ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ

Βάσεις Δεδομένων ΙΙ

Βελτιστοποίηση Ερωτημάτων

Μανώλης Μαραγκουδάκης

Τμήμα Μηχανικών Πληροφοριακών και Επικοινωνιακών Συστημάτων

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Αιγαίου**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Πανεπιστήμιο Αιγαίου-
Τμήμα Μηχανικών
Πληροφοριακών και
Επικοινωνιακών Συστημάτων

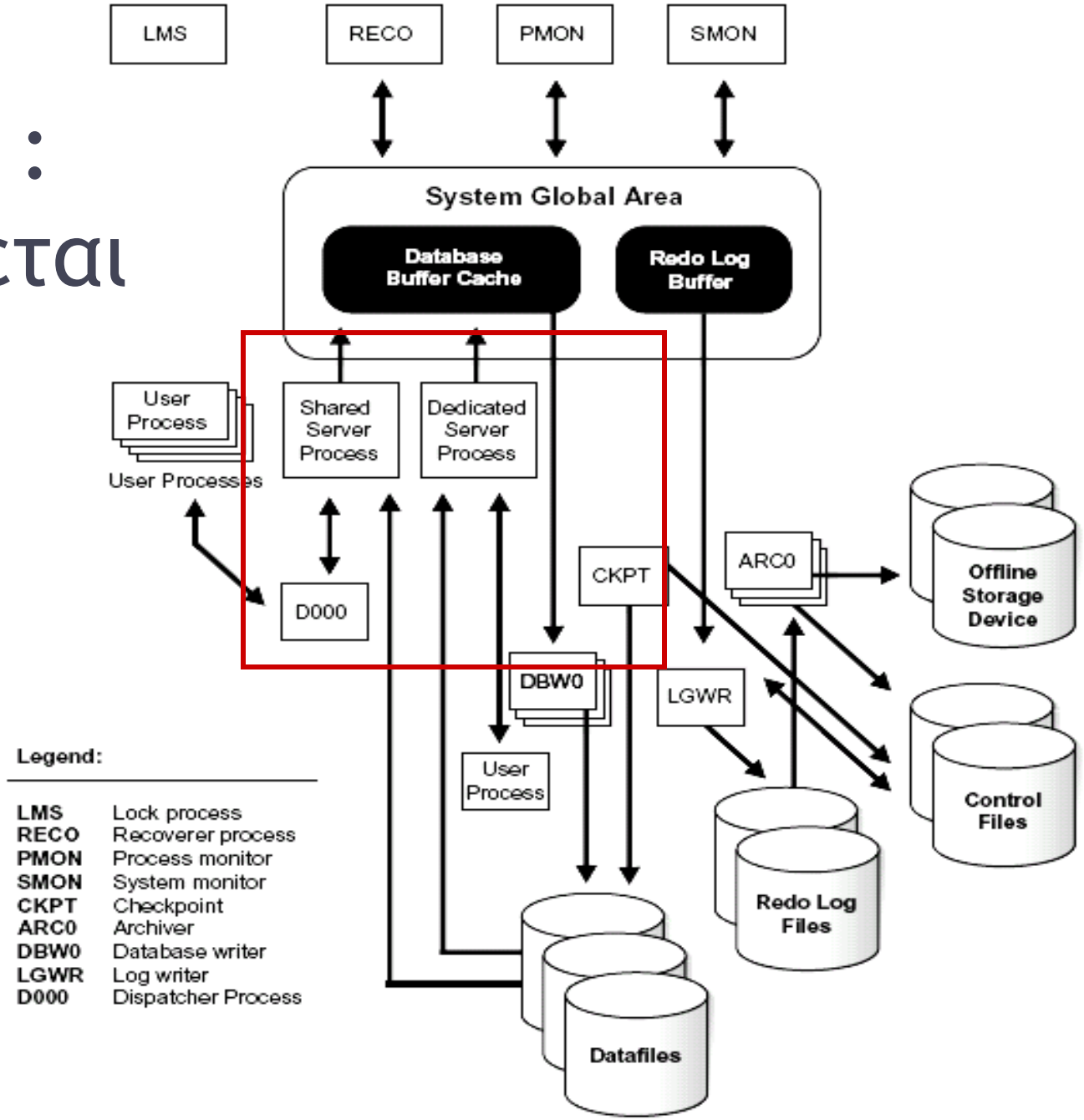
Βάσεις Δεδομένων ΙΙ

Ενότητα 5- Βελτιστοποίηση Ερωτημάτων

Μανώλης Μαραγκουδάκης

www.icsd.aegean.gr/mmarag

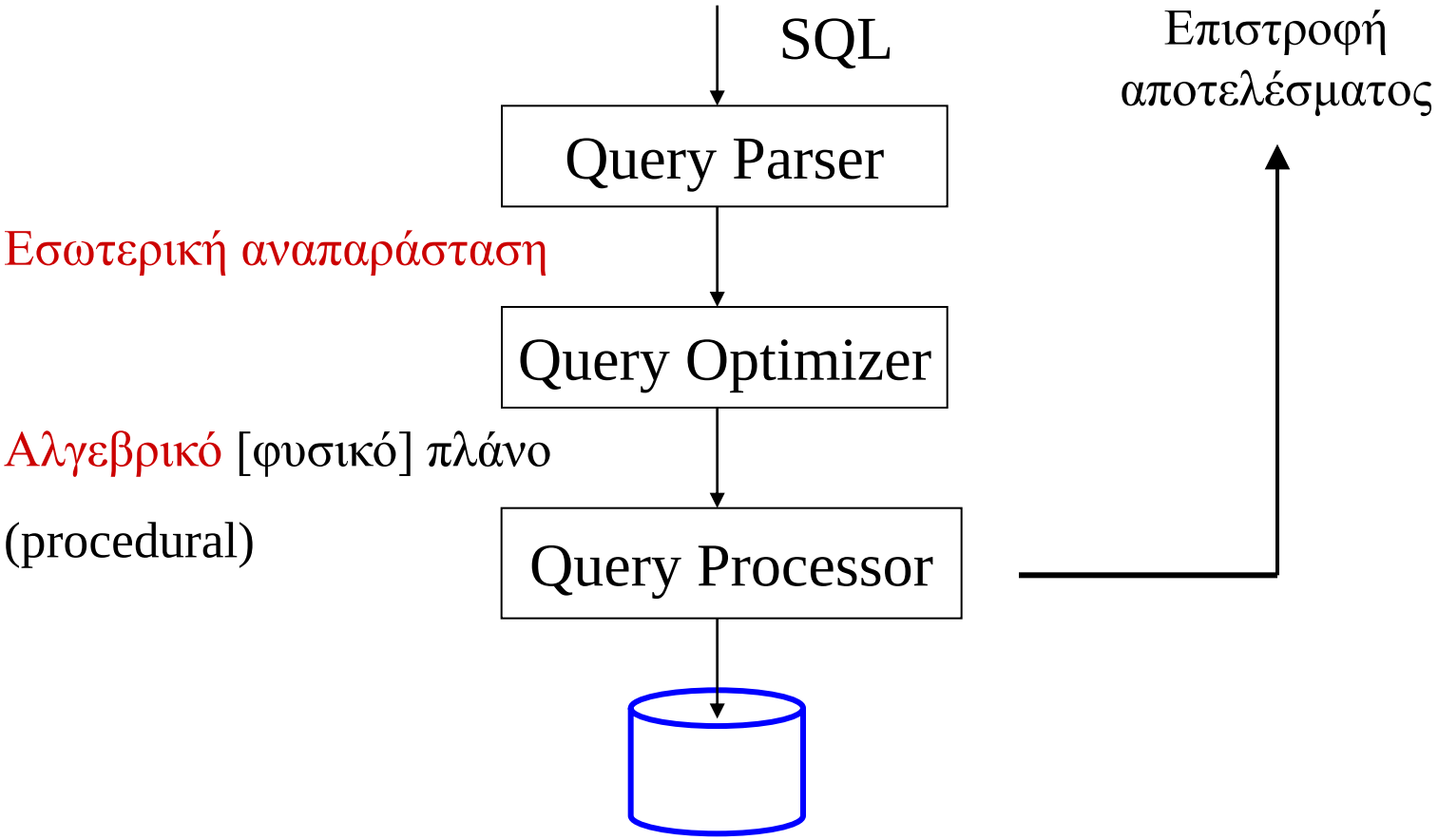
Server processes : και τι γίνεται εκεί?



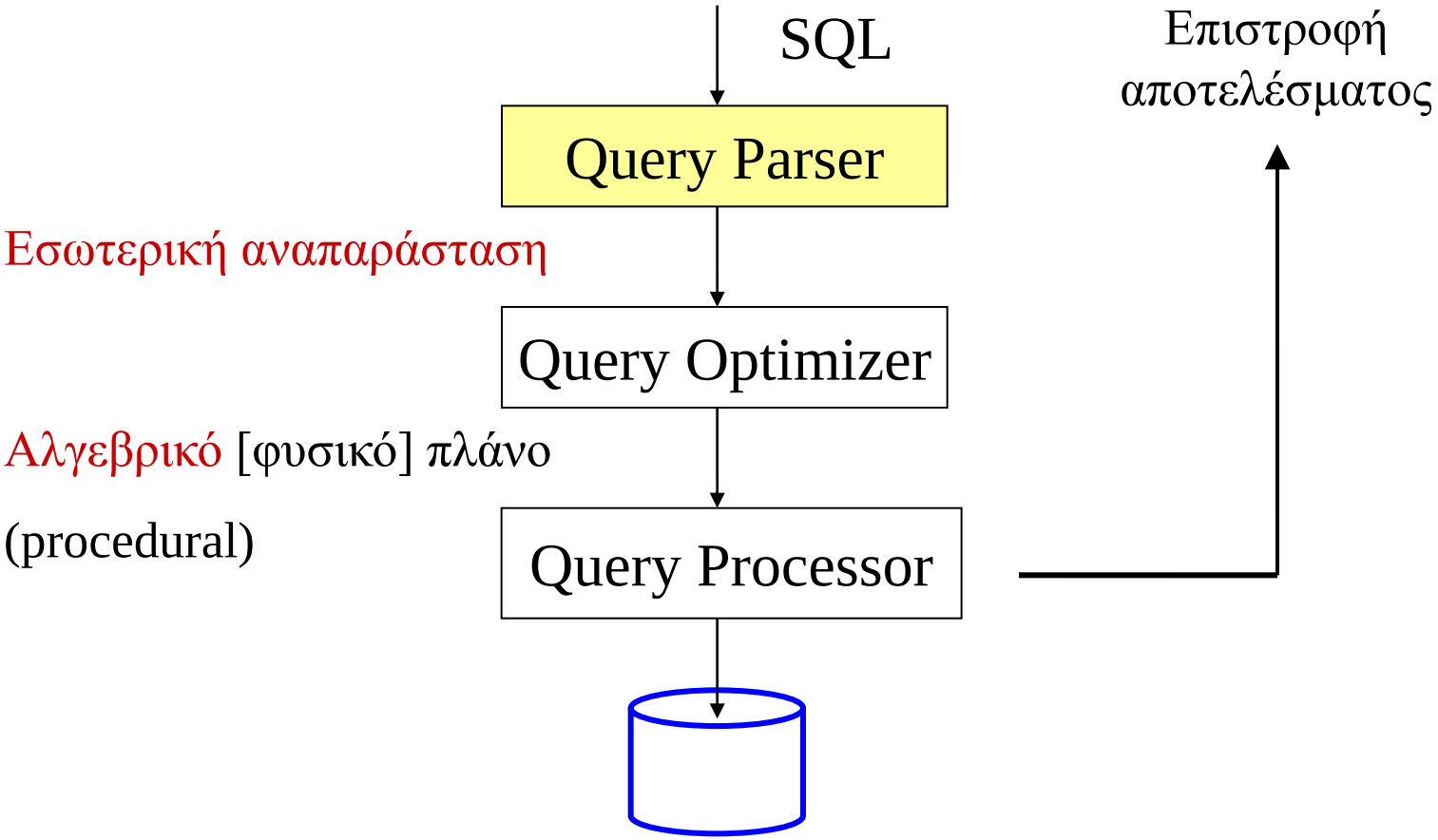
Επεξεργασία ερωτήσεων

- Οι clients θέτουν μια ερώτηση SQL στο server
- Χαρακτηριστικά: δηλωτική γλώσσα + το μόνο που πρέπει να ξέρει ο χρήστης είναι ονόματα πινάκων και γνωρισμάτων
- Το DBMS μέσω της server process κάνει τα εξής:
  - Parsing
  - Έλεγχο συντακτικής ορθότητας
  - Σύνθεση ενός αλγεβρικού πλάνου εκτέλεσης (μέσω ενός δέντρου τελεστών)
  - Εκτέλεση του πλάνου και επιστροφή του αποτελέσματος

