1 2 3 ΣΥΝ	

ΟΝΟΜΑ_____

Αρ. Μητρώου_____

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ
ΕΞΕΤΑΣΗ ΦΕΒΡΟΥΑΡΙΟΥ 2013

Ι. Βασιλείου
Τ. Σελλής

ΘΕΜΑ 1.- ΣΧΕΣΙΑΚΟ ΜΟΝΤΕΛΟ ΚΑΙ ΓΛΩΣΣΕΣ [30]

Α.-**[12]** Έστω ο παρακάτω πίνακας (relation) στο σχεσιακό μοντέλο για την οντότητα / σχέσιακό πίνακα SAILOR:

Sid	Sname	Rating	Age
18	jones	3	30
41	jonah	6	56
22	ahab	7	44
63	moby	*null*	15

Έχετε τρέξει SQL queries για να υπολογίσετε το μέσο rating (χρησιμοποιώντας το SQL AVERAGE), το άθροισμα των ratings (με το SUM) και τον αριθμό των ratings (με το COUNT).

```
SELECT    AVG (S.Rating) AS   AVERAGE
FROM      SAILORS S

SELECT    SUM (S.Rating)
FROM      SAILORS S

SELECT    COUNT (S.Rating)
FROM      SAILORS S
```

(α) **[4]** Αν διαιρέσετε το άθροισμα που υπολογίστηκε παραπάνω με τον αριθμό των ratings το αποτέλεσμα θα είναι το ίδιο με το μέσο rating??? Δώστε επιχειρήματα για την απάντησή σας.

Το αποτέλεσμα όταν χρησιμοποιειθούν τα COUNT και SUM θα είναι μικρότερο από το αποτέλεσμα του AVERAGE διότι υπάρχουν Tuples με τιμή στο rating ίσον με null. Οι συναρτήσεις για aggregates της SQL δεν λαμβάνουν υπόψη το null --- Εκτός του COUNT (η τιμή υπάρχει αλλά δεν την ξέρουμε)

(β) **[8]** Θεωρήστε το ερώτημα (query): "*Βρείτε τα ονόματα των sailors που έχουν υψηλότερο rating από όλους τους sailors με ηλικία μεγαλύτερη του 21"* Τα παρακάτω δύο SQL queries προσπαθούν να απαντήσουν στό ερώτημα. Είναι και τα δύο σωστά??? Επιχειρηματολογήστε για την απάντησή σας. Αν δεν υπολογίζουν το ίδιο αποτέλεσμα, τότε κάτω από ποιες συνθήκες το υπολογίζουν το ίδιο ???

```
SELECT  S.Sname
FROM    SAILORS   S
WHERE  NOT EXISTS       (SELECT  *
                         FROM     Sailors S2
                         WHERE  S2.Age < 21 AND S.Rating <= S2.Rating)
```

```
SELECT  S.Sname
FROM    SAILORS   S
WHERE  S.Rating > ANY (SELECT  S2.Rating
                         FROM     Sailors S2
                         WHERE  S2.Age < 21)
```

Μόνο το πρώτο query είναι σωστό. Το δεύτερο query επιστρέφει τα ονόματα των sailors με υψηλότερο Rating τουλάχιστον κάποιου sailor ηλικίας άνω των 21 ετών. Επιπλέον, σημειώστε ότι η απάντηση στο δεύτερο query δεν περιλαμβάνει αναγκαστικά την απάντηση στο πρώτο query. Για παράδειγμα, αν όλοι οι sailors είναι τουλάχιστον 21 ετών, τότε το δεύτερο query θα επιστρέψει το κενό σύνολο ενώ το πρώτο θα επιστρέψει όλους τους sailors.

Για να έχουμε την ίδια απάντηση πρέπει να ισχύουν μια εκ των δύο συνθηκών:

(α) Η relation Sailors είναι κενή, ή
(β) Υπάρχει τουλάχιστον ένας Sailor με ηλικία > 21 στην relation Sailors, και για κάθε sailor S, είτε ο S έχει rating ψηλότερο από όλους τους μικρότερους των 21 ετών ή ο S έχει rating που δεν υπερβαίνει αυτό όλων των Sailors με ηλικία κάτω των 21.

