2ο Σύνολο Ασκήσεων

Οι βαθμοί θα ανακοινωθούν αύριο μαζί με τους βαθμούς της προγραμματιστικής άσκησης
Τα αστεράκια δείχνουν τον εκτιμώμενο βαθμό δυσκολίας
(*) εύκολο (**) μέτριο (***) δύσκολο

Σχεσιακή Άλγεβρα

Άσκηση 1(α) (σχεσιακή άλγεβρα)

USER(USER_NAME, GENDER, NATIONALITY, DATE_OF_BIRTH)
MOVIE(MOVIE_ID, TITLE, YEAR, COUNTRY)
MOVIE_GENRE(MOVIE_ID, GENRE)
FOLLOWS(USER1, USER2)
LIKES(USER, MOVIE_GENRE)

(α) (**) Τα συμμετρικά ζευγάρια χρηστών, δηλαδή τα ζευγάρια που

- ο user1 ακολουθεί τον user2 και
- ο user2 ακολουθεί τον user1

Κάθε ζευγάρι πρέπει να εμφανίζεται στο αποτέλεσμα *μόνο μια φορά* (δηλαδή, αν εμφανίζεται το ζευγάρι (user1, user2) να μην εμφανίζεται το ζευγάρι (user2, user1).

Άσκηση 1(β) (σχεσιακή άλγεβρα)

USER(USER_NAME, GENDER, **NATIONALITY**, DATE_OF_BIRTH)
MOVIE(MOVIE_ID, **TITLE, YEAR**, COUNTRY)
MOVIE_GENRE(MOVIE_ID, GENRE)
FOLLOWS(USER1, USER2)
LIKES(USER, MOVIE_GENRE)

(β) (*) Για τους χρήστες με Ελληνική εθνικότητα, τον τίτλο όλων των ταινιών του 2010 που έχουν ένα είδος που τους αρέσει.

Άσκηση 1(γ) (σχεσιακή άλγεβρα)

USER(USER_NAME, GENDER, NATIONALITY, DATE_OF_BIRTH)
MOVIE(MOVIE_ID, TITLE, YEAR, COUNTRY)
MOVIE_GENRE(MOVIE_ID, GENRE)
FOLLOWS(USER1, USER2)
LIKES(USER, MOVIE_GENRE)

(γ) (***) Το χρήστη που του αρέσουν ***ακριβώς*** τα ίδια είδη ταινιών με το χρήστη maria.

Πρέπει να βρούμε τους χρήστες που (Α) τους αρέσουν ΟΛΑ τα είδη που αρέσουν στη Μαρία ΚΑΙ (Β) τους αρέσουν μόνο αυτά - δηλαδή, όλα τα είδη που τους αρέσουν είναι είδη που αρέσουν και στη Μαρία

Σημείωση: Η διαίρεση με τα είδη που αρέσουν στη Μαρία μας δίνει μόνο το (Α)
Αλλά μπορούμε να χρησιμοποιήσουμε την ιδέα που χρησιμοποιήσαμε για την ισοδύναμη έκφρασή της

Άσκηση 1(δ) (σχεσιακή άλγεβρα)

USER(USER_NAME, GENDER, NATIONALITY, DATE_OF_BIRTH)
MOVIE(MOVIE_ID, TITLE, YEAR, COUNTRY)
MOVIE_GENRE(MOVIE_ID, GENRE)
FOLLOWS(USER1, USER2)
LIKES(USER, MOVIE_GENRE)

(δ) (**) Το χρήστη που έχει διαφορετικές προτιμήσεις από τους χρήστες που ακολουθεί, δηλαδή, το χρήστη που κανένα από τα είδη ταινιών που του αρέσουν δεν αρέσει σε κάποιον (έστω έναν) χρήστη που ακολουθεί

