Επεξεργασία Δοσοληψιών (συνέχεια)

Πρόβλημα

«Σωστή» εκτέλεση προγραμμάτων όταν επιτρέπουμε *ταυτοχρονισμό* και ακόμα και αν υπάρχουν *αποτυχίες*

Δοσοληψία (transaction)

Η εκτέλεση ενός προγράμματος σε μια βάση δεδομένων. Μια ακολουθία από πράξεις R, W που τελειώνει ή με Commit ή με Abort

- **BEGIN**
- **R(X) W(X)**
- **END**
- **COMMIT (επικύρωση) – επιτυχία – όλες οι τροποποιήσεις επικυρώνονται και δεν μπορούν να αναιρεθούν**
- **ABORT (ακύρωση ή ανάκληση) – αποτυχία – όλες οι τροποποιήσεις πρέπει να αναιρεθούν**

Ιδιότητες Δοσοληψιών

- **A**tomicity (ατομικότητα) – είτε όλες οι πράξεις είτε καμία
- **C**onsistency (συνέπεια) – διατήρηση συνέπειας της ΒΔ
- **I**solation (απομόνωση) – δεν αποκαλύπτει ενδιάμεσα αποτελέσματα
- **D**urability (μονιμότητα ή διάρκεια) – μετά την επικύρωση μιας δοσοληψίας οι αλλαγές δεν είναι δυνατόν να χαθούν

- **A**tomicity (ατομικότητα) → ΤΕΧΝΙΚΕΣ ΑΝΑΚΑΜΨΕΙΣ
- **C**onsistency (συνέπεια) → ΥΠΕΥΘΥΝΟΤΗΤΑ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΙΣΤΗ
- **I**solation (απομόνωση) → ΕΛΕΓΧΟΣ ΣΥΝΔΡΟΜΙΚΟΤΗΤΑΣ
- **D**urability (μονιμότητα ή διάρκεια) → ΤΕΧΝΙΚΕΣ ΑΝΑΚΑΜΨΕΙΣ

Ισοδύναμα Χρονοπρογράμματα βάσει Συγκρούσεων:

Δυο χρονοπρογράμματα είναι ισοδύναμα βάσει συγκρούσεων αν η διάταξη κάθε ζεύγους συγκρουόμενων πράξεων είναι ίδια και στα δυο χρονοπρογράμματα.

Ένα χρονοπρόγραμμα είναι σειριοποιήσιμο (βάσει συγκρούσεων) αν και μόνο αν ο γράφος προήγησής του είναι ακυκλικός.

Άσκηση 1

Ο γράφος προήγησης για ένα συγκεκριμένο χρονοπρόγραμμα περιέχει μόνο τις παρακάτω ακμές και κόμβους. Πόσα διαφορετικά σειριακά χρονοπρογράμματα είναι ισοδύναμα με αυτό το χρονοπρόγραμμα;

ΠΑΡΑΤΗΡΗΣΗ: Διαφορετικά χρονοπρογράμματα; Έστω T_i με m_i πράξεις

$(m_1 + m_2 + ... + m_N) ! / (m_1! m_2! ... m_N!)$

- Χρονοπρογράμματα με δυνατότητα ανάκαμψης (recoverable)

αν καμιά δοσοληψία T στο S δεν επικυρώνεται έως ότου επικυρωθούν όλες οι δοσοληψίες οι οποίες τροποποίησαν ένα δεδομένο που διαβάζει η T

- Χρονοπρογράμματα χωρίς διάδοση ανακλήσεων (avoids cascading aborts)

αν κάθε δοσοληψία T στο S διαβάζει μόνο στοιχεία που έχουν γραφεί από επικυρωμένες δοσοληψίες

- Αυστηρά Χρονοπρογράμματα (strict)

οι δοσοληψίες δεν μπορούν ούτε να διαβάσουν ούτε να γράψουν ένα στοιχείο X έως ότου επικυρωθεί η δοσοληψία που έγραψε το X

- Ο χρήστης δεν ασχολείται με τη συνδρομικότητα
- Το ΣΔΒΔ εξασφαλίζει «σωστή συνδρομικότητα»: *δρομολογεί* τις πράξεις των δοσοληψιών ώστε να προκύπτουν χρονοπρογράμματα σειριοποιήσιμα βάσει συγκρούσεων
- Μέσω τεχνικών ελέγχου συνδρομικότητας

Τεχνικές

1. Κλειδώματος (locking) για να αποτρέψουν τη συνδρομική (ταυτόχρονη) προσπέλαση των δεδομένων από πολλές δοσοληψίες

2. Διάταξης χρονοσημάτων (timestamps)

3. Πιστοποίησης (validation) μιας δοσοληψίας (αισιόδοξα πρωτόκολλα)

Δύο ειδών κλειδιά:

- διαμοιραζόμενο (shared) κλειδί ή κλειδί ανάγνωσης
- αποκλειστικό (exclusive) κλειδί ή κλειδί εγγραφής

	S-Lock(X)	X-Lock(X)
S-Lock(X)	√	
X-Lock(X)		

Πρωτόκολλο κλειδώματος δυο φάσεων (Two-Phase Locking 2PL)

