

4. Σχεδίαση Σχεσιακών ΒΔ



- **4.1. Θεωρία Κανονικοποίησης**
 - Συναρτησιακές εξαρτήσεις
 - Κανονικές Μορφές
- **4.2. Εννοιολογικός σχεδιασμός**
 - το Μοντέλο Οντοτήτων-Συσχετίσεων (ER)
 - Από τα διαγράμματα ER στο Σχεσιακό Μοντέλο
- **4.3. Αντικείμενα στις ΒΔ**

εκτός ύλης

Lectures on Databases: section IV “DB Design”, v. 2023.05
by

Data Science Lab. @ Univ. Piraeus (www.datastories.org)

Main source of slides: Silberschatz et al., “Database System Concepts”, 6/e

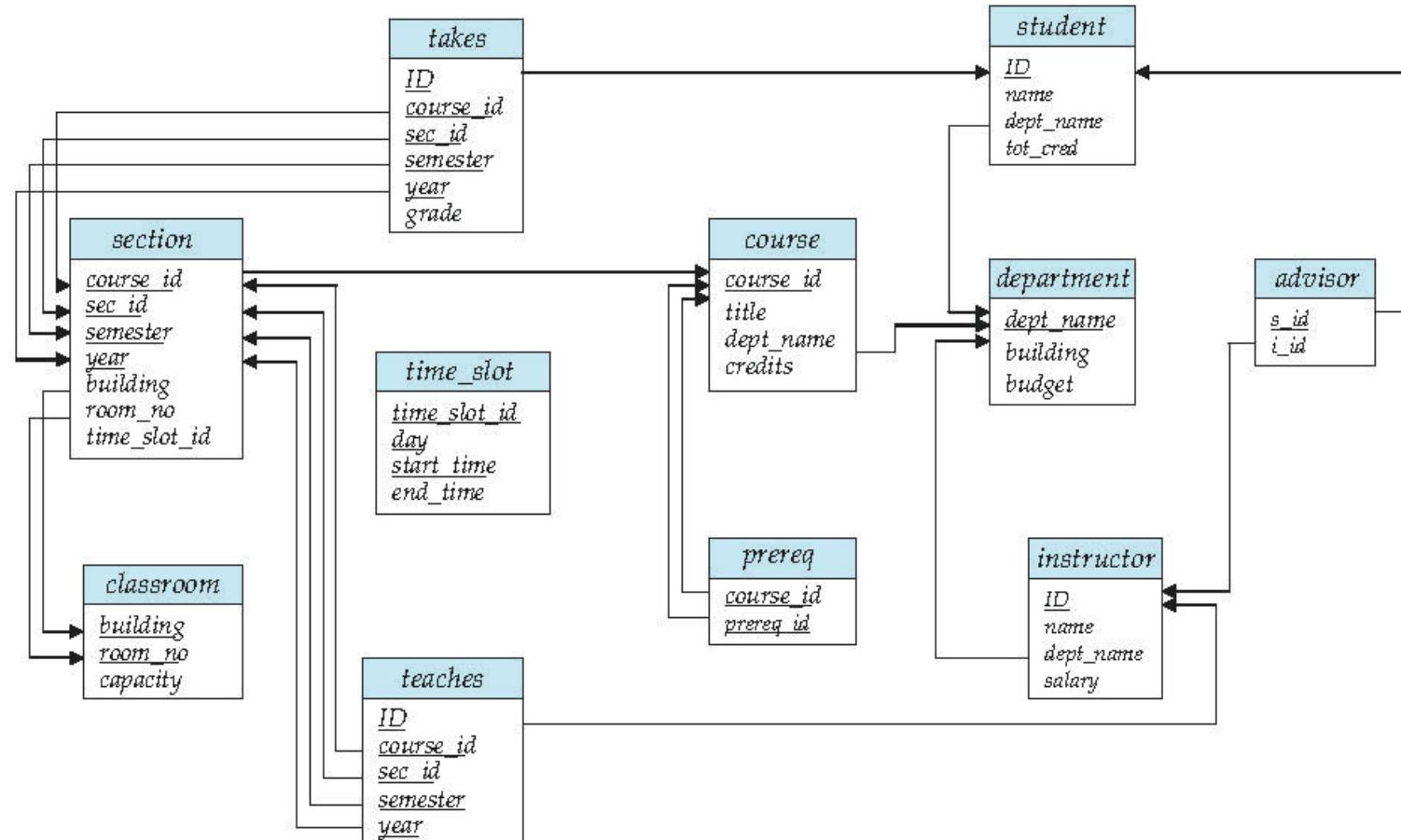


4.1. Θεωρία Κανονικοποίησης

Θεωρία κανονικοποίησης - το πρόβλημα ...



- Γιατί αυτοί οι 11 πίνακες και όχι κάποιοι άλλοι (περισσότεροι ή λιγότεροι, με διαφορετικό μοίρασμα χαρακτηριστικών);



Παράδειγμα «προβληματικού» σχήματος



- Έστω ο πίνακας `Inst_Dept` (αντί των `Instructor`, `Department`):
- Ποια προβλήματα βλέπετε;

- Προβλήματα:

- Πλεονασμός – επανάληψη δεδομένων
- Πιθανή ανάγκη για τιμές `null`

<code>ID</code>	<code>name</code>	<code>salary</code>	<code>dept_name</code>	<code>building</code>	<code>budget</code>
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

Στόχοι στη σχεδίαση σχεσιακών ΒΔ



- Στόχος της σχεδίασης σχεσιακών ΒΔ είναι να βρούμε μια συλλογή «σωστά» σχεδιασμένων τύπων σχέσεων.
- Σχεδιαστικοί στόχοι:
 - Αποφυγή πλεονασμού στα δεδομένα
 - Εξασφάλιση ότι αναπαρίστatai όλη η δυνατή πληροφορία της ΒΔ
- Το «εργαλείο»: **κανονικοποίηση ΒΔ**
 - Καταγραφή **συναρτηριακών εξαρτήσεων**
 - Υπακοή στη **κανονική μορφή Boyce-Codd** (Boyce-Codd Normal Form - BCNF)
 - Έχουν οριστεί 5 κανονικές μορφές που η επόμενη είναι πιο αυστηρή από την προηγούμενη. Η BCNF βρίσκεται μεταξύ 3NF και 4NF

Συναρτησιακές Εξαρτήσεις



- Περιορισμοί πάνω στο σύνολο των έγκυρων σχέσεων
- Απαιτούν η τιμή ορισμένων χαρακτηριστικών να προσδιορίζει μοναδικά την τιμή άλλων χαρακτηριστικών
- Παραδείγματα:
 - $ID \rightarrow name, dept_name, salary, building, budget$
 - $dept_name \rightarrow building, budget$

<i>ID</i>	<i>name</i>	<i>salary</i>	<i>dept_name</i>	<i>building</i>	<i>budget</i>
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

Συναρτησιακές Εξαρτήσεις (συν.)



- Ορισμός: Έστω R ένας τύπος σχέσης, με $\alpha \subseteq R$ and $\beta \subseteq R$.
Η **συναρτησιακή εξάρτηση (functional dependence - FD)** $\alpha \rightarrow \beta$ ισχύει πάνω στον R εάν και μόνο εάν

$$t_1[\alpha] = t_2[\alpha] \Rightarrow t_1[\beta] = t_2[\beta]$$

- Σχετικό με την έννοια του **κλειδιού**
- Υπενθύμιση ορισμών σχετικών με τα κλειδιά*:
 - Το σύνολο χαρακτηριστικών K αποτελεί **υπερκλειδί (superkey)** για τον τύπο σχέσης R εάν και μόνο εάν $K \rightarrow R$
 - Το K αποτελεί **υποψήφιο κλειδί (candidate key)** για τον R εάν και μόνο εάν: $K \rightarrow R$ και δεν υπάρχει $\alpha \subset K$, $\alpha \rightarrow R$

* **Σημείωση:** η SQL δεν παρέχει κάποιον άμεσο τρόπο υποστήριξης των συναρτησιακών εξαρτήσεων, πέρα από τους περιορισμούς του κλειδιού. Υποστηρίζονται μέσω assertions / triggers (πρακτικά γίνονται επιπλέον έλεγχοι με κώδικα)

Συναρτησιακές Εξαρτήσεις (συν.)



- Χρησιμοποιούμε συναρτησιακές εξαρτήσεις για να:
 - Ελέγχουμε την εγκυρότητα μιας σχέσης r σύμφωνα με ένα δοθέν σύνολο συναρτησιακών εξαρτήσεων FD (οπότε λέμε ότι η r **ικανοποιεί** την FD_i).
 - Προδιαγράψουμε περιορισμούς FD πάνω σε έναν τύπο σχέσης R (οπότε λέμε ότι η FD_i **ισχύει για** τον R)

- Μια συναρτησιακή εξάρτηση καλείται **τετριμένη** εάν ικανοποιείται οπωσδήποτε σε κάθε στιγμιότυπο ενός τύπου σχέσης
 - $\alpha \rightarrow \beta$ αποτελεί τετριμένη FD εάν $\beta \subseteq \alpha$
 - π.χ. $ID, name \rightarrow ID, name \rightarrow name$

<i>ID</i>	<i>name</i>	<i>salary</i>	<i>dept_name</i>	<i>building</i>	<i>budget</i>
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

Κανονικοποίηση σχήματος ΒΔ με χρήση συναρτησιακών εξαρτήσεων



Έστω ότι (για τις ανάγκες κανονικοποίησης του σχήματος ΒΔ) αποσυνθέτουμε έναν «προβληματικό» τύπο σχέσης R , στον οποίο ισχύει ένα σύνολο συναρτησιακών εξαρτήσεων FD , σε n τύπους σχέσεων R_1, R_2, \dots, R_n . Ο στόχος της κανονικοποίησης είναι τριπλός:

- **Στόχος 1) όχι απώλεια πληροφορίας** : η αποσύνθεση δεν πρέπει να οδηγήσει σε απώλεια πληροφορίας (lossless decomposition)
 - ✓ 'Όλα τα χαρακτηριστικά της αρχικής σχέσης πρέπει να διατηρούνται
 - ✓ 'Όλες οι πλειάδες της αρχικής σχέσης πρέπει να προκύπτουν με κατάλληλη σύνδεση των σχέσεων στις οποίες αυτή έχει αποσυντεθεί
- **Στόχος 2) όχι πλεονασμός πληροφορίας** : οι τύποι σχέσεων R_i που προκύπτουν από την αποσύνθεση να είναι «օρθοί» σχεδιαστικά
 - ✓ σε Boyce-Codd KM ή τουλάχιστον 3^η KM
(θα τις ορίσουμε σε λίγο ...)
- **Στόχος 3) διατήρηση εξαρτήσεων** : παρά την αποσύνθεση, όλες οι εξαρτήσεις FD πρέπει να διατηρούνται (άμεσα ή έμμεσα) μέσα από τους νέους τύπους σχέσεων.

<i>ID</i>	<i>name</i>	<i>salary</i>	<i>dept_name</i>	<i>building</i>	<i>budget</i>
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000

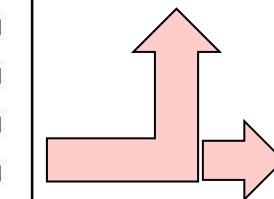
Παράδειγμα αποσύνθεσης (decomposition)

- Αποσύνθεση του «προβληματικού» πίνακα Inst_Dept στους «օρθούς» σχεδιαστικά Instructor, Department

<i>ID</i>	<i>name</i>	<i>salary</i>	<i>dept_name</i>	<i>building</i>	<i>budget</i>
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

<i>ID</i>	<i>name</i>	<i>dept_name</i>	<i>salary</i>
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

(a) The *instructor* table

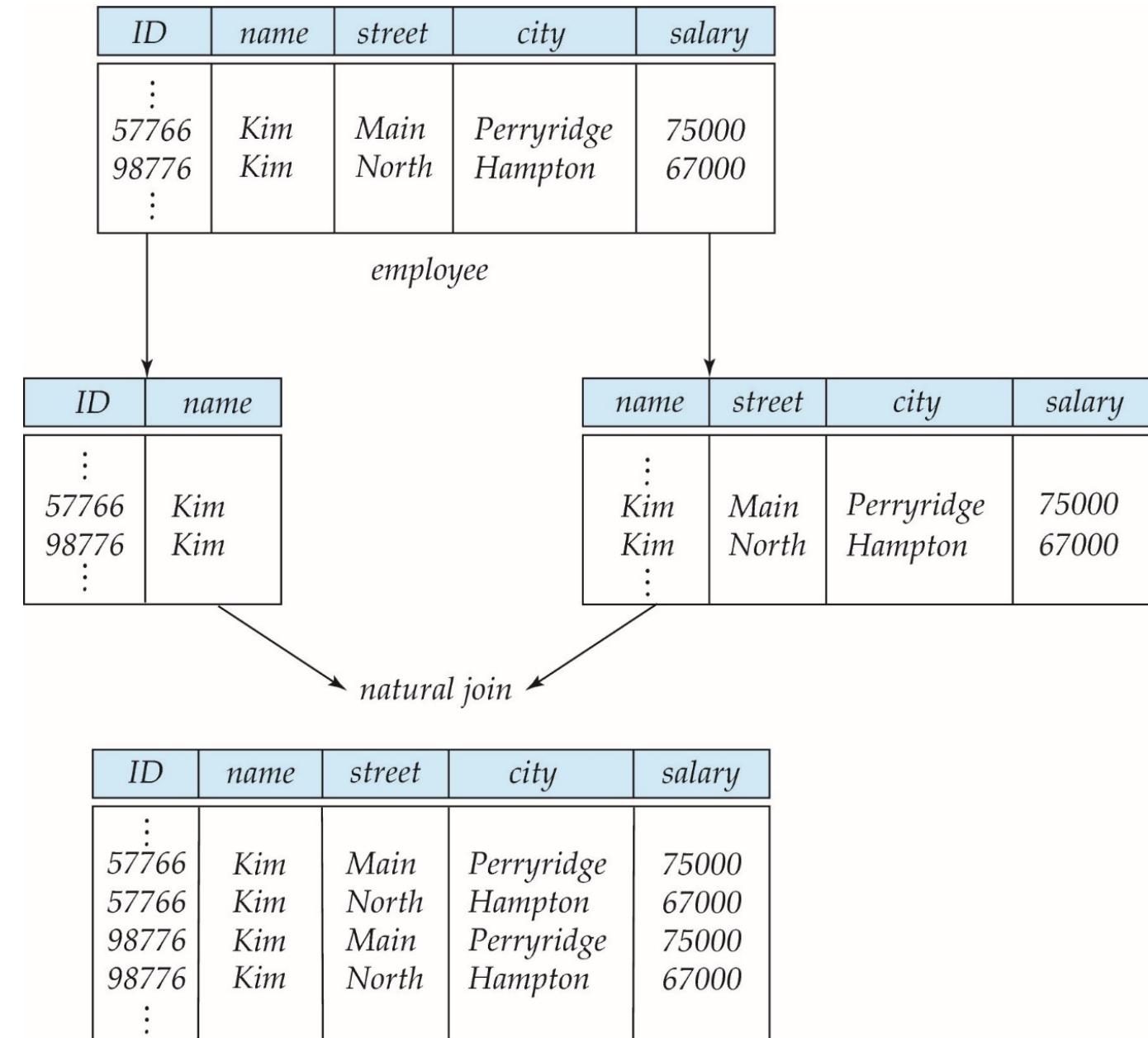


<i>dept_name</i>	<i>building</i>	<i>budget</i>
Comp. Sci.	Taylor	100000
Biology	Watson	90000
Elec. Eng.	Taylor	85000
Music	Packard	80000
Finance	Painter	120000
History	Painter	50000
Physics	Watson	70000

