ΥΠΟΥΡΓΕΙΟ ΕΘΝΙΚΗΣ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ ΕΠΕΑΕΚ
ΕΥΡΩΠΑΪΚΗ ΕΝΩΣΗ
ΣΥΓΧΡΗΜΑΤΟΔΟΤΗΣΗ
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
ΕΛΛΑΔΑ 2008
Ανάπτυξη παντού. Ανάπτυξη για όλους.
Η ΠΑΙΔΕΙΑ ΣΤΗΝ ΚΟΡΥΦΗ
Επιχειρησιακό Πρόγραμμα
Εκπαίδευσης και Αρχικής
Επαγγελματικής Κατάρτισης

Προχωρημένα Θέματα Τεχνολογίας και Εφαρμογών Βάσεων Δεδομένων

Φυσική Σχεδίαση & Ρύθμιση Βάσεων Δεδομένων

Πάνος Βασιλειάδης
pvassil@cs.uoi.gr

Δεκέμβρης 2004

www.cs.uoi.gr/~pvassil/courses/db_III/

Θεματολόγιο

- Η διαδικασία σχεδίασης και ρύθμισης
- Επιλογή ευρετηρίων
  - Γενικές αρχές
  - Συνδέσεις
  - Διάφορα
- Προσαρμογή του σχήματος
  - Αποκανονικοποίηση σχήματος
  - Κατάτμηση σχήματος και χρήση όψεων
- Ρύθμιση ερωτήσεων
- Σύνοψη

Θεματολόγιο

- Η διαδικασία σχεδίασης και ρύθμισης
- Επιλογή ευρετηρίων
  - Γενικές αρχές
  - Συνδέσεις
  - Διάφορα
- Προσαρμογή του σχήματος
  - Αποκανονικοποίηση σχήματος
  - Κατάτμηση σχήματος και χρήση όψεων
- Ρύθμιση ερωτήσεων
- Σύνοψη

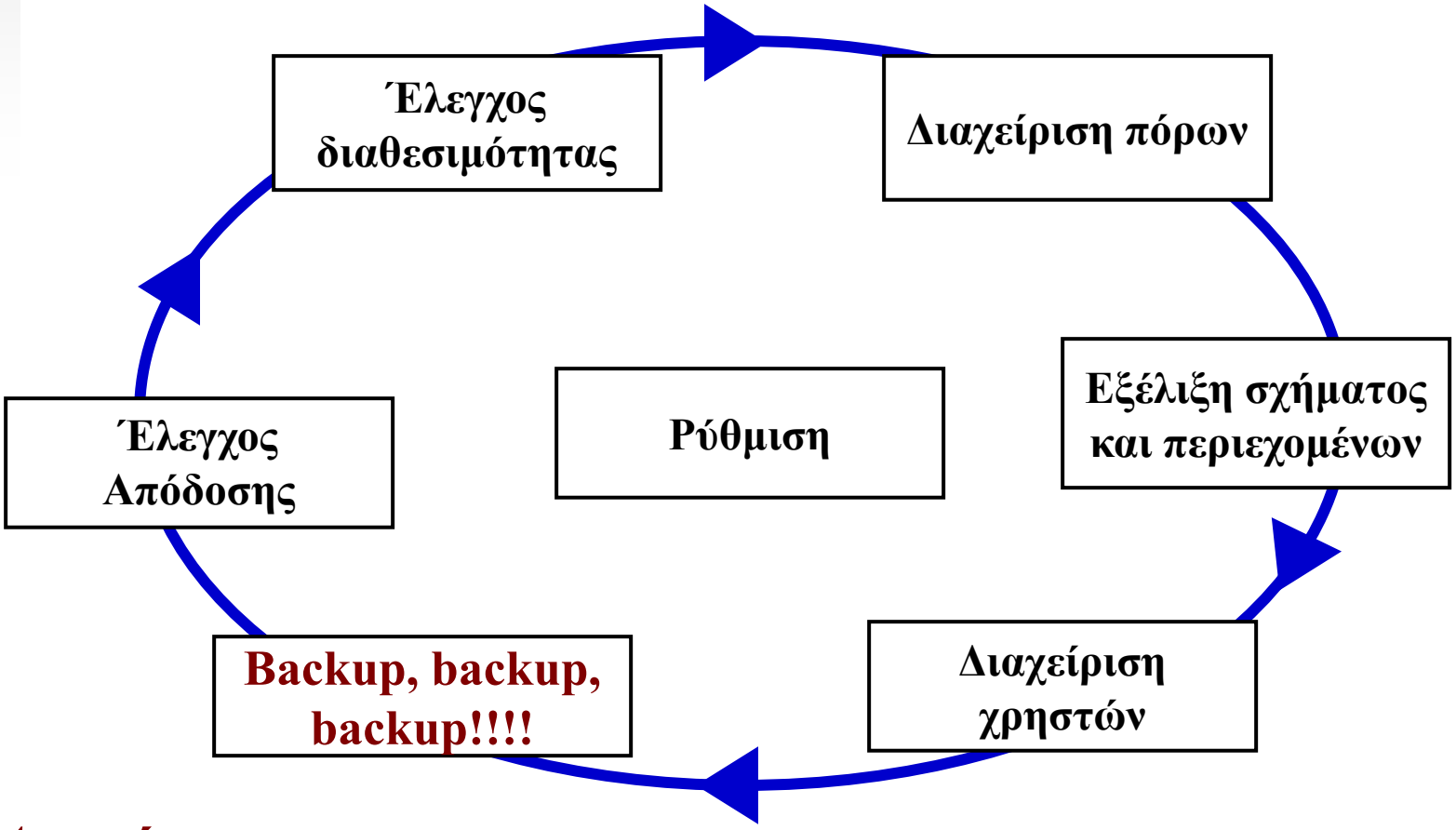
Σχεδίαση Βάσης Δεδομένων

- Αρχικά, στη βάση των απαιτήσεων των χρηστών, παράγουμε το **εννοιολογικό σχήμα** της βάσης δεδομένων, στο μοντέλο Οντοτήτων-Συσχετίσεων (ER model)
- Στη συνέχεια, παράγουμε το **λογικό σχήμα** της βάσης, μεταφράζοντας το εννοιολογικό σχήμα σε ένα σύνολο πινάκων στο σχεσιακό μοντέλο

Δεν αρκούν αυτά !!

Σχεδίαση

Δεν αρκούν αυτά !!



Διαχείριση

Φυσική σχεδίαση

- Αρχιτεκτονική του συστήματος (επιλογή hardware, φυσική τοποθέτηση)
- Βάση Δεδομένων:
  - οργάνωση σε δίσκους,
  - επιλογή ευρετηρίων (indexes),
  - πιθανή αναπροσαρμογή του σχήματος
  - …

Αποφάσεις που πρέπει να ληφθούν

- Τι **ευρετήρια (indexes)** πρέπει να δημιουργήσουμε?
- Μήπως πρέπει να κάνουμε **αλλαγές στο σχήμα** της βάσης?
- Μήπως πρέπει **να ξαναγράψουμε κάποια ερωτήματα** για καλύτερη απόδοση?

Αποφάσεις που πρέπει να ληφθούν

- Τι **ευρετήρια (indexes)** πρέπει να δημιουργήσουμε?
  - Ποιες σχέσεις πρέπει να έχουν ευρετήρια? Πάνω σε ποια πεδία? Αρκεί ένα ευρετήριο ανά σχέση?
- Τα ευρετήρια τι είδους είναι?
  - Clustered? Hash ή δέντρο?

Αποφάσεις που πρέπει να ληφθούν

- Πρέπει να κάνουμε **αλλαγές στο σχήμα** της βάσης δεδομένων?
  - Μήπως να δημιουργήσουμε υποβοηθητικές όψεις (views)?
  - Μήπως εναλλακτικά κανονικοποιημένα σχήματα? Μήπως να από-κανονικοποιήσουμε κάποια σχήματα ?
  - Μήπως να σπάσουμε τους πίνακες σε θεματικά προσδιορισμένους υποπίνακες (horizontal partitioning)?
  - Μήπως να φτιάξουμε αντίγραφα των πινάκων (replication)?

