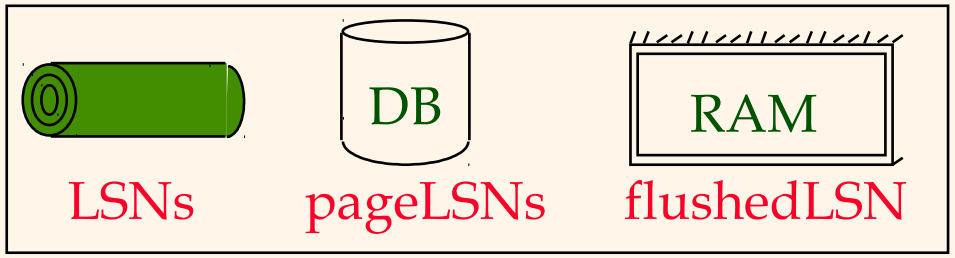
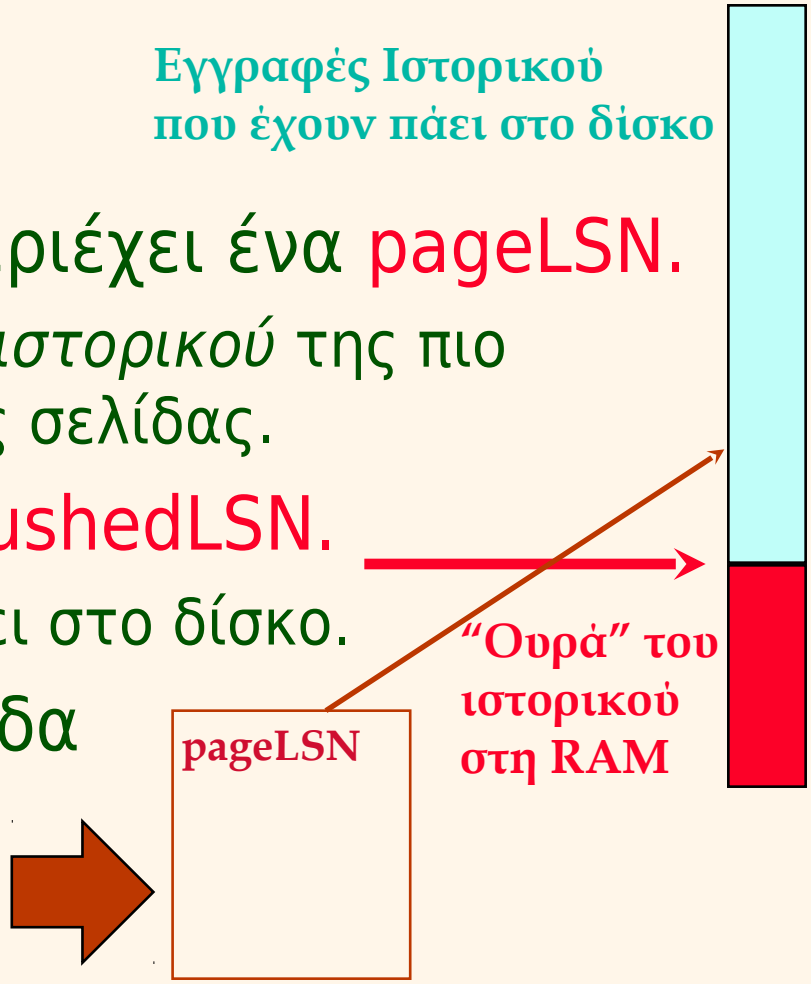
Επαναφορά του Συστήματος (Μέρος Β')

