Συστήματα Διαχείρισης και Ανάλυσης Δεδομένων

2η Σειρά Ασκήσεων

Παντελίδης Ιπποκράτης – 3210150

Άσκηση 1)

Δεδομένα)

i) Από την εκφώνηση μας δίνεται ότι η σχέση R1 έχει πλήθος εγγραφών **T(R1)** ίσο με **5.000.000** και καταλαμβάνει στον δίσκο αριθμό σελίδων **B(R1)** ίσο με **50.000**. Από αυτό μπορούμε να συμπεράνουμε ότι σε **μία σελίδα** χωράνε T(R1) / B(R1) = 5.00~~0.000~~ / 5~~0.000~~ = **100** εγγραφές της σχέσης R1. Ξέρουμε ακόμα ότι ο αριθμός των διακριτών τιμών του γνωρίσματος a στην R1 **V(R1, a)** είναι **100** και του γνωρίσματος b **V(R1, b)** είναι **50**.

ii) Από την εκφώνηση μας δίνεται ότι η σχέση R2 έχει πλήθος εγγραφών **T(R2)** ίσο με **10.000** και καταλαμβάνει στον δίσκο αριθμό σελίδων **B(R2)** ίσο με **2.000**. Από αυτό συμπεραίνουμε ότι σε **μία σελίδα** χωράνε T(R2) / B(R2) = 10.~~000~~ / 2.~~000~~ = **5** εγγραφές της σχέσης R2. Επίσης έχουμε το ακόλουθο ιστόγραμμα :

s	Αριθμός Εγγραφών
[1..100]	2,500
[101..200]	500
[201..400]	4,000
[401..500]	3,000

A)

Υπολογισμός κόστους 1ου επερωτήματος :

Εφόσον δίνεται ότι υπάρχει **non clustered** ευρετήριο B+ δέντρο στο ζεύγος γνωρισμάτων **(a,b),** λαμβάνοντας υπόψη τα προηγούμενα δεδομένα προκύπτει ότι η συνθήκη (a = 40) περιορίζει τις εγγραφές που θα διαβαστούν στις T(R1) / V(R1,a) δηλαδή στις 5.000.0~~00~~/1~~00~~ = 50.000, ενώ η συνθήκη (b = 50) στις T(R1) / V(R1,b) = 5.000.00~~0~~ / 5~~0~~ = 100.000 εγγραφές. Συνδυάζοντας λοιπόν αυτές τις δύο συνθήκες, ώστε να πάρουμε το WHERE του ερωτήματος Q1 (a = 40 and b = 50) προκύπτει ότι θα διαβάσουμε συνολικά με την χρήση του ευρετηρίου T(R1) / (V(R1,a) x V(R1,b)) εγγραφές, δηλαδή θα έχουμε κόστος I/O ίσο με 5.000.~~000~~ / (1~~00~~ x 5~~0~~) =

1.000 I/O.
Αυτό συμβαίνει επειδή το κόστος I/O είναι το κόστος ανάκτησης εγγραφών από τον δίσκο και λόγω του απλού ευρετηρίου οι εγγραφές αυτές μπορεί να βρίσκονται ο-πουδήποτε στον δίσκο και επομένως οι 1000 εγγραφές μας μπορούν να βρίσκονται στην χειρότερη περίπτωση η κάθε μία σε διαφορετικό μπλοκ. Αξίζει να σημειωθεί τέλος, ότι επειδή το **ευρετήριο** βρίσκεται στην **μνήμη**, το κόστος **αναζήτησης** εγ-γραφών στην μνήμη είναι αμελητέο και δίνει **0 I/O**.

Υπολογισμός κόστους 2ου επερωτήματος :
Αναλύοντας το ιστόγραμμα σύμφωνα με τις τιμές που ορίζουν το διάστημα του WHERE [97, 220] φτάνουμε στα ακόλουθα συμπεράσματα :