Επεξεργασία ερωτήσεων

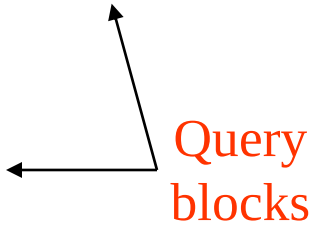


Επεξεργασία ερωτήσεων

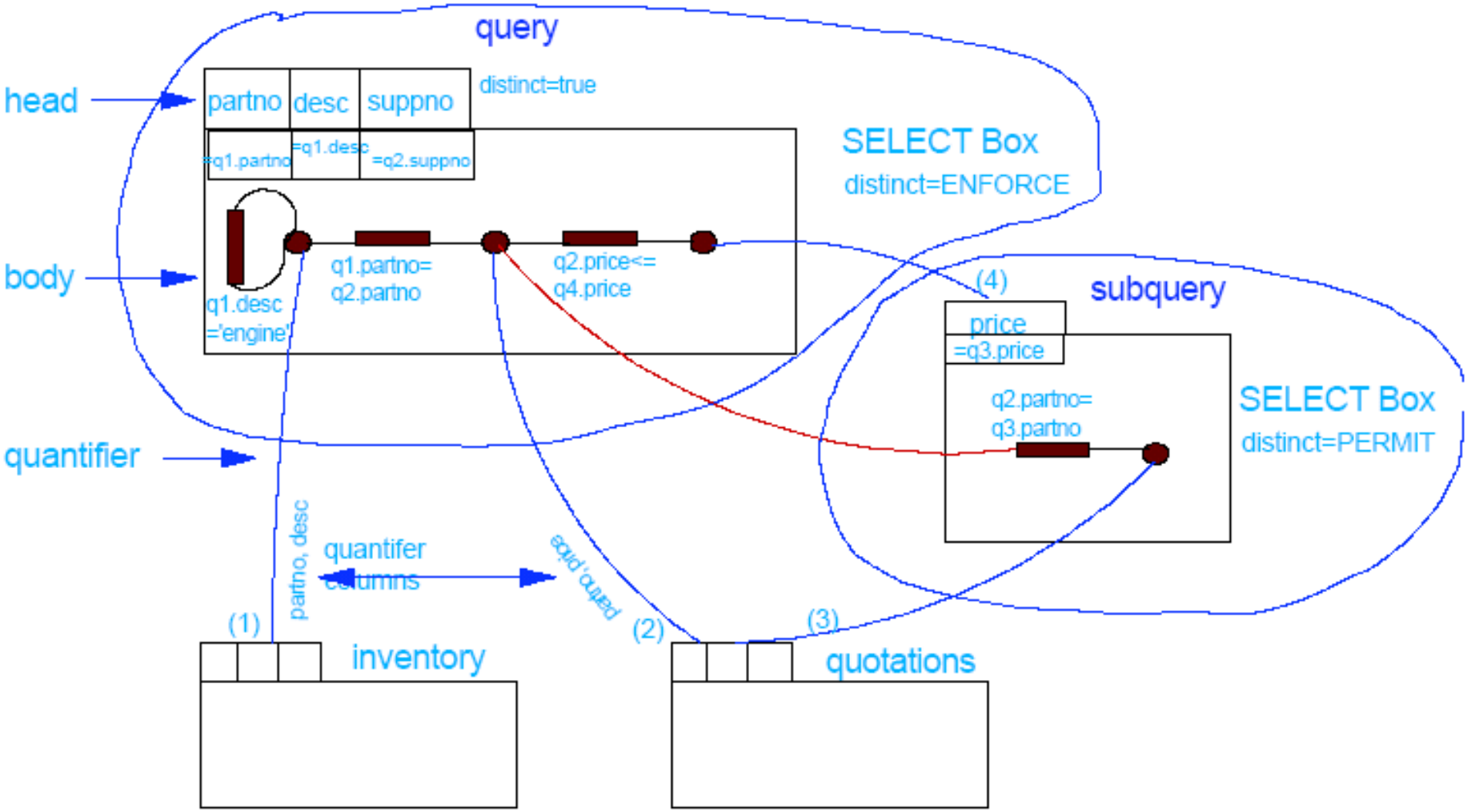


Παράδειγμα Ερώτησης

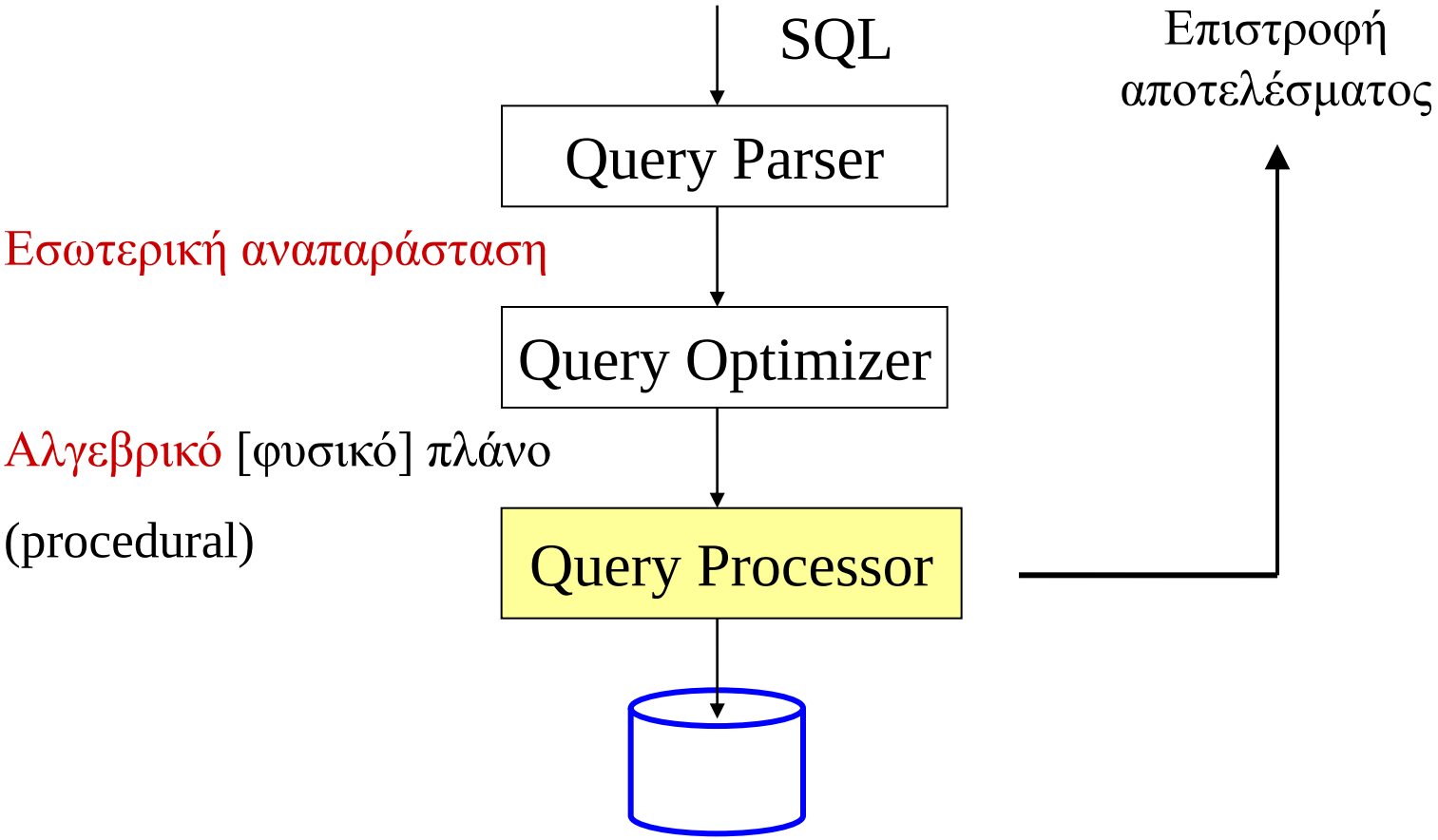
```
SELECT DISTINCT q1.partno, q1.descr, q2.suppno
FROM inventory q1, quotations q2
WHERE q1.partno = q2.partno
   AND q1.descr = 'engine'
   AND q2.price <= ALL
      ( SELECT q3.price
      FROM quotations q3
      WHERE q2.partno = q3.partno
      );
```