Παράμετροι που παίζουν ρόλο

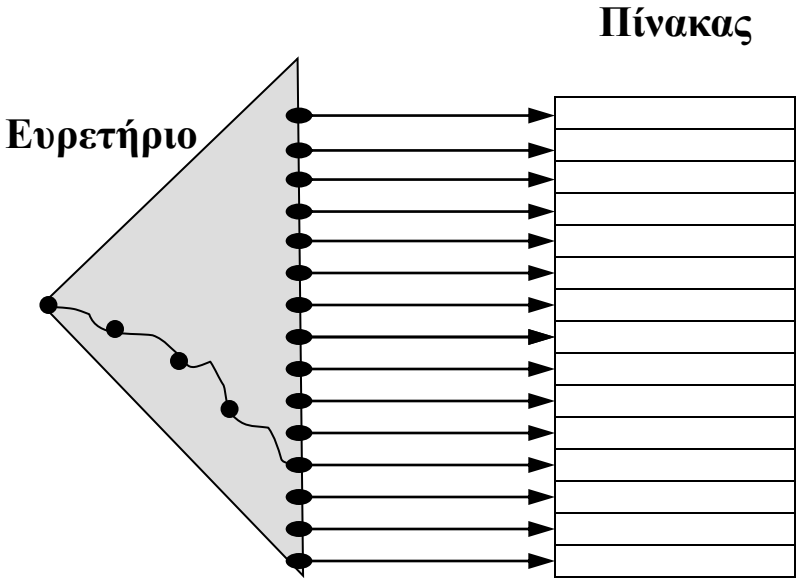
- Ο φόρτος εργασίας του συστήματος (workload)
  - Οι πιο συχνές ερωτήσεις, οι πιο συχνές ενημερώσεις και οι συχνότητές τους
  - Ο αριθμός των χρηστών που είναι ταυτόχρονα συνδεδεμένοι στο σύστημα
- Μη λειτουργικές απαιτήσεις
  - Η διαθεσιμότητα του συστήματος (uptime / downtime)
  - Η μέση/χειρότερη/… επιθυμητή/ανεκτή/… απόκριση του συστήματος
- Οι διαθέσιμοι (υπολογιστικοί, χρηματικοί, ανθρώπινοι) πόροι

Θεματολόγιο

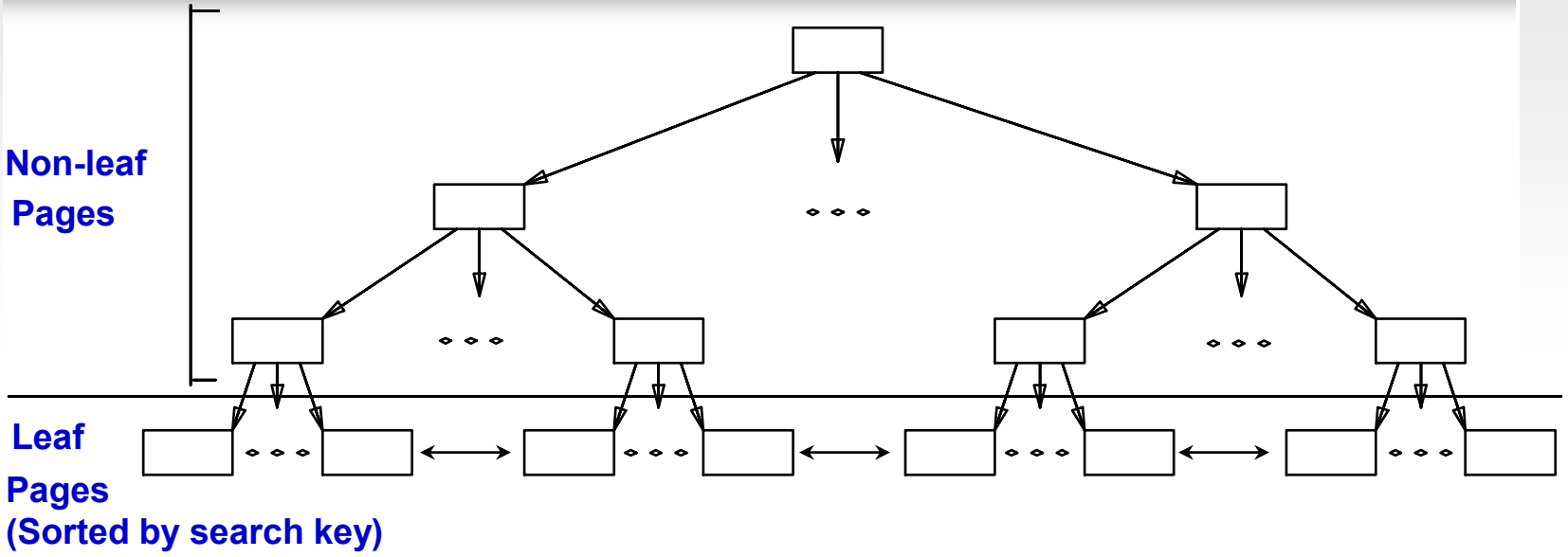
- Η διαδικασία σχεδίασης και ρύθμισης
- Επιλογή ευρετηρίων
  - Γενικές αρχές
  - Συνδέσεις
  - Διάφορα
- Προσαρμογή του σχήματος
  - Αποκανονικοποίηση σχήματος
  - Κατάτμηση σχήματος και χρήση όψεων
- Ρύθμιση ερωτήσεων
- Σύνοψη

Ευρετήρια

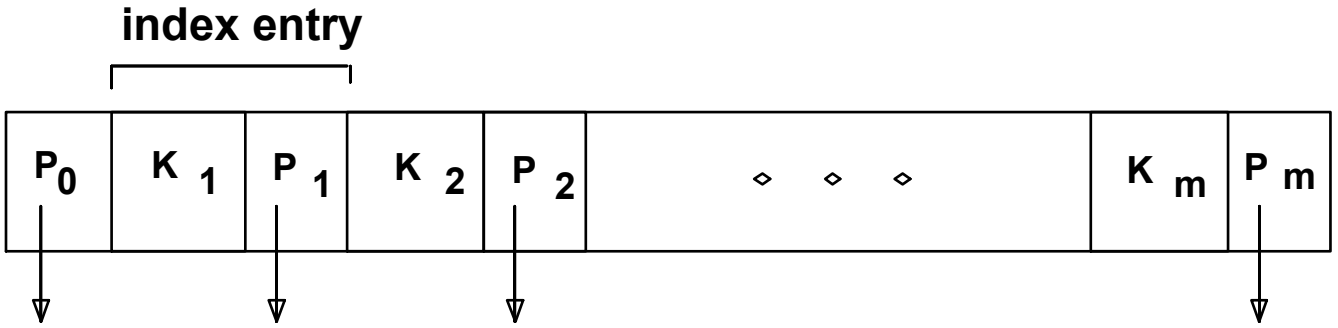
- Βασική λειτουργία: με βάση την τιμή ενός πεδίου, να μας λένε πού βρίσκεται στο δίσκο ολόκληρη η εγγραφή



B+ Tree Indexes

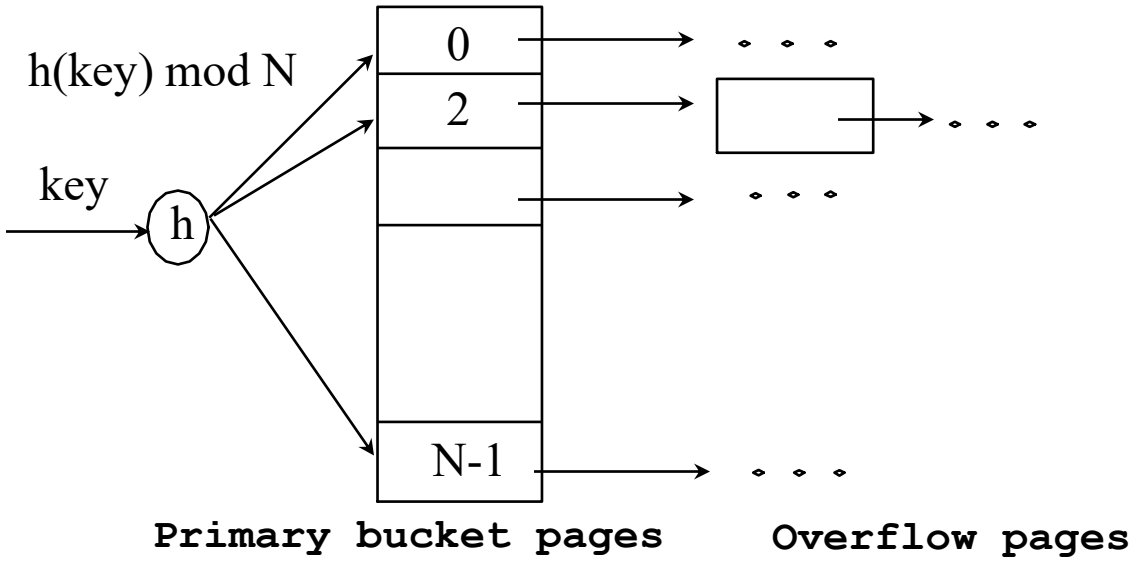


- Φύλλα: pointers στα data + στο επόμενο/προηγούμενο φύλλο
- Εσωτερικοί κόμβοι: pointers στα παιδιά