Αφαίρεση
ΟΛΟΥΣ – ΑΥΤΟΥΣ_ΠΟΥ_ΕΧΟΥΝ_ΕΣΤΩ_ΚΑΙ_ΕΝΑ_ΚΟΙΝΟ_ΕΙΔΟΣ

Σχεσιακός Λογισμός

Άσκηση 2 (σχεσιακός λογισμός)

USER(USER_NAME, GENDER, NATIONALITY, DATE_OF_BIRTH)
MOVIE(MOVIE_ID, TITLE, YEAR, COUNTRY)
MOVIE_GENRE(MOVIE_ID, GENRE)
FOLLOWS(USER1, USER2)
LIKES(USER, MOVIE_GENRE)

(α) (*) Τα συμμετρικά ζευγάρια χρηστών, δηλαδή τα ζευγάρια που ο user1 ακολουθεί τον user2 και ο user2 ακολουθεί τον user1
Κάθε ζευγάρι πρέπει να εμφανίζεται στο αποτέλεσμα μόνο μια φορά (δηλαδή, αν εμφανίζεται το ζευγάρι (user1, user2) να μην εμφανίζεται το ζευγάρι (user2, user1).

(β) (**) Για τους χρήστες με Ελληνική εθνικότητα, τον τίτλο όλων των ταινιών του 2010 που έχουν ένα είδος που τους αρέσει.

Άσκηση 2 (σχεσιακός λογισμός)

USER(USER_NAME, GENDER, NATIONALITY, DATE_OF_BIRTH)
MOVIE(MOVIE_ID, TITLE, YEAR, COUNTRY)
MOVIE_GENRE(MOVIE_ID, GENRE)
FOLLOWS(USER1, USER2)
LIKES(USER, MOVIE_GENRE)

(γ) (***) Το χρήστη που του αρέσουν *ακριβώς* τα ίδια είδη ταινιών με το χρήστη maria.

Τους χρήστες που
(Α) όλα τα είδη που τους αρέσουν είναι είδη που αρέσουν και στη Μαρία ΚΑΙ
(Β) όλα τα είδη που αρέσουν στη Μαρία είναι είδη που αρέσουν και σε αυτούς (με χρήση καθολικού ποσοδείκτη)
Τους χρήστες που
(Α) δεν υπάρχει είδος που να αρέσει σε αυτούς και να μην αρέσει στη Μαρία και
(Β) δεν υπάρχει είδος που να αρέσει στη Μαρία και να μην αρέσει σε αυτούς (με χρήση υπαρξιακού ποσοδείκτη)

(δ) (**) Το χρήστη που έχει διαφορετικές προτιμήσεις από τους χρήστες που ακολουθεί, δηλαδή, το χρήστη που κανένα από τα είδη ταινιών που του αρέσουν δεν αρέσει σε κάποιον (έστω ένα) χρήστη που ακολουθεί

SQL

Άσκηση 3α(i)(*) (SQL)

USER(USER_NAME, GENDER, NATIONALITY, DATE_OF_BIRTH)
MOVIE(MOVIE_ID, TITLE, YEAR, COUNTRY)
MOVIE_GENRE(MOVIE_ID, GENRE)
FOLLOWS(USER1, USER2)
LIKES(USER, MOVIE_GENRE)

```
CREATE TABLE IF NOT EXISTS `user` (
  `user_name` varchar(500) NOT NULL,
  `genre` varchar(40) NOT NULL,
  `NATIONALITY` varchar(500) NOT NULL,
  `DATE_OF_BIRTH` date NOT NULL,
  PRIMARY KEY (`user_name`)
) ENGINE=InnoDB;

CREATE TABLE IF NOT EXISTS `movie` (
  `movie_id` int(10) NOT NULL,
  `title` varchar(40) DEFAULT NULL,
  `year` int(4) DEFAULT NULL,
  `country` varchar(40) DEFAULT NULL,
  PRIMARY KEY (`movie_id`)
) ENGINE=InnoDB;

CREATE TABLE IF NOT EXISTS `movie_genre` (
  `movie_id` int(10) NOT NULL,
  `genre` varchar(40) NOT NULL,
  PRIMARY KEY (`movie_id`,`genre`),
  INDEX `gn` (`genre`),    [συνώνυμο KEY `gn`ˆgenre`,]
FOREIGN KEY (`movie_id`) REFERENCES `movie` (`movie_id`);
) ENGINE=InnoDB;
```