- **[1...100]** : Σε αυτό το διάστημα ανήκουν μόνο **4 τιμές** (97, 98, 99, 100) από αυτές του διαστήματος που υπάρχει στο Q2 και με την υπόθεση ότι η κατανο-μή των τιμών είναι **ομοιόμορφη** προκύπτει ότι το **ποσοστό** των τιμών αυτού του διαστήματος που ικανοποιούν την συνθήκη είναι 4 / 100 = **0.04** όπου **100** είναι το **εύρος** του διαστήματος. Από την στιγμή που το διάστημα έχει **2.500 εγγραφές**, ο αριθμός των **εγγραφών** που **πληρούν** την συνθήκη είναι 0.04 x 2.500 = **100**.
- **[101...200]** : Όλες οι **200 τιμές** αυτές του διαστήματος είναι μέρος του δια-στήματος του επερωτήματος Q2, οπότε με την ίδια υπόθεση **ομοιόμορφης** κατανομής το **ποσοστό** των τιμών που ικανοποιούν την συνθήκη είναι **100%**. Εφόσον το διάστημα αυτό έχει **500** εγγραφές, οι **εγγραφές** που τελικά **ικα-νοποιούν** την συνθήκη είναι 500 x 1.00 = **500**.
- **[201...400]** : Αντίστοιχα σε αυτό το διάστημα ανήκουν **20 τιμές** και με την υπόθεση **ομοιόμορφης** κατανομής προκύπτει ότι το **ποσοστό** των τιμών που ικανοποιούν την συνθήκη είναι 2θ / 20θ = **0.1** όπου **200** είναι το **εύρος** του διαστήματος. Αφού το συγκεκριμένο διάστημα έχει **4.000 εγγραφές**, αυτές που **πληρούν** την συνθήκη είναι 4.000 x 0.1 = **400**.
- **[401...500]** : Το διάστημα αυτό δεν έχει **καμία εγγραφή** από αυτές του δια-στήματος του επερωτήματος, επομένως τελικά οι **εγγραφές** που **πληρούν** την συνθήκη είναι **0**.

Αθροίζοντας τα αποτελέσματα από όλα τα διαστήματα του ιστογράμματος βρίσκου-με ότι οι **συνολικές** εγγραφές που είναι **εντός** του διαστήματος του επερωτήματος (s > 96 and s <= 220) είναι 100 + 500 + 400 = **1.000**. Άρα, με την ίδια εξήγηση όπβς και στο πορηγούμενο επερώτημα θα χρειαστούμε **1000 I/O**.

Συμπεραίνουμε λοιπόν ότι τα δύο επερωτήματα έχουν το ίδιο κόστος I/O και αυτό είναι ίσο με 1.000.

Β)

Υπολογισμός κόστους 1ου επερωτήματος :

Έστω τώρα ότι το ευρετήριο B+ δέντρο είναι clustered, που σημαίνει ότι οι εγγραφές που είναι ίδιες οργανώνονται διαδοχικά στον δίσκο. Επομένως όλες οι 1.000 εγγραφές που ικανοποιούν την συνθήκη (a = 40 and b = 50) θα βρίσκονται η μία μετά με την άλλη στον δίσκο. Όπως υπολογίστηκε προηγουμένως, σε κάθε σελίδα στον δίσκο χωράνε 100 εγγραφές της σχέσης R1 οπότε για να διαβάσουμε τις εγγραφές του επερωτήματος Q1 θα χρειαστούμε 1.0~~00~~ / 1~~00~~ = **10 I/O**.

Υπολογισμός κόστους 2ου επερωτήματος :

Έστω τώρα ότι και το ευρετήριο που χρησιμοποιείται για το επερώτημα Q2 είναι clustered που σημαίνει ότι οι εγγραφές που βρίσκονται εντός του διαστήματος [97, 220] βρίσκονται διαδοχικά στον δίσκο. Υπολογίσαμε παραπάνω ότι σε κάθε σελίδα στον δίσκο χωράνε 5 εγγραφές της σχέσης R2, οπότε για να διαβάσουμε τις 1000 που επιστρέφει το επερώτημα Q2 θα κάνουμε 1.000 / 5 = **200 I/O**.

Αν χρησιμοποιήσουμε και για τα δύο επερωτήματα clustered ευρετήριο παρατηρούμε ότι τα αποτελέσματα είναι αρκετά διαφορετικά. Από εκεί που το κόστος I/O ήταν το ίδιο με το non clustered ευρετήριο, τώρα το επερώτημα Q1 έχει πολύ πιο μικρό κόστος από το Q2.

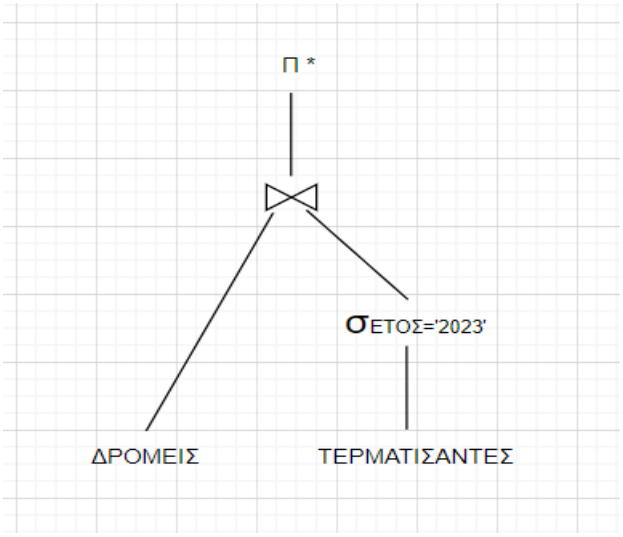
Άσκηση 2)

