ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

ΣΧΟΛΗ ΕΠΙΣΤΗΜΩΝ & ΤΕΧΝΟΛΟΓΙΑΣ ΤΗΣ ΠΛΗΡΟΦΟΡΙΑΣ
SCHOOL OF INFORMATION SCIENCES & TECHNOLOGY
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ
DEPARTMENT OF INFORMATICS

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΣΥΣΤΗΜΑΤΑ ΔΙΑΧΕΙΡΙΣΗΣ ΚΑΙ ΑΝΑΛΥΣΗΣ ΔΕΔΟΜΕΝΩΝ

2η Σειρά Ασκήσεων

Ονοματεπώνυμο: ***Μαρία Σχοινάκη***

Αριθμός Μητρώου: ***3210191***

Email: ***p3210191@aueb.gr***

Άσκηση 1

Α)

Τα ευρετήρια βρίσκονται στην μνήμη, άρα το κόστος I/O του να τα διαβάσουμε είναι μηδενικό. Τα ευρετήρια είναι **non-clustered**, άρα οι εγγραφές στον δίσκο είναι μη ταξινομημένες και άρα για το διάβασμα κάθε εγγραφής πρέπει κάθε φορά να διαβάζεται μία σελίδα (*ακόμα και αν 1 σελίδα περιέχει 2 εγγραφές που πληρούν τις προϋποθέσεις, πρέπει να προσπελαστεί η ίδια σελίδα 2 φορές*). Έτσι το πλήθος των εγγραφών που ικανοποιούν το επερώτημα, **ισούται** με το πλήθος των σελίδων που πρέπει να διαβαστούν, δηλαδή το κόστος σε I/O, αφού και τα ευρετήρια είναι στην μνήμη και δεν χρειάζεται επιπλέον διάβασμα.

Κόστος I/O επερωτήματος Q1

Έστω η σχέση:

x = **σ a=40 AND b=50** (R1)

$$T(x) = \frac{T(R1)}{V(R1.a,\ R1.b)} = \frac{T(R1)}{V(R1.a)*V(R1.b)} = \frac{5.000.000}{100*50} = \frac{5.000}{5} = \mathbf{1.000}\ \text{εγγραφές}$$

$$\text{Cost}(x) = \min \begin{cases} B(R1)\ tablescan \\ T(x)\ index\ seek \end{cases} = \min \begin{cases} 50.000 \\ 1.000 \end{cases} = \mathbf{1.000\ \ I/Os}$$

Κόστος I/O επερωτήματος R2

Έστω η σχέση:

y = σ s>96 AND s<=220 (R2)

Ψάχνουμε το πλήθος των εγγραφών που ανήκουν στο εύρος τιμών του s **[97, 220]** του ιστογράμματος:

[x..y] → #τιμών που ικανοποιούν την συνθήκη * $\frac{\text{αριθμός εγγραφών εύρους}}{\text{εύρος τιμών}}$

[1..100] → $4 * \frac{2.500}{100} = 4 * 25 =$ **100** εγγραφές

[101..200] → $100 * \frac{500}{100} = 100 * 5 =$ **500** εγγραφές

[201..400] → $20 * \frac{4.000}{200} = 20 * 20 =$ **400** εγγραφές

[401..500] → $0 * \frac{3.000}{100} =$ **0** εγγραφές

T(y) = 100 + 500 + 400 + 0 = **1.000** εγγραφές

$$\text{Cost}(y) = \min \begin{cases} B(R2)\ tablescan \\ T(y)\ index\ seek \end{cases} = \min \begin{cases} 2.000 \\ 1.000 \end{cases} = \mathbf{1.000\ \ I/Os}$$

Άρα το επερώτημα Q1 έχει ίσο κόστος με αυτό του Q2.

B) Αν αντί για **non-clustered** indexes, οι 2 σχέσεις είχαν **clustered** indexes, οι εγγραφές θα ήταν ταξινομημένες στον δίσκο και άρα για το διάβασμα των εγγραφών που ικανοποιούν την συνθήκη (*και αφού και τα ευρετήρια είναι πάνω στα γνωρίσματα κάθε συνθήκης*), θα χρειάζονταν να διαβαστούν τόσες σελίδες (*δηλαδή κόστος I/O*), όσες να χωράνε οι εγγραφές επιλογής του επερωτήματος. Όπως και στο πάνω ερώτημα, δεν έχουμε κόστος για την ανάγνωση των ευρετηρίων.

