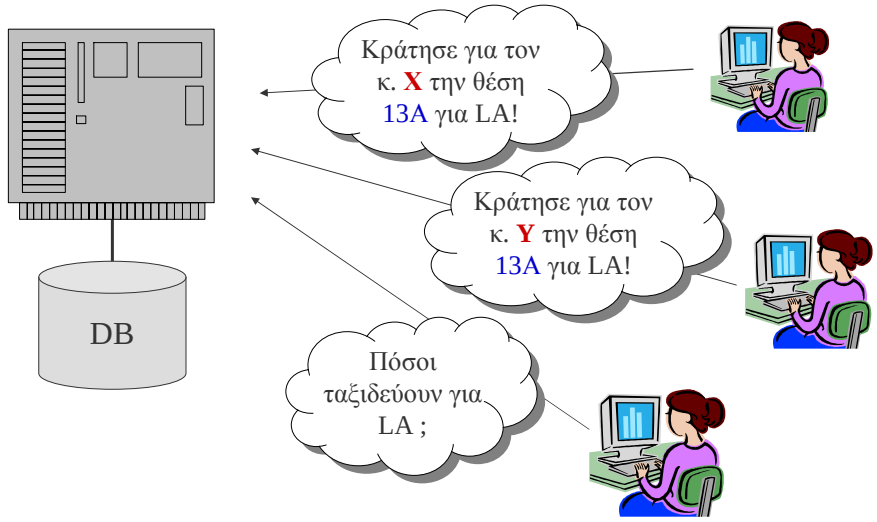
Δοσοληψίες & Ταυτοχρονισμός



1

Θεματολόγιο

- Η έννοια της δοσοληψίας
- Ιδιότητες των δοσοληψιών
- Καταστάσεις μιας δοσοληψίας
- Χρονοπρογράμματα
- Σειριοποιησιμότητα
- Έλεγχος σειριοποιησιμότητας

2

Δοσοληψία

- **Δοσοληψία** είναι
  - μια σειρά από ενέργειες, οι οποίες
  - διαβάζουν ή γράφουν
  - αντικείμενα της βάσης
- στα αγγλικά “transaction”
- σειρά: διατεταγμένο σύνολο, λίστα

Παράδειγμα δοσοληψίας

T_0: μεταφορά 50€ από το λογαριασμό Α στο λογαριασμό Β

```
read(A);
A := A - 50;
write(A);
read(B);
B := B + 50;
write(B).
```

Σε ότι αφορά τη ΒΔ:

```
R(A);W(A);R(B);W(B).
```

Προβληματισμός

- Δύο είναι τα βασικά προβλήματα με τις δοσοληψίες:
  - Τι θα γίνει αν κατά τη διάρκεια της εκτέλεσης, πέσει το σύστημα;
  - Τι θα γίνει αν δύο δοσοληψίες επιχειρούν να μεταβάλλουν το ίδιο αντικείμενο ταυτοχρόνως;

Ιδιότητες των δοσοληψιών

- **Ατομικότητα**: είτε **όλες** οι πράξεις της δοσοληψίας επιτυγχάνουν, είτε **όλες** αποτυγχάνουν.
- **Συνέπεια**: στο τέλος της δοσοληψίας, η βάση πρέπει να είναι σε συνεπή μορφή.
- **Απομόνωση**: ακόμα κι αν τρέχουν πολλές δοσοληψίες ταυτόχρονα, **κάθε δοσοληψία πρέπει να νομίζει ότι τρέχει μόνη της**.
- **Μονιμότητα**: αν η δοσοληψία επιτύχει, πρέπει το αποτέλεσμά της να επιβιώνει, ακόμα κι αν αποτύχει το σύστημα.

ACID Test

- (**A**)tomicity — Ατομικότητα
- (**C**)onsistency — Συνέπεια
- (**I**)solation — Απομόνωση
- (**D**)urability — Μονιμότητα

Διεθνώς γνωστό ως ACID test...

7

Συνέπεια

- Μια βάση δεδομένων διέπεται από κανόνες ακεραιότητας (π.χ. πρωτεύοντος κλειδιού ...)
- Επιπλέον, υπάρχουν και λογικοί περιορισμοί, τους οποίους «κρύβουμε» στις εφαρμογές.

- *Πριν και μετά την εκτέλεση της δοσοληψίας (αλλά όχι απαραίτητα ενδιαμέσως), όλοι οι περιορισμοί αυτοί, πρέπει να πληρούνται...*

8

Συνέπεια

EMP(EMP_ID,NAME,AGE,DEPT_ID,SALARY)

- Περιορισμοί ακεραιότητας:
  - EMP_ID πρωτεύον κλειδί
  - AGE <= 65
  - SALARY > 0

Συνέπεια

- Παράδειγμα λογικού περιορισμού:

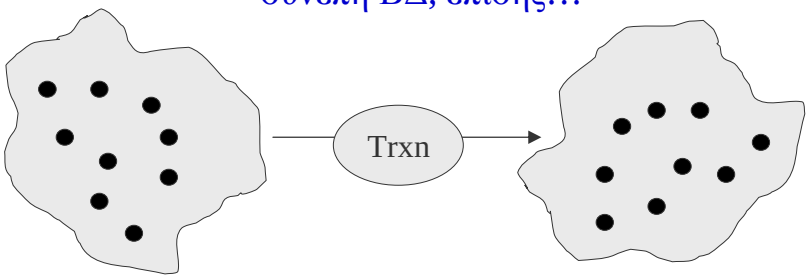
κατά τη διάρκεια της μεταφοράς χρημάτων από ένα λογαριασμό σε ένα άλλο,

το άθροισμα των δύο λογαριασμών στο τέλος, πρέπει να ισούται με το άθροισμα τους στην αρχή

- **Συνεπής** [κατάσταση της] ΒΔ: όλοι οι περιορισμοί ικανοποιούνται!

Συνέπεια

Κάνουμε και την υπόθεση ότι μια δοσοληψία που ξεκινά να τροποποιεί μια συνεπή ΒΔ, θα καταλήξει σε μια συνεπή ΒΔ, επίσης!!!



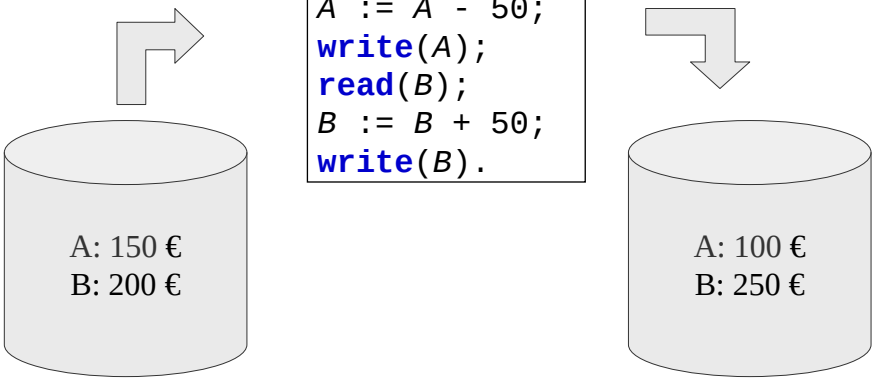
Συνεπής κατάσταση της βάσης πριν την δοσοληψία

Συνεπής κατάσταση της βάσης μετά την δοσοληψία

11

Συνέπεια

```
read(A);
A := A - 50;
write(A);
read(B);
B := B + 50;
write(B).
```