(b) The *department* table



Παράδειγμα αποσύνθεσης με απώλεια πληροφορίας



1^η, 2^η, 3^η Κανονική Μορφή



- Ένας τύπος σχέσης R είναι σε **1^η, 2^η, 3^η ΚΜ (1NF, 2NF, 3NF)** εάν:
 - 1NF** : τα πεδία τιμών όλων των χαρακτηριστικών του R είναι **ατομικά**
 - ... και, επιπλέον, για όλες τις συναρτησιακές εξαρτήσεις $a \rightarrow b$ που ισχύουν στον R , ισχύουν τα εξής:
- 2NF** : κανένα χαρακτηριστικό (από αυτά που δεν συμμετέχουν σε υποψήφιο κλειδί) δεν **προσδιορίζεται** από **τμήμα** ενός υποψήφιου κλειδιού K , δηλ.

$\nexists (a \rightarrow b) \text{ in FD: } a \subset K \wedge b \notin K$

3NF : κανένα χαρακτηριστικό (από αυτά που δεν συμμετέχουν σε υποψήφιο κλειδί) δεν **προσδιορίζεται μεταβατικά** (δηλ. μέσω άλλου χαρακτηριστικού – το οποίο **δεν** συμμετέχει σε υποψήφιο κλειδί) από ένα υποψήφιο κλειδί K , δηλ.

$\nexists (a \rightarrow b), (b \rightarrow c) \text{ in FD: }$
 $a = K \wedge b \notin K \wedge c \notin K$

<i>ID</i>	<i>name</i>	<i>salary</i>	<i>dept_name</i>	<i>building</i>	<i>budget</i>
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000

1η, 2η, 3η Κανονική Μορφή (συν.)

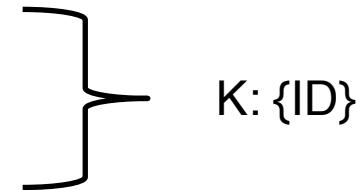


- Παράδειγμα:

`Inst_Dept = (ID, name, salary, dept_name, building, budget)`

- FD1: $ID \rightarrow name, dept_name, salary, building, budget$
 - FD2: $dept_name \rightarrow building, budget$
- Ο τύπος σχέσης `Inst_Dept` είναι σε 1NF, 2NF αλλά δεν είναι σε 3NF λόγω της «μεταβατικής» εξάρτησης:

- $ID \rightarrow dept_name$
- $dept_name \rightarrow building, budget$



<code>ID</code>	<code>name</code>	<code>salary</code>	<code>dept_name</code>	<code>building</code>	<code>budget</code>
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000

Boyce-Codd Κανονική Μορφή



- Ένας τύπος σχέσης R είναι σε **Boyce-Codd Κανονική Μορφή (Boyce-Codd Normal Form - BCNF)** εάν για όλες τις συναρτησιακές εξαρτήσεις $a \rightarrow b$ που ισχύουν στον R , το a αποτελεί υποψήφιο κλειδί του R , δηλ.

$\nexists (a \rightarrow b) \text{ in FD: } a \neq K$

- Σημειώσεις:
 - Για να αποφανθούμε αν ένας τύπος σχέσης είναι σε BCNF δεν ελέγχουμε προηγουμένως αν είναι σε 3NF αλλά εξετάζουμε απευθείας τις συναρτησιακές εξαρτήσεις
 - η BCNF είναι πιο «αυστηρή» από την 3NF: αν ένας τύπος σχέσης R είναι σε BCNF τότε σίγουρα είναι και σε 3NF

- Παράδειγμα: ο τύπος σχέσης Inst_Dept (με $K = ID$) δεν είναι σε BCNF λόγω της FD2:
 - $\text{dept_name} \rightarrow \text{building}, \text{budget}$

<i>ID</i>	<i>name</i>	<i>salary</i>	<i>dept_name</i>	<i>building</i>	<i>budget</i>
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000

Boyce-Codd Κανονική Μορφή (συν.)

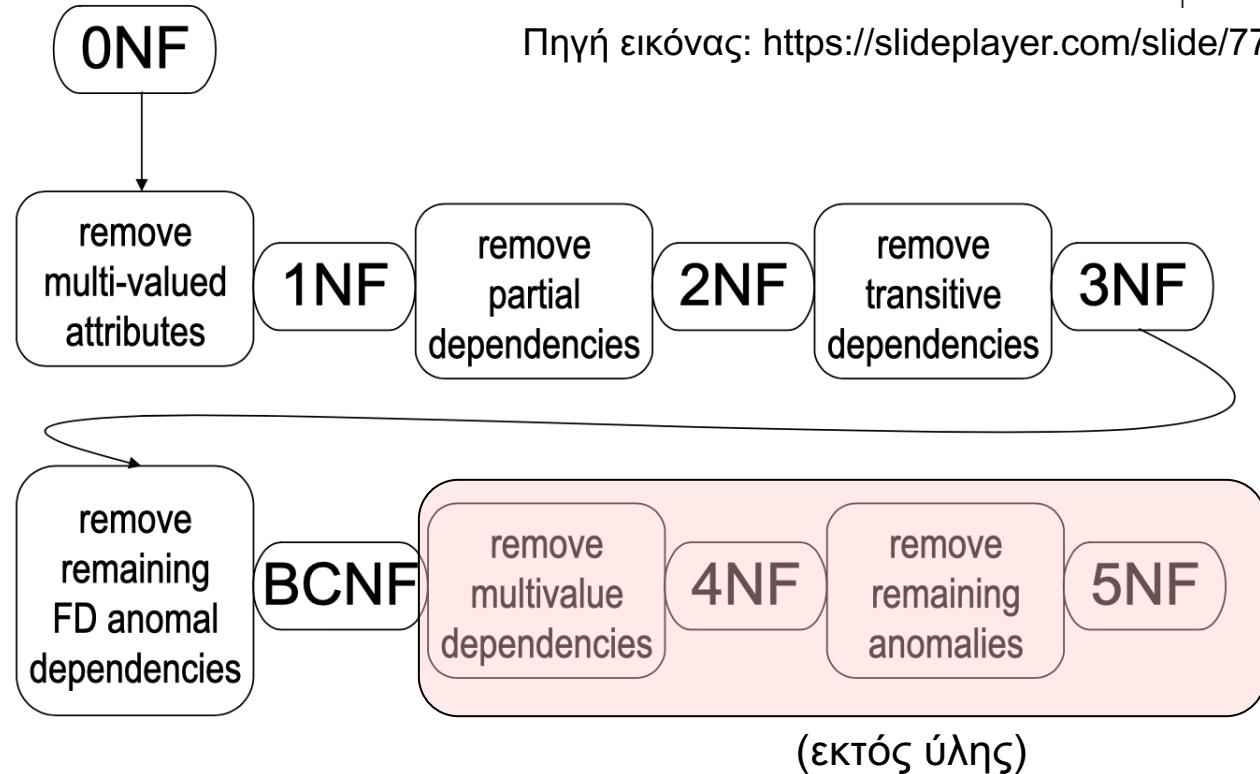


- **Πώς γίνεται αποσύνθεση σε BCNF***: οι «προβληματικές» συναρτησιακές εξαρτήσεις σχηματίζουν νέους τύπους σχέσεων
 - Συνήθως, ένας τύπος σχέσης ανά FD (αλλά αυτό δεν αποτελεί κανόνα, π.χ. όταν υπάρχουν FDs που 'μοιράζονται' τα ίδια χαρακτηριστικά)
- Παράδειγμα: ο τύπος σχέσης Inst_Dept με
 - FD1: ID → name, dept_name, salary, building, budget
 - FD2: dept_name → building, budget
- ... που δείξαμε ότι δεν είναι σε BCNF
 - λόγω της FD2
- ... αποσυντίθεται στους τύπους σχέσεων:
 - Instructor = (ID, name, salary, dept_name)
 - Department = (dept_name, building, budget)
- ... οι οποίοι είναι σε BCNF !!

* **Σημείωση:** Θεωρητικά μπορεί να προκύψει το ενδεχόμενο να μην μπορούμε να κάνουμε αποσύνθεση σε BCNF διατηρώντας τις συναρτησιακές εξαρτήσεις. Σε αυτή την περίπτωση, αναγκαστικά «υποχωρούμε» σε 3NF (που πάντοτε μπορεί να επιτευχθεί), επιτρέποντας όμως έτσι πλεονασμό πληροφορίας

ID	name	salary	dept_name	building	budget
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000

Ως προς την ‘αυστηρότητα’ των KM



Για την ιστορία, οι 1NF-2NF-3NF ορίστηκαν στην εργασία [1] και η BCNF (ή 3.5NF) ορίστηκε στη [2]:

[1] Codd, E. F. "Further Normalization of the Data Base Relational Model". IBM Research Report RJ909 (1971)

[2] Codd, E. F. "Recent Investigations into Relational Data Base" in Proc. 6th IFIP Congress (1974)

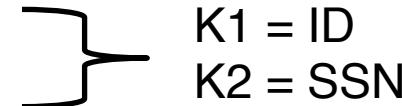
Επαναληπτική άσκηση στις ΚΜ (1/4)



Πληροφορίες σχετικά με τους φοιτητές (ταυτοποίηση είτε με ID είτε με SSN) καθώς και το Τμήμα στο οποίο έχουν εγγραφεί

- **STUDENT (ID, SSN, NAME, DEPT-NAME)**

- FD1: ID \rightarrow SSN, NAME, DEPT-NAME
- FD2: SSN \rightarrow ID, NAME, DEPT-NAME



NF? **BCNF**

Επαναληπτική άσκηση στις KM (2/4)

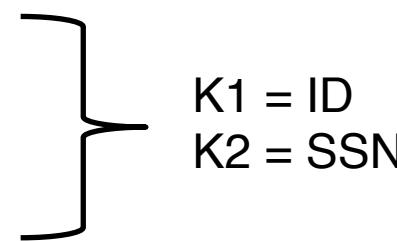


Πληροφορίες σχετικά με τους φοιτητές ... καθώς και το URL του Τμήματος

- παραδοχή: κάθε Τμήμα έχει το δικό του (ένα μόνο) URL

- **STUDENT2 (ID, SSN, NAME, DEPT-NAME, DEPT-URL)**

- FD1: ID --> SSN, NAME, DEPT-NAME, DEPT-URL
- FD2: SSN --> ID, NAME, DEPT-NAME, DEPT-URL
- FD3: DEPT-NAME --> DEPT-URL
- FD4: DEPT-URL --> DEPT-NAME



NF? **2NF**

Αποσύνθεση:

- **STUDENT2a (ID, SSN, NAME, DEPT-NAME)** // maintains FD1, FD2

- K1 = ID; K2 = SSN

NF? **BCNF**

- **STUDENT2b (DEPT-NAME, DEPT-URL)** // maintains FD3, FD4

- K1 = DEPT-NAME; K2 = DEPT-URL

NF? **BCNF**

Επαναληπτική άσκηση στις KM (3/4)

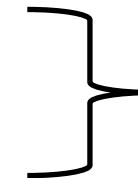


Πληροφορίες σχετικά με τους φοιτητές ... καθώς και το URL του Τμήματος

- παραδοχή: κάθε Τμήμα έχει ένα ή περισσότερα URLs, ένα URL ταυτοποιεί ένα Τμήμα

▪ **STUDENT3 (ID, SSN, NAME, DEPT-NAME, DEPT-URL)**

- FD1: ID \rightarrow SSN, NAME, DEPT-NAME
- FD2: SSN \rightarrow ID, NAME, DEPT-NAME
- FD3: DEPT-URL \rightarrow DEPT-NAME



K1 = ID, DEPT-URL
K2 = SSN, DEPT-URL

NF? **1NF**

Αποσύνθεση:

- **STUDENT3a (ID, SSN, NAME, DEPT-NAME)** // maintains FD1, FD2 } K1 = ID
} K2 = SSN NF? **BCNF**
- **STUDENT3b (DEPT-URL, DEPT-NAME)** // maintains FD3 } K = DEPT-URL NF? **BCNF**
- **STUDENT3c (DEPT-NAME)** // no FD } K = DEPT-NAME NF? **BCNF**

