Αποθήκευση Δεδομένων

ΜΕΡΟΣ Β':
Το «εσωτερικό» ενός ΣΔΒΔ

Εισαγωγή

Δομή ενός ΣΔΒΔ

Λογισμικό για τη διαχείριση

Σύστημα Διαχείρισης ΒΔ

ΒΔ

Αρχεία με τις σχέσεις

+

Κατάλογος του συστήματος

Τυπικά, κάθε σχέση σε ένα αρχείο στο δίσκο

Δεδομένα αποθηκευμένα στο δίσκο

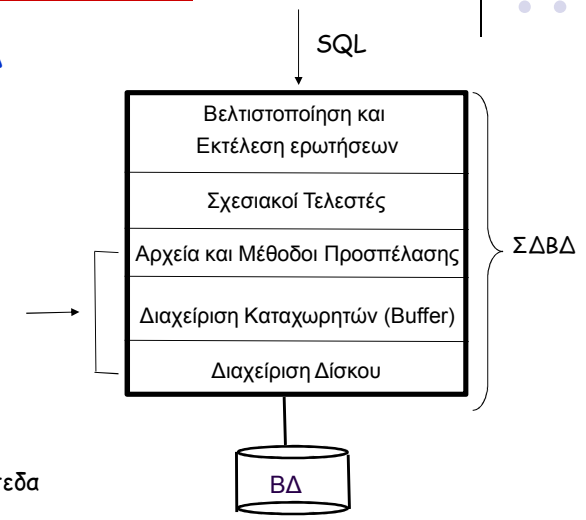
ΜΕΡΟΣ Β':
Το «εσωτερικό» ενός ΣΔΒΔ

Δομή ενός ΣΔΒΔ

Η (εσωτερική) αρχιτεκτονική ενός ΣΔΒΔ είναι σε επίπεδα

Σήμερα θα δούμε:
- Αποθήκευση
- Δομή αρχείων

Στη συνέχεια
- Τα παραπάνω επίπεδα



Η βάση δεδομένων θα πρέπει να αποθηκευτεί σε κάποιο αποθηκευτικό μέσο

Ιεραρχία αποθήκευσης

πρωτεύουσα αποθήκευση (primary storage)

κύρια μνήμη (main memory) - κρυφή μνήμη (cache)

- άμεση προσπέλαση από την κύρια ΚΜΕ (CPU)
- γρήγορη προσπέλαση
- περιορισμένη χωρητικότητα αποθήκευσης

Δευτερεύουσα αποθήκευση

(μαγνητικοί δίσκοι, ταινίες, δισκέτες, κλπ)

- για την επεξεργασία των δεδομένων **απαιτείται η μεταφορά των δεδομένων στην πρωτεύουσα αποθήκευση**
- πιο αργή προσπέλαση
- μεγάλη χωρητικότητα
- μικρότερο κόστος (για την ίδια ποσότητα χώρου η κύρια μνήμη 100 φορές ακριβότερη από τη δευτερεύουσα)

Οι περισσότερες βάσεις δεδομένων αποθηκεύονται σε δευτερεύουσες αποθηκευτικές μονάδες κυρίως σε δίσκους

- πολύ μεγάλες (10-100 TB) ⇒ μεγάλο κόστος ($1/GB – 100$/GB)
- μόνιμη αποθήκευση (nonvolatile storage)

Μαγνητικές ταινίες για

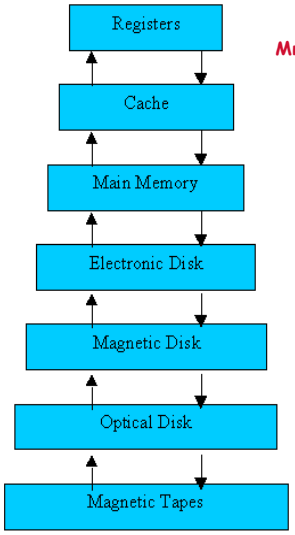
- τήρηση εφεδρικών αντιγράφων
- αρχειοθέτηση (archiving) (δεδομένα που θέλουμε να κρατήσουμε για πολύ καιρό αλλά η προσπέλαση τους είναι σπάνια)

Ιεραρχία Αποθήκευσης

- Κύρια μνήμη (RAM) για τα δεδομένα σε χρήση

- Δίσκοι για τη βδ (δευτερεύουσα αποθήκευση)

-Ταινίες για παλιές εκδοχές δεδομένων (tertiary storage).