ΕΠΙΣΗΜΑΝΣΗ: ΟΛΕΣ ΟΙ ΑΠΑΝΤΗΣΕΙΣ που δόθηκαν στο ερώτημα βαθμολογήθηκαν ως σωστές εκτός από αυτές που δήλωναν και τα δύο queries ως σωστά (λόγω της ασάφειας του ερωτήματος).

Β.- **[18]** Παρακάτω έχουμε το σχήμα μιας βάσης δεδομένων για την διαχείριση των πωλήσεων μιας επιχείρησης. Να δημιουργήσετε τα κατάλληλα ερωτήματα σε γλώσσα SQL προκειμένου να απαντηθούν τα παρακάτω:

PRODUCT = (proNum, description, price)
CUSTOMER = (customerId, name, age)
SALES = (proNum, customerId, quantity, cost)

(a) **[4]** Βρείτε τα ονόματα των πελατών σε αύξουσα σειρά με βάση την ηλικία τους.

```
SELECT name
FROM CUSTOMER
ORDER BY age ASC;
```

(b) **[4]** Βρείτε τις συνολικές πωλήσεις ανά κωδικό προϊόντος.

```
SELECT proNum, sum(quantity)
FROM SALES
GROUP BY proNum;
```

(c) **[5]** Βρείτε τα ονόματα και το σύνολο αγορών εκείνων μόνο των πελατών που είχαν σύνολο αγορών πάνω από 1000€.

```
SELECT c.name, SUM(s.cost)
FROM CUSTOMER AS c, SALES AS s
WHERE c.customerId = s.customerId
GROUP BY c.customerId
HAVING SUM(s.cost)>1000;
```

(d) **[5]** Βρείτε όλες τις πωλήσεις που η αξία τους είναι πάνω από τον μέσο όρο.

```
SELECT s1.*
FROM SALES AS s1
WHERE s1.costs>(SELECT AVG(s2.costs)
                FROM SALES AS s2);
```

ΘΕΜΑ 2 – <u>ΦΥΣΙΚΟΣ ΣΧΕΔΙΑΣΜΟΣ– ΕΠΕΞΕΡΓΑΣΙΑ ΕΡΩΤΗΜΑΤΩΝ</u> [30]

Α.- [10] Έστω ένα B+-δένδρο τάξης με μέγιστο συντελεστή διακλάδωσης (δηλ. πόσα το πολύ παιδιά μπορεί να έχει) 5 για τους εσωτερικούς κόμβους, μέγιστο αριθμό ζευγών (*κλειδί, δείκτης σε εγγραφή*) 7 για τα φύλλα και που έχει 4 επίπεδα (συμπεριλαμβανομένου του επιπέδου της ρίζας και των φύλλων). Θεωρείστε ότι χρησιμοποιείται ως ευρετήριο πάνω στο γνώρισμα (πεδίο) διάταξης ενός διατεταγμένου αρχείου και ότι το γνώρισμα αυτό είναι και κλειδί. Το ευρετήριο είναι όσο ποιο *αδειανό* επιτρέπεται από τον ορισμό του B+-δένδρου.

(α) Πόσα blocks καταλαμβάνει το ευρετήριο; Εξηγείστε την απάντησή σας.

Αφού είναι μισογεμάτες οι σελίδες έχουμε 3 κλειδιά ανά εσωτ. Κόμβο και 4 ανά φύλλο.

*Συνεπώς 1 ρίζα + 3 παιδιά + 3*3 παιδιά αυτών + 3*3*3 φύλλα = 40 blocks το ευρετήριο.*

(β) Πόσα blocks καταλαμβάνει το αρχείο; Εξηγείστε την απάντησή σας.

*Αφού έχουμε 4 κλειδιά ανά φύλλο και κάθε εγγραφή καταλαμβάνει ένα block, το αρχείο είναι 27*4=108 blocks.*

Β.- [20] Έστω η παρακάτω σχεσιακή βάση που αφορά ομάδες και παίκτες.

ΟΜΑΔΑ (id-ομάδας, όνομα-ομάδας, αριθμός-κυπέλων, πόλη-εδρα)

ΠΑΙΚΤΗΣ (id-παίκτη , όνομα-παίκτη, id-ομάδας, χρόνια-εμπειρίας, θέση)

(α) **[6]** Θεωρείστε τη σχέση ΟΜΑΔΑ. Δώστε ένα παράδειγμα μιας SQL ερώτησης την οποία ένα ευρετήριο κατακερματισμού στο γνώρισμα (πεδίο) αριθμός-κυπέλων.