Όλες οι πράξεις (αιτήσεις) κλειδώματος μιας δοσοληψίας προηγούνται της πρώτης πράξης (αίτησης) άρσης κλειδώματος της διαδικασίας

- Δυνατή η «αναβάθμιση» κλειδιών

- Κάθε δοσοληψία δυο φάσεις
  - μια φάση επέκτασης ή εξάπλωσης
  - μια φάση συρρίκνωσης
- Αποδεικνύεται ότι είναι σωστό: παράγει χρονοπρογράμματα σειριοποιήσιμα με συγκρούσεις. Υπάρχουν σειριοποιήσιμα με συγκρούσεις χρονοπρογράμματα που δεν παράγονται με κλείδωμα δύο φάσεων;
- Παράγεται αυστηρό χρονοπρόγραμμα; Όχι.

Παραλλαγές:

Αυστηρό Κλείδωμα Δύο Φάσεων: μια δοσοληψία δεν απελευθερώνει κανένα αποκλειστικό κλειδί πριν επικυρωθεί ή ακυρωθεί

Υπερ-αυστηρό Κλείδωμα Δύο Φάσεων: μια δοσοληψία δεν απελευθερώνει κανένα (αποκλειστικό και διαμοιραζόμενο) κλειδί πριν επικυρωθεί ή ακυρωθεί

Τα πιο δημοφιλή πρωτόκολλα – ευρεία χρήση σε εμπορικά συστήματα

- Πρόβλημα με τη δημιουργία αδιεξόδων
- Κόστος διαχείρισης κλειδιών

Δυο τεχνικές:

• Πρωτόκολλα Πρόληψης Αδιεξόδων (Deadlock Prevention): Αποφυγή δημιουργίας αδιεξόδου

  • όλα τα κλειδιά δίνονται μαζί

  • διάταξη των δοσοληψιών (με χρήση χρονοσήμων)

• Πρωτόκολλα Ανίχνευσης Αδιεξόδου (Deadlock Detection): Ελέγχουμε περιοδικά αν το σύστημα βρίσκεται σε κατάσταση αδιεξόδου (π.χ με χρήση γράφου αναμονής)

Άσκηση

Αποδείξτε ότι το βασικό πρωτόκολλο κλειδώματος δύο φάσεων εξασφαλίζει σειριοποιησιμότητα βάσει συγκρούσεων

Με απαγωγή στο άτοπο

Έστω ότι δεν το εξασφαλίζει, δηλαδή κλείδωμα δυο φάσεων και μη σειριοποιησιμότητα βάσει συγκρούσεων ⇒

Κύκλος στο γράφο σειριοποιησιμότητας

Άσκηση

Θεωρείστε το παρακάτω χρονοπρόγραμμα των δοσοληψιών T_1 και T_2:

S: b $W_1(X)$ $R_2(X)$ $W_2(Y)$ C_2 $W_1(Z)$ C_1 e

b αρχή και e τέλος του χρονοπρογράμματος. Υποθέστε 2PL.

Πότε -- μεταξύ ποιών γεγονότων μπορούν να συμβούν τα παρακάτω

π.χ., T1 μπορεί να κλειδώσει το X μεταξύ των b και W1(X)

Δυο τεχνικές:

• Πρωτόκολλα Πρόληψης Αδιεξόδων (Deadlock Prevention): Αποφυγή δημιουργίας αδιεξόδου –*χρονόσημα (προτεραιότητες) σε κάθε δοσοληψία, επανεκκίνηση δοσοληψίας με το **ίδιο** χρονόσημα*

• *Αναμονή – Θανάτωση*

• *Τραυματισμός αναμονή*

• Πρωτόκολλα Ανίχνευσης Αδιεξόδου (Deadlock Detection): Ελέγχουμε περιοδικά αν το σύστημα βρίσκεται σε κατάσταση αδιεξόδου *γράφος αναμονής*

S: b $W_1(X)$ $R_2(X)$ $W_2(Y)$ C_2 $W_1(Z)$ C_1 e

Πότε -- μεταξύ ποιών γεγονότων μπορούν να συμβούν τα παρακάτω

(α) Πότε μπορεί η T_1 να κάνει unlock(X)

(β) Πότε μπορεί η T_1 να κάνει lock(Z)

Άσκηση

Θεωρείστε ότι οι πράξεις υποβάλλονται με την παρακάτω σειρά. Θεωρείστε ότι οι εντολές έρχονται στο scheduler με την παρακάτω σειρά. Περιγράψτε για το καθένα από τα παρακάτω πρωτόκολλα πως χειρίζεται τις πράξεις ο μηχανισμός ελέγχου ταυτοχρονισμού

$R_1(X)$ $W_2(X)$ $W_2(Y)$ $W_3(Y)$ $W_1(Y)$ C_1 C_2 C_3

$R_1(X)$ $W_2(Y)$ $W_2(X)$ $W_3(Y)$ $W_1(Y)$ C_1 C_2 C_3

(α) Αυστηρό 2PL με χρήση χρονοσημάτων για αποφυγή (πρόληψη) αδιεξόδου και αναμονή-θανάτωση (το κλειδί το έχει με μικρότερο χρονόσημα (παλιότερη) – ακυρώνεται, αλλιώς περιμένει)