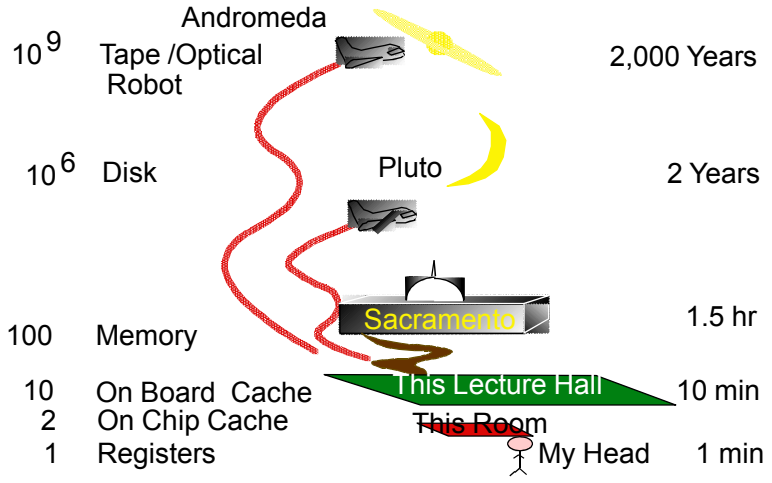


Μικρότερες, Γρηγορότερες

Μεγαλύτερες, Πιο αργές

Source: Operating Systems Concepts 5th Edition

Η αναλογία του Jim Gray's για latency (χρόνο απόκρισης) αποθήκευσης



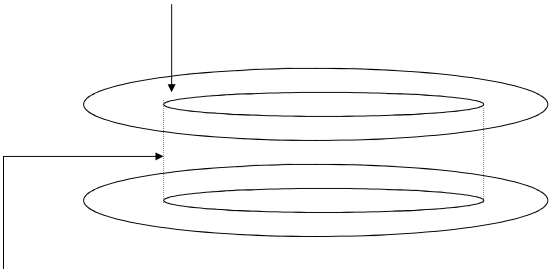
Μαγνητικοί Δίσκοι

- Μαγνητισμός μιας περιοχής του δίσκου κατά ορισμένο τρόπο ώστε 1 ή 0
- *Χωρητικότητα* (capacity) σε Kbyte - Mbyte - Gbyte
- Μαγνητικό υλικό σε σχήμα κυκλικού δίσκου

- Απλής και διπλής όψης

Σε πακέτα δίσκων

Οι πληροφορίες σε ομόκεντρους κύκλους διαφορετικής διαμέτρου: **άτρακτοι track** (συνήθως κάθε άτρακτος την ίδια ποσότητα πληροφορίας)

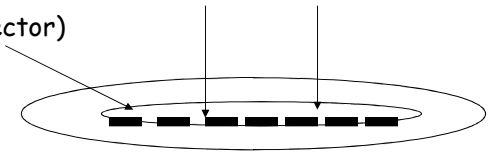


Ομόκεντροι κύκλοι σε διαφορετικές επιφάνειες: **κύλινδρος (cylinder)**

Μαγνητικοί Δίσκοι

Block (μονάδα μεταφοράς)

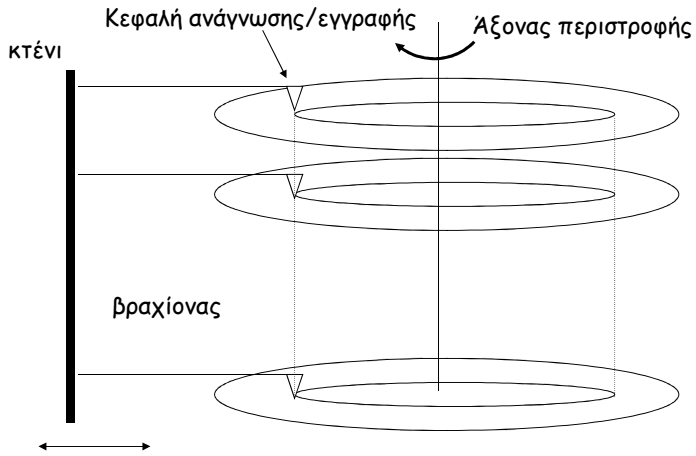
Τομέας (sector)



Κάθε άτρακτος χωρίζεται σε τόξα που ονομάζονται τομείς (sectors) και είναι χαρακτηριστικό του κάθε δίσκου και δε μπορεί να τροποιηθεί

Το μέγεθος ενός block τίθεται κατά την αρχικοποίηση του δίσκου και είναι κάποιο πολλαπλάσιο του τομέα

Μαγνητικοί Δίσκοι



χρόνος εντοπισμού (seek time) Τοποθέτηση κεφαλής στη σωστή άτρακτο 0.3 – 10

χρόνος περιστροφής (rotational delay ή latency) Ώσπου η αρχή του σωστού block να βρεθεί κάτω από την κεφαλή

χρόνος μεταφοράς block (block transfer time) χρόνος μεταφοράς δεδομένων από το δίσκο στη μνήμη