Query Graph Model



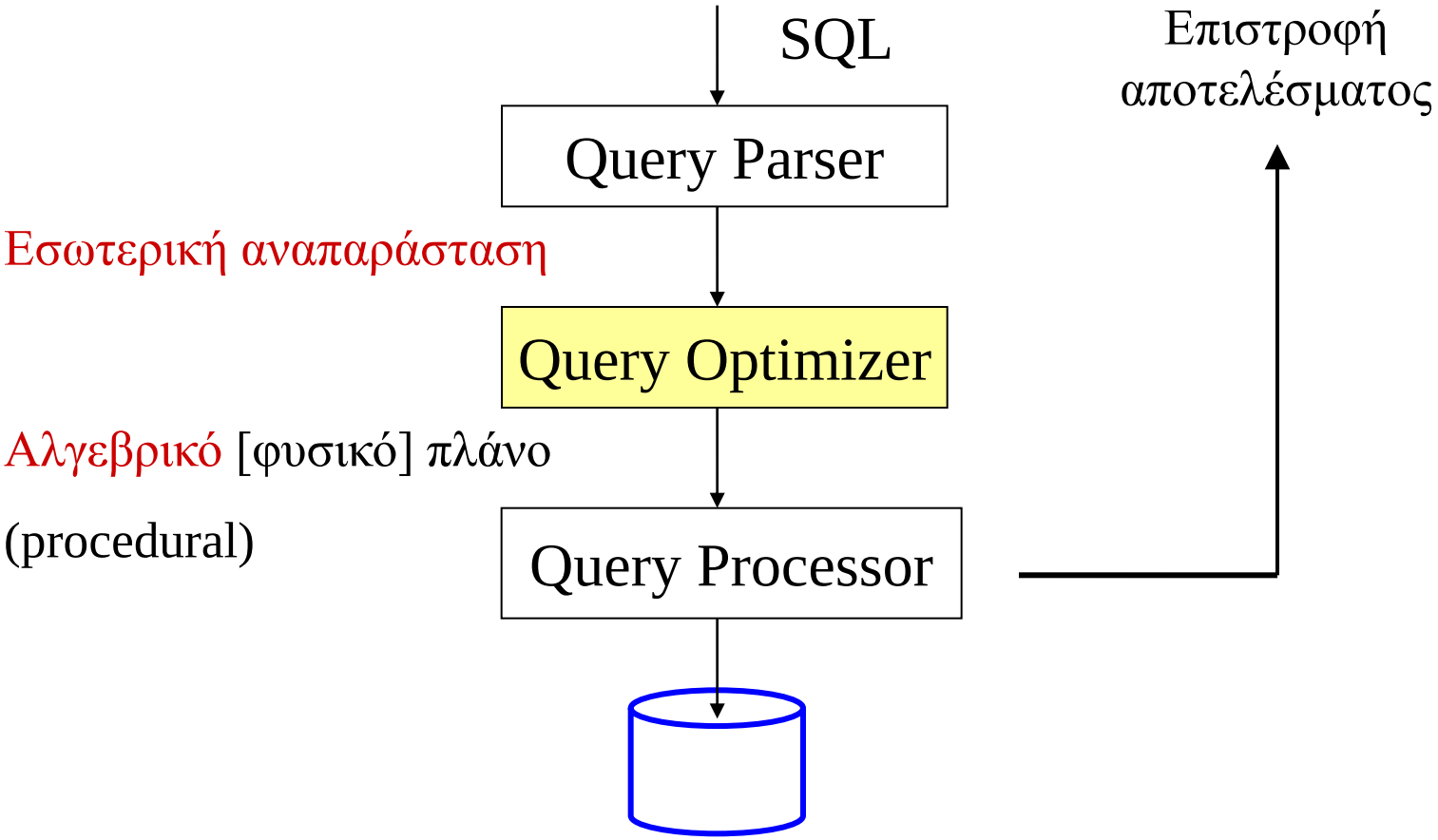
Επεξεργασία ερωτήσεων



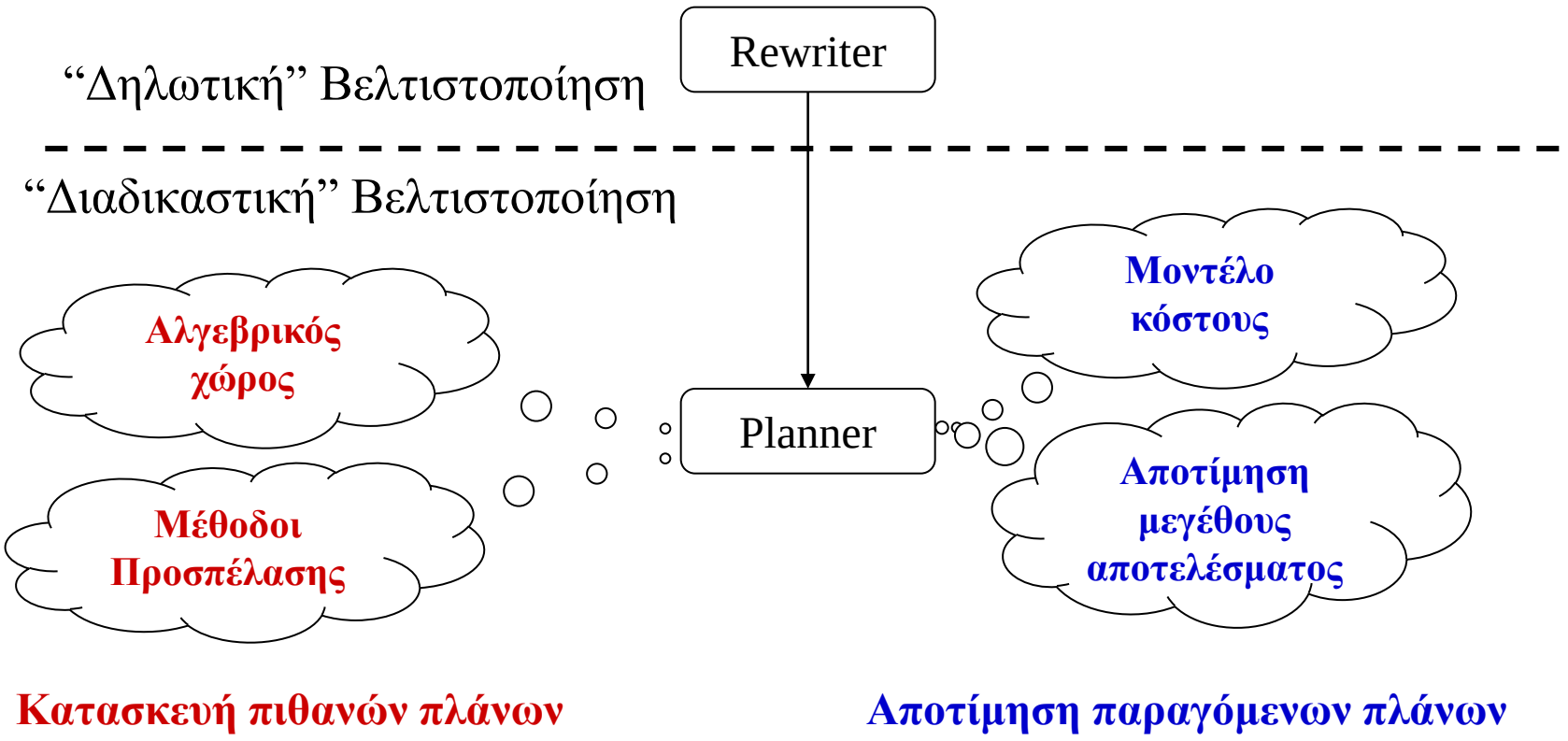
Επεξεργασία ερωτήσεων

- Δοθέντος του πλάνου, κάθε λογικός τελεστής (π.χ., σ, π, ▷◁) έχει περισσότερους του ενός τρόπους να εκτελεστεί φυσικά (δηλ., στην πράξη).
- Στην προηγούμενη διάλεξη εξετάσαμε τέτοιους διαφορετικούς φυσικούς τρόπους εκτέλεσης για τους πιο σημαντικούς τελεστές της σχεσιακής άλγεβρας.

Επεξεργασία ερωτήσεων



Αφαιρετική δομή του βελτιστοποιητή



Επίπεδα βελτιστοποίησης

- Υπάρχει ένα επίπεδο «δηλωτικής» βελτιστοποίησης, ή επανεγγραφής, όπου παράγουμε λογικά ισοδύναμους τρόπους να εκφράσουμε μια ερώτηση μέσω του **rewriter**

```
select *
from emp
where emp.dno in
        (select dept.dno
         from dept) and
        sal > 100K
```

```
select name,age,sal,ndo
from emp, dept
where emp.dno = dept.dno
        and sal > 100K
```

Επίπεδα βελτιστοποίησης

- Υπάρχει ένα επίπεδο «δηλωτικής» βελτιστοποίησης, ή επανεγγραφής, όπου παράγουμε λογικά ισοδύναμους τρόπους να εκφράσουμε μια ερώτηση μέσω του **rewriter**
- Αυτό συμπεριλαμβάνει, συνήθως:
  - Μετατροπή εκφράσεων σε «βολική» μορφή
  - Απλοποίηση εμφωλευμένων ερωτήσεων
  - Σημασιολογικά έξυπνες μετατροπές

Επίπεδα βελτιστοποίησης

- Υπάρχει ένα επίπεδο «διαδικαστικής» βελτιστοποίησης, όπου παράγουμε (όλα τα) διαφορετικά πλάνα εκτέλεσης μέχρι να διαλέξουμε το πιο αποδοτικό. Η δουλειά αυτή ανατίθεται στον planner.
- Ο planner οφείλει:
  - Να αποφασίσει ποια πλάνα εκτέλεσης θα δημιουργηθούν
  - Ποιο εξ αυτών είναι το καλύτερο

Ισοδυναμίες της σχεσιακής άλγεβρας

- Επιλογή: $\sigma_{c1\wedge\ldots\wedge cn}(R) \equiv \sigma_{c1}(\ldots \sigma_{cn}(R))$ *(Cascade)*

  $\sigma_{c1}(\sigma_{c2}(R)) \equiv \sigma_{c2}(\sigma_{c1}(R))$ *(Commute)*

- Προβολή: $\pi_{a1}(R) \equiv \pi_{a1}(\ldots(\pi_{an}(R)))$ *(Cascade)*

- Σύνδεση: $R \bowtie (S \bowtie T) \equiv (R \bowtie S) \bowtie T$ *(Associative)*

  $(R \bowtie S) \equiv (S \bowtie R)$ *(Commute)*

  $R \bowtie (S \bowtie T) \equiv (T \bowtie R) \bowtie S$

Σύνθετες ισοδυναμίες

- $\sigma_\theta(\pi_{\boldsymbol{A}}(R)) \equiv \pi_{\boldsymbol{A}}(\sigma_\theta(R))$, αρκεί τα πεδία που εμπλέκονται στη συνθήκη θ να είναι υποσύνολο των πεδίων του $\boldsymbol{A}$
- $\sigma_\theta(R \times S) \equiv (R \triangleright\triangleleft_\theta S)$, αρκεί τα πεδία που εμπλέκονται στη συνθήκη θ να είναι από τις σχέσεις R και S
- $\sigma_\theta(R \triangleright\triangleleft S) \equiv \sigma_\theta(R) \triangleright\triangleleft S$, αρκεί τα πεδία που εμπλέκονται στη συνθήκη θ να είναι αποκλειστικά της R
- $\pi_{\mathbf{A}}(R \triangleright\triangleleft S) \equiv \pi_{\mathbf{A}}(R) \triangleright\triangleleft S$, με την $\boldsymbol{A}$' να περιέχει (α) τα πεδία που είχε η $\boldsymbol{A}$ (αρκεί να είναι αποκλειστικά της R) και (β) τα πεδία που εμπλέκονται στη συνθήκη σύνδεσης

Επανεγγραφή Ερωτήσεων (παράδειγμα από IBM DB2)

Distribute NOT

... WHERE NOT(COL1 = 10 OR COL2 > 3)

γίνεται

... WHERE COL1 <> 10 AND COL2 <= 3

Μετασχηματισμοί τιμών:

...WHERE COL = YEAR(`1994-09-08')