Hash-Based Indexes

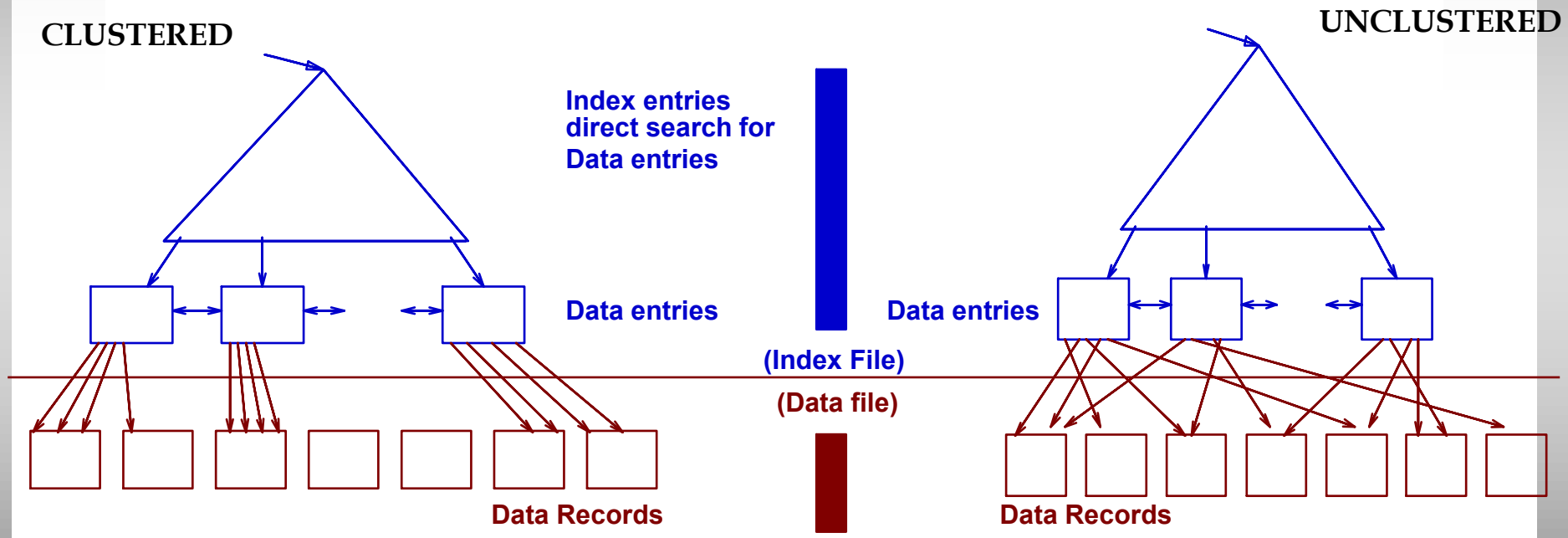
- Ο Index είναι συλλογή **κάδων** (buckets).
  - Bucket = **primary page** και κάποιες **overflow pages**.
  - Οι buckets περιέχουν τις εγγραφές.
- **Hashing function h:** **h(*r*)** = ο bucket στον οποίο ανήκει η εγγραφή r, με βάση κάποιο κλειδί.
- Π.χ., **h(*key*) = (a**key*+b) mod N** όπου N = #buckets



Ευρετήρια -- ορολογία

- **Πρωτεύον** (primary index): πάνω σε πρωτεύον κλειδί
  - Για την ακρίβεια: σε υπερσύνολο του primary key
  - Αντίθετα, *δευτερεύον* ευρετήριο
- **Πυκνό** (dense index): για κάθε τιμή στον πίνακα, έχω και μια τιμή στον index
  - Αντίθετα, *αραιό* ευρετήριο (π.χ., μια τιμή για κάθε σελίδα του πίνακα)
- **Συγκροτημένο** (clustered index) : τα δεδομένα του πίνακα είναι αποθηκευμένα στο σκληρό δίσκο με την ίδια σειρά που έχει και ο index
  - Για κάθε πίνακα μπορώ να έχω ακριβώς ένα τέτοιο index
  - Αντίθετα, *μη συγκροτημένο* ευρετήριο

Συγκρότηση ευρετηρίου



Θεματολόγιο

- Η διαδικασία σχεδίασης και ρύθμισης
- Επιλογή ευρετηρίων
  - Γενικές αρχές
  - Συνδέσεις
  - Διάφορα
- Προσαρμογή του σχήματος
  - Αποκανονικοποίηση σχήματος
  - Κατάτμηση σχήματος και χρήση όψεων
- Ρύθμιση ερωτήσεων
- Σύνοψη

Γενικές αρχές επιλογής ευρετηρίων

- Οι αποφάσεις που έχουμε να πάρουμε:
  - Αν θα φτιάξουμε κάποιο ευρετήριο
  - Τι είδους ευρετήριο να φτιάξουμε
  - Αν θα είναι συγκροτημένο ή όχι
  - Η σειρά των πεδίων ευρετηριοποίησης
- Τα κριτήρια που επηρεάζουν την απόφαση είναι:
  - Το είδος των ερωτήσεων που πάμε να εξυπηρετήσουμε
  - Η επίπτωση της ύπαρξης του ευρετηρίου στις αλλαγές (INS/DEL/UPD)

Κριτήρια επιλογής ευρετηρίων

- Το **είδος των ερωτήσεων**:
  - Σημειακές επιλογές (point queries) εξυπηρετούνται βέλτιστα από hash index και καλά από B+ trees

```
SELECT * FROM EMP WHERE AGE = 25;
```

  - Επιλογές εύρους (range queries) εξυπηρετούνται πιθανώς από B+ trees

```
SELECT * FROM EMP WHERE AGE BETWEEN 25 AND 30;
```

  - Συνδέσεις (joins): εξαρτάται από το πλάνο (και αντίστροφα, το πλάνο εξαρτάται και από τι index θα φτιάξουμε)

Κριτήρια επιλογής ευρετηρίων

- Η **επίπτωση των ευρετηρίων**:
  - Η ύπαρξη ευρετηρίου επηρεάζει την απόδοση στις εισαγωγές, διαγραφές, ανανεώσεις (ΟΛΟΙ οι indexes πρέπει να ενημερωθούν)
    - Ενίοτε, ο index επιταχύνει κάποια update και delete
  - Ο χώρος που χρησιμοποιείται αυξάνει
  - Το κόστος διαχείρισης αυξάνει: συχνά χρειάζεται να καταστρέψουμε και να ξαναφτιάξουμε τον index λόγω κακής δομής του (περίπου όταν αλλάξει κατά 30%)

Άλλα σοβαρά ζητήματα

- Πρωτεύοντα κλειδιά: εν γένει, αν το σύστημα δεν φτιάξει ένα B+ δέντρο αυτόματα, έχει νόημα να σκεφτεί κανείς να φτιάξει
- Ο ρόλος του WHERE clause: εκεί κρίνεται η επιλογή ευρετηρίων
- Η σημασία της συγκρότησης
- Η σημασία των ευρετηρίων στα joins

Θεματολόγιο

- Η διαδικασία σχεδίασης και ρύθμισης
- Επιλογή ευρετηρίων
  - Γενικές αρχές
  - Συνδέσεις
  - Διάφορα
- Προσαρμογή του σχήματος
  - Αποκανονικοποίηση σχήματος
  - Κατάτμηση σχήματος και χρήση όψεων
- Ρύθμιση ερωτήσεων
- Σύνοψη

Επιλογή ευρετηρίων για ερωτήσεις σύνδεσης

- Η σύνδεση είναι από τις πιο ακριβές πράξεις που ζητάμε από ένα DBMS
- Πρακτικά ζητάμε από το σύστημα να συνδυάσει δύο (ή περισσότερους πίνακες) στα κοινά τους στοιχεία

Σύνδεση

```
SELECT  E.ename, D.mgr
FROM  Emp E, Dept D
WHERE  D.dname='Toys' AND E.dno=D.dno
```

Eno	Ename	Dno	
103	Demis	1	..
202	Vassilis	2	..
208	Ilia	2	..
..	..	..	..