Επαναληπτική άσκηση στις KM (4/4)

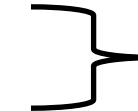


Πληροφορίες σχετικά με τους φοιτητές ... καθώς και το URL του Τμήματος

- παραδοχή: κάθε Τμήμα έχει ένα ή περισσότερα URLs, ένα URL μπορεί να ‘διαμοιράζεται’ μεταξύ Τμημάτων

- **STUDENT4 (ID, SSN, NAME, DEPT-NAME, DEPT-URL)**

- FD1: ID $\rightarrow\!\!\!$ SSN, NAME, DEPT-NAME
- FD2: SSN $\rightarrow\!\!\!$ ID, NAME, DEPT-NAME



K1 = ID, DEPT-URL
K2 = SSN, DEPT-URL

NF? **1NF**

Αποσύνθεση:

- **STUDENT4a (ID, SSN, NAME, DEPT-NAME)** // maintains FD1, FD2

- K1 = ID, K2 = SSN

NF? **BCNF**

- **STUDENT4b (DEPT-NAME, DEPT-URL)** // no FD

- K = DEPT-NAME, DEPT-URL

NF? **BCNF**



4.2. Εννοιολογικός Σχεδιασμός

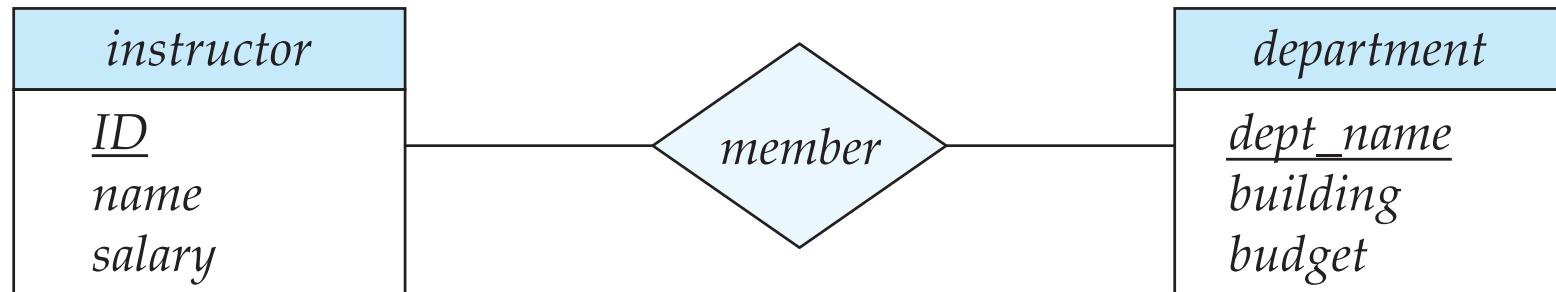
Εννοιολογικός Σχεδιασμός



(conceptual design)

- Προσπάθεια για ξεκαθάρισμα των εννοιών
- Εύρεση και καταγραφή των Οντοτήτων της ΒΔ, των μεταξύ τους Συσχετίσεων καθώς και της σημασιολογίας που τα συνοδεύει
- ΣΤΟΧΟΣ:
 - Μια **αφαιρετική**, αλλά **πλήρης** περιγραφή του τμήματος του μικρόκοσμου που θα αναπαρασταθεί στην βάση δεδομένων.
 - Αυτή η περιγραφή γίνεται με τη χρήση μιας ημι-τυπικής σημειογραφίας / συμβολισμού.

Παράδειγμα εννοιολογικού σχεδιασμού



- Μοντέλο οντοτήτων-συσχετίσεων (Entity-Relationship model – ER), π.χ.
 - Ένας Διδάσκων χαρακτηρίζεται από έναν κωδικό και άλλες πληροφορίες που καταγράφονται είναι το όνομα και ο μισθός του
 - Ένα Τμήμα χαρακτηρίζεται από το όνομά του και άλλες πληροφορίες που καταγράφονται είναι το κτήριο στο οποίο βρίσκεται και ο προϋπολογισμός του
 - Ένας Διδάσκων είναι μέλος ενός Τμήματος

Συστατικά του Ε-R Μοντέλου (1)



- Υπάρχουν δυο βασικές εννοιολογικές έννοιες:
- **Οντότητες (entities)**
 - Συγκεκριμένα αντικείμενα που υπάρχουν (ή πιστεύεται ότι υπάρχουν) και μπορούν να αναπαρασταθούν στην ΒΔ
 - π.χ., ο τάδε ΔΙΔΑΣΚΩΝ, ο τάδε ΦΟΙΤΗΤΗΣ, το τάδε ΜΑΘΗΜΑ
 - Για κάθε οντότητα καταγράφουμε ορισμένα **χαρακτηριστικά (attributes)**
- **Συσχετίσεις (relationships)**
 - Είναι επίσης (ειδικά) αντικείμενα που αντιστοιχούν δύο ή περισσότερες ξεχωριστές οντότητες με ένα συγκεκριμένο νόημα (τυπικά, μια Συσχέτιση είναι ένα διατεταγμένο σύνολο οντοτήτων)
 - π.χ. ο φοιτητής X έχει πάρει το μάθημα Y, ο καθηγητής X διδάσκει το μάθημα Y, ο καθηγητής X είναι επιβλέπων του φοιτητή Y, κοκ.

Οντότητες



- Κάθε οντότητα έχει χαρακτηριστικά που την προσδιορίζουν

instructor_ID instructor_name

76766	Crick
45565	Katz
10101	Srinivasan
98345	Kim
76543	Singh
22222	Einstein

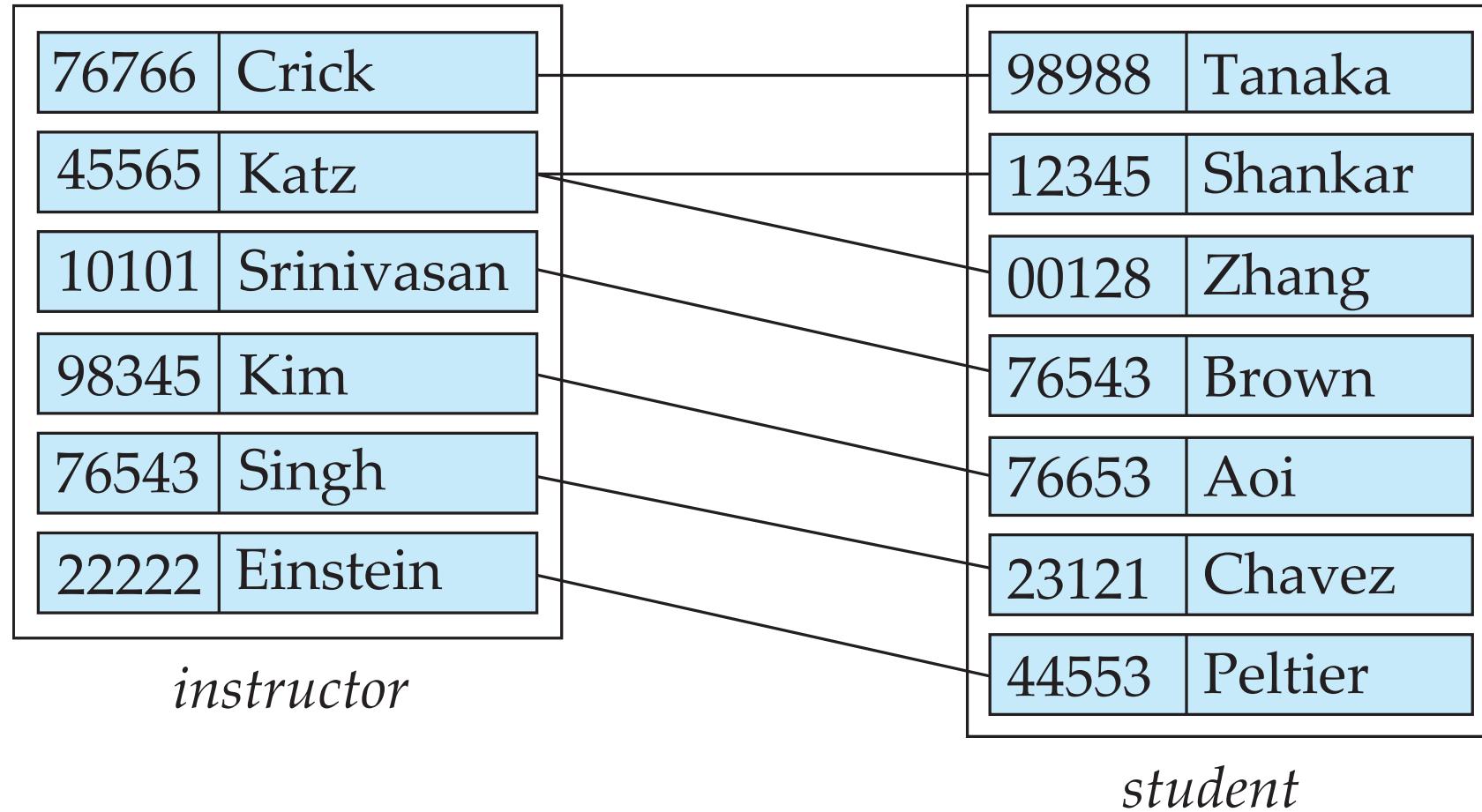
instructor

student-ID student_name

98988	Tanaka
12345	Shankar
00128	Zhang
76543	Brown
76653	Aoi
23121	Chavez
44553	Peltier

student

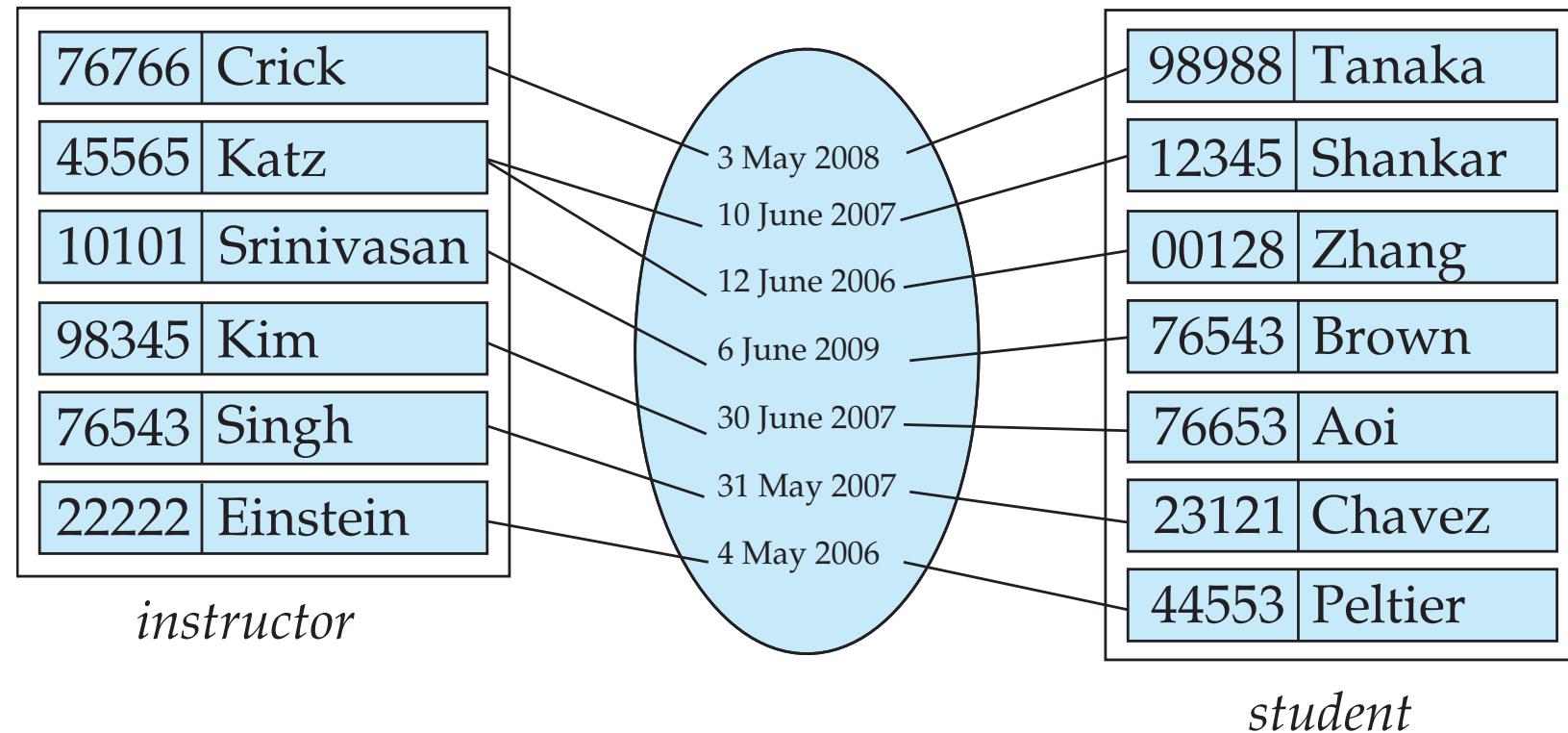
Συσχετίσεις (1)



Συσχετίσεις (2)