Χρόνος προσπέλασης = χρόνος εντοπισμού + χρόνος περιστροφής + χρόνος μεταφοράς

Μεταφορά αρκετών γειτονικών block

Παράδειγμα IBM Deskstar 14GPX **Seegate Barracuda 7200.9**

Χωρητικότητα: 14.4 GB **80 – 500 GB**

(μέσος) Χρόνος Εντοπισμού: 9.1 msec **11ms**

(2.2 για γειτονικά - 15.5 μέγιστο)

(μέσος) Χρόνος Περιστροφής: 4.17 msec **4.16ms**

5 διπλής όψης κυκλικούς δίσκους - 7,200 περιστροφές το λεπτό **7,200**

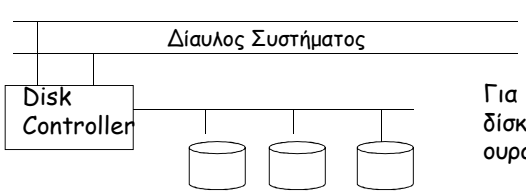
Χρόνος Μεταφοράς 13MB ανά sec **300MB ανά sec (σειριακός)**

Χρόνος προσπέλασης από το δίσκο ~ 10 msec (micro 10^{-6}) ενώ για θέσης μνήμης 60 nanosecond (nano 10^{-9})

Συνήθως μόνο μία κεφαλή τη φορά

Disk controller

- λειτουργίες εγγραφής/ανάγνωσης
- υπολογισμός αθροίσματος ελέγχου (checksum)



Για διαμοιραζόμενους δίσκους: χρόνος στην ουρά του controller

Συμπεράσματα

1. Τα δεδομένα πρέπει να βρίσκονται στη μνήμη

2. Η μονάδα μεταφοράς από το δίσκο στη μνήμη είναι ένα block. Το διάβασμα ή γράψιμο ενός block ονομάζεται λειτουργία Εισόδου/Εξόδου (Input/Output – I/O)

3. Ο χρόνος προσπέλασης (εγγραφής ή ανάγνωσης) ενός block διαφέρει και εξαρτάται από τη θέση του block

χρόνος προσπέλασης = χρόνος εντοπισμού + χρόνου περιστροφής + χρόνος μεταφοράς

Μαγνητικές Ταινίες

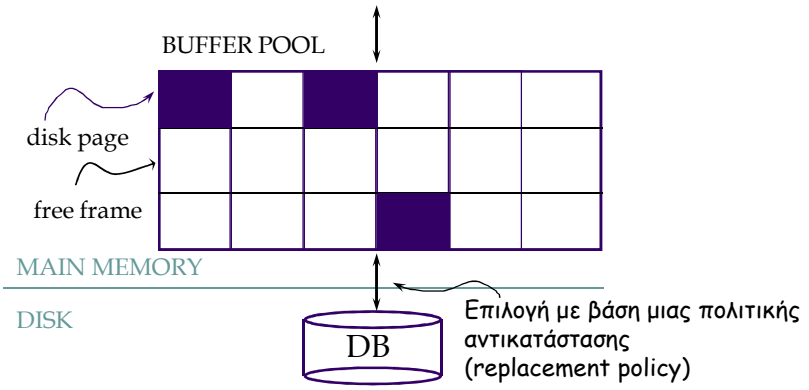
- Δίσκοι *τυχαίας προσπέλασης* (random access)
- Ταινίες *σειριακής προσπέλασης* (serial access) για να διαβάσουμε το *n*-οστό block πρέπει να ξεκινήσουμε από την αρχή και να διαβάσουμε και τα *n-1* blocks

Μεταφορά block σε ενδιάμεση μνήμη

Ενώ γίνεται η μεταφορά των δεδομένων από την δευτερεύουσα στην κύρια μνήμη - παράλληλα και ανεξάρτητα η ΚΜΕ μπορεί να επεξεργάζεται δεδομένα

Ένας ανεξάρτητος επεξεργαστής Εισόδου/Εξόδου ή πολλαπλοί επεξεργαστές

Μεταφορά block σε ενδιάμεση μνήμη (καταχωρητές)

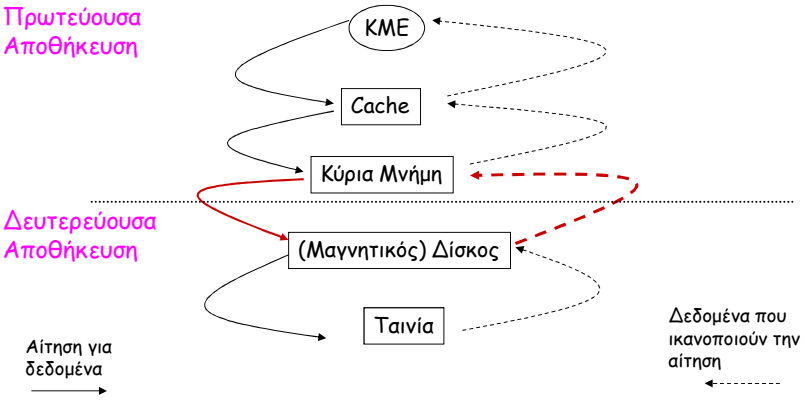


Μεταφορά block σε ενδιάμεση μνήμη

RAID: πλεονάζουσες συστοιχίες ανεξάρτητων δίσκων

(καταμερισμός δεδομένων και πλεονασμός)