Οι NOT NULL και DEFAULT είναι μόνο ενδεικτικές

Άσκηση 3α(i) (SQL)

USER(USER_NAME, GENDER, NATIONALITY, DATE_OF_BIRTH)
MOVIE(MOVIE_ID, TITLE, YEAR, COUNTRY)
MOVIE_GENRE(MOVIE_ID, GENRE)
FOLLOWS(USER1, USER2)
LIKES(USER, MOVIE_GENRE)

```
CREATE TABLE IF NOT EXISTS `follows` (
  `user1` varchar(500) NOT NULL,
  `user2` varchar(500) NOT NULL,
  PRIMARY KEY (`user1`,`user2`),
  FOREIGN KEY (`user1`) REFERENCES `user` (`USER_NAME`),
  FOREIGN KEY (`user2`) REFERENCES `user` (`USER_NAME`);
) ENGINE=InnoDB;

CREATE TABLE IF NOT EXISTS `likes` (
  `user` varchar(40) NOT NULL,
  `movie_genre` varchar(40) NOT NULL,
  PRIMARY KEY (`user`,`movie_genre`),
  FOREIGN KEY (`user`) REFERENCES `user` (`USER_NAME`),
  FOREIGN KEY (`movie_genre`) REFERENCES `movie_genre` (`genre`);
) ENGINE=InnoDB;
```

Μπορείτε και να προσθέσετε constraints αργότερα

```
ALTER TABLE `follows`
  ADD CONSTRAINT `follows_ibfk_1` FOREIGN KEY (`user1`) REFERENCES `user` (`USER_NAME`),
  ADD CONSTRAINT `follows_ibfk_2` FOREIGN KEY (`user2`) REFERENCES `user` (`USER_NAME`);
```

Άσκηση 3β(i) (SQL)

USER(USER_NAME, GENDER, NATIONALITY, DATE_OF_BIRTH)
MOVIE(MOVIE_ID, TITLE, YEAR, COUNTRY)
MOVIE_GENRE(MOVIE_ID, GENRE)
FOLLOWS(USER1, USER2)
LIKES(USER, MOVIE_GENRE)

(i) (**) Για κάθε χρήστη, τον αριθμό των χρηστών που αυτός ακολουθεί, *τον αριθμό των χρηστών που τον ακολουθούν* και τον αριθμό των χρηστών που ακολουθεί οι οποίοι των ακολουθούν (αν θέλετε δώστε 3 διαφορετικές ερωτήσεις).

```
SELECT user_name , COUNT(*)
FROM follows
GROUP BY user1
```
(δεν βγάζει 0 για τους χρήστες που δεν τους ακολουθεί κανείς)

```
(SELECT user_name , COUNT(*)
FROM follows
GROUP BY user1)
UNION
(SELECT user_name as U , 0
FROM  user
WHERE U NOT IN  (SELECT user1
                 FROM  follows))
```

Άσκηση 3β(i) (SQL)

USER(USER_NAME, GENDER, NATIONALITY, DATE_OF_BIRTH)
MOVIE(MOVIE_ID, TITLE, YEAR, COUNTRY)
MOVIE_GENRE(MOVIE_ID, GENRE)
FOLLOWS(USER1, USER2)
LIKES(USER, MOVIE_GENRE)

(i) (**) Για κάθε χρήστη, τον αριθμό των χρηστών που αυτός ακολουθεί, τον αριθμό των χρηστών που τον ακολουθούν και **τον αριθμό των χρηστών που ακολουθεί οι οποίοι τον ακολουθούν** (αν θέλετε δώστε 3 διαφορετικές ερωτήσεις).

```
SELECT F1.user1 , COUNT(*)
FROM follows as F1
WHERE F1.user1 IN  (SELECT F2.user2
                    FROM follows as F2
                    WHERE F2.user1 = F1.user2)
GROUP BY F1.user1
```