Dno	Dname	Mgr	
1	Toys	103	..
2	Food	208	..
3	Clothes	657	..
..	..	..	..

$$\text{Emp} \triangleright\!\triangleleft_{\text{DNO=DNO}} \text{Dept}$$

Σύνδεση με Nested Loops

```
SELECT  E.ename, D.mgr
FROM  Emp E, Dept D
WHERE  D.dname='Toys' AND E.dno=D.dno
```

Για κάθε D ∈ Dept [& Dname='Toys']

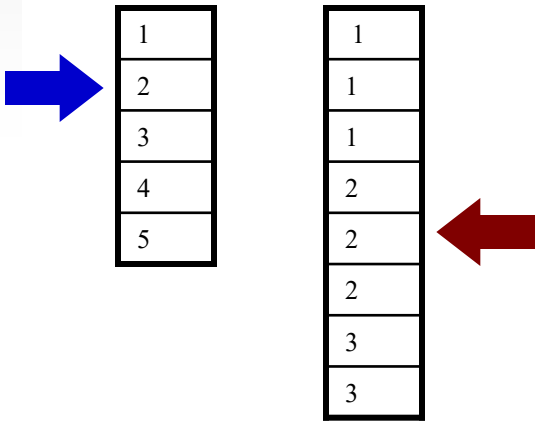
  Για κάθε E ∈ Emp

    Αν ταιριάζουν στο πεδίο Dno

      Τότε επέστρεψε E.ename, D.mgr

Τι αλλάζει αν χρησιμοποιήσω index για την έσω σχέση?

Σύνδεση με Merge Join




```
SELECT  E.ename, D.mgr
FROM  Emp E, Dept D
WHERE  D.dname='Toys' AND E.dno=D.dno
```

Ταξινόμησε τα Emp, Dept με βάση το πεδίο Dno

Για κάθε D ∈ Dept [& Dname='Toys']

Για κάθε E ∈ Emp με ίδιο Dno

Επέστρεψε E.ename, D.mgr

Επιλογή ευρετηρίων για ερωτήσεις σύνδεσης

- Nested Loops: Hash index για την έσω σχέση
  - Βάζουμε την πιο μικρή σχέση έξω και για κάθε εγγραφή της έξω σχέσης $\mathbf{t_{outer}}$, αναζητούμε τη σχετική εγγραφή της έσω σχέσης $\mathbf{t_{inner}}$, με κλειδί την τιμή της $\mathbf{t_{outer}}$ στο πεδίο σύνδεσης
- Merge Join: Clustered B+ tree στα πεδία που γίνεται η σύνδεση
  - Οι δύο σχέσεις είναι ήδη sorted στα πεδία του join => ο index χρησιμεύει a priori στην ταξινόμηση του πίνακα

Παράδειγμα 1

```
SELECT  E.ename, D.mgr
FROM  Emp E, Dept D
WHERE  D.dname='Toys' AND E.dno=D.dno
```

- **DEPT**: υποψήφια πεδία για ευρετήριο είναι τα **Dept.dname** και **Dept.dno**
- **EMP**: υποψήφιο πεδίο για ευρετήριο είναι το **Emp.dno**

Παράδειγμα 1

```
SELECT  E.ename, D.mgr
FROM  Emp E, Dept D
WHERE  D.dname='Toys' AND E.dno=D.dno
```

- **DEPT**: Ένας hash index στο *D.dname* βοηθά στο να βρούμε γρήγορα τα (λίγα) departments με dname = 'Toy'.
  - Τότε, το ευρετήριο στο D.dno δεν χρειάζεται.
- **EMP**: Ένας hash index στο *E.dno* βοηθά στο να βρούμε γρήγορα τις πλειάδες από το Emp για κάθε πλειάδα από το Dept.

Επιλογή & Σύνδεση με Nested Loops

```
SELECT  E.ename, D.mgr
FROM  Emp E, Dept D
WHERE  D.dname='Toys' AND E.dno=D.dno
```

Μέσω hash index: Dname='Toys'

Για κάθε D ∈ Dept *//εξωτερικός πίνακας*

Μέσω hash index: Για κάθε E ∈ Emp *//εσωτερικός πίνακας*

Αν ταιριάζουν στο πεδίο Dno

Τότε επέστρεψε E.ename, D.mgr

Παράδειγμα 1

```
SELECT  E.ename, D.mgr
FROM  Emp E, Dept D
WHERE  D.dname='Toys' AND E.dno=D.dno
AND E.age=25
```

- Τι γίνεται αν προσθέσουμε: " ... AND E.age=25" ?
  - Μπορούμε να απομονώσουμε τις σχετικές πλειάδες από το Emp μέσω ενός ευρετηρίου στο *E.age*
  - Το ίδιο για τις πλειάδες του Dept και την επιλογή στο πεδίο *D.dname*
  - Μετά από αυτά, είναι πολύ πιθανό, τα ενδιάμεσα αποτελέσματα να είναι τόσο μικρά, ώστε το ευρετήριο στο *E.dno* να μη χρειάζεται.

Παράδειγμα 2

```
SELECT  E.ename, D.mgr
FROM  Emp E, Dept D
WHERE  E.sal BETWEEN 10000 AND 20000
AND E.hobby='Stamps' AND E.dno=D.dno
```

- **EMP**: Με βάση τις επιλογές στο *sal* και *hobby*, φαίνεται ότι η ενδιάμεση σχέση από το Emp θα προκύψει αρκετά μικρή ώστε να χρησιμοποιηθεί ως η εξωτερική σχέση.
- **DEPT**: Συμφέρει να χτίσουμε ένα hash index στο *D.dno*.

Παράδειγμα 2

```
SELECT  E.ename, D.mgr
FROM  Emp E, Dept D
WHERE  E.sal BETWEEN 10000 AND 20000
AND E.hobby='Stamps' AND E.dno=D.dno
```

- **EMP**: Τι δείκτη να χτίσουμε στον πίνακα Emp?
  - B+ tree στο *E.sal*
  - Εναλλακτικά, κάποιο ευρετήριο στο *E.hobby*
- Αρκεί ένα εξ' αυτών, στη βάση της επιλεκτικότητας των συνθηκών
- Συνήθως, οι συνθήκες ισότητας είναι πιο περιοριστικές από συνθήκες εύρους.

ΠΡΟΣΟΧΗ!

- Με βάση και τα δύο προηγούμενα παραδείγματα, φαίνεται ότι η επιλογή των ευρετηρίων εξαρτάται από το πλάνο που θα μας δώσει ο βελτιστοποιητής (optimizer). *Μάθετε τι κάνει ο optimizer του DBMS σας!*
- Στα καινούρια DBMS αυτό λειτουργεί και αντίστροφα: μπορούμε να επιβάλλουμε στον βελτιστοποιητή τι πλάνο να ακολουθήσει…

Θεματολόγιο

- Η διαδικασία σχεδίασης και ρύθμισης
- Επιλογή ευρετηρίων
  - Γενικές αρχές
  - Συνδέσεις
  - Διάφορα
- Προσαρμογή του σχήματος
  - Αποκανονικοποίηση σχήματος
  - Κατάτμηση σχήματος και χρήση όψεων
- Ρύθμιση ερωτήσεων
- Σύνοψη

Ευρετήρια πολλών γνωρισμάτων

- Έστω οι ερωτήσεις:

```
SELECT * FROM EMP WHERE AGE BETWEEN 25 AND 30 AND
   SAL >= 100;
SELECT * FROM EMP WHERE AGE = 25;
```

- Μπορώ να χτίσω διάφορα ευρετήρια:
  - Στο Age: B+ tree / hash
  - Στο Sal: B+ tree
  - Στο <Age,Sal>: ??
  - Στο <Sal,Age>: ??

