***Συστήματα Διαχείρισης Βάσεων Δεδομένων***

***Μυρτώ-Χριστίνα Ελευθέρου, 3170046***

***Δεύτερη Σειρά Ασκήσεων***

# Άσκηση 1

## 1.

a. NLJ ➔Κόστος = 33.200 I/Os

Εξήγηση:
Χρησιμοποιούμε ως εξωτερική σχέση εκείνη με τις περισσότερες σελίδες.
Επομένως ως εξωτερική σχέση έχουμε την R.
B(R) = 24000/20 = 1200 σελίδες
B(S) = 40000/50 = 800 σελίδες

B(R)+ (B(R)/(M-1))*B(S) = 1200 + (1200/31-1) *800 = 33.200 I/Os

b. SMJ ➔ Κόστος = 12.400 I/Os

Εξήγηση:

B(R)+B(S) < $M^2$ ➔ 2000 < $31^2$ ➔ 2000 < 961 δεν ισχύει. Το μέγεθος της μνήμης είναι μικρό για να εφαρμοστεί η αποδοτική μορφή του αλγορίθμου.

Επομένως χρησιμοποιώ την απλή έκδοση.

Κάνω 3 phase sorting για την R με Κόστος = 6B(R)
Κάνω 2 phase sorting για την S με Κόστος = 4B(S)
Επίσης από το merge έχω επιπλέον κόστος B(R)+B(S)

Συνολικό Κόστος = 6B(R)+4B(S)+B(R)+B(S) = 7B(R)+5B(S) =
= 7*1200 + 5*800 = 12.400 I/Os

c. HJ ➔ Κόστος = 6.000 I/Os

Εξήγηση:
Το μέγεθος της μνήμης είναι μικρό για να εφαρμοστεί η βέλτιστη μορφή του αλγορίθμου, αφού καμία από τις δύο σχέσεις δεν χωράει ολόκληρη στην μνήμη.
M < min(B(R),B(S))

Ωστόσο το μέγεθος της μνήμης είναι αρκετό για να εφαρμόσουμε hash join στη μνήμη.
M > sqrt(min (B(R),B(S))) ➔ 31 > 28

Οπότε θα χρησιμοποιήσω την κανονική έκδοση του αλγορίθμου και θα έχουμε κόστος:
3(B(R) + B(S)) = 3*(2.000) = 6.000 I/Os

**2.** Για M=10
B(R)+B(S) < $M^2$ →2000<100 προφανώς δεν ισχύει οπότε η αποδοτική μέθοδος δεν εφαρμόζεται.

Κάνω 3-phase sorting για την S

1η φάση

Το κόστος σε αυτή τη φάση είναι 2B(S) εφόσον διαβάζονται οι σελίδες από το δίσκο, ταξινομούνται σε sublists στη μνήμη και έπειτα γράφεται το αποτέλεσμα στον δίσκο. Έχω 800 blocks επομένως οι λίστες που θα δημιουργηθούν είναι B(S)/M = 800/10 = 80.

2η φάση

Στη δεύτερη φάση έχουν δημιουργηθεί 80 λίστες οι οποίες αποτελούνται από 10 blocks η καθεμία. Στη συνέχεια διαβάζονται ανά 10 από τον δίσκο, ταξινομούνται στην μνήμη και γράφονται ξανά στον δίσκο δημιουργώντας έτσι 8 sublists των 100 block το κάθε ένα. [2B(S)]

3η φάση

Έπειτα τα 8 sublists διαβάζονται από τον δίσκο, ταξινομούνται στην μνήμη και γράφονται ξανά στον δίσκο δημιουργώντας μία λίστα των 800 blocks. [2B(S)]

**Κόστος** = 2B(S) + 2B(S) + 2B(S) = **6B(S)**

Κάνω 4 phase sorting για την R

1η φάση

Το κόστος σε αυτή τη φάση είναι 2B(R) εφόσον διαβάζονται οι σελίδες από το δίσκο, ταξινομούνται σε sublists στη μνήμη και έπειτα γράφεται το αποτέλεσμα στον δίσκο. Έχω 1200 blocks επομένως οι λίστες που θα δημιουργηθούν είναι B(R)/M = 1200/10 = 120.