A+B=350 ⟷ A+B=350

12

ACID Test

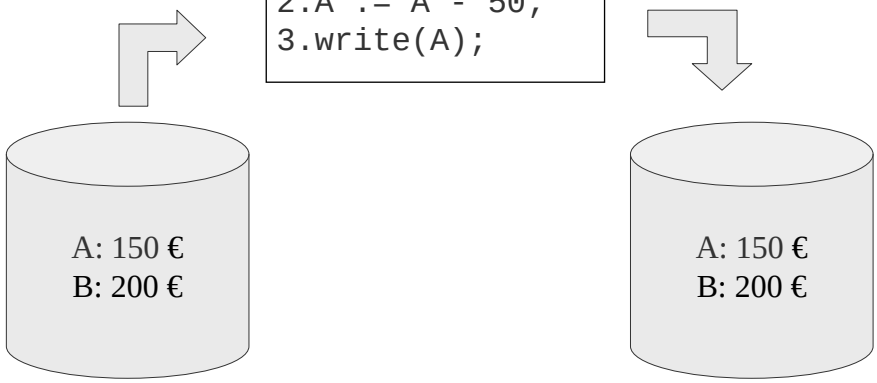
- (**A**)tomicity — Ατομικότητα ←
- (**C**)onsistency — Συνέπεια
- (**I**)solation — Απομόνωση
- (**D**)urability — Μονιμότητα

Ατομικότητα

- Η δοσοληψία είναι μια **μονάδα εργασίας**
- Παρότι αποτελείται από πολλές ενέργειες, δεν είναι αποδεκτό να εκτελεστούν μόνο μερικές από αυτές
- Αυτό μπορεί να συμβεί, π.χ., γιατί στη διάρκεια εκτέλεσης, το σύστημα αποτυγχάνει
- Σαν αποτέλεσμα, κάποιοι κανόνες μπορεί να παραβιαστούν (και μαζί και η συνέπεια της βάσης)...

Ατομικότητα

```
1.read(A);
2.A := A - 50;
3.write(A);
```

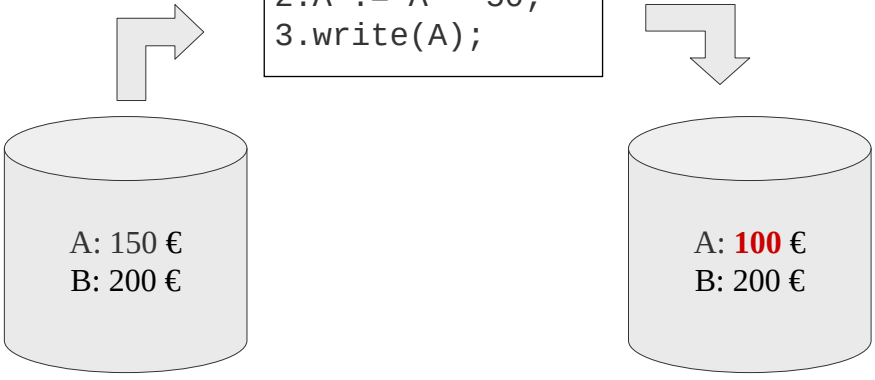


A+B=350 A+B=300

15

Ατομικότητα

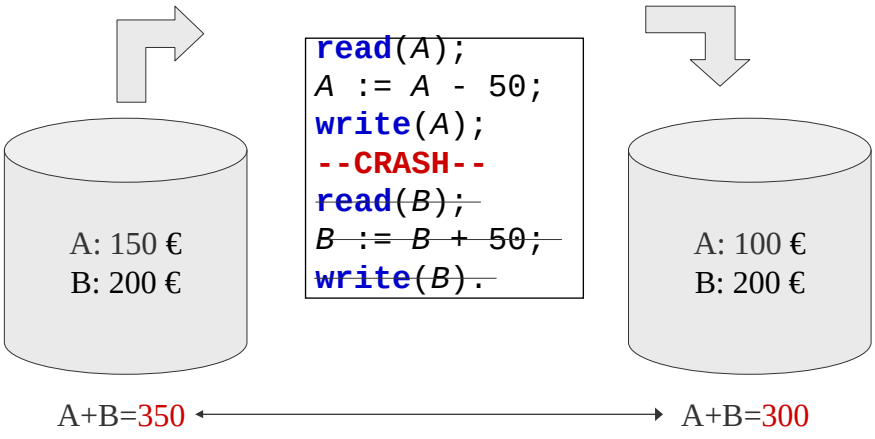
```
1.read(A);
2.A := A - 50;
3.write(A);
```



A+B=350 A+B=300

16

Ατομικότητα



17

Ατομικότητα

- Το DBMS εξασφαλίζει ότι η ατομικότητα θα διατηρηθεί, **αναιρώντας** όλες τις δοσοληψίες που αποτυγχάνουν
- Το πώς γίνεται αυτό, θα το δούμε στο κεφάλαιο της ανάνηψης...

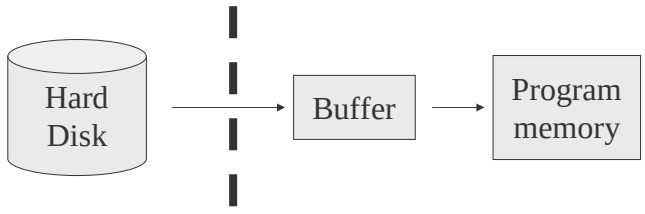
18

Παράπλευρη παρατήρηση

- Τι πάει να πει `read(A);`
- `A` είναι μια μεταβλητή του προγράμματος
- `read(A)` σημαίνει:
  - Διάβασε από το δίσκο την αντίστοιχη με το **A** εγγραφή στη βάση,
  - Φέρε την σε κάποιο buffer
  - Αντίγραψέ την στην περιοχή μνήμης του προγράμματος

19

Παράπλευρη παρατήρηση



- Το αντίστοιχο συμβαίνει και με τη `write`
- Όπως θα δούμε, το `A` μπορεί και να μην είναι εγγραφή, αλλά π.χ., σελίδα στο δίσκο ...

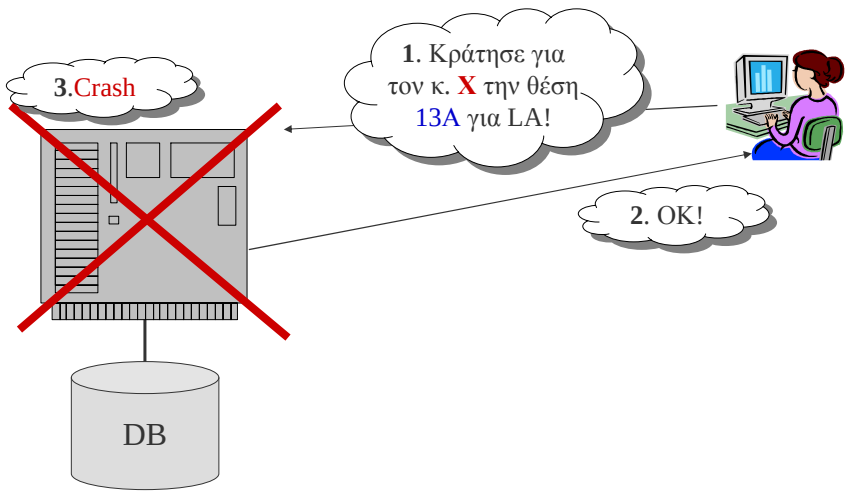
20

ACID Test

- (**A**)tomicity — Ατομικότητα
- (**C**)onsistency — Συνέπεια
- (**I**)solation — Απομόνωση
- (**D**)urability — Μονιμότητα ←