Ευρετήρια πολλών γνωρισμάτων

- Εν γένει είναι καλύτερα να έχουμε ευρετήριο σε ένα σύνθετο κλειδί, παρά δύο χωριστά ευρετήρια
- Η **σειρά παίζει ρόλο**: αν έχω ένα κλειδί <Eno,Ename,Dno> μπορώ να απαντώ καλά σε ερωτήσεις πάνω στα:
  - Eno,Ename,Dno
  - Eno,Ename
  - Eno
- αλλά όχι στα:
  - Ename,Dno
  - Dno
  - …

"The curse of dimensionality"

Clustering

```
SELECT *
FROM Emp
WHERE E.sal>=100
```

- Αν η συνθήκη επιλογής φέρει κάτι σαν το 60% των εγγραφών, τότε η χρήση του ευρετηρίου απλώς μας καθυστερεί
- Ακόμα κι αν η επιλεκτικότητα είναι μικρή (π.χ., 10%), πιθανώς ο μη κατακερματισμός του index να μας καθυστερήσει
- Αν ο πίνακας είναι clustered στο sal όμως, αρκεί
  - να κατεβούμε μέχρι το φύλλο με την τιμή 100,
  - να μεταβούμε στην πρώτη εγγραφή του πίνακα και
  - να μη χρησιμοποιήσουμε το ευρετήριο άλλο, αλλά να διασχίσουμε σειριακά τον πίνακα (αφού αυτός είναι ταξινομημένος στο sal)

Clustering

```
SELECT  Dno, Count(*)
FROM  Emp
WHERE  E.sal>=100
Group By Dno
```

- Πού θα χτίσω index και τι είδους ?
- Αν η συνθήκη στο sal δεν είναι επιλεκτική, μπορώ να έχω ένα clustered B+ tree στο Dno
- Αλλιώς, μπορώ να έχω ένα non-clustered index στο sal, που βγάζει μια μικρή ενδιάμεση σχέση πριν γίνει το group by.

Clustering & Συνδέσεις

```
SELECT  E.ename, D.mgr
FROM  Emp E, Dept D
WHERE  D.dname='Toy' AND E.dno=D.dno
```

- Σε ότι αφορά την εσωτερική σχέση, βοηθά ο index που την αφορά να είναι clustered, ιδίως όταν πρόκειται να ανακτηθούν πολλές εγγραφές
- Ένας clustered index στο *E.dno* βοηθά

Clustering & Συνδέσεις

```
SELECT  E.ename, D.mgr
FROM  Emp E, Dept D
WHERE  E.hobby='Stamps' AND E.dno=D.dno
```

- Αν πολλοί υπάλληλοι μαζεύουν γραμματόσημα, μπορεί να συμφέρει καλύτερα ένα Merge Join. Τότε ένας *clustered* index στο *D.dno* θα βοηθούσε.

Γενικά περί clustering

- Όσο πιο πολλές πλειάδες χρησιμοποιούνται από μια σχέση, τόσο πιο πολύ το clustering βοηθά (ελαχιστοποιούμε την μετακίνηση της κεφαλής του δίσκου)
- Προσοχή, το clustering μπορεί να γίνει σε ακριβώς ένα index!
- Επίσης, το clustering κοστίζει στις εισαγωγές/διαγραφές/ανανεώσεις, όπου αφενός πρέπει να τηρείται η σειρά, να μη γεμίζουν οι σελίδες, να μην είναι σχεδόν άδειες, κλπ.
- Θυμηθείτε επίσης: αν ο πίνακας δεν είναι ταξινομημένος, μπορεί να φέρουμε μια σελίδα στη μνήμη τόσες φορές, όσες κι οι εγγραφές της...

Θεματολόγιο

- Η διαδικασία σχεδίασης και ρύθμισης
- Επιλογή ευρετηρίων
  - Γενικές αρχές
  - Συνδέσεις
  - Διάφορα
- Προσαρμογή του σχήματος
  - Αποκανονικοποίηση σχήματος
  - Κατάτμηση σχήματος και χρήση όψεων
- Ρύθμιση ερωτήσεων
- Σύνοψη

Επανάληψη: Κανονικές Μορφές

- Πόσο καλό είναι το σχήμα της ΒΔ?
  - Τόσο καλό, όσο αποφεύγει άχρηστες επαναλήψεις της πληροφορίας
- Κανονικοποίηση: η διαδικασία μέσω της οποίας μετατρέπουμε ένα σχήμα σε κανονική μορφή, ήτοι, σε μια μορφή που πληροί συγκεκριμένα χαρακτηριστικά σχεδίασης
- 4 κανονικές μορφές (Normal Forms): 1NF, 2NF, 3NF, BCNF

Επανάληψη: Κανονικές Μορφές

- Functional Dependency (FD): ορισμός εξάρτησης, σε επίπεδο σχήματος
- Αν X,Y σύνολα γνωρισμάτων της σχέσης R, $X \rightarrow Y$, σημαίνει ότι η τιμή για το X καθορίζει μονοσήμαντα την τιμή του Y
- Prime attribute: γνώρισμα που είναι υποσύνολο ενός candidate key
- Trivial FD: $A,B \rightarrow A$
- Partial FD: $A \rightarrow C$ και $A \subset$ candidate key, π.χ., A,B
- Transitive FD: $A \rightarrow B \rightarrow C$, A prime & B non-prime

Πρώτη Κανονική Μορφή -- 1NF

- **1NF: Κάθε γνώρισμα παίρνει ακριβώς μία τιμή**

- Π.χ., η σχέση

  `EMP(EID, ENAME, ECHILDREN),`

- με πιθανή τιμή

  `EMP(4, Κώστας, {Μαρία, Γιάννης}),`

- ΔΕΝ είναι σε 1NF μορφή

Δεύτερη Κανονική Μορφή – 2NF

- **2NF: 1NF + δεν υπάρχουν partial dependencies**
- Με άλλα λόγια αν το (A,B) είναι candidate key και έχω κάποια εξάρτηση της μορφής A→C, τότε δεν είμαι σε 2NF
- Π.χ., η παρακάτω σχέση ΔΕΝ είναι σε 2NF

SUPPLY(SNO,PARTNO,PRJNO,SCITY,PRJCITY,QTY),

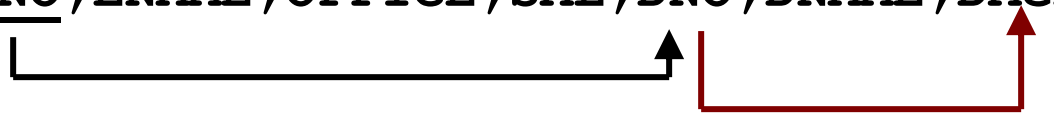
Τρίτη Κανονική Μορφή – 3NF

- **3NF: 2NF + δεν υπάρχουν transitive dependencies σε non-prime attributes**
- Με άλλα λόγια, αν μια $X \rightarrow A$, είναι στη σχέση R, τότε
  - Είτε είναι trivial ($A \subset X$),
  - Είτε το X είναι κάποιο υπερκλειδί, είτε
  - $A \subset$ candidate key
- Θυμηθείτε ότι υπονοούνται όλες οι FD's από το πρωτεύον κλειδί σε όλα τα άλλα γνωρίσματα

Τρίτη Κανονική Μορφή – 3NF

- Π.χ., η παρακάτω σχέση, ενώ είναι σε 2NF, ΔΕΝ είναι σε 3NF

EMPDEPT(ENO, ENAME, OFFICE, SAL, DNO, DNAME, DMGR),

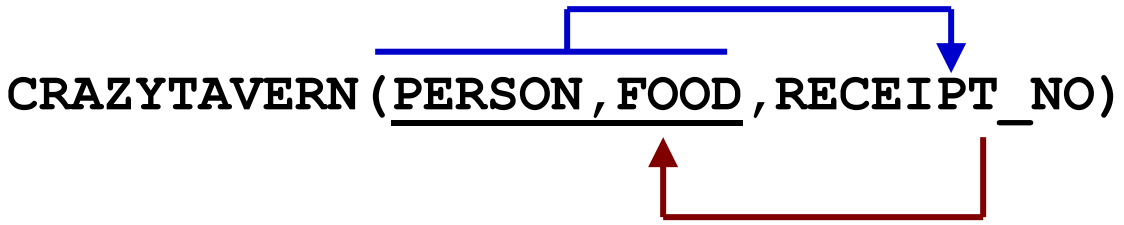


Boyce-Codd Normal Form – BCNF

- **BCNF: 3NF + δεν υπάρχουν καθόλου transitive dependencies**
- Με άλλα λόγια, αν μια $X \rightarrow A$, είναι στη σχέση R, τότε
  - Είτε είναι trivial ($A \subset X$),
  - Είτε το X είναι κάποιο υπερκλειδί,
- Θυμηθείτε ότι στην 3NF, μπορούσε το A να είναι υποσύνολο ενός candidate key

Boyce-Codd Normal Form – BCNF

- Π.χ., η παρακάτω σχέση, ενώ είναι σε 3NF, ΔΕΝ είναι σε BCNF



- Κάθε απόδειξη αφορά ακριβώς ένα φαγητό και κάθε άνθρωπος τρώει ακριβώς ένα φαΐ

Κανονικοποίηση

- Για να φέρουμε μια σχέση R σε BCNF μορφή,
  - για κάθε FD **X** $\rightarrow$**Y** που δημιουργεί πρόβλημα, φτιάχνουμε μια σχέση R'(X,Y) και αφαιρούμε τα γνωρίσματα Y από την R
  - επαναλάβατε μέχρι όλες οι προκύπτουσες σχέσεις να είναι σε BCNF μορφή
- Τι είναι:
  - Σύνδεση χωρίς **απώλειες** (lossless join)?
  - Διατήρηση των εξαρτήσεων (dependency preservation)?

