

Φυσική σύνδεση και άλλες
εντολές

Περισσότεροι πίνακες

- Θυμίζουμε:

Φυσική σύνδεση

Customer $><$ Borrow

- Φέρνουμε δίπλα σε κάθε δάνειο τα στοιχεία του πελάτη που το έχει
Πώς?
- Από το καρτεσιανό γινόμενο («όλα-με-όλα») κρατάμε μόνο όσες γραμμές έχουν **ίδια τιμή στο κοινό πεδίο**

Φυσική σύνδεση

- Δεν υπάρχει κάποια εντολή που κάνει φυσική σύνδεση
- Εφαρμόζουμε το μηχανισμό με τον οποίο γίνεται η σύνδεση
- Πίνακες: FROM Customer, Borrow
- Συνθήκη: WHERE Customer.Cust-name = Borrow.Cust-name
- Άρα: Αρ.δανείου, όνομα πελάτη, πόλη πελάτη

ΦΥΣΙΚΗ
ΣΥΝΔΕΣΗ



```
SELECT loan-no, cust-name, cust-city FROM Customer, Borrow
WHERE Customer.Cust-name = Borrow.Cust-name AND
```

Φυσική σύνδεση

- Τα υπόλοιπα στοιχεία της εντολής SELECT συνεχίζουν να βρίσκονται στην ίδια θέση με την ίδια λειτουργία
- Η συνθήκη σύνδεσης πρέπει να υπάρχει πάντα
- Οι υπόλοιπες συνθήκες συνδέονται με αυτήν με AND
- Μπορεί να έχουμε και περισσότερους από δύο πίνακες – διαδοχικές συνδέσεις με τον ίδιο τρόπο

Παραδείγματα

- Αρ. λογαριασμού σε καταστήματα της Αθήνας
SELECT acc-no FROM Deposit, branch
WHERE branch.br-name = deposit.br-name AND
Br-city = 'Αθήνα'
- Αρ. δανείου και διεύθυνση πελάτη για πελάτες με δάνεια πάνω από 50000
SELECT loan-no, address FROM Borrow, customer
WHERE Borrow.cust-name = customer.cust-name AND
Amount > 50000

Παραδείγματα

- Αρ. λογαριασμού και όνομα πελάτη με λογαριασμό στην πόλη του
SELECT acc-no, cust-name FROM Deposit, branch, customer
WHERE branch.br-name = deposit.br-name AND customer.cust-name =
deposit.cust-name AND
Br-city = cust-city
- Ονόματα δανειοληπτών που έχουν τους λογαριασμούς 961 και 1003 και τα
αντίστοιχα καταστήματα και πόλεις των λογαριασμών
SELECT cust-name, branch.br-name, br-city FROM Borrow, branch
WHERE branch.br-name = borrow.br-name AND
(Acc-no = 961 OR acc-no = 1003)

Εμφωλευμένα ερωτήματα

- Πολλές φορές δεν χρειάζεται πραγματική η σύνδεση
- Αρ. λογαριασμού σε καταστήματα της Αθήνας
 - Χρειαζόμαστε μόνο να ξέρουμε ποια καταστήματα είναι στην Αθήνα

```
SELECT br-name FROM branch  
WHERE Br-city = 'Αθήνα'
```
 - Παράγεται μια λίστα ονομάτων που μπορούμε να χρησιμοποιήσουμε
 - Συνεπώς

```
SELECT acc-no FROM Deposit  
WHERE br-name IN (SELECT br-name FROM branch WHERE  
Br-city = 'Αθήνα')
```
- Και πάλι πρέπει να υπάρχει συμβατότητα

Πράξεις συνόλων

- Τομή: INTERSECT
- Ένωση: UNION
- Διαφορά: MINUS

- Ισχύουν οι προϋποθέσεις συμβατότητας
 - Ίδιο πλήθος χαρακτηριστικών
 - Ίδιος τύπος δεδομένων

Παραδείγματα

- Ονόματα πελατών με δάνειο και λογαριασμό ταυτόχρονα στο ίδιο κατάστημα
(SELECT cust-name FROM Deposit)
INTERSECT
(SELECT cust-name FROM Borrow)
- Ονόματα πελατών με δάνειο και λογαριασμό ταυτόχρονα **στο ίδιο κατάστημα**
 - Ψάχνουμε και το όνομα και το κατάστημα, σαν ζευγάρι
(SELECT cust-name, br-name FROM Deposit)
INTERSECT
(SELECT cust-name, br-name FROM Borrow)

Παραδείγματα

- Ονόματα πελατών είτε με δάνειο είτε με λογαριασμό
(SELECT cust-name FROM Deposit)
UNION
(SELECT cust-name FROM Borrow)
- Ονόματα πελατών με δάνειο αλλά χωρίς λογαριασμό
(SELECT cust-name FROM Borrow)
MINUS
(SELECT cust-name FROM Deposit)

Παραδείγματα

- Έργα που προμηθεύονται από τον προμηθευτή S1 αλλά όχι από κάποιον άλλο

SELECT J# FROM SPJ

MINUS

(SELECT J# FROM SPJ WHERE S# <> 'S1')

- Βρίσκουμε τα έργα που παίρνουν **και** από κάποιον άλλο
- Αφαιρούμε αυτά τα έργα από την πλήρη λίστα έργων