Με EXISTS? αντί IN

```
WHERE EXISTS (SELECT *
              FROM follows as F2
              WHERE  F2.user1  = F1.user2
                 AND F2.user2 =  F1.user1)
```

Άσκηση 3β(ii) (SQL)

USER(USER_NAME, GENDER, NATIONALITY, DATE_OF_BIRTH)
MOVIE(MOVIE_ID, TITLE, **YEAR**, COUNTRY)
MOVIE_GENRE(MOVIE_ID, GENRE)
FOLLOWS(USER1, USER2)
LIKES(USER, MOVIE_GENRE)

(ii) (*) Για το χρήστη με όνομα maria τον τίτλο όλων των ταινιών του 2010 που έχουν ένα είδος που της αρέσει.

SELECT DISTINCT title
FROM movie M, movie_genre G, likes L
WHERE M.year = 2010 **AND** L.user = 'maria' **AND** L.movie_genre = G.genre **AND** G.movie_id = M.movie_id

Άσκηση 3β(iv) (SQL)

USER(USER_NAME, GENDER, NATIONALITY, DATE_OF_BIRTH)
MOVIE(MOVIE_ID, TITLE, YEAR, COUNTRY)
MOVIE_GENRE(MOVIE_ID, GENRE)
FOLLOWS(USER1, USER2)
LIKES(USER, MOVIE_GENRE)

(iv) Για το χρήστη maria, το είδος ταινίας που αρέσει στους περισσότερους από τους χρήστες που ακολουθεί.

Για κάθε είδος, σε πόσους φίλους της Μαρίας αρέσει και να κρατήσουμε αυτό με το maxcount

```
SELECT movie_genre
FROM likes
WHERE user IN (SELECT F.user2
               FROM follows F
               WHERE F.user1 = 'maria')
GROUP BY movie_genre
HAVING COUNT(*) = (SELECT MAX()
                   FROM  (SELECT COUNT(*)
                          FROM likes L
                          WHERE L.user IN  (SELECT F1.user2
                                            FROM follows F1
                                            WHERE F1.user1 = 'maria')
                          GROUP BY movie_genre)  AS mg)
```

```
FROM likes as L, follows as F
WHERE F.user1 = 'maria‘ and
AND L.user = F.user2)
```

Επιτρέπονται subqueries στο FROM υποχρεωτικά χρήση AS

Επειδή,
Δεν επιτρέπεται max(count())
Εναλλακτικά,
ORDER BY
LIMIT 1

Άσκηση 3β(iii) (SQL)

LIKES(USER, MOVIE_GENRE)

(iii) (****) Τα ζεύγη χρηστών που τους αρέσουν *ακριβώς* τα ίδια είδη ταινιών.

Παίρνουμε όλα τα ζεύγη (pairs)
user1 genre1 user2 genre2

```
SELECT *
FROM   (SELECT distinct a.user AS user1, b.user as user2
        FROM likes a JOIN likes b on a.movie_genre = b.movie_genre) AS pair
```

(A) Δεν πρέπει να υπάρχει είδος που να αρέσει στον user1 για το οποίο δεν υπάρχει ίδιο είδος που να αρέσει στο user2

Άσκηση 3β(iii) (SQL)

LIKES(<u>USER</u>, <u>MOVIE GENRE</u>)

```
SELECT *
FROM   (SELECT distinct a.user AS user1, b.user as user2
        FROM likes a JOIN likes b on a.movie_genre = b.movie_genre) AS pair
WHERE pair.user1 < pair.user2
AND NOT EXISTS (SELECT *
                FROM likes l1
                 where l1.user = pair.user1
                     AND NOT EXISTS (SELECT *
                                     FROM likes l2
                                     where l2.user = pair.user2
                                     AND l2.movie_genre = l1.movie_genre))
```