Προσαρμογή του σχήματος

- Η εκλογή του σχήματος της βάσης δεδομένων μπορεί να επηρεαστεί από χαρακτηριστικά που αφορούν τις ερωτήσεις. Πιθανές επεμβάσεις μπορεί να είναι:
  - Να αρκεστούμε σε 3NF αντί για BCNF σχήμα.
  - Αντίθετα, να αναλύσουμε περαιτέρω ένα BCNF σχήμα
  - Να αποκανονικοποιήσουμε το σχήμα ή να του προσθέσουμε επιπλέον πεδία.
  - Να επιλέξουμε κάποιου είδους οριζόντιες κατατμήσεις του σχήματος.

ΠΡΟΣΟΧΗ!!!!

- Οι αποκανονικοποιήσεις και σχετικές άλλες προσαρμογές του σχήματος απαιτούν εμπειρία
- **ΠΡΩΤΑ ΣΧΕΔΙΑΖΟΥΜΕ ΤΗΝ ΚΑΝΟΝΙΚΟΠΟΙΗΜΕΝΗ ΜΟΡΦΗ ΚΑΙ ΜΕΤΑ ΑΠΟΚΑΝΟΝΙΚΟΠΟΙΟΥΜΕ!!**

- Λάθη στην κανονικοποίηση δεν «ακυρώνονται» από απαιτήσεις λόγω ερωτήσεων: **η κανονικοποίηση οφείλει να γίνει ως βήμα στη διαδικασία σχεδίασης!**

Προσαρμογή του σχήματος

- Πολλές φορές καλούμαστε να ανασχεδιάσουμε μια βάση δεδομένων με κακή απόδοση στην απάντηση των ερωτήσεων. Η διαδικασία προσαρμογής του σχήματος αποκαλείται, τότε, ***εξέλιξη του σχήματος*** (*schema evolution*).
- Στις βάσεις που ήδη χρησιμοποιούνται, υπάρχουν ήδη εφαρμογές που ανακτούν ή τροποποιούν τα δεδομένα. Πολλές φορές ο καλύτερος τρόπος για να αλλάξουμε το σχήμα χωρίς να επηρεαστούν οι υπάρχουσες εφαρμογές είναι να χρησιμοποιήσουμε *όψεις* (*views*).
- **ΠΟΤΕ ΔΕΝ ΑΠΟΚΑΝΟΝΙΚΟΠΟΙΟΥΜΕ ΑΝ ΠΡΩΤΑ ΔΕΝ ΕΧΟΥΜΕ ΤΗΝ ΚΑΝΟΝΙΚΟΠΟΙΗΜΕΝΗ ΜΟΡΦΗ !!**

Παράδειγμα

Contracts (Cid, Sid, Jid, Did, Pid, Qty, Val)
Depts (Did, Budget, Report)
Suppliers (Sid, Address)
Parts (Pid, Cost)
Projects (Jid, Mgr)

- Έστω η παραπάνω βάση δεδομένων, η οποία αφορά τα τμήματα (Depts) μιας τεχνικής εταιρείας.
- Η εταιρεία προμηθεύεται ανταλλακτικά (Parts) από προμηθευτές (Suppliers). Η εταιρεία εμπλέκεται σε διάφορα έργα (Projects).
- Η βασική σχέση που θα μας απασχολήσει είναι η σχέση Contracts, στην οποία θα αναφερόμαστε ως CSJDPQV από τα ονόματα των γνωρισμάτων της

Παράδειγμα

Contracts (Cid, Sid, Jid, Did, Pid, Qty, Val)
Depts (Did, Budget, Report)
Suppliers (Sid, Address)
Parts (Pid, Cost)
Projects (Jid, Mgr)

- Contracts (Cid, Sid, Jid, Did, Pid, Qty, Val): CSJDPQV.
- Οι παρακάτω συνθήκες ισχύουν:

  JP → C, SD → P, C είναι το πρωτεύον κλειδί.

- Τροφή για σκέψη:
  - Τι σημαίνουν οι εν λόγω συναρτησιακές εξαρτήσεις ? Ποιες είναι όλες οι συναρτησιακές εξαρτήσεις της σχέσης Contracts?
  - Ποια είναι τα υποψήφια κλειδιά της σχέσης ?
  - Σε τι κανονική μορφή είναι η σχέση ?

Πρώτη απόπειρα κανονικοποίησης

- Contracts (Cid, Sid, Jid, Did, Pid, Qty, Val): CSJDPQV, JP → C, SD → P, C είναι το πρωτεύον κλειδί.
- Ας υποθέσουμε ότι σπάμε τη σχέση ως PartSupp(SDP) και Contracts(CSJDQV).
  - Ποια είναι τα υποψήφια κλειδιά κάθε σχέσης ?
  - Σε τι κανονική μορφή είναι κάθε σχέση ?
  - Έχουμε απώλειες στο σπάσιμο?
  - Αν σας δοθεί ότι και οι δύο σχέσεις είναι σε BCNF, έχουμε απώλεια εξαρτήσεων ?

Contracts (Cid, Sid, Jid, Did, Pid, Qty, Val)
Depts (Did, Budget, Report)
Suppliers (Sid, Address)
Parts (Pid, Cost)
Projects (Jid, Mgr)

JP → C, SD → P

Κι άλλη μία …

- Contracts (Cid, Sid, Jid, Did, Pid, Qty, Val): CSJDPQV, JP → C, SD → P, C είναι το πρωτεύον κλειδί.
- Ας υποθέσουμε ότι σπάμε τη σχέση ως PartSupp(SDP), ProjectParts(CJP) και Contracts(CSJDQV)
  - Ποια είναι τα υποψήφια κλειδιά κάθε σχέσης ?
  - Σε τι κανονική μορφή είναι κάθε σχέση ?
  - Έχουμε απώλειες στο σπάσιμο?
  - Αν σας δοθεί ότι και οι τρεις σχέσεις είναι σε BCNF, έχουμε απώλεια εξαρτήσεων ?

Contracts (Cid, Sid, Jid, Did, Pid, Qty, Val)
Depts (Did, Budget, Report)
Suppliers (Sid, Address)
Parts (Pid, Cost)
Projects (Jid, Mgr)

JP → C, SD → P

Ενίοτε η αποκανονικοποίηση αρκεί…

- Ας υποθέσουμε ότι σπάμε τη σχέση ως PartSupp(SDP), ProjectParts(CJP) και Contracts(CSJDQV)
- Έστω η παρακάτω ερώτηση (που μπορεί να είναι σημαντικό να απαντηθεί γρήγορα):
  - *Πόσα ανταλλακτικά του part P έχουν παραγγελθεί στο contract C ?*
- Στο αρχικό σχήμα απαντάται αμέσως, στο κανονικοποιημένο με σύνδεση…

Ίσως η αποκανονικοποιημένη μορφή φτάνει …

Contracts (Cid, Sid, Jid, Did, Pid, Qty, Val)
Depts (Did, Budget, Report)
Suppliers (Sid, Address)
Parts (Pid, Cost)
Projects (Jid, Mgr)

JP → C, SD → P

Αποκανονικοποίησης συνέχεια...