(i) θα την έκανε πιο γρήγορη

SELECT *id-ομάδας* ***FROM*** *ΟΜΑΔΑ* ***WHERE*** *αριθμός-κυπέλων = 5 γιατί εκμεταλλεύεται τον κατακερματισμό.*

(ii) θα την έκανε πιο αργή

SELECT *id-ομάδας* ***FROM*** *ΟΜΑΔΑ* ***WHERE*** *αριθμός-κυπέλων > 5 γιατί δεν μπορεί ο κατακερματισμός να βοηθήσει και γίνεται σειριακά αλλά πληρώνει το overhead ότι πρέπει να χρησιμοποιήσει το ευρετήριο που υπάρχει (υποθέτουμε ότι δεν υπάρχει βελτιστοποιητής).*

(iii) δε θα την επηρέαζε.

Οποιαδήποτε ερώτηση σε άλλο πεδίο.

(β) **[14]** Έστω ότι η Σχέση ΟΜΑΔΑ (R1) έχει 20.000 εγγραφές, η Σχέση ΠΑΙΚΤΗΣ (R2) έχει 45.000 εγγραφές, οι 25 εγγραφές του R1 χωρούν σε ένα μπλοκ και 30 εγγραφές του R2 χωρούν σε ένα μπλοκ.
Εκτιμήστε τον αριθμό των προσπελάσεων μπλοκ που απαιτούνται, χρησιμοποιώντας καθεμία από τις στρατηγικές συνδέσμου για το R1 ⋉ R2 (φυσικός σύνδεσμος –natural join)

- σύνδεσμος ένθετου βρόχου (nested loop join)
- σύνδεσμος μπλοκ ένθετου βρόχου (block nested loop join)
- σύνδεσμος συγχώνευσης (merge join)

Σύμφωνα με την εκφώνηση, η r1 χρειάζεται 800 μπλοκ και η r2 χρειάζεται 1500 μπλοκ για αποθήκευση.

Έστω ότι έχουμε M σελίδες διαθέσιμες στη μνήμη (buffer). Αν το M > 800, τότε χρησιμοποιούμε τις 800 σελίδες της μνήμης και κάνουμε ένα απλό σύνδεσμο ένθετου βρόχου με συνολικό κόστος (βέλτιστο) το 1500+800 μπλοκ προσβάσεις.

Σε περίπτωση όπου το M < 800.

- ***σύνδεσμος ένθετου βρόχου (nested loop join)***

*Με το r1 εξωτερική σχέση, απαιτούνται 20000 * 1500 + 800 = 30000800 προσβάσεις. Με το r2*

*εξωτερική σχέση, απαιτούνται 45000 * 800 +1500 = 36001500 προσβάσεις.*

- ***σύνδεσμος μπλοκ ένθετου βρόχου (block nested loop join)***

*Με το r1 εξωτερική σχέση, απαιτούνται 800 / (M-1) * 1500 +800. Με το r2 εξωτερική σχέση,*

*απαιτούνται 1500 / (M-1) * 800 +1500.*

- ***σύνδεσμος συγχώνευσης (merge join)***

*Απαιτούνται 1500 + 800 + B προσβάσεις, όπου B είναι το κόστος ταξινόμησης στο γνώρισμα της συνένωσης (B = 1500 * (2 log(m-1)(1500 / M)) + 800 * (2 log(m-1)(800 / M)))*

ΘΕΜΑ 3. -- <u>ΓΕΝΙΚΕΣ ΕΡΩΤΗΣΕΙΣ</u> - <u>ΜΟΝΤΕΛΛΟΠΟΙΗΣΗ - ΚΑΝΟΝΙΚΟΠΟΙΗΣΗ</u> [40]

Α. **<u>ΚΑΝΟΝΙΚΟΠΟΙΗΣΗ</u> [15]**

(α) **[7]** Αποδείξτε ότι ΚΑΘΕ σχέση με δύο γνωρίσματα (attributes) είναι σε κανονική μορφή BCNF (Boyce Codd Normal Form)---- **Υπόδειξη**: εξετάστε όλες τις περιπτώσεις ύπαρξης μη-τετριμμένων συναρτησιακών (λειτουργικών) εξαρτήσεων.