Κόστος I/O επερωτήματος Q1

$$B(x) = \frac{T(x)}{\frac{T(R1)}{B(R1)}} = \frac{1.000}{\frac{5.000.000}{50.000}} = \frac{1.000}{100} = \textbf{10 σελίδες}\ (I/O)$$

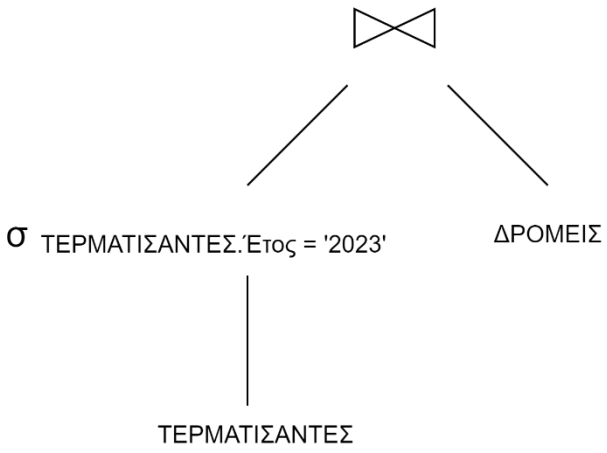
Κόστος I/O επερωτήματος Q2

$$B(y) = \frac{T(y)}{\frac{T(R2)}{B(R2)}} = \frac{1.000}{\frac{10.000}{2.000}} = \frac{1.000}{5} = \textbf{200 σελίδες}\ (I/O)$$

Άρα αν τα ευρετήρια, ήταν ευρετήρια συστάδων, το κόστος I/O του επερωτήματος Q1, θα ήταν μικρότερο από αυτό του επερωτήματος Q2. Παρατηρούμε ότι στα non clustered indexes, δεν υπήρχε διαφορά, καθώς το κόστος υπολογίζεται από τις εγγραφές, ενώ στα clustered indexes έπαιξε μεγάλο ρόλο πόσες εγγραφές μπορούσε να χωρέσει η κάθε σελίδα. Με την σχέση R2 να μπορεί να χωρέσει μόλις 5 εγγραφές ανά σελίδα, το κόστος σε σελίδες ήταν σαφώς πιο μεγάλο από ότι της R1, που μπορεί να χωρέσει 100 εγγραφές ανά σελίδα.

Άσκηση 2

Α)



Β) Έστω σχέση**:**

x = **σ Έτος = '2023'** (ΤΕΡΜΑΤΙΣΑΝΤΕΣ)

Κάθε σελίδα του πίνακα ΤΕΡΜΑΤΙΣΑΝΤΕΣ έχει $\frac{T(\text{ΤΕΡΜΑΤΙΣΑΝΤΕΣ})}{\text{Β(ΤΕΡΜΑΤΙΣΑΝΤΕΣ)}} = \frac{60.000}{600} = 100$ εγγραφές.

Από τις 60.000 συνολικές εγγραφές του πίνακα ΤΕΡΜΑΤΙΣΑΝΤΕΣ, οι $\frac{T(\text{ΤΕΡΜΑΤΙΣΑΝΤΕΣ})}{V(TEPMATIΣANTEΣ.Έτος)} = \frac{60.000}{4} = 15.000 = \text{T(x)}$ αντιστοιχούν στις εγγραφές του πίνακα, που αφορούν το έτος 2023. Άρα οι συνολικές σελίδες που αντιστοιχούν στο έτος 2023 του πίνακα ΤΕΡΜΑΤΙΣΑΝΤΕΣ είναι $\frac{15.000}{100} = 150$ σελίδες = Β(**x**).

Κόστος με SMJ

Ο αλγόριθμος **SMJ**, απαιτεί ταξινόμηση των πινάκων που λαμβάνουν μέρος στην συγχώνευση. Ο ένας πίνακας αποτελεί τον καινούριο πίνακα της σχέσης **x**, που είναι ένα κομμάτι του πίνακα ΤΕΡΜΑΤΙΣΑΝΤΕΣ. Ο άλλος πίνακας είναι ο ίδιος ο πίνακας ΔΡΟΜΕΙΣ, καθώς δεν υπόκεινται σε κάποιο φιλτράρισμα. Ο πίνακας ΔΡΟΜΕΙΣ, εφόσον υπάρχει ευρετήριο συστάδων, είναι ήδη ταξινομημένος ως προς το γνώρισμα ΚΔ και άρα δεν χρειάζεται περεταίρω ταξινόμηση.