Ιεραρχία Μνήμης



Σύγχρονα Αποθηκευτικά Μέσα

Flash memory (solid state) δευτερεύουσα αποθήκευση κυρίως σε φορητές συσκευές

- Μεγαλύτερη αντοχή από μαγνητικούς δίσκους, πιο ελαφριά, γρηγορότερη προσπέλαση (access time)
- Δεν έχει χρόνο εντοπισμού και περιστροφής
- Τρεις λειτουργίες: Read, Write, Erase
- Πριν γίνει εγγραφή, πρέπει να προηγηθεί Erase
- Erase και Write πολύ πιο αργά από το Read

Οργάνωση Αρχείων

Αρχεία

- Τα δεδομένα συνήθως αποθηκεύονται σε αρχεία στο δίσκο
- Για να επεξεργαστούμε τα δεδομένα θα πρέπει αυτά να βρίσκονται στη μνήμη.
- Η μεταφορά δεδομένων από το δίσκο στη μνήμη και από τη μνήμη στο δίσκο γίνεται σε *μονάδες blocks*
- Το διάβασμα ή γράψιμο ενός block ονομάζεται λειτουργία Εισόδου/Εξόδου (Input/Output – I/O)

Βασικός στόχος η ελαχιστοποίηση της επικοινωνίας με το δίσκο:

ελαχιστοποίηση του αριθμού των blocks **που μεταφέρονται μεταξύ της πρωτεύουσας (κύριας μνήμης, cache – ενδιάμεση μνήμη – buffers-καταχωρητές) και της δευτερεύουσας αποθήκευσης (δίσκος)**

Αρχεία

Τα δεδομένα συνήθως αποθηκεύονται με τη μορφή εγγραφών

Οι εγγραφές συνήθως περιγράφουν οντότητες (σχέσεις) και τα γνωρίσματά τους

Ένα αρχείο είναι λογικά οργανωμένο σε μια ακολουθία από εγγραφές που μπορεί να βρίσκονται αποθηκευμένες σε πολλές σελίδες (pages) – θα θεωρούμε page = block

Blobs

Εγγραφές

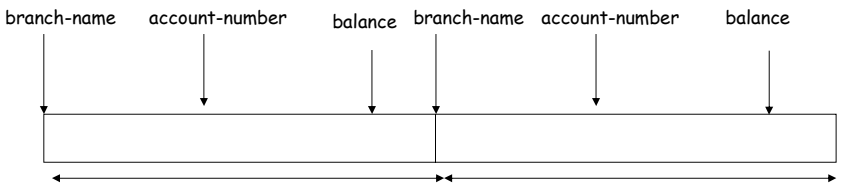
Πως οργανώνονται τα πεδία μέσα σε μία εγγραφή

Εγγραφές σταθερού και μεταβλητού μήκους

```
type film = record
        branch-name: char(22);
        account-number: char(20);
        balance:real;
        end
```

Έστω κάθε char 1 byte - real 8 bytes

Κάθε εγγραφή 50 bytes



Εγγραφές

Γιατί είναι προτιμότερες οι εγγραφές σταθερού μήκους:

εύκολος ο εντοπισμός ενός πεδίου και η διατήρηση πληροφορίας για «άδειες» θέσεις

Εγγραφές

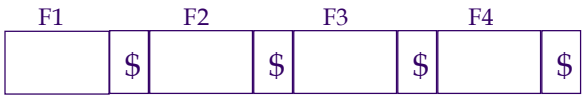
Πως προκύπτουν οι εγγραφές μεταβλητού τύπου;

Στο σχεσιακό μοντέλο κάθε εγγραφή (πλειάδα) μιας σχέσης περιέχει το ίδιο πλήθος πεδίων (αριθμό γνωρισμάτων). Άρα

- Εγγραφές του ίδιου τύπου αλλά έχουν ένα ή περισσότερα πεδία *μεταβλητού μεγέθους*
  - Ανάμεικτο (mixed) αρχείο: εγγραφές διαφορετικού τύπου

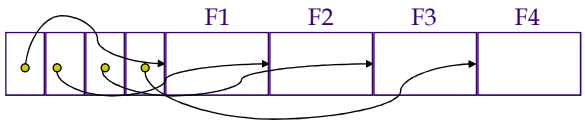
Εγγραφές

- Αποθήκευση των πεδίων συνεχόμενα, χωρισμένα με διαχωριστές (ειδικούς χαρακτήρες που δεν εμφανίζονται ως δεδομένα)



