

ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ (Β ΕΤΟΣ, Α ΕΞΑΜΗΝΟ)

Υλοποίηση ΣΔΒΔ

Βάσεις Δεδομένων. Κατηγοριοποίηση Φυσικών Μέσων Αποθήκευσης

Η κατηγοριοποίηση των μέσων μπορεί να γίνει με βάση

- ❑ ταχύτητα προσπέλασης των δεδομένων
- ❑ κόστος ανά μονάδα δεδομένων
- ❑ αξιοπιστία που παρέχεται στην περίπτωση
 - ▣ διακοπής ρεύματος ή κατάρρευσης συστήματος
 - ▣ φυσικής απώλειας του αποθηκευτικού μέσου
- ❑ αποθήκευση των δεδομένων
 - ▣ προσωρινή αποθήκευση (volatile storage), χάνει τα δεδομένα όταν σβήνει το ρεύμα
 - ▣ μνήμη που δεν σβήνει (nonvolatile storage), διατηρεί τα δεδομένα ακόμη και όταν σβήνει το ρεύμα

Βάσεις Δεδομένων. Φυσικά Μέσα Αποθήκευσης

- ❑ **cache:** είναι η πιο γρήγορη και ακριβή μορφή αποθήκευσης, η χρήση της γίνεται μέσω του υλικού του συστήματος, *volatile*
- ❑ **κύρια μνήμη:** προσφέρει γρήγορη προσπέλαση δεδομένων (nanoseconds), είναι πολύ μικρή για να χωρέσει όλη τη ΒΔ, *volatile*

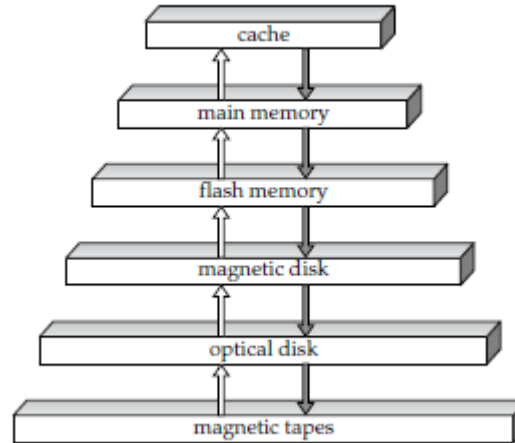
Βάσεις Δεδομένων. Φυσικά Μέσα Αποθήκευσης (συν.)

- ❑ **μνήμη flash:** τα δεδομένα διατηρούνται ακόμη και με διακοπή ρεύματος, γρήγορο διάβασμα αλλά είναι πολύπλοκο και αργό το γράψιμο και το σβήσιμο τους
- ❑ **μαγνητικοί δίσκοι:**
 - τα δεδομένα αποθηκεύονται σε περιστρεφόμενους μαγνητικούς δίσκους
 - αποτελούν το κύριο μέσο αποθήκευσης των δεδομένων
 - τα δεδομένα μεταφέρονται από αυτούς στην κύρια μνήμη για επεξεργασία και ξανά πίσω για εγγραφή
 - άμεσης πρόσβασης (direct-access), δηλαδή δυνατότητα πρόσβασης σε δεδομένα που βρίσκονται σε οποιαδήποτε θέση(όχι σειριακά)
 - τα δεδομένα διατηρούνται μετά από απώλεια ρεύματος ή αστοχία συστήματος

Βάσεις Δεδομένων. Φυσικά Μέσα Αποθήκευσης (συν.)

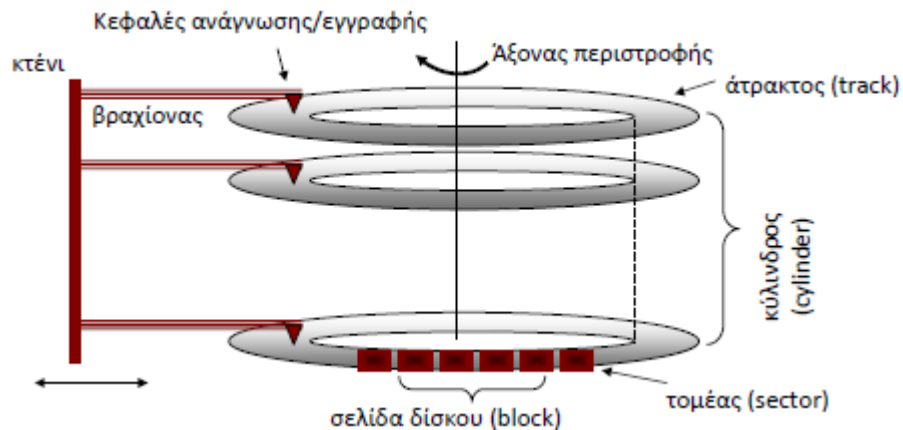
- **οπτική αποθήκευση**
- **αποθήκευση σε ταινία:**
 - ▣ χρησιμοποιούνται κυρίως για αντίγραφα ασφαλείας και αρχειοθέτηση δεδομένων
 - ▣ σειριακή πρόσβαση
 - ▣ μεγάλης χωρητικότητας

Βάσεις Δεδομένων. Ιεραρχία Συσκευών Αποθήκευσης

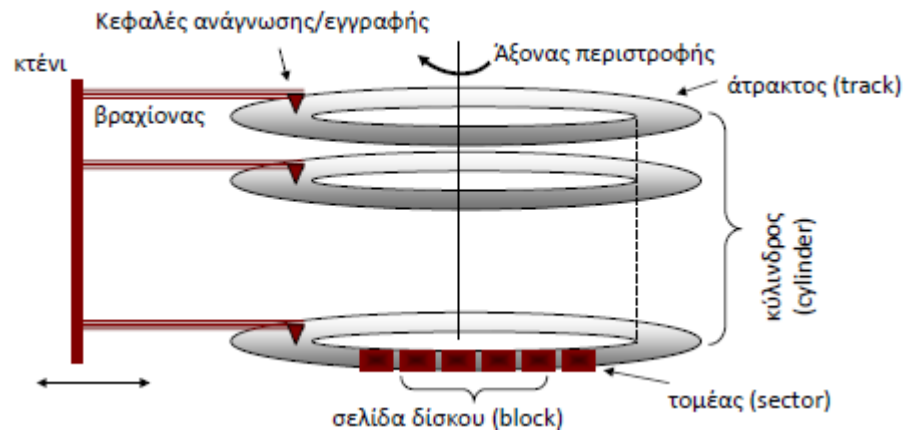


hierarchy01 access time01 relative distance01 relative distance02 typical size01 relative size01
relative size02