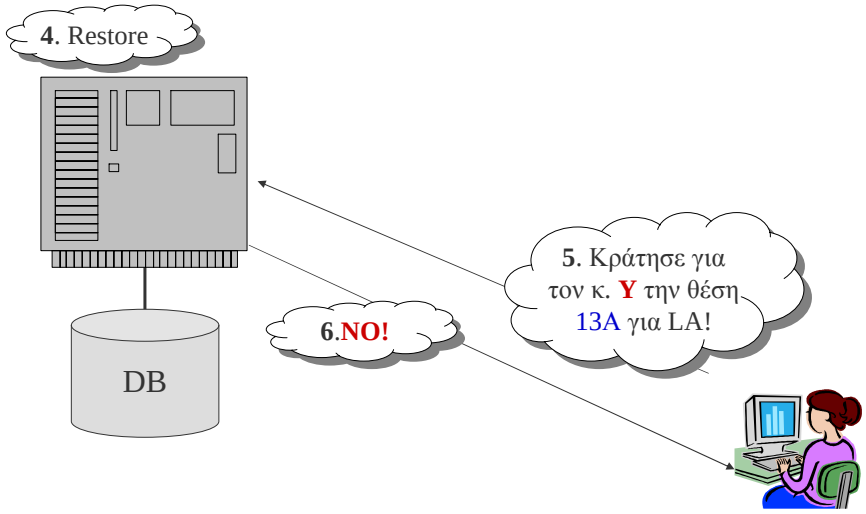
21

Μονιμότητα



22

Μονιμότητα



23

ACID Test

- (**A**)tomicity — Ατομικότητα
- (**C**)onsistency — Συνέπεια
- (**I**)solation — Απομόνωση ←
- (**D**)urability — Μονιμότητα

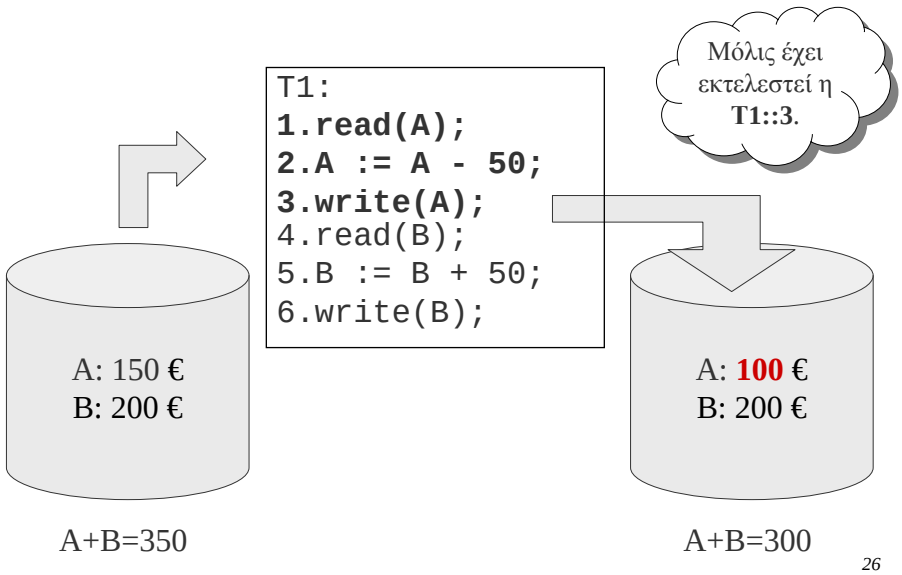
24

Απομόνωση

- Ας υποθέσουμε ότι στο σύστημα τρέχουν περισσότερες από μια δοσοληψίες ταυτοχρόνως.
- Αν μια από αυτές μπορέσει να δει τα ενδιάμεσα αποτελέσματα της άλλης, τότε μπορεί να έχουμε ανεπιθύμητα αποτελέσματα (διότι ενδιαμέσως στη δοσοληψία η βάση μπορεί να είναι ασυνεπής).
- Γι' αυτό, θα θέλαμε, ιδεατά, οι δοσοληψίες να τρέχουν η μία μετά την άλλη **σειριακά**.
- Για λόγους απόδοσης, όμως, αυτό δεν γίνεται ...