Α)

Για παρακάτω επερώτημα σε SQL

```
SELECT *
 FROM ΔΡΟΜΕΙΣ, ΤΕΡΜΑΤΙΣΑΝΤΕΣ
WHERE ΔΡΟΜΕΙΣ.ΚΔ=ΤΕΡΜΑΤΙΣΑΝΤΕΣ.ΚΔ AND Έτος='2023'
```

φαίνεται το ακόλουθο τελικό βελτιστοποιημένο λογικό πλάνο :



Β)
Από τα δεδομένα της εκφώνησης προκύπτουν :

i) Η σχέση **ΔΡΟΜΕΙΣ** περιέχει 40.000 εγγραφές **Τ(ΔΡΟΜΕΙΣ) = 40.000** και κατάλαμβάνει στον δίσκο αριθμό σελίδων ίσο με 200 **Β(ΔΡΟΜΕΙΣ) = 200**. Επομένως, σε **μία σελίδα** χωράνε Τ(ΔΡΟΜΕΙΣ) / Β(ΔΡΟΜΕΙΣ) = 40.0~~00~~ / 2~~00~~ = **400 εγγραφές** της σχέσης ΔΡΟΜΕΙΣ.

ii) Η σχέση **ΤΕΡΜΑΤΙΣΑΝΤΕΣ** περιέχει 60.000 εγγραφές **Τ(ΤΕΡΜΑΤΙΣΑΝΤΕΣ) = 60.000** και καταλαμβάνει στον δίσκο 600 σελίδες **Β(ΤΕΡΜΑΤΙΣΑΝΤΕΣ) = 600**. Άρα, σε μία σελίδα χωράνε Τ(ΤΕΡΜΑΤΙΣΑΝΤΕΣ) / Β(ΤΕΡΜΑΤΙΣΑΝΤΕΣ) = 60.0~~00~~ / 6~~00~~ = **100 εγγραφές** της σχέσης ΤΕΡΜΑΤΙΣΑΝΤΕΣ. Επίσης, μπορούμε να εξάγουμε από την εκφώνηση ότι το πεδίο Έτος της σχέσης αυτής μπορεί να πάρει 4 μόνο τιμές (2020, 2021, 2022, 2023) και άρα **V(ΤΕΡΜΑΤΙΣΑΝΤΕΣ, Έτος) = 4**.

<u>Αριστερή Συνθήκη Λογικού Πλάνου :</u>
Έστω ότι **Χ** είναι οι **εγγραφές** της σχέσης **ΔΡΟΜΕΙΣ**, οι οποίες δεν φιλτράρονται από κάποια συνθήκη και φτάνουν όπως είναι στην σύζευξη που ακολουθεί. Το ευρετήριο συστάδων που υπάρχει στο γνώρισμα ΔΡΟΜΕΙΣ.ΚΔ δεν χρησιμοποιείται ακόμα και τελικά με **table scan** βρίσκουμε ότι **Τ(Χ) = 40.000** και **Β(Χ) = 200**. Άρα το συνολικό κόστος υπολογισμού είναι του Χ είναι **cost(X) = 200**.

<u>Δεξιά Συνθήκη Λογικού Πλάνου :</u>
Έστω ότι **Υ** είναι το αποτέλεσμα της συνθήκης $\sigma_{\text{Έτος='2023'}}$(ΤΕΡΜΑΤΙΣΑΝΤΕΣ), για να υπολογίσουμε τον αριθμό των εγγραφών του **Τ(Υ)** θα χρησιμοποιήσουμε τα παραπάνω δεδομένα. Μπορούμε να συμπεράνουμε ότι η συνθήκη (Έτος = '2023') περιο-

