600	Write B "hij" -> "klm"	Update	-
103	102	T_{2000}	P500	Write D "mnp" -> "qrs"	Update	-
104	101	T_{1000}	P505	Write C "tuv" -> "wxy"	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	

Buffer Pool

PageLSN= 103 PageLSN= 102

A = def D = qrs B = klm

P505
PageLSN= 104

C = wxy

P700
PageLSN= -

E = pq

Disk

Log written to disk
Note: no force = not the dirty pages changed by T_{2000}!

A = abc D = mnp B = hij

P505
PageLSN= -

C = tuv

P700
PageLSN= -

E = pq

Whenever a transaction commits,
log is flushed to the disk == the log-tail is written to disk

NOTE:

1. The “Commit” record is required to be flushed (i.e. all logs up to and including that commit record)
2. The “End” record is not required to be flushed, in this case we are only assuming that it has been flushed as well (so that we have a good example while doing recovery ☺)

Next:

6. T_{1000} changes the value of E from "pq" to "rs" on page P700

Dirty page table

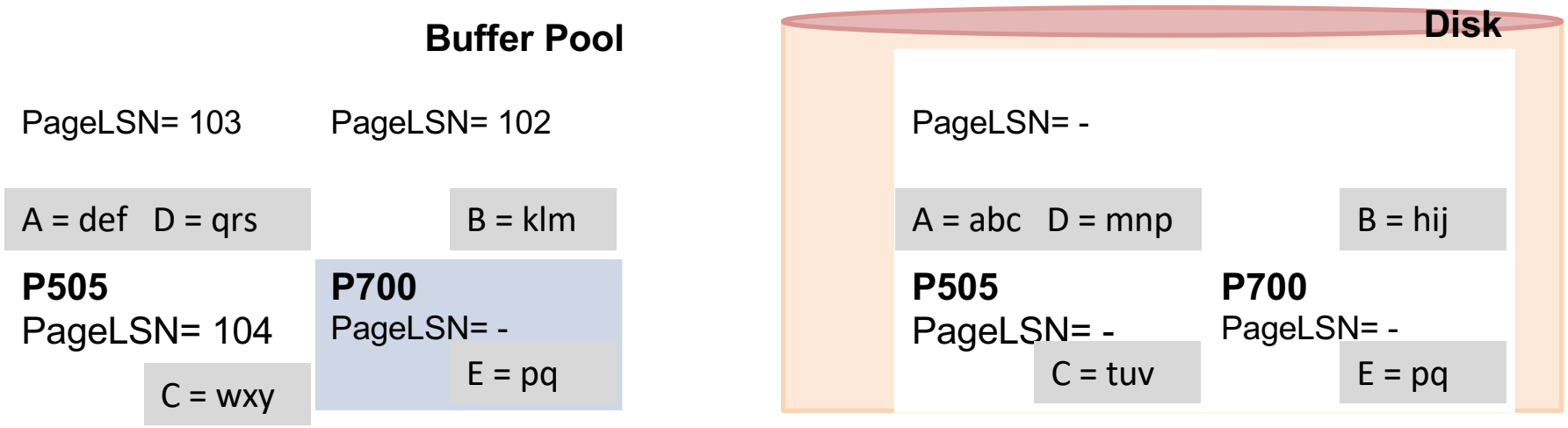
pageID	recLSN
P500	101
P600	102
P505	104

Transaction table

transID	lastLSN	status
T_{1000}	104	Running

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A "abc" -> "def"	Update	-
102	-	T_{2000}	P600	Write B "hij" -> "klm"	Update	-
103	102	T_{2000}	P500	Write D "mnp" -> "qrs"	Update	-
104	101	T_{1000}	P505	Write C "tuv" -> "wxy"	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	



Changes:

6. T_{1000} changes the value of E from "pq" to "rs" on page P700

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A "abc" -> "def"	Update	-
102	-	T_{2000}	P600	Write B "hij" -> "klm"	Update	-
103	102	T_{2000}	P500	Write D "mnp" -> "qrs"	Update	-
104	101	T_{1000}	P505	Write C "tuv" -> "wxy"	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	
107	104	T_{1000}	P700	Write E "pq" -> "rs"	Update	-