γίνεται

... WHERE COL = 1994

Μεταβατική κλειστότητα

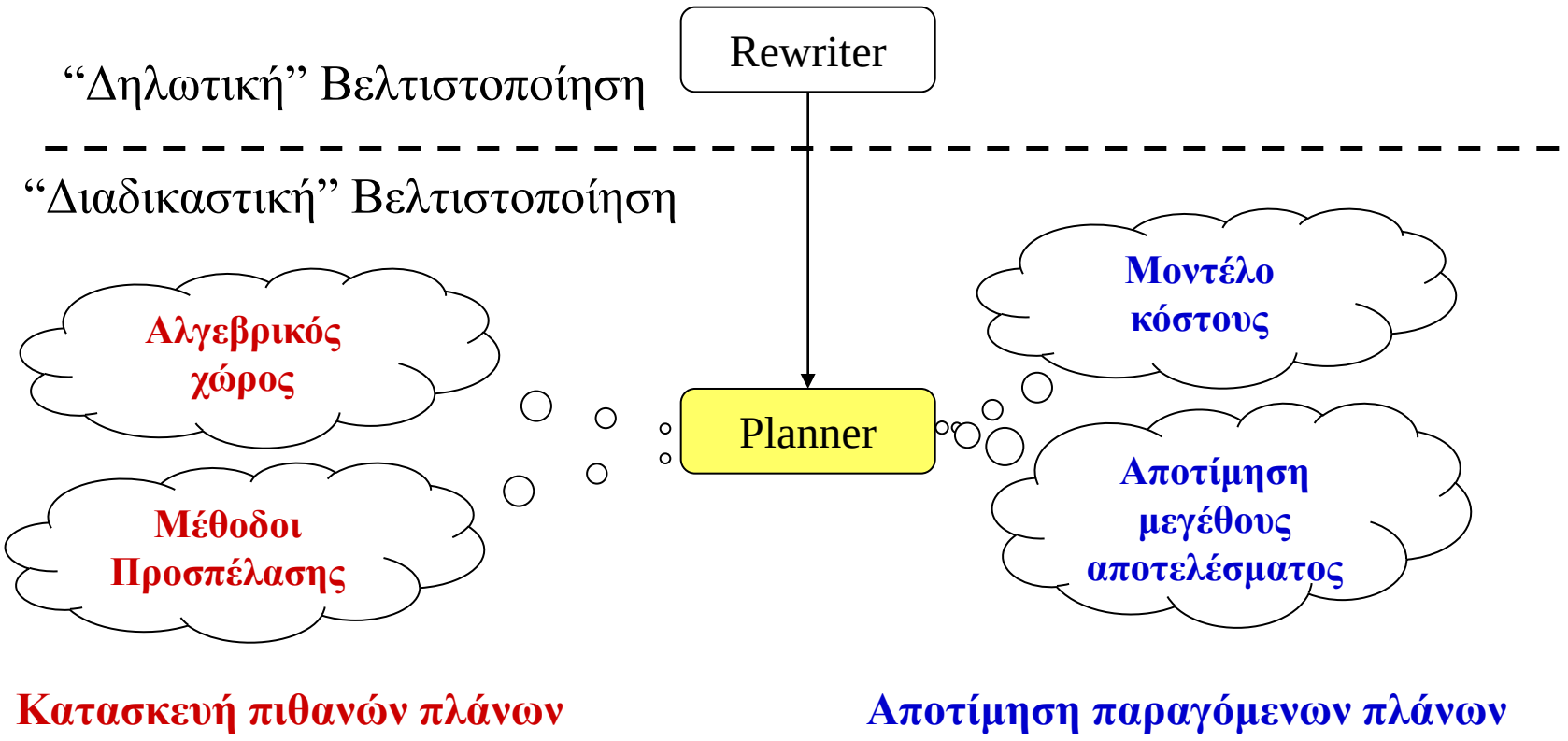
δοθέντος:

T1.C1 = T2.C2, T2.C2 = T3.C3, T1.C1 > 5

προστίθενται...

T1.C1 = T3.C3 AND T2.C2 > 5 AND T3.C3 > 5

Αφαιρετική δομή του βελτιστοποιητή



Επίπεδα βελτιστοποίησης

- Υπάρχει ένα επίπεδο «διαδικαστικής» βελτιστοποίησης, όπου παράγουμε (όλα τα) διαφορετικά πλάνα εκτέλεσης μέχρι να διαλέξουμε το πιο αποδοτικό. Η δουλειά αυτή ανατίθεται στον planner.
- Ο planner οφείλει:
  - Να αποφασίσει ποια πλάνα εκτέλεσης θα δημιουργηθούν
  - Ποιο εξ αυτών είναι το καλύτερο

Ο planner οφείλει …

- Να **κατασκευάσει ένα σύνολο πλάνων**, με βάση
  - Ένα αλγεβρικό χώρο για τη σειρά εκτέλεσης των λειτουργιών (π.χ., να αποφασίσει με ποια σειρά θα κάνει το R ⋈ S ⋈ T)
  - Ένα σύνολο από μεθόδους προσπέλασης στα δεδομένα (π.χ., full-index scan, full table scan, …)
- Να **αποτιμά κάθε πλάνο** που παράγει, μέχρι στο τέλος να βρει το πιο αποδοτικό, με βάση
  - Ένα μοντέλο κόστους που προβλέπει πόσο χρόνο/disk I/O/… κοστίζει το κάθε πλάνο
  - Ένα μοντέλο πρόβλεψης του μεγέθους, κυρίως των ενδιάμεσων αποτελεσμάτων
  - Προσοχή: η αποτίμηση είναι πάντα προσέγγιση/πρόβλεψη και όχι ακριβής υπολογισμός…

Σύνολο Κανόνων

1. $\sigma_{p \wedge q \wedge w}(X) = \sigma_p(\sigma_q(\sigma_w(X)))$
2. $\sigma_p(\sigma_q(X)) = \sigma_q(\sigma_p(X))$
3. $\Pi_A \Pi_B \Pi_C(X) = \Pi_A(X)$
4. $\Pi_{a_1,\ldots,a_N}(\sigma_p(X)) = \sigma_p(\Pi_{a_1,\ldots,a_N}(X))$
5. $X \times Y = Y \times X$ και $X \bowtie Y = Y \bowtie X$
6. $\sigma_p(X \times Y) = (\sigma_p(X)) \times Y$ και $\sigma_p(X \bowtie_q Y) = (\sigma_p(X)) \bowtie_q Y$ $\sigma_{p \wedge q}(X \times Y) = (\sigma_p(X) \times (\sigma_q(Y))$ και $\sigma_{p \wedge q}(X \bowtie_w Y) = (\sigma_p(X) \bowtie_w (\sigma_q(Y))$
7. $\Pi_{A \vee B}(X \bowtie_p Y) = (\Pi_A(X)) \bowtie_p (\Pi_B(Y)))$
8. $X \cup Y = Y \cup X$ και $X \cap Y = Y \cap X$
9. $\sigma_p(X \cup Y) = \sigma_p(X) \cup \sigma_p(Y)$ $\sigma_p(X \cap Y) = \sigma_p(X) \cap \sigma_p(Y)$ $\sigma_p(X - Y) = \sigma_p(X) - \sigma_p(Y)$
10. $\Pi_A(X \cup Y) = \Pi_A(X) \cup \Pi_A(Y)$
11. $(X \bowtie Y) \times Z = X \bowtie (Y \times Z)$ και $(X \bowtie Y) \bowtie Z = X \bowtie (Y \bowtie Z)$ $(X \bowtie_p Y) \bowtie_{q \wedge w} Z = X \bowtie_{p \wedge w} (Y \bowtie_q Z)$
12. $(X \cup Y) \cup Z = X \cup (Y \cup Z)$ $(X \cap Y) \cup Z = X \cap (Y \cap Z)$

Ευριστική Βελτιστοποίηση

Ευριστικός Κανόνας 1: *Οι πράξεις της επιλογής πρέπει να εκτελούνται όσο νωρίτερα γίνεται.*

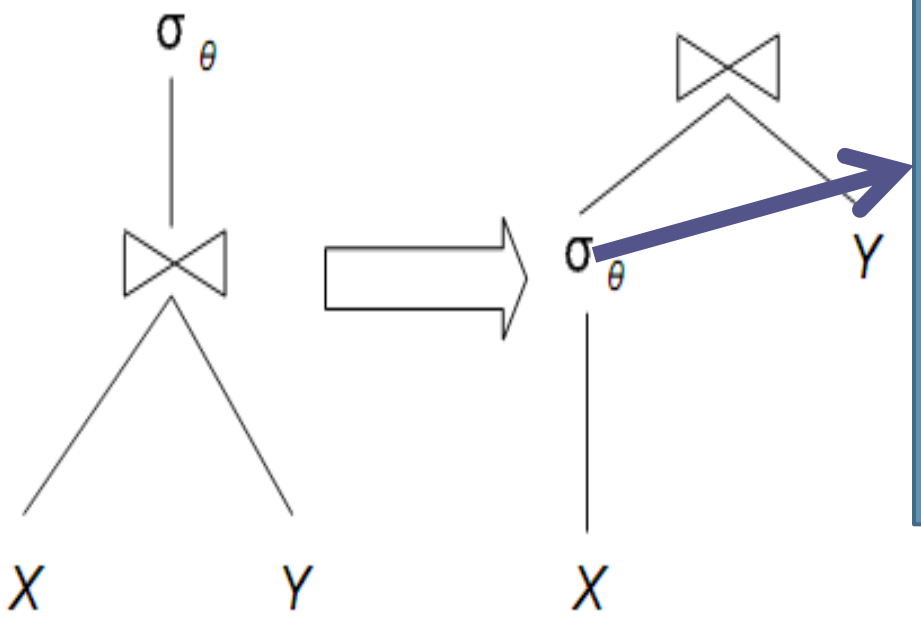
Ευριστικός Κανόνας 2: *Οι πράξεις της προβολής πρέπει να εκτελούνται όσο νωρίτερα γίνεται.*

Ευριστικός Κανόνας 3: *Η πράξη του καρτεσιανού γινομένου συνδυάζεται με τη συνθήκη της πράξης επιλογής, έτσι ώστε να δημιουργηθεί μία πράξη σύνδεσης.*

Ευριστικός Κανόνας 4: *Οι πράξεις επιλογής και σύνδεσης, οι οποίες παράγουν το μικρότερο αποτέλεσμα πρέπει να εκτελούνται πρώτες.*

Υλοποίηση πράξεων και εκτίμηση κόστους

- Οι ευριστικοί κανόνες δεν βρίσκουν πάντα την καλύτερη λύση.
- Π.χ:



Αν X<<Y και ο κατάλογος σύνδεσης δεν είναι στη συνθήκη θ τότε αναγκαστικά θα εξεταστούν μια-μια όλες οι γραμμές του Y
ΔΕΝ ΕΙΝΑΙ ΑΠΟΔΟΤΙΚΟΤΕΡΟ ΑΠΟ ΠΡΙΝ

Υλοποίηση πράξεων και εκτίμηση κόστους

- Το ΣΔΒΔ διατηρεί στατιστικά στοιχεία για τα αποθηκευμένα δεδομένα:

Σύμβολο	Περιγραφή
n_T	αριθμός γραμμών του πίνακα T
b_T	αριθμός σελίδων για την αποθήκευση του πίνακα T
bfr_T	αριθμός γραμμών του T που αποθηκεύονται σε μία σελίδα
s_T	μέγεθος του πίνακα T σε χαρακτήρες (bytes)
$d(T, a)$	αριθμός διακριτών τιμών της στήλης a του πίνακα T
$sc(T, a)$	μέσος αριθμός γραμμών που ικανοποιούν συνθήκη ισότητας της στήλης a
fan_I	ο μέσος αριθμός παιδιών (fanout) του δενδρικού καταλόγου i
$height_I$	το ύψος του δενδρικού καταλόγου i
l_I	ο αριθμός των φύλλων του καταλόγου i