- Μια συσχέτιση μπορεί να έχει δικά της χαρακτηριστικά

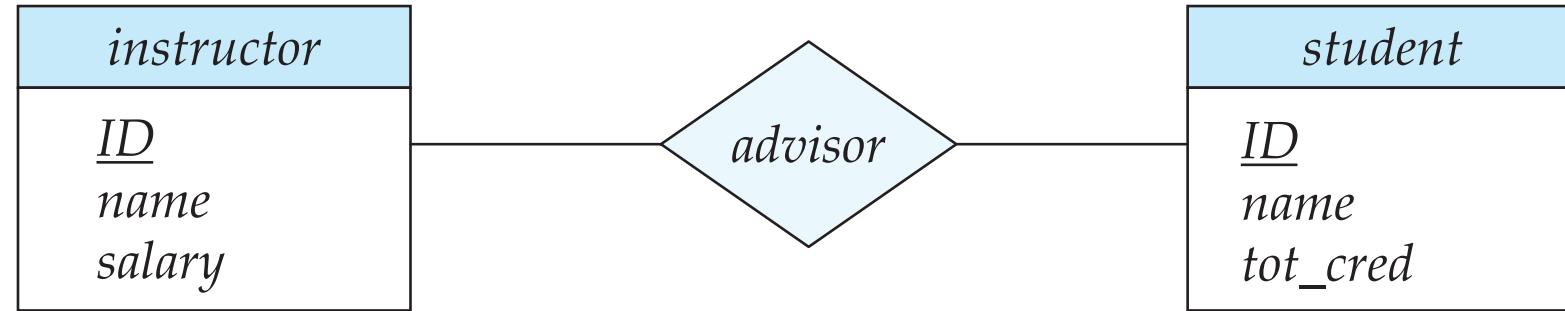


Συστατικά του E-R Μοντέλου (3)



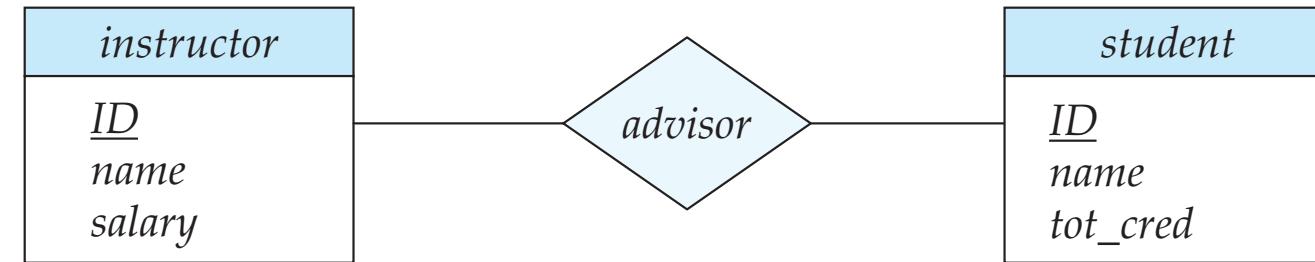
- Είδη χαρακτηριστικών
 - **Απλά (simple)**: μια οντότητα έχει ατομική τιμή για αυτό
 - π.χ., μισθός καθηγητή
 - **Σύνθετα (composite)**: το χαρακτηριστικό αποτελείται από 2 ή περισσότερα τμήματα
 - π.χ., διεύθυνση = {Δρόμος, Αριθμός, ΤΚ, Πόλη, Χώρα}
 - **Πλειότιμα (multi-valued)**: το χαρακτηριστικό έχει πολλαπλές τιμές
 - π.χ., τηλέφωνο(-α) ενός καθηγητή
 - **Εξαρτημένα (derived)**: η τιμή του χαρακτηριστικού προκύπτει από τις τιμές άλλων χαρακτηριστικών
 - π.χ., ηλικία φοιτητή, μ.ο. βαθμολογίας του

Διάγραμμα Οντοτήτων-Συσχετίσεων (Ε-R)

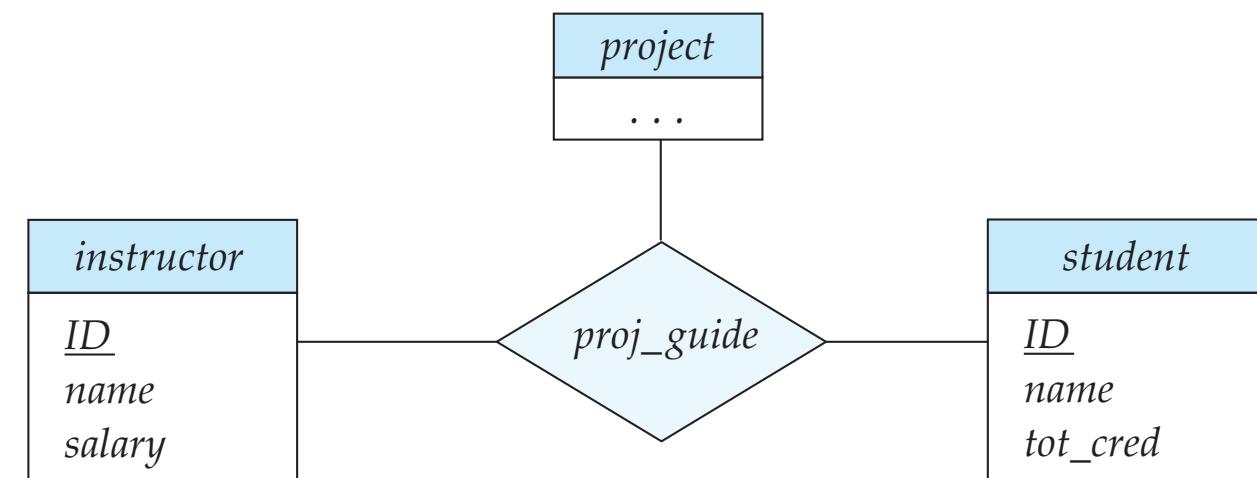


- **Ορθογώνια:** οντότητες
 - Τα χαρακτηριστικά παρουσιάζονται ως λίστα
 - Τα υπογραμμισμένα χαρακτηριστικά είναι τα πρωτεύοντα κλειδιά
- **Ρόμβοι:** συσχετίσεις

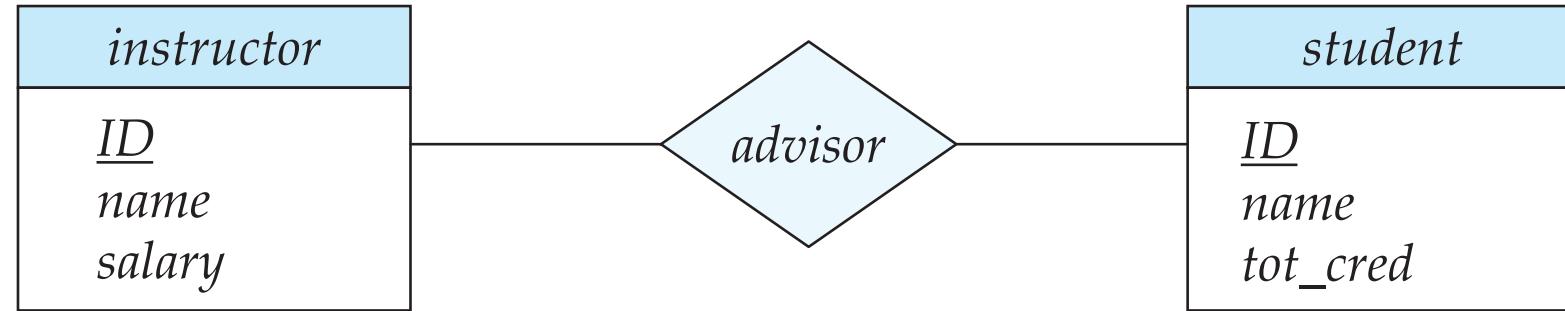
Ιδιότητες τύπων συσχετίσεων: βαθμός



- Συσχέτιση μεταξύ 2 συνόλων οντοτήτων → **διμερής**
- Συσχέτιση μεταξύ 3 συνόλων οντοτήτων → **τριμερής**
- ΚΟΚ.

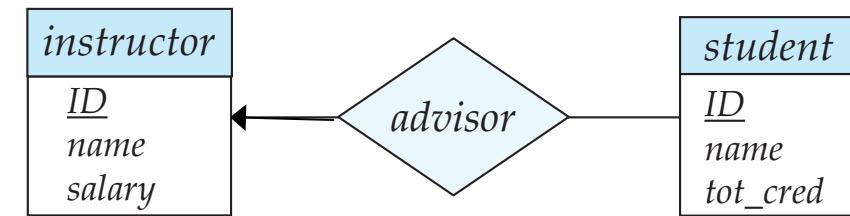
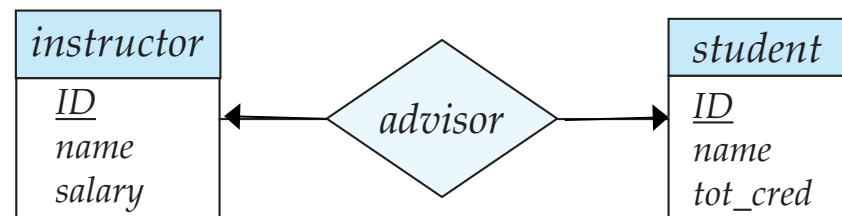
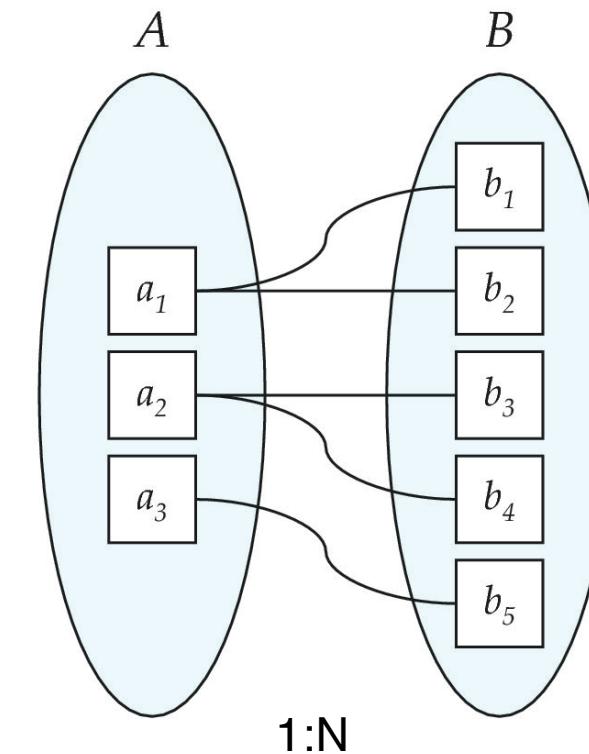
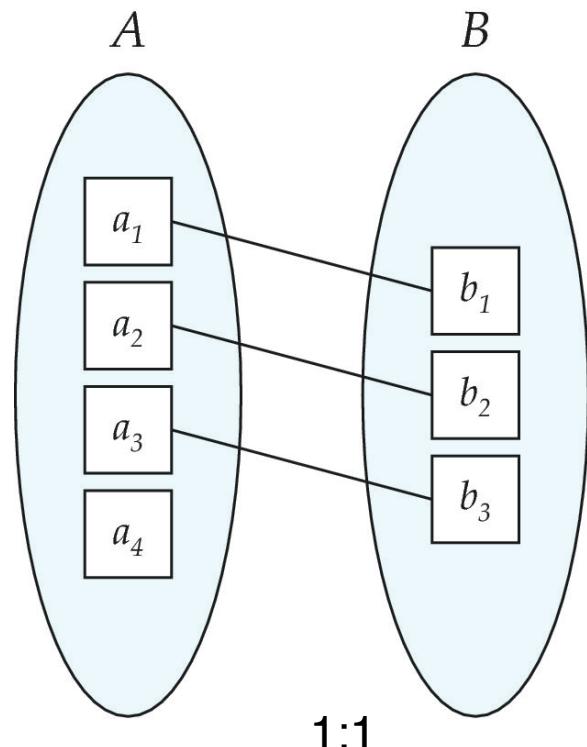


Ιδιότητες τύπων συσχετίσεων: πληθικότητα

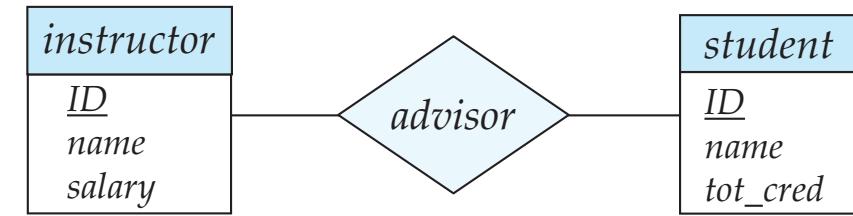
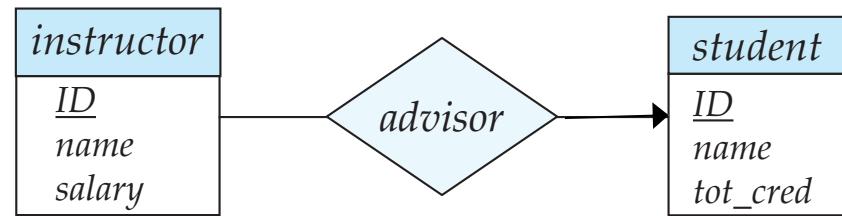
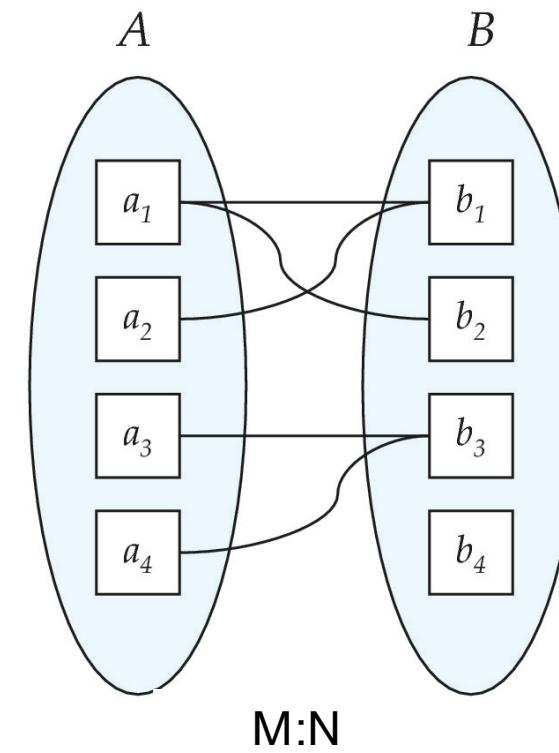
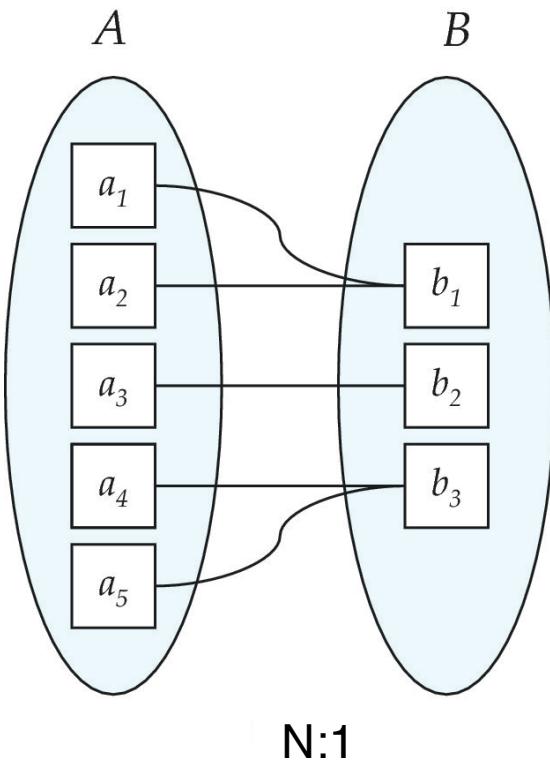