AND *Δεν πρέπει να υπάρχει είδος που να αρέσει στον user2 για το οποίο δεν υπάρχει ίδιο είδος που να αρέσει στο user1*

Άσκηση 3β(iii) (SQL)

```
SELECT *
FROM   (SELECT distinct a.user AS user1, b.user as user2
        FROM likes a JOIN likes b on a.movie_genre = b.movie_genre) AS pair
WHERE pair.user1 < pair.user2
AND NOT EXISTS (SELECT *
                FROM likes l1
                WHERE l1.user = pair.user1
                    AND NOT EXISTS (SELECT *
                                    FROM likes l2
                                    WHERE l2.user = pair.user2
                                        AND l2.movie_genre = l1.movie_genre))
AND NOT EXISTS (SELECT *
                FROM likes l1
                WHERE l1.user = pair.user2
                    AND NOT EXISTS (SELECT *
                                    FROM likes l2
                                    WHERE l2.user = pair.user1
                                        AND l2.movie_genre = l1.movie_genre))
```

Άσκηση 3γ(i) (SQL)

USER(USER_NAME, GENDER, NATIONALITY, DATE_OF_BIRTH)
MOVIE(MOVIE_ID, TITLE, YEAR, COUNTRY)
MOVIE_GENRE(MOVIE_ID, GENRE)
FOLLOWS(USER1, USER2)
LIKES(USER, MOVIE_GENRE)

(i) (*) Ορίστε *μια όψη η οποία να είναι τροποποιήσιμη* (updatable). Εξηγείστε. Δώστε ένα παράδειγμα τροποποίησης αυτής της όψης και δείξτε πως αλλάζει *το περιεχόμενο της βασικής σχέσης*.

```
CREATE VIEW userInfo AS SELECT user_name, gender, nationality
FROM user;

UPDATE userInfo SET nationality = 'Greece' WHERE nationality = 'UK';

SELECT * FROM user;
```

Άσκηση 3γ(ii) (SQL)

USER(USER_NAME, GENDER, NATIONALITY, DATE_OF_BIRTH)
MOVIE(MOVIE_ID, TITLE, YEAR, COUNTRY)
MOVIE_GENRE(MOVIE_ID, GENRE)
FOLLOWS(USER1, USER2)
LIKES(USER, MOVIE_GENRE)

(ii) (**) Ορίστε μια όψη FOLLOW_LIKES(USER, MOVIE_GENRE) που να περιέχει για κάθε χρήστη (USER) τα είδη (MOVIE_GENRE) των ταινιών που αρέσουν σε τουλάχιστον έναν από τους χρήστες που ακολουθεί.

```
CREATE VIEW follow_likes AS
    SELECT DISTINCT F.user1, L.movie_genre
    FROM likes L, follows  F
    WHERE F.user2 = L.user
```

Αν δεν ακολουθεί κανένα, τότε δεν εμφανίζεται - μπορούμε να κάνουμε ότι στο ερώτημα3(α)

Δείξτε το περιεχόμενο αυτής της όψης, χρησιμοποιώντας το select * from FOLLOW_LIKES

Στη συνέχεια, εισάγετε μια πλειάδα στη σχέση FOLLOWS που να οδηγεί σε τροποποίηση του περιεχομένου της όψης.

Εισαγωγή μιας πλειάδας (user1, user2) που (1) δεν υπάρχει στο FOLLOWS και (2) ο user2 έχει ένα είδος ταινίας που δεν αρέσει σε κανέναν από τους χρήστες που ακολουθεί ο user1

Άσκηση 3δ(i) (SQL)

USER(USER_NAME, GENDER, NATIONALITY, DATE_OF_BIRTH)
MOVIE(MOVIE_ID, TITLE, YEAR, COUNTRY)
MOVIE_GENRE(MOVIE_ID, GENRE)
FOLLOWS(USER1, USER2)
LIKES(USER, MOVIE_GENRE)