Κόστος Επιλογής (Select (S))

- Γραμμική αναζήτηση (S1)
  - Όχι ταξινομημένος ή κλειδί εκτός συνθήκης:
    - $C_{S_1} = b_T$
  - Ταξινομημένος:
    - $C_{S_1} = b_T/2$ (μέσο κόστος)
- Δυαδική αναζήτηση (S2)
  - $C_{S_2} = \lceil log_2(b_T) \rceil + \lceil \frac{sc(T,a)}{bfr_T} \rceil - 1$

προσπελάσεων για την πρώτη γραμμή που ικανοποιεί τη συνθήκη

προσπελάσεων για όλες τις γραμμές. Είναι συνολικά sc(T,α) και αν θεωρήσουμε ότι η αποθήκευση ακολουθεί τη διάταξη της ταξινόμησης

Κόστος Επιλογής (Select (S))

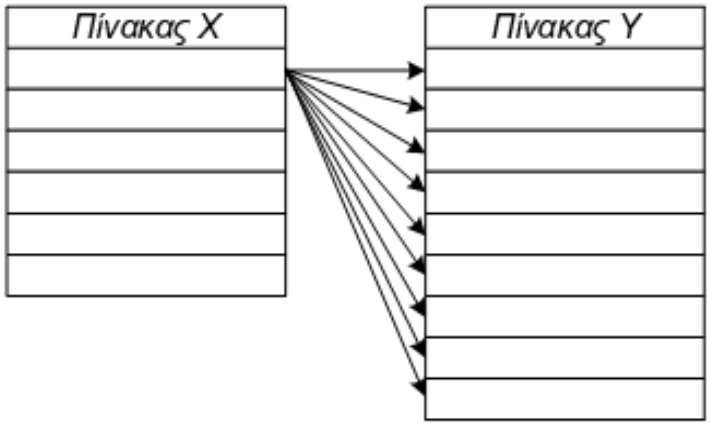
- Ισότητα στο πρωτεύον κλειδί με κατάλογο (S3)
  - $C_{S_3} = height_I + 1$
- Ισότητα σε στήλη που δεν είναι πρωτεύον κλειδί με κατάλογο (S4)
  - $C_{S_4} = height_I + \lceil \frac{sc(T,a)}{bfr_T} \rceil$
- Ανισότητα με πρωτεύον κατάλογο
  - $C_{S_5} = height_I + \frac{b_T}{2}$

Υπόθεση: οι μισές γραμμές του πίνακα θα ικανοποιούν την ανισότητα

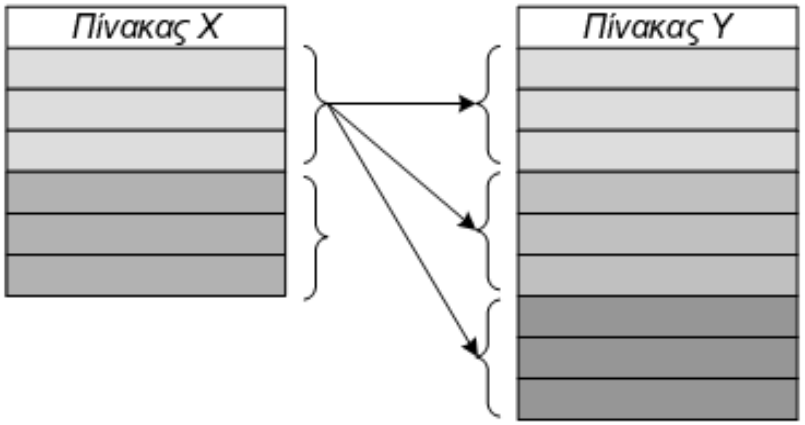
Κόστος Προβολής (Project- (P))

- Προβολή με ταξινόμηση (P1)
  - Οι γραμμές ταξινομούνται ως προς όλες τις στήλες, ώστε αν δυο ή περισσότερες γραμμές έχουν ίδιες τιμές να είναι τελικά γειτονικές. Αν μια γραμμή εμφανίζεται > 1 φορά οι επιπλέον εμφανίσεις διαγράφονται
- Προβολή με κατακερματισμό (P2)
  - Συνάρτηση hash που λαμβάνει υπόψη τις στήλες που ενδιαφέρουν. Κάθε φορά που επιλέγεται κάδος που αντιστοιχεί μια γραμμή, ελέγχεται αν υπάρχει όμοια γραμμή στο συγκεκριμένο κάδο. Αν ναι, δεν εισάγεται στον κάδο αλλιώς εισάγεται και συνεχίζεται μέχρι να εξαντληθούν όλες οι γραμμές

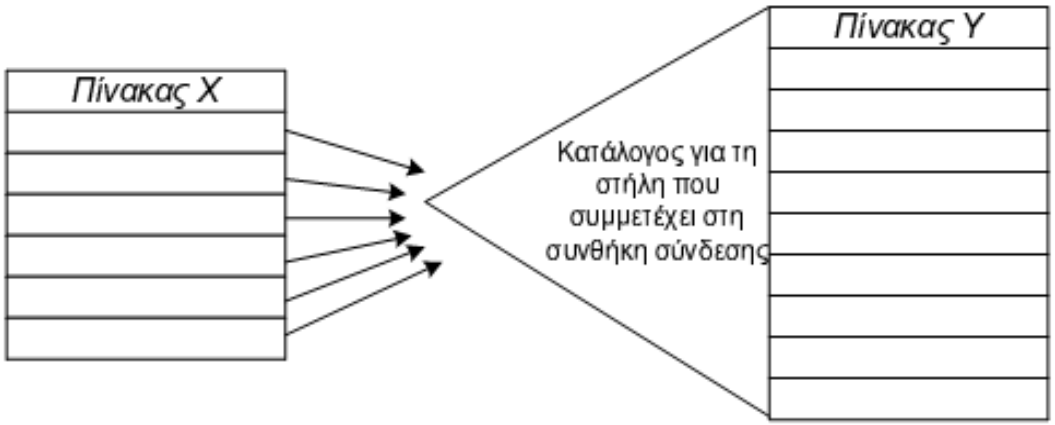
Εμφωλιασμένοι Βρόχοι



(α) εμφωλιασμένοι βρόγχοι (nested-loop)



(β) εμφωλιασμένοι βρόγχοι με σελίδες (block nested-loop)



(γ) εμφωλιασμένοι βρόγχοι με κατάλογο (indexed nested-loop)

Κόστος Συνένωσης (Join - (J))

- Εμφωλιασμένοι Βρόχοι (nested loops – J1)
  - $C_{J_1} = b_X + b_Y$ (αν X,Y μικροί)
  - $C_{J_1} = n_X \cdot b_Y + b_X$ (διαφορετικά)
- Εμφωλιασμένοι βρόχοι με σελίδες (J2)
  - $C_{J_2} = b_X \cdot b_Y + b_X$
- Εμφωλιασμένοι βρόχοι με κατάλογο (J3)
  - $C_{J_3} = b_X + n_X \cdot C$,όπου C το κόστος επιλογής με κατάλογο

Κόστος Συνένωσης (Join - (J))

- Μέθοδος ταξινόμησης-συγχώνευσης (Merge Join) (J4)
  - $C_{J_4} = b_X + b_Y$

a	x_1	x_2
9	2	4
5	1	2
6	2	4
1	1	2

a	y_1	y_2
9	11	12
1	3	4
7	7	8
6	5	6
8	9	10
1	1	2
9	13	14

(α)

a	x_1	x_2
1	1	2
5	1	2
6	2	4
9	2	4

a	y_1	y_2
1	1	2
1	3	4
6	5	6
7	7	8
8	9	10
9	11	12
9	13	14

(β)

a	x_1	x_2	y_1	y_2
1	1	2	1	2
1	1	2	3	4
6	2	4	5	6
9	2	4	11	12
9	2	4	13	14

(γ)

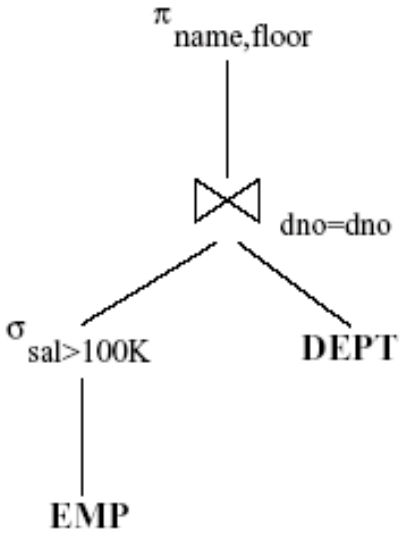
Υποθέσεις …

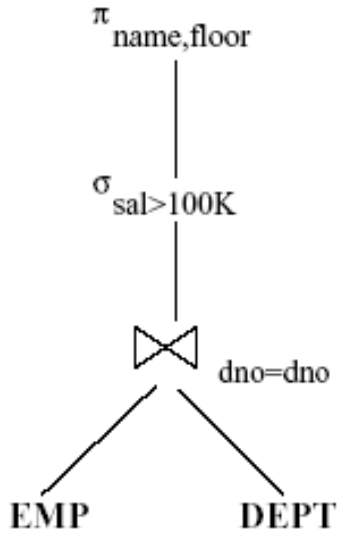
- Θα κάνουμε τις εξής υποθέσεις εργασίας (που αφορούν πρακτικά το σύνολο των DBMS) σε ότι αφορά τις εναλλακτικές λύσεις που θα εξετάσουμε:
- Οι μέθοδοι προσπέλασης που έχουμε είναι (α) πλήρες διάβασμα ενός πίνακα (β) προσπέλαση μέσω ενός B+ ευρετηρίου
- Οι μέθοδοι σύνδεσης που έχουμε είναι (α) nested loops και (β) merge-join, στα οποία χρησιμοποιούμε και των δύο ειδών τις μεθόδους προσπέλασης

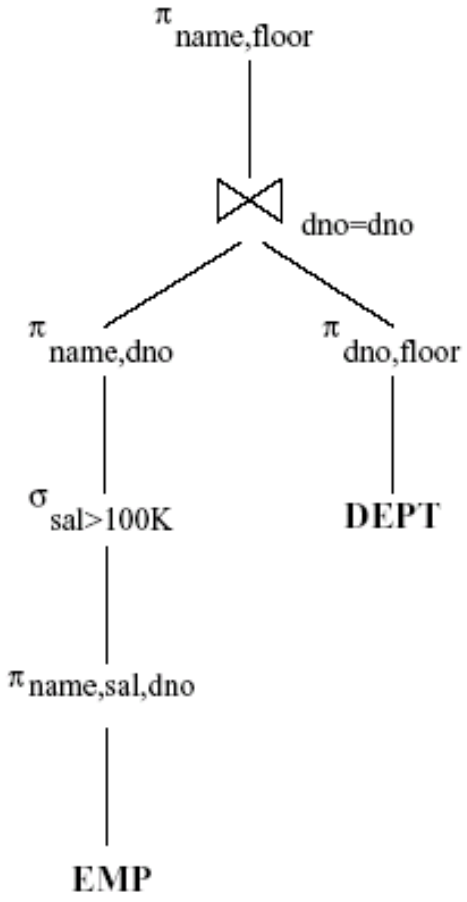
Αλγεβρικός χώρος: τι είναι ένα πλάνο

- Το πλάνο εκτέλεσης μιας SQL ερώτησης είναι ένα **δέντρο**, με:
  - Τις σχέσεις που συμμετέχουν στην ερώτηση, για φύλλα
  - Αλγεβρικούς τελεστές για ενδιάμεσους κόμβους και συγκεκριμένα τους π, σ και ⋈
- Το πλάνο έχει **σειρά εκτέλεσης από κάτω και αριστερά προς τα πάνω**.
- Κοιτώντας ένα ενδιάμεσο κόμβο, ξέρουμε ότι τα παιδιά του έχουν εκτελεστεί και αυτός στέλνει το αποτέλεσμα προς τα πάνω