- Η **πληθικότητα (cardinality)** ενός τύπου συσχέτισης ορίζει το πόσες οντότητες από το πρώτο σύνολο οντοτήτων στην συσχέτιση μπορούν να συνδεθούν με πόσες οντότητες από το δεύτερο σύνολο οντοτήτων
- π.χ. ο τύπος συσχέτισης *instructor*-*student* είναι 1:N (ένα-προς-πολλά)
 - ένας καθηγητής μπορεί να επιβλέπει πολλούς φοιτητές
 - ένας φοιτητής μπορεί να επιβλέπεται από έναν μόνο καθηγητή

Πληθικότητα συσχέτισης (4 περιπτώσεις)



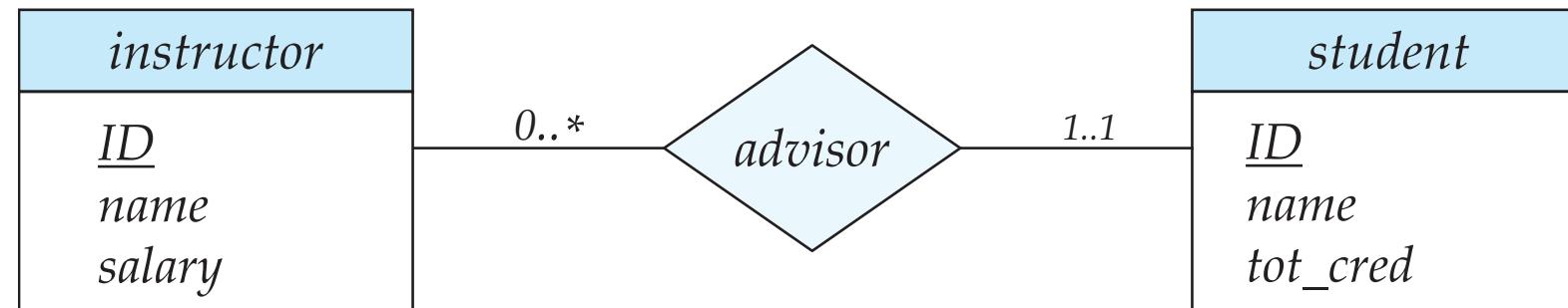
Πληθικότητα συσχέτισης (4 περιπτώσεις)



Ιδιότητες τύπων συσχετίσεων: συμμετοχή



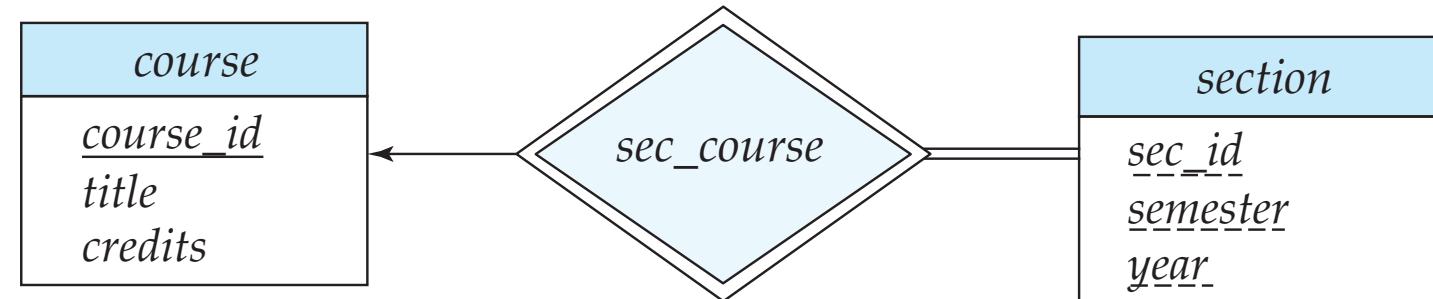
- Η **συμμετοχή** μιας οντότητας σε μια συσχέτιση μπορεί να είναι **ολική (total)** ή **μερική (partial)**
- Παράδειγμα: όλοι οι φοιτητές πρέπει να έχουν επιβλέποντα καθηγητή, αυτό όμως δεν είναι απαραίτητο για τους καθηγητές



Αδύναμα σύνολα οντοτήτων



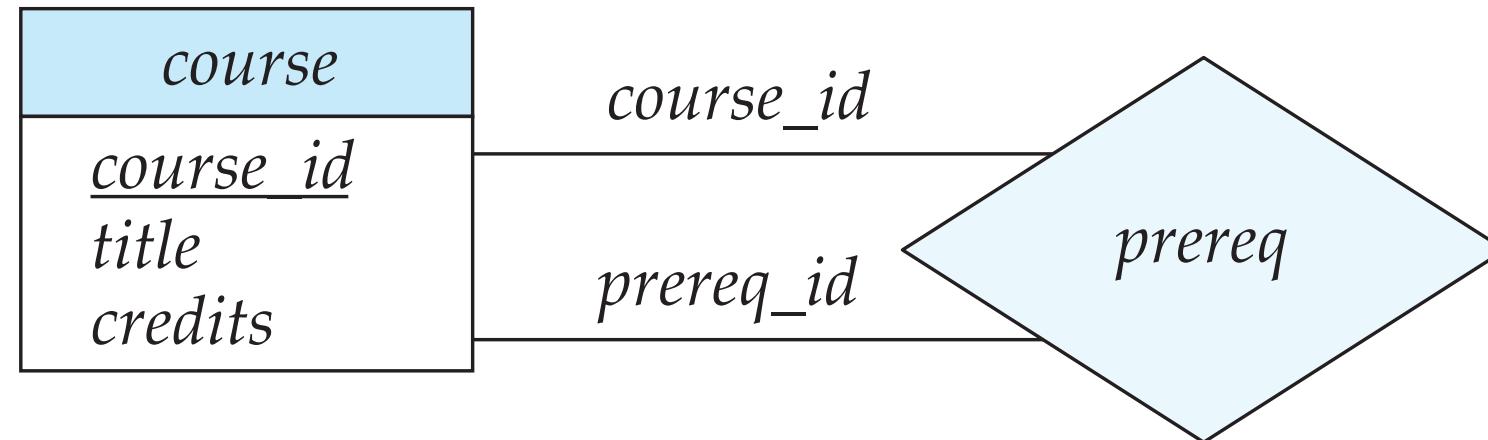
- Σύνολα οντοτήτων των οποίων η ύπαρξη εξαρτάται από την ύπαρξη μιας άλλης (ισχυρής) οντότητας
- Η (αδύναμη) συσχέτιση που προκύπτει έχει πληθικότητα 1:N πάντα (γιατί;)



Η έννοια του ρόλου



- Οι οντότητες που συσχετίζονται μέσω μιας συσχέτισης δε χρειάζεται να είναι διακριτές
- Οι ρόλοι εμφανίζονται στα διαγράμματα E-R με ετικέτες πάνω στις γραμμές που συνδέουν ρόμβους με ορθογώνια.
- Οι ετικέτες ρόλων είναι προαιρετικές (ξεκαθαρίζουν τη σημασιολογία της συσχέτισης)

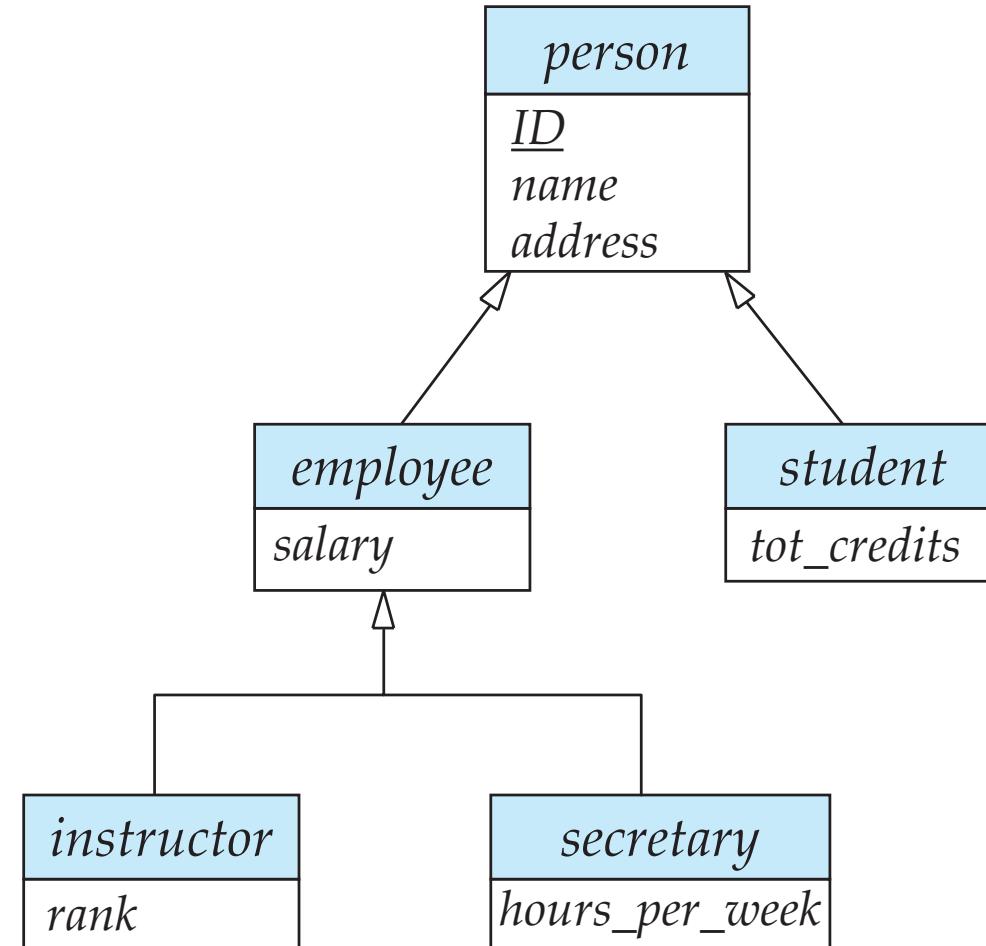


- Παράδειγμα: Οι ετικέτες “course_id” and “prereq_id” δείχνουν πώς τα μαθήματα αλληλεπιδρούν μέσω της συσχέτισης *prereq* (προαπαιτούμενα)

Εξειδίκευση (Specialization)



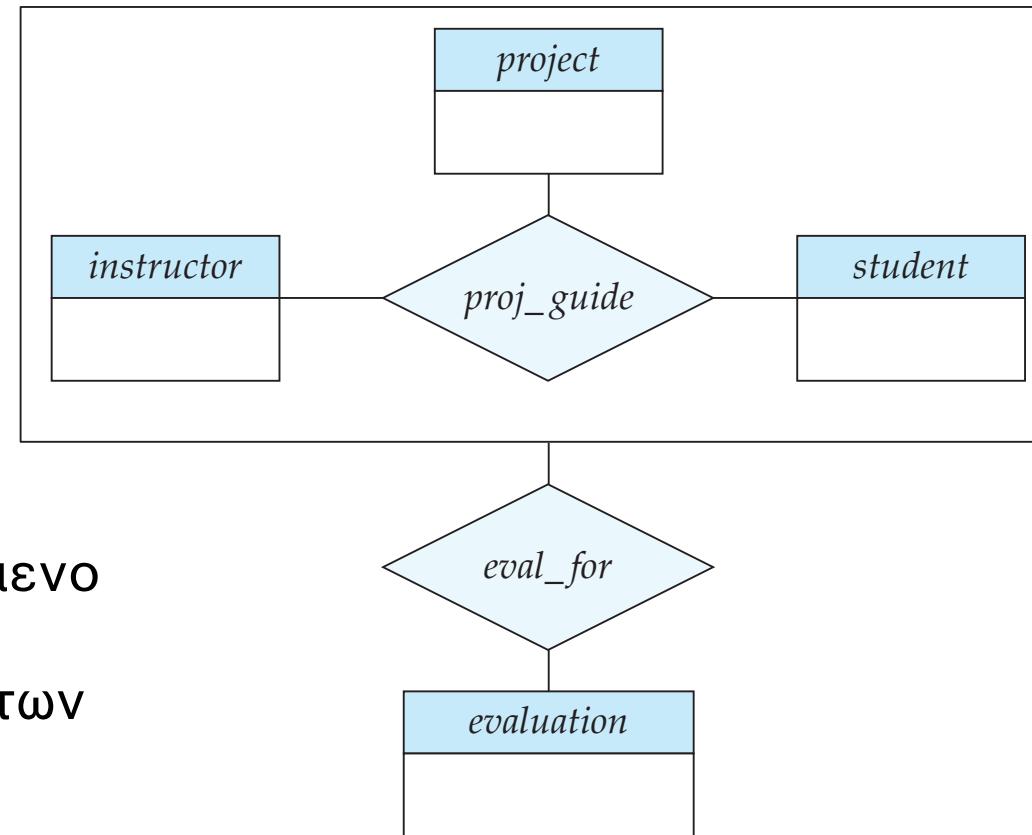
- Μια οντότητα (π.χ. *person*) εξειδικεύεται περαιτέρω σε άλλες οντότητες (π.χ. *employee* ή *student*)
- Περιπτώσεις εξειδίκευσης:
 - Πλήρης (total) ή μερική (partial)
 - Διακριτή (disjoint) ή επικαλυπτόμενη (overlapping)