Αφού Β(**x**) = 150 > Μ = 21, η ταξινόμηση δεν μπορεί να γίνει στην μνήμη, όμως Β(**x**) = 150 < M^2 = 441 άρα μπορεί να τρέξει η αποδοτική μέθοδος του αλγορίθμου.

Cost(Q) = Cost(x) + Cost(ΔΡΟΜΕΙΣ) + 2*B(x), * *1 read + 1 write
αφού, για την συγχώνευση χρειάζεται το **Cost(x) + Cost(ΔΡΟΜΕΙΣ)** προκειμένου να διαβαστούν για την συγχώνευση οι σελίδες των εκάστοτε πινάκων και για την ταξινόμηση*(ο πίνακας ΔΡΟΜΕΙΣ είναι ήδη ταξινομημένος ως προς το ΚΔ)* του πίνακα **x**, χρειάζονται **2*B(x)** σελίδες, **B(x)** για το διάβασμα της ταξινόμησης και **B(x)** για το γράψιμο των ταξινομημένων σελίδων.

Cost(x) = 150, αφού υπάρχει ευρετήριο συστάδων στο γνώρισμα ΤΕΡΜΑΤΙΣΑΝΤΕΣ.Έτος, το οποίο βρίσκεται στην μνήμη. Άρα δεν θα έχουμε επιπλέον κόστος για να διαβάσουμε το ευρετήριο και οι σελίδες που θα διαβάσουμε είναι **B(x) = 150**, καθώς όλες οι επιθυμητές εγγραφές χωράνε σε 150 σελίδες και δεν χρειάζεται να ψάξουμε σε άλλες σελίδες, αφού οι εγγραφές είναι ταξινομημένες ως προς το Έτος.

Cost(ΔΡΟΜΕΙΣ) = 200, αφού υπάρχει ευρετήριο συστάδων στο γνώρισμα ΚΔ, το οποίο βρίσκεται στην μνήμη. Άρα δεν θα έχουμε επιπλέον κόστος για να διαβάσουμε το ευρετήριο και οι σελίδες που θα διαβάσουμε είναι τόσες όσες οι σελίδες του πίνακα ΔΡΟΜΕΙΣ, δηλαδή **B(ΔΡΟΜΕΙΣ) = 200**.

Άρα **Cost(Q) = 150 + 200 + 2*150 = 650 I/O's**.

Κόστος με NLJ

Όπως υπολογίσαμε και στο προηγούμενο ερώτημα,
T(x)=15.000, B(x)=150, Cost(x)=150
T(ΔΡΟΜΕΙΣ)=40.000, B(ΔΡΟΜΕΙΣ)= 200, Cost(ΔΡΟΜΕΙΣ)=200 & M=21

Θα εξετάσουμε τόσο την πράξη x⋈ΔΡΟΜΕΙΣ όσο και την πράξη ΔΡΟΜΕΙΣ⋈x καθώς στον αλγόριθμο **NLJ**, η σειρά ενδέχεται να διαφοροποιήσει το κόστος.

x ⋈ ΔΡΟΜΕΙΣ

Cost(NLJ$_{x \bowtie \Delta\text{ΡΟΜΕΙΣ}}$) = Cost(x) + $\lceil \frac{B(x)}{M-1} \rceil$ * Cost(ΔΡΟΜΕΙΣ) = 150 + $\lceil \frac{150}{20} \rceil$ * 200 = 150 + 8 * 200 = 150 + 1.600 = **1.750 I/O's**

ΔΡΟΜΕΙΣ ⋈ x

Cost(NLJ$_{\Delta\text{ΡΟΜΕΙΣ} \bowtie x}$) = Cost(ΔΡΟΜΕΙΣ) + $\lceil \frac{B(\Delta ROMEIS)}{M-1} \rceil$ * Cost(x) = 200 + $\lceil \frac{200}{20} \rceil$ * 150 = 200 + 10 * 150 = 200 + 1.500 = **1.700 I/O's**

Άρα, η καλύτερη επιλογή για τον **SMJ** ήταν **650 I/O's**, ενώ για τον **NLJ**, ήταν **1.700 I/O's.**