Εγγραφές

- Χώρο στην αρχή κάθε εγγραφής – πίνακας ακεραίων I[j] όπου j η μετατόπιση (offset) της j-οστής εγγραφής (κρατά την αρχή του j-οστού πεδίου) + τη μετατόπιση του τέλους της εγγραφής
  - απευθείας πρόσβαση σε οποιαδήποτε πεδίο
  - καλό χειρισμό της τιμής null



Εγγραφές

- Ως εγγραφές σταθερού μήκους, θεωρώντας το μέγιστο μέγεθος για κάθε εγγραφή

Παράγοντας Ομαδοποίησης

Η μονάδα μεταφοράς μεταξύ δίσκου και μνήμης είναι ένα block δίσκου

Έστω εγγραφές σταθερού μήκους

Όταν B ≥ R περισσότερες από μια εγγραφή ανά block - κάθε εγγραφή σε ένα μόνο block

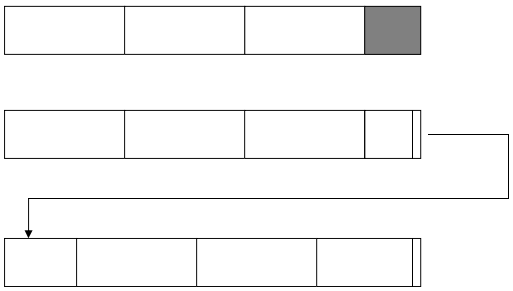
Παράγοντας ομαδοποίησης (blocking factor), όταν B ≥ R

bfr = ⌊ (B / R) ⌋, όπου B μέγεθος block σε byte και R μέγεθος εγγραφής σε bytes

Δηλαδή, πόσες «ολόκληρες» εγγραφές χωρούν σε ένα block

Εκτεινόμενη και μη εκτεινόμενη καταχώρηση εγγραφών

- **Μη εκτεινόμενη** (unspanned) οργάνωση:οι εγγραφές δεν επιτρέπεται να διασχίζουν τα όρια ενός block
  - Αχρησιμοποίητος χώρος: B - bfr * R bytes ανά block
  - Πιο εύκολη η προσπέλαση
- **Εκτεινόμενη** (spanned) οργάνωση: αποθήκευση μέρους μιας εγγραφής σε ένα block και το υπόλοιπο σε ένα άλλο block - δείκτης στο τέλος του πρώτου τμήματος δείχνει στο block που περιέχει το υπόλοιπο



b: Αριθμός blocks για την αποθήκευση ενός αρχείου r εγγραφών:

$$b = \lceil (r/bfr) \rceil$$

Τοποθέτηση block αρχείου στο δίσκο

συνεχόμενη τοποθέτηση (contiguous allocation) τα block του αρχείου τοποθετούνται σε διαδοχικά blocks του δίσκου

συνδεδεμένη τοποθέτηση (linked allocation) κάθε block του αρχείου περιλαμβάνει ένα δείκτη προς το επόμενο block του αρχείου

Εύκολη επέκταση - πιο αργή ανάγνωση όλου του αρχείου

συστάδες διαδοχικών blocks δίσκου (τμήματα (segments) ή επεκτάματα (extents)

ευρετηριοποιημένη τοποθέτηση (indexed allocation)

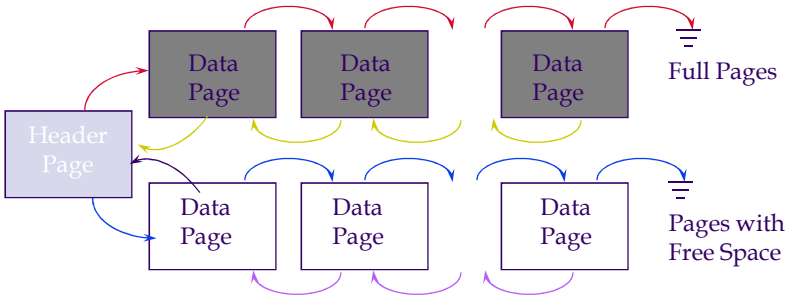
Επικεφαλίδες αρχείων

Μια **επικεφαλίδα ή περιγραφέας αρχείου** (file header ή file descriptor) περιέχει πληροφορίες σχετικά με ένα αρχείο που είναι απαραίτητες στα προγράμματα που προσπελαύνουν τις εγγραφές του αρχείου

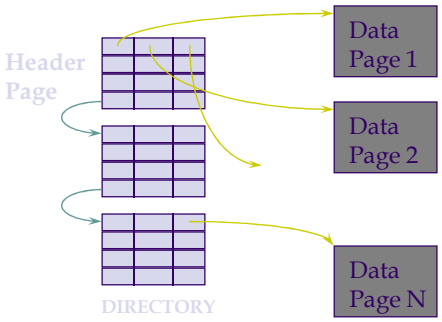
Πληροφορίες για προσδιορισμό διεύθυνσης των blocks αρχείου στο δίσκο + περιγραφές μορφοποίησης εγγραφών