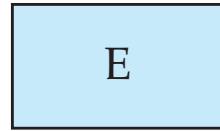
Συσσώρευση (Aggregation)



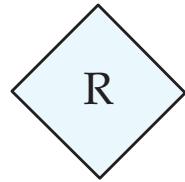
- Αποτελεί ειδική περίπτωση συσχέτισης και επιτρέπει συσχέτιση μεταξύ συνόλου οντοτήτων και συνόλου συσχετίσεων
- Στην περίπτωση αυτή, το εμπλεκόμενο σύνολο συσχετίσεων θεωρείται (καταχρηστικά) ως σύνολο οντοτήτων



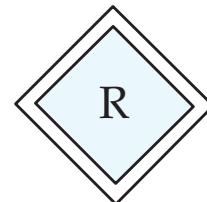
Συμβολισμοί στο μοντέλο E-R (1)



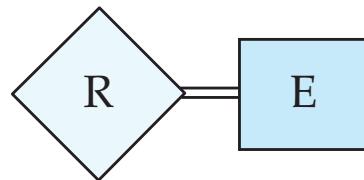
entity set



relationship set



identifying
relationship set
for weak entity set



total participation
of entity set in
relationship

E
A1
A2
A2.1
A2.2
{A3}
A4()

attributes:
simple (A1),
composite (A2) and
multivalued (A3)
derived (A4)

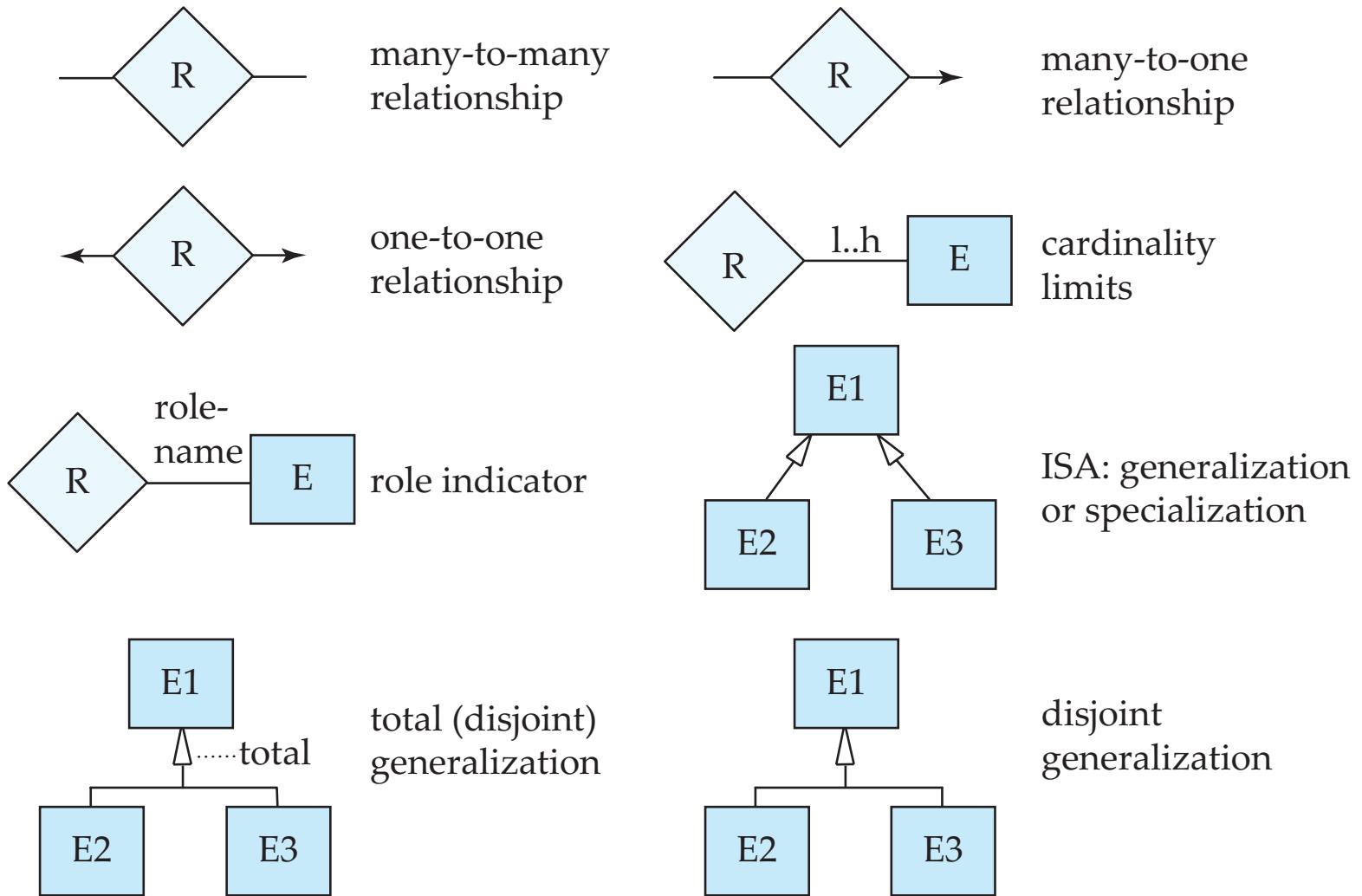
E
<u>A1</u>

primary key

E
A1
.....

discriminating
attribute of
weak entity set

Συμβολισμοί στο μοντέλο E-R (2)



Μετατροπή σχήματος E-R σε σχεσιακό σχήμα



- Τα πρωτεύοντα κλειδιά επιτρέπουν να εκφραστούν τα σύνολα οντοτήτων και τα σύνολα συσχετίσεων ως πίνακες που αναπαριστούν τα περιεχόμενα μιας βάσης δεδομένων
- Μια βάση δεδομένων συμβατή με ένα διάγραμμα E-R μπορεί να αναπαρασταθεί με μια συλλογή πινάκων
- Η μετατροπή ενός διαγράμματος E-R σε συλλογή πινάκων αποτελεί την αφετηρία για να προκύψει μια σχεσιακή βάση δεδομένων από μια εννοιολογική σχεδίαση στο μοντέλο E-R
- Βασικοί κανόνες μετατροπής E-R σε σχεσιακό ...



Αναπαράσταση οντοτήτων & συσχετίσεων

1. Ένα (ισχυρό) σύνολο οντοτήτων μετατρέπεται σε πίνακα (με τα ίδια χαρακτηριστικά)
2. Ένα σύνολο συσχετίσεων M:N αναπαρίσταται ως πίνακας με στήλες για τα πρωτεύοντα κλειδιά των οντοτήτων που συμμετέχουν, και επιπλέον όλα τα χαρακτηριστικά του συνόλου συσχετίσεων
3. Οι συσχετίσεις N:1 και 1:N μπορούν να αναπαρασταθούν απλά με προσθήκη ενός επιπλέον χαρακτηριστικού στην πλευρά 'N', με το πρωτεύον κλειδί της πλευράς '1'
 - Εάν η συμμετοχή στην πλευρά 'N' είναι μερική, μπορεί να προκύψει η περίπτωση μια στήλη του πίνακα να έχει πολλές κενές τιμές. Οπότε ίσως θα συνέφερε η δημιουργία νέου πίνακα (όπως κάνουμε στην περίπτωση M:N)



Αναπαράσταση οντοτήτων & συσχετίσεων

4. Για τις συσχετίσεις 1:1 έχουμε 3 εναλλακτικές προσεγγίσεις:
- είτε να προσθέσουμε το πρωτεύον κλειδί της μιας πλευράς ως επιπλέον χαρακτηριστικό στον πίνακα της άλλης πλευράς
 - και αυτό να γίνει για τον ένα ή και τους δύο πίνακες (στη δεύτερη περίπτωση, υπάρχει πλεονασμός πληροφορίας)
 - είτε να δημιουργηθεί νέος πίνακας (όπως στην περίπτωση M:N)



Αναπαράσταση σύνθετων / πλειότιμων χαρακτηριστικών

5. Τα σύνθετα χαρακτηριστικά μετατρέπονται σε ένα σύνολο απλών
 - Παράδειγμα: έστω το σύνθετο χαρακτηριστικό name με συστατικά first-name και last-name. Ο πίνακας που προκύπτει θα έχει, μεταξύ άλλων, δύο χαρακτηριστικά name.first-name και name.last-name
6. Από ένα πλειότιμο χαρακτηριστικό ενός συνόλου οντοτήτων προκύπτει νέος πίνακας!
 - Ο πίνακας έχει ως στήλες το πρωτεύον κλειδί του συνόλου οντοτήτων και μια ακόμη που αντιστοιχεί στο πλειότιμο χαρακτηριστικό
 - Παράδειγμα: έστω το πλειότιμο χαρακτηριστικό phone του συνόλου οντοτήτων instructor (με πρωτεύον κλειδί ID). Προκύπτει ένας νέος πίνακας instructor-phones (ID, phone)



Αναπαράσταση εξειδίκευσης & συσσώρευσης

7. Αναπαράσταση εξειδίκευσης, 2 εναλλακτικές:

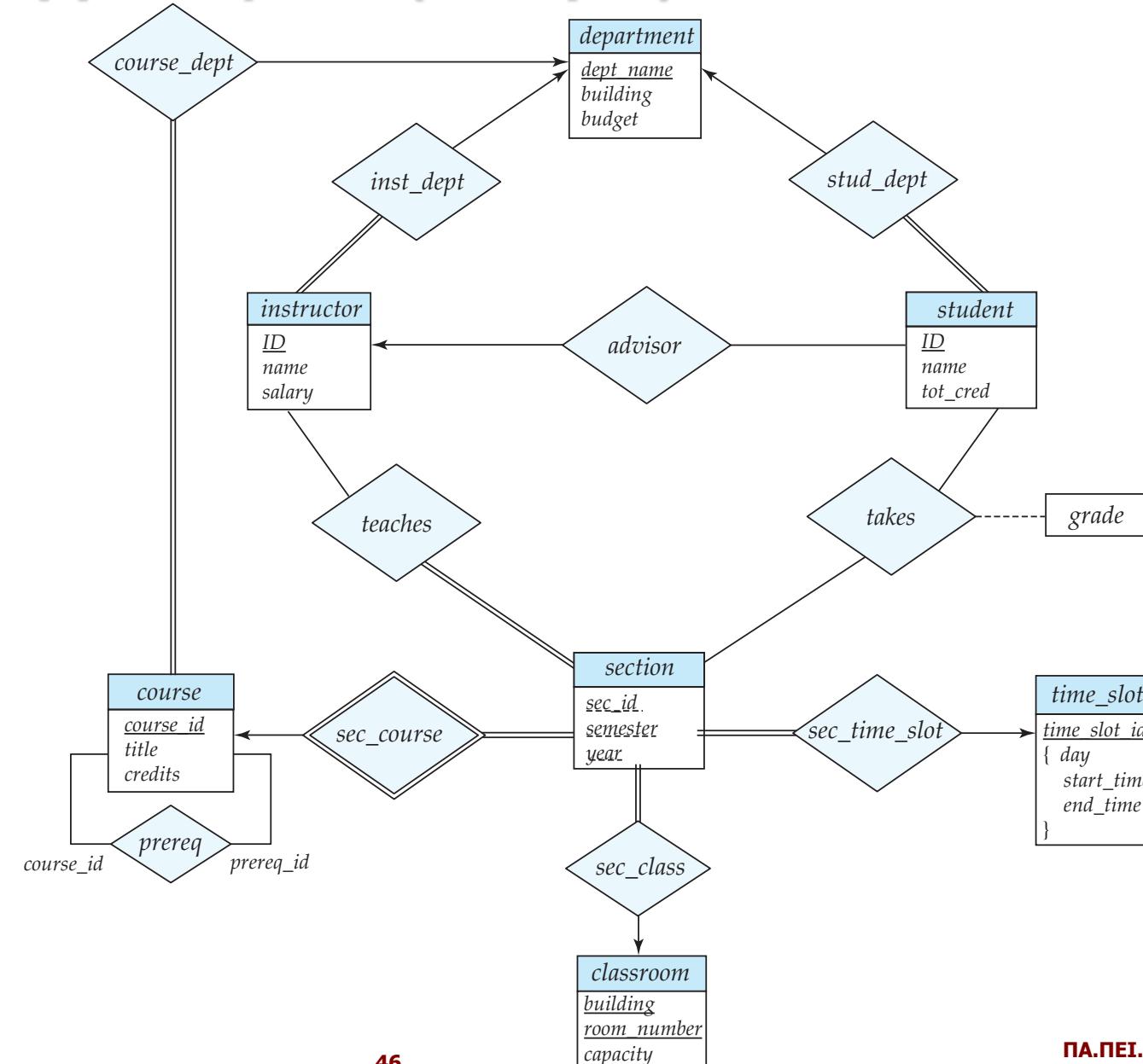
- 1^η μέθοδος: Προκύπτει ένας πίνακας για κάθε εμπλεκόμενο σύνολο οντοτήτων, όπου καθένας από τους πίνακες εξειδίκευσης συμπεριλαμβάνει ως στήλη το πρωτεύον κλειδί του πίνακα γενίκευσης
- 2^η μέθοδος: Προκύπτει ένας πίνακας για κάθε εμπλεκόμενο σύνολο οντοτήτων, όπου καθένας από τους πίνακες εξειδίκευσης συμπεριλαμβάνει ως στήλες **όλα** τα χαρακτηριστικά του συνόλου οντοτήτων ανώτερου επιπέδου (γενίκευσης)
- Εάν η εξειδίκευση είναι πλήρης, δεν απαιτείται ο πίνακας γενίκευσης