(i) (**) Για το χρήστη maria, προσθέστε μια σχέση FOLLOWS προς όλους τους χρήστες που την ακολουθούν, δηλαδή για κάθε πλειάδα (u, maria) της FOLLOWS προσθέστε μια πλειάδα (maria, u) (αν αυτή δεν υπάρχει).

```
INSERT INTO follows(user1, user2)
        SELECT follows.user2, follows.user1
        FROM follows
        WHERE follows.user2 = 'maria'  AND
                follows.user1 NOT IN (SELECT follows.user2
                                      FROM follows
                                      WHERE follows.user1='maria');
```

Άσκηση 3δ(ii) (SQL)

USER(USER_NAME, GENDER, NATIONALITY, DATE_OF_BIRTH)
MOVIE(MOVIE_ID, TITLE, YEAR, COUNTRY)
MOVIE_GENRE(MOVIE_ID, GENRE)
FOLLOWS(USER1, USER2)
LIKES(USER, MOVIE_GENRE)

(ii) (*) Τροποποιείστε το έτος γέννησης του χρήστη maria σε 1/12/1993.

```
UPDATE user
SET date_of_birth = '01/12/1993'
WHERE user_name = 'maria';
```

Άσκηση 3δ(iii) (SQL)

USER(USER_NAME, GENDER, NATIONALITY, DATE_OF_BIRTH)
MOVIE(MOVIE_ID, TITLE, YEAR, COUNTRY)
MOVIE_GENRE(MOVIE_ID, GENRE)
FOLLOWS(USER1, USER2)
LIKES(USER, MOVIE_GENRE)

(iii) (*) Διαγράψτε από τη σχέση FOLLOWS όλα τα μη συμμετρικά ζευγάρια..

```
DELETE FROM follows as F1
WHERE (F1.user2, F1.user1) NOT IN (SELECT *
                                   FROM follows);
```

```
DELETE FROM follows F1
WHERE NOT EXISTS (SELECT *
                  FROM follows F2
                  WHERE F1.user1 = F2.user2 AND F1.user2 = F2.user1);
```

Λογικός Σχεδιασμός

Άσκηση 4(α)

Σχεσιακό σχήμα R = (P, Q, S, T, U, V) και σύνολο συναρτησιακών εξαρτήσεων F = {PQ → S, QS → P, QT → U, P → T, PS → Q, U → V}.
(α) (*) Η F είναι ή όχι ελάχιστο κάλυμμα. Εξηγείστε την απάντηση σας.

Πρέπει να εξετάσουμε
(1) Αν υπάρχει περιττό γνώρισμα στο αριστερό μέρος και (2) αν κάποια από τις εξαρτήσεις που προκύπτουν από το (1) είναι περιττές

Για το (1) πρέπει να εξετάσουμε τις PQ → S, QS → P, QT → U, PS → Q
Για παράδειγμα για την PQ → S
Υπολογίζουμε P^+ και Q^+ με την F
Αν S ∈ P^+, τότε το Q είναι περιττό και η PQ → S μπορεί να αντικατασταθεί από την P → S
(όμοια, Αν S ∈ Q^+, τότε το P είναι περιττό και η PQ → S μπορεί να αντικατασταθεί από την Q→ S)
Προκύπτει ότι κανένα γνώρισμα δεν είναι περιττό
Για το (2) πρέπει να εξετάσουμε τις PQ → S, QS → P, QT → U, P → T, PS → Q, U → V.
Για παράδειγμα για την PQ → S
Υπολογίζουμε PQ^+ με F' = {QS → P, QT → U, P → T, PS → Q, U → V}.
Αν S ∈ PQ^+, τότε η PQ → S είναι περιττή
Προκύπτει ότι καμία εξάρτηση δεν είναι περιττή

Άσκηση 4(β)

Σχεσιακό σχήμα R = (P, Q, S, T, U, V) και σύνολο συναρτησιακών εξαρτήσεων F = {PQ $\rightarrow$ S, QS $\rightarrow$ P, QT $\rightarrow$ U, P $\rightarrow$ T, PS $\rightarrow$ Q, U $\rightarrow$ V}.