(β) Αυστηρό 2PL με χρήση γράφου αναμονής για ανίχνευση αδιεξόδου

Διάταξη Χρονοσημάτων

Το χρονόσημα δημιουργείται από το ΣΔΒΔ και προσδιορίζει μοναδικά μια δοσοληψία

Ιδέα: *διάταξη των δοσοληψιών με βάση το χρονόσημα τους* (δηλαδή, χρονοπρόγραμμα ισοδύναμο με σειριακό στο οποίο οι δοσοληψίες εμφανίζονται διατεταγμένες με βάση τις τιμές των χρονοσημάτων)

⇒ άρα η σειρά προσπέλασης στα δεδομένα πρέπει να μη παραβιάζει τη σειριοποιησιμότητα

Δηλαδή: αν μια πράξη a_i μιας δοσοληψίας T_i συγκρούεται με μια πράξη a_j μιας δοσοληψίας T_j και $TS(T_i) < TS(T_j)$, τότε η a_i πρέπει να προηγείται της a_j. Αλλιώς, restart τη δοσοληψία.

Διάταξη Χρονοσημάτων

Κάθε δεδομένο X έχει δύο τιμές χρονοσημάτων:

ΧΣΑ(X) (χρονόσημα ανάγνωσης) το μεγαλύτερο μεταξύ όλων των χρονοσημάτων των δοσοληψιών που διάβασαν το X (διαισθητικά, η πιο πρόσφατη που το διάβασε)

ΧΣΕ(X) (χρονόσημα εγγραφής) το μεγαλύτερο μεταξύ όλων των χρονοσημάτων των δοσοληψιών που έγραψαν το X

Κάθε πράξη ανάγνωσης και εγγραφής μιας δοσοληψίας συνοδεύεται από κατάλληλους ελέγχους του χρονοσήματός της και των χρονοσημάτων του αντίστοιχου δεδομένου.

Αν ο έλεγχος αποτύχει, η δοσοληψία απορρίπτεται και αρχίζει πάλι

Διάταξη Χρονοσημάτων

- Παράγει σωστά χρονοπρογράμματα
- Δε δημιουργούνται αδιέξοδα (αφού καμιά δοσοληψία δεν περιμένει)
- Μπορεί να παραχθούν χρονοπρογράμματα που δεν έχουν δυνατότητα ανάκαμψης (not recoverable)

  Παραλλαγή για χρονοπρόγραμμα χωρίς διάδοση ανακλήσεων (ιδέα: commit bit με κάθε δεδομένο – καθυστέρηση αναγνώσεων)

  Παραλλαγή για αυστηρά (αναμονή αναγνώσεων και εγγραφών)

  Άλλη ιδέα (R(X) ή W(X) από T με ΧΣ(T) > ΧΣΕ(X) (αν μικρότερο ακυρώνεται) περιμένει μέχρι να επικυρωθεί ή να ακυρωθεί η δοσοληψία που έγραψε το X (όχι αδιέξοδα)

Επιβάρυνση διαχείρισης χρονοσημάτων

Απόρριψη και επανεκκίνηση δοσοληψιών

Διάταξη Χρονοσημάτων: Άσκηση 7

Άσκηση

Εφαρμόστε τον αλγόριθμο διάταξης χρονοσημάτων στα χρονοπρογράμματα (α), (β), και (γ)

(α)

$R_1(X)$ $W_2(Y)$ $W_2(X)$ $W_3(X)$ $W_1(Y)$ C_1 C_2 C_3

Διάταξη Χρονοσημάτων: Άσκηση 7(β)

T1	T2	T3
	$R_2(Z)$	
	$R_2(Y)$	
	$W_2(Y)$	
		$R_3(Y)$
		$R_3(Z)$
$R_1(X)$		
$W_1(X)$		
		$W_3(Y)$
		$W_3(Z)$
	$R_2(X)$	
$R_1(Y)$		
$W_1(Y)$		
	$W_2(X)$	

• σειριοποίησιμο;

Διάταξη Χρονοσημάτων: Άσκηση 7(γ)

T1	T2	T3
		$R_3(Y)$
		$R_3(Z)$
$R_1(X)$		
$W_1(X)$		
		$W_3(Y)$
		$W_3(Z)$
	$R_2(Z)$	
$R_1(Y)$		
$W_1(Y)$		
	$R_2(Y)$	
	$W_2(Y)$	
	$R_2(X)$	
	$W_2(X)$	

• σειριοποίησιμο;

Αισιόδοξες Τεχνικές

Οι τεχνικές κλειδώματος είναι συντηρητικές (αποφεύγονται οι συγκρούσεις)

Μειονεκτήματα

- επιβάρυνση (overhead) χειρισμού κλειδώματος
- αποφυγή/ανίχνευση αδιεξόδων
- lock contention για τα δεδομένα που χρησιμοποιούνται συχνά