2η φάση

Στη δεύτερη φάση έχουν δημιουργηθεί 120 λίστες οι οποίες αποτελούνται από 10 blocks η καθεμία. Διαβάζονται ανά 10 από τον δίσκο, ταξινομούνται στην μνήμη και γράφονται ξανά στον δίσκο δημιουργώντας έτσι 12 sublists των 100 block το κάθε ένα. [2B(R)]

3η φάση

Τα 12 sublists είναι περισσότερα από M=10 επομένως δεν χωράνε κατευθείαν στην μνήμη. Για αυτό διαβάζονται από τον δίσκο τα πρώτα 10, ταξινομούνται στην μνήμη και γράφονται ξανά στον δίσκο. Το ίδιο συμβαίνει για τα επόμενα 2 sublists που απομένουν. Έτσι δημιουργούνται δύο λίστες, η μία των 1000 block και η άλλη των 200 block. [2B(R)]

4η φάση

Τέλος οι 2 λίστες που έχουν πλέον γραφτεί στον δίσκο, διαβάζονται από τον δίσκο, ταξινομούνται στην μνήμη και γράφονται ξανά στον δίσκο δημιουργώντας το τελικό list των 1200 block. [2B(R)]

**Κόστος** = 2B(R) + 2B(R) + 2B(R) + 2B(R) = **8B(R)**

Σύνολο

Τέλος γίνεται ακόμα ένα B(R) και ένα B(S) για το διάβασμα των δύο λιστών ώστε να γίνει η συγχώνευση τους.

Συνολικά το κόστος είναι:

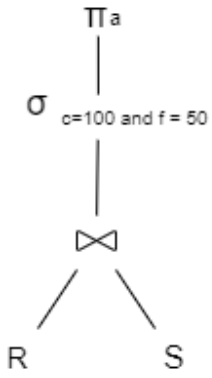
6B(S) + 8B(R) + B(S) + B(R) =

= 7(BS) + 9B(R) =

= 7*800 + 9*1200 = 16400 I/Os.

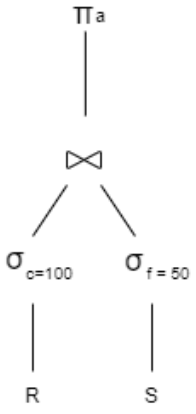
## Άσκηση 2

1. Το λογικό πλάνο στην αρχική του μορφή είναι:




---

Το λογικό πλάνο στην τελική του μορφή είναι:



**2.** Εφόσον τα δεδομένα είναι ομοιόμορφα:

Έστω **X = $\sigma_{c=100}$(R)** και **Y = $\sigma_{f=50}$(S)**

T(X) = T(R)/V(R,c) = 30.000/50 = 600 εγγραφές.
Cost(X) = min(TableScan(R),IdexScan(R,c)) = min(1500,600) = 600 I/Os
B(X) = 600/(30000/1500) = 30 σελίδες

T(Y) = T(S)/V(S,f) = 50.000/100 = 500 εγγραφές.
Cost(Y) = min(TableScan(S),IdexScan(S,f)) = min(2000,500/25-20) = 20 I/Os
B(Y) = 500/(50000/2000)= 20 σελίδες

**NLJ**

1ος τρόπος (εσωτερική X, εξωτερική Y)

Cost(NLJ(X,Y)) = Cost(X) + ceiling[B(X)/(M-1)]*Cost(Y) =
= 600 + ceiling(30/9) * 20 = 600 + 4*20 = **680 I/Os**

2ος τρόπος (εσωτερική Y, εξωτερική X)

Cost(NLJ(Y,X)) = Cost(Y) + ceiling[B(Y)/(M-1)]*Cost(X) =
= 20 + ceiling(20/9) * 600 = 20+ 3*600 = **1820 I/Os**

Άρα κρατάμε την καλύτερη περίπτωση, άρα κόστος NLJ = 680 I/Os.

**SMJ**

B(X) = 30

B(Y) = 20

B(X) + B(Y) = 50 < $M^2$ => 50 < 100

Άρα μπορούμε να εφαρμόσουμε την αποδοτική έκδοση του SMJ.