ρίζει τις εγγραφές που επιστρέφονται στις T(ΤΕΡΜΑΤΙΣΑΝΤΕΣ)/V(ΤΕΡΜΑΤΙΣΑΝΤΕΣ, Έτος) = 60.000 / 4 = **15.000**. Επίσης, επειδή σε κάθε σελίδα χωράνε 100 εγγραφές βρίσκουμε ότι **B(Y)** = 15.00θ / 10θ = **150 σελίδες**. Το κόστος του υπολογισμού του Y με **table scan** είναι **B(ΤΕΡΜΑΤΙΣΑΝΤΕΣ) = 600**, ενώ με την χρήση του **ευρετηρίου συστάδων** στο πεδίο ΤΕΡΜΑΤΙΣΑΝΤΕΣ.Έτος είναι **B(Y) = 150**. Προτιμούμε το μικρότερο κόστος οπότε και επιλέγουμε **cost(Y) = 150.**

Κόστος SMJ

Ο αλγόριθμος **SMJ** προϋποθέτει οι πίνακες που συμμετέχουν στην σύζευξη, στην περίπτωση μας οι X και Y, να είναι **ταξινομημένοι**. Στην άσκηση μας, ο πίνακας X είναι ταξινομημένος, ενώ ο Y όχι οπότε και θα τον ταξινομήσουμε. Αρχικά, ελέγχουμε αν το μέγεθος του πίνακα είναι μικρότερο από αυτό της **διαθέσιμης μνήμης (M = 21)**, ώστε να δούμε αν μπορούμε να τον ταξινομήσουμε στην μνήμη χωρίς κόστος I/O με κάποιον από τους γνωστούς αλγορίθμους ταξινόμησης. Παρατηρούμε **ότι B(Y) > M (150 > 21)**, επομένως για να δούμε αν μπορούμε να χρησιμοποιήσουμε την αποδοτική έκδοση του SMJ πρέπει υπολογίσουμε τις υπολίστες που φτιάχνει ο πίνακας μας.

- Για τον **Y** έχουμε : ⌈B(Y) / M⌉ = ⌈150 / 21⌉ = **8 υπολίστες**

Οπότε έχουμε **8 < 21 = M** όποτε μπορούμε να τρέξουμε την **αποδοτική** έκδοση του αλγορίθμου.

Άρα το **κόστος** του αλγορίθμου **SMJ** είναι :

cost(SMJ) = cost(X) + cost(Y) + 2 * B(Y)

= 200 + 150 + 2 * 150 = **650 I/O**

καθώς δεν χρειάζεται να λάβουμε υπόψη το κόστος ταξινόμησης του πίνακα X.

Κόστος NLJ

Για τον αλγόριθμο NLJ, επειδή η σειρά με την οποία εκτελείται η πράξη της συζευξης μεταβάλλει το κόστος, θα εξετάσουμε τις δύο ακόλουθες περιπτώσεις :

1) Κόστος NLJ με εξωτερική την σχέση X

Το κόστος του αλγορίθμου NLJ με εξωτερική σχέση την X είναι το ακόλουθο :

cost(NLJ) = cost(X) + ⌈B(X) / M - 1⌉ * cost(Y)

= 200 + ⌈200 / 21⌉ * 150 = 200 + 10 * 150 = **1700 I/O**

2) Κόστος NLJ με εξωτερική την σχέση Y

Το κόστος του αλγορίθμου NLJ με εξωτερική σχέση την Y είναι το ακόλουθο :

cost(NLJ) = cost(Y) + ⌈B(Y) / M - 1⌉ * cost(X)

= 150 + ⌈150 / 21⌉ * 200 = 150 + 8 * 200 = **1750 I/O**

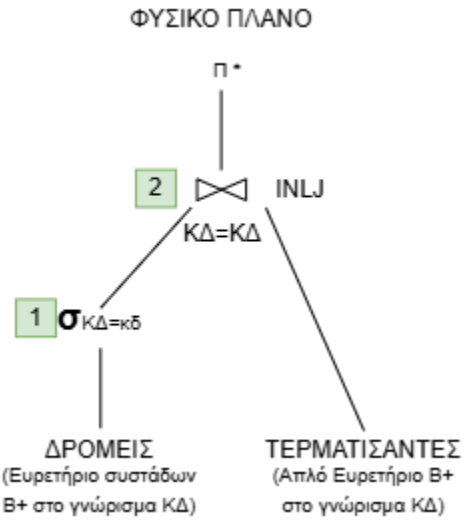
Παρατηρούμε ότι ο αλγόριθμος **NLJ** με **εξωτερική** την σχέση **X** έχει το **μικρότερο** κόστος εκ των δύο, και αυτό άλλωστε είναι λογικό καθώς είναι πιο φθηνό να έχω ως εξωτερική την μεγάλη σε κόστος σχέση (X) και να επαναλαμβάνω την μικρότερη παρά το αντίθετο.

Το **μικρότερο** κόστος επιτεύχθηκε με την χρήση του αλγορίθμου **SMJ** και είναι ίσο με **650 I/O**.