Αν οι συγκρούσεις είναι σπάνιες, μεγαλύτερη συγχρονικότητα, αν αντί για κλείδωμα, έλεγχος για συγκρούσεις όταν μια δοσοληψία επικυρώνεται (commits)

Αισιόδοξες Τεχνικές

Κάθε δοσοληψία έχει τρεις φάσεις

- **ΑΝΑΓΝΩΣΗ**: η δοσοληψία διαβάζει από τη βδ, αλλά τροποποιεί προσωπικά αντίγραφα των δεδομένων
- **ΠΙΣΤΟΠΟΙΗΣΗ**: έλεγχος για συγκρούσεις
- **ΕΓΓΡΑΦΗ**: γράφει τα τοπικά αντίγραφα στη βδ

Κατά την πιστοποίηση μιας δοσοληψίας ελέγχουμε τα σύνολα

ReadSet(T): το σύνολο των δεδομένων που διάβασε η Τ

WriteSet(T): το σύνολο των δεδομένων που διάβασε η Τ

με τα αντίστοιχα σύνολα όλων των ταυτόχρονα εκτελούμενων δοσοληψιών του συστήματος

Πιστοποίηση

Έλεγχος συνθηκών που είναι **ικανές** για να εξασφαλίσουν ότι δεν υπήρχαν συγκρούσεις

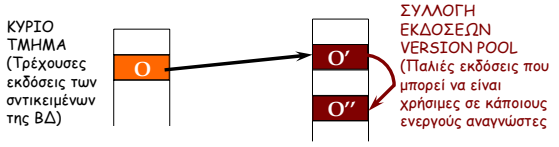
- Κάθε δοσοληψία Τ έχει ένα μοναδικό αριθμό TID (χρονόσημα)
- Το TID ανατίθεται στο τέλος της φάσης ΑΝΑΓΝΩΣΗΣ (ακριβώς πριν αρχίσει η πιστοποίηση)
- Με κάθε δοσοληψία

ReadSet(T): το σύνολο των δεδομένων που διάβασε η Τ

WriteSet(T): το σύνολο των δεδομένων που διάβασε η Τ

Πολλαπλές Εκδόσεις

Κάθε εγγραφή παράγει ένα καινούργιο αντίγραφο ενώ οι αναγνώσεις διαβάζουν ένα κατάλληλο παλιό



Οι αναγνώστες μπορούν πάντα να προχωρήσουν -- αλλά μπορεί να χρειαστεί να περιμένουν μέχρι την επικύρωση των εγγραφών

Διατήρηση παλαιότερων εκδοχών, χρήσιμη ως ιστορικό εξέλιξης των τιμών (χρονικές (temporal) βάσεις δεδομένων)

Πολλαπλές Εκδόσεις

Βασική ιδέα:

οι αναγνώσεις διαβάζουν κατάλληλες εκδόσεις ώστε να διατηρήσουν τη σειριποιησιμότητα

Παραλλαγές των τεχνικών ελέγχου συνδρομικότητας

Θα δούμε με χρονοσήματα

Πολλαπλές Εκδόσεις και Χρονοσήματα

- **Κάθε έκδοση** ενός αντικειμένου έχει ως ΧΣΕ το ΧΣ της Τ που την έγγραψε και ως ΧΣΑ το ΧΣ της Τ που διάβασε αυτήν την έκδοση πιο πρόσφατα

  Μια πράξη εγγραφής μια δοσοληψίας Τ δημιουργεί μια καινούργια έκδοση του δεδομένου, με ΧΣΑ και ΧΣΕ ίσο με ΧΣ(Τ)

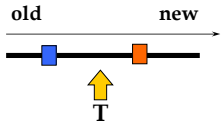
- Οι εκδόσεις συνδέονται (chained backward) μπορούμε να σβήσουμε εκδόσεις που είναι πολύ «παλιές για να έχουν ενδιαφέρον»

- Κάθε δοσοληψία: Reader ή Writer.
  - Writer *μπορεί* να γράψει κάποιο αντικείμενο; Reader μόνο διαβάσει.
  - Κάθε δοσοληψία δηλώνει αν είναι Reader όταν ξεκινά.

  Στόχος είναι η Readers να μην αποτυγχάνουν

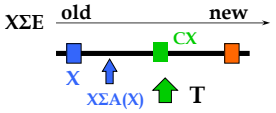
Reader T

- Για την ανάγνωση ενός αντικειμένου:
  - Βρες την πιο νεώτερη έκδοση με ΧΣΕ < ΧΣ(Τ) (Η αναζήτηση ξεκινά από την πιο πρόσφατη - κοίτα σχήμα)
  - Αν υποθέσουμε ότι όλες οι εκδόσεις υπάρχουν, μια Reader T δε χρειάζεται να επανικηθεί.