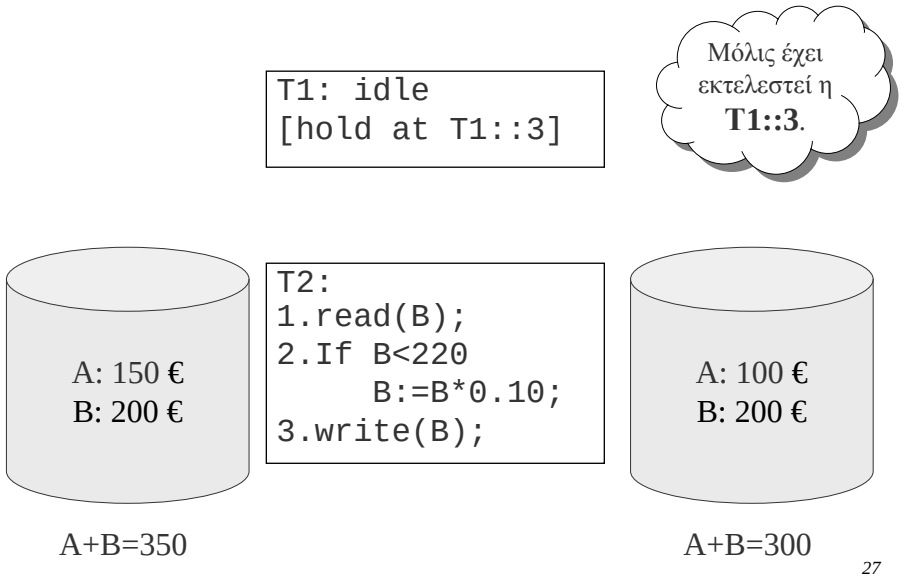
25

Απομόνωση



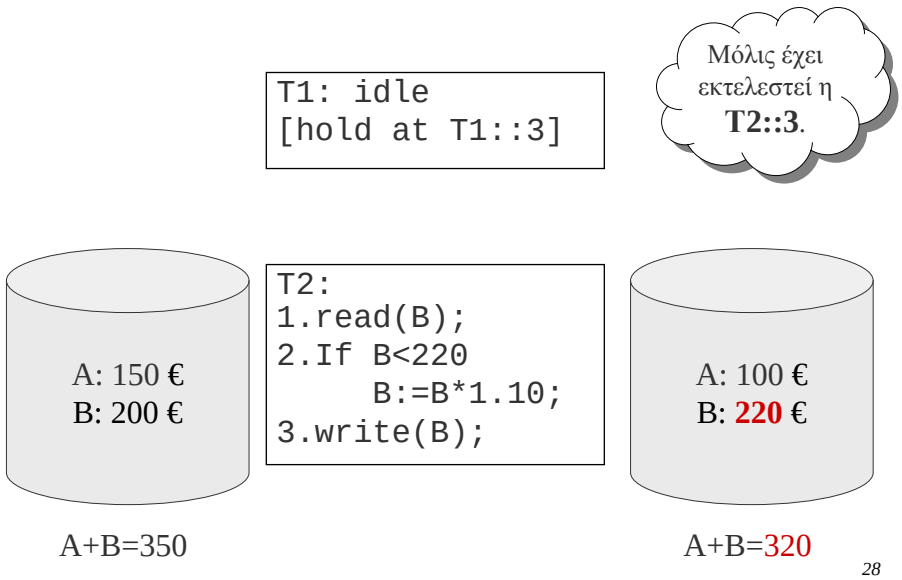
26

Απομόνωση



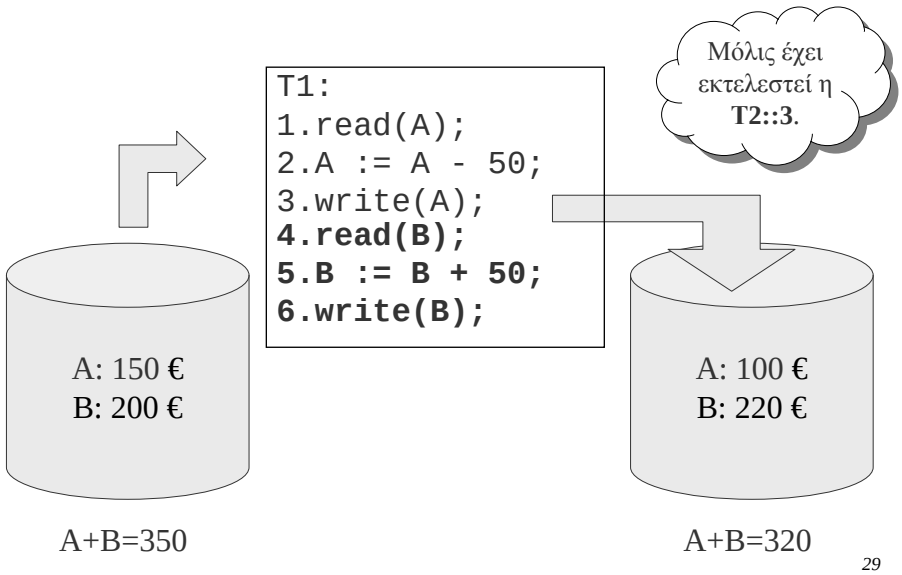
27

Απομόνωση



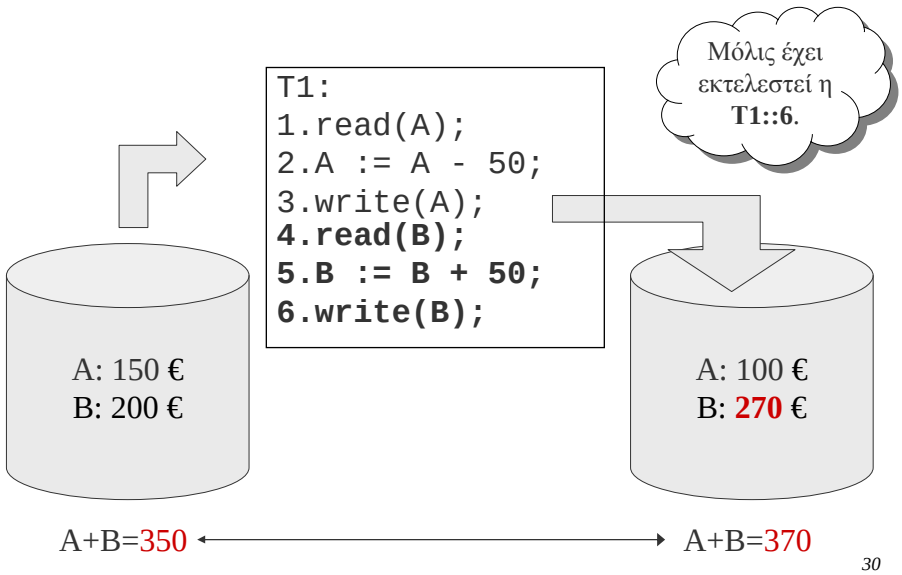
28

Απομόνωση



29

Απομόνωση



30

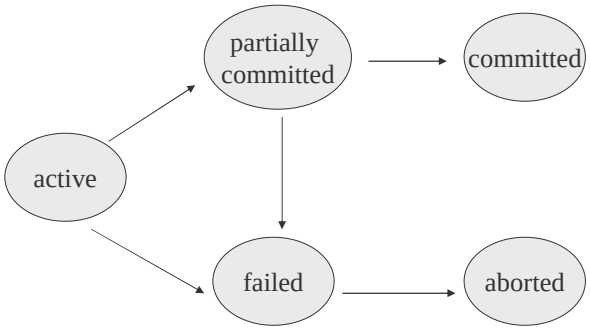
Απομόνωση

- Και γιατί να μην τρέχουμε σειριακά τις δοσοληψίες, τη μία μετά την άλλη;
  - Παράλληλη χρήση της CPU και του I/O
  - Οι σύντομες δοσοληψίες, δεν έχουν λόγο να αναμένουν την ολοκλήρωση των πιο χρονοβόρων
  - Έχουμε έξυπνους αλγόριθμους διαπλοκής των δοσοληψιών

Θεματολόγιο

- Η έννοια της δοσοληψίας
- Ιδιότητες των δοσοληψιών
- Καταστάσεις μιας δοσοληψίας
- Χρονοπρογράμματα
- Σειριοποιησιμότητα
- Έλεγχος σειριοποιησιμότητας

Καταστάσεις μιας δοσοληψίας



33

Καταστάσεις μιας δοσοληψίας

- **Active**: στο ξεκίνημα και κατά τη διάρκειά της
- **Failed**: όταν το DBMS αντιληφθεί ότι η δοσοληψία δεν μπορεί να συνεχίσει
- **Aborted**: όταν η αποτυχημένη δοσοληψία έχει αναιρεθεί από το σύστημα και η ΒΔ είναι σε συνεπή μορφή
- **Committed**: όταν η δοσοληψία επιτύχει και η ΒΔ είναι σε συνεπή μορφή.

34

Καταστάσεις μιας δοσοληψίας

- **Partially committed**: όταν έχει εκτελεστεί η τελευταία εντολή της δοσοληψίας

Λεπτή διάκριση με την committed, θα επανέλθουμε στο κεφάλαιο της ανάκαμψης...

35

Δοσοληψία – Ορθή επαναδιατύπωση

- **Δοσοληψία** είναι
  - μια σειρά από ενέργειες, οι οποίες
  - διαβάζουν ή γράφουν
  - αντικείμενα της βάσης
  - και η οποία τελειώνει είτε με COMMIT, είτε με ABORT

36

Συμβολισμός

- $R_X(A)$: η δοσοληψία X διαβάζει το αντικείμενο A
- $W_X(A)$: η δοσοληψία X γράφει το αντικείμενο A
- $COMMIT_X$: η δοσοληψία X τερματίζει επιτυχώς
- $ABORT_x$: η δοσοληψία X αποτυγχάνει

Π.χ.,

R_4(r3): η δοσοληψία T4 διαβάζει το αντικείμενο r3

37

Θεματολόγιο

- Η έννοια της δοσοληψίας
- Ιδιότητες των δοσοληψιών
- Καταστάσεις μιας δοσοληψίας
- Χρονοπρογράμματα
- Σειριοποιησιμότητα
- Έλεγχος σειριοποιησιμότητας

38

Χρονοπρογράμματα

- **Χρονοπρόγραμμα** είναι
  - μια σειρά από ενέργειες (read, write, commit, abort)
  - μιας ομάδας δοσοληψιών
  - όπου εμφανίζονται όλες οι ενέργειες αυτών των δοσοληψιών
  - διατηρώντας τη σειρά με την οποία εμφανίζονται σε κάθε δοσοληψία

Στην αγγλική: "schedule"

39

Παράδειγμα

- Δοσοληψία T_1: R(A);R(B);W(A);COMMIT
- Δοσοληψία T_2: R(A);R(B);W(B);COMMIT

- Schedule S_1:
  $R_1(A);R_1(B);W_1(A);C1;R_2(A);R_2(B);W_2(B);C2.$
- Schedule S_2:
  $R_2(A);R_2(B);W_2(B);C2;R_1(A);R_1(B);W_1(A);C1.$
- Schedule S_3:
  $R_1(A);R_1(B);R_2(A);W_1(A);R_2(B);C1;W_2(B);C2.$

40

Αντιπαράδειγμα

- Δοσοληψία T_1: R(A);R(B);W(A);COMMIT
- Δοσοληψία T_2: R(A);R(B);W(B);COMMIT

- Schedule S_4:
  $R_1(B);W_1(A);C1;R_2(A);R_2(B);W_2(B);C2.$
- Schedule S_5:
  $W_2(B);R_1(A);R_1(B);R_2(A);W_1(A);R_2(B);C1;C2.$

Χρονοπρόγραμμα

- Schedule S_3:
  $R_1(A);R_1(B);R_2(A);W_1(A);R_2(B);C1;W_2(B);C2.$

- Ένα χρονοπρόγραμμα περιγράφει *τι βλέπει το DBMS* και όχι τι προγραμματίζουμε εμείς!

Παράδειγμα

T1: μεταφέρει €50 από Α σε Β

Συνέπεια: Α+Β σταθερό

T_1	T_2
	read(*A*); *temp* := *A* * 0.1; *A* := *A* - *temp*; **write**(*A*); **read**(*B*); *B* := *B* + *temp*; **write**(*B*);
read(*A*); *A* := *A* - 50; **write**(*A*); **read**(*B*); *B* := *B* + 50; **write**(*B*).	