Κεφάλαιο 18

WAL & Ιστορικό

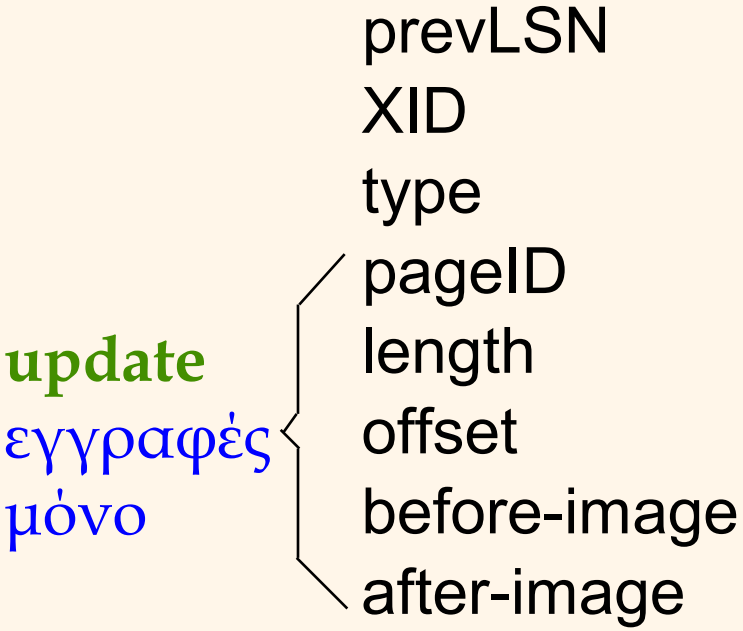


- Κάθε εγγραφή ιστορικού έχει ένα μοναδικό Log Sequence Number (LSN).
  - Τα LSN πάντα αυξάνουν.
- Κάθε *σελίδα δεδομένων* περιέχει ένα pageLSN.
  - Είναι το LSN της *εγγραφής ιστορικού* της πιο πρόσφατης ενημέρωσης της σελίδας.
- Το Σύστημα διατηρεί το flushedLSN.
  - Το μέγιστο LSN που έχει πάει στο δίσκο.
- WAL: *Πριν* γραφεί μια σελίδα στο δίσκο,
  - pageLSN ≤ flushedLSN



Εγγραφές Ιστορικού

Πεδία Εγγραφών Ιστορικού:



Πιθανοί τύποι εγγραφών ιστορικού:

- **Update (Τροποποίηση)**
- **Commit (Ολοκλήρωση)**
- **Abort (Εγκατάλειψη)**
- **End (Τερματισμός)** (τερματισμός του commit ή του abort)
- Compensation Log Records (CLRs)
  - για ενέργειες UNDO

Άλλη Πληροφορία σχετική με το Ιστορικό

- Transaction Table (Πίνακας Συναλλαγών):
  - Μια καταχώριση ανά ενεργή συναλλαγή.
  - Περιέχει XID, status (running/commited/aborted), και lastLSN.
- Dirty Page Table (Πίνακας Τροποποιημένων Σελίδων):
  - Μια καταχώριση ανά τροποποιημένη σελίδα της ενδιάμεσης μνήμης.
  - Περιέχει recLSN - το LSN της εγγραφής ιστορικού που ***πρώτη*** προκάλεσε την ενημέρωση της σελίδας.

Κανονική Εκτέλεση μιας Συναλλαγής

- Μια ακολουθία από αναγνώσεις & γραψίματα, ακολουθούμενη από commit ή abort.
  - Θα θεωρήσουμε ότι το γράψιμο στο δίσκο είναι ατομική πράξη.
    - Πρακτικά, πρέπει να αντιμετωπιστούν και τα μη-ατομικά γραψίματα.
- Αυστηρό 2PL.
- STEAL, NO-FORCE διαχείριση της ενδιάμεσης μνήμης, με Write-Ahead Logging.