Έστω ότι τα δύο γνωρίσματα είναι το Α και το Β.
Σύμφωνα με τον ορισμό του BCNF, εξετάζουμε όλες τις περιπτώσεις.
1.- ΔΕΝ υπάρχουν μη-τετριμμένες FD. Τότε προφανώς έχουμε BCNF διότι μόνο οι μη-τετριμμένες FD μπορούν να καταστρατηγήσουν τη συνθήκη του ορισμού (παρενθετικά, το μοναδικό κλειδί είναι ο συνδυασμός ΑΒ.)
2.- Α → Β ισχύει, αλλά δεν ισχύει η Β → Α. Τότε το Α είναι το μοναδικό κλειδί και κάθε μη τετριμμένη FD περιέχει το Α στο αριστερό σκέλος, άρα έχουμε BCNF.
3.- Β → Α ισχύει, αλλά δεν ισχύει η Α → Β. Περίπτωση συμμετρική της 2.
4.- Α → Β ισχύει και Β → Α ισχύει. Τότε και το Α και το Β είναι κλειδιά – χωρίς να καταστρατηγείται η συνθήκη του ορισμού της BCNF.

(β) **[4]** Έστω ότι κάνουμε decomposition (αποσύνθεση) του Σχήματος R(A, B, C, D, E) σε 2 Σχήματα
R1(A, B, C) και R2(A, D, E). Επίσης υποθέτουμε ότι ισχύουν οι παρακάτω εξαρτήσεις

A→ BC
CD→ E
B→ D
E→A

Δείξτε ότι η αποσύνθεση είναι χωρίς απώλειες (lossless join decomposition)

Το Α είναι κλειδί για την R1 (από την πρώτη συνάρτηση).
Επίσης, το Α είναι η τομή (κοινό χαρακτηριστικό) μεταξύ R1 και R2. Αυτή είναι ικανή και αναγκαία συνθήκη για το lossless join decomposition.

(γ) **[4]** Δείξτε ότι η αποσύνθεση R1(A, B, C), R2(C, D, E) ΔΕΝ ΕΙΝΑΙ lossless join decomposition (**Υπόδειξη:** Δώστε ένα παράδειγμα ενός στιγμιότυπου του R όπου δεν ισχύει η ικανή και αναγκαία συνθήκη για lossless join decomposition)

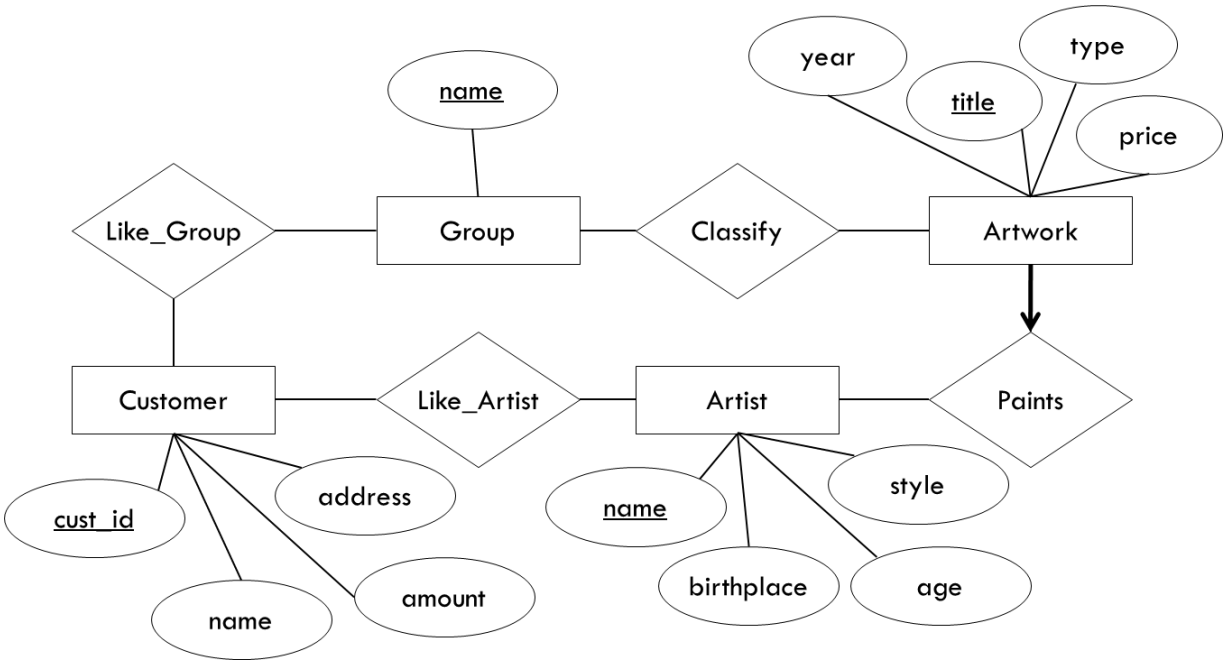
Αν πάρουμε τις προβολές του στιγμιότυπου του R στα χρακτηριστικά των R1, R2 και μετά τη συνένωση αυτών ΔΕΝ θα έχουμε πάλι το ίδιο στιγμιότυπο.

