Επισκόπηση της Διαχείρισης Συναλλαγών

Κεφάλαιο 16

Συναλλαγές

- Η ταυτόχρονη εκτέλεση προγραμμάτων χρηστών είναι απαραίτητη για την καλή επίδοση του DBMS.
  - Επειδή οι προσπελάσεις του δίσκου είναι συχνές και σχετικά αργές, είναι σημαντικό να διατηρούμε τη CPU απασχολημένη με την ταυτόχρονη εκτέλεση πολλών προγραμμάτων χρηστών.
- Ένα πρόγραμμα χρήστη μπορεί να εκτελεί πολλές πράξεις πάνω στα δεδομένα που ανακτά από τη βάση, αλλά το DBMS ενδιαφέρεται μόνο για τα δεδομένα που διαβάζονται από τη βάση ή γράφονται στη βάση.
- Η *συναλλαγή* είναι ο τρόπος που βλέπει το DBMS ένα πρόγραμμα χρήστη: ως μια ακολουθία αναγνώσεων και εγγραφών.

Ταυτοχρονισμός στο DBMS

- Οι χρήστες εκτελούν συναλλαγές, και θεωρούν ότι κάθε συναλλαγή εκτελείται από μόνη της.
  - Ο ταυτοχρονισμός επιτυγχάνεται από το DBMS, που "πλέκει" τις ενέργειες (διάβασε/γράψε αντικείμενα της βάσης) διάφορων συναλλαγών.
  - Κάθε συναλλαγή πρέπει να αφήνει τη βάση σε συνεπή κατάσταση, εφόσον αυτή ήταν συνεπής κατά την έναρξη της συναλλαγής.
    - Τα DBMS επιβάλλουν κάποιους περιορισμούς ακεραιότητας, ανάλογα με τις δηλώσεις των εντολών CREATE TABLE.
    - Πέραν αυτού, το DBMS δεν καταλαβαίνει τη σημασιολογία των δεδομένων. (π.χ., δεν καταλαβαίνει πώς υπολογίζεται ο τόκος σε έναν τραπεζικό λογαριασμό).
- *Θέματα:* Αποτέλεσμα *πλεξίματος* συναλλαγών και *καταρρεύσεις του συστήματος*.

Ατομικότητα Συναλλαγών

- Μια συναλλαγή μπορεί να *ολοκληρώσει* αφού εκτελέσει όλες τις ενέργειές της, ή μπορεί να *εγκαταλείψει* (ή να εγκαταληφθεί από το DBMS) αφού εκτελέσει μερικές ενέργειες της.
- Μια πολύ σημαντική ιδιότητα που εγγυάται το DBMS για όλες τις συναλλαγές είναι το ότι αυτές είναι *ατομικές.* Δηλαδή, ο χρήστης μπορεί να θεωρήσει ότι μια συναλλγή πάντα εκτελεί όλες τις ενέργειές της ή καμία ενέργεια.
  - Το DBMS *καταγράφει* όλες τις ενέργειες ώστε να μπορεί να *αναιρέσει* τις ενέργειες εγκαταληφθησών συναλλαγών.

Παράδειγμα

- Θεωρείστε δυο συναλλαγές:

```
T1:    BEGIN   A=A+100,   B=B-100   END
T2:    BEGIN   A=1.06*A,   B=1.06*B   END
```

- Διαισθητικά, η πρώτη συναλλαγή μεταφέρει $100 από τον λογαριασμό B στον A. Η δεύτερη πιστώνει και τους δυο λογαριασμούς με 6% τόκους.
- Δεν υπάρχει εγγύηση ότι η T1 θα εκτελεσθεί πριν την T2 ή το αντίθετο, αν δοθούν στο σύστημα μαζί. Όμως, το τελικό αποτέλεσμα πρέπει να είναι ισοδύναμο με κάποια σειριακή εκτέλεση των δυο αυτών συναλλαγών.

Παράδειγμα (συνέχεια)

- Έστω η παρακάτω πεπλεγμένη εκτέλεση *(χρονοπρόγραμμα)*:

T1:	A=A+100,		B=B-100
T2:		A=1.06*A,	B=1.06*B

- Αυτό είναι OK. Αλλά τί γίνεται με το παρακάτω;

T1:	A=A+100,		B=B-100
T2:		A=1.06*A, B=1.06*B	

- Η οπτική του DBMS για το 2ο χρονοπρόγραμμα:

T1:	R(A), W(A),		R(B), W(B)
T2:		R(A), W(A), R(B), W(B)	

Χρονοπρογραμματισμός συναλλαγών

- *Σειριακό χρονοπρόγραμμα:* Δεν πλέκει τις ενέργειες διαφορετικών συναλλαγών.
- *Ισοδύναμα χρονοπρογράμματα*: Για κάθε κατάσταση της βάσης, το αποτέλεσμα (πάνω στα αντικείμενα της βάσης) της εκτέλεσης του 1ου χρονοπρογράμματος είναι το ίδιο με την εκτέλεση του 2ου χρονοπρογράμματος.
- *Σειριποιήσιμα χρονοπρογράμματα*: Ένα χρονοπρόγραμμα είναι ισοδύναμο κάποια σειριακή εκτέλεση των συναλλαγών.

(Σημείωση: Αν κάθε συναλλαγή διατηρεί τη συνέπεια, κάθε σειριοποιήσιμο χρονοπρόγραμμα τη διατηρεί.)

Ανωμαλίες πεπλεγμένης εκτέλεσης

- Διαβάζονται μη Ολοκληρωμένα Δεδομένα (Σύγκρουση WR, “dirty reads”):

```
T1:   R(A), W(A),                          R(B), W(B), Abort
T2:                  R(A), W(A), C
```

- Μη επαναλαμβανόμενες Αναγνώσεις - Unrepeatable Reads (Συγκρούσεις RW):

```
T1:   R(A),                          R(A), W(A), C
T2:            R(A), W(A), C
```

Ανωμαλίες (συνέχεια)

- Επανεγγραφή Μη-Ολοκληρωμένων Δεδομένων - Overwriting Uncommitted Data (Συγκρούσεις WW):

```
T1:     W(A),                   W(B), C
T2:             W(A), W(B), C
```