Βάσεις Δεδομένων

Σχεσιακή Σχεδίαση ΒΔ:
Συναρτησιακές Εξαρτήσεις

SQL Συναθροιστικές Συναρτήσεις

Συναθροιστικές συναρτήσεις(Aggregate functions)

- **Είσοδος**: ένα σύνολο τιμών ή πλειάδων
- **Έξοδος**: μία τιμή

- Στην SQL οι βασικές συναρτήσεις είναι:
- **COUNT**([DISTINCT] x) — αριθμός των [μοναδικών] τιμών
  - Ή COUNT(*) ή COUNT (DISTINCT x)
- **SUM**([DISTINCT] x) — άθροισμα των " "
- **AVG**([DISTINCT] x) — μέσος όρος των " "
- **MIN**(x) — ελάχιστη τιμή
- **MAX**(x) — μέγιστη τιμή

Συναθροιστικές συναρτήσεις:COUNT(*)-παράδειγμα

- Αριθμός των φοιτητών:
- **SELECT**COUNT(*) **FROM** ΦΟΙΤΗΤΕΣ;
  - Το αποτέλεσμα θα είναι ένας πίνακας:

Count(*)
52

- Εφαρμόζεται ό,τι ξέρουμε ως τώρα για το FROM… WHERE(δηλ. η υπόλοιπη εκφραστικότητα της SQLπαραμένει σε ισχύ) και μετά εφαρμόζουμε τις σ.σ. στις πλειάδες που μένουν, μετά από κάποιες πιθανές προβολές.

Συναθροιστικές συναρτήσεις: SUM() - παράδειγμα

- Η συνάρτηση SUM() επιστρέφει το άθροισμα (αριθμητικό) από μια στήλη (πεδίο):
- **SELECT** SUM(Τιμή) **FROM** ΠΑΡΑΓΓΕΛΙΕΣ;

κωδπ	Ημερομηνία	Τιμή	Όνομα
1	2010/11/12	1000	Ελένη
2	2010/10/23	1600	Πάρις
3	2010/09/02	700	Μενέλαος
4	2010/09/03	300	Αχιλλέας
5	2010/08/30	2000	Ηλέκτρα
6	2010/10/04	100	Πάτροκλος

SUM(Τιμή)
5700

Συναθροιστικές συναρτήσεις: AVG() - παράδειγμα

- Η συνάρτηση AVG() επιστρέφει το μέσο όρο (αριθμητικό) από μια στήλη (πεδίο):
- **SELECT** AVG(Τιμή) **FROM** ΠΑΡΑΓΓΕΛΙΕΣ;

κωδπ	Ημερομηνία	Τιμή	Όνομα
1	2010/11/12	1000	Ελένη
2	2010/10/23	1600	Πάρις
3	2010/09/02	700	Μενέλαος
4	2010/09/03	300	Αχιλλέας
5	2010/08/30	2000	Ηλέκτρα
6	2010/10/04	100	Πάτροκλος

SUM(Τιμή)
950

Συναθροιστικές συναρτήσεις: MIN() - παράδειγμα

- Η συνάρτηση MIN() επιστρέφει τη μικρότερη τιμή (αριθμητική) από μια στήλη (πεδίο):
- **SELECT** MIN(Τιμή) **FROM** ΠΑΡΑΓΓΕΛΙΕΣ;

κωδπ	Ημερομηνία	Τιμή	Όνομα
1	2010/11/12	1000	Ελένη
2	2010/10/23	1600	Πάρις
3	2010/09/02	700	Μενέλαος
4	2010/09/03	300	Αχιλλέας
5	2010/08/30	2000	Ηλέκτρα
6	2010/10/04	100	Πάτροκλος

SUM(Τιμή)
100

Συναθροιστικές συναρτήσεις: MAX() - παράδειγμα

- Η συνάρτηση MAX() επιστρέφει τη μεγαλύτερη τιμή (αριθμητική) από μια στήλη (πεδίο):
- **SELECT** MAX(Τιμή) **FROM** ΠΑΡΑΓΓΕΛΙΕΣ;