Άσκηση 3)

```
SELECT *
 FROM ΔΡΟΜΕΙΣ, ΤΕΡΜΑΤΙΣΑΝΤΕΣ
WHERE ΔΡΟΜΕΙΣ.ΚΔ=ΤΕΡΜΑΤΙΣΑΝΤΕΣ.ΚΔ AND ΔΡΟΜΕΙΣ.ΚΔ=κδ.
```

Για το παραπάνω ερώτημα θα σχεδιάσουμε το ακόλουθο φυσικό πλάνο :



Υπολογισμός Κόστους :

Βήμα 1)

Στον 1ο βήμα φιλτράρουμε τις εγγραφές της σχέσης ΔΡΟΜΕΙΣ για να βρούμε αυτές που έχουν στο πεδίο ΚΔ τιμή ίση με κδ. Το γνώρισμα **ΚΔ** είναι **πρωτεύον κλειδί** της

σχέσης οπότε υπάρχει μόνο **μία εγγραφή** που ικανοποιεί την συνθήκη. Όπως φαίνεται και από το πλάνο χρησιμοποιούμε ένα **ευρετήριο συστάδων** στο γνώρισμα **ΚΔ** της σχέσης ΔΡΟΜΕΙΣ, το οποίο μας εξασφαλίζει ότι οι σελίδες αποθηκεύονται ταξινομημένες ως προς το πεδίο ΚΔ και εν συνεχεία αποδοτικότερη αναζήτηση και πράξη join. Όπως προαναφέραμε το πρωτεύον κλειδί της σχέσης είναι το πεδίο ΚΔ επομένως εφόσον η **σχέση** έχει **40.000 εγγραφές**, κάθε μία από αυτές έχει διαφορετικό ΚΔ, οπότε **V(ΔΡΟΜΕΙΣ, ΚΔ) = 40.000**. Τελικά, το βήμα 1 επιστρέφει Τ(ΔΡΟΜΕΙΣ) / V(ΔΡΟΜΕΙΣ, ΚΔ) = ~~40.000~~ / ~~40.000~~ = **1 εγγραφή** και επομένως έχουμε και **κόστος I/O** ίσο με **1 I/O**.

Βήμα 2)
Για την πράξη του **join** έχει επιλεγεί ο αλγόριθμος **INLJ**. Πιο συγκεκριμένα για κάθε εγγραφή, μία στην περίπτωση μας, που φτάνει ως είσοδος σε αυτόν χρησιμοποιεί το **απλό ευρετήριο** που φτιάξαμε στο πεδίο **ΚΔ** της σχέσης **ΤΕΡΜΑΤΙΣΑΝΤΕΣ** για να ανακτήσει τις εγγραφές που πληρούν την συνθήκη (ΔΡΟΜΕΙΣ.ΚΔ=ΤΕΡΜΑΤΙΣΑΝΤΕΣ.ΚΔ). Για την σχέση ΤΕΡΜΑΤΙΣΑΝΤΕΣ όπως είπαμε φτιάξαμε απλό ευρετήριο, θεωρητικά θα μπορούσε να είναι και clustered, απλά επειδή το clustered index συνήθως φτιάχνεται για τα πρωτεύοντα κλειδιά επιλέχθηκε το απλό. Η σχέση τώρα έχει **Τ(ΤΕΡΜΑΤΙΣΑΝΤΕΣ) = 60.000 εγγραφές** και επειδή ξέρουμε πως το πεδίο **ΚΔ** για την σχέση **ΔΡΟΜΕΙΣ** έχει **40.000 μοναδικές τιμές**, για την σχέση ΤΕΡΜΑΤΙΣΑΝΤΕΣ θα έχει πάλι το **πολύ 40.000 μοναδικές τιμές**, **άρα V(ΤΕΡΜΑΤΙΣΑΝΤΕΣ, ΚΔ) = 40.000**. Επομένως, με την χρήση του ευρετηρίου στο γνώρισμα ΚΔ αναμένουμε να επιστραφούν Τ(ΤΕΡΜΑΤΙΣΑΝΤΕΣ) / V(ΤΕΡΜΑΤΙΣΑΝΤΕΣ, ΚΔ) = 60.000 / 40.000 = **1.5 εγγραφές**, και επειδή μιλάμε για εγγραφές θα πάρουμε τον αμέσως επόμενο ακέραιο, δηλαδή **2 εγγραφές**. Οπότε έχουμε και **κόστος I/O** ίσο με **2 I/O** για την **μία εγγραφή**, αλλά από την στιγμή που έχουμε 1 εγγραφή, **συνολικά** θα χρειαστούμε 1 * 2 = **2 I/O**.