Dirty page table

pageID	recLSN
P500	101
P600	102
P505	104
P700	107

Transaction table

transID	lastLSN	status
T_{1000}	107	Running

Buffer Pool

PageLSN= 103

A = def D = qrs

P505
PageLSN= 104

C = wxy

PageLSN= 102

B = klm

P700
PageLSN= 107

E = rs

Disk

PageLSN= -

A = abc D = mnp

P505
PageLSN= -

C = tuv

B = hij

P700
PageLSN= -

E = pq

Next:

7. Page P600 is flushed to disk

Dirty page table

pageID	recLSN
P500	101
P600	102
P505	104
P700	**107**

Transaction table

transID	lastLSN	status
T_{1000}	107	Running

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A "abc" -> "def"	Update	-
102	-	T_{2000}	P600	Write B "hij" -> "klm"	Update	-
103	102	T_{2000}	P500	Write D "mnp" -> "qrs"	Update	-
104	101	T_{1000}	P505	Write C "tuv" -> "wxy"	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	
107	104	T_{1000}	P700	Write E "pq" -> "rs"	Update	-

Buffer Pool

PageLSN= 103

A = def D = qrs

P505
PageLSN= 104

C = wxy

PageLSN= 102

B = klm

P700
PageLSN= 107

E = rs

Disk

PageLSN= -

A = abc D = mnp

P505
PageLSN= -

C = tuv

B = hij

P700
PageLSN= -

E = pq

Next:

7. Page P600 is flushed to disk – Step 1

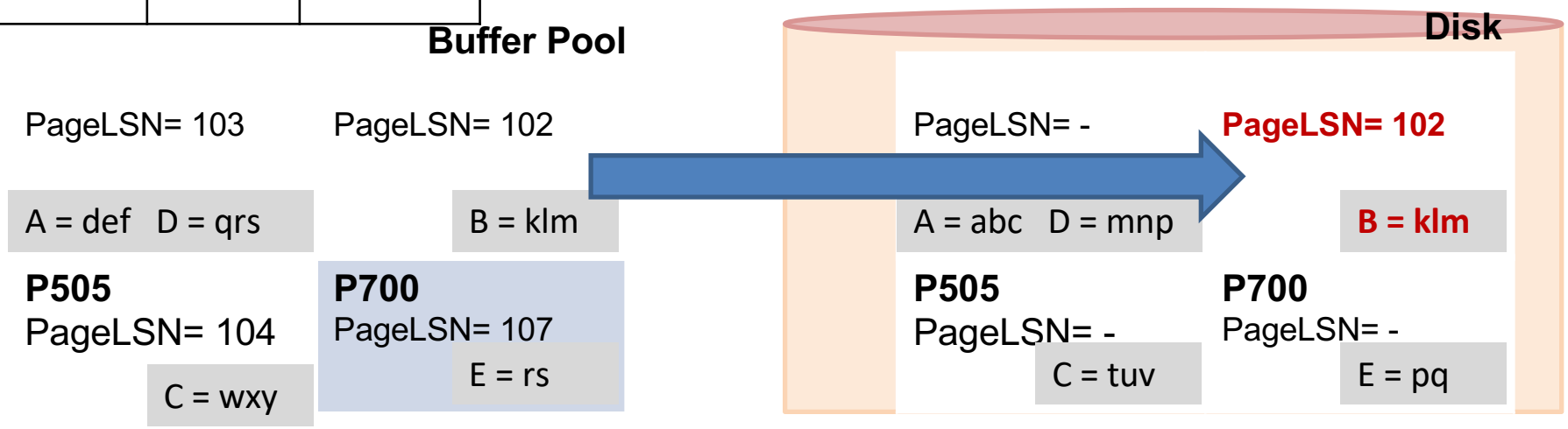
Dirty page table

pageID	recLSN
P500	101
~~P600~~	~~102~~
P505	104
P700	107

Transaction table

transID	lastLSN	status
T_{1000}	107	Running

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102	-	T_{2000}	P600	Write B “hij” -> “klm”	Update	-
103	102	T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	
107	104	T_{1000}	P700	Write E “pq” -> “rs”	Update	-



Next:

7. Page P600 is flushed to disk – Step 2

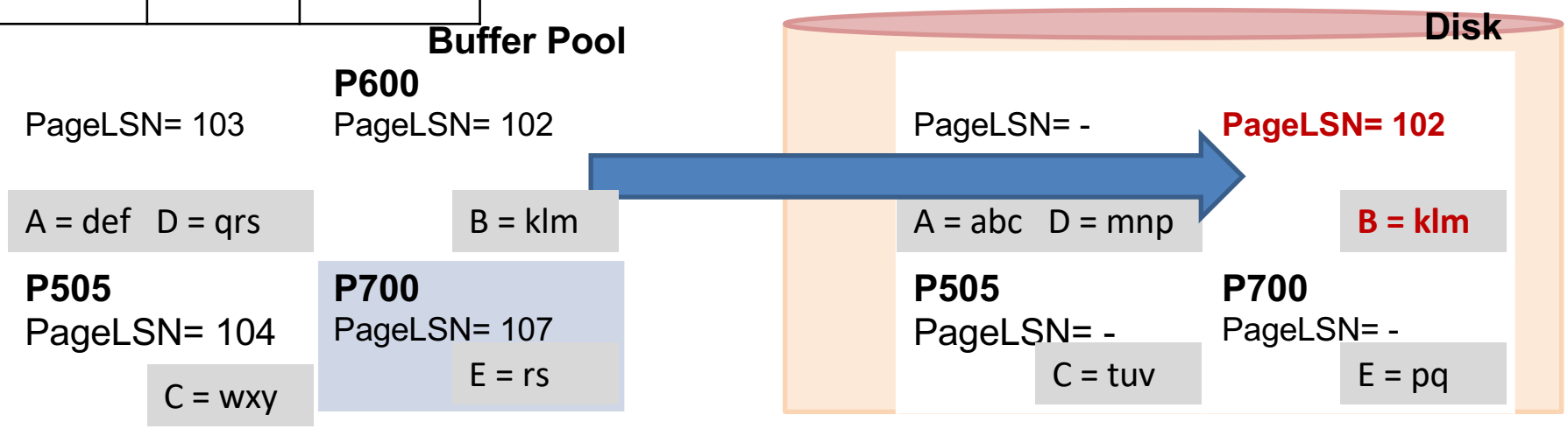
Dirty page table

pageID	recLSN
P500	101
P505	104
P700	107

Transaction table

transID	lastLSN	status
T_{1000}	107	Running

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102	-	T_{2000}	P600	Write B “hij” -> “klm”	Update	-
103	102	T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	
107	104	T_{1000}	P700	Write E “pq” -> “rs”	Update	-




8. CRASH!!

8. Crash!! ---- These are gone from memory

Dirty page table

P700	

Transaction table

transID	lastLSN	status
		ing

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102	-	T_{2000}	P600	Write B “hij” -> “klm”	Update	-
103	102	T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	
107						

Buffer Pool

PageLSN= 103

A = def

P505

PageLSN= 107

C = wxy

E = rs

Disk

PageLSN= -

A = abc D = mnp

B = klm

P505

PageLSN= -

C = tuv

P700

PageLSN= -

E = pq

Three Recovery Phases of ARIES

- Analysis
  - Reconstructs Dirty page table (for Redo) + Transaction Table (for Undo, active transactions at crash)
- Redo
  - Restores the database state at the time of crash by repeating Updates + CLR
- Undo
  - Undoes the actions of uncommitted transactions
  - Only updates can be undone, No CLR is ever undone!

Analysis Phase

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A "abc" -> "def"	Update	-
102	-	T_{2000}	P600	Write B "hij" -> "klm"	Update	-
103	102	T_{2000}	P500	Write D "mnp" -> "qrs"	Update	-
104	101	T_{1000}	P505	Write C "tuv" -> "wxy"	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	

Dirty page table

pageID	recLSN

Transaction table

transID	lastLSN	status

Disk

Buffer Pool

PageLSN= -

A = abc D = mnp

B = klm

P505
PageLSN= -

C = tuv

P700
PageLSN= -

E = pq

Will not show the buffer pool from the next slide

Analysis Phase

Dirty page table

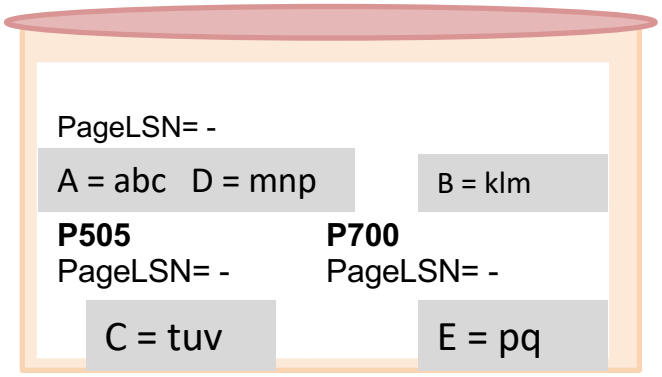
pageID	recLSN
P500	101

Transaction table

transID	lastLSN	status
T1000	101	U= Unknown

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102	-	T_{2000}	P600	Write B “hij” -> “klm”	Update	-
103	102	T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	