κωδπ	Ημερομηνία	Τιμή	Όνομα
1	2010/11/12	1000	Ελένη
2	2010/10/23	1600	Πάρις
3	2010/09/02	700	Μενέλαος
4	2010/09/03	300	Αχιλλέας
5	2010/08/30	2000	Ηλέκτρα
6	2010/10/04	100	Πάτροκλος

SUM(Τιμή)
2000

Συναθροιστικές συναρτήσεις: παράδειγμα

- Στο τμήμα SELECT μπορούν να υπάρχουν πολλές συναθροιστικές συναθροίσεις (όσες χρειάζονται).
- Π.χ. Το άθροισμα των τιμών των παραγγελιών όλων των υπαλλήλων καθώς κι η μέγιστη, η ελάχιστη και η μέση τιμή παραγγελίας:

- Συναθροιστικές συναρτήσεις: παράδειγμα
- **SELECT** SUM(Τιμή), MAX(Τιμή), MIN(Τιμή), AVG(Τιμή) **FROM** ΠΑΡΑΓΓΕΛΙΕΣ;

SUM(Τιμή)	MAX(Τιμή)	MIN(Τιμή)	AVG(Τιμή)
5700	2000	100	950

Συναθροιστικές συναρτήσεις ομάδων

- Συναθροιστικές συναρτήσεις **ομάδων**: ομαδοποιείται το αποτέλεσμα του WHERE (καρτεσιανό γινόμενο) *ως προς τις τιμές σε ένα ή περισσότερα πεδία του (κάποιο σύνολο πεδίων)* και υπολογισμός της συναθροιστικής συνάρτησης για κάθε ομάδα.

- Τα πεδία ομαδοποίησης καθορίζονται στο **GROUP BY**

GROUP BY–παράδειγμα 1

- Έστω ότι θέλουμε τη μέση τιμή για κάθε νομό:
- **SELECT** Νομός, AVG(Τιμή)
- **FROM** ΠΡΟΪΟΝΤΑ
- **GROUP BY** Νομός

Όνομα προϊόντος	Τιμή	Ποσότητα	Νομός
ΑΧΛΑΔΙ	100	1100	Αρκαδία
ΦΡΑΟΥΛΑ	200	1100	Ηλεία
ΠΕΠΟΝΙ	800	300	Ηλεία
ΜΗΛΟ	120	1700	Αρκαδία

Νομός	AVG(Τιμή)
Αρκαδία	110
Ηλεία	500

GROUP BY–παράδειγμα 2

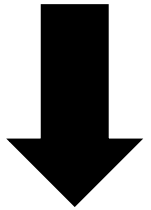
- Έστω ότι θέλουμε την ελάχιστη ηλικία υπαλλήλου που βγάζει τουλάχιστον 3Κ, σε κάθε τμήμα:

- **SELECT** κωδτ, MIN(ηλικία)
- **FROM** ΥΠ
- **WHERE** μισθός >= 3Κ
- **GROUP BY** κωδτ

κωδυ	όνομα	μισθός	κωδτ	ηλικια
44	Ελένη	3000	106	35
82	Πάρις	4600	80	28
7	Μενέλαος	700	106	42
100	Αχιλλέας	4000	80	60
11	Ηλέκτρα	5000	80	50

GROUP BY–παράδειγμα 2

κωδυ	όνομα	μισθός	κωδτ	ηλικια
44	Ελένη	3000	106	35
82	Πάρις	4600	80	28
7	Μενέλαος	700	106	42
100	Αχιλλέας	4000	80	60
11	Ηλέκτρα	5000	80	50



- Έστω ότι θέλουμε την ελάχιστη ηλικία υπαλλήλου που βγάζει τουλάχιστον 3Κ, σε κάθε τμήμα:

- **SELECT** κωδτ, MIN(ηλικία)
- **FROM** ΥΠ
- **WHERE** μισθός >= 3Κ
- **GROUP BY** κωδτ