Checkpointing – Λήψη Σημείου Ελέγχου

- Περιοδικά, το DBMS δημιουργεί ένα checkpoint, ώστε να ελαχιστοποιήσει τον χρόνο επαναφοράς μετά από κατάρρευση του συστήματος. Γράφει στο ιστορικό:
  - εγγραφή begin_checkpoint: υποδηλώνει πότε ξεκίνησε το σημείο ελέγχου.
  - εγγραφή end_checkpoint: περιέχει τους τρέχοντες πίνακες *συναλλαγών* και *τροποιημένων σελίδων*. Πρόκειται για ένα `fuzzy checkpoint':
    - Οι υπόλοιπες συναλλαγές συνεχίζουν να εκτελούνται. Άρα οι πίνακες αυτοί είναι ακριβείς ως προς τη χρονική στιγμή της εγγραφής begin_checkpoint.
    - Δεν εξαναγκάζονται τροποποιημένες σελίδες στο δίσκο. Η αποτελεσματικότητα του σημείου ελέγχου περιορίζεται από την παλαιότερη τροποποίηση σε σελίδα που δεν έχει γραφεί στο δίσκο. (Άρα, είναι καλή ιδέα να γράφουμε περιοδικά τροποποιημένες σελίδες στο δίσκο!)
  - Αποθηκεύει σε ασφαλές μέρος το LSN της εγγραφής του σημείου ελέγχου (εγγραφή *master*).

Όλη η εικόνα: Τί αποθηκεύεται πού



Εγγραφές Ιστορικού

prevLSN
XID
type
pageID
length
offset
before-image
after-image



Σελίδες Δεδομένων

η καθεμιά με pageLSN

εγγραφή master



Πίνακας Συναλλαγών

lastLSN
status

Πίνακας Τροποποιημένων Σελίδων

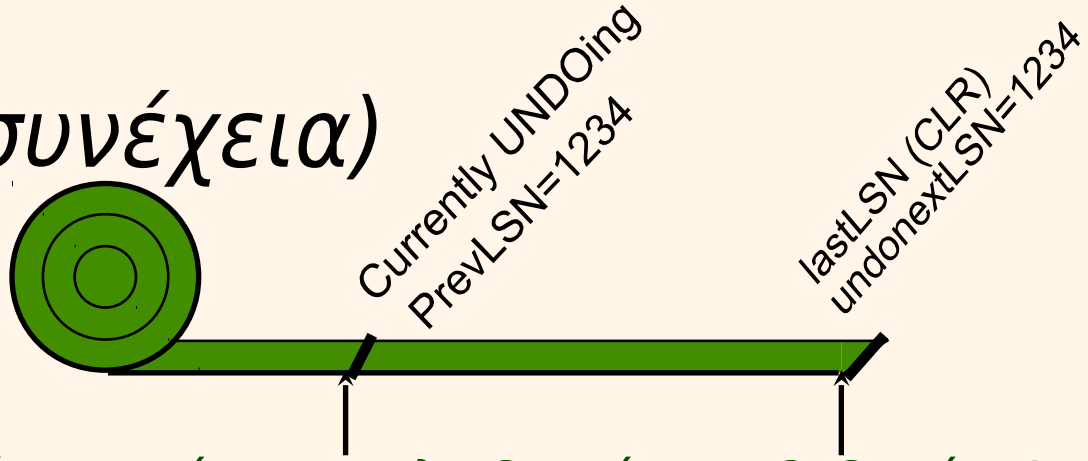
recLSN

flushedLSN

Απλή Εγκατάλειψη Συναλλαγής

- Προς το παρόν, ας θεωρήσουμε την απευθείας εγκατάλειψη μιας συναλλαγής.
  - Δεν υπάρχει κατάρρευση του συστήματος.
- Θέλουμε να "παίξουμε" το ιστορικό ανάποδα, για να αναιρέσουμε (UNDO) τις ενημερώσεις.
  - Διαβάζουμε το lastLSN της συναλλαγής από τον πίνακα συναλλαγών.
  - Με τη βοήθεια του πεδίου prevLSN μπορούμε να διασχίσουμε ανάποδα την αλυσίδα των εγγραφών ιστορικού.
  - Πριν την έναρξη του UNDO, γράφουμε μια *εγγραφή ιστορικού Abort*.
    - Ώστε να επαναφέρουμε το σύστημα από κατάρρευση κατά τη φάση UNDO!