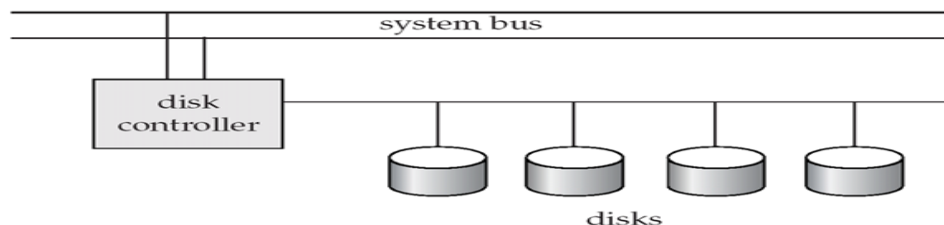
Βάσεις Δεδομένων. Μαγνητικοί Δίσκοι



Βάσεις Δεδομένων. Μαγνητικοί Δίσκοι (συν.)



Βάσεις Δεδομένων. Υποσύστημα Δίσκων



- Ένας **disk controller (ελεγκτής δίσκου)** κάνει την διασύνδεση μεταξύ του υπολογιστή και του υλικού των μονάδων δίσκου. Οι διασυνδέσεις που χρησιμοποιούνται είναι:
 - ATA (AT adaptor), γρηγορότερη της IDE
 - SATA (Serial ATA)
 - SCSI (Small Computer System Interconnect)
 - SAS (Serial Attached SCSI)
- Οι δίσκοι συνήθως συνδέονται άμεσα στον Η/Υ όμως υπάρχουν και διαμορφώσεις όπως:
 - **Storage Area Networks (SAN)**, όπου ένας μεγάλος αριθμός δίσκων συνδέεται με δίκτυο υψηλής ταχύτητας
 - **Network Attached Storage (NAS)**, όπου ένα δικτυακό storage (αριθμός δίσκων) συνδέεται δικτυακά μέσω (διεπαφή) file system αντί για disk system

- ❑ RAID: Redundant Arrays of Independent Disks
 - ▣ τεχνικές οργάνωσης και αποτελεσματική διαχείρισης μεγάλου αριθμού δίσκων
 - ▣ εμφανίζονται ως ένας ενιαίος δίσκος ο οποίος προσφέρει υψηλή χωρητικότητα, ταχύτητα και αξιοπιστία
 - ▣ Υπάρχουν διάφορα επίπεδα RAID (0, 1, 2, 3, 4, 5,6)

Βάσεις Δεδομένων. Μέτρηση Απόδοσης των Δίσκων

- ❑ **Χρόνος πρόσβασης (access time):** είναι ο χρόνος από την αίτηση ανάγνωσης ή εγγραφής, έως ότου αρχίσει η μεταφορά των δεδομένων
- ❑ **Ρυθμός μεταφοράς των δεδομένων (data-transfer rate):** είναι ο ρυθμός με τον οποίον τα δεδομένα ανακαλούνται από το δίσκο ή αποθηκεύονται στο δίσκο
- ❑ **Μέσος χρόνος αποτυχίας (mean time to failure – MTTF):** είναι το χρονικό διάστημα που, κατά μέσο όρο, μπορούμε να περιμένουμε ότι το σύστημα θα λειτουργεί χωρίς πρόβλημα

Βάσεις Δεδομένων. Βελτιστοποίηση της Πρόσβασης σε ένα Μπλοκ του Δίσκου

- Τα δεδομένα οργανώνονται σε **μπλοκ** (σελίδες δίσκου) τα οποία είναι μια συνεχής σειρά από τομείς, από ένα μόνο αυλάκι μιας επιφάνειας*
 - ▣ τα δεδομένα μεταφέρονται μεταξύ δίσκου και μνήμης σε μπλοκ
 - ▣ το μέγεθος των μπλοκ κυμαίνεται από 512 bytes σε kilobytes
 - μικρά μπλοκ, σημαίνει περισσότερες μεταφορές
 - μεγάλα μπλοκ, σημαίνει κατασπατάληση χώρου
 - τυπικά μεγέθη 4-16 kilobytes
- Υπάρχουν διάφορες τεχνικές βελτίωσης της ταχύτητας πρόσβασης στα μπλοκ του δίσκου
 - ▣ **Προγραμματισμός:** προγραμματίζουμε την κίνηση των κεφαλών ώστε να είναι ελάχιστη με βάση τα μπλοκ που έχουν ζητηθεί. Αυτό επιτυγχάνεται αλλάζοντας την σειρά με την οποία έχει αίτηση των μπλοκ. (αλγόριθμος ανελκυστήρα)
 - ▣ **Οργάνωση αρχείου:** οργανώνουμε τα μπλοκ στο δίσκο με ένα τρόπο που αντιστοιχεί με τον τρόπο που περιμένουμε να προσπελαστούν τα δεδομένα
 - ▣ **Μόνιμο buffer εγγραφής:** αντί τα δεδομένα να γράφονται στο δίσκο απευθείας γράφονται πρώτα σε μια **μόνιμη μνήμη τυχαίας πρόσβασης (nonvolatile random access memory NV-RAM)** η οποία έχει την ιδιότητα να διατηρεί τα περιεχόμενα της ακόμη και όταν σβήσει το ρεύμα. Όταν γράφουν τα δεδομένα στην NV-RAM επιστρέφεται στο ΣΔΒΔ ότι η εγγραφή έχει γίνει, στην πραγματικότητα η αλλαγές θα περάσουν στον δίσκο όταν αυτός δεν είναι απασχολημένος ή όταν γεμίσει το μόνιμο buffer της RAM
 - ▣ **Δίσκος καταγραφής:** χρήση ενός δίσκου (καταγραφής) με τον ίδιο τρόπο όπως η NV-RAM. Η εγγραφή εδώ όμως είναι σειριακή, οπότε απαλείφονται οι αναζητήσεις. Ο δίσκος καταγραφής θα γράψει τα δεδομένα αργότερα στον δίσκο.