Disk

In hw5, you can also write “Running/In progress” instead of “Unknown”

Analysis Phase

Dirty page table

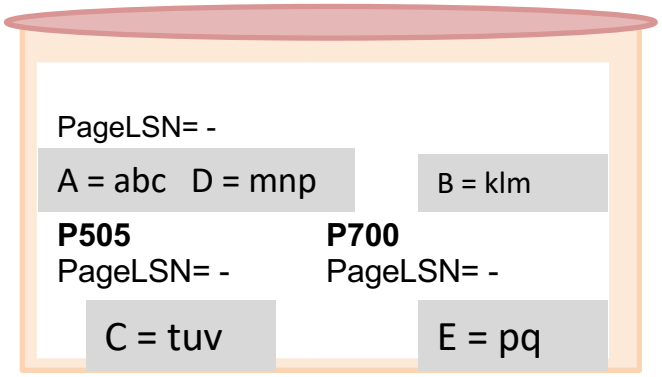
pageID	recLSN
P500	101
P600	**102**

Transaction table

transID	lastLSN	status
T1000	101	U
T2000	**102**	**U**

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102	-	T_{2000}	P600	Write B “hij” -> “klm”	Update	-
103	102	T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	



Disk

Analysis Phase

Dirty page table

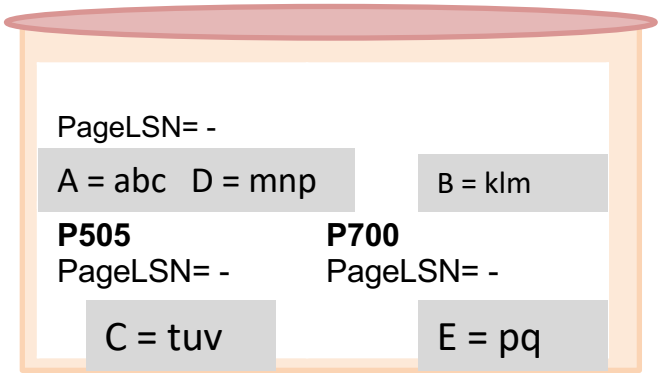
pageID	recLSN
P500	101
P600	102

Transaction table

transID	lastLSN	status
T1000	101	U
T2000	**103**	**U**

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102	-	T_{2000}	P600	Write B “hij” -> “klm”	Update	-
103	102	T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	



Disk

Analysis Phase

Dirty page table

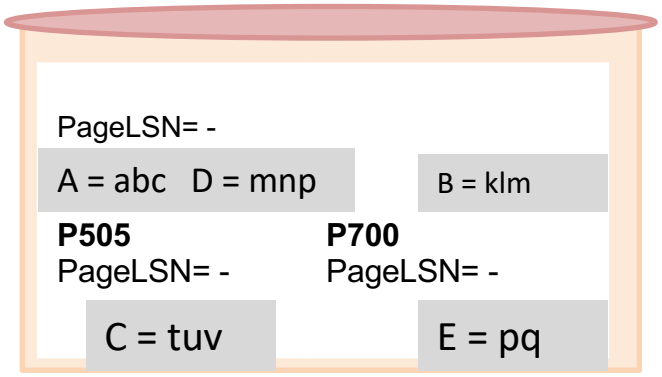
pageID	recLSN
P500	101
P600	102
P505	104

Transaction table

transID	lastLSN	status
T1000	104	U
T2000	103	U

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102	-	T_{2000}	P600	Write B “hij” -> “klm”	Update	-
103	102	T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	