Αποθηκεύεται στο αρχείο

Θεωρούμε ότι «ξέρουμε» σε ποιο block είναι αποθηκευμένη η i-οστή σελίδα (block) του αρχείου



Αρχεία



Αποθήκευση (επανάληψη)

Βασικά Σημεία

1. Τα δεδομένα αποθηκεύονται σε αρχεία στο δίσκο
2. Για να γίνει η επεξεργασία τους πρέπει να μεταφερθούν στη μνήμη
3. Η μονάδα μεταφοράς από το δίσκο στη μνήμη είναι ένα block
4. Ο χρόνος προσπέλασης (εγγραφής ή ανάγνωσης) ενός block διαφέρει και εξαρτάται από τη θέση του block – *δε θα το εξετάσουμε στο μάθημα*
5. Μας ενδιαφέρει η ελαχιστοποίηση του I/O (πολυπλοκότητα σε σχέση με blocks)

Οργάνωση Αρχείων (επανάληψη)

Ένα αρχείο είναι λογικά οργανωμένο σε μια ακολουθία από εγγραφές

Συνήθως ένα αρχείο ανά (σχήμα) σχέσης και μια εγγραφή αντιστοιχεί σε μια πλειάδα

Μη εκτεινόμενη (unspanned) οργάνωση:

οι εγγραφές δεν επιτρέπεται να διασχίζουν τα όρια ενός block

(-) Αχρησιμοποίητος χώρος

(+) Πιο εύκολη η προσπέλαση

Οργάνωση Αρχείων (επανάληψη)

Έστω B μέγεθος block σε byte και R μέγεθος εγγραφής σε bytes

Παράγοντας ομαδοποίησης (blocking factor), όταν $B \geq R$

$$bfr = \lfloor (B / R) \rfloor$$

Πόσες εγγραφές χωρούν σε ένα block

b: Αριθμός blocks για την αποθήκευση ενός αρχείου r εγγραφών:

$$b = \lceil (r/bfr) \rceil$$

Κατάλογος Συστήματος

Για κάθε σχέση:

όνομα, αρχείο, δομή αρχείου (πχ αρχείο σωρού)
Όνομα και τύπο για κάθε γνώρισμα
Όνομα ευρετηρίου για κάθε ευρετήριο
Περιορισμοί ακεραιότητας

Για κάθε ευρετήριο:
Δομή (πχ B+ δέντρο) και κλειδιά αναζήτησης

Για κάθε όψη:
Το όνομα και τον ορισμό της

Επίσης, στατιστικά, μέγεθος του buffer pool, δικαιώματα προσπέλασης κλπ.

Ο κατάλογος αποθηκεύεται επίσης ως σχέση

Κατάλογος Συστήματος

Attr_Cat(attr_name, rel_name, type, position)

attr_name	rel_name	type	position
attr_name	Attribute_Cat	string	1
rel_name	Attribute_Cat	string	2
type	Attribute_Cat	string	3
position	Attribute_Cat	integer	4
sid	Students	string	1
name	Students	string	2
login	Students	string	3
age	Students	integer	4
gpa	Students	real	5
fid	Faculty	string	1
fname	Faculty	string	2
sal	Faculty	real	3

Αποθήκευση Δεδομένων (γενικά)

Παραδοσιακά,

- Κάθε σχέση (το στιγμιότυπο της) αποθηκεύεται σε ένα αρχείο
- Η αποθήκευση είναι οριζόντια: κάθε πλειάδα της σχέσης αντιστοιχεί σε μια εγγραφή του αρχείου
  - Δηλαδή, ένα αρχείο είναι μια ακολουθία από πλειάδες

Σύγχρονη Τάση: Column stores (κάθετη αποθήκευση, ανά στήλη)

Αποθήκευση Δεδομένων (παράδειγμα)

Έστω μία σχέση R(A, B, C, D, E), τα γνωρίσματα A, B, D και E είναι τύπου ακέραιοι μεγέθους 16 bytes και το γνώρισμα C σειρά χαρακτήρων μεγέθους 36 bytes. Έστω αρχείο με r_A = 30.000 εγγραφές, μέγεθος block B = 1024 bytes, και μη εκτεινόμενη καταχώρηση.