T2: μεταφέρει 10% του Α στο Β

43

Σειριακό Χρονοπρόγραμμα

Serial Schedule: Όταν οι ενέργειες που ανήκουν σε μια δοσοληψία εμφανίζονται κολλητά η μία με την άλλη

Εναλλακτικά: όταν οι συναλλαγές εκτελούνται εξ ολοκλήρου η μία μετά την άλλη

T_1	T_2
	read(*A*); *temp* := *A* * 0.1; *A* := *A* - *temp*; **write**(*A*); **read**(*B*); *B* := *B* + *temp*; **write**(*B*);
read(*A*); *A* := *A* - 50; **write**(*A*); **read**(*B*); *B* := *B* + 50; **write**(*B*).	

44

Σειριακό Χρονοπρόγραμμα

n! δυνατά σειριακά χρονοπρογράμματα για n δοσοληψίες

T_1	T_2
read(A);	
$A := A - 50$;	
write(A);	
read(B);	
$B := B + 50$;	
write(B);	
	read(A);
	$temp := A * 0.1$;
	$A := A - temp$;
	write(A); **read**(B);
	$B := B + temp$;
	write(B).

45

Σειριακό Χρονοπρόγραμμα

A:150, B:200, A+B: 350

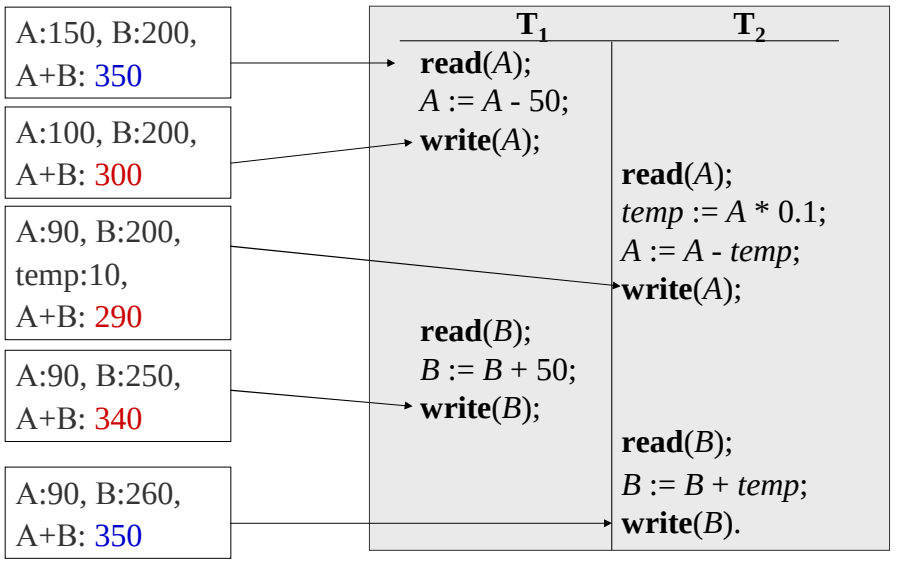
A:100, B:250, A+B: 350

A:90, B:260, A+B: 350

T_1	T_2
read(A);	
$A := A - 50$;	
write(A);	
read(B);	
$B := B + 50$;	
write(B);	
	read(A);
	$temp := A * 0.1$;
	$A := A - temp$;
	write(A); **read**(B);
	$B := B + temp$;
	write(B).

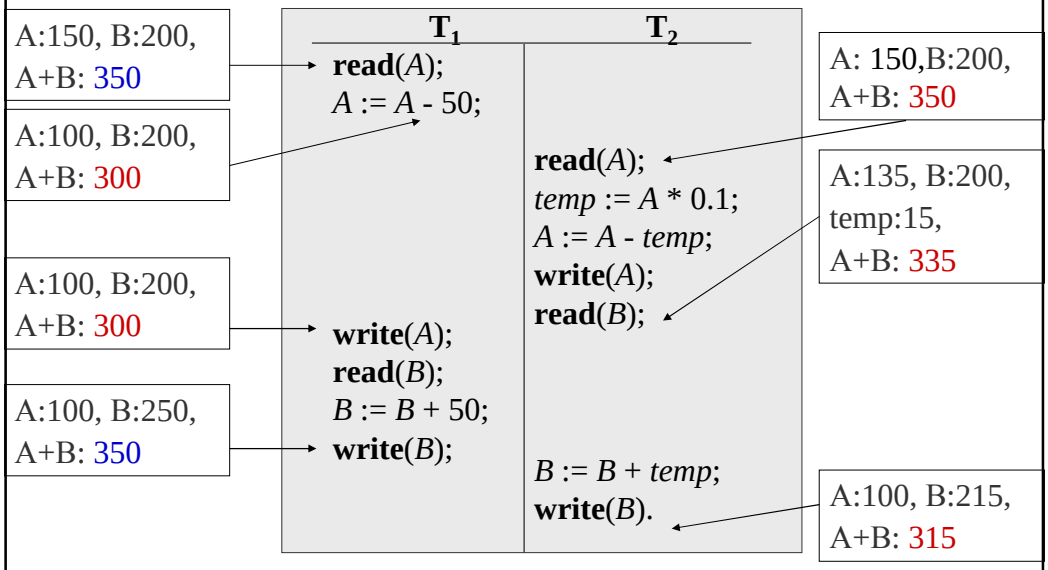
46

Μη Σειριακό Χρονοπρόγραμμα



47

Προβληματικό Χρονοπρόγραμμα



48

Προσοχή!!

- Κάθε δοσοληψία στο χρονοπρόγραμμα, είναι σαν συνάρτηση: έχει δικό της χώρο μνήμης [γι' αυτό και η τιμή των A, B εξαρτάται MONO από τα read, write και τις τοπικές μεταβολές –και όχι από τις αλλαγές σε άλλες δοσοληψίες]

49

3 ειδών προβλήματα με τα χρονοπρογράμματα

- Ασυνεπείς αναγνώσεις (dirty reads)
- Απώλειες ενημερώσεων (lost updates)
- Μη επαναλήψιμες αναγνώσεις (non-repeatable reads)

50

Παράδειγμα

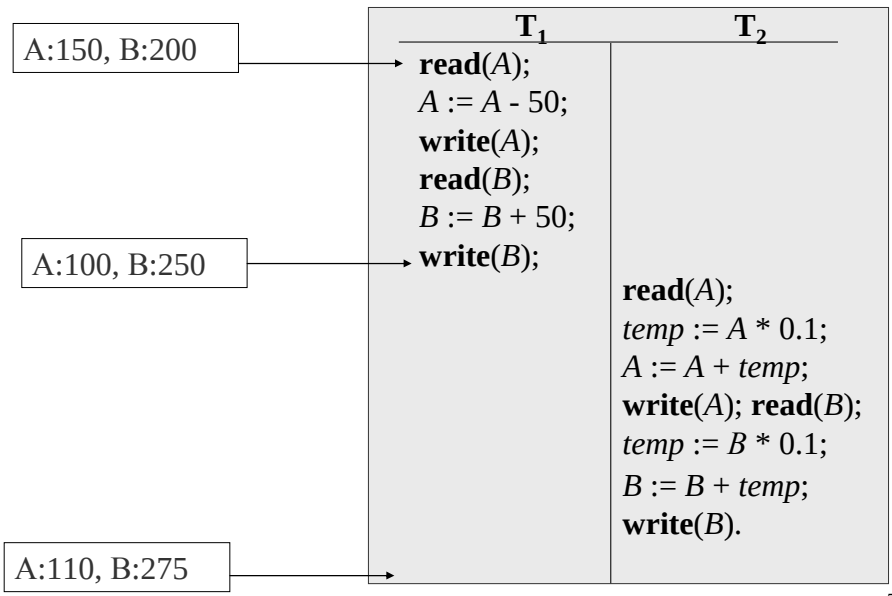
- T1: μεταφέρει 50 € από τον A στον B
- T2: κάνει αύξηση στον A και το B κατά 10%

Προσοχή: Δεν υφίσταται περιορισμός για το A+B, πλέον!

