



EPL342 –Databases

Lecture 14: SQL DML I

SQL Structured Query Language

(Chapter 6.4, Elmasri-Navathe 5ED)

Demetris Zeinalipour

<http://www.cs.ucy.ac.cy/courses/EPL342>

Περιεχόμενο Διάλεξης



Κεφάλαιο 6.4: SQL DML I

- **Εισαγωγή στην SQL-DML** (Διαφορές SQL-DML με Σχεσιακό Μοντέλο / Άλγεβρα).
- **Άπλες Εκφράσεις** σε SQL (SELECT-FROM-WHERE)
- **Απλές Συνενώσεις** σε SQL
- **Διφορούμενα Ονόματα** Γνωρισμάτων και το Γνώρισμα *
- Περίπτωση **Μη-προσδιορισμένου WHERE**
- Χρήση **DISTINCT** σε γνωρίσματα.
- **Πράξεις Συνόλων** με SQL

Εισαγωγή στη SQL



- Στις προηγούμενες δυο διαλέξεις ξεκινήσαμε την μελέτη μιας **πραγματική** γλώσσα βάσεων δεδομένων την **SQL (Structured Query Language)**.
 - Γλώσσα Ορισμού Δεδομένων (*Data Definition Language*, **SQL-DDL**)
 - Γλώσσα Επεξεργασίας Δεδομένων (*Data Manipulation Language*, **SQL-DML**)
- Εάν και θα αναφερόμαστε στο πρότυπο **SQL:99-DML**, αυτό **υλοποιείται σε μεγάλο βαθμό** από τους **κατασκευαστές** βάσεων δεδομένων (π.χ., στην TSQL)
 - Αυτό είναι **σε αντίθεση με την SQL-DDL** η οποία αντιμετωπίζει αρκετά προβλήματα συμβατότητας μεταξύ κατασκευαστών.
- Σε επερωτήσεις SQL συνιστάται η συμβατότητα με το πρότυπο ANSI/ISO SQL:99
 - Παρόλο που πολλές φορές θα σας δημιουργείται ένα δίλλημα μεταξύ **Επίδοσης (Performance)** και **Μεταφερσιμότητας (Portability)** του κώδικα.

Διαφορές της SQL και του Σχεσιακού Μοντέλου/Άλγεβρας:



Διαφορές SQL και Σχεσιακού Μοντέλου / Άλγεβρας

- **A)** Η SQL στηρίζεται σε **Σύνολα-μη-Διακριτών-Τιμών** ή αλλιώς **Πολυσύνολα (Multi-set)**, ενώ το Σχεσιακό Μοντέλο / Άλγεβρα σε **Απλά Σύνολα (διακριτών-τιμών)**.
 - Συνεπώς, η SQL επιτρέπει τα **διπλότυπα (duplicates)** σε **σχέσεις** και **αποτελέσματα** επερωτήσεων ενώ το Σχεσιακό Μοντέλο / Άλγεβρα όχι.
 - Σημειώστε ότι οι **σχέσεις SQL** μπορούν να **περιοριστούν** έτσι ώστε να **συμπεριφέρονται ως μαθηματικά σύνολα**
 - κάνοντας χρήση περιορισμών **PRIMARY KEY, UNIQUE**, ή με χρήση του **DISTINCT** το οποίο θα δούμε σε λίγο
- **B)** **Σχέσεις SQL** και **αποτελέσματα** έχουν **διάταξη (order)** ενώ στο Σχεσιακό Μοντέλο / Άλγεβρα δεν έχουν, δηλ.,
 - **Επίπεδο Σχέσης:** Σειρά αποθήκευσης στοιχείων στον δίσκο
 - **Επίπεδο Επερώτησης:** Αύξουσα Σειρά / Φθίνουσα Σειρά

Διαφορές της SQL και του Σχεσιακού Μοντέλου/Άλγεβρας:



- Ένα **πολυσύνολο (multi-set or bag)** είναι ένα **μη-διατεταγμένο** σύνολο στοιχείων, όπου ένα στοιχείο μπορεί να εμφανίζεται **περισσότερο από μια φορά**.
 - Παράδειγμα: $\{A, B, C, A\}$ είναι ένα **πολυσύνολο**, ενώ το $\{A, B, C\}$ είναι **πολυσύνολο** και **απλό σύνολο**.
 - Η SQL παράγει πολυσύνολα στα οποία **υπάρχει διάταξη** (κάποια σειρά) στα **αποτελέσματα (ουσιαστικά λίστες)**.
- Παράδειγμα:
 - $\{A, B\} = \{B, A, A\}$ ως Σύνολα
 - $\{A, B, A\} = \{B, A, A\}$ ως Πολυσύνολα
 - $[A, B, A] \neq [B, A, A]$ ως Λίστες (που παράγονται στην SQL)

Απλές Επερωτήσεις σε SQL



- Η βασική έκφραση SQL για διατύπωση επερωτήσεων ονομάζεται **SELECT-FROM-WHERE** μπλοκ (ή *mapping*)

SELECT <attribute list>

FROM <table list>

[**WHERE** <condition>]

- **<attribute list>** Είναι μια **λίστα γνωρισμάτων** των οποίων η τιμή πρέπει να ανακτηθεί από μια επερώτηση.
 - Αντίστοιχο του **τελεστή προβολής π.**
- **<table list>** είναι μια **λίστα από ονόματα σχέσεων** από τα οποία θα γίνει η ανάκτηση των αποτελεσμάτων.
 - Αντίστοιχο της σχέσης που συμμετέχει σε ένα Σχεσιακό Τελεστή.
- **<condition>** είναι μια **λογική έκφραση** (Boolean) ή οποία πρέπει να αποτιμηθεί για να επιστραφούν τα αποτελέσματα
 - Αντίστοιχο του **τελεστή επιλογής σ.**

- Τα αποτελέσματα επιστρέφονται κάποτε σε αύξουσα σειρά του attribute list (όχι σε SQL Server).

- Θα μιλήσουμε περισσότερο για το Ordering στην επόμενη διάλεξη

Άπλες Επερωτήσεις SQL (Simple SQL Queries)



- Η SQL μπορεί να χρησιμοποιηθεί για να δηλωθούν πολύ **σύνθετες** και **περίπλοκες** επερωτήσεις.
- Σήμερα θα επικεντρωθούμε μόνο στα **bold** μέρη:

SELECT attribute list

FROM table list

WHERE selection-condition

GROUP BY grouping attribute(s)

HAVING grouping condition

ORDER BY ASC | DESC

- Το απλό **SELECT-FROM-WHERE** μπλοκ θα επεκταθεί αργότερα με έννοιες **ομαδοποίησης, συναθροιστικών συναρτήσεων, εμφωλευμένων επερωτήσεων**, κ.α.
- TSQL SELECT-FROM-WHERE Reference:

<http://msdn.microsoft.com/en-us/library/ms189499.aspx>

Απλές Επερωτήσεις (Επερώτηση σε 1 Σχέση)



- **Query 0:** Ανάκτηση το **birthdate** και **address** των employee των οποίων το **name** είναι 'John B. Smith'.

```
Q0: SELECT      BDATE, ADDRESS
      FROM      EMPLOYEE
      WHERE     FNAME='John' AND MINIT='B'
              AND LNAME='Smith'
```

- Αντίστοιχο με την έκφραση σχεσιακής άλγεβρας
 $T1 \leftarrow \sigma_{(FNAME='John' \text{ AND } MINIT='B' \text{ AND } LNAME='Smith')}(EMPLOYEE)$
 $T2 \leftarrow \pi_{BDATE, ADDRESS}(T1)$
- Η **προβολή (π)** παρουσιάζεται με Μπλε ενώ οι **επιλογή (σ)** με Κόκκινο.
- Σημειώστε ότι στο Q0 μπορεί να εμπεριέχονται **διπλότυπα** ενώ στην αντίστοιχη έκφραση Σχεσιακής Άλγεβρας δεν υπάρχουν

Απλές Επερωτήσεις (Επερώτηση σε 2 Σχέσεις)



- **Query 1:** Ανάκτησε τα **names** και **addresses** όλων των υπαλλήλων που δουλεύουν για το 'Research' department.