Writer T

- Για την ανάγνωση, ακολούθησε το πρωτόκολλο του Reader
- Για να γράψεις ένα αντικείμενο Χ:
  - Βρες τη νεώτερη έκδοση με ΧΣΕ(Χ) < ΧΣΕ(Τ)
  - Αν ΧΣΑ(Χ) < ΧΣ(Τ), η Τ δημιουργεί ένα αντίγραφο CX του Χ, με ένα δείκτη στο Χ, και ΧΣΕ(CX) = ΧΣ(Τ), ΧΣΑ(CX) = ΧΣ(Τ). (Οι εγγραφές γίνονται buffered μέχρι την επικύρωση της Τ, οι άλλες δοσοληψίες μπορούν να δουν τις τιμές των ΧΣ αλλά δεν μπορούν να διαβάσουν την έκδοση CX μέχρι η Τ να επικυρωθεί.)
  - Αλλιώς, απόρριψη.



Αν επιτρέψουμε εγγραφές και διαγραφές στοιχείων, τα πρωτόκολλα δε δουλεύουν σωστά (ούτε το αυστηρό 2PL)

Sailors(sid, sname, rating, age)
Boats(bid, bname, color)
Reserves(sid, bid, day)

T1: βρίσκει το γηραιότερο ναύτη με rating 1 και 2

T2: εισάγει ναύτη με rating 1 και age 96 και διαγράφει τον πιο ηλικιωμένο ναύτη με rating 2

- T1 κλειδώνει όλες τις σελίδες που περιέχουν εγγραφές sailor με *rating* = 1, και βρίσκει τον πιο ηλικιωμένο (έστω, *age* = 71).
- Μετά, η T2 εισάγει ένα νέο sailor; *rating* = 1, *age* = 96.
- T2 επίσης διαγράφει τον πιο ηλικιωμένο ναυτικό με rating = 2 (έστω, *age* = 80), και επικυρώνεται.
- T1 τώρα κλειδώνει όλες τις σελίδες που περιέχουν εγγραφές sailor με *rating* = 2, βρίσκει τον πιο ηλικιωμένο (έστω, *age* = 63).

Αποτέλεσμα: 71 63

- Δεν υπάρχει ισοδύναμη με την παραπάνω σειριακή εκτέλεση

Sailors(sid, sname, rating, age)
Boats(bid, bname, color)
Reserves(sid, bid, day)

T1: βρίσκει το γηραιότερο ναύτη με rating 1 και 2

T2: εισάγει ναύτη με rating 1 και age 96 και διαγράφει τον πιο ηλικιωμένο ναύτη με rating 2

- T1 κλειδώνει όλες τις σελίδες που περιέχουν εγγραφές sailor με *rating* = 1, και βρίσκει τον πιο ηλικιωμένο (έστω, *age* = 71).
- Μετά, η T2 εισάγει ένα νέο sailor; *rating* = 1, *age* = 96.
- T2 επίσης διαγράφει τον πιο ηλικιωμένο ναυτικό με rating = 2 (έστω, *age* = 80), και επικυρώνεται.
- T1 τώρα κλειδώνει όλες τις σελίδες που περιέχουν εγγραφές sailor με *rating* = 2, βρίσκει τον πιο ηλικιωμένο (έστω, *age* = 63).

Αποτέλεσμα 71 63

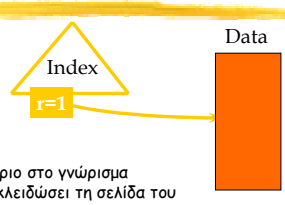
- Δεν υπάρχει ισοδύναμη με την παραπάνω σειριακή εκτέλεση

T1 T2: αποτέλεσμα 71 80

T2 T1: αποτέλεσμα 96 63

- Η T1 έμμεσα υποθέτει ότι έχει κλειδώσει το σύνολο όλων των εγγραφών με *rating* = 1. (εγγραφές φαντάσματα σε άλλες σελίδες)
  - Η υπόθεση ισχύει μόνο αν δεν προστεθούν εγγραφές ενώ εκτελείται η T1
  - Χρειάζεται κάποιος μηχανισμός για να το επιβάλει: κλείδωμα ευρετηρίου (index locking) και κλείδωμα συνθηκών (predicate locking.)

Τα παράδειγμα δείχνει ότι η σειριοποιησιμότητα βάσει συγκρούσεων δίνει σειριοποιησιμότητα μόνο όταν τα αντικείμενα στη βδ είναι σταθερά



Κλείδωμα Ευρετηρίου

- Αν υπάρχει ένα πυκνό ευρετήριο στο γνώρισμα *rating*, η T1 θα μπορούσε να κλειδώσει τη σελίδα του ευρετηρίου που περιέχει τις εγγραφές με *rating* = 1.
  - Αν δεν υπάρχει καμία εγγραφή με *rating* = 1

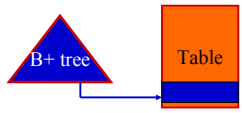
Κλείδωμα Συνθήκης

- Κλείδωσε όλες τις εγγραφές που ικανοποιούν κάποια συνθήκη λογικής, π.χ., *age > 2*salary*.
- Το κλείδωμα ευρετηρίου είναι μια ειδική μορφή κλειδώματος συνθήκης όπου το ευρετήριο επιτρέπει αποδοτική υλοποίηση του κλειδώματος συνθήκης

ΑΡΑ:

Αν είχαμε ένα index (π.χ., B+ tree) πάνω στο πεδίο rating, θα μπορούσαμε να κλειδώναμε γρήγορα όλες τις εγγραφές με RATING=1.