Disk

Analysis Phase

Dirty page table

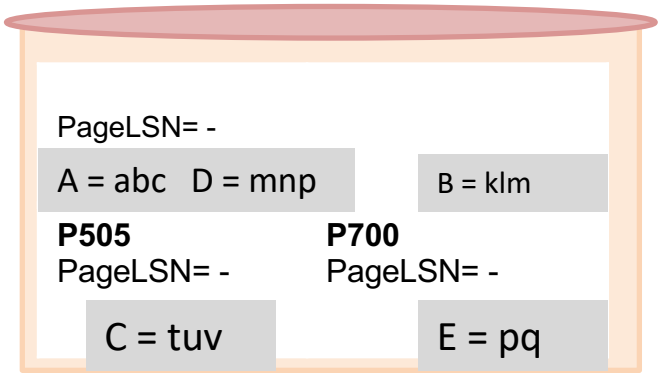
pageID	recLSN
P500	101
P600	102
P505	104

Transaction table

transID	lastLSN	status
T1000	104	U
T2000	**105**	**C**

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102	-	T_{2000}	P600	Write B “hij” -> “klm”	Update	-
103	102	T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	



Disk

Write A or Abort if you see an Abort log instead

Analysis Phase

Dirty page table

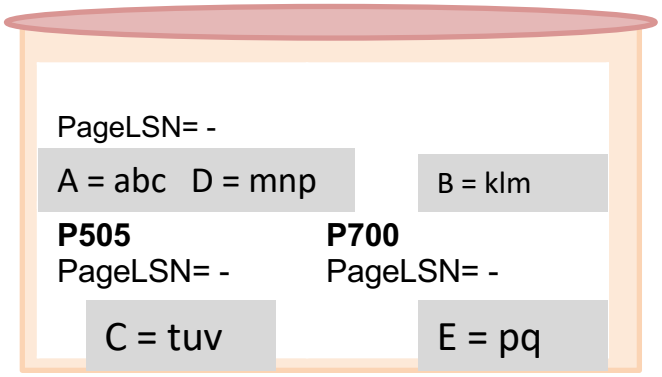
pageID	recLSN
P500	101
P600	102
P505	104

Transaction table

transID	lastLSN	status
T1000	104	U

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102	-	T_{2000}	P600	Write B “hij” -> “klm”	Update	-
103	102	T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	



Disk

Already written to disk, but reappears

Compare with Dirty Table and Transaction Table right before Crash!!

Dirty page table

pageID	recLSN
P500	101
P505	104
P700	107

Transaction table

transID	lastLSN	status
T_{1000}	107	Running

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102	-	T_{2000}	P600	Write B “hij” -> “klm”	Update	-
103	102	T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	
107	104	T_{1000}	P700	Write E “pq” -> “rs”	Update	-

Buffer Pool

PageLSN= 103

A = def D = qrs

P505
PageLSN= 104

C = tuv

P700
PageLSN= 107

E = rs

Lost update during crash, but write ahead log, so safe!

Disk

PageLSN= 102

B = klm

P505
PageLSN= -

C = tuv

P700
PageLSN= -

E = pq

Analysis Phase

Dirty page table

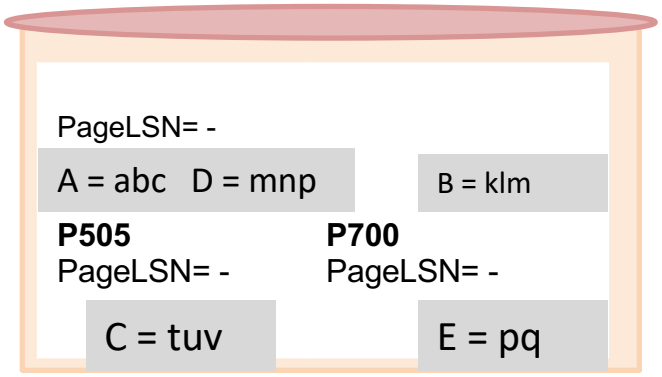
pageID	recLSN
P500	101
P600	102
P505	104

Transaction table

transID	lastLSN	status
T1000	104	U
~~T2000~~	~~105~~	~~C~~

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102	-	T_{2000}	P600	Write B “hij” -> “klm”	Update	-
103	102	T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	



Disk

Remove entry from Transaction Table if you see an End record (both for Aborted and Committed transactions)

REDO Phase

- "Repeating history" (including actions by transactions that would be aborted in the undo phase)
- Work with the Dirty Page Table
- Find the smallest recLSN in the dirty page table = FirstLSN
- Redo the "Update/CLR" action, unless (in this order)
  - Affected page is not in the dirty page table
  - Or, recLSN > LSN being checked (i.e. the page was dirtied later than this LSN)
  - Or, pageLSN >= LSN being checked (i.e. LSN still not at most recent change)
- End/Commit/Abort LSNs are "skipped"
- **In HW, write "Redone" or "Skipped" for each LSN**