- Q1: SELECT **FNAME, LNAME, ADDRESS**
FROM EMPLOYEE, DEPARTMENT
WHERE **DNAME='Research'** AND
DNUMBER=DNO

- Αντίστοιχο με την έκφραση σχεσιακής άλγεβρας

$T1 \leftarrow \text{EMPLOYEE} \otimes_{DNO=Dnumber} (\sigma_{DNAME='Research'} \text{Department})$

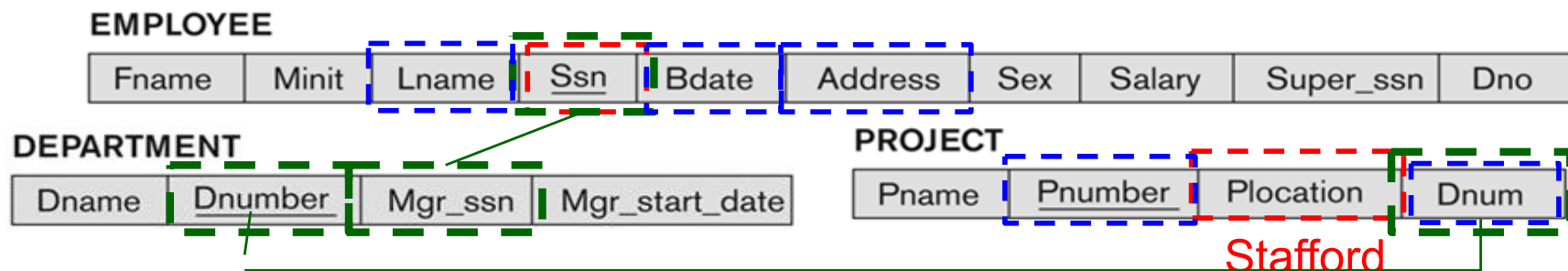
$T2 \leftarrow \pi_{FNAME, LNAME, ADDRESS}(T1)$

- **DNAME='Research'**: Συνθήκη Επιλογής (σ)
- **DNUMBER=DNO**: Συνθήκης Συνένωσης (\otimes)
- **FNAME,LNAME,ADDRESS**: Γνωρίσματα Προβολής (π)

Απλές Επερωτήσεις (Επερώτηση σε 3 Σχέσεις)



- **Query 2:** Για κάθε project στην πόλη 'Stafford', παρουσιάσε το project number, το τμήμα που ελέγχει το project και τα ακόλουθα: **επίθετο (last name)** του department manager, **διεύθυνση (address)** και ημερομηνία γέννησης (**birthdate**).



Q2: SELECT PNUMBER,DNUM,LNAME,BDATE,ADDRESS
FROM PROJECT, DEPARTMENT, EMPLOYEE
WHERE **DNUM=DNUMBER** AND **MGRSSN=SSN**
AND **PLOCATION='Stafford'**

Σχεσιακή Άλγεβρα

$T1 \leftarrow \pi_{Pnumber, Dnum}(\sigma_{PLOCATION='Stafford'} Project)$

$T2 \leftarrow \pi_{Pnumber, Dnum, Mgr_ssn}(Department \otimes_{Dnumber=DNum} T1)$

$T2 \leftarrow \pi_{Pnumber, Dnum, Lname, Bdate, Address}(Employee \otimes_{SSN=Mgr_ssn} T2)$

Διφορούμενα Ονόματα Γνωρισμάτων (Ambiguous Attribute Names)