κωδτ	MIN(ηλικία)
106	35
80	28

GROUP BY–παράδειγμα 2

κωδτ	ηλικία
42	28
42	47
39	60
39	39
12	50
70	35
70	31

Where μισθός >= 3K

κωδτ	ηλικία
42	28
42	47
39	60
39	39
12	50
70	35
70	31

GROUP BY κωδτ

κωδτ	ηλικία
42	28
39	39
12	50
70	31

MIN ηλικια

GROUP BY–παράδειγμα 3(ομαδοποίηση ανά 2 πεδία)

- Αν κάνουμε ομαδοποίησηανά δύο πεδία (χ, ψ), φτιάχνουμε ομάδες όπου κάθε πλειάδα στην ομάδα έχει το ίδιο χ και το ίδιο ψ. Με άλλα λόγια, κάνουμε ομαδοποίηση σε ομάδες που έχουν το **συνδυασμό** των χ, ψ
- Π.χ., αν θέλουμε ομαδοποίηση ανά έτος και μήνα για τη σχέση

έτος	μήνας	ημέρα
2020	Ιανουάριος	10
2020	Ιανουάριος	25
2019	Ιανουάριος	20
2020	Φεβρουάριος	10

έτος	μήνας
2020	Ιανουάριος
2019	Ιανουάριος
2020	Φεβρουάριος

Συναθροιστικές συναρτήσεις ομάδων:HAVING

- Αν θέλουμε να περιορίσουμε τα αποτελέσματα που μας δίνει το GROUP BY, δηλαδή να ανακτήσουμε τιμές **μόνο για ομάδες** που ικανοποιούν ορισμένες συνθήκες, χρησιμοποιούμε το **HAVING**.
- Δηλαδή, **συνθήκες για τις ομάδες** που δημιουργούνται από το GROUP BY εκφράζονται με το τμήμα **HAVING**.
- (Το WHERE δεν μας κάνει διότι εκφράζει συνθήκες για κάθε πλειάδα)

Συναθροιστικές συναρτήσεις ομάδων: HAVING –π.χ.

- Έστω ότι θέλουμε τους νομούς με μέση τιμή προϊόντων μεγαλύτερη ή ίση με 400:

Όνομα προϊόντος	Τιμή	Ποσότητα	Νομός
ΑΧΛΑΔΙ	100	1100	Αρκαδία
ΦΡΑΟΥΛΑ	200	1100	Ηλεία
ΠΕΠΟΝΙ	800	300	Ηλεία
ΜΗΛΟ	120	1700	Αρκαδία

- **SELECT** Νομός, AVG(Τιμή)
- **FROM** ΠΡΟΪΟΝΤΑ
- **GROUP BY** Νομός
- **HAVING** AVG(Τιμή)>=400

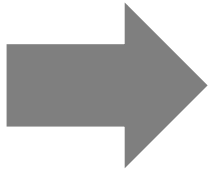
- Φιλτράρει τα αποτελέσματα μετά την ομαδοποίηση

Νομός	AVG(Τιμή)
Ηλεία	500

Συναθροιστικές συναρτήσεις ομάδων: HAVING

- Έστω ότι θέλουμε μέσο μισθό υπαλλήλων κάτω από 30 για κάθε τμήμα με τουλάχιστον 4 **τέτοιους**υπαλλήλους

- **SELECT** κωδτ, AVG(υ.μισθός)
- **FROM** ΥΠ υ
- **WHERE** υ.ηλικία< 30 and
- 4<= ANY(**SELECT**COUNT(*)
- **FROM** ΥΠ υ1
- **WHERE** υ1.κωδτ=υ.κωδτ)
- **GROUP BY** κωδτ



Αυτό μας δίνει το μέσο μισθό όλων των υπαλλήλων των οποίων το τμήμα έχει πάνω από 4 υπαλλήλους