Άσκηση 3

Ο ελάχιστος αριθμός ευρετηρίων που χρειάζονται, προκειμένου να επιτευχθεί το ελάχιστο κόστος εκτέλεσης είναι 2. Αυτά τα ευρετήρια είναι:

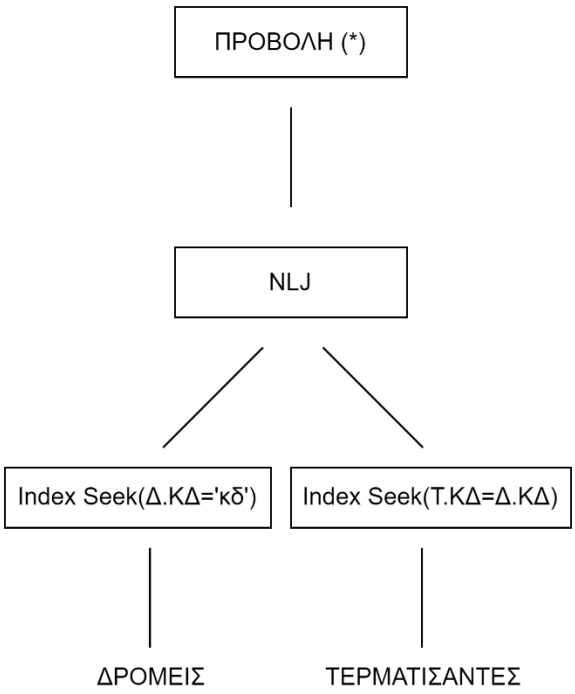
Clustered B+ tree index στο γνώρισμα ΚΔ του πίνακα ΔΡΟΜΕΙΣ

Γνωρίζουμε ότι το γνώρισμα ΚΔ αναφέρεται στην συνθήκη WHERE του επερωτήματος. Άρα ένα ευρετήριο πάνω σε αυτό το γνώρισμα, θα επιτάχυνε αρκετά την διαδικασία της αναζήτησης. Γνωρίζουμε επίσης ότι το ΚΔ είναι πρωτεύον κλειδί του πίνακα ΔΡΟΜΕΙΣ και άρα μοναδικό. Οπότε προκειμένου να ελαχιστοποιηθεί ακόμα περισσότερο το κόστος εκτέλεσης ως επί το πλείστων, μπορούμε αντί για απλό ευρετήριο (*non-clustered*), να φτιάξουμε ένα ευρετήριο συστάδων (*clustered index*).

Clustered B+ tree index στο γνώρισμα ΚΔ του πίνακα ΤΕΡΜΑΤΙΣΑΝΤΕΣ

Γνωρίζουμε ότι το γνώρισμα ΚΔ αναφέρεται στην συνθήκη WHERE του επερωτήματος. Άρα ένα ευρετήριο πάνω σε αυτό το γνώρισμα, θα επιτάχυνε αρκετά την διαδικασία της αναζήτησης. Το ΚΔ δεν είναι πρωτεύον κλειδί του πίνακα, ωστόσο δεν έχουμε θέμα επιλογής σε περίπτωση που ένα ΚΔ αντιστοιχεί σε <=4 εγγραφές, καθώς στο επερώτημα ζητούνται όλες. Άρα προκειμένου να ελαχιστοποιήσουμε ακόμα περισσότερο το κόστος, μπορούμε να φτιάξουμε ένα ευρετήριο συστάδων (*clustered index*).