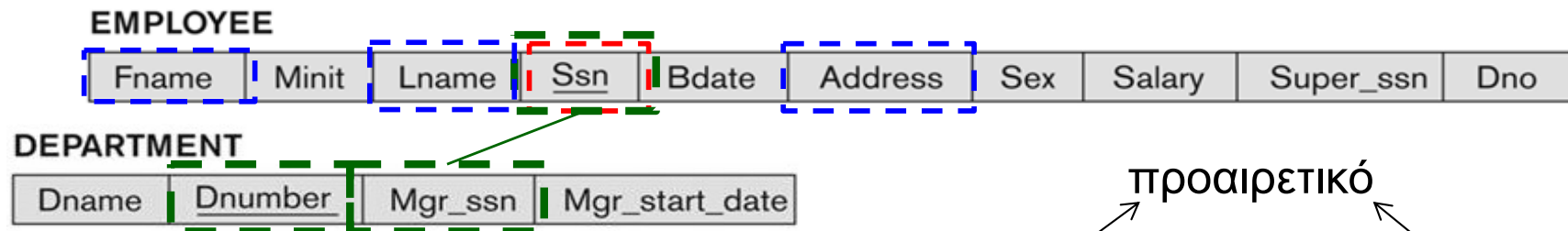


- Στην SQL, μπορούμε να χρησιμοποιήσουμε το **ίδιο όνομα** για δυο ή **περισσότερα γνωρίσματα** εφόσον τα γνωρίσματα αυτά ανήκουν σε **διαφορετικές σχέσεις**
 - Π.χ., Employee(ssn, name, dno) και Department(dno, name)
- Για να αποφευχθούν **διφορούμενες καταστάσεις** σε περιπτώσεις πολλαπλών σχέσεων (στις οποίες δεν θα είναι γνωστό σε πιο ακριβώς γνώρισμα αναφέρεται η επερώτηση), χρησιμοποιούμε ως **πρόθεμα (prefix)** το όνομα της σχέσης, π.χ.,
 - **EMPLOYEE.Name, DEPARTMENT.Name**
 - ή ακόμη καλύτερα να χρησιμοποιήσουμε ένα alias (δες επομένη διαφάνεια)

Διφορούμενα Ονόματα Γνωρισμάτων (Ambiguous Attribute Names)



- **Aliases** για μετονομασία Σχέσεων/Γνωρισμάτων



- Q1a: SELECT **E.FNAME, E.LNAME, E.ADDRESS**
FROM **EMPLOYEE [AS] E, DEPARTMENT [AS] D**
WHERE **D.DNAME='Research' AND**
D.DNUMBER=E.DNO

- Ακόμη και εάν **δεν υπάρχουν διφορούμενα** ονόματα, είναι καλό να χρησιμοποιείται το **Alias** γιατί έτσι γίνεται πιο **ευανάγνωστος** ο κώδικας SQL.

- Για μετονομασία γνωρισμάτων (όχι σε TSQL)

FROM Employee AS E(Fn,Mi,Ln,Ssn,Bd,Addr,Sex,Sal,Sssn,Dno)

Διφορούμενα Ονόματα Γνωρισμάτων (Ambiguous Attribute Names)



- Μερικές ερωτήσεις αναφέρονται στην ίδια σχέση πολλαπλές φορές.
 - Σε αυτή την περίπτωση, ανατίθενται **διαφορετικά *aliases*** στο όνομα της σχέσης έτσι ώστε να δίνεται η εντύπωση ότι υπάρχουν διαφορετικά στιγμιότυπα.
- **Query 8:** Για κάθε employee, ανάκτησε το όνομα του employee, και το όνομα του άμεσα προϊσταμένου του.



- Q8: SELECT **E.FNAME, E.LNAME, S.FNAME, S.LNAME**
FROM EMPLOYEE E, EMPLOYEE S
WHERE **E.SUPERSSN=S.SSN**
 - E: ρόλος υφιστάμενου (*supervisee*)
 - S: ρόλος προϊσταμένου (*supervisor*)

Μη-προσδιορισμένο WHERE



- Ο όρος *WHERE* είναι προαιρετικός σε μια επερώτηση SQL.
SELECT <attribute list>
FROM <table list>
[WHERE <condition>]
- Η μη-ύπαρξη ενός τέτοιου όρου υποδηλώνει ότι ΔΕΝ υπάρχει συνθήκη επιλογής.
 - Ουσιαστικά, είναι αντίστοιχο της συνθήκης WHERE TRUE
- Συνεπώς, όλες οι πλειάδες μιας σχέσης επιλέγονται σε μια τέτοια περίπτωση.

• **Query 9:** Ανάκτησε το SSN όλων των υπαλλήλων.