Σύνοψη:
Όπως φαίνεται από το παραπάνω φυσικό πλάνο, ο **ελάχιστος αριθμός ευρετηρίων** που δίνουν το **ελάχιστο κόστος** είναι **2**. Οι λόγοι που χρησιμοποιήσαμε τα συγκεκριμένα ευρετήρια αναλύονται παραπάνω. Ο πιο **αποδοτικός αλγόριθμος** για την συγκεκριμένη πράξη **join** φαίνεται να είναι ο **INLJ**, καθώς κάνει επανάληψη πάνω σε ένα μικρό set (μήκος 1 -> αριστερή είσοδος βήματος 1) και βρίσκει με πολύ μικρό κόστος , λόγω της χρήσης ευρετηρίου, τις εγγραφές που χρειάζονται από την δεξιά σχέση. Η χρήση του **NLJ δεν** θα **αξιοποιούσε** το **ευρετήριο** που φτιάξαμε στην σχέση ΤΕΡΜΑΤΙΣΑΝΤΕΣ, ενώ του **SMJ** ίσως να ήταν λίγο πιο **αυξημένο** λόγω

του πιθανού κόστους **ταξινόμησης**. Καταλήγουμε λοιπόν, ότι ο **ελάχιστος αριθμός ευρετηρίων** είναι **2 ευρετήρια** και επιτυγχάνουν **ελάχιστο κόστος** και αυτό ίσο με **2 I/O**.

Άσκηση 4)
Από τις πληροφορίες της εκφώνησης ξέρουμε τα εξής για τις 3 σχέσεις :

i) Η σχέση **ΑΚΡΟΑΤΕΣ** περιέχει 30.000 εγγραφές **Τ(ΑΚΡΟΑΤΕΣ) = 30.000** και σε κάθε **σελίδα** χωράνε **10 εγγραφές**, οπότε οι σελίδες που καταλαμβάνει η σχέση στον δίσκο είναι **Β(ΑΚΡΟΑΤΕΣ)** = Τ(ΑΚΡΟΑΤΕΣ) / 10 = 30.00~~0~~ /1~~0~~ = **3.000**. Επίσης το γνώρισμα Ηλικία της σχέσης παίρνει τιμές από 21 μέχρι 60, δηλαδή 40 τιμές, όποτε προκύπτει ότι οι μοναδικές τιμές του γνωρίσματος είναι **V(ΑΚΡΟΑΤΕΣ, Ηλικία) = 40**. Η σχέση έχει **πρωτεύον κλειδί** το γνώρισμα **ΚΑ**, οπότε οι μοναδικές τιμές του γνωρίσματος αυτού είναι όσες και οι εγγραφές της σχέσης, δηλαδή ισχύει **V(ΑΚΡΟΑΤΕΣ, ΚΑ) = 30.000 εγγραφές**.

ii) Η σχέση **ΤΡΑΓΟΥΔΙΑ** περιέχει 1.000 εγγραφές **Τ(ΤΡΑΓΟΥΔΙΑ) = 1.000** και σε κάθε **σελίδα** χωράνε **5 εγγραφές**, οπότε οι σελίδες που καταλαμβάνει η σχέση στον δίσκο **Β(ΤΡΑΓΟΥΔΙΑ)** = Τ(ΤΡΑΓΟΥΔΙΑ) / 5 = 1000 / 5 = **200**. Μας δίνεται επίσης ότι το γνώρισμα της σχέσης Συνθέτης μπορεί να πάρει 100 διαφορετικές τιμές, οπότε **V(ΤΡΑΓΟΥΔΙΑ, Συνθέτης) = 100**. Η σχέση έχει **πρωτεύον κλειδί** το γνώρισμα **Τίτλος**, οπότε οι μοναδικές τιμές του γνωρίσματος αυτού είναι όσες και οι εγγραφές της σχέσης, δηλαδή ισχύει **V(ΤΡΑΓΟΥΔΙΑ, Τίτλος) = 1.000 εγγραφές**.

iii) Η σχέση **ΑΡΕΣΕΙ** περιέχει 500.000 εγγραφές **Τ(ΑΡΕΣΕΙ) = 500.000** και σε κάθε **σελίδα** χωράνε **50 εγγραφές**, οπότε οι σελίδες που καταλαμβάνει η σχέση στον δίσκο **Β(ΑΡΕΣΕΙ)** = Τ(ΑΡΕΣΕΙ) / 50 = 500.00~~0~~ / 5~~0~~ = **10.000**.