Βάσεις Δεδομένων. Δίσκοι και Αρχεία

- ❑ Μια ΒΔ απεικονίζεται σε αρχεία ως μια σειρά από εγγραφές. Τα αρχεία διατηρούνται από το ΛΣ και αποθηκεύονται στον δίσκο
- ❑ Κάθε αρχείο, όπως είπαμε, αντιστοιχεί σε ένα ή περισσότερα **μπλοκ** (ή σελίδες-pages)
- ❑ Αυτό έχει σημαντικές επιπτώσεις στο σχεδιασμό των ΣΔΒΔ στα οποία διακρίνουμε δύο πολύ σημαντικές πράξεις (READ: από το σκληρό δίσκο στην μνήμη, WRITE: από την μνήμη στο σκληρό δίσκο). Επιθυμούμε όσο το δυνατό λιγότερες μεταφορές **μπλοκ*** και οι εργασίες να γίνονται στην μνήμη.
- ❑ Για τον σκοπό αυτό υπάρχει ο **buffer**: μέρος της κύριας μνήμης για την αποθήκευση αντιγράφων των μπλοκ
- ❑ Άρα πως θα διαχειριστούμε τον buffer (buffer manager);

- Υποπρογράμματα ενός ΣΔΒΔ κάνουν κλήσεις στον buffer manager (BM) όταν χρειάζονται ένα μπλοκ στον δίσκο
 - Εάν το μπλοκ είναι στον buffer, τότε ο BM περνά την διεύθυνση του μπλοκ στην κύρια μνήμη στον αιτούντα
 - Εάν το μπλοκ δεν είναι στον buffer τότε
 - Ο BM δημιουργεί χώρο στη μνήμη για τα νέα μπλοκ, βγάζοντας κάποια μπλοκ (ποια;)
 - Τα μπλοκ που διώχνονται γράφονται στο δίσκο μόνο αν έχουν τροποποιηθεί από την πιο πρόσφατη φορά που γράφτηκαν στον δίσκο

Βάσεις Δεδομένων. Πολιτικές Αντικατάστασης Buffer

- Στα περισσότερα ΛΣ τα μπλοκ αντικαθιστούνται με βάση την στρατηγική **LRU** (least recently used), **ελάχιστου πρόσφατα χρησιμοποιηθέντος**. Βασίζεται σε παλαιότερα μοτίβα χρησιμοποίησης για να προβλέψει την μελλοντική χρήση
- Ωστόσο με βάση τον τρόπο εκτέλεσης των ερωτημάτων τα ΣΔΒΔ μπορούν να προβλέψουν μελλοντική χρήση των μπλοκ και καθώς η LRU μπορεί να αποδειχθεί κακή επιλογή για τα ΣΔΒΔ υπάρχουν και άλλες στρατηγικές
- Η **MRU** (most recently used), **πιο πρόσφατα χρησιμοποιηθέντος** κατά την οποία αντικαθίσταται το πιο πρόσφατα χρησιμοποιηθέν μπλοκ. Το μπλοκ το οποίο βρίσκεται σε επεξεργασία σταθεροποιείται (pinned) και ελευθερώνεται-αποσταθεροποιείται (unpinned) αφού γίνει η επεξεργασία. Στη συνέχεια θεωρείται το πιο πρόσφατα χρησιμοποιηθέν.
- Στον **MRU** επηρεάζεται και από άλλα υποσυστήματα του ΣΔΒΔ όπως υποσύστημα ελέγχου συχρονικότητας, ανάκαμψης κ.α. καθώς και στατιστικές πληροφορίες κλπ

Βάσεις Δεδομένων. Τρόποι Αναπαράστασης. Εγγραφές Σταθερού Μήκους-Παραδοχές

- Κάθε αρχείο έχει εγγραφές ενός τύπου μόνο
- Το μέγεθος των εγγραφών είναι σταθερό (με μέγεθος r bytes)
- Οι εγγραφές δεν επιτρέπεται να διασχίζουν τα όρια ενός block
- Αν υποθέσουμε ότι το μέγεθος ενός μπλοκ είναι b bytes τότε ορίζουμε τον **παράγοντα ομαδοποίησης (blocking factor – bfr)**

$$bfr = b/r$$

κατ' επέκταση αν θέλουμε να αποθηκεύσουμε ένα αρχείο R εγγραφών τότε ο αριθμός των μπλοκ B που χρειάζονται είναι

$$B = R/bfr$$

Βάσεις Δεδομένων. Τρόποι Αναπαράστασης. Εγγραφές Σταθερού Μήκους

- Έστω i εγγραφές με μέγεθος n η κάθε μία, οι εγγραφές δεν επιτρέπεται να διασχίζουν τα όρια ενός block
- Που θα εισαχθούν οι νέες εγγραφές, στην περίπτωση που υπάρχουν διαγραφές ποία η αντιμετώπιση (π.χ στην περίπτωση που διαγραφεί η i εγγραφή)
 - μετακίνηση των $i+1, \dots, n$ σε $i, \dots, n-1$
 - μετακίνηση n στην θέση της i
 - να μην γίνει καμία μετακίνηση αλλά να υπάρχει καταγραφή των ελεύθερων θέσεων (με **free list** - ελεύθερη λίστα)

record 0	10101	Srinivasan	Comp. Sci.	65000
record 1	12121	Wu	Finance	90000
record 2	15151	Mozart	Music	40000
record 3	22222	Einstein	Physics	95000
record 4	32343	El Said	History	60000
record 5	33456	Gold	Physics	87000
record 6	45565	Katz	Comp. Sci.	75000
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000
record 11	98345	Kim	Elec. Eng.	80000

Βάσεις Δεδομένων. Τρόποι Αναπαράστασης. Παράδειγμα διαγραφής και μετακίνησης σε Εγγραφές Σταθερού Μήκους

record 0	10101	Srinivasan	Comp. Sci.	65000
record 1	12121	Wu	Finance	90000
record 2	15151	Mozart	Music	40000
record 4	32343	El Said	History	60000
record 5	33456	Gold	Physics	87000
record 6	45565	Katz	Comp. Sci.	75000
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000
record 11	98345	Kim	Elec. Eng.	80000

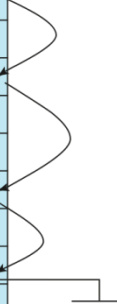
Βάσεις Δεδομένων. Τρόποι Αναπαράστασης. Παράδειγμα διαγραφής και μετακίνησης σε Εγγραφές Σταθερού Μήκους (συν.)

record 0	10101	Srinivasan	Comp. Sci.	65000
record 1	12121	Wu	Finance	90000
record 2	15151	Mozart	Music	40000
record 11	98345	Kim	Elec. Eng.	80000
record 4	32343	El Said	History	60000
record 5	33456	Gold	Physics	87000
record 6	45565	Katz	Comp. Sci.	75000
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000

Βάσεις Δεδομένων. Τρόποι Αναπαράστασης. Παράδειγμα διαγραφής και μετακίνησης σε Εγγραφές Σταθερού Μήκους (συν.)