Analysis Phase

Dirty page table

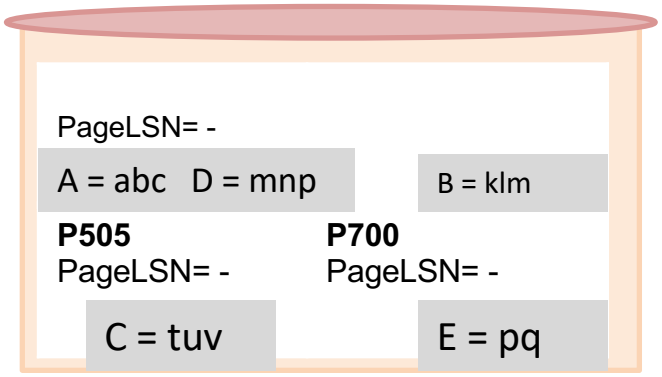
pageID	recLSN
P500	101
P600	102
P505	104

Transaction table

transID	lastLSN	status
T1000	104	U

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102	-	T_{2000}	P600	Write B “hij” -> “klm”	Update	-
103	102	T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	



Disk

REDO Phase: find firstLSN

Dirty page table

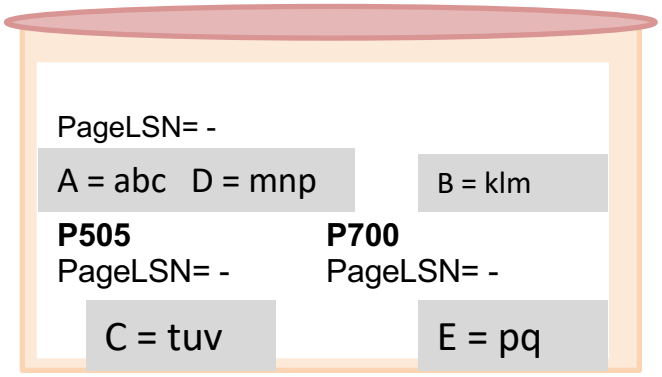
pageID	recLSN
P500	101
P600	102
P505	104

Transaction table

transID	lastLSN	status
T1000	104	U

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102	-	T_{2000}	P600	Write B “hij” -> “klm”	Update	-
103	102	T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	



Disk

REDO Phase

Dirty page table

pageID	recLSN
P500	101
P600	102
P505	104

Transaction table

transID	lastLSN	status
T1000	104	U

Buffer Pool

PageLSN= "-" to **101**

A = def D = mnp

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A "abc" -> "def"	Update	-
102	-	T_{2000}	P600	Write B "hij" -> "klm"	Update	-
103	102	T_{2000}	P500	Write D "mnp" -> "qrs"	Update	-
104	101	T_{1000}	P505	Write C "tuv" -> "wxy"	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	

Disk

PageLSN= -

A = abc D = mnp

B = klm

P505
PageLSN= -

C = tuv

P700
PageLSN= -

E = pq

- Affected page is not in the dirty page table: **N**
- Else, recLSN > LSN being checked: **N**
- Else, pageLSN >= LSN being checked: **N**
- **REDO**

REDO Phase

Dirty page table

pageID	recLSN
P500	101
P600	102
P505	104

Transaction table

transID	lastLSN	status
T1000	104	U

Buffer Pool

PageLSN= 101

A = def D = mnp

PageLSN= 102

B = klm

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102	-	T_{2000}	P600	Write B “hij” -> “klm”	Update	-
103	102	T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	

Disk

PageLSN= -

A = abc D = mnp

B = klm

P505
PageLSN= -

C = tuv

P700
PageLSN= -

E = pq

- Affected page is not in the dirty page table: **N**
- Else, recLSN > LSN being checked: **N**
- Else, pageLSN >= LSN being checked: **Y**
- **NO REDO = SKIPPED**