Σκοπός είναι να δείξουμε προβλήματα που μπορούν να προκύψουν...

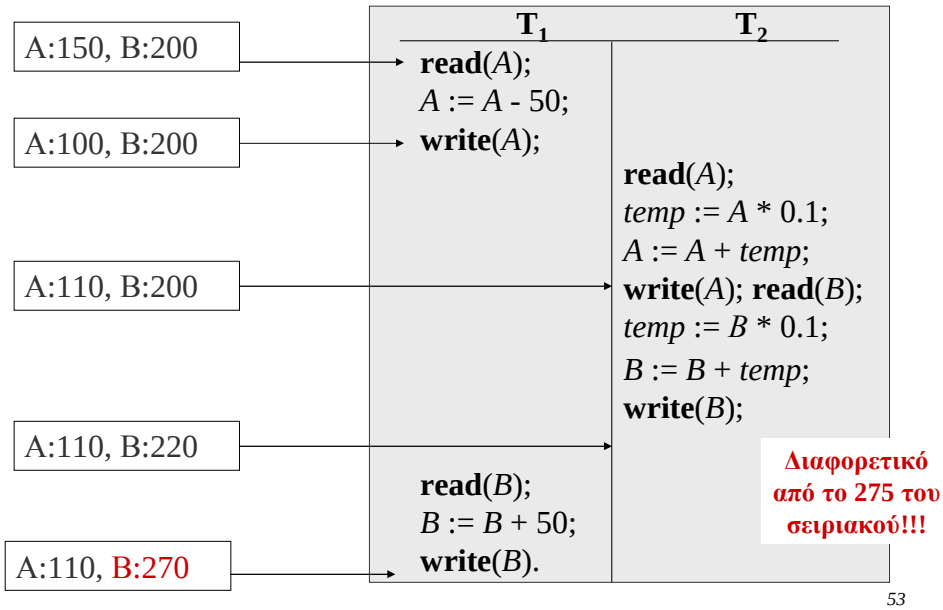
51

Σειριακό Χρονοπρόγραμμα



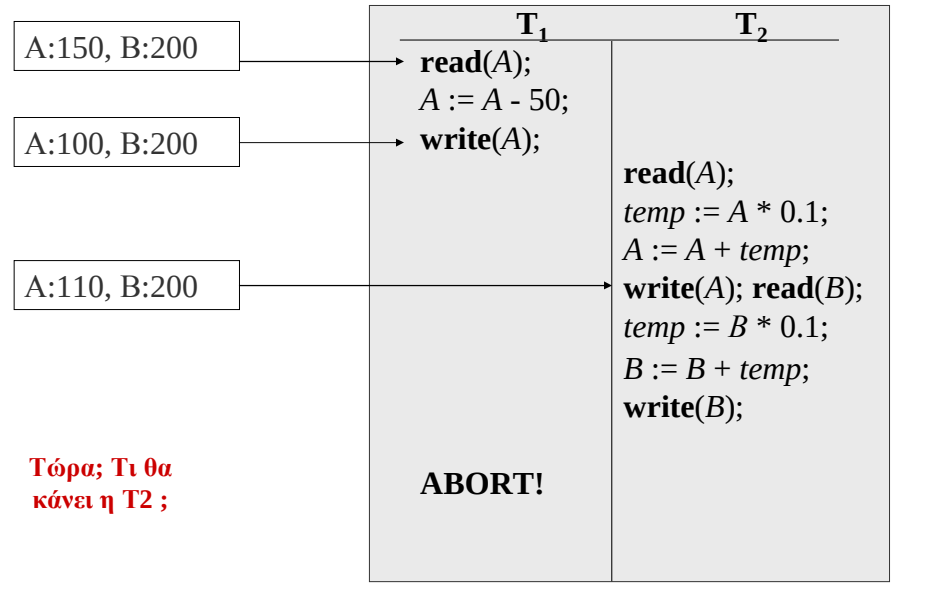
52

Ασυνεπής Ανάγνωση (Dirty Read)



53

Ασυνεπής Ανάγνωση (Dirty Read)



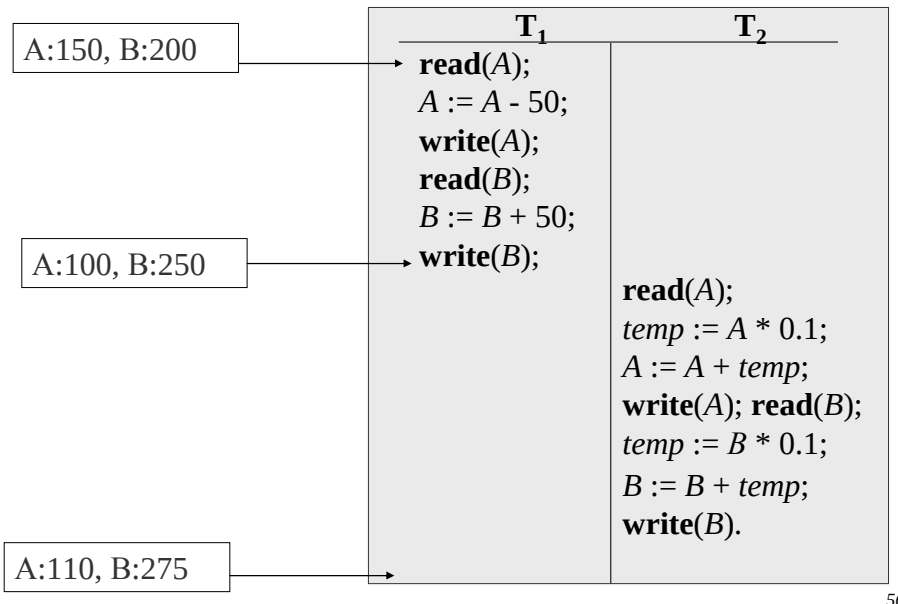
54

Dirty read

- Ο όρος προκύπτει από το γεγονός ότι η T2 διαβάζει μια τιμή για το A, ενώ η T1, η οποία είχε ξεκινήσει να το τροποποιεί, δεν έχει ολοκληρώσει ακόμα.
- Κατά συνέπεια, αν η T1 κάνει abort, πρέπει να κάνει και η T2 [ασχέτως που έχει ήδη ολοκληρώσει]...

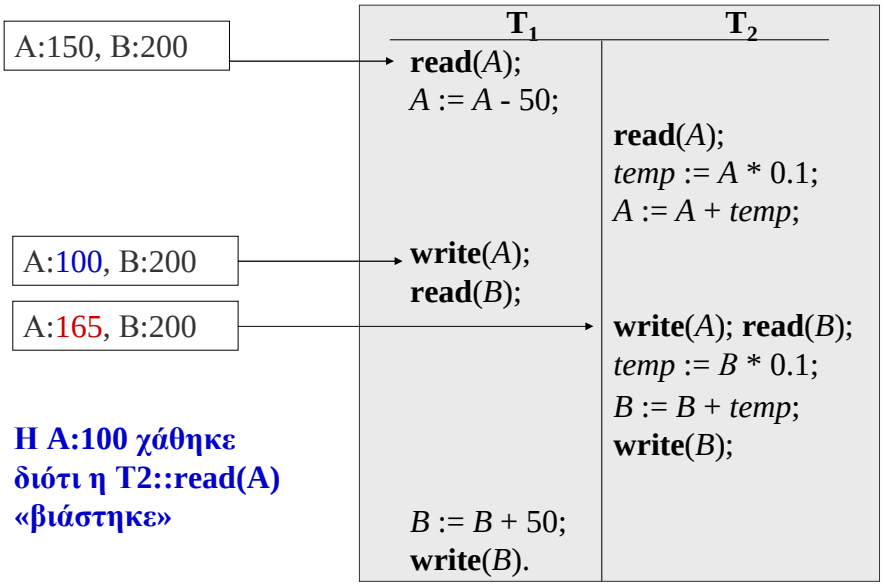
55

Σειριακό Χρονοπρόγραμμα



56

Απώλεια Ενημερώσεων (lost updates)