- Χρήση **ελεύθερης λίστας** όπου αποθηκεύεται η διεύθυνση της πρώτης διαγεγραμμένης εγγραφής στην επικεφαλίδα του αρχείου (file header). Στη πρώτη εγγραφή αποθηκεύεται η διεύθυνση της δεύτερης διαγεγραμμένης εγγραφής κ.ο.κ. (μπορούμε να τις σκεφτούμε ως **δείκτες**)

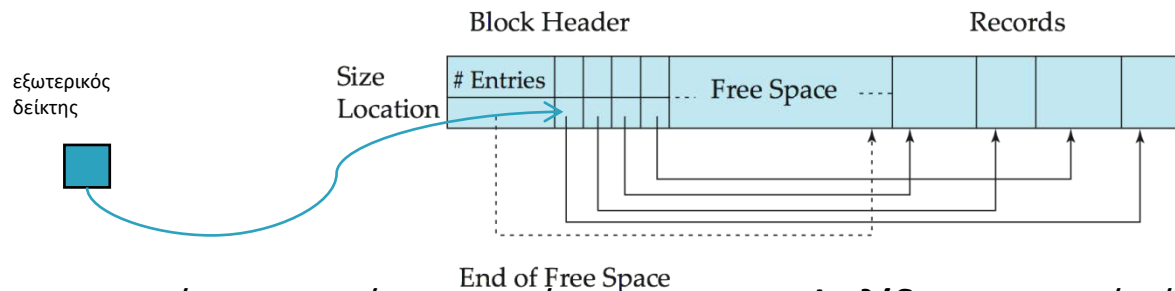
header				
record 0	10101	Srinivasan	Comp. Sci.	65000
record 1				
record 2	15151	Mozart	Music	40000
record 3	22222	Einstein	Physics	95000
record 4				
record 5	33456	Gold	Physics	87000
record 6				
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000
record 11	98345	Kim	Elec. Eng.	80000



Βάσεις Δεδομένων. Τρόποι Αναπαράστασης. Εγγραφές Μεταβλητού Μήκους

- ❑ Οι εγγραφές **μεταβλητού μήκους** μπορούν να προκύψουν στις περιπτώσεις όπου
 - ▣ ένα αρχείο έχει εγγραφές περισσότερες του ενός τύπου*
 - ▣ το μέγεθος των εγγραφών είναι μεταβλητό (καθώς επιτρέπεται σε ένα ή περισσότερα πεδία να έχουν μεταβλητά μήκη)**
 - ▣ οι τύποι των εγγραφών να επιτρέπουν επανάληψη πεδίων
- ❑ Υπάρχουν δύο προσεγγίσεις στην αναπαράσταση των εγγραφών αυτών
 - ▣ αναπαράσταση συμβολοσειρών με byte
 - ▣ αναπαράσταση σταθερού μήκους

Βάσεις Δεδομένων. Τρόποι Αναπαράστασης. Εγγραφές Μεταβλητού Μήκους. Αναπαράσταση Συμβολοσειρών με Byte. Δομή Τεμαχισμένων Σελίδων (slotted page)



- Σε αυτή την προσέγγιση υπάρχει μια **επικεφαλίδα** στην αρχή κάθε μπλοκ. Περιλαμβάνει:
 - αριθμό στοιχείων στην επικεφαλίδα
 - τέλος του ελεύθερου μπλοκ
 - πίνακα με την θέση και το μέγεθος κάθε εγγραφής
- Με τον τρόπο αυτό
 - οι εγγραφές μπορούν να μετακινηθούν μέσα στο **μπλοκ** του ΛΣ “χωρίς κόστος” – data independence
 - οι εξωτερικοί δείκτες δείχνουν μόνο στους εσωτερικούς δείκτες

Βάσεις Δεδομένων. Τρόποι Αναπαράστασης. Εγγραφές Μεταβλητού Μήκους. Αναπαράσταση Σταθερού Μήκους

- Σε αυτή την προσέγγιση χρησιμοποιούμε μια ή περισσότερες εγγραφές σταθερού μήκους για να αναπαραστήσουμε μια εγγραφή μεταβλητού μήκους
- Οι τρόποι υλοποίησης είναι:
 - **Δεσμευμένος χώρος (padding):** όπου εάν υπάρχει ένα μέγιστο μήκος που δεν ξεπερνιέται ποτέ, μπορούμε να χρησιμοποιήσουμε εγγραφές σταθερού μήκους αυτού του μήκους
 - **Αναπαράσταση λίστας:** με εγγραφές μεταβλητού μήκους από λίστες με εγγραφές σταθερού μήκους συνδεδεμένες μαζί με δείκτες

Βάσεις Δεδομένων. Τρόποι Αναπαράστασης. Εγγραφές Μεταβλητού Μήκους. Αναπαράσταση Σταθερού Μήκους. Μέθοδος Δεσμευμένου Χώρου

- Ο μη χρησιμοποιηθείς χώρος (διαγραφές μικρότερες από το μέγιστο χώρο) γεμίζει με ένα ειδικό κενό ή σύμβολο τέλους εγγραφής
- Η μέθοδος είναι χρήσιμη όταν οι περισσότερες εγγραφές έχουν μήκος κοντά στο μέγιστο, διαφορετικά μπορεί να χαθεί μια σημαντική ποσότητα χώρου

0	Perryridge	A-102	400	A-201	900	A-218	700
1	Round Hill	A-305	350	⊥	⊥	⊥	⊥
2	Mianus	A-215	700	⊥	⊥	⊥	⊥
3	Downtown	A-101	500	A-110	600	⊥	⊥
4	Redwood	A-222	700	⊥	⊥	⊥	⊥
5	Brighton	A-217	750	⊥	⊥	⊥	⊥