REDO Phase

Dirty page table

pageID	recLSN
P500	101
P600	102
P505	104

Transaction table

transID	lastLSN	status
T1000	104	U

Buffer Pool

PageLSN= 101 to **103**

A = def **D = qrs**

B = klm

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A "abc" -> "def"	Update	-
102	-	T_{2000}	P600	Write B "hij" -> "klm"	Update	-
103	102	T_{2000}	P500	Write D "mnp" -> "qrs"	Update	-
104	101	T_{1000}	P505	Write C "tuv" -> "wxy"	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	

Disk

PageLSN= -

A = abc D = mnp

B = klm

P505
PageLSN= -

C = tuv

P700
PageLSN= -

E = pq

- Affected page is not in the dirty page table: **N**
- Else, recLSN > LSN being checked: **N**
- Else, pageLSN >= LSN being checked: **N**
- **REDO**

REDO Phase

Dirty page table

pageID	recLSN
P500	101
P600	102
P505	104

Transaction table

transID	lastLSN	status
T1000	104	U

Buffer Pool

PageLSN= 103

A = def D = qrs

B = klm

P505
PageLSN="" to **104**

C =wxy

PageLSN= -

A = abc D = mnp

B = klm

P505
PageLSN= -

C = tuv

P700
PageLSN= -

E = pq

Disk

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A "abc" -> "def"	Update	-
102	-	T_{2000}	P600	Write B "hij" -> "klm"	Update	-
103	102	T_{2000}	P500	Write D "mnp" -> "qrs"	Update	-
104	101	T_{1000}	P505	Write C "tuv" -> "wxy"	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	

- Affected page is not in the dirty page table: **N**
- Else, recLSN > LSN being checked: **N**
- Else, pageLSN >= LSN being checked: **N**
- **REDO**

UNDO Phase

- Work with the Transaction table in the analysis phase
  - “Loser transactions” must be undone
  - Changes during undo phases are written (CLR) so that it is not repeated at the time of repeated restarts

- Scan backward

- Maintain a set ToUndo
  - Initialize to lastLSNs of all “U” transactions at Transaction Table
  - undo the “largest LSN” in ToUndo at each step (the latest one in bottom-up order)

UNDO Phase

Dirty page table

pageID	recLSN
P500	101
P600	102
P505	104

Transaction table

transID	lastLSN	status
T1000	104	U

Buffer Pool

PageLSN= 103

A = def D = qrs

B = klm

P505
PageLSN= 104

C =wxy

PageLSN= -

A = abc D = mnp

B = klm

P505
PageLSN= -

C = tuv

P700
PageLSN= -

E = pq

Disk

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102	-	T_{2000}	P600	Write B “hij” -> “klm”	Update	-
103	102	T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	

ToUNDO = {104}

CLR

- CLR is added so that no “Undo” action is undone
- If a CLR is encountered during UNDO phase, goes to the LSN in UndoNextLSN

UNDO Phase

Dirty page table

pageID	recLSN
P500	101
P600	102
P505	104

Transaction table

transID	lastLSN	status
T1000	104	U

Buffer Pool

PageLSN= 103

A = def D = qrs

B = klm

P505
PageLSN= 107

C =tuv

PageLSN= -

A = abc D = mnp

B = klm

P505
PageLSN= -

C = tuv

P700
PageLSN= -

E = pq

Disk

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A "abc" -> "def"	Update	-
102	-	T_{2000}	P600	Write B "hij" -> "klm"	Update	-
103	102	T_{2000}	P500	Write D "mnp" -> "qrs"	Update	-
104	101	T_{1000}	P505	Write C "tuv" -> "wxy"	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	
107		T_{1000}		UndoT_{1000} LSN104	CLR	101

- A CLR is written
- PageLSN = LSN (CLR)
- Value of C is undone

ToUNDO = {101}

UNDO Phase

Dirty page table

pageID	recLSN
P500	101
P600	102
P505	104

Transaction table

transID	lastLSN	status
T1000	104	U

Buffer Pool

PageLSN= 108

A = abc D = qrs

B = klm

P505
PageLSN= 107

C =tuv

PageLSN= -

A = abc D = mnp

B = klm

P505
PageLSN= -

C = tuv

P700
PageLSN= -

E = pq

Disk

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102	-	T_{2000}	P600	Write B “hij” -> “klm”	Update	-
103	102	T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	
107		T_{1000}		UndoT_{1000} LSN104	CLR	101
108		T_{1000}		UndoT_{1000} LSN101	CLR	-