Cost (SMJ(X,Y)) = Cost(X) + Cost(Y) + 2B(X) + 2B(Y) =
600+20+2*30+2*20 = **720 I/Os**

Άρα το ελάχιστο κόστος του παραπάνω φυσικού πλάνου είναι 680 I/Os με χρήση του NLJ.

## Άσκηση 4

Έστω ΠΡ=ΠΡΟΪΟΝΤΑ, Κ=ΚΑΤΑΣΤΗΜΑΤΑ, ΠΩΛ=ΠΩΛΗΣΕΙΣ.

- Τ(ΠΡ) = 50.000, Β(ΠΡ) = 5.000
- Τ(Κ) = 1000, Β(Κ) = 200, V(Κ, Πόλη) = 100
- Τ(ΠΩΛ) = 500.000, Β(ΠΩΛ) = 20.000.
- Μ = 50
- Μία σελίδα χωράει 10 εγγραφές της σχέσης ΠΡ
- Μία σελίδα χωράει 5 εγγραφές της σχέσης Κ
- Μία σελίδα χωράει 25 εγγραφές της σχέσης ΠΩΛ

**Λειτουργία 1:**

Κόστος = 2 I/Os και Αριθμός εγγραφών στην έξοδο = 10

Επειδή:

$\sigma_{\text{πόλη='Αθήνα'}}$ = Τ(Κ)/100 = 1000/100 = 10 εγγραφές

Εφόσον έχουμε ευρετήριο στο Κ.Πόλη, οι τιμές της Πόλης βρίσκονται ήδη στην μνήμη. Επειδή όμως χρειαζόμαστε και τον Κωδικό, πρέπει να ανακτηθούν οι 10 εγγραφές από τη μνήμη.

cost($\sigma_{\text{πόλη='Αθήνα'}}$) = 2 I/Os. (5 εγγραφές/block)

**Λειτουργία 2:**

Κόστος = 10 I/Os και Αριθμός εγγραφών στην έξοδο = 100.

Επειδή:

Εφόσον σε κάθε κατάστημα γίνονται καθημερινά 10 πωλήσεις τότε για κάθε εγγραφή (κατάστημα) που θα πάρει ως είσοδο ο INLJ θα επιστρέψει 10 εγγραφές πωλήσεων.

Οι εγγραφές στην έξοδο είναι 10*10 = 100.

Για κάθε μία από τις 10 εγγραφές που δέχεται ως είσοδο ο INLJ χρησιμοποιεί το ευρετήριο που υπάρχει στα γνωρίσματα (Κωδικός, Ημερομηνία) τη σχέσης ΠΩΛ για να ανακτήσει τις αντίστοιχες εγγραφές από την σχέση ΠΩΛ.

Οι τιμές των γνωρισμάτων Κωδικός, Ημερομηνία υπάρχουν στην μνήμη διότι βρίσκονται στα φύλλα του ευρετηρίου. Ωστόσο, χρειαζόμαστε και το γνώρισμα Barcode οπότε οι 100 αυτές εγγραφές πρέπει να ανακτηθούν από τη μνήμη. Το ευρετήριο στα γνωρίσματα (Κωδικός, Ημερομηνία) τη σχέσης ΠΩΛ είναι clustered index, και δεδομένου ότι 10 εγγραφές πωλήσεων χωράνε σε μία σελίδα , το κόστος είναι 100/10 = 10 I/Os.

**Λειτουργία 3:**

Κόστος = 0 I/Os και Αριθμός εγγραφών στην έξοδο = 100.

Έχουμε ήδη 100 εγγραφές από τα προηγούμενα βήματα, οι οποίες θα γίνουν Join με τις κατάλληλες εγγραφές των προϊόντων. Επομένως στην έξοδο οι εγγραφές θα είναι πάλι 100.

Υπάρχει απλό ευρετήριο στο ΠΡ.Barcode το οποίο είναι ήδη φορτωμένο στην μνήμη. Δεν χρειάζεται να κάνουμε κάποια ανάκτηση στην μνήμη. Επομένως το κόστος είναι 0.

**Σύνολο:**

Συνολικό Κόστος = 12 I/Os