Βάσεις Δεδομένων. Τρόποι Αναπαράστασης. Εγγραφές Μεταβλητού Μήκους. Αναπαράσταση Σταθερού Μήκους. Αναπαράσταση Λίστας

- Στη μέθοδο αυτή χρησιμοποιούνται **δείκτες** για να συνδέσουν όλες τις εγγραφές που αφορούν την ίδια πλειάδα*
- Μπορούμε αν αναπαραστήσουμε μια εγγραφή ανεξάρτητα του μεγέθους της
- Χάνουμε χώρο σε όλες τις εγγραφές, εκτός από την πρώτη της αλυσίδας, οπότε χρησιμοποιούμε δύο είδη μπλοκ
 - **μπλοκ αγκύρωσης (anchor block)**: που περιέχει την πρώτη εγγραφή της αλυσίδας
 - **μπλοκ υπερχείλισης (overflow block)**: που περιέχει εγγραφές που είναι διαφορετικές από αυτές που είναι στην πρώτη εγγραφή της αλυσίδας



Βάσεις Δεδομένων. Οργάνωση Εγγραφών σε Αρχεία

- Πέρα από την αναπαράσταση των εγγραφών πρέπει να εξετάσουμε και την οργάνωση των εγγραφών μέσα στα αρχεία. Οι πιθανοί τρόποι είναι:
 - **Οργάνωση αρχείου σε σωρό (heap or pipe file):** όπου οι εγγραφές τοποθετούνται στο τέλος του αρχείου. Δεν υπάρχει διάταξη στις εγγραφές. Χρησιμοποιεί όσο ακριβώς block χρειάζονται και συνδέει τα block για το ίδιο αρχείο μεταξύ τους
 - **Σειριακή οργάνωση αρχείων (sequential):** οι εγγραφές αποθηκεύονται σε σειριακή σειρά, σύμφωνα με την τιμή ενός κλειδιού αναζήτησης της εγγραφής
 - **Οργάνωση αρχείου κατακερματισμού (hash):** το αποτέλεσμα μιας συνάρτησης hash σε κάποια ιδιότητα κάθε εγγραφής καθορίζει σε ποιο μπλοκ του αρχείου θα τοποθετηθεί η εγγραφή

Βάσεις Δεδομένων. Οργάνωση Εγγραφών σε Αρχεία. Αρχεία Σωρού

- ❑ Οι εγγραφές τοποθετούνται στο τέλος του αρχείου
- ❑ Για την αναζήτηση μιας εγγραφής είναι απαραίτητη μια **γραμμική αναζήτηση** των εγγραφών του αρχείου
 - ❑ αυτό κατά μέσο όρο απαιτεί διάβασμα και αναζήτηση των μισών μπλοκ του αρχείου, και επομένως έχει μεγάλο κόστος
- ❑ Η εισαγωγή είναι πολύ αποτελεσματική
- ❑ Η ανάγνωση εγγραφών με συγκεκριμένη σειρά κάποιου πεδίου απαιτεί ταξινόμηση όλου του αρχείου

header				
record 0	A-102	Perryridge	400	
record 1				
record 2	A-215	Mianus	700	
record 3	A-101	Downtown	500	
record 4				
record 5	A-201	Perryridge	900	
record 6				
record 7	A-110	Downtown	600	
record 8	A-218	Perryridge	700	

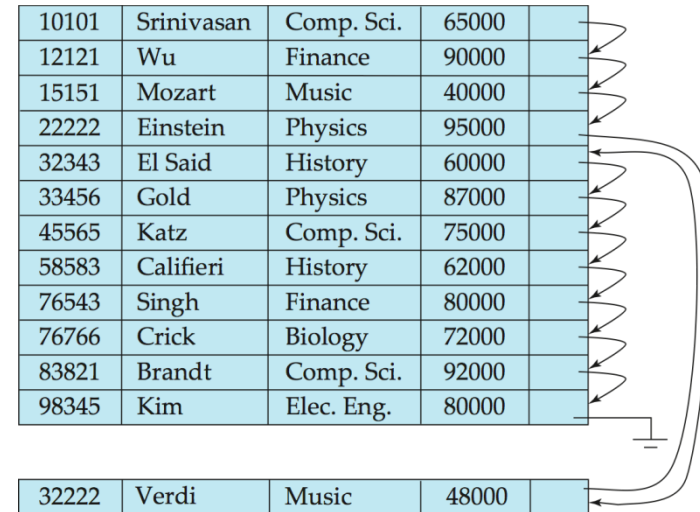
Βάσεις Δεδομένων. Οργάνωση Εγγραφών σε Αρχεία. Σειριακή Οργάνωση Αρχείου

- Είναι κατάλληλα για χρήση κατά την οποία απαιτείται σειριακή επεξεργασία του αρχείου
- Οι εγγραφές στο αρχείο ταξινομούνται με βάση ένα **κλειδί αναζήτησης**
- Το κλειδί αναζήτησης είναι οποιαδήποτε ιδιότητα ή σύνολο ιδιοτήτων και δεν χρειάζεται να είναι πρωτεύων κλειδί ή υπερκλειδί
- Συνδέουμε τις εγγραφές σε σειρά κλειδιού αναζήτησης με δείκτες

header				
record 0	A-102	Perryridge	400	
record 1				
record 2	A-215	Mianus	700	
record 3	A-101	Downtown	500	
record 4				
record 5	A-201	Perryridge	900	
record 6				
record 7	A-110	Downtown	600	
record 8	A-218	Perryridge	700	

Βάσεις Δεδομένων. Οργάνωση Εγγραφών σε Αρχεία. Σειριακή Οργάνωση Αρχείου (συν.)