Α)
Το κόστος I/O κάθε μίας από τις 4 λειτουργίες του πλάνου Α φαίνεται παρακάτω :

Βήμα 1)
Το hash ευρετήριο λειτουργεί με τον ίδιο τρόπο όπως και το non clustered B+ δεντρο ευρετήριο. Εφόσον υπάρχει **hash index** στο γνώρισμα **Συνθέτης** της σχέσης **ΤΡΑΓΟΥΔΙΑ** οι **εγγραφές** που επιστρέφει η συνθήκη (Συνθέτης=”Σταύρος Ξαρχάκος” είναι Τ(ΤΡΑΓΟΥΔΙΑ) / V(ΤΡΑΓΟΥΔΙΑ, Συνθέτης) = 1.0~~0~~~~0~~ / 1~~0~~~~0~~ = **10**. Επομένως

επειδή ουσιαστικά το ευρετήριο μας είναι non clustered η κάθε μία εγγραφή μπορεί να βρίσκεται σε διαφορετική σελίδα στον δίσκο, οπότε θα χρειαστούμε **10 I/O**.

Βήμα 2)
Η σχέση **ΑΚΡΟΑΤΕΣ** έχει **clustered** index στο πεδίο **Ηλικία**. Οι εγγραφές της σχέσης που έχουν σε αυτό το πεδίο τιμή μέσα στο διάστημα 4 τιμών [26, 29], δηλαδή αυτές που **πληρούν** την **συνθήκη,** είναι Τ(ΑΚΡΟΑΤΕΣ) / V(ΑΚΡΟΑΤΕΣ, Ηλικία) * 4 = 30.00θ / 4θ * 4 = **3.000**. Επομένως, λόγω του clustered index οι εγγραφές αυτές είναι σε συνεχόμενες θέσεις στον δίσκο οπότε τελικά θα έχουμε **κόστος I/O** ίσο με 3.000 / 10 = **300 I/O**, όπου 10 είναι ο αριθμός των εγγραφών της σχέσης αυτής που χωράνε σε μία σελίδα.

Βήμα 3)
Ο αλγόριθμος **INLJ** για κάθε μία από τις **3.000 εγγραφές** που δέχεται σαν **είσοδο** (αριστερό κλαδί) χρησιμοποιεί το **ευρετήριο** που υπάρχει στο γνώρισμα **ΑΡΕΣΕΙ.ΚΑ** για ανακτήσει τις αντίστοιχες εγγραφές της σχέσης ΑΡΕΣΕΙ. Επειδή υποθέτουμε ομοιόμορφη κατανομή τιμών για **κάθε εγγραφή** που δέχεται ως **είσοδο** ο INLJ θα επιστρέψει Τ(ΑΡΕΣΕΙ) / V(ΑΡΕΣΕΙ, ΚΑ) = 500.000 / 30.000 = 16,6 δηλαδή **17 εγγραφές**. Το **V(ΑΡΕΣΕΙ, ΚΑ)** έχει τιμή **30.000** λόγω του ότι ισχύει **V(ΑΚΡΟΑΤΕΣ, ΚΑ) = 30.000**. Επομένως, ο αριθμός των **εγγραφών** στην **έξοδο** του βήματος είναι 3.000 * 17 = **51.000** και το κόστος λόγω του ότι υπάρχει **clustered index** στο γνώρισμα **ΚΑ** της σχέσης ΑΡΕΣΕΙ και δεδομένου ότι 17 εγγραφές της σχέσης χωράνε σε μία σελίδα είναι **3000 I/O**. Αυτό συμβαίνει διότι για κάθε μία από τις 3.000 εγγραφές έχουμε κόστος 1 I/O.

Βήμα 4)
Η λειτουργία αυτή μπορεί να εκτελεστεί στην **μνήμη χωρίς κόστος**, καθώς οι εγγραφές βρίσκονται ήδη εκεί από τα προηγούμενα βήματα, οπότε έχουμε **κόστος I/O** ίσο με **0 I/O**.

Αθροίζοντας τα κόστη των επιμέρους λειτουργιών του πλάνου Α παίρνουμε ως **συνολικό κόστος I/O** για το **πλάνο Α** το 10 + 300 + 3000 + 0 = **3310 I/O**.

Β)
Το κόστος I/O κάθε μίας από τις 4 λειτουργίες του πλάνου Β φαίνεται παρακάτω :

Βήμα 1)
Το Βήμα 1 του πλάνου Β είναι το **ίδιο** ακριβώς με αυτό του **πλάνου Α**, οπότε στην έξοδο βγαίνουν **10 εγγραφές** και θα χρειαστούμε **10 I/O**.