Εγκατάλειψη (συνέχεια)



- Για να γίνει UNDO, πρέπει να έχουμε κλειδαριά στα δεδομένα!
  - Κανένα πρόβλημα!
- Πριν την επαναφορά της παλιάς τιμής σε μια σελίδα, γράφεται μια εγγραφή ιστορικού CLR:
  - Κατά τη διάρκεια του UNDO συνεχίζεται το γράψιμο στο ιστορικό!!
  - Το CLR έχει ένα επιπλέον πεδίο: undonextLSN
    - Δείχνει στο επόμενο προς αναίρεση LSN (δηλαδή, στο prevLSN της εγγραφής ιστορικού που εκείνη τη στιγμή αναιρείται).
  - Τα CLRs *ποτέ* δεν αναιρούνται (αλλά μπορεί να επανεκτελεστούν κατά την επανάληψη της ιστορίας: αυτό εγγυάται την Ατομικότητα!)
- Στο τέλος του UNDO, γράφεται μια εγγραφή ιστορικού “end”.

Ολοκλήρωση Συναλλαγής

- Γράφεται στο ιστορικό μια εγγραφή commit.
- Όλες οι εγγραφές ιστορικού ως το lastLSN της συναλλαγής γράφονται στο δίσκο.
  - Αυτό εγγυάται ότι flushedLSN ≥ lastLSN.
  - Σημειώστε ότι το γράψιμο του ιστορικού στο δίσκο είναι ακολουθιακό και συγχρονισμένο.
  - Υπάρχουν πολλές εγγραφές ιστορικού σε μια σελίδα του ιστορικού.
- Επιστρέφει το Commit().
- Γράφεται μια εγγραφή end στο ιστορικό.

Επαναφορά από Κατάρρευση: Η Γενική Εικόνα



- Ξεκινάμε από ένα checkpoint (το βρίσκουμε μέσω της εγγραφής master).
- Τρεις φάσεις. Πρέπει να:
  - Να βρούμε ποιες συναλλαγές ολοκλήρωσαν και ποιες απέτυχαν μετά το σημείο ελέγχου (Ανάλυση).
  - Να επανεκτελέσουμε (REDO) ***όλες*** τις ενέργειες.
    - (επανάληψη της ιστορίας)
  - Να αναιρέσουμε (UNDO) τα αποτελέσματα των αποτυχημένων συναλλαγών.

Επαναφορά: Η Φάση Ανάλυσης

- Ανακατασκευάζουμε την κατάσταση του συστήματος στο σημείο ελέγχου
  - μέσω της εγγραφής end_checkpoint.
- Σαρώνουμε το ιστορικό προς τα εμπρός από το σημείο ελέγχου, και για:
  - Εγγραφή End: απομακρύνουμε τη συναλλαγή από τον Πίνακα Συναλλαγών.
  - Άλλες εγγραφές: Προσθέτουμε τη συναλλαγή στον Πίνακα Συναλλαγών, θέτουμε lastLSN=LSN, αλλάζουμε την κατάσταση της συναλλαγής σε commit.
  - Εγγραφή Update: Αν η σελίδα P δεν υπάρχει στον Πίνακα Τροποποιημένων Σελίδων,
    - την προσθέτουμε και θέτουμε recLSN=LSN.

Επαναφορά: Η Φάση REDO

- *Επαναλαμβάνουμε την Ιστορία* ώστε να ανακατασκευάσουμε την κατάσταση του συστήματος τη στιγμή της κατάρρευσης:
  - Επανεφαρμόζουμε *όλες* τις ενημερώσεις (ακόμα και εγκαταλελειμμένων συναλλαγών!), επανεκτελούμε τα CLR.
- Σαρώνουμε προς τα εμπρός από την εγγραφή ιστορικού που αντιστοιχεί στο μικρότερο recLSN στον Πίνακα Τροποποιημένων Σελίδων. Για κάθε LSN εγγραφής CLR ή update, επανεκτελούμε (REDO) την ενέργεια εκτός και αν:
  - Η τροποποιημένη σελίδα δεν υπάρχει στον Πίνακα Τροπ. Σελίδων, ή
  - Η σελίδα υπάρχει στον Πίνακα Τροπ. Σελίδων αλλά έχει recLSN > LSN, ή
  - pageLSN (στη βάση δεδομένων) ≥ LSN.
- Για να επανεκτελέσουμε (REDO) μια ενέργεια:
  - Επανεφαρμόζουμε την καταγεγραμμένη στο ιστορικό τιμή.
  - Θέτουμε pageLSN σε LSN. Δεν χρειάζεται καταγραφή στο ιστορικό!