8. Αναπαράσταση συσσώρευσης: προκύπτει πίνακας με στήλες το πρωτεύον κλειδί της εμπλεκόμενης συσχέτισης, το πρωτεύον κλειδί του εμπλεκόμενου συνόλου οντοτήτων και τυχόν επιπλέον χαρακτηριστικά της συσσώρευσης

Παράδειγμα διαγράμματος E-R (Παν/μιο)



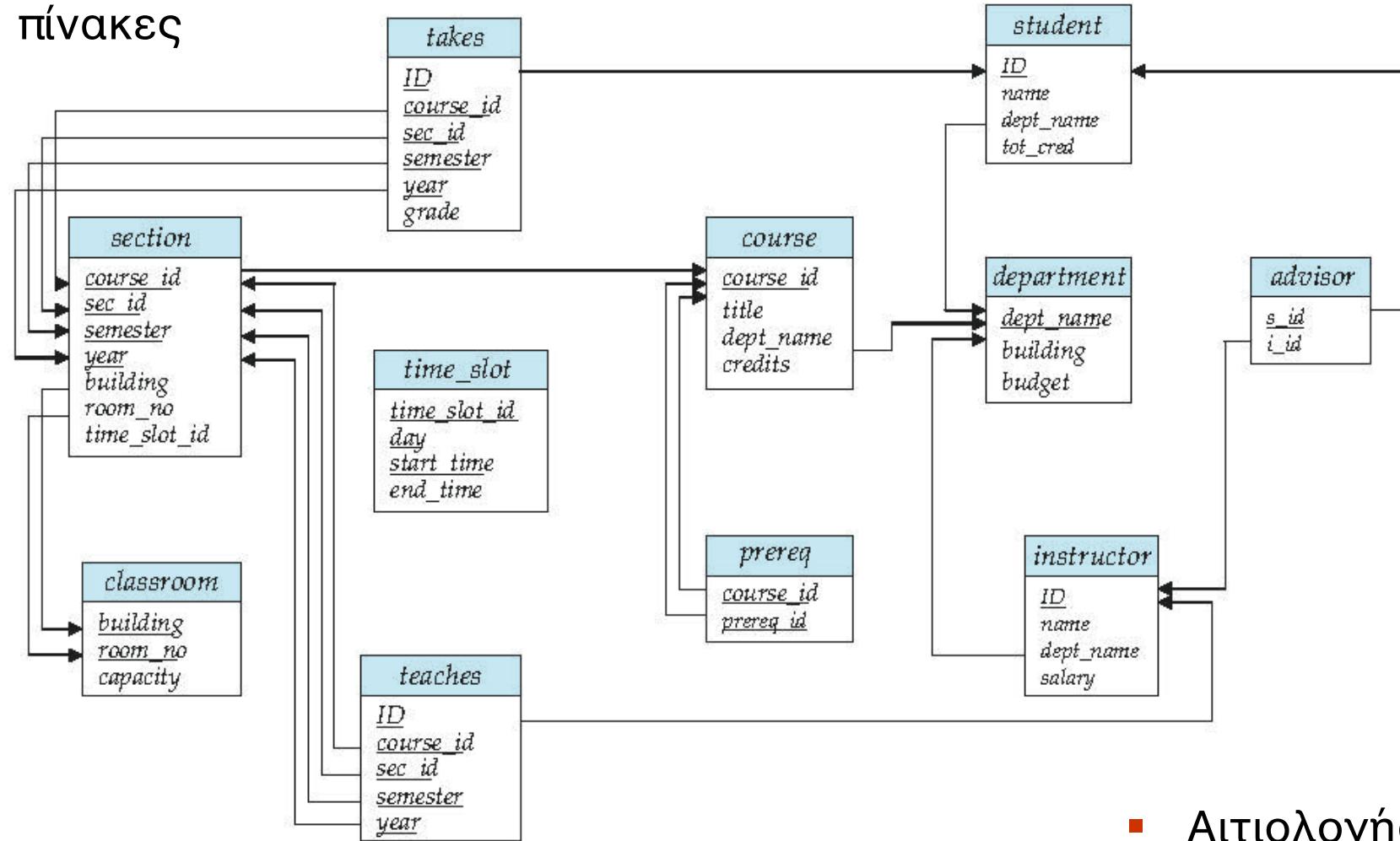
- 7 οντότητες
- 10 συσχετίσεις
(3 τύπου M:N
και 7 τύπου N:1)





Παράδειγμα σχεσιακού σχήματος (Παν/μιο)

- 11 πίνακες



- Αιτιολογήστε γιατί είναι 11 ...



4.3. Αντικείμενα στις ΒΔ



Περιεχόμενα

- Σύνθετοι τύποι δεδομένων και αντικειμενοστρέφεια
- Δομημένοι τύποι δεδομένων και κληρονομικότητα στην SQL
- Κληρονομικότητα πινάκων
- Τύποι πινάκων και πολλαπλών συνόλων στην SQL
- Ταυτότητα αντικειμένων και τύποι αναφοράς στην SQL
- Υλοποίηση αντικειμενο-σχεσιακών λειτουργιών

Αντικειμενικό-Σχεσιακό Μοντέλο Δεδομένων (Object-Relational Data Model)



- Επεκτείνει το σχεσιακό μοντέλο δεδομένων συμπεριλαμβάνοντας πρόσθετους τύπους δεδομένων και προσανατολισμό στο αντικείμενο.
- Επιτρέπει στα χαρακτηριστικά των εγγραφών να υποστηρίζουν σύνθετους, συμπεριλαμβανομένων μη-ατομικών (non-atomic) τιμών, όπως ένθετες σχέσεις (nested relations).
- Διατηρούνται τα σχεσιακά θεμέλια, ειδικότερα η δηλωτική πρόσβαση σε δεδομένα, επεκτείνοντας τη δύναμη μοντελοποίησης.
- Εξασφαλίζει συμβατότητα με τις υπάρχουσες σχεσιακές γλώσσες.

Σύνθετοι Τύποι Δεδομένων (Complex Data Types)



- Κίνητρο:
 - Επιτρέπουν μη-ατομικά πεδία (ατομικό ≡ αδιαίρετο).
 - Παράδειγμα μη-ατομικού πεδίου: μία συλλογή από ακέραιους ή μια συλλογή από εγγραφές.
 - Επιτρέπουν πιο απλούς τρόπους μοντελοποίησης για εφαρμογές που προϋποθέτουν σύνθετα δεδομένα.
- Ορισμός απλότητας:
 - Προβλέπεται η ύπαρξη σχέσεων για τις περιπτώσεις που επιτρέπονται ατομικές (βαθμωτές) τιμές — σχέσεις εντός σχέσεων.
 - Διατηρούνται τα μαθηματικά θεμέλια του σχεσιακού μοντέλου.
 - Παραβιάζεται η πρώτη κανονική μορφή (1NF).

Παράδειγμα 'Ένθετης Σχέσης



- Παράδειγμα: εφαρμογή βιβλιοθήκης
- Κάθε βιβλίο έχει
 - τίτλο,
 - μια λίστα (array) συγγραφέων,
 - εκδότη που περιέχει τα υποπεδία *name* και *branch*, και
 - ένα σύνολο από λέξεις-κλειδιά
- Η σχέση *books* δεν είναι σε πρώτη κανονική μορφή (Non-1NF).

<i>title</i>	<i>author_array</i>	<i>publisher</i>	<i>keyword_set</i>
		(<i>name</i> , <i>branch</i>)	
Compilers	[Smith, Jones]	(McGraw-Hill, New York)	{parsing, analysis}
Networks	[Jones, Frick]	(Oxford, London)	{Internet, Web}

4NF Αποσύνθεση 'Ενθετης Σχέσης

- Χάριν απλοποίησης, υποθέτουμε ότι ο τίτλος προσδιορίζει μοναδικά ένα βιβλίο.
 - Στην πραγματικότητα, ο αριθμός ISBN προσδιορίζει μοναδικά κάθε δημοσιευμένο βιβλίο
- Αποσυνθέτει το books σε 4NF σχεδίαση χρησιμοποιώντας τα σχήματα:
 - (title, author, position)
 - (title, keyword)
 - (title, pub-name, pub-branch)
- Η σχεδίαση 4NF απαιτεί από τους χρήστες να εκτελούν συνδέσεις (joins) για τη διατύπωση των ερωτημάτων τους.

<i>title</i>	<i>author</i>	<i>position</i>
Compilers	Smith	1
Compilers	Jones	2
Networks	Jones	1
Networks	Frick	2

authors

<i>title</i>	<i>keyword</i>
Compilers	parsing
Compilers	analysis
Networks	Internet
Networks	Web

keywords

<i>title</i>	<i>pub_name</i>	<i>pub_branch</i>
Compilers	McGraw-Hill	New York
Networks	Oxford	London

books4



Σύνθετοι Τύποι και SQL



- Επεκτάσεις που εισήχθησαν στην SQL:1999 για την υποστήριξη σύνθετων τύπων:
 - Τύποι συλλογής και μεγάλοι τύποι αντικειμένων
 - Οι ένθετες σχέσεις αποτελούν παράδειγμα τύπων συλλογής
 - Δομημένοι τύποι
 - Ένθετες δομές εγγραφής, όπως σύνθετα χαρακτηριστικά
 - Κληρονομικότητα
 - Προσανατολισμός στο αντικείμενο
 - Συμπεριλαμβανομένων προσδιοριστικών (identifiers) και αναφορών (references)
- Δεν εφαρμόζονται πλήρως στα υπάρχοντα ΣΔΒΔ
 - Ωστόσο, ορισμένα χαρακτηριστικά υποστηρίζονται στα περισσότερα εμπορικά συστήματα βάσεων δεδομένων
 - Βλ. το εγχειρίδιο χρήσης του εκάστοτε ΣΔΒΔ

Δομημένοι Τύποι και Κληρονομικότητα στην SQL



- **Δομημένοι τύποι (τύποι που ορίζονται από το χρήστη)** μπορούν να δηλωθούν και να χρησιμοποιηθούν στην SQL:

```
create type Name as
    (firstname      varchar(20),
     lastname       varchar(20))
```

final

```
create type Address as
    (street        varchar(20),
     city          varchar(20),
     zipcode       varchar(20))
```

not final

- Οι δομημένοι τύποι μπορούν να χρησιμοποιηθούν για τη δημιουργία σύνθετων χαρακτηριστικών σε μία σχέση:

```
create table person (
    name      Name,
    address   Address,
    dateOfBirth date)
```

- Τα συστατικά ενός σύνθετου χαρακτηριστικού (name.firstname) μπορούν να προσπελαστούν χρησιμοποιώντας μία σύνταξη «τελείας»

Σημείωση: οι προδιαγραφές **final** και **not final** υποδεικνύουν αν μπορούν να δημιουργηθούν υποτύποι

Δομημένοι Τύποι (συν.)



- **Τύποι γραμμών που ορίζονται από το χρήστη (User-defined row types)**

```
create type PersonType as (
```

```
    name Name,  
    address Address,  
    dateOfBirth date)
```

```
not final
```

- Στη συνέχεια, μπορούμε να δημιουργήσουμε έναν πίνακα, του οποίου οι γραμμές είναι ενός τύπου (ορισμένου από το χρήστη).

```
create table customer of CustomerType
```

- Εναλλακτικά, χρησιμοποιώντας **απροσδιόριστους τύπους γραμμών**:

```
create table person_r(
```

```
    name    row(firstname varchar(20),  
                lastname varchar(20)),  
    address row(street   varchar(20),  
                 city     varchar(20),  
                 zipcode varchar(20)),  
    dateOfBirth date)
```

Μέθοδοι (Methods)



- Μπορεί να προστεθεί μία δήλωση μεθόδου χρησιμοποιώντας έναν δομημένο τύπο:

```
method ageOnDate (onDate date)
    returns interval year
```

- Το σώμα της μεθόδου δημιουργείται χωριστά:

```
create instance method ageOnDate (onDate date)
    returns interval year
for CustomerType
begin
    return onDate - self.dateOfBirth;
end
```

- Μπορούμε να καλέσουμε τη μέθοδο ageOnDate() για να βρούμε την ηλικία κάθε πελάτη:

```
select name.lastname, ageOnDate (current_date)
from customer
```

Συναρτήσεις Δημιουργίας (Constructor Functions)



- **Οι συναρτήσεις δημιουργίας** χρησιμοποιούνται ώστε να δημιουργούν τιμές δομημένων τύπων.
- Παράδειγμα:

```
create function Name(firstname varchar(20), lastname varchar(20))
returns Name
begin
    set self.firstname = firstname;
    set self.lastname = lastname;
end
```

- Για να δημιουργήσουμε μία τιμή τύπου Name, εργαζόμαστε ως εξής:
new Name('John', 'Smith')
- Συνήθως, χρησιμοποιούνται σε προτάσεις εισαγωγής:
insert into Person values
(new Name('John', 'Smith),
new Address('20 Main St', 'New York', '11001'),
date '1960-8-22');

Κληρονομικότητα Τύπων (Type Inheritance)



- Υποθέστε ότι έχουμε τον παρακάτω ορισμό τύπου για τα άτομα:

```
create type Person
    (name varchar(20),
     address varchar(20))
```

- Μπορούμε να χρησιμοποιήσουμε κληρονομικότητα για να ορίσουμε τους τύπους για τους φοιτητές και τους καθηγητές:

```
create type Student
    under Person
    (degree      varchar(20),
     department  varchar(20))
```

```
create type Teacher
    under Person
    (salary      integer,
     department  varchar(20))
```

- Ένας υποτύπος μπορεί να ξαναορίσει το αποτέλεσμα μίας μεθόδου δηλώνοντας ξανά τη μέθοδο, χρησιμοποιώντας το **overriding method** (επικάλυψη μεθόδου) στη θέση του **method** στη δήλωση της μεθόδου.