Πλάνα εκτέλεσης








```
select name, floor
from emp, dept
where emp.dno = dept.dno
and sal > 100K
```

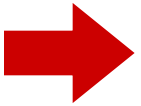
Πλάνα εκτέλεσης

- Για μια απλή SELECT-FROM-WHERE ερώτηση SQL, ο αριθμός των ισοδύναμων εναλλακτικών πλάνων είναι τεράστιος.
- Υπάρχουν κάποιοι λογικοί κανόνες, που επιτρέπουν στον βελτιστοποιητή να μειώσει τον αλγεβρικό χώρο πλάνων

Λογικοί κανόνες βελτιστοποίησης

- Σπρώξε όλες τις επιλογές όσο πιο χαμηλά στο δέντρο γίνεται
- Ενσωμάτωσε τις προβολές μέσα στους άλλους τελεστές

- *… και πάλι όμως, ο αλγεβρικός χώρος παραμένει τεράστιος …*

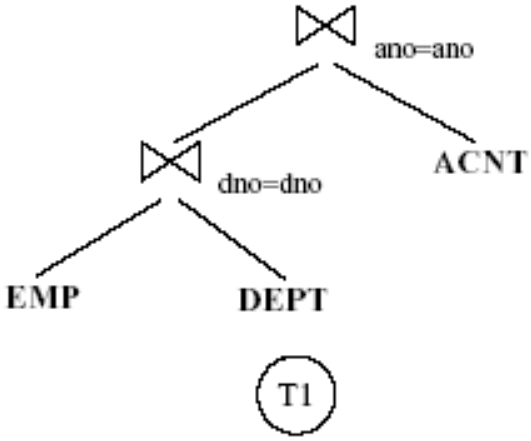


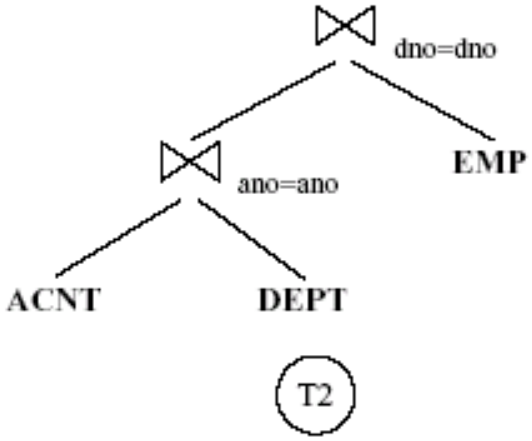
Λογικοί κανόνες βελτιστοποίησης

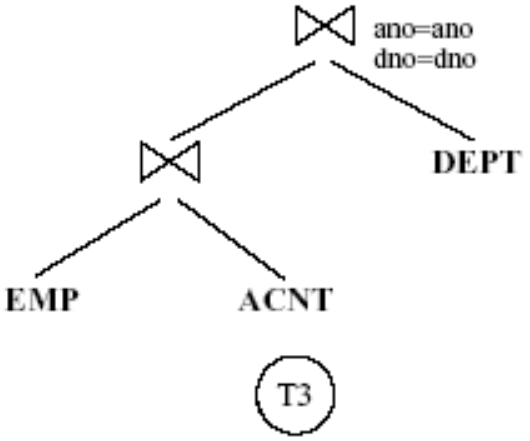
- Η βασική αιτία είναι οι ιδιότητες της σύνδεσης:
  - $R \bowtie S \Leftrightarrow S \bowtie R$
  - $(R \bowtie S) \bowtie T \Leftrightarrow R \bowtie (S \bowtie T)$
- Το αποτέλεσμα είναι ότι για **n** σχέσεις στο FROM clause έχω **n!** διατάξεις…

- Επιπλέον κανόνας:
  - Ποτέ μην κάνεις καρτεσιανά γινόμενα, εκτός κι αν πρέπει…

Ποτέ μην κάνεις καρτεσιανά γινόμενα



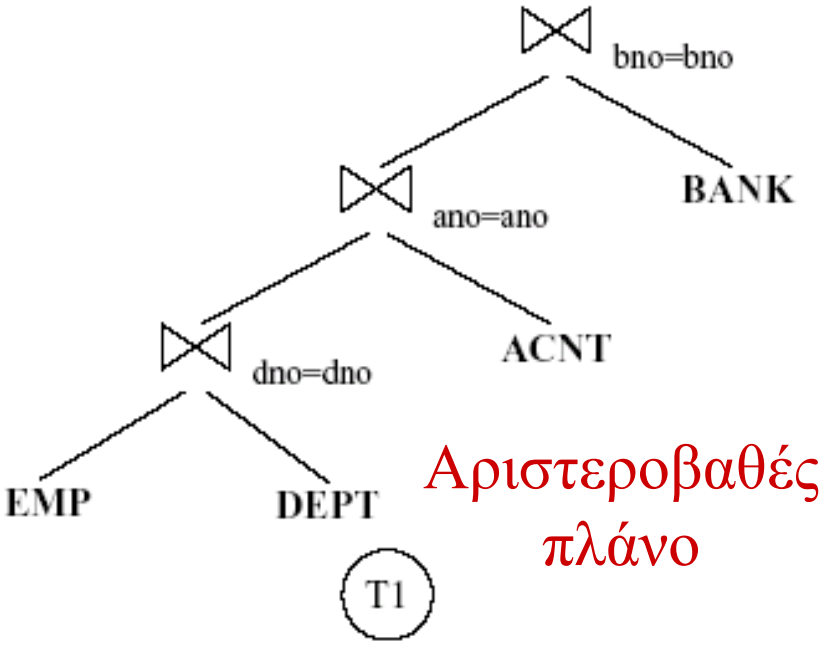




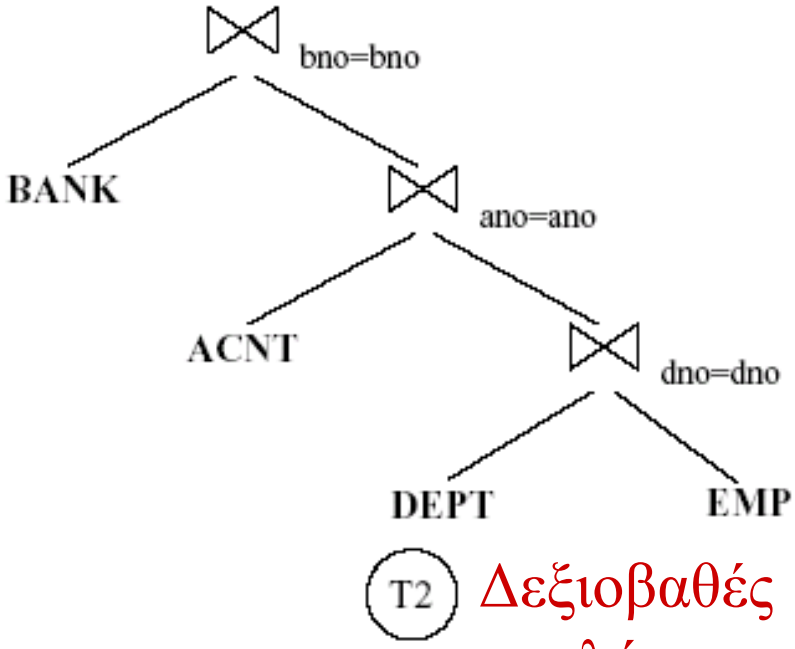

```
select name, floor, balance
from emp, dept, acnt
where emp.dno = dept.dno
and dept.ano = acnt.ano
```

Αριστεροβαθή Δέντρα

- Ακόμα και τώρα όμως, ο αλγεβρικός χώρος είναι μεγάλος
- Όλα τα σύγχρονα DBMS έχουν εισάγει τον ακόλουθο πρακτικό κανόνα:
- Η εσωτερική σχέση ενός τελεστή είναι ΠΑΝΤΑ μια σχέση της ΒΔ και ποτέ ενδιάμεσο αποτέλεσμα!
- Τα δέντρα που προκύπτουν έτσι, λέγονται **αριστεροβαθή** (left-deep)

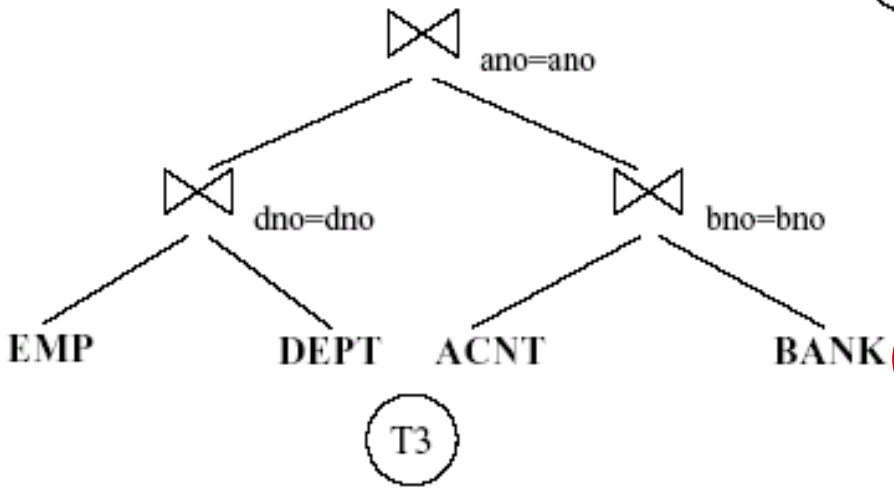


Αριστεροβαθές πλάνο



Δεξιοβαθές πλάνο

```
select name, floor,
balance, bank
from emp, dept, acnt
where
emp.dno = dept.dno
and
dept.ano = acnt.ano
and acnt.bno =
bank.bno
```



Θαμνώδες (bushy) πλάνο

Αριστεροβαθή Δέντρα

- Κέρδη από αριστεροβαθή δέντρα:
  - Μπορούμε εύκολα να χρησιμοποιούμε ευρετήρια για τις σχέσεις!
  - Τα αποτελέσματα από μια σύνδεση μπορούν να γίνουν pipeline σε μια επόμενη σύνδεση!

Ωραία, και με ποια σειρά?