Επαναφορά: Η Φάση UNDO

ToUndo={ / | / το lastLSN μιας χαμένης συναλλαγής}

Repeat:

- Διάλεξε το μεγαλύτερο LSN στο ToUndo.
- Αν το LSN είναι CLR και undonextLSN==NULL
  - Γράψε μια εγγραφή ιστορικού End για τη συναλλαγή αυτή.
- Αν το LSN είναι CLR, και undonextLSN != NULL
  - Πρόσθεσε το undonextLSN στο ToUndo
- Αλλιώς το LSN είναι update. Αναίρεσε την ενημέρωση, γράψε ένα CLR, πρόσθεσε το prevLSN στο ToUndo.

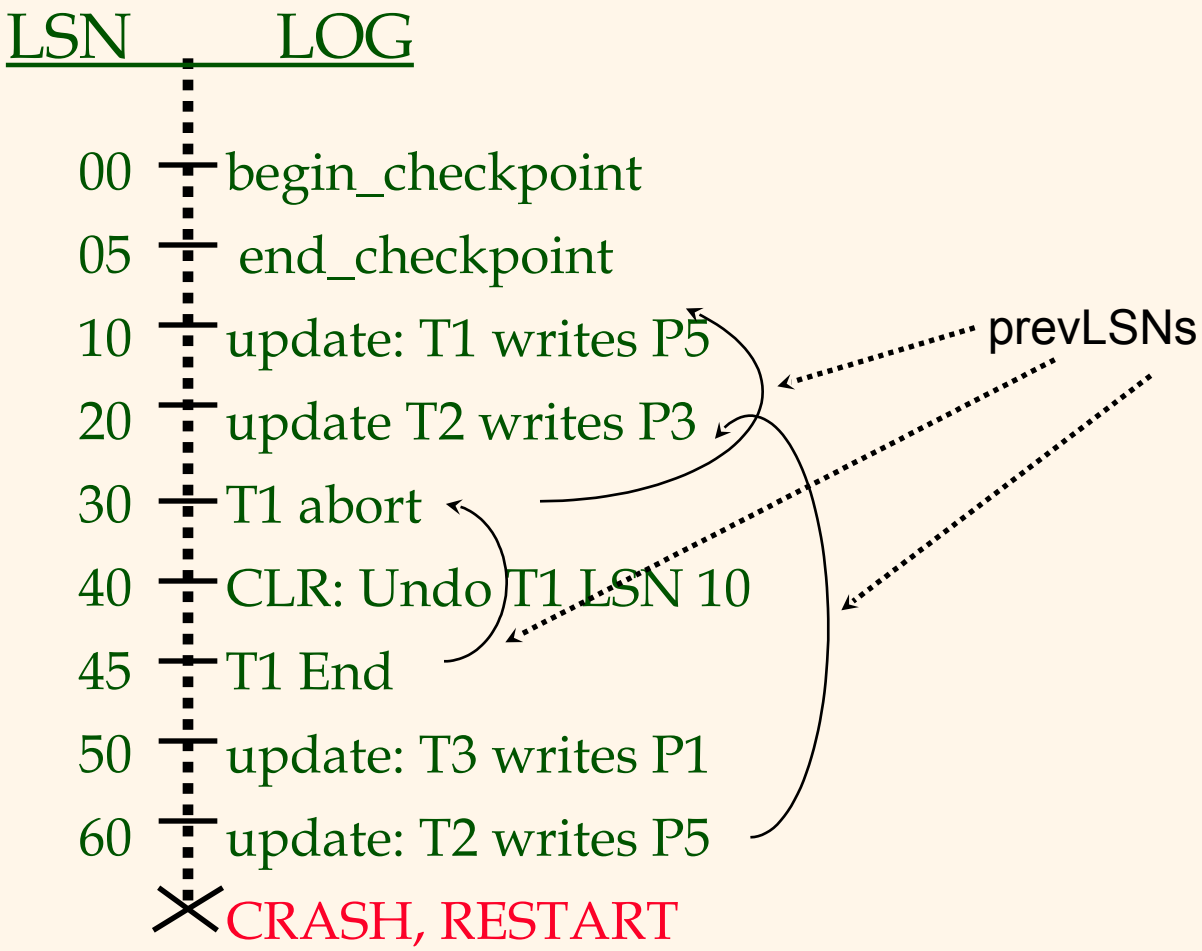
Until ToUndo είναι κενό.