- Ακόμα κι αν δεν υπάρχουν τέτοιες εγγραφές, κλειδώνοντας τον index, μπορούμε να απαγορεύσουμε να δημιουργηθούν από άλλους κατά τη διάρκεια της δοσοληψίας
- Ειδική περίπτωση του κλειδώματος κατηγορήματος



Το κλείδωμα του ευρετηρίου είναι μια ειδική μορφή δεδομένων, πως θα γίνει το κλείδωμα;

- Ας υποθέσουμε ότι θέλουμε να εισάγουμε μια εγγραφή με RATING = 1 στη διάρκεια της T1
- Κατεβαίνοντας από τη ρίζα προς τα φύλλα του B+ δέντρου, μπορούμε να κλειδώνουμε κάθε κόμβο με X-lock [του φύλλου συμπεριλαμβανομένου]

ΠΡΟΒΛΗΜΑ: ήδη από το πρώτο βήμα έχουμε κλειδώσει τη ρίζα, δηλαδή, ΟΛΟ το δέντρο.

ΠΑΡΑΤΗΡΗΣΗ

Στα B+ δέντρα, οι εσωτερικοί κόμβοι χρησιμεύουν ΜΟΝΟ ως μονοπάτια για τα φύλλα, όλα τα πραγματικά δεδομένα είναι στα φύλλα [βασική διαφορά με τα B δέντρα]

Δηλαδή, «συνήθως» οι εσωτερικοί κόμβοι δεν τροποποιούνται, εκτός ΑΝ;

- Αν κλείδωνα με S-lock τους ενδιάμεσους κόμβους και με X-lock τα φύλλα?

Εκτός, αν χρειαστεί να διασπάσω ή να συνενώσω κάποιο εσωτερικό κόμβο

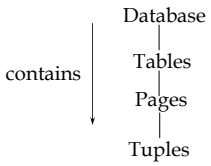
Πότε;

Αν overflow/underflow του παιδιού του!

- Αναζητήσεις: Ξεκινώντας από τη ρίζα κλειδώνουμε με S-lock κάθε κόμβο. Μετά, ξεκλειδώνουμε τον γονέα του.
- Εισαγωγές: Ξεκινώντας από τη ρίζα κλειδώνουμε με X-locks *κατά περίπτωση*. Μετά ελέγχουμε αν ο κλειδωθείς κόμβος είναι **ασφαλής**:
  - Αν ναι, ξεκλειδώνουμε τους προγόνους του
- Ασφαλής κόμβος: οι όποιες αλλαγές δεν θα διαδοθούν προς τα πάνω από τον κόμβο αυτό
  - INS: ο κόμβος δεν είναι πλήρης (όχι overflow)
  - DEL: ο κόμβος δεν είναι άδειος κατά το ήμισυ. (όχι underflow)

Διακριτότητα Κλειδώματος

- Μέχρι τώρα θεωρούσαμε εγγραφές
- Στην πράξη, συνήθως μιλάμε για σελίδες
- Μπορούμε να κλειδώσουμε τη βάση σε πολλά «επίπεδα» που ορίζουν μια «ιεραρχία διακριτότητας»



Ορισμοί

- **Διακριτότητα** (granularity): πόσο μικρό είναι ένα αντικείμενο υπό παρατήρηση
- Μικρή διακριτότητα = μεγάλο μέγεθος στοιχείου (π.χ., πίνακας ή ΒΔ)
- Μεγάλη διακριτότητα = μικρό μέγεθος στοιχείου (π.χ., σελίδα ή εγγραφή)
- **Πρόγονος ενός αντικειμένου**: οποιοδήποτε αντικείμενο στην ιεραρχία διακριτότητας το περικλείει (π.χ., η ΒΔ είναι πρόγονος μιας σελίδας)

Κλείδωμα σε διαφορετικά επίπεδα διακριτότητας

- Κλείδωμα σε μικρή διακριτότητα: αποκλεισμός και καθυστέρηση των δοσοληψιών που τρέχουν εν παραλλήλω (π.χ., τι θα γίνει αν κλειδώσω όλη τη ΒΔ?)
- Κλείδωμα σε μεγάλη διακριτότητα: μεγάλος αριθμός κλειδωμάτων (χώρος μνήμης, καθυστέρηση διαχείρισης)

Εκμετάλλευση της ιεραρχίας

- Η ιδέα είναι να κλειδώνουμε με κατάλληλο τρόπο σε διαφορετικά επίπεδα διακριτότητας
- Χρειαζόμαστε ένα νέο είδος κλειδώματος
- **Κλείδωμα πρόθεσης** (intention lock): εφαρμόζεται σε ένα πρόγονο, χαρακτηρίζοντας τι πράξη (για την ακρίβεια: κλείδωμα) θα εφαρμοστεί στον απόγονό του
- Δύο ειδών κλειδώματα πρόθεσης
  - Πρόθεση αποκλειστικού κλειδιού
  - Πρόσθεση διαμοιραζόμενου κλειδιού