- Για την διαγραφή μπορούμε να χρησιμοποιήσουμε αλυσίδες δεικτών (διαφ 20)
- Κατά την εισαγωγή εντοπίζουμε την εγγραφή του αρχείου που έρχεται πριν την εγγραφή που θα εισαχθεί σε σειρά κλειδιού
 - ▣ αν υπάρχει ελεύθερη θέση τότε γίνεται η εισαγωγή εκεί
 - ▣ αν δεν υπάρχει ελεύθερη θέση τότε γίνεται η εισαγωγή σε μπλοκ υπερχείλισης
 - ▣ σε κάθε περίπτωση οι δείκτες πρέπει να αναδιοργανώνονται (σε σειρά κλειδιού αναζήτησης)



Βάσεις Δεδομένων. Οργάνωση Εγγραφών σε Αρχεία. Αρχεία Κατακερματισμού

- ❑ Ένας **κάδος (bucket)** είναι μια μονάδα αποθήκευσης που περιέχει μια ή περισσότερες εγγραφές (συνήθως κάδος = block)
- ❑ Σε ένα αρχείο κατακερματισμού (hashing) βρίσκουμε τον κάδο στον οποίο περιέχεται μια εγγραφή απευθείας από την τιμή του κλειδιού διάταξης της εγγραφής χρησιμοποιώντας μια συνάρτηση κατακερματισμού (hashing function)

bucket 0

--	--	--

bucket 1

--	--	--

bucket 2

--	--	--

bucket 3

A-217	Brighton	750
A-305	Round Hill	350

bucket 4

A-222	Redwood	700

bucket 5

A-102	Perryridge	400
A-201	Perryridge	900
A-218	Perryridge	700

bucket 6

--	--	--

bucket 7

A-215	Mianus	700

bucket 8

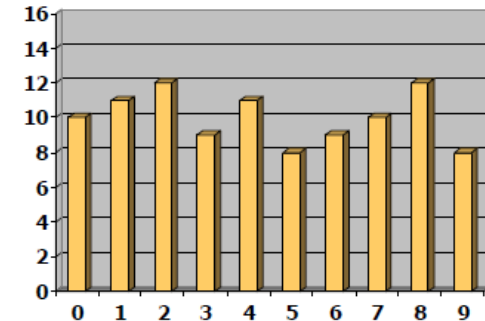
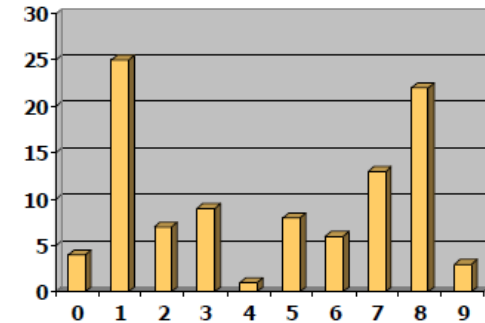
A-101	Downtown	500
A-110	Downtown	600

bucket 9

--	--	--

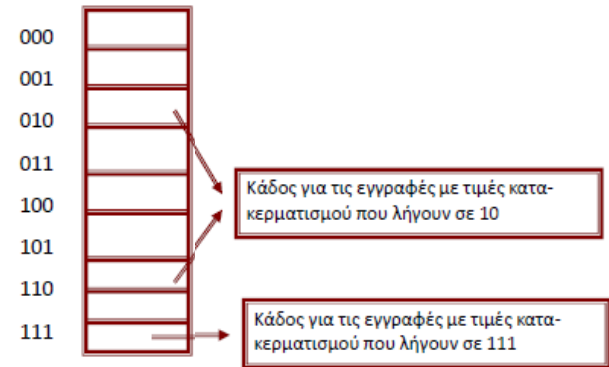
Βάσεις Δεδομένων. Οργάνωση Εγγραφών σε Αρχεία. Συναρτήσεις Κατακερματισμού

- Η **χειρότερη συνάρτηση** κατακερματισμού αντιστοιχεί όλες τις τιμές του κλειδιού διάταξης στον ίδιο κάδο
- Μια **ιδανική συνάρτηση** κατακερματισμού είναι:
 - ▣ **ομοιόμορφη**, σε κάθε κάδο ανατίθεται ο ίδιος αριθμός από τιμές του κλειδιού διάταξης από το σύνολο όλων των πιθανών τιμών
 - ▣ **τυχαία**, κάθε κάδος έχει τον ίδιο αριθμό από εγγραφές ανεξάρτητα από την πραγματική κατανομή των τιμών του κλειδιού διάταξης στο αρχείο



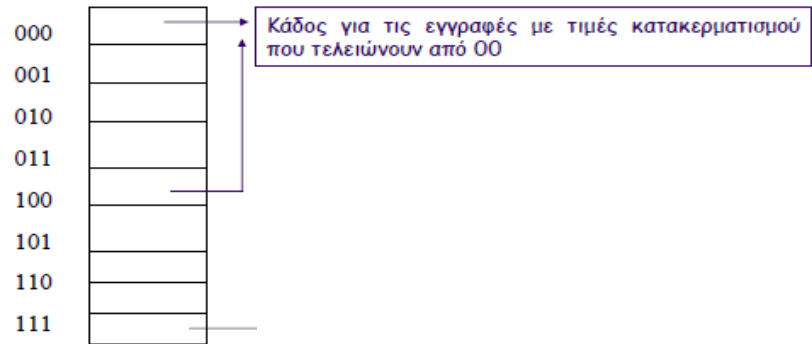
Βάσεις Δεδομένων. Οργάνωση Εγγραφών σε Αρχεία. Επεκτάσιμος Κατακερματισμός

- Κατά την τεχνική αυτή γίνεται δυαδική αναπαράσταση του αποτελέσματος της συνάρτησης κατακερματισμού, δηλαδή ως μια ακολουθία δυαδικών ψηφίων
- Στη συνέχεια η κατανομή των εγγραφών βασίζεται στην τιμή των λιγότερο σημαντικών δυαδικών ψηφίων (Least Significant Bit – LSB)
- Διαδικασία
 - Το αρχείο ξεκινά με ένα μόνο κάδο
 - Μόλις γεμίσει ένας κάδος διασπάται σε δύο κάδους (0 και 1) με βάση την τιμή του 1^{ου} LSB των τιμών κατακερματισμού. Οι εγγραφές που το τελευταίο ψηφίο της τιμής κατακερματισμού τους είναι 1 τοποθετούνται σε ένα κάδο και οι άλλες (με 0) σε άλλο
 - Νέα υπερχειλίση ενός κάδου οδηγεί σε διάσπαση του με βάση το 2^ο LSB, κ.ο.κ.
- Έτσι δημιουργείται μια δυαδική δενδρική δομή που λέγεται **κατάλογος (directory)** ή **ευρετήριο (index)** με δύο ειδών κόμβους
 - **εσωτερικούς:** που καθοδηγούν την αναζήτηση
 - **εξωτερικούς:** που δείχνουν σε ένα κάδο
- Ο κατάλογος διατηρείται στην μνήμη, εκτός αν είναι μεγάλος



Βάσεις Δεδομένων. Οργάνωση Εγγραφών σε Αρχεία. Επεκτάσιμος Κατακερματισμός (συν.)