Ο λόγος που χρησιμοποιήσαμε clustered indexes, είναι επειδή τα αποτελέσματα είναι ταξινομημένα στον δίσκο. Στον πίνακα δρομείς, τα αποτελέσματα κάθε φορά θα είναι 0 ή 1, οπότε δεν έχει τόση μεγάλη διαφορά στα πλαίσια του συγκεκριμένου ερωτήματος αν θα ήταν clustered ή non clustered. Αν υποθέσουμε ότι φτιάχνουμε ένα clustered ευρετήριο στο ΚΔ του πίνακα, τότε το κόστος θα ήταν <1, άρα 1 σελίδα κόστος. Αν τώρα φτιάχναμε non clustered ευρετήριο στο ΚΔ το κόστος θα ήταν ακριβώς 1, άρα 1 σελίδα κόστος. Δεν έχουν κάποια φανερή διαφορά, απλά χρησιμοποιείται το clustered γιατί είναι γενικά πιο δομημένο, τα δεδομένα είναι ταξινομημένα και το ΚΔ είναι πρωτεύον κλειδί. Στον πίνακα Τερματίσαντες τώρα, προτιμάται το clustered index καθώς θα έχει κόστος <1, άρα 1 σελίδα κόστος, ενώ το non clustered index έχει κόστος >1 άρα **τουλάχιστον** 2 σελίδες κόστος. Επειδή για κάθε ΚΔ στον πίνακα Τερματίσαντες μπορεί να χρειαστούμε έως και 4 εγγραφές, θα ήταν συνετό αυτά τα δεδομένα να ήταν ταξινομημένα ώστε να μην γίνονται άσκοπες προσπελάσεις, γι ’αυτό προτιμάται το clustered index. Επίσης ο αλγόριθμος συγχώνευσης δεν έχει μεγάλη διαφορά σε σχέση με άλλους, καθώς τα δεδομένα σε αυτό το ερώτημα που προκύπτουν είναι εξαιρετικά λίγα. Γενικά στα μικρά δεδομένα δουλεύει καλύτερα ο NLJ, οπότε αυτόν και θα ακολουθήσουμε.



X = **σ ΚΔ=κδ** (ΔΡΟΜΕΙΣ)

$$T(\mathbf{x}) = \frac{T(\Delta POMEIS)}{V(K\Delta)} = \frac{40.000}{40.000} = 1 \text{ εγγραφή}$$

$$B(\mathbf{x}) = \frac{T(x)}{\frac{T(\Delta POMEIS)}{B(\Delta POMEIS)}} = \frac{1}{\frac{40.000}{200}} = \frac{1}{200} = \mathbf{1\ σελίδα}\ (I/O)$$

Cost(x) = 1, αφού υπάρχει ευρετήριο συστάδων στο γνώρισμα ΔΡΟΜΕΙΣ.ΚΔ, το οποίο βρίσκεται στην μνήμη. Άρα δεν θα έχουμε επιπλέον κόστος για να διαβάσουμε το ευρετήριο και οι σελίδες που θα διαβάσουμε είναι **B(x) = 1**, καθώς όλες οι επιθυμητές εγγραφές (*1*) χωράνε σε 1 σελίδα και δεν χρειάζεται να ψάξουμε σε άλλες σελίδες.

Cost(ΤΕΡΜΑΤΙΣΑΝΤΕΣ) = 1, αφού υπάρχει ευρετήριο συστάδων στο γνώρισμα ΚΔ, το οποίο βρίσκεται στην μνήμη. Άρα δεν θα έχουμε επιπλέον κόστος για να διαβάσουμε το ευρετήριο και οι σελίδες που θα διαβάσουμε είναι **B(ΤΕΡΜΑΤΙΣΑΝΤΕΣ) = 1**, καθώς όλες οι επιθυμητές εγγραφές (max *4*) χωράνε σε 1 σελίδα και δεν χρειάζεται να ψάξουμε σε άλλες σελίδες.

Cost(NLJ$_{X \bowtie \text{ΤΕΡΜΑΤΙΣΑΝΤΕΣ}}$) = Cost(x) + $\lceil \frac{B(x)}{M-1} \rceil$ * Cost(ΤΕΡΜΑΤΙΣΑΝΤΕΣ) = 1 + $\lceil \frac{1}{20} \rceil$ * 1 = 1 + 1 = **2 I/O's**

Cost(NLJ$_{\text{ΤΕΡΜΑΤΙΣΑΝΤΕΣ} \bowtie X}$) = Cost(ΤΕΡΜΑΤΙΣΑΝΤΕΣ) + $\lceil \frac{B(ΤΕΡΜΑΤΙΣΑΝΤΕΣ)}{M-1} \rceil$ * Cost(x) = 1 + $\lceil \frac{1}{20} \rceil$ * 1 = 1 + 1 = **2 I/O's**