Κλείδωμα Πολλαπλής Κλιμάκωσης

Ιεραρχική δομή

Νέα είδη κλειδιών IS (πρόθεση διαμοιραζόμενου) IX (πρόθεση αποκλειστικό)

Για να κλειδώσει με S (X) πρέπει να κλειδώσει όλους τους προγόνους με IS (IX) αρχίζοντας από τη ρίζα

	IS	IX	S	X
IS	T	T	T	
IX	T	T		
S	T		T	
X				

Κλείδωμα Πολλαπλής Κλιμάκωσης

Συχνά IX και S (διάβασμα όλου, τροποποίηση ενός) είδος κλειδιού SIX

	IS	IX	S	SIX	X
IS	T	T	T	T	
IX	T	T			
S	T		T		
SIX	T				
X					

Αλγόριθμος Κλειδωμάτων

- Το κλείδωμα ξεκινά από την κορυφή της ιεραρχίας
- Για να πάρω ένα κλείδωμα S- (ή IS-) σε ένα αντικείμενο, πρέπει να έχω IS- (ή IX-) στον πρόγονό του [αλλιώς, περιμένω]
- Για να πάρω ένα κλείδωμα X- (ή IX- ή SIX) σε ένα αντικείμενο, πρέπει να έχω IX- (ή SIX-) στον πρόγονό του [αλλιώς, περιμένω]
- Η αποδέσμευση των κλειδιών πρέπει να γίνει με την αντίστροφη σειρά (bottom-up)

Κλείδωμα Πολλαπλής Κλιμάκωσης

Δύο φάσεις

Σειρά αποδέσμευσης κλειδιών;

Υποστήριξη δοσοληψιών στην SQL-92

Δοσοληψία αυτόματα με προτάσεις που τροποποιούν είτε τη βάση δεδομένων είτε το σχήμα (πχ select, update, create table)

Commit ή Rollback (για abort)

- Μέγεθος διαγνωστικών
- Μέθοδος πρόσβασης (ή μέθοδος προσπέλασης):

  READ ONLY ή

  READ WRITE

- Επίπεδο απομόνωσης READ UNCOMIITED, READ COMMITTED, REPEATABLE READ, SERIALIZABLE (αυστηρό 2PL)

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE READ ONLY

default: SERIALIZABLE, READ WRITE

Υποστήριξη δοσοληψιών στην SQL-92

Isolation Level	Dirty Read	Unrepeatable Read	Phantom Problem
Read Uncommitted	Maybe	Maybe	Maybe
Read Committed	No	Maybe	Maybe
Repeatable Reads	No	No	Maybe
Serializable	No	No	No

Αναλυτικά στο βιβλίο

Υποστήριξη δοσοληψιών

IBM DB2, Informix, Microsoft SQL server, Sybase Αυστηρό 2PL (με παραλλαγές για επίπεδα απομόνωσης λιγότερα αυστηρά του SERIALIZABLE)

Microsoft SQL server: χρονόσημα εγγραφής

Oracle 8: πολλαπλές εκδόσεις, ώστε οι αναγνώσεις δεν περιμένουν ποτέ

Κλείδωμα πολλαπλής κλιμάκωσης σε επίπεδο πίνακα, σελίδας και εγγραφής

Ανάκαμψη από Αποτυχίες

Είδη Αποτυχιών

Δυο κατηγορίες: καταστροφή ή όχι της μόνιμης αποθήκευσης (δίσκου)

Στόχος της ανάκαμψης: επαναφορά στην πιο πρόσφατη συνεπή κατάσταση ακριβώς πριν τη στιγμή της αποτυχίας

- Ατομικότητα:
  - όχι επικυρωμένες -- undo -- αναίρεση
- Διάρκεια:
  - επικυρωμένες -- redo -- επανάληψη

Στην πιο εύκολη περίπτωση

- όλες οι ενημερώσεις μιας δοσοληψίας γράφονται στον τοπικό χώρο εργασίας της δοσοληψίας
- η βδ ενημερώνεται *μόνον* αφού μια δοσοληψία φτάσει στο σημείο επικύρωσής της

Στην πραγματικότητα --Steal (flush) ::

- κάποιες ενημερώσεις μιας δοσοληψίας γράφονται στον δίσκο (βδ) πριν η δοσοληψία επικυρωθεί
- Γιατί; βελτίωση throughput
- Πρόβλημα: Ατομικότητα (αναίρεση (undo) στην περίπτωση που η δοσοληψία αποτύχει)
- Χρειαζόμαστε την παλιά τιμή (BeFore IMage)

Το αντίθετο

ετεροχρονισμός των ενημερώσεων -- deferred updates (no steal) όλες οι ενημερώσεις μιας δοσοληψίας γράφονται στον τοπικό χώρο εργασίας της δοσοληψίας -- η βδ ενημερώνεται μόνον αφού μια δοσοληψία φτάσει στο σημείο επικύρωσης της

- *χωρίς αναίρεση/ με επανάληψη (no undo/redo)*

Στην πραγματικότητα -- No Force ::