- Τα τελευταία d ψηφία της τιμής κατακερματισμού χρησιμοποιούνται ως δείκτης στον κατάλογο
 - ▣ Δεν χρειάζεται διαφορετικός κάδος για κάθε μια από τις 2^d θέσεις – μπορεί η θέση του καταλόγου να δείχνει στη διεύθυνση του ίδιου κάδου αν αυτές χωράνε σε ένα κάδο
 - ▣ Η τιμή του d μπορεί να αυξάνεται ή να μειώνεται



Βάσεις Δεδομένων.



Ευρετήρια

Βάσεις Δεδομένων. Ευρετήρια - Εισαγωγή

- ❑ Οι μηχανισμοί ευρετηριοποίησης επιταχύνουν την διαδικασία ανάκτησης των επιθυμητών δεδομένων
 - ▣ π.χ. κατάλογος συγγραφέων στην βιβλιοθήκη
- ❑ Το ευρετήριο (index) αποτελείται από εγγραφές της μορφής:

Κλειδί αναζήτησης

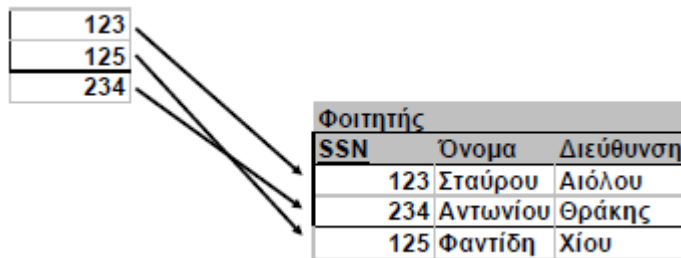
Δείκτης

κλειδί αναζήτησης (search key): ονομάζεται ένα γνώρισμα ή σύνολο γνωρισμάτων που χρησιμοποιείται για την εύρεση εγγραφών στο αρχείο

- ❑ Το αρχείο του ευρετηρίου συνήθως καταλαμβάνει σημαντικά μικρότερο πλήθος μπλοκ από ότι το αρχείο δεδομένων επειδή οι καταχωρήσεις του είναι κατά πολύ μικρότερες
- ❑ Υπάρχουν δύο βασικά είδη ευρετηρίων:
 - ▣ **ταξινομημένα/διατεταγμένα ευρετήρια (ordered indices):** οι εγγραφές του κλειδιού αναζήτησης αποθηκεύονται ταξινομημένα
 - ▣ **ευρετήρια κατακερματισμού (hash indices):** οι εγγραφές των κλειδιών αναζήτησης διανέμονται ομοιόμορφα σε “κάδους” με χρήση κάποιας συνάρτησης κατακερματισμού

Βάσεις Δεδομένων. Ευρετήρια. Μέτρηση Αποτελεσματικότητας

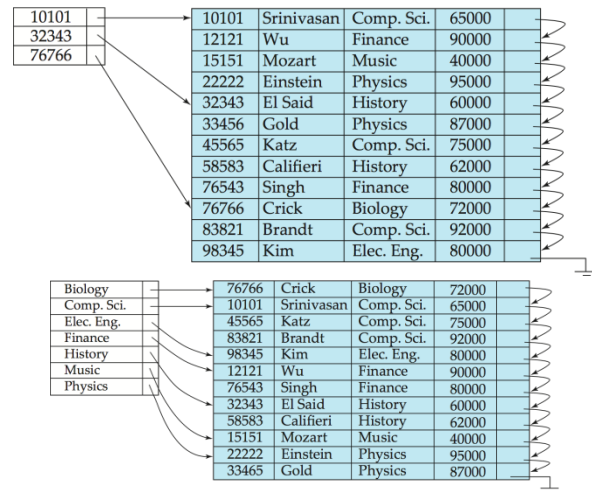
- Πως ψάχνουμε αποτελεσματικά εφόσον οι εγγραφές είναι αποθηκευμένες σε ένα αρχείο;
 - ▣ **brute force**: ανέκτησε όλες τις εγγραφές, επέστρεψε μόνο αυτές που πληρούν το κριτήριο αναζήτησης
 - ▣ **καλύτερα**: χρησιμοποίησε δείκτες για τον απευθείας εντοπισμό των εγγραφών
- Χρόνος ανάκτησης
- Χρόνος εισαγωγής/διαγραφής
- Επιβάρυνση χώρου
- Αναδιοργάνωση
- Ερωτήματα διαστήματος



- ❑ **Πρωτεύον ευρετήριο:** ορίζεται για κάποιο ταξινομημένο, σε κάποιο κλειδί αναζήτησης (μπορεί –ή όχι να είναι υποψήφιο κλειδί), αρχείο δεδομένων
- ❑ **Δευτερεύον ευρετήριο:** υποστηρίζει ένα δευτερεύοντα τρόπο προσπέλασης ενός αρχείου για το οποίο υπάρχει ήδη πρωτεύουσα οργάνωση, δηλαδή το κλειδί αναζήτησης καθορίζει διαφορετική σειρά από την σειρά οργάνωσης των εγγραφών του αρχείου

Βάσεις Δεδομένων. Ευρετήρια. Αραιά/Πυκνά Ευρετήρια

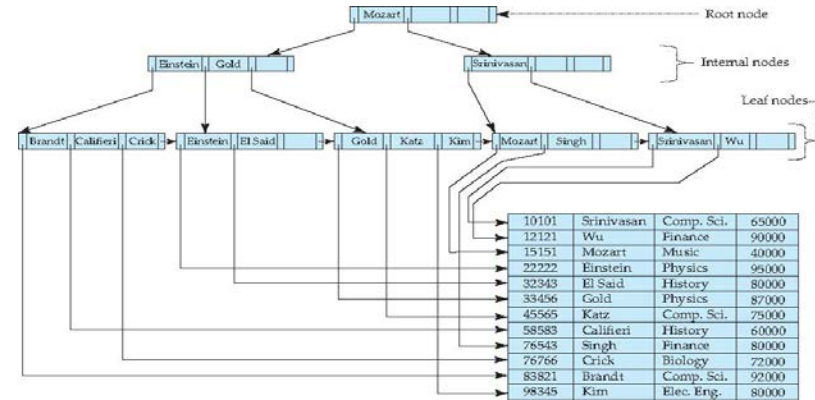
- **Αραιό ευρετήριο***: δεν συμπεριλαμβάνονται στο ευρετήριο όλες οι τιμές του κλειδιού αναζήτησης (πεδίου ευρετηριοποίησης)
- **Πυκνό ευρετήριο**: το αντίθετο