Άρα το κόστος του πλάνου εκτέλεσης είναι 2 I/O's

Τα ευρετήρια βρίσκονται στην μνήμη άρα δεν χρειαζόμαστε επιπλέον κόστος για να τα διαβάσουμε. Επίσης προκειμένου να ικανοποιηθεί η συνθήκη where και με την βοήθεια των ευρετηρίων, καταλήγουμε σε ένα κόστος 2, το οποίο είναι και το βέλτιστο καθώς προσπελαύνουμε 1 σελίδα για κάθε πίνακα. Αν είχαμε non clustered indexes, ή καθόλου indexes, το συνολικό κόστος επιλογής θα ήταν αρκετά μεγαλύτερο του 2. Θα ήταν σε ένα εύρος [5..800]. Οπότε το 2 που επιτύχαμε είναι βέλτιστο. Σε θέμα δόμησης των λειτουργιών τώρα, είναι πολύ πιο συμφέρον να περιοριστούν εξαρχής οι εγγραφές με το ΚΔ, παρά στο τέλος. Έτσι πετύχαμε και σε αυτό το κομμάτι με το πλάνο βελτιστοποίηση.

Άσκηση 4

Α)

1. Έστω x = σ Συνθέτης='Σταύρος Ξαρχάκος'

Η σχέση ΤΡΑΓΟΥΔΙΑ έχει = B(ΤΡΑΓΟΥΔΙΑ) = $\frac{T(ΤΡΑΓΟΥΔΙΑ)}{εγγραφές\ ανά\ σελίδα}$ = $\frac{1.000}{5}$ = 200 σελίδες

T(x) = $\frac{T(ΤΡΑΓΟΥΔΙΑ)}{V(Συνθέτης)}$ = $\frac{1.000}{100}$ = 10 εγγραφές

Υπάρχει ευρετήριο hash στο γνώρισμα Συνθέτης, το οποίο είναι στην μνήμη, άρα δεν θα έχουμε επιπλέον κόστος διαβάσματος του ευρετηρίου. Επίσης το hash index είναι είναι **non-clustered**, άρα οι εγγραφές στον δίσκο είναι μη ταξινομημένες και άρα για το διάβασμα κάθε εγγραφής πρέπει κάθε φορά να διαβάζεται μία σελίδα (*ακόμα και αν 1 σελίδα περιέχει 2 εγγραφές που πληρούν τις προϋποθέσεις, πρέπει να προσπελαστεί η ίδια σελίδα 2 φορές*). Έτσι το πλήθος των εγγραφών που ικανοποιούν το επερώτημα, **ισούται** με το πλήθος των σελίδων που πρέπει να διαβαστούν, δηλαδή το κόστος σε I/O. Άρα,

Cost(x) = min $\begin{cases} B(ΤΡΑΓΟΥΔΙΑ)\ tablescan \\ T(x)\ index\ seek \end{cases}$ = min $\begin{cases} 200 \\ 10 \end{cases}$ = **10 I/Os**

2. Έστω y = σ 26<=Ηλικία<=29

Η σχέση ΑΚΡΟΑΤΕΣ έχει = B(ΑΚΡΟΑΤΕΣ) = $\frac{\text{T(ΑΚΡΟΑΤΕΣ)}}{εγγραφές\ ανά\ σελίδα}$ = $\frac{30.000}{10}$ = 3.000 σελίδες

Η έρευνα διεξήχθη σε ακροατές ηλικίας 21 έως και 60 ετών, άρα 40 διαφορετικές ηλικίες, οπότε V(Ηλικία)=40. Επίσης, εμάς μας αφορούν 4 ηλικίες οπότε:

T(y) = $4 * \frac{\text{T}(ΑΚΡΟΑΤΕΣ)}{\text{V(Ηλικία)}} = 4 * \frac{30.000}{40} = 4 * 750$ = 3.000 εγγραφές

Αφού στο γνώρισμα Ηλικία υπάρχει ευρετήριο συστάδων, οι εγγραφές είναι ταξινομημένες στον δίσκο και άρα για το διάβασμα των εγγραφών που ικανοποιούν την συνθήκη, χρειάζεται να διαβαστούν τόσες σελίδες (*δηλαδή κόστος I/O*), όσες να χωράνε οι εγγραφές επιλογής του επερωτήματος. Όπως και στο πάνω ερώτημα, δεν έχουμε κόστος για την ανάγνωση των ευρετηρίων. Άρα,

Cost(y) = B(y) = $\frac{\text{T(y)}}{\frac{\text{T}(ΑΚΡΟΑΤΕΣ)}{\text{B}(ΑΚΡΟΑΤΕΣ)}} = \frac{3.000}{\frac{30.000}{3.000}} = \frac{3.000}{10}$ = **300 σελίδες** (I/O)

3. INLJ

Γνωρίζουμε ότι οι εγγραφές που δέχεται από την λειτουργία 2 ο αλγόριθμος είναι 3.000.