Μέγεθος αρχείου δεδομένων: 3.000 blocks

Θα συζητήσουμε πως πρέπει να οργανώσουμε τις εγγραφές σε ένα αρχείο για αποδοτική επεξεργασία ερωτήσεων

Βασικές λειτουργίες:

- Εισαγωγή/διαγραφή/τροποποίηση εγγραφής
- Εντοπισμός (αναζήτηση) μια συγκεκριμένης εγγραφής με βάση συνθήκη ισότητας ή διαστήματος τιμών
- Διάσχιση (scan) όλων των εγγραφών του αρχείου

Βασικός στόχος η *ελαχιστοποίηση του αριθμού* των blocks που μεταφέρονται

Θεωρούμε ότι η πληροφορία για τη θέση στο δίσκο ενός block υπάρχει (π.χ., στην επικεφαλίδα του αρχείου)

Σε πραγματικά συστήματα

- Ίσως και άλλοι τύποι κόστους (πχ κόστος CPU)
- Πρόσβαση κατά block (διάβασμα γειτονικών block με μια μόνο αίτηση I/O: αναζήτηση 1ου block + μεταφορά όλων των επόμενων)

Στα επόμενα, αναφέρεται και το κόστος επεξεργασίας (αλλά γενικά θα το αγνοούμε)

B blocks - R εγγραφές ανά block - T_D εγγραφή/ανάγνωση - T_C χρόνος επεξεργασίας ανά εγγραφή

T_D = 15 milliseconds -- T_C = 100 nanoseconds

Οργάνωση αρχείων: πως είναι τοποθετημένες οι εγγραφές ενός αρχείου όταν αποθηκεύονται στο δίσκο

1. Αρχεία Σωρού
2. Ταξινομημένα Αρχεία

   Φυσική διάταξη των εγγραφών ενός αρχείου με βάση την τιμή ενός από τα πεδία του το οποίο λέγεται **πεδίο διάταξης** (ordering field)

Αρχεία Σωρού

Αρχείο Σωρού (heap file ή pile file): Οι εγγραφές τοποθετούνται στο αρχείο με τη σειρά που εισάγονται

Μη ταξινομημένο αρχείο

1. **Εισαγωγή**

   $2 * T_D + T_C$

2. **Αναζήτηση** (μέσος χρόνος)

   $0.5 * B * (T_D + R * T_C)$

B blocks

R εγγραφές ανά block

T_D χρόνος μεταφοράς block

T_C χρόνος επεξεργασίας ανά εγγραφή

3. **Διαγραφή εγγραφής**

   Σημάδι διαγραφής

   Περιοδική αναδιοργάνωση

   Χρόνος Αναζήτησης + (T_C + T_D)

4. Τροποποίηση εγγραφής

- εγγραφή μεταβλητού μήκους

5. Σάρωση (scan) Ανάγνωση όλων των εγγραφών

$B*(T_D+R*T_C)$

6. Ανάγνωση όλων των εγγραφών σε διάταξη

Εξωτερική ταξινόμηση συνήθως μια παραλλαγή της ταξινόμησης με συγχώνευση

Ταξινομημένα/Διατεταγμένα Αρχεία

Φυσική διάταξη των εγγραφών ενός αρχείου με βάση την τιμή ενός από τα πεδία του το οποίο λέγεται **πεδίο διάταξης** (ordering field)

Διατεταγμένο ή φυσικό αρχείο

- Αν το πεδίο διάταξης είναι και κλειδί τότε λέγεται και **κλειδί διάταξης**

1. Εισαγωγή

i. Εύρεση της σωστής θέσης της εγγραφής στο αρχείο

ii. Μετακίνηση εγγραφών για να κάνουμε χώρο για την εισαγωγή της

Κατά μέσο όρο μετακίνηση των μισών εγγραφών

Χρόνος αναζήτησης +

$2 * (0.5 * B *(T_D + R*T_C))$

B blocks
R εγγραφές ανά block
T_D χρόνος μεταφοράς block
T_C χρόνος επεξεργασίας ανά εγγραφή

1. Εισαγωγή (συνέχεια)

- Διατήρηση κάποιου αχρησιμοποίητου χώρου ανά block
- Δημιουργία ενός προσωρινού μη διατεταγμένου αρχείου (αρχείο υπερχείλισης) + κυρίως αρχείο

2. Αναζήτηση εγγραφής (με επιλογή ισότητας)

αποδοτική αν η συνθήκη αναζήτησης είναι στο πεδίο ταξινόμησης

Έστω B blocks, αναζήτηση της εγγραφής με τιμή K στο πεδίο διάταξης

Σημείωση: Υποθέτουμε ότι οι διευθύνσεις των blocks του αρχείου είναι αποθηκευμένες στην επικεφαλίδα του αρχείου

2. Αναζήτηση εγγραφής (συνέχεια)

```
lower := 1; upper := B;
while (upper ≥ lower)
      i := (lower + upper) div 2;
      read block i
      if (K < τιμής διάταξης της πρώτης εγγραφής)
            upper := i - 1;
      else if (K > τιμής διάταξης της τελευταίας εγγραφής)
            lower := i + 1;
      else ...
```