- Έστω ότι είναι σημαντικό η παρακάτω ερώτηση να απαντηθεί πολύ γρήγορα:
  - *Υπάρχει συμβόλαιο που η αξία του να είναι μεγαλύτερη από τον προϋπολογισμό ενός department?*
- Έστω ότι αποφασίζουμε να προσθέσουμε ένα πεδίο *budget* στη σχέση *Contracts*. Θα συμβολίζουμε το εν λόγο πεδίο με *B* και η σχέση γίνεται:

- Contracts (Cid, Sid, Jid, Did, Pid, Qty, Val, Budget): CSJDPQVB,
- JP → C, SD → P, D → B, C είναι το πρωτεύον κλειδί

Αποκανονικοποίησης συνέχεια...

- Έχουμε δικαίωμα να κάνουμε τέτοια αλλαγή?
  - **ΣΠΑΝΙΑ!**
  - **ΜΕ ΠΟΛΥ ΠΡΟΣΟΧΗ!**
- ΜΟΝΟ αν η απόδοση είναι κακή, ενώ
  - η απόδοση στην ερώτηση αυτή είναι πολύ σημαντική,
  - οι άλλες τεχνικές (προσθήκη ευρετηρίων, άλλο 3NF σχήμα, ...) αποτυγχάνουν

Γενικά, η μεταβολή του σχήματος για λόγους απόδοσης είναι ΠΟΛΥ ΕΠΙΚΙΝΔΥΝΗ και πρέπει να γίνεται με ΜΕΓΑΛΗ ΠΡΟΣΟΧΗ!

Θυμηθείτε: 50% του κόστους του λογισμικού δαπανάται στη συντήρηση!

Η μεταβολή του σχήματος για λόγους απόδοσης είναι ΠΟΛΥ ΕΠΙΚΙΝΔΥΝΗ και πρέπει να γίνεται με ΜΕΓΑΛΗ ΠΡΟΣΟΧΗ!

- Ένας από τους λόγους που εισήχθησαν οι κανονικές μορφές είναι διότι εγγυώνται τη συνέπεια των δεδομένων ΑΣΧΕΤΩΣ των εφαρμογών που τα παράγουν!
  - **Στις βάσεις δεδομένων έχουμε <u>ΠΑΝΤΑ</u>, ΠΕΡΙΣΣΟΤΕΡΟΥΣ ΤΟΥ ΕΝΟΣ ΤΡΟΠΟΥ ΝΑ ΠΕΙΡΑΞΟΥΜΕ ΤΑ ΔΕΔΟΜΕΝΑ (π.χ., από την εφαρμογή, από το SQL prompt, από το bulk loader, από άλλες εφαρμογές, από stored procedures, …)**
- 50% του κόστους του λογισμικού δαπανάται στη συντήρηση!

Θεματολόγιο

- Η διαδικασία σχεδίασης και ρύθμισης
- Επιλογή ευρετηρίων
  - Γενικές αρχές
  - Συνδέσεις
  - Διάφορα
- Προσαρμογή του σχήματος
  - Αποκανονικοποίηση σχήματος
  - Κατάτμηση σχήματος και χρήση όψεων
- Ρύθμιση ερωτήσεων
- Σύνοψη

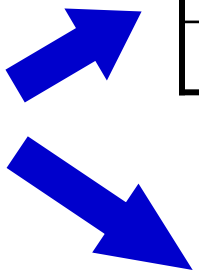
Κατάτμηση σχήματος

- Μπορούμε να σπάσουμε έναν πίνακα σε «υπο-πίνακες»
- Δύο εναλλακτικές:
  - Κάθετη κατάτμηση: χωρίζουμε τα πεδία του πίνακα σε δύο (ή περισσότερες) ομάδες, και φτιάχνουμε από ένα νέο πίνακα για κάθε μία ομάδα (προσοχή στο πρωτεύον κλειδί)
  - Οριζόντια κατάτμηση: ανάλογα με το περιεχόμενο, χωρίζουμε τις πλειάδες του πίνακα σε ομάδες

Κάθετη Κατάτμηση σχήματος

Emp

Eno	**Ename**	**Dno**	**Sal**	**Hobby**
103	Demis	1	10K	Music
202	Vassilis	2	8K	Food
208	Ilia	2	10K	Coffee
123	Theo	2	5K	sports



Emp_Info

Eno	**Ename**	**Dno**
103	Demis	1
202	Vassilis	2
208	Ilia	2
123	Theo	2

Emp_Personal

Eno	**Sal**	**Hobby**
103	10K	Music
202	8K	Food
208	10K	Coffee
123	5K	sports

Κάθε νέος πίνακας έχει το πρωτεύον κλειδί του Emp

Η ένωση των στηλών δίνει τις στήλες του αρχικού πίνακα

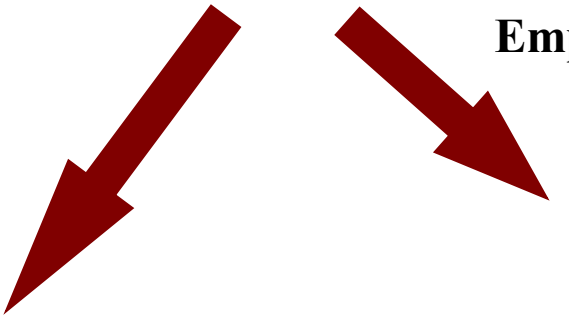
Οριζόντια Κατάτμηση σχήματος

Emp

Eno	**Ename**	**Dno**	**Sal**	**Hobby**
103	Demis	1	10K	Music
202	Vassilis	2	8K	Food
208	Ilia	2	10K	Coffee
123	Theo	2	5K	sports

Το σχήμα των πινάκων είναι ίδιο

Η ένωση των πλειάδων δίνει τις πλειάδες του αρχικού πίνακα



Emp_Rich *Sal >= 10K*

Eno	**Ename**	**Dno**	**Sal**	**Hobby**
103	Demis	1	10K	Music
208	Ilia	2	10K	Coffee

Emp_Poor *Sal < 10K*

Eno	**Ename**	**Dno**	**Sal**	**Hobby**
202	Vassilis	2	8K	Food
123	Theo	2	5K	sports

Οριζόντια Κατάτμηση σχήματος

- Ας υποθέσουμε ότι έχουμε συχνά ερωτήσεις που αφορούν τους υπαλλήλους με υψηλό μισθό στον πίνακα Emp.Η συνθήκη *sal >= 10K* εμφανίζεται συχνά.
- Μπορούμε να φτιάξουμε ένα clustered B+ tree στο πεδίο *sal*
- Μπορούμε όμως και να κατατμήσουμε τον πίνακα Emp σε δύο άλλους πίνακες *Emp_Poor, Emp_Rich*

Οριζόντια Κατάτμηση σχήματος

- Μπορούμε όμως και να κατατμήσουμε τον πίνακα Emp σε δύο άλλους πίνακες *Emp_Poor, Emp_Rich* με το ίδιο σχήμα
- Ενώ η συμπεριφορά είναι ίδια, πλέον δε χρειαζόμαστε το ευρετήριο
- Με το χώρο που εξοικονομήσαμε, μπορούμε να χτίσουμε ευρετήρια σε άλλα πεδία
- *… βέβαια, οι ερωτήσεις που αφορούν όλους τους υπαλλήλους, είναι πιο πολύπλοκες πλέον*

Οριζόντια Κατάτμηση σχήματος

```
CREATE VIEW  Contracts(cid, sid, jid, did, pid, qty, val) AS
        SELECT  * FROM  LargeContracts
        UNION
        SELECT  * FROM  SmallContracts
```

- Μπορούμε να καλύψουμε την αντικατάσταση του πίνακα Contracts από τη σχετική όψη (view)
- Οι εφαρμογές που έχουν όμως ερωτήσεις με τη συνθήκη *val>10000* πρέπει να ξαναγραφούν αντικαθιστώντας το *Contracts* με το *LargeContracts*, αν το ζητούμενο είναι η απόδοση (προσοχή: συντακτικά και σημασιολογικά εξακολουθούν να είναι σωστές)

Θεματολόγιο

- Η διαδικασία σχεδίασης και ρύθμισης
- Επιλογή ευρετηρίων
  - Γενικές αρχές
  - Συνδέσεις
  - Διάφορα
- Προσαρμογή του σχήματος
  - Αποκανονικοποίηση σχήματος
  - Κατάτμηση σχήματος και χρήση όψεων
- Ρύθμιση ερωτήσεων
- Σύνοψη

Ρύθμιση Όψεων και Ερωτήσεων

- Ενίοτε το DBMS δεν εκτελεί το πλάνο που έχουμε υποθέσει ότι θα εκτελέσει. Πιθανές αιτίες:
  - Επιλογές που αφορούν null τιμές
  - Επιλογές που αφορούν αριθμητικές ή αλφαριθμητικές εκφράσεις
  - Επιλογές που αφορούν συνθήκες με OR
  - Κακή εκτίμηση του μεγέθους των ενδιάμεσων αποτελεσμάτων, ή κακή χρήστη των ευρετηρίων.