Βήμα 2)
Ο αλγόριθμος **INLJ** για κάθε από τις **10 εγγραφές** που δέχεται σαν **είσοδο** (αριστερό κλαδί) χρησιμοποιεί το **ευρετήριο** που υπάρχει στο γνώρισμα **ΑΡΕΣΕΙ.Τίτλος** για να ανακτήσει τις αντίστοιχες εγγραφές. Το hash ευρετήριο αυτό όπως είπαμε προηγουμένως συμπεριφέρεται με παρόμοιο τρόπο όπως ένα non clustered index, οπότε οι **εγγραφές** που θα **επιστρέψει** είναι Τ(ΑΡΕΣΕΙ) / V(ΑΡΕΣΕΙ, Τίτλος) = 500.~~θθθ~~ / 1.~~θθθ~~ = **500**. Το **V(ΑΡΕΣΕΙ, Τίτλος)** έχει τιμή **1.000** καθώς ο **Τίτλος** είναι **πρωτεύον** κλειδί της σχέσης **ΤΡΑΓΟΥΔΙΑ** και η σχέση τραγούδια έχει 1.000 οπότε υπάρχουν 1.000 μοναδικές τιμές για αυτό το γνώρισμα. Επομένως, στην **έξοδο** του βήματος έχουμε 10 * 500 = **5.000 εγγραφές** και κόστος I/O ίσο με **5000 I/O**.

Βήμα 3)
Ο αλγόριθμος **INLJ** για κάθε μία από τις **5.000 εγγραφές** που δέχεται ως **είσοδο** από το αριστερό κλαδί χρησιμοποιεί το απλό **ευρετήριο** που υπάρχει στο γνώρισμα **ΚΑ** της σχέσης ΑΚΡΟΑΤΕΣ για να ανακτήσει τις αντίστοιχες εγγραφές. Οι **εγγραφές** που θα επιστραφούν είναι Τ(ΑΚΡΟΑΤΕΣ) / V(ΑΚΡΟΑΤΕΣ, ΚΑ) = 30.000 / 30.000 = **1**. Όμοια με πριν αφού το **ΚΑ** είναι **πρωτεύον κλειδί** του πίνακα **ΑΚΡΟΑΤΕΣ** και επειδή ο πίνακας έχει 30.000 εγγραφές, υπάρχουν **30.000 μοναδικές τιμές**. Επομένως στην **έξοδο** έχουμε 5.000 * 1 = **5.000 εγγραφές** οι οποίες αντιστοιχούν σε **κόστος I/O** ίσο με **5.000 I/O**.

Βήμα 4)
Η λειτουργία 4 σε αυτό το σημείο μπορεί να εκτελεστεί στην **μνήμη** οπότε έχουμε **κόστος I/O** ίσο με **0 I/O**. Όμως μπορούμε να υπολογίσουμε τις **εγγραφές** που θα επιστραφούν και μπορούμε να τις βρούμε με παρόμοιο τρόπο όπως το Βήμα 2) του πλάνου Α. Οι εγγραφές που εξετάζουμε τώρα δεν είναι όλες οι εγγραφές της σχέσης αλλά οι 5.000 που μπαίνουν στην είσοδο του βήματος. Τα άλλα δύο στοιχεία, τα οποία είναι οι **μοναδικές τιμές** του πεδίου **Ηλικία** καθώς και το **εύρος** του **διαστήματος** της συνθήκης παραμένουν τα **ίδια**. Άρα τελικά θα **επιστραφούν** Εγγραφές Εισόδου / V(ΑΚΡΟΑΤΕΣ, Ηλικία) * Εύρος Διαστήματος = 5.000 / 40 * 4 = **500 εγγραφές**.

Αθροίζοντας τα κόστη των επιμέρους λειτουργιών του πλάνου Β παίρνουμε ως

συνολικό κόστος I/O για το **πλάνο B** το 10 + 5.000 + 5.000 + 0 = **10.010 I/O**

Γ)
Παρατηρούμε ότι το φυσικό **πλάνο A** έχει αρκετά **μικρότερο κόστος I/O** από το **πλάνο B**, σχεδόν 3 φορές μικρότερο, και αυτό είναι λογικό καθώς φιλτράρει τις εγγραφές του νωρίτερα και δεν τις κουβαλάει στα κοστοβόρα join των παραπάνω βημάτων. Επομένως θεωρούμε ότι το **καλύτερο φυσικό πλάνο** είναι το **πλάνο A**.