Χρόνος: $\log B * (T_D + \log R * T_C)$

Συνθήκη πχ., <=

B blocks

R εγγραφές ανά block

T_D χρόνος μεταφοράς block

T_C χρόνος επεξεργασίας ανά εγγραφή

3. Διαγραφή εγγραφής

Μετακίνηση εγγραφών

Χρήση σημαδιού διαγραφής

4. Τροποποίηση εγγραφής

5. Ανάγνωση όλων των εγγραφών σε διάταξη

- Αρχεία Κατακερματισμού

Βασική ιδέα: η τοποθέτηση των εγγραφών στα blocks του αρχείου γίνεται εφαρμόζοντας μια συνάρτηση κατακερματισμού σε κάποιο από τα πεδία της

Εσωτερικός Κατακερματισμός (τα δεδομένα είναι στη μνήμη, όπως στις δομές δεδομένων)

Πίνακας κατακερματισμού με M θέσεις - κάδους (buckets)

h: *συνάρτηση κατακερματισμού*

h(k) = i ← Σε ποιο κάδο - τιμή από 0 έως M-1

Πεδίο αναζήτησης - Πεδίο κατακερματισμού

Εξωτερικός Κατακερματισμός (εφαρμογή σε δεδομένα αποθηκευμένα σε αρχεία)

Στόχος

h(k) = i ← Διεύθυνση (αριθμός) block του αρχείου που είναι αποθηκευμένη

Τιμή του πεδίου κατακερματισμού

Η εγγραφή με τιμή στο πεδίο κατακερματισμού k αποθηκεύεται στο i-οστο block (κάδο) του αρχείου

h: *συνάρτηση κατακερματισμού*

Ομοιόμορφη κατανομή των κλειδιών στους κάδους (blocks)

- Συνηθισμένη συνάρτηση κατακερματισμού:

$$h(k) = k \bmod M$$

Συχνά M πρώτος

• **Σύγκρουση (collision)**: όταν μια νέα εγγραφή κατακερματίζεται σε μία ήδη γεμάτη θέση

• **Καλή συνάρτηση κατακερματισμού**: κατανέμει τις εγγραφές ομοιόμορφα στο χώρο των διευθύνσεων (ελαχιστοποίηση συγκρούσεων και λίγες αχρησιμοποίητες θέσεις)

• **Ευριστικοί**:

-- αν r εγγραφές, πρέπει να επιλέξουμε το M ώστε το r/M να είναι μεταξύ του 0.7 και 0.9

-- όταν χρησιμοποιείται η mod τότε είναι καλύτερα το M να είναι πρώτος

Επίλυση Συγκρούσεων

1. *Ανοιχτή Διευθυνσιοδότηση* (open addressing): χρησιμοποίησε την επόμενη κενή θέση

2. *Αλυσιδωτή Σύνδεση* (chaining): για κάθε θέση μια συνδεδεμένη λίστα με εγγραφές υπερχείλισης

3. *Πολλαπλός Κατακερματισμός* (multiple hashing): εφαρμογή μιας δεύτερης συνάρτησης κατακερματισμού

Εξωτερικός Κατακερματισμός

Κάδος: μια συστάδα από συνεχόμενα blocks του αρχείου

h(k) = i ← Σχετική διεύθυνση του κάδου (ποιος κάδος του αρχείου)

Τιμή του πεδίου κατακερματισμού

Ο κατακερματισμός είναι πολύ αποδοτικός για επιλογές ισότητας

Εξωτερικός Κατακερματισμός

Ένας πίνακας που αποθηκεύεται στην επικεφαλίδα του αρχείου μετατρέπει τον αριθμό κάδου στην αντίστοιχη διεύθυνση block

0	διεύθυνση 1ου block του κάδου στο δίσκο
1	διεύθυνση 1ου block του κάδου στο δίσκο
2	διεύθυνση 1ου block του κάδου στο δίσκο
…	…
M-1	διεύθυνση 1ου block του κάδου στο δίσκο

Εξωτερικός Κατακερματισμός

Συγκρούσεις - αλυσιδωτή σύνδεση - εγγραφές υπερχείλισης ανά κάδο

1. **Ανάγνωση όλου του αρχείου (scan)**

   Έστω ότι διατηρούμε κάθε κάδο γεμάτο κατά 80% άρα ένα αρχείο με μέγεθος B blocks χρειάζεται 1.25 B blocks

   $1.25 * B * (T_D + R * T_C)$

2. **Αναζήτηση**

   Συνθήκη ισότητας και μόνο ένα block ανά κάδο: $T_D + R * C$

   Αν συνθήκη περιοχής (διαστήματος): scan!

Οργάνωση Αρχείων

Κόστος: μεταφορά blocks (I/O)

	Σωρός	Ταξινομημένο	Κατακερματισμένο
Ανάγνωση του αρχείου	B	B	1.25B
Αναζήτηση με συνθήκη ισότητας	0.5 B	logB	1
Αναζήτηση με συνθήκη περιοχής	B	logB + ταιριάσματα	1.25 B
Εισαγωγή	2	αναζήτηση + B	2
Διαγραφή	αναζήτηση + 1	αναζήτηση + B	αναζήτηση + 1