Η A:100 χάθηκε διότι η T2::read(A) «βιάστηκε»

57

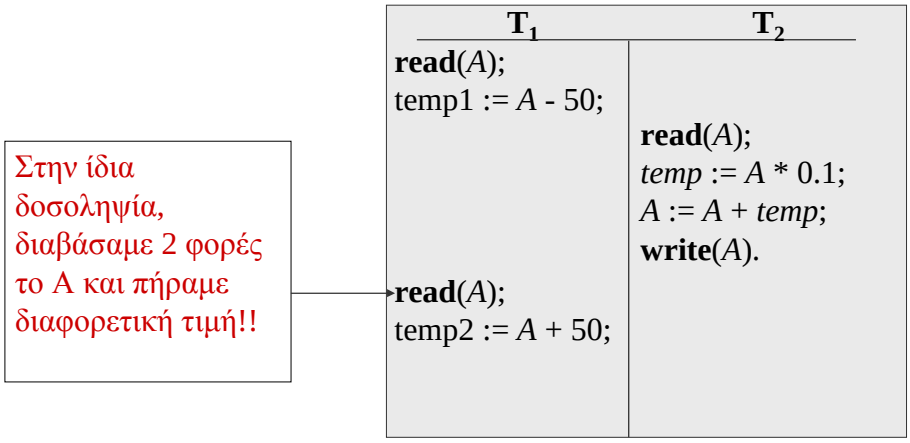
Μη επαναλήψιμες αναγνώσεις (Non-repeatable reads)

Ακόμα κι αν δεν έχει νόημα να διαβάσει 2 φορές το A, το T1 είναι ένα έγκυρο transaction

T_1	T_2
read(*A*);	
temp1 := *A* - 50;	
read(*A*);	
temp2 := *A* + 50;	
	read(*A*);
	temp := *A* * 0.1;
	A := *A* + *temp*;
	write(*A*).

58

Μη επαναλήψιμες αναγνώσεις (Non-repeatable reads)



Πλήρες χρονοπρόγραμμα

- **Πλήρες χρονοπρόγραμμα**: ένα χρονοπρόγραμμα που συμπεριλαμβάνει abort ή commit στο τέλος της κάθε δοσοληψίας.

Θεματολόγιο

- Η έννοια της δοσοληψίας
- Ιδιότητες των δοσοληψιών
- Καταστάσεις μιας δοσοληψίας
- Χρονοπρογράμματα
- Σειριοποιησιμότητα
- Έλεγχος σειριοποιησιμότητας

Σειριοποιησιμότητα

- Βασικό ζητούμενο είναι η συνέπεια της βάσης
- Η σειριακή εκτέλεση των δοσοληψιών (σειριακό χρονοπρόγραμμα) εγγυάται τη συνέπεια
- **Σειριοποιήσιμο χρονοπρόγραμμα**: ένα χρονοπρόγραμμα που εγγυημένα έχει το ίδιο αποτέλεσμα με ένα πλήρες σειριακό χρονοπρόγραμμα.

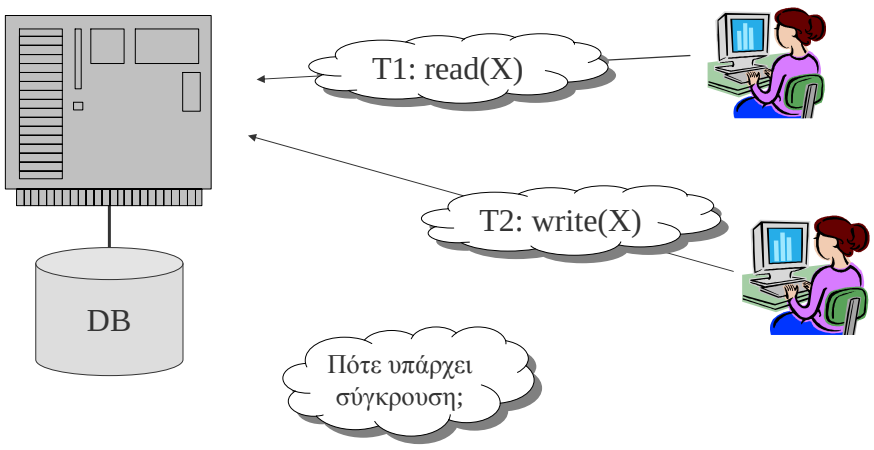
Απομόνωση: θυμάστε τι είναι;

Σειριοποιησιμότητα

- Κατά συνέπεια, αν μας δοθεί ένα χρονοπρόγραμμα, πρέπει να μπορέσουμε να αποφανθούμε αν είναι σειριοποιήσιμο ή όχι!
- Δύο τεχνικές:
  - Σειριοποιησιμότητα συγκρούσεων
  - Σειριοποιησιμότητα όψεως //δε θα επικεντρώσουμε!
- **Στο εξής, θα αγνοούμε το processing στη μνήμη και θα μας απασχολούν μόνο οι read και write αλληλεπιδράσεις των δοσοληψιών με τη βάση δεδομένων!!**

63

Σειριοποιησιμότητα συγκρούσεων



64

Συγκρούσεις

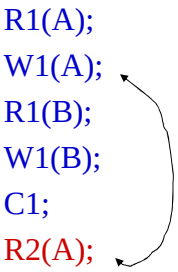
- Έστω δύο δοσοληψίες, T1 και T2, οι οποίες θέλουν να ενεργήσουν μέσα στο ίδιο χρονοπρόγραμμα πάνω στο ίδιο αντικείμενο A. Πότε θα το επιτρέψουμε; [Εναλλακτικά:, πότε συγκρούονται οι ενέργειές τους;]

T1\T2	Read	Write
Read	☑	☒
Write	☒	☒

Σειριοποιησιμότητα συγκρούσεων

- Δύο χρονοπρογράμματα καλούνται **ισοδύναμα συγκρούσεων** αν για κάθε σύγκρουση, οι συγκρουόμενες πράξεις έχουν την ίδια σειρά στα δύο προγράμματα.

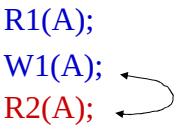
Σειριακό Χρονοπρόγραμμα S1



T_1	T_2
read(*A*);	
A := *A* - 50;	
write(*A*);	
read(*B*);	
B := *B* + 50;	
write(*B*);	
	read(*A*);
	temp := *A* * 0.1;
	A := *A* - *temp*;
	write(*A*); **read**(*B*);
	B := *B* + *temp*;
	write(*B*).

67

Μη Σειριακό Χρονοπρόγραμμα S2



T_1	T_2
read(*A*);	
A := *A* - 50;	
write(*A*);	
	read(*A*);
	temp := *A* * 0.1;
	A := *A* - *temp*;
	write(*A*);
read(*B*);	
B := *B* + 50;	
write(*B*);	
	read(*B*);
	B := *B* + *temp*;
	write(*B*).

68

Προβληματικό Χρονοπρόγραμμα S3

R1(A);
R2(A);
W2(A);
R2(B);
W1(A);
R1(B);
W1(B);
C1;
W2(B);
C2

T_1	T_2
read(*A*);	
A := *A* - 50;	
	read(*A*);
	temp := *A* * 0.1;
	A := *A* - *temp*;
	write(*A*);
	read(*B*);
write(*A*);	
read(*B*);	
B := *B* + 50;	
write(*B*);	
	B := *B* + *temp*;
	write(*B*).