Παράδειγμα Επαναφοράς



Πίνακας Συναλλαγών
lastLSN
status
Πίνακας Τροπ. Σελίδων
recLSN
flushedLSN

ToUndo



Παράδειγμα: Κατάρρευση κατά την Επανεκκίνηση!



Πίνακας Συναλλαγών
lastLSN
status
Πίνακας Τροπ. Σελίδων
recLSN
flushedLSN

ToUndo

LSN	LOG
00,05	begin_checkpoint, end_checkpoint
10	update: T1 writes P5
20	update T2 writes P3
30	T1 abort
40,45	CLR: Undo T1 LSN 10, T1 End
50	update: T3 writes P1
60	update: T2 writes P5
	CRASH, RESTART
70	CLR: Undo T2 LSN 60
80,85	CLR: Undo T3 LSN 50, T3 end
	CRASH, RESTART
90	CLR: Undo T2 LSN 20, T2 end

undonextLSN

Επιπλέον Ζητήματα της Κατάρρευσης

- Τί συμβαίνει αν το σύστημα καταρρεύσει κατά τη Φάση Ανάλυσης; Κατά τη Φάση REDO;
- Πώς περιορίζουμε το φόρτο εργασίας κατά τη Φάση REDO;
  - Γράφουμε σελίδες στο δίσκο ασύγχρονα και στο υπόβαθρο.
  - Προσέχουμε για “hot spots”!
- Πώς περιορίζουμε το φόρτο εργασίας κατά τη Φάση UNDO?
  - Αποφεύγουμε συναλλαγές μακράς διάρκειας.

Σύνοψη της Διατήρησης Ιστορικού/ Επαναφοράς του Συστήματος

- Ο Διαχειριστής Επαναφοράς εγγυάται την Ατομικότητα & Μονιμότητα (A & D).
- Χρησιμοποιούμε WAL ώστε να επιτρέπουμε STEAL/NO-FORCE χωρίς να θυσιάζουμε την ορθότητα και συνέπεια των δεδομένων.
- Τα LSN προσδιορίζουν τις εγγραφές ιστορικού. Είναι συνδεδεμένα σε λίστες για κάθε συναλλαγή (μέσω του prevLSN).
- Το pageLSN επιτρέπει τη σύγκριση των σελίδων δεδομένων και των εγγραφών ιστορικού.

Σύνοψη (συνέχεια)

- Λήψη Σημείου Ελέγχου (Checkpointing): Ένας γρήγορος τρόπος για να περιοριστεί το εύρος σαρώματος του ιστορικού κατά την επαναφορά.
- Η Επαναφορά έχει 3 φάσεις:
  - Ανάλυση: Από το checkpoint και προς τα εμπρός.
  - Redo: Από το παλαιότερο recLSN και προς τα εμπρός.
  - Undo: Από το τέλος προς το πρώτο LSN της παλαιότερης συναλλαγής που ήταν ενεργή κατά την κατάρρευση και προς τα πίσω.
- Κατά τη φάση Undo, γράφουμε CLR.
- Η φάση Redo “επαναλαμβάνει την Ιστορία”: Έτσι απλοποιείται η λογική του αλγορίθμου!