– Q9: SELECT SSN
 FROM EMPLOYEE

Μη-προσδιορισμένο WHERE (Περίπτωση Καρτεσιανού Γινομένου)



- Εάν ορίζονται περισσότερο από μια σχέσεις στον όρο FROM, τότε αυτό υποδηλώνει τον τελεστή **Καρτεσιανού Γινομένου (Cartesian Product)**

- **Παράδειγμα:**

```
Q10:      SELECT SSN, DNAME
           FROM EMPLOYEE, DEPARTMENT
```

- Είναι πολύ σημαντικό να **μην παραβλέπεται** ο ορισμός συνθηκών **επιλογής (WHERE)** σε **συνενώσεις** για να μην δημιουργούνται μη-επιθυμητά μεγάλα αποτελέσματα.
- Κάποιες βάσεις υλοποιούν **εξειδικευμένους τελεστές καρτεσιανού γινομένου**
 - Π.χ. στην TSQL υπάρχει η εντολή **CROSS JOIN**
`SELECT SSN, DNAME FROM EMPLOYEE CROSS JOIN DEPARTMENT`
 - Καλύτερα ωστόσο να χρησιμοποιείται η ANSI/ISO εντολή (Q10)

Χρήση Γνωρίσματος-*



- Για να ανακτήσουμε ΌΛΑ τα γνωρίσματα μιας πλειάδας σε μια επερώτηση κάνουμε χρήση του **γνωρίσματος *** (all attributes)

- **Παραδείγματα:**

Q1C: SELECT *

 FROM EMPLOYEE

 WHERE DNO=5

Q1D: SELECT *

 FROM EMPLOYEE, DEPARTMENT

 WHERE DNAME='Research' AND

 DNO=DNUMBER

Χρήση DISTINCT σε Γνωρίσματα



«**DISTINCT** *»: όλες οι στήλες μαζί να είναι

μοναδικές (ίδιο με «*» όταν μια σχέση έχει PK)

- Όπως είχαμε αναφέρει νωρίτερα, η SQL χειρίζεται τις σχέσεις ως Πολυσύνολα με διάταξη, συνεπώς είναι δυνατό να υπάρχουν **διπλότυπα (duplicate tuples)**.

- Για να **εξαλείψουμε τα διπλότυπα** σε μια επερώτηση, κάνουμε χρήση της λέξης **DISTINCT** στον όρο SELECT.

Παράδειγμα

Q11: SELECT
FROM

SALARY
EMPLOYEE

Q11A: SELECT
FROM

DISTINCT SALARY
EMPLOYEE

Employee:

EMPLOYEE	FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
John	B		Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T		Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J		Zelaya	999887777	1968-07-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S		Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K		Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A		English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V		Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E		Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	null	1

Q11: **Q11A:**

Salary
30000
40000
25000
43000
38000
25000
25000
55000

Salary
30000
40000
25000
43000
38000
55000

Χρήση DISTINCT σε Γνωρίσματα

- Σημειώστε ότι πέρα από το **SELECT DISTINCT** υπάρχει και το **SELECT ALL**, το οποίο ΔΕΝ αφαιρεί τα διπλότυπα.
- Συγκεκριμένα,

```
SELECT [DISTINCT | ALL] <attribute-list>  
FROM <table-list>
```

- Το **SELECT ALL** αντιπροσωπεύει ουσιαστικά την εξορισμού λειτουργία του **SELECT**.
- Όπως αναφέραμε και σε προηγούμενες διαλέξεις, στην **SQL** πολλά πράγματα δηλώνονται ρητά γιατί έτσι:
 - Ξεκαθαρίζει η πρόθεση του σχεδιαστή
 - Αποφεύγονται προβλήματα συμβατότητας που μπορεί να προκύψουν από την μεταφορά της υλοποίησης σε άλλη βάση δεδομένων.