69

Ισοδυναμία ;

- S1:
  R1(A);W1(A);R1(B);W1(B);C1;R2(A);W2(A);R2(B);W2(B);C2
- S2: order preserving
  R1(A);W1(A); R2(A);W2(A);R1(B);W1(B);C1;R2(B);W2(B);C2
- S3: order changed!
  R1(A); R2(A);W2(A); R2(B);W1(A); R1(B);W1(B);C1;W2(B);C2

70

Σειριοποιησιμότητα [τυπικά]

- Ένα χρονοπρόγραμμα είναι **σειριοποιήσιμο συγκρούσεων** αν είναι ισοδύναμο συγκρούσεων με ένα σειριακό.
- ... αν, δηλαδή, όλες οι συγκρουόμενες πράξεις έχουν την ίδια σειρά που θα είχαν σε ένα σειριακό...

71

Διαισθητικά [και όχι τυπικά]

- Στο σειριακό χρονοπρόγραμμα, κάθε δοσοληψία «ξεμπερδεύει» ξεχωριστά (σε απομόνωση) με κάθε αντικείμενο και μετά το «αναλαμβάνει» μια άλλη...
- Σε ένα σειριοποιήσιμο, το να ΜΗΝ υπάρχει σύγκρουση σημαίνει ότι το χρονοπρόγραμμα «ξεμπερδεύει» με τα αντικείμενα με την ίδια σειρά ανά δοσοληψία, με την οποία θα το έκανε και το σειριακό...

Επιβεβαιώστε με τα προηγούμενα...

72

Ερώτηση

T_1	T_2
read(*A*);	
write(*A*);	
	read(*B*);
	write(*B*);
read(*B*);	
write(*B*).	
	read(*A*);
	write(*A*).

Είναι σειριοποιήσιμο ή όχι;

Θεματολόγιο

- Η έννοια της δοσοληψίας
- Ιδιότητες των δοσοληψιών
- Καταστάσεις μιας δοσοληψίας
- Χρονοπρογράμματα
- Σειριοποιησιμότητα
- Έλεγχος σειριοποιησιμότητας

Γράφος Σειριοποιησιμότητας

- Μοντελοποιούμε τις συγκρούσεις ενός χρονοπρογράμματος με ένα γράφο, ο οποίος έχει:
  - Για κάθε δοσοληψία και ένα κόμβο
  - Μια κατευθυνόμενη ακμή από την δοσοληψία T_i στην δοσοληψία T_j, αν μια ενέργεια της T_i συγκρούεται με μια επακόλουθή της, της Tj.

Ήτοι, για κάθε σύγκρουση `Πράξη`$_i$`(X);Πράξη`$_j$`(X)` μια ακμή από τον προηγούμενο κόμβο i στον επόμενο κόμβο j

75

Γράφος σειριοποιησιμότητας

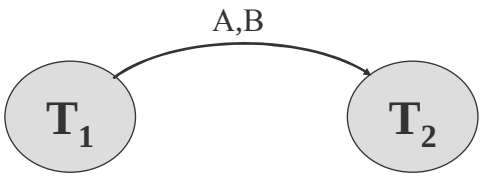
- Για ευκολία, μπορούμε να σημειώνουμε και το αντικείμενο για το οποίο οι δύο δοσοληψίες συγκρούονται ...

76

Γράφος σειριοποιησιμότητας

- S1:

  R1(A);W1(A);R1(B);W1(B);C1;R2(A);W2(A);R2(B);W2(B);C2



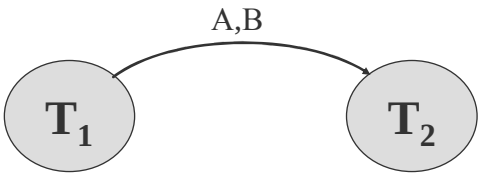
Θυμίζω: ακμή από τον προηγούμενο κόμβο i στον επόμενο κόμβο j

77

Γράφος σειριοποιησιμότητας

- S2:

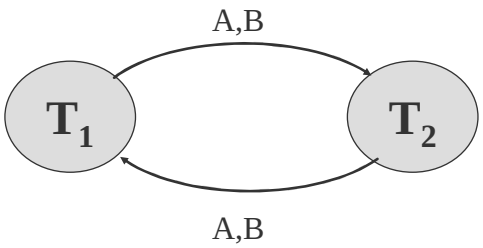
  R1(A);W1(A); R2(A);W2(A);R1(B);W1(B);C1;R2(B);W2(B);C2



78

Γράφος σειριοποιησιμότητας

- S3:
  R1(A); R2(A);W2(A); R2(B);W1(A); R1(B);W1(B);C1;W2(B);C2

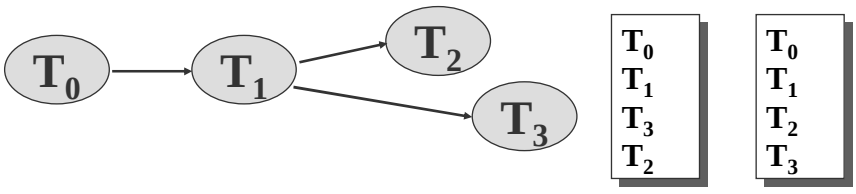


79

Θεώρημα

- Γράφος με κύκλο είναι μη σειριοποιήσιμος σε σχέση με τις συγκρούσεις
- Γράφος χωρίς κύκλο είναι σειριοποιήσιμος σε σχέση με τις συγκρούσεις

Το ισοδύναμο σειριακό πρόγραμμα προκύπτει από την τοπολογική ταξινόμηση του γράφου.



80

Ερώτηση

T_1	T_2
read(A);	
write(A);	
	read(B);
	write(B);
read(B);	
write(B).	
	read(A);
	write(A).

Είναι σειριοποιήσιμο ή όχι;

81

Θεώρημα

Σειριοποιησιμότητα συγκρούσεων ⇔ έλλειψη κύκλου στο γράφο

- Υπάρχει όμως και άλλη εκδοχή σειριοποιησιμότητας, η σειριοποιησιμότητα όψεως...

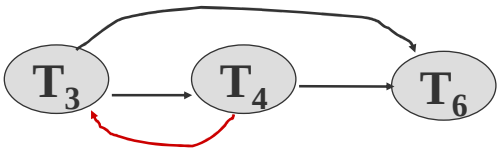
82

Σειριοποιησιμότητα όψεως

- Ποιος ο γράφος σειριοποίησης του παρακάτω χρονοπρογράμματος;

T_3	T_4	T_6
read(Q)		
	write(Q)	
write(Q)		
		write(Q)

Σειριοποιησιμότητα όψεως



T_3	T_4	T_6
read(Q)		
	write(Q)	
write(Q)		
		write(Q)

Σειριοποιησιμότητα όψεως

- Και λοιπόν; Αφού ούτως ή άλλως, σημασία έχει τι γράφει η T6...

T_3	T_4	T_6
read(Q)		
	write(Q)	
write(Q)		
		write(Q)

Σειριοποιησιμότητα συγκρούσεων και όψεων

- Κάθε χρονοπρόγραμμα που είναι σειριοποιήσιμο σε σχέση με συγκρούσεις, είναι σειριοποιήσιμο όψεως.
- Το αντίστροφο ΔΕΝ ισχύει.
- *Κάθε χρονοπρόγραμμα που είναι σειριοποιήσιμο όψεως, και ΔΕΝ είναι σειριοποιήσιμο σε σχέση με συγκρούσεις, περιέχει* ***τυφλές εγγραφές*** *(writes που δεν έχει προηγηθεί read γι' αυτές στην δοσοληψία τους)*

Ελέγχοντας την σειριοποιησιμότητα όψεως

- Ο αλγόριθμος είναι NP-complete και κατά συνέπεια, όχι πρακτικός

Εν γένει, σε σχέση με τη σειριοποιησιμότητα όψεως, στο πλαίσιο του μαθήματος, το πολύ να σας ζητηθεί να υποψιαστείτε αν ένα χρονοπρόγραμμα είναι σειριοποιήσιμο...

87

Αλγόριθμος για view serializability

- Από τον γράφο προτεραιότητας με ετικέτες στις ακμές,
- προσπάθησε να βρεις ένα συνεκτικό ακυκλικό γράφο,
- διαλέγοντας από κάθε ζεύγος ακμών με ίδια ετικέτα, μία εκ των δύο...

88