- Ακόμα δεν μας είπες για τη σειρά των συνδέσεων!

$$(R \bowtie S) \bowtie T \Leftrightarrow R \bowtie (S \bowtie T)$$

- Ο planner, σε όλα τα εμπορικά DBMS χρησιμοποιεί ένα αλγόριθμο δυναμικού προγραμματισμού για να ανακαλύψει τη σειρά

- Προτού δώσουμε το γενικό τρόπο δημιουργίας των πλάνων, **θα κατηγοριοποιήσουμε τις ερωτήσεις** ως:
  - Ερωτήσεις που αφορούν **μία σχέση** στο FROM clause
  - Ερωτήσεις που αφορούν **πολλές σχέσεις** στο FROM clause

Ερωτήσεις με μία σχέση στο FROM clause

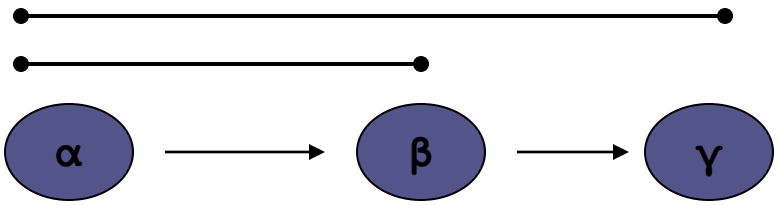
- Οι ερωτήσεις αφορούν ένα συνδυασμό προβολών, επιλογών και συναθροίσεων. Η επιλογή του πλάνου γίνεται ως ακολούθως:
  - Εξετάζεται κάθε διαθέσιμη μέθοδος προσπέλασης (file scan / index) και επιλέγεται αυτή με το ελάχιστο κόστος
  - Οι τελεστές εκτελούνται, όσο το δυνατόν γίνεται, μαζί (π.χ., οι προβολές και οι επιλογές ενσωματώνονται στην προσπέλαση μέσω ευρετηρίου).

Αριστεροβαθή πλάνα για ερωτήσεις πολλών σχέσεων

- Τα αριστεροβαθή πλάνα διαφέρουν
  - στη σειρά των σχέσεων,
  - στη μέθοδο προσπέλασης για κάθε σχέση (index/file scan), και
  - στον τρόπο εκτέλεσης κάθε σύνδεσης (NLJ,SMJ,HJ)
- Οι πράξεις ORDER BY, GROUP BY, κλπ., εξετάζονται ως μια τελική πράξη που επικάθεται ενός πλάνου, ενδεχομένως ταξινομώντας το αποτέλεσμα των συνδέσεων αν αυτό δεν είναι ήδη βολικά ταξινομημένο.
- Και πάλι, όμως, ο αριθμός των υπό εξέταση πλάνων είναι εκθετικός σε σχέση με τον αριθμό των εμπλεκόμενων σχέσεων

Δυναμικός προγραμματισμός

- Εφαρμόζεται σε προβλήματα, στα οποία η λύση μπορεί να εκφρασθεί ως μια ακολουθία αποφάσεων
- Εκμεταλλεύεται το principle of optimality: *μια ακολουθία αποφάσεων (λύση) δεν μπορεί να είναι βέλτιστη, αν μια υπακολουθία της δεν είναι βέλτιστη*



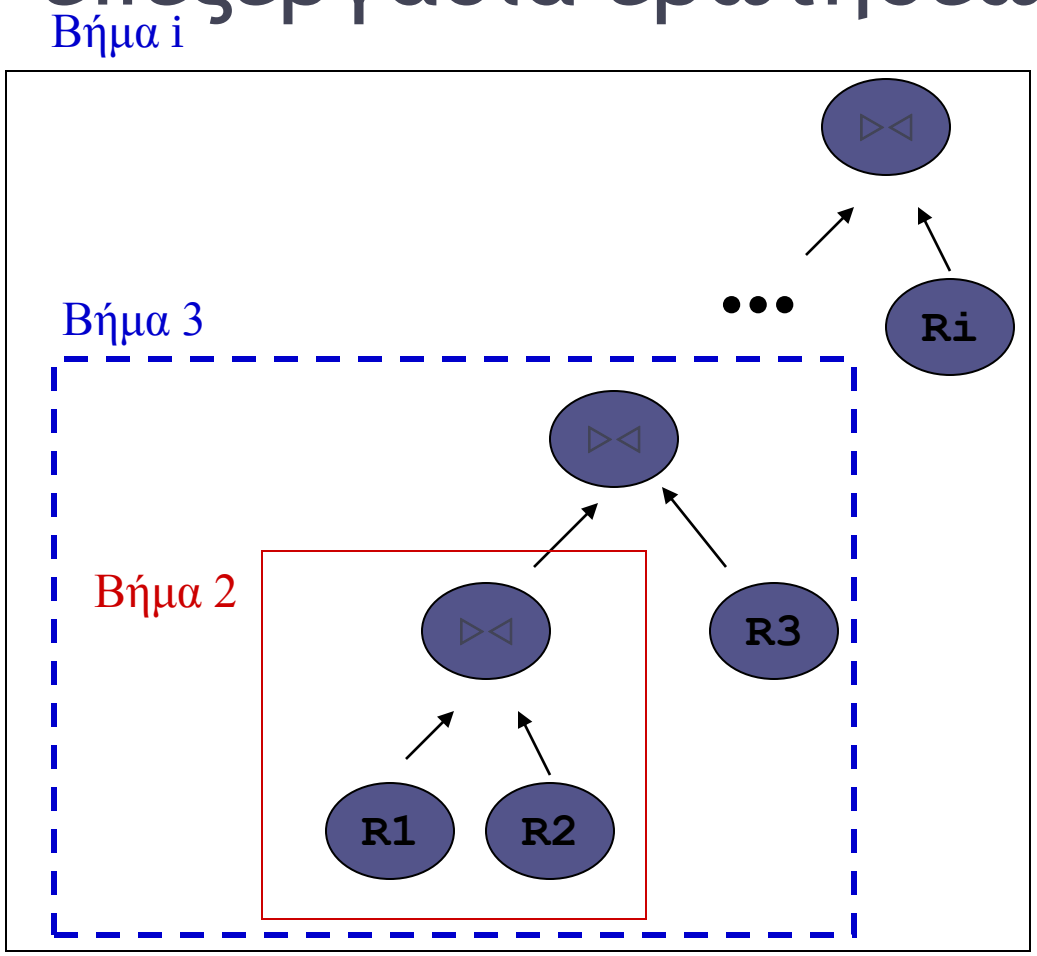
Δυναμικός προγραμματισμός για επεξεργασία ερωτήσεων

- Πρόβλημα: ποια η σωστή σειρά για να εκτελέσω το R ⋈ S ⋈ T?
- Δυναμικός Προγραμματισμός:
  1. Θα βρω όλους τους τρόπους για να προσπελάσω κάθε σχέση χωριστά
  2. Θα πάρω κάθε τέτοιο τρόπο προσπέλασης και θα φτιάξω το καλύτερο υποδέντρο με δύο φύλλα που του αντιστοιχεί
  3. Θα πάρω κάθε τέτοιο υποδέντρο και θα φτιάξω το καλύτερο υποδέντρο με τρία φύλλα που του αντιστοιχεί

Αριστεροβαθή πλάνα για ερωτήσεις πολλών σχέσεων

- Πρόβλημα: ποια η σωστή σειρά για να εκτελέσω το $R_1 \triangleright\triangleleft R_2 \triangleright\triangleleft \ldots R_N$?
- Κάνε Ν περάσματα, (αν συνδέουμε Ν σχέσεις):
  - Pass 0: Ομαδοποίησε τις μεθόδους προσπέλασης κάθε σχέσης σε σχέση με την ταξινόμηση των εγγραφών. Κάθε τέτοια ομάδα ονομάζεται «**ενδιαφέρουσα σειρά**» (**interesting order**)
  - Pass 1: Βρες το καλύτερο 1-relation πλάνο για κάθε ενδιαφέρουσα ομάδα μιας σχέσης
  - Pass 2: Βρες τον καλύτερο τρόπο σύνδεσης κάθε 1-relation plan (ως outer) με μια άλλη σχέση και φτιάξε όλα τα 2-relation plans
  - Pass N: Βρες τον καλύτερο τρόπο σύνδεσης κάθε (N-1)-relation plan (ως outer) με τη N-στη σχέση που του απομένει.

Δυναμικός προγραμματισμός για επεξεργασία ερωτήσεων



Εν παραλλήλω, φτιάχνω πολλά δέντρα.

Σιγά σιγά όμως, μειώνω τον αριθμό τους...

Δυναμικός προγραμματισμός για επεξεργασία ερωτήσεων

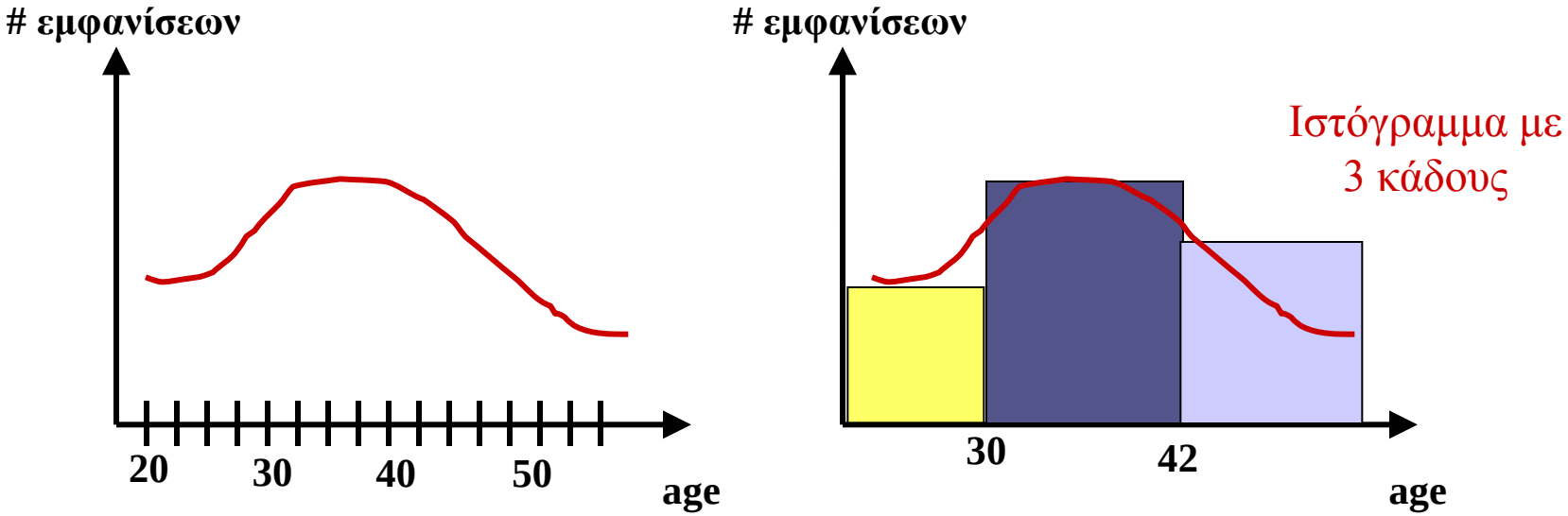
- Θυμηθείτε ότι έχω πολλούς τρόπους για να προσπελάζω ένα πίνακα (είτε με full table scan, είτε μέσω κάποιου ευρετηρίου – πιθανώς να έχω πολλά για ένα πίνακα)
- Επίσης ο αλγόριθμος χρησιμοποιεί ένα έξυπνο τρόπο να εκμεταλλεύεται το γεγονός ότι κάποιοι τρόποι σύνδεσης παράγουν τα αποτελέσματα ταξινομημένα σε σχέση με κάποιο πεδίο….
- ***Για την ακρίβεια, δεν παράγουμε δέντρα για κάθε συνδυασμό, αλλά για κάθε δυνατό τρόπο ταξινόμησης των ενδιάμεσων αποτελεσμάτων σε σχέση με τα πεδία που παίρνουν μέρος στη σύνδεση***