Το στιγμιότυπο της R:

A	***B***	***C***	***D***	***E***
4	*5*	*3*	*2*	*3*
4	*3*	*3*	*5*	*6*
2	*1*	*5*	*4*	*2*

Β. **ΜΟΝΤΕΛΟΠΟΙΗΣΗ** **[25]** Οι ΓΚΑΛΛΕΡΙ Τέχνης κρατούν πληροφορίες για Καλλιτέχνες (Artists), τα ονόματά τους (που είναι μοναδικά), τον τόπο γέννησής τους, ηλικία και στυλ (τεχνική) της τέχνης που δημιουργούν. Για κάθε έργο Τέχνης (Artwork) κρατιέται ο Καλλιτέχνης που το δημιούργησε (μόνο ένας ανά έργο), το έτος δημιουργίας, ο μοναδικός του τίτλος, το είδος του έργου τέχνης (π.χ., πίνακας, σχέδιο, φωτογραφία, γλυπτο, κλπ.) και η τιμή πώλησης. Τα έργα τέχνης κατατάσσονται σε ομάδες (Groups) διαφόρων ειδών, για παράδειγμα, πορτραίτα, φύση, έργα Πικάσσο, έργα του 19^{ου} αιώνα, κλπ. Ένα έργο τέχνης μπορεί να ανήκει σε περισσότερα του ενός group. Κάθε group έχει ένα όνομα (όπως στο παράδειγμα) που το περιγράφει. Τέλος οι Γκαλλερί κρατούν πληροφορίες για πελάτες (Customers). Για κάθε πελάτη, το μοναδικό του όνομα, διεύθυνση, συνολικό ποσό που έχει ξοδέψει στη Γκαλλερί και τις προτιμήσεις που έχει και σε Καλλιτέχνες και σε group έργων Τέχνης.

(α) **[15]** Κατασκευάστε ένα διάγραμμα οντότητας-σχέσης (entity-relationship). Τεκμηριώστε όλες τις υποθέσεις που κάνετε για τους περιορισμούς απεικόνισης.



(β) **[10]** Κατασκευάστε την αντίστοιχη σχεσιακή βάση δεδομένων

```
CREATE TABLE Like Group (
        name CHAR(20),
                cust name CHAR(20),
                PRIMARY KEY (name, cust_name),
                FOREIGN KEY (name)          REFERENCES Group,
                FOREIGN KEY (cust name)     REFERENCES Customer)

CREATE TABLE Like Artist (
        name CHAR(20),
        cust name CHAR(20),
        PRIMARY KEY (name, cust name),
        FOREIGN KEY (name) REFERENCES Artist,
        FOREIGN KEY (cust name) REFERENCES Customer)

CREATE TABLE Artwork Paints(
        title CHAR(20),
        artist name CHAR(20),
        type CHAR(20),
        price INTEGER,
        year INTEGER,
        PRIMARY KEY (title),
        FOREIGN KEY (artist name)
                REFERENCES Artist

CREATE TABLE Classify (
         title CHAR(20),
                name CHAR(20),
                PRIMARY KEY (title, name),
                FOREIGN KEY (title) REFERENCES Artwork_Paints,
                FOREIGN KEY (name) REFERENCES Group )
```

ΟΙ ΟΝΤΟΤΗΤΕΣ ΜΕΤΑΦΡΑΖΟΝΤΑΙ ΣΕ ΣΧΕΣΕΙΣ (Artist, Group, Customer, Artwork) με τα αντίστοιχα χαρακτηριστικά