Ρύθμιση Όψεων και Ερωτήσεων

- Αν μια ερώτηση καθυστερεί,
  - πιθανώς να χρειάζεται να χτίσουμε κάποιο index (αν δεν υπάρχει) ή να κάνουμε re-built κάποιον υπάρχοντα index (που έχει μεγαλώσει άσχημα λόγω διαγραφών και εισαγωγών)
  - πιθανώς να χρειάζεται να υποχρεώσουμε το DBMS να επιλέξει άλλο πλάνο
  - πιθανώς να χρειάζεται να ξαναγράψουμε την ερώτηση εξ' ολοκλήρου..

Επανεγγραφή ερωτήσεων

- ***Οδηγία: Χρησιμοποιήστε όσο το δυνατόν λιγότερα "query blocks" !!***

```
SELECT DISTINCT *
  FROM Sailors S
 WHERE S.sname IN
       (SELECT Y.sname
          FROM YoungSailors Y)
```

=

```
SELECT DISTINCT S.*
  FROM Sailors S,
       YoungSailors Y
 WHERE S.sname = Y.sname
```

Επανεγγραφή ερωτήσεων

- *Ενίοτε αυτό είναι αδύνατο …*

```
SELECT *
  FROM Sailors S
 WHERE S.sname IN
       (SELECT DISTINCT Y.sname
          FROM YoungSailors Y)
```

≠

```
SELECT S.*
  FROM Sailors S,
       YoungSailors Y
 WHERE S.sname = Y.sname
```

Το διάσημο COUNT Bug

```
SELECT dname FROM Department D
 WHERE D.num_emps  > ANY
          (SELECT COUNT(*) FROM Employee E
            WHERE D.building = E.building)
```

```
CREATE VIEW Temp (empcount, building) AS
       SELECT COUNT(*), E.building
         FROM Employee E
       GROUP BY E.building

SELECT dname
  FROM Department D,Temp
 WHERE D.building = Temp.building
   AND D.num_emps > Temp.empcount;
```

- Τι γίνεται όταν η Employee είναι άδεια?

Μπορώ να ενσωματώσω εμφωλευμένες ερωτήσεις στην κύρια ερώτηση?

- DISTINCT στην εξώτερη ερώτηση: *Μπορώ να αγνοήσω τις διπλοεγγραφές μέσα.*
- DISTINCT σε εσωτερική ερώτηση, χωρίς DISTINCT εξωτερικά: *Δύσκολο να τις μετατρέψω.*
- Εμφωλευμένες ερωτήσεις μέσα σε OR: *Δύσκολο να τις μετατρέψω.*
- ALL subqueries: *Δύσκολο να τις μετατρέψω.*
  - EXISTS και ANY είναι ίδιες με IN.
- Αθροιστικές ερωτήσεις μέσα στις εμφωλευμένες: *Δύσκολο να τις μετατρέψω*

Μπορώ να ενσωματώσω εμφωλευμένες ερωτήσεις στην κύρια ερώτηση?

- Πολλά συστήματα κάνουν την ενσωμάτωση αυτόματα.
- Αυτό δουλεύει κυρίως για την περίπτωση του IN (που είναι και η σχετικά πιο απλή) – για τις υπόλοιπες εξαρτάται από το σύστημα

Βελτιώσεις ...

- ***Οδηγία: Σπάστε τα OR => βελτιστοποιημένα πλάνα!***

```
SELECT E.dno
FROM Emp E
WHERE age = 20 OR
      age = 10
```

=

```
SELECT E.dno
FROM Emp E
WHERE age = 20
      UNION
SELECT E.dno
FROM Emp E
WHERE age = 10
```

Βελτιώσεις ...

- ***Οδηγία**: Use "BETWEEN" (ιδίως αν έχεις B+ στο sal)*

```
SELECT E.dno
FROM Emp E
WHERE age <= 20 AND
      age >= 10
```

=

```
SELECT E.dno
FROM Emp E
WHERE age BETWEEN
      10 AND 20
```

Βελτιώσεις ...

- ***Οδηγία**: απλοποιήστε την αριθμητική (για να εκμεταλλευτείτε τον index στον sal)*

```
SELECT E.dno
FROM Emp E
WHERE 2*age > 20
```

=

```
SELECT E.dno
FROM Emp E
WHERE age > 10
```

Βελτιώσεις ...

- ***<u>Οδηγία</u>: αποφύγετε άχρηστα GROUP BY και DISTINCT => άχρηστη ταξινόμηση εσωτερικά!***

```
SELECT Avg(E.sal)
FROM Emp E
GROUP BY E.DNO
HAVING E.DNO = 110
```

=

```
SELECT Avg(E.sal)
FROM Emp E
WHERE E.DNO = 110
```

```
SELECT DISTINCT *
FROM Emp E
```

=

```
SELECT *
FROM Emp E
```

Βελτιώσεις ...

- ***Οδηγία: αποφύγετε άχρηστα joins***

```
SELECT E.EID, E.Name
FROM Emp E, Dept D
WHERE E.DID = D.DID
```

=

```
SELECT E.EID,E.name
FROM Emp E
WHERE E.DID NOT NULL
```

Αν το E.DID είναι ξένο κλειδί στο D.DID

Θεματολόγιο

- Η διαδικασία σχεδίασης και ρύθμισης
- Επιλογή ευρετηρίων
  - Γενικές αρχές
  - Συνδέσεις
  - Διάφορα
- Προσαρμογή του σχήματος
  - Αποκανονικοποίηση σχήματος
  - Κατάτμηση σχήματος και χρήση όψεων
- Ρύθμιση ερωτήσεων
- **Σύνοψη**

ΜΗΝ ΞΕΧΝΑΤΕ!!

- Η σχεδίαση μιας ΒΔ έχει πολλά στάδια: *ανάλυση απαιτήσεων*, *σχεδίαση* και *επεξεργασία του λογικού σχήματος*, *σχεδίαση του φυσικού σχήματος* και *ρύθμιση*.
  - Το λογισμικό εξελίσσεται: κάθε τόσο, χρειάζεται να εκτελεσθούν ξανά κάποια από τα στάδια αυτά!
- Η ρύθμιση της βάσης δεδομένων έχει να κάνει αφενός με τον *προγραμματιστικό φόρτο* και αφετέρου με το *φόρτο εργασίας (workload)* και την επίδοση του συστήματος. Πρέπει πάντα να είστε ενήμεροι για τις σοβαρές διεργασίες και την εσωτερική οργάνωση της βάσης!

ΜΗΝ ΞΕΧΝΑΤΕ!!

- Μπορούμε να προβούμε σε διάφορες επεμβάσεις για να προσαρμόσουμε το σχήμα στις απαιτήσεις απόδοσης.
- Σε επίπεδο λογικού σχήματος:
  - Μπορεί να επιλέξουμε 3NF αντί για BCNF κανονική μορφή.
    - Το ποια διαμόρφωση θα έχει τελικά το σχήμα, ανάμεσα στις πολλές εναλλακτικές σχεδιάσεις είναι επίσης ρυθμίσιμο.
  - Μπορεί να αποκανονικοποιήσουμε μια σχέση.
  - Μπορεί να κατατμήσουμε το σχήμα της βάσης
- Σε επίπεδο φυσικού σχήματος:
  - Μπορεί να χρειαστεί να επανεγγράψουμε ερωτήσεις για να επιτύχουμε καλύτερη επίδοση, να χτίσουμε ευρετήρια ή να μετακινήσουμε δεδομένα

ΜΗΝ ΞΕΧΝΑΤΕ!!

- Η σχεδίαση και η ρύθμιση είναι **διαρκείς** διαδικασίες
- Η **τυπική διαδικασία** διαχείρισης περιλαμβάνει λήψη αντιγράφων (backup), έλεγχο επίδοσης, κατανάλωση πόρων (δίσκου, CPU, …) και διαρκές monitoring του συστήματος
- **Οι επεμβάσεις είναι κομμάτι της δουλειάς του DBA** (επιπλέον backups, rebuilding των ευρετηρίων, ανακατανομή του χώρου στους δίσκους...)