- κάποιες ενημερώσεις μιας δοσοληψίας δε γράφονται στον δίσκο (βδ) ακόμα και αφού η δοσοληψία επικυρωθεί
- Γιατί; βελτίωση χρόνου απόκρισης
- Πρόβλημα: Διάρκεια (επανάληψη (redo) στην περίπτωση που το σύστημα αποτύχει)
- Χρειαζόμαστε την νέα τιμή (AFter IMage)

Ημερολόγιο Συστήματος (Log)

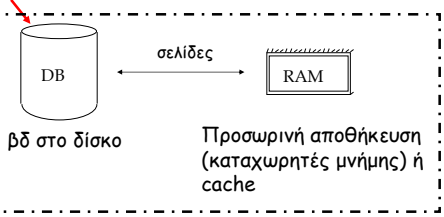
- Για να είναι δυνατή η ανάκαμψη από αποτυχίες, καταχωρούνται πληροφορίες για τις πράξεις των δοσοληψιών
- Αποθηκεύονται στο δίσκο
- Τύποι πληροφορίας: έναρξη δοσοληψίας
  εγγραφή στοιχείου (παλιά, νέα τιμή)
  ανάγνωση στοιχείου
  επικύρωση/ακύρωση

Αυξητικό ημερολόγιο

Ημερολόγιο Συστήματος (system log)

Πληροφορίες για κάθε δοσοληψία

το μοντέλο του συστήματος



Σημείο Ελέγχου

Γενικότερη έννοια από ότι στην ARIES

Εγγραφή όλων των τροποποιημένων σελίδων στο δίσκο

Ασαφή ή όχι (αναστολή εκτέλεσης των δοσοληψιών)

Δε χρειάζεται η επανάληψη (redo) όλων των πράξεων εγγραφής κατά την ανάκαμψη

Προεγγραφή Ημερολογίου

Το πρωτόκολλο προεγγραφής ημερολογίου (**W**rite **A**head **L**og)

(1) Η καταχώρηση στο log για μια εγγραφή γράφεται στο δίσκο πριν οι αντίστοιχες σελίδες να γραφούν στο δίσκο

(2) Πριν την επικύρωση (commit) μιας δοσοληψίας όλες οι καταχωρήσεις του log που την αφορούν γράφονται στο δίσκο

Το (1) δίνει ατομικότητα

Το (2) δίνει διάρκεια

Ημερολόγιο Συστήματος (Log)

Σειριακή εγγραφή στο log

Για κάθε εγγραφή στοιχείου, στο log μια καταχώρηση:

<XID, pageID, offset, length, old data, new data>

ARIES: Καταχωρήσεις Log

Πεδία LogRecord:

Προηγούμενο LSN για τη δοσοληψία

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

Μόνο για εγγραφές

Τύποι καταχωρήσεων

- Update
- Commit
- Abort
- End
- Compensation Log Records (CLRs)
  - για πράξεις UNDO

ARIES: Η Γενική Εικόνα

LogRecords

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

Σελίδες Δεδομένων

σε κάθε σελίδα pageLSN

Πίνακας Δοσοληψιών

lastLSN
status

Πίνακας Τροποποιημένων Σελίδων

recLSN

flushedLSN

ARIES: Ανάκαμψη από Αποτυχία

Τρεις Φάσεις

ΑΝΑΛΥΣΗ -- ανακατασκευή τον πινάκων του συστήματος: ποιες σελίδες είναι τροποποιημένες, ποια είναι η κατάσταση των δοσοληψιών

ΕΠΑΝΑΛΗΨΗ (REDO) -- επανάληψη *όλων* των δοσοληψιών

ΑΚΥΡΩΣΗ (UNDO) -- αναίρεση των δοσοληψιών που δεν ήταν επικυρωμένες

ARIES: Ανάκαμψη από Αποτυχία

- Άρχισε από ένα σημείο ελέγχου (που βρίσκεται μέσω του **master** record).
- Τρεις Φάσεις.
  - Εύρεση ποιες δοσοληψίες μετά το σημείο ελέγχου επικυρώθηκαν, ποιες ακυρώθηκαν (ΑΝΑΛΥΣΗ).
  - REDO ***όλες*** τις δοσοληψίες.
    - (repeat history)
  - UNDO το αποτέλεσμα των αποτυχημένων δοσοληψιών

Άσκηση 8

00 begin_checkpoint
10 end_checkpoint
20 update: T1 writes P5
40 T2 Commit
50 T2 end
60 update: T3 writes P3
70 T1 abort

Φάση της Ανάλυσης

T1	abort	70
T3	running	60

Πίνακας Δοσοληψιών

P5	20
P3	60

Πίνακας Τροποποιημένων Σελίδων

Φάση της Επανάληψης (REDO)

Ξεκινάμε από το 20, γιατί;

Φάση της Αναίρεσης (UNDO)

ToUndo = {70, 60}

Στο ARIES υποθέτουμε αυστηρό 2PL, άρα όχι διάδοση ανακλήσεων, εγγραφές

Γενικά μπορεί να χρειαστεί ανάκληση

(άρα στο log εκτός των εντολών εγγραφής και εντολές ανάγνωσης, για να ελέγξουμε *ποιες* πρέπει να ανακληθούν)