Υπάρχει ευρετήριο συστάδων (*clustered index*) B+ δέντρο στον πίνακα ΑΡΕΣΕΙ στο γνώρισμα ΚΑ και όλο το ευρετήριο βρίσκεται στην μνήμη. Επίσης κάθε ακροατής κατά μέσο όρο μπορεί να δηλώσει ότι του άρεσαν **17** διαφορετικά τραγούδια.

Σύμφωνα με τον NLJ, για κάθε εγγραφή του Υ, γίνεται αναζήτηση μέσω του ευρετηρίου για εγγραφές της σχέσης ΑΡΕΣΕΙ που ικανοποιούν την συνθήκη Υ.ΚΑ = ΑΡΕΣΕΙ.ΚΑ. Το κόστος για το διάβασμα του ευρετηρίου είναι μηδενικό, αφού βρίσκεται στην μνήμη, και επειδή είναι ευρετήριο συστάδων τότε για την ανάκτηση κάθε φορά των αποτελεσμάτων χρειάζονται **1 I/O**, αφού κάθε σελίδα χωράνε μέχρι 50 εγγραφές, οπότε οι 17 του ερωτήματος χωράνε σε 1 σελίδα.

Cost $_{\text{INLJ}}$ = T(Y) * 1 = 3.000 * 1 = 3.000

4. BNLJ

Δεν υπάρχει επιπλέον κόστος σε I/O's. Το join δεν έχει άμεση συσχέτιση με κάποια σχέση, με αποτέλεσμα οι εγγραφές που ικανοποιούν ή δεν ικανοποιούν τη συνθήκη του τίτλου φιλτράρονται στη μνήμη κατά τη διάρκεια της διαδικασίας του join, αποφεύγοντας επιπλέον προσπελάσεις δίσκου. (M=21)

ΤΕΛΙΚΟ ΚΟΣΤΟΣ

Cost **(σ Συνθέτης=΄Σταύρος Ξαρχάκος΄) + Cost(σ 26<=Ηλικία<=29) + Cost(INLJ) + Cost (BNLJ)**

= 10 + 300 + 3.000 + 0 = 3.310

Β)

1. Η λειτουργία 1 του πλάνου Β είναι ίδια με την λειτουργία 1 του πλάνου Α, άρα το συνολικό κόστος είναι **10 I/O**.

2. INLJ

Η σχέση ΑΡΕΣΕΙ έχει = Β(ΑΡΕΣΕΙ) = $\frac{\text{Τ(ΑΡΕΣΕΙ)}}{εγγραφές\ ανά\ σελίδα} = \frac{500.000}{50}$ = 10.000 σελίδες

Γνωρίζουμε ότι οι εγγραφές που δέχεται από την λειτουργία 1 ο αλγόριθμος είναι 10.

Υπάρχει απλό ευρετήριο (non *clustered index*) hash index στον πίνακα ΑΚΡΟΑΤΕΣ στο γνώρισμα ΚΑ και όλο το ευρετήριο βρίσκεται στην μνήμη. Ξέρουμε ότι V(Τίτλος) = 1000 και αφού τα δεδομένα είναι ομοιόμορφα κατανεμημένα, τότε οι συνολικές εγγραφές που έχουν τίτλο στον πίνακα ΑΡΕΣΕΙ, έναν οποιοδήποτε τίτλο x είναι, $\frac{\text{Τ(ΑΡΕΣΕΙ)}}{V(Τίτλος)} = \frac{500.000}{1.000}$ = 500 εγγραφές ανά τίτλο.

Σύμφωνα με τον NLJ, για κάθε εγγραφή του Χ, γίνεται αναζήτηση μέσω του ευρετηρίου για εγγραφές της σχέσης ΑΡΕΣΕΙ που ικανοποιούν την συνθήκη Χ.Τίτλος = ΑΡΕΣΕΙ.Τίτλος. Το κόστος για το διάβασμα του ευρετηρίου είναι μηδενικό, αφού βρίσκεται στην μνήμη, αλλά επειδή το ευρετήριο είναι απλό, τότε για την ανάκτηση κάθε φορά των αποτελεσμάτων χρειάζονται **500 I/O**, αφού στην χειρότερη περίπτωση κάθε εγγραφή θα βρίσκεται σε διαφορετική σελίδα.