Πράξεις Συνόλων σε SQL

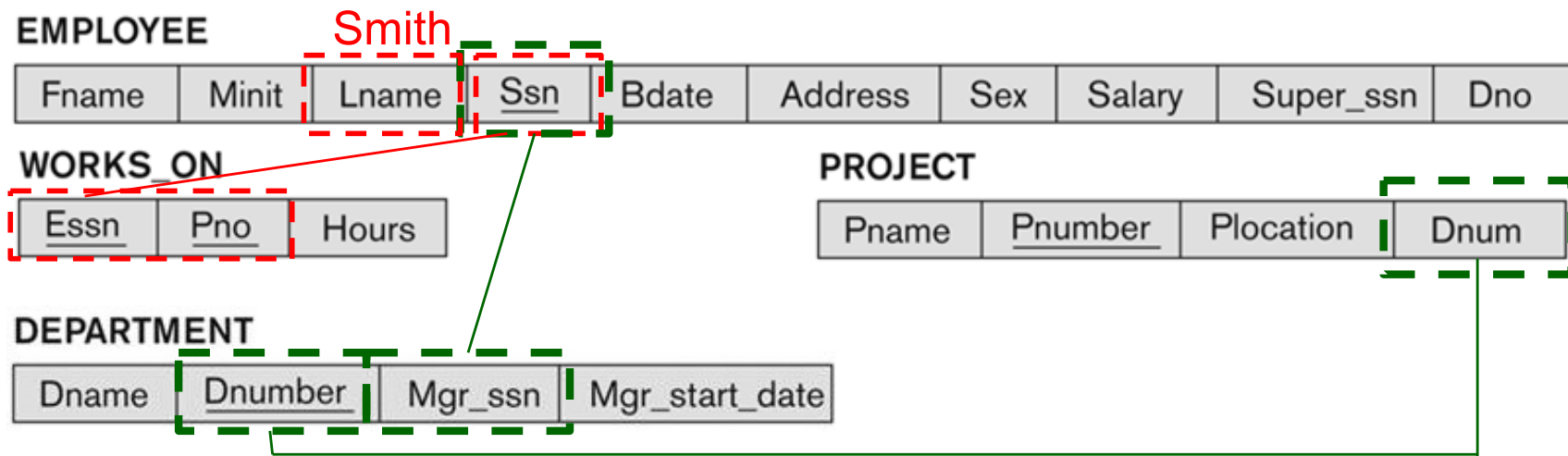


- Η SQL:99 υλοποιεί αρκετές **πράξεις συνόλων** τις οποίες ορίσαμε στα πλαίσια της Σχεσιακής Άλγεβρας.
- Συγκεκριμένα, υποστηρίζονται οι ακόλουθες πράξεις:
 - Ένωση: **UNION [ALL]**
 - Τομή: **INTERSECT**
 - Διαφορά Συνόλων: **EXCEPT**
- Προϋποθέτουν ότι τα σύνολα είναι i) **συμβατά-προς-τον-τύπο** και ii) η **διάταξη των γνωρισμάτων** είναι η ίδια (δεν χρειάζεται να έχουν το ίδιο όνομα)
- **Διαφορά Πράξεων Συνόλων από άλλες πράξεις SQL:**
 - Τα αποτελέσματα είναι **ΣΥΝΟΛΑ** όχι **ΠΟΛΥΣΥΝΟΛΑ** (συνεπώς δεν υπάρχουν διπλότυπα στις πράξεις αυτές)
 - Θα δούμε πως παράγονται Πολυσύνολα σε λίγο.