ToUNDO = {}

UNDO Phase

Dirty page table

pageID	recLSN
P500	101
P600	102
P505	104

Transaction table

transID	lastLSN	status
T1000	104	U

Buffer Pool

PageLSN= 103

A = def D = qrs

B = klm

P505
PageLSN= 107

C =tuv

PageLSN= -

A = abc D = mnp

B = klm

P505
PageLSN= -

C = tuv

P700
PageLSN= -

E = pq

Disk

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102	-	T_{2000}	P600	Write B “hij” -> “klm”	Update	-
103	102	T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	
107		T_{1000}		UndoT_{1000} LSN104	CLR	101

ToUNDO = {101}

UNDO Phase

Dirty page table

pageID	recLSN
P500	101
P600	102
P505	104

Transaction table

transID	lastLSN	status
T1000	104	U

Buffer Pool

PageLSN= 108

A = abc D = qrs

B = klm

P505
PageLSN= 107

C =tuv

PageLSN= -

A = abc D = mnp

B = klm

P505
PageLSN= -

C = tuv

P700
PageLSN= -

E = pq

Disk

Log

LSN	prevLSN	tID	pID	Log entry	Type	undoNextLSN
101	-	T1000	P500	Write A “abc” -> “def”	Update	-
102	-	T_{2000}	P600	Write B “hij” -> “klm”	Update	-
103	102	T_{2000}	P500	Write D “mnp” -> “qrs”	Update	-
104	101	T_{1000}	P505	Write C “tuv” -> “wxy”	Update	-
105	103	T_{2000}			Commit	
106	105	T_{2000}			End	
107		T_{1000}		UndoT_{1000} LSN104	CLR	101
108		T_{1000}		UndoT_{1000} LSN101	CLR	-
109		T_{1000}			End	-

Write an END record
(explicitly mentioned in hw 4 for the aborted transaction)

What happens if T aborts?

1. Write an “abort” log record for T
   - Like “commit”
   - Also the status in Transaction table from “Running” to “Aborted”

2 Follow “prevLSN” to undo all updates by T
   - Like the “UNDO” phase
   - Undo page content in buffer pool, pageLSN changed to that LSN(CLR)
   - Write the CLR records
     - Log entry like “Undo T3 LSN5”
     - No prevLSN, but undoNextLSN field present
     - Until undoNextLSN is null

3. Write an “End” log record for T

Handling Crashes during Undo

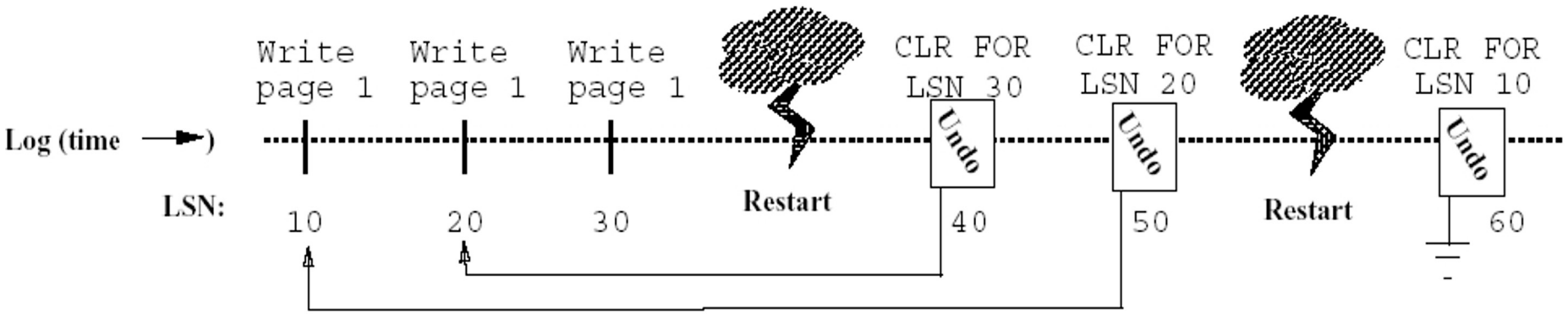


Figure 4: The Use of CLRs for UNDO

[Figure 4 from Franklin97]

In general, for every single crash (even for crash during Analysis/Redo/Undo phases), start again with Analysis

If some CLR records are written to disk during an UNDO phase, then a crash happens (e.g. here LSN 40, 50 are written to disk before the second crash), then the next UNDO phase will skip undoing those CLRs.