Cost INLJ = T(X) * 500 = 10 * 500 = 5.000

3. INLJ

Γνωρίζουμε ότι οι εγγραφές που δέχεται από την λειτουργία 2 ο αλγόριθμος είναι 5.000.

Υπάρχει απλό ευρετήριο (non *clustered index*) hash index στον πίνακα ΑΡΕΣΕΙ στο γνώρισμα Τίτλος και όλο το ευρετήριο βρίσκεται στην μνήμη.

Σύμφωνα με τον NLJ, για κάθε εγγραφή της λειτουργία 2 (έστω Z), γίνεται αναζήτηση μέσω του ευρετηρίου για εγγραφές της σχέσης ΑΚΡΟΑΤΕΣ που ικανοποιούν την συνθήκη Z.ΚΑ = ΑΚΡΟΑΤΕΣ.ΚΑ. Το κόστος για το διάβασμα του ευρετηρίου είναι μηδενικό, αφού βρίσκεται στην μνήμη, αλλά επειδή το ευρετήριο είναι απλό, τότε για την ανάκτηση κάθε φορά των αποτελεσμάτων χρειάζονται **1 I/O**. Ο λόγος είναι ότι ψάχνουμε 1 εγγραφή για κάθε αποτέλεσμα του Z, οπότε το ευρετήριο δεν χρειάζεται να είναι συστάδων.

Cost INLJ = T(Z) * 1 = 5.000 * 1 = 5.000

4. σ 26<=Ηλικία<=29

Δεν υπάρχει επιπλέον κόστος σε I/O's. Η επιλογή ηλικίας έχει μηδενικό κόστος σε I/O's επειδή οι εγγραφές που ικανοποιούν ή δεν ικανοποιούν τη συνθήκη ηλικίας φιλτράρονται στη μνήμη κατά τη διάρκεια της διαδικασίας του join, αποφεύγοντας επιπλέον προσπελάσεις δίσκου. Έτσι, από στις 62 σελίδες του buffer, χωράνε να αποθηκευτούν όλες οι τελικές εγγραφές του ερωτήματος που είναι 500 ($\frac{\text{Εγγραφές Εισόδου}}{\text{V(ΑΚΡΟΑΤΕΣ,Ηλικία)}}$ * Εύρος Διαστήματος), αφού σε μία σελίδα χωράνε 10 εγγραφές από την σχέση ΑΚΡΟΑΤΕΣ. ($\frac{500}{10}$ = 50)

ΤΕΛΙΚΟ ΚΟΣΤΟΣ

Cost **(σ Συνθέτης='Σταύρος Ξαρχάκος') + Cost(INLJ) + Cost(INLJ) + Cost (σ 26<=Ηλικία<=29)**

= 10 + 5.000 + 5.000 + 0 = 10.010

Γ) Όπως παρατηρούμε και από τα κόστη, το Α πλάνο είναι αρκετά πιο οικονομικό σε κόστος I/O. Επίσης όπως παρατηρείται στο πλάνο Α, χρησιμοποιούνται ως επί το πλείστων ευρετήρια συστάδων, τα οποία μειώνουν κατά πολύ το κόστος I/O καθώς τα δεδομένα είναι ταξινομημένα, σε αντίθεση με το πλάνο Β στο οποίο χρησιμοποιούνται απλά ευρετήρια. Ένα άλλο γεγονός που κάνει το πλάνο Α να υπερισχύει του Β είναι η χρονολογική σειρά υπολογισμού. Στο πλάνο Β ξεκινάμε υπολογίζοντας τεράστιες ποσότητες δεδομένων για να καταλήξουμε εν τέλη στην επιλογή της ηλικίας. Στο πλάνο Α από την άλλη η επιλογή της ηλικίας γίνεται από το αρχικό στάδιο, εξοικονομώντας από την αρχή υπολογιστικό κόστος και επεξεργασμένα δεδομένα από βήμα σε βήμα.

Παρατήρηση: Όπου κρίνεται απαραίτητο θεωρούμε ceiling. (πχ. δεκαδικός αριθμός σελίδων)