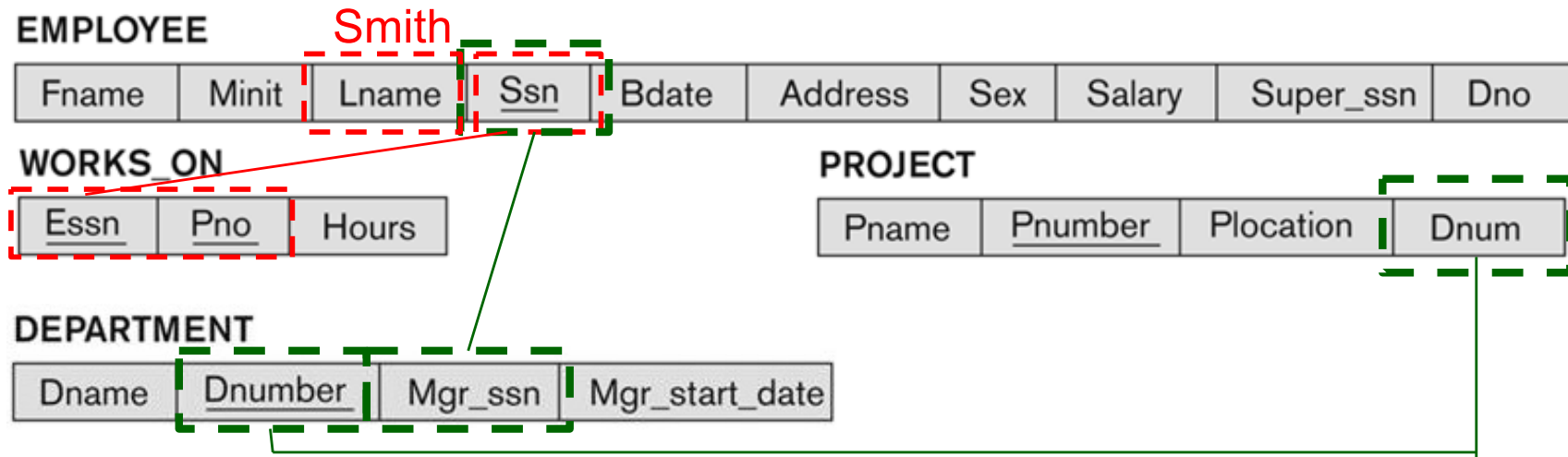
Πράξεις Συνόλων σε SQL



- Query 4: Δημιουργήστε μια λίστα από **projects** τα οποία περιλαμβάνουν ένα **υπάλληλο** με το επίθετο “**Smith**”, ως **υπάλληλο** ή* ως **manager** του τμήματος που ελέγχει το εν λόγω **project**.



Πράξεις Συνόλων σε SQL



Q4: (SELECT W.Pno
FROM EMPLOYEE E, WORKS_ON W
WHERE W.ESSN=E.SSN AND E.LNAME='Smith')

(SMITH_WOR
KER_PROJS)

UNION

(SELECT P.Pnumber
FROM PROJECT P, DEPARTMENT D, EMPLOYEE E
WHERE P.DNUM=D.DNUMBER AND
D.MGRSSN=E.SSN AND
E.LNAME='Smith')

(SMITH_MGR
_PROJS)

Πράξεις Συνόλων σε SQL



- Για να επιστραφούν αποτελέσματα Πολυσυνόλων, αντί αποτελέσματα συνόλων, μπορεί να γίνει χρήση των ακόλουθων εντολών:

- **UNION ALL**, υποστηρίζεται και σε TSQL

Π.χ., {(1,Pet), (2,Cat)} **UNION ALL** {(2,Cat),(1, Pet)}

Επιστρέφει: { (1,Pet), (2,Cat), (2,Cat), (1, Pet) }

- **INTERSECT ALL (δεν υλοποιείται σε TSQL)**

Π.χ., {(1,Pet), (2,Cat)} **INTERSECT ALL** {(1, Pet)}

Επιστρέφει: { (1,Pet), (1, Pet) }

- **EXCEPT ALL (δεν υλοποιείται σε TSQL)**

Π.χ., {(1,Pet), (2,Cat), (2,Cat)} **EXCEPT ALL** {(2, Cat)}

Επιστρέφει:{(1,Pet), (2,Cat) }

- Συνεπώς, με τις πιο πάνω πράξεις δεν διαγράφονται τα διπλότυπα από ένα αποτέλεσμα¹⁴⁻²²

Πράξεις Συνόλων σε SQL



- Σημειώστε ότι εάν ένας τελεστής **δεν υλοποιείται στην SQL** τότε μπορεί να **υλοποιηθεί με χρήση βασικών τελεστών** (π.χ., Συμμετρική Διαφορά, $R \oplus S = (R - S) \cup (S - R)$)
- Στην συνέχεια θα δούμε ότι υπάρχουν **πολλές άλλες πράξεις σύγκρισης με σύνολα** που χρησιμοποιούνται για διατύπωση ερωτήσεων σε SQL
 - **IN, ANY, ALL, CONTAINS, EXISTS, NOT EXISTS, κτλ.**