- ❑ Με βάση τα προηγούμενα έχουν προταθεί διάφορες δομές ευρετηρίων όπως π.χ. η **ISAM (indexed sequential access method)** η οποία παρουσιάζει προβλήματα απόδοσης και διαχείρισης.
- ❑ Η πιο διαδεδομένη και αποδοτική μορφή διατεταγμένου ευρετηρίου είναι το **B⁺-tree***. Η δομή αυτή έχει επικρατήσει και αποτελεί τη βασική επιλογή σε όλα τα ΣΔΒΔ

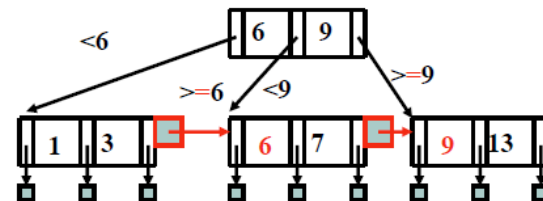
Βάσεις Δεδομένων. Παράδειγμα B⁺-tree

- ❑ Κάθε κόμβος έχει k τιμές και $k+1$ δείκτες
- ❑ Όλες οι τιμές του κλειδιού αναζήτησης (διάταξης) υπάρχουν στα φύλλα
- ❑ Στα φύλλα, οι δείκτες (εκτός από έναν) δείχνουν σε εγγραφές στο αρχείο



Βάσεις Δεδομένων. Ορισμός B⁺-tree

- Ένα B⁺-tree τάξης n είναι ένα δένδρο με ρίζα που ικανοποιεί τις παρακάτω ιδιότητες:
 - ▣ όλα τα μονοπάτια από τη ρίζα προς το φύλλο έχουν το ίδιο μήκος (ισοζυγισμένο – balanced δέντρο)
 - ▣ ένας εσωτερικός κόμβος με k τιμές κλειδιών έχει $k+1$ παιδιά*
 - ▣ ένας εσωτερικός κόμβος μπορεί να έχει από $n/2$ έως n παιδιά
 - ειδικά η ρίζα (αν είναι εσωτερικός κόμβος και όχι φύλλο) έχει 2 έως n παιδιά
- Ένα φύλλο έχει $(n-1)/2$ έως $n-1$ τιμές
 - ▣ Ειδικά η ρίζα (αν είναι φύλλο) έχει από 0 έως $n-1$ τιμές



- Ένας τυπικός κόμβος
 - ▣ όπου K_i είναι οι τιμές του κλειδιού αναζήτησης
 - ▣ όπου P_i είναι οι δείκτες σε παιδιά (για εσωτερικούς κόμβους) ή δείκτες σε μπλοκ εγγραφών (για φύλλα)



- Τα κλειδιά αναζήτησης σε ένα κόμβο είναι ταξινομημένα $K_1 < K_2 < K_3 < \dots < K_{n-1}$

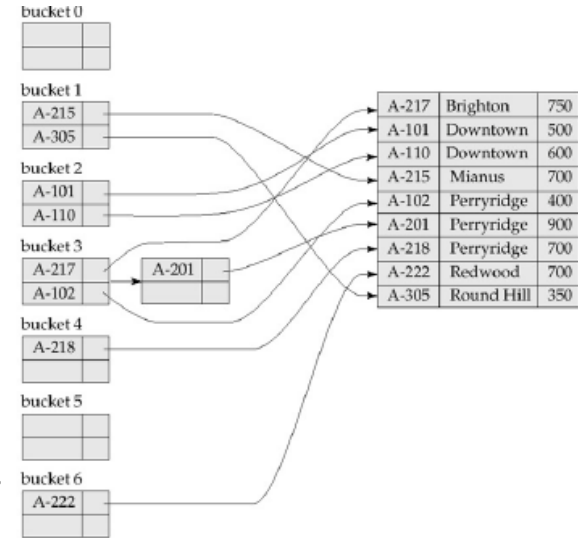
Βάσεις Δεδομένων. Τα φύλλα του B⁺-tree

- ❑ Ιδιότητες ενός φύλλου
 - Για $i = 1, 2, \dots, n-1$, ο δείκτης P_i δείχνει σε μια εγγραφή (στο αρχείο) με τιμή κλειδιού αναζήτησης K_i
 - Δηλαδή, το τελευταίο επίπεδο (φύλλα) λειτουργεί ως διατεταγμένο ευρετήριο
- ❑ Ο δείκτης P_n (τελευταίος δείκτης) δείχνει στο επόμενο φύλλο βάσει της σειράς του κλειδιού αναζήτησης. Άρα, υπάρχει ταξινόμηση στα φύλλα του δέντρου



Βάσεις Δεδομένων. Ευρετήρια Κατακερματισμού

- Ο κατακερματισμός δεν χρησιμοποιείται μόνο για οργάνωση αρχείων, αλλά και για την δημιουργία δομών ευρετηρίων
- Ένα **ευρετήριο κατακερματισμού (hash index)** οργανώνει τα κλειδιά αναζήτησης και τους αντίστοιχους δείκτες στις εγγραφές σε μια δομή ενός αρχείου κατακερματισμού
- Εάν το αρχείο είναι οργανωμένο με κατακερματισμό, είναι πλεονασμός να υπάρχει ξεχωριστό ευρετήριο πάνω σε αυτό χρησιμοποιώντας το ίδιο κλειδί διάταξης



Βάσεις Δεδομένων. Ευρετήρια Κατακερματισμού ή Διαταγμένα;

- ❑ Σχετική συχνότητα εισαγωγών και διαγραφών
- ❑ Αναμενόμενοι τύποι ερωτημάτων:
 - ▣ Ο κατακερματισμός είναι προτιμότερος για **ερωτήσεις ταυτότητας** πάνω στο κλειδί αναζήτησης (διάταξης)
 - ▣ Για **ερωτήσεις διαστήματος** πάνω στο κλειδί επιλέγονται τα διατεταγμένα ευρετήρια καθώς ο κατακερματισμός **δεν** μπορεί να υποστηρίξει ερωτήσεις διαστήματος