Εκτίμηση μεγέθους

- Για να δουλέψουν οι φόρμουλες κόστους που έχουμε, πρέπει να μπορούμε να αποτιμήσουμε το μέγεθος των ενδιάμεσων αποτελεσμάτων
- Εν γένει, δεν είμαστε πολύ καλοί σ' αυτό, πρακτικά οι τρόποι εκτίμησης που έχουμε δουλεύουν σε δέντρα με ύψος πάνω από 5...
- Η πιο καλή τεχνική που έχουμε είναι τα **ιστογράμματα**

Ιστογράμματα

- Σ' ένα ιστόγραμμα, διαιρούμε το εύρος των τιμών ενός πεδίου σε κάδους (αγγλιστί: buckets)
- Για κάθε τιμή που παίρνει το πεδίο, μετράω τον αριθμό που αυτή εμφανίζεται



Ιστογράμματα

- Και πώς αποφασίζω **πόσους** κάδους?
  - Ιστογράμματα ίσου πλάτους: κάθε κάδος έχει τον ίδιο αριθμό τιμών στον άξονα των x
  - Ιστογράμματα ίσου ύψος: κάθε κάδος έχει το ίδιο ύψος στον άξονα των y
  - Σειριακά ιστογράμματα: οι συχνότητες ενός κάδου είναι μεγαλύτερες από αυτές του προηγούμενου

Ιστογράμματα

- Αν κάνω μια επιλογή $\sigma_{age > 43}(emp)$ το DBMS μπορεί να εκτιμήσει περίπου πόσες εγγραφές θα μου επιστραφούν
- Αντίστοιχα, αν κάνω μια σύνδεση $R \triangleright\!\triangleleft S$ πάλι μπορεί να κάνει την αντίστοιχη εκτίμηση ανά ζεύγος κάδων.

Ιστογράμματα

- Είναι σαφές ότι όσο πιο πολλές πράξεις, τόσο πιο πολύ απομακρύνεται η εκτίμηση από την πραγματικότητα…
- (Λανθασμένες) Υποθέσεις εργασίας: οι τιμές των πεδίων είναι ισοπίθανα μοιρασμένες + τα πεδία είναι ανεξάρτητα μεταξύ τους…

- Και τι γίνεται όταν έχω INS/DEL/UPD ?

Ιστογράμματα

Name	Salary	Department
Zeus	100K	General Manager
Poseidon	80K	Defense
Pluto	80K	Justice
Aris	50K	Defense
Ermis	60K	Commerce
Apollo	60K	Energy
Hefestus	50K	Energy
Hera	90K	General Manager
Athena	70K	Education
Aphro	60K	Domestic Affairs
Demeter	60K	Agriculture
Hestia	50K	Domestic Affairs
Artemis	60K	Energy

Department	Frequency
General Manager	2
Defense	2
Education	1
Domestic Affairs	2
Agriculture	1
Commerce	1
Justice	1
Energy	3

Ιστογράμματα

Department	Απόλυτη Συχνότητα	Συχνότητα Κάδου
Agriculture	1	1.50
Commerce	1	1.50
Defense	2	1.50
Domestic Affairs	2	1.50
Education	1	1.75
Energy	3	1.75
General Manager	2	1.75
Justice	1	1.75

#πραγματικών εμφανίσεων

εκτίμηση ιστογράμματος

- **Ιστόγραμμα ίσου πλάτους**: κάθε κάδος έχει τον ίδιο αριθμό τιμών στον άξονα των x
- Εδώ: **δύο κάδοι**, ο πρώτος από A – D και ο άλλος από E-Z
- Συχν. Κάδου: Σ(x)/count(x), x∈κάδο

Ιστογράμματα

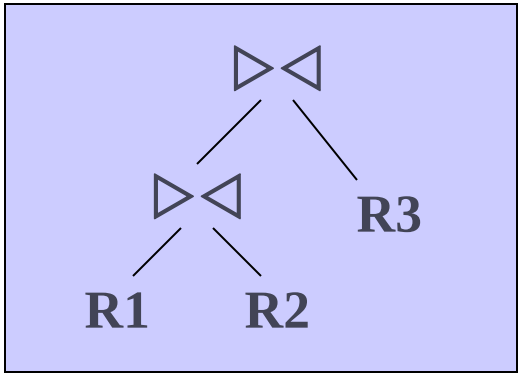
Department	Απόλυτη Συχνότητα	Συχνότητα Κάδου
Agriculture	1	1.33
Commerce	1	1.33
Defense	2	1.33
Education	1	1.33
General Manager	2	1.33
Justice	1	1.33
Domestic Affairs	2	2.50
Energy	3	2.50

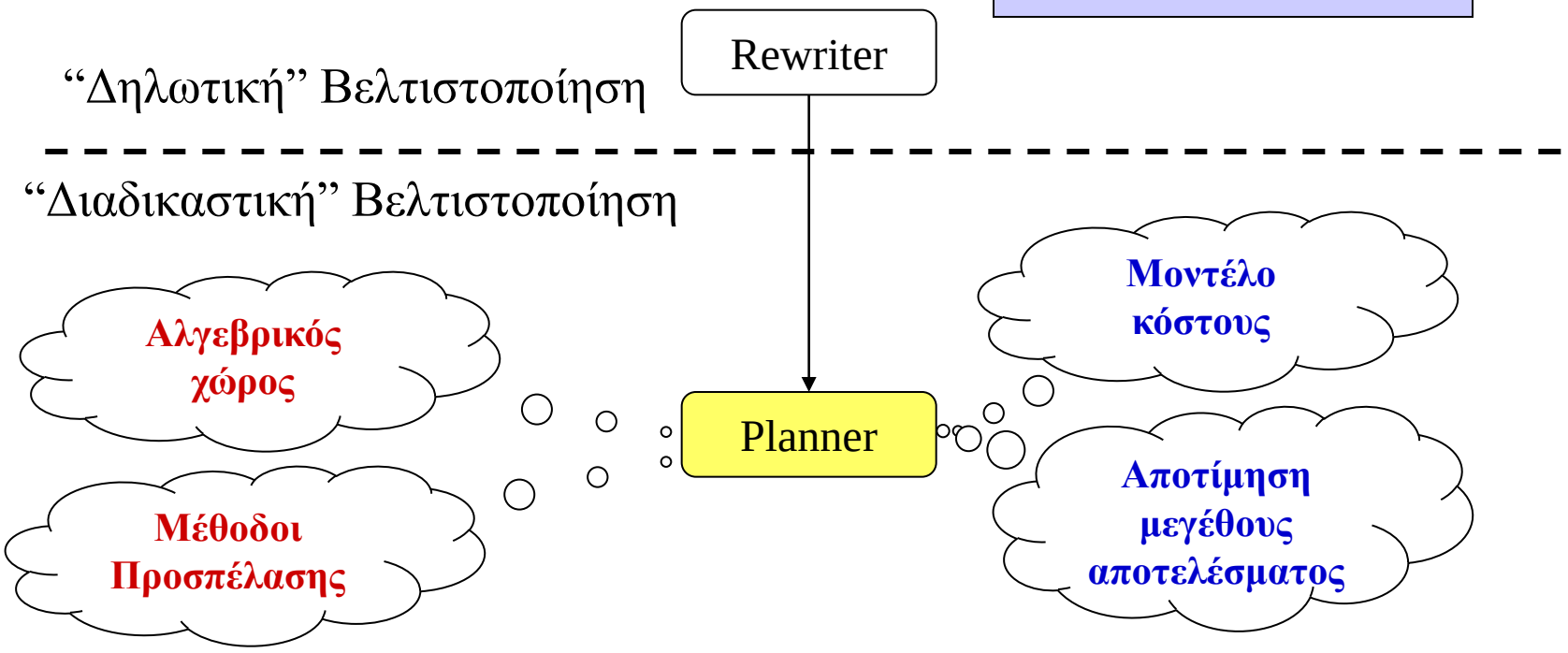
#πραγματικών εμφανίσεων

εκτίμηση ιστογράμματος

- **Ιστόγραμμα σειριακό**: οι συχνότητες του δεύτερου κάδου είναι μεγαλύτερες από αυτές του πρώτου

Επανάληψη


R3
R1
R2


Rewriter
“Δηλωτική” Βελτιστοποίηση
“Διαδικαστική” Βελτιστοποίηση
Αλγεβρικός χώρος
Μέθοδοι Προσπέλασης
Planner
Μοντέλο κόστους
Αποτίμηση μεγέθους αποτελέσματος
Κατασκευή πιθανών πλάνων
Αποτίμηση παραγόμενων πλάνων

Παράρτημα

- Μετασχηματισμοί ερωτήσεων

Επανεγγραφή Ερωτήσεων (παράδειγμα από IBM DB2)

- **Αρχική Ερώτηση**

```
SELECT ps.*
FROM tpcd.partsupp ps
WHERE ps.ps_partkey IN
    (SELECT p_partkey
    FROM tpcd.parts
    WHERE p_name LIKE 'forest%');
```

- **Μετασχηματισμένη Ερώτηση**

```
SELECT ps.*
FROM parts, partsupp ps
WHERE ps.ps_partkey = p_partkey AND
p_name LIKE `forest%';
```

Επανεγγραφή Ερωτήσεων (παράδειγμα από IBM DB2)

- **Αρχική Ερώτηση**

```
SELECT SUM(O_TOTAL_PRICE) AS OSUM,
    AVG(O_TOTAL_PRICE) AS OAVG
FROM ORDERS;
```

- **Μετασχηματισμένη Ερώτηση**

```
SELECT OSUM, OSUM/OCOUNT AS OAVG
FROM (SELECT SUM(O_TOTAL_PRICE) AS OSUM,
          COUNT(O_TOTAL_PRICE) AS OCOUNT
          FROM ORDERS) AS SHARED_AGG;
```

- Από 2 sum και 1 count σε 1 sum και 1 count!

Επανεγγραφή Ερωτήσεων (παράδειγμα από IBM DB2)

- **Αρχική Ερώτηση**

```
SELECT PS_SUPPLYCOST FROM PARTSUPP
WHERE PS_PARTKEY <> ALL
    (SELECT L_PARTKEY FROM LINEITEM
    WHERE PS_SUPPKEY = L_SUPPKEY)
```

- **Μετασχηματισμένη Ερώτηση**

```
SELECT PS_SUPPLYCOST FROM PARTSUPP
WHERE NOT EXISTS
    (SELECT * FROM LINEITEM
    WHERE PS_SUPPKEY = L_SUPPKEY
    AND PS_PARTKEY = L_PARTKEY)
```