(β) Για τις παρακάτω διασπάσεις δείξτε αν είναι ή όχι χωρίς απώλειες στη συνένωση και αν διατηρούν ή όχι τις εξαρτήσεις;

(i) Διάσπαση σε R1 = {P, Q}, R2 = {P, S}, R3 = {P, Q, T, U}, R4 = {U, V}

(ii) Διάσπαση σε R1 = {P, Q, S}, R2 = {Q, S, T, U, V}, R3 = {Q, T, V}

(***) Απώλειες στη συνένωση

Διασπάμε διαδοχικά την R σε δύο σχέσεις, προσπαθώντας σε κάθε βήμα τα γνωρίσματα στην τομή των σχέσεων να αποτελούν κλειδί

– αν αυτό είναι δυνατόν, τότε απόδειξη ότι χωρίς απώλειες
– αν αυτό δεν είναι δυνατόν, τότε ψάχνουμε για αντιπαράδειγμα

Ερώτημα (ii) χωρίς απώλειες

Ερώτημα (i) με απώλειες

Άσκηση 4(β)

Σχεσιακό σχήμα R = (P, Q, S, T, U, V) και σύνολο συναρτησιακών εξαρτήσεων F = {PQ $\rightarrow$ S, QS $\rightarrow$ P, QT $\rightarrow$ U, P $\rightarrow$ T, PS $\rightarrow$ Q, U $\rightarrow$ V}.

(β) Για τις παρακάτω διασπάσεις δείξτε αν είναι ή όχι χωρίς απώλειες στη συνένωση και αν διατηρούν ή όχι τις εξαρτήσεις;

(i) Διάσπαση σε R1 = {P, Q}, R2 = {P, S}, R3 = {P, Q, T, U}, R4 = {U, V}

(ii) Διάσπαση σε R1 = {P, Q, S}, R2 = {Q, S, T, U, V}, R3 = {Q, T, V}

(*) Διατήρηση των εξαρτήσεων

Θα πρέπει να εξετάσουμε για κάθε συναρτησιακή εξάρτηση στο F αν ανήκει στην κλειστότητα της ένωσης των προβολών της F^+

Για το (i) δε διατηρούνται οι QS $\rightarrow$ P, PQ $\rightarrow$ S και QT $\rightarrow$ U και για το (ii) η P $\rightarrow$ T

Άσκηση 4(γ)

(γ) (**) Σχεσιακό σχήμα R = (P, Q, S, T, U, V) και σύνολο συναρτησιακών εξαρτήσεων F = {PQ → S, QS → P, QT → U, P → T, PS → Q, U → V}.

Θεωρείστε τα σχεσιακά σχήματα R1 = {P, Q, S}, R2 = {P, Q, S, U, V} και R3 = {P, Q, S, T}
(i) Δείξτε αν η R1 είναι ή όχι σε 3NF.
(ii) Δείξτε αν η R1 είναι ή όχι σε BCNF.
(iii) Υπολογίστε τα υποψήφια κλειδιά της R2.
(iv) Δείξτε αν η R2 είναι ή όχι σε BCNF.

Για τα (i) – (ii) Υπολογίζουμε ποιες συναρτησιακές εξαρτήσεις ισχύουν στην R1 (δηλαδή, υπολογίζουμε την προβολή τις F+ στην R1)
Βρίσκουμε τα υποψήφια κλειδιά
Για BCNF εξετάζουμε αν στο αριστερό μέρος κάθε συναρτησιακής εξάρτησης υπάρχει υπερκλειδί
Για 3NF, αρκεί είτε το παραπάνω είτε για τις συναρτησιακές εξαρτήσεις που δεν ισχύει το παραπάνω να εξετάσουμε αν στα δεξιά πρωτεύον γνώρισμα
Η R1 ίναι σε BCNF άρα και σε 3NF

Ερωτήσεις;