Παραδείγματα

- Ζεύγη S# και P# ώστε ο προμηθευτής να ΜΗΝ προμηθεύει το υλικό
 - Και πάλι πρέπει να βρούμε όλα τα ζευγάρια που ο προμηθευτής προμηθεύει το υλικό και να τα αφαιρέσουμε από τα δυνατά ζευγάρια
 - Ποια είναι αυτά τα ζευγάρια? Είναι αυτά που βρίσκω στο SPJ

```
SELECT S#, P# FROM SPJ
```

- Μετά αφαιρώ από όλα τα πιθανά (καρτεσιανό γινόμενο) ζεύγη (S#, P#)

```
SELECT S#, P# FROM S, P
```

```
MINUS
```

```
S#, P# FROM SPJ
```

Παραδείγματα

- P# που δίνουν Αθηναίοι προμηθευτές ή παίρνουν έργα από Αθήνα
- Ο προφανής τρόπος: χρειαζόμαστε τους πίνακες SPJ, J, S → φυσική σύνδεση
SELECT P# FROM S, SPJ, J
WHERE S.S# = SPJ.S# AND J.J# = SPJ.J# AND
(S.CITY = 'Αθήνα' OR J.CITY = 'Αθήνα')
- Πιο οικονομικός τρόπος: η Αθήνα δεν μας χρειάζεται, πέρα από το να βρούμε τα έργα και τους προμηθευτές που είναι από εκεί.
SELECT P# FROM SPJ
WHERE S# IN (λίστα Αθηναίων προμηθευτών) OR J# IN (λίστα αθηναϊκών έργων)
- Η πρώτη είναι SELECT S# FROM S WHERE CITY = 'Αθήνα'. Η δεύτερη είναι SELECT J# FROM J WHERE CITY = 'Αθήνα'

Παραδείγματα

- Μπορούμε να χειριστούμε το παράδειγμα και κάνοντας ένωση των συνόλων που προκύπτουν από τις δύο συνθήκες
(SELECT P# FROM SPJ
WHERE S# IN (SELECT S# FROM S WHERE CITY = 'Αθήνα'))
UNION
(SELECT P# FROM SPJ
WHERE J# IN (SELECT J# FROM J WHERE CITY = 'Αθήνα'))
- Πόλεις που έχουν τουλάχιστον ένα υλικό / έργο/ προμηθευτή
- Χρειαζόμαστε τις λίστες των πόλεων για το καθένα και ακολούθως την ένωση τους
(SELECT CITY FROM P) UNION (SELECT CITY FROM J) UNION (SELECT CITY FROM S)

Παραδείγματα

- Έργα που παίρνουν κόκκινα υλικά από Αθηναίους προμηθευτές
- Χρειαζόμαστε τους πίνακες SPJ, P (για το χρώμα) και S (για την Αθήνα)

```
SELECT J# FROM SPJ, P, S
```

```
WHERE S.S# =SPJ.S# AND P.P# = SPJ.P# AND
```

```
(COLOR = 'κόκκινο' AND S.CITY = 'Αθήνα')
```

Σύνδεση μεταξύ γραμμών του πίνακα

- Ζεύγη P# για υλικά που προμηθεύει ο ίδιος προμηθευτής

S1	P1
S1	P2
S2	P3
S2	P4
S3	P1
S4	P1
S4	P3
S5	P2

- Αν ο πίνακας SPI έχει την παραπάνω μορφή, πρέπει να σχηματίσουμε τα ζευγάρια (P1, P2) από τον S1, (P3, P4) από τον S2, (P1, P3) από τον S4. Από τους S3 και S5, δεν θα προκύψει ζευγάρι γιατί δεν φαίνεται να δίνουν άλλο υλικό.

Σύνδεση μεταξύ γραμμών του πίνακα

- Τι ζητάμε λοιπόν?
 - Να συσχετίσουμε διαφορετικές γραμμές του ίδιου πίνακα με βάση την ίδια τιμή στο πεδίο S#
 - Συνεπώς χρειαζόμαστε έναν πίνακα που να περιέχει αυτές τις γραμμές και να κάνουμε **φυσική σύνδεση** με βάση το κοινό πεδίο. Αυτός ο πίνακας είναι μια δεύτερη έκδοση του SPJ !
SELECT A.P#, B.P# FROM SPJ AS A, SPJ AS B
WHERE A.S# = B.S#

Σύνδεση μεταξύ γραμμών του πίνακα

- Προσοχή: αναγκαστικά ενώνεται και η κάθε γραμμή με τον εαυτό της
- Προκύπτει γραμμή με ίδια P# και αυτή δεν μας χρειάζεται:
WHERE A.P# <> B.P#
- Επειδή οι γραμμές είναι ίδιες, θα προκύψει ζεύγος (P1, P2) αλλά και το αντίστροφο (P2,P1) και πρέπει να διώξουμε το ένα (αδιάφορο ποιο).
- Πάντα το ένα θα είναι μεγαλύτερο από το άλλο. Άρα μπορούμε να σβήσουμε αυτό που έχει το πρώτο «μεγαλύτερο», άρα κρατάμε αυτό που έχει το πρώτο «μικρότερο»

WHERE A.P# < B.P#

Ουσιαστικά αυτή η συνθήκη περιλαμβάνει και την πρώτη

Άρα τελικά

```
SELECT A.P#, B.P# FROM SPJ AS A, SPJ AS B
WHERE A.S# = B.S# AND
( A.P# < B.P#)
```