Πολλαπλή Κληρονομικότητα (Multiple Type Inheritance)



- SQL:1999 και SQL:2003 δεν υποστηρίζουν πολλαπλή κληρονομικότητα.
- Αν το σύστημα τύπων υποστηρίζει πολλαπλή κληρονομικότητα, μπορούμε να ορίσουμε έναν τύπο για βοηθό καθηγητή ως εξής:

```
create type Teaching Assistant  
      under Student, Teacher
```
- Για να αποφευχθούν διενέξεις μεταξύ των δύο επαναλήψεων της σχέσης department, μπορούμε να τις μετονομάσουμε:

```
create type Teaching Assistant  
      under  
          Student with (department as student_dept ),  
          Teacher with (department as teacher_dept )
```
- Κάθε τιμή πρέπει να έχει έναν πιο συγκεκριμένο τύπο (**most-specific type**)

Κληρονομικότητα Πινάκων (Table Inheritance)



- Οι πίνακες που έχουν δημιουργηθεί από υποτύπους, μπορούν να οριστούν περεταίρω ως **υποπίνακες**
- π.χ. **create table** people **of** Person;
create table students **of** Student **under** people;
create table teachers **of** Teacher **under** people;
- Οι εγγραφές που προστίθενται σε έναν υποπίνακα είναι αυτόματα ορατές σε ερωτήματα που τίθενται στον υπερπίνακα
 - π.χ. το ερώτημα στον πίνακα people θα βρει τις εγγραφές που υπάρχουν στους πίνακες students και teachers.
 - Ομοίως, ενημερώσεις/διαγραφές στον πίνακα people οδηγούν επίσης σε ενημερώσεις/διαγραφές στους υποπίνακες.
 - Για την παράκαμψη αυτής της συμπεριφοράς, χρησιμοποιούμε “**only** people” στο ερώτημα.
- Εννοιολογικά, η πολλαπλή κληρονομικότητα είναι εφικτή με τη χρήση πινάκων
 - π.χ. teaching_assistants under students and teachers
 - Ωστόσο, δεν υποστηρίζεται πολλαπλή κληρονομικότητα πινάκων από την SQL.
 - Συνεπώς, δεν μπορούμε να δημιουργήσουμε μια εγγραφή στον πίνακα people που να είναι και φοιτητής και καθηγητής!

Απαιτήσεις Συνέπειας για τους Υποπίνακες



- Απαιτήσεις συνέπειας για τους υποπίνακες και τους υπερπίνακες.
 - Κάθε εγγραφή του υπερπίνακα (π.χ. *people*) μπορεί να αντιστοιχεί το πολύ σε μία εγγραφή σε κάθε άμεσο υποπίνακά του (π.χ. *students* και *teachers*).
 - Επιπλέον περιορισμός στην SQL:1999:
Όλες οι εγγραφές που αντιστοιχούν η μία στην άλλη (δηλαδή, με τις ίδιες τιμές για τα κληρονομημένα χαρακτηριστικά) πρέπει να παράγονται από μία εγγραφή (που έχει εισαχθεί σ' έναν πίνακα)
 - Ήτοι, κάθε οντότητα πρέπει να έχει έναν πιο συγκεκριμένο τύπο.
 - Δεν μπορούμε να έχουμε μια εγγραφή στον *people* που να αντιστοιχεί και σε μία εγγραφή του *students* και σε μία εγγραφή του *teachers*.

Τύποι Πινάκων και Πολλαπλών Συνόλων στην SQL



- Παράδειγμα ορισμού χαρακτηριστικών πίνακα (array) και πολλαπλών τιμών:

```
create type Publisher as
  (name      varchar(20),
   branch    varchar(20));

create type Book as
  (title      varchar(20),
   author_array  varchar(20) array [10],
   pub_date    date,
   publisher   Publisher,
   keyword-set varchar(20) multiset);

create table books of Book;
```

Δημιουργία Τιμών σε Συλλογές



- Δημιουργία πίνακα (array):
array ['Silberschatz', `Korth', `Sudarshan']
- Πολλαπλά σύνολα (multisets):
multiset ['computer', 'database', 'SQL']
- Δημιουργία μίας εγγραφής του τύπου που ορίζεται από τη σχέση books :
(‘Compilers’, **array**[` Smith', `Jones'],
new Publisher ('McGraw-Hill', `New York'),
multiset [` parsing', `analysis'])
- Εισαγωγή της προηγούμενης εγγραφής στη σχέση books:
insert into books
 values
(‘Compilers’, **array**[` Smith', `Jones'],
new Publisher ('McGraw-Hill', `New York'),
multiset [` parsing', `analysis']);

Ερωτήματα για Χαρακτηριστικά Συλλογών-Τιμές χαρακτηριστικών



- Εάν θέλουμε να βρούμε όλα τα βιβλία που έχουν τη λέξη “database” ως μία από τις λέξεις-κλειδιά τους:

```
select title  
from books  
where 'database' in (unnest(keyword-set ))
```

- Μπορούμε να έχουμε πρόσβαση σε μεμονωμένα στοιχεία ενός πίνακα με τη χρήση δεικτών (indices)

- π.χ.: Εάν ξέρουμε ότι ένα συγκεκριμένο βιβλίο έχει τρεις συγγραφείς:

```
select author_array[1], author_array[2], author_array[3]  
from books  
where title = 'Database System Concepts'
```

- Έστω ότι θέλουμε μία σχέση που περιέχει ζευγάρια της μορφής “title, author_name” για κάθε βιβλίο και κάθε συγγραφέα του βιβλίου:

```
select B.title, A.author  
from books as B, unnest (B.author_array) as A (author )
```

- Για να διατηρήσουμε τη σειρά διάταξης, προσθέτουμε τον όρο **with ordinality**:

```
select B.title, A.author, A.position  
from books as B, unnest (B.author_array) with ordinality as  
A (author, position )
```

Ακύρωση Ένθεσης (Unnesting)



- Ο μετασχηματισμός μίας ένθετης σχέσης σε μία μορφή με λιγότερα (ή καθόλου) χαρακτηριστικά με τιμές σχέσεων ονομάζεται ακύρωση ένθεσης (**unnesting**).
- Παράδειγμα:

```
select title, A as author, publisher.name as pub_name,
       publisher.branch as pub_branch, K.keyword
  from books as B, unnest(B.author_array ) as A (author ),
       unnest (B.keyword_set ) as K (keyword )
```
- Αποτέλεσμα της ακύρωσης της ένθεσης για τα χαρακτηριστικά `author_array` και `keyword_set` της σχέσης `books`

<i>title</i>	<i>author</i>	<i>pub_name</i>	<i>pub_branch</i>	<i>keyword</i>
Compilers	Smith	McGraw-Hill	New York	parsing
Compilers	Jones	McGraw-Hill	New York	parsing
Compilers	Smith	McGraw-Hill	New York	analysis
Compilers	Jones	McGraw-Hill	New York	analysis
Networks	Jones	Oxford	London	Internet
Networks	Frick	Oxford	London	Internet
Networks	Jones	Oxford	London	Web
Networks	Frick	Oxford	London	Web

Ένθεση (Nesting)



- Ένθεση (**Nesting**) είναι το αντίθετο του unnesting, δημιουργώντας ένα χαρακτηριστικό συλλογής τιμών.
- Η ένθεση λειτουργεί παρόμοια με μία συνοπτική συνάρτηση (aggregation), με τη διαφορά ότι χρησιμοποιεί τη συνάρτηση **collect()** στη θέση της συνοπτικής συνάρτησης για να δημιουργήσει πολλαπλά σύνολα (multiset).
- Για να κάνουμε ένθεση της σχέσης flat_books πάνω στο χαρακτηριστικό keyword:

```
select title, author, Publisher (pub_name, pub_branch ) as publisher,  
      collect (keyword) as keyword_set  
from flat_books  
group by title, author, publisher
```

- Αν θέλουμε να κάνουμε ένθετη την ιδιότητα authors σ' ένα multiset:

```
select title, collect (author ) as author_set,  
      Publisher (pub_name, pub_branch) as publisher,  
      collect (keyword ) as keyword_set  
from flat_books  
group by title, publisher
```

'Ενθεση (Nesting) (συν.)



- Μια άλλη προσέγγιση στη δημιουργία ένθετων σχέσεων είναι να χρησιμοποιηθούν υπο-ερωτήματα στον όρο **select**, ξεκινώντας από τη σχέση books (4NF):

select title,

array (select author
 from authors **as** A
 where A.title = B.title

order by A.position) **as** author_array,
 Publisher (pub-name, pub-branch) **as** publisher,
 multiset (select keyword
 from keywords **as** K
 where K.title = B.title) **as** keyword_set
from books4 **as** B

Ταυτότητα Αντικειμένων και Τύποι Αναφοράς



- Ορίζουμε έναν τύπο Department μ' ένα πεδίο name και ένα πεδίο head, που είναι μια αναφορά στον τύπο Person (η αναφορά περιορίζεται σε εγγραφές του πίνακα people):

```
create type Department (
    name varchar (20),
    head ref (Person) scope people)
```

- Στη συνέχεια, μπορούμε να ορίσουμε τον πίνακα ως εξής:

```
create table departments of Department
```

- Μπορούμε να παραλείψουμε τη δήλωση **scope** people από τη δήλωση του τύπου και αντίθετα να κάνουμε μια προσθήκη στην εντολή **create table**:

```
create table departments of Department
    (head with options scope people)
```

- Ο αναφερόμενος πίνακας πρέπει να έχει ένα χαρακτηριστικό το οποίο να αποθηκεύει το προσδιοριστικό της εγγραφής, που ονομάζεται αυτο-αναφερόμενο χαρακτηριστικό (**self-referential attribute**):

```
create table people of Person
    ref is person_id system generated;
```

Αρχικοποίηση Χαρακτηριστικών Αναφοράς



- Για να δημιουργήσουμε μία εγγραφή με την τιμή αναφοράς, πρέπει πρώτα να δημιουργήσουμε την εγγραφή με μία κενή (null) αναφορά και μετά να ορίσουμε την αναφορά ξεχωριστά:

```
insert into departments
values ('CS', null)
update departments
set head = (select p.person_id
            from people as p
            where name = 'John')
where name = 'CS'
```

Προσδιοριστικά που Δημιουργούνται από το Χρήστη



- Ο τύπος του αυτο-αναφερόμενου χαρακτηριστικού πρέπει να καθορισθεί ως μέρος του ορισμού του τύπου του αναφερόμενου πίνακα, και
- Ο ορισμός του πίνακα πρέπει να καθορίζει ότι η αναφορά δημιουργείται από το χρήστη:

```
create type Person
  (name varchar(20)
   address varchar(20))
  ref using varchar(20)
create table people of Person
  ref is person_id user generated
```

- Όταν θα εισάγεται μία εγγραφή στο people, θα πρέπει να παρέχουμε μία τιμή για το προσδιοριστικό:

```
insert into people (person_id, name, address ) values
  ('01284567', 'John', '23 Coyote Run')
```

- Μπορούμε μετά να χρησιμοποιήσουμε την τιμή του προσδιοριστικού όταν εισάγεται μία εγγραφή στο departments
 - Με αυτόν τον τρόπο, αποφεύγεται η ανάγκη για ένα ξεχωριστό ερώτημα, το οποίο να ανακαλεί το προσδιοριστικό:

```
insert into departments
values('CS', '02184567')
```

Προσδιοριστικά που Δημιουργούνται από το Χρήστη (συν.)



- Είναι ακόμη δυνατόν να χρησιμοποιηθεί ως προσδιοριστικό μία τιμή από ένα υπάρχον πρωτεύον κλειδί:

```
create type Person
  (name varchar (20) primary key,
   address varchar(20))
  ref from (name)
create table people of Person
  ref is person_id derived
```

- Όταν εισάγεται μία εγγραφή στο departments, μπορούμε να χρησιμοποιήσουμε το:

```
insert into departments
values(`CS`,`John`)
```

Εκφράσεις Διαδρομής (Path Expressions)



- Βρείτε τα ονόματα και τις διευθύνσεις των επικεφαλής όλων των τμημάτων:

```
select head->name, head->address  
from departments
```
- Μία έκφραση, όπως “head->name” ονομάζεται έκφραση διαδρομής (**path expression**).
- Οι εκφράσεις διαδρομής μπορούν να χρησιμοποιηθούν αντί για συνδέσμους (explicit joins)
 - Εάν οι επικεφαλής των τμημάτων δεν είχαν καθοριστεί ως αναφορά, θα έπρεπε να χρησιμοποιήσουμε έναν άμεσο σύνδεσμο των σχέσεων departments και people, για να βρούμε τη διεύθυνση των επικεφαλής.
 - Η χρήση αναφορών απλοποιεί αρκετά το ερώτημα.



Υλοποίηση Αντικειμενο-Σχεσιακών Λειτουργιών

- Ο τρόπος λειτουργίας είναι παρόμοιος με τον αντίστοιχο που χρησιμοποιείται για τη μετάφραση (σε επίπεδο σχέσεων) μερικών λειτουργιών του μοντέλου E-R (οντότητας-σχέσης).
- Υλοποίηση υποπίνακα
 - Κάθε πίνακας αποθηκεύει το πρωτεύον κλειδί και τα χαρακτηριστικά που ορίζονται τοπικά
 - Κάθε πίνακας αποθηκεύει όλα τα κληρονομημένα χαρακτηριστικά, καθώς και τα χαρακτηριστικά που ορίζονται τοπικά.



Τέλος Ενότητας 4