Συναθροιστικές συναρτήσεις ομάδων: HAVING

- Έστω ότι θέλουμε μέσο μισθό υπαλλήλων κάτω από 30 για κάθε τμήμα με τουλάχιστον 4 **τέτοιους**υπαλλήλους

- **SELECT** κωδτ, AVG(μισθός)
- **FROM** ΥΠ
- **WHERE** ηλικία< 30
- **GROUP BY** κωδτ
- **HAVING** COUNT(*)>=4

Συναθροιστικές συναρτήσεις ομάδων: πεδία GROUP BY

- Για ένα ερώτημα με τμήμα GROUP BY, τα τμήματα SELECT και HAVING μπορούν να περιέχουν μόνο τα πεδία του GROUP BY αυτούσια και μέσα σε συναθροιστικές συναρτήσεις οποιοδήποτε πεδίο.

- **SELECT** x, SUM(z), ~~φ~~
- **GROUP BY** x,ψ
- **HAVING** ψ … MAX(ω)

Δεν μπορουμε να εχουμε ασχετα πεδία μεταξύ τους

Συναθροιστικέςσυναρτήσεις ομάδων:πεδία GROUP BY

- ΙΣ (κωδι, όνομα, επίπεδο)
- ΣΚ (κωδσ, όνομα, χρώμα)
- ΚΡ (κωδι, κωδσ, ημερομηνία)

- Έστω για κάθε επίπεδο ιστιοπλόου και χρώμα σκάφους, ο αριθμός αντίστοιχων κρατήσεων:
- **SELECT** επίπεδο, χρώμα, COUNT(*), ΙΣ.όνομα
- **FROM** ΙΣ,ΚΡ, ΣΚ
- **WHERE** ΚΡ.κωδι=ΙΣ.κωδι AND ΚΡ.κωδσ=ΣΚ.κωδσ
- **GROUP BY** επίπεδο, χρώμα

Δεν εχει νοημα γιατι εχω ομάδες και όχι ένα ιστιοπλόο

- Αν υπήρχε HAVING επίπεδο > … AND ημερομηνία > …

Δεν εχει νοημα γιατι το HAVING δουλεύει με ομάδες

Μηδενικές τιμές (NULL values)

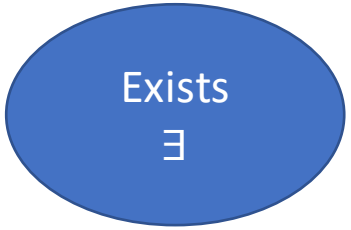
- Η SQL δεν διακρίνει μεταξύ των ερμηνειών του NULL
- Η SQL θεωρεί κάθε NULL τιμή διαφορετική από κάθε άλλη NULL τιμή
- Γι’ αυτό ελέγχει τη NULL με το **IS** / **IS NOT**(κι όχι με = / <>) που δίνει TRUE / FALSE
- Π.χ. το όνομα όλων των υπαλλήλων που δεν έχουν διευθυντή:
- **SELECT** υ.όνομα **FROM** ΥΠ υ
- **WHERE** υ.διευθυντής **IS** NULL

Εμφωλιασμένα ερωτήματα:EXISTS

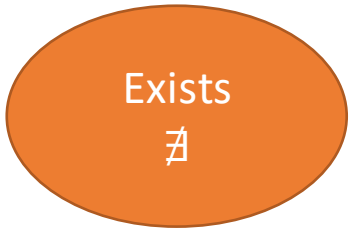
- Μέχρι τώρα συγκρίναμε **τιμή** με **σύνολο**
- π.χ. με το IN, <ALL, κ.ά. συγκρίνουμε μία τιμή με ένα αποτέλεσμα ενός ερωτήματος
- Η SQL δεν επιτρέπει να συγκρίνουμε 2 σύνολα.
- Επιτρέπει όμως να ελέγξουμε αν το εμφωλιασμένοε ρώτημα **δεν έχει πλειάδες**(= το κενό σύνολο) ή **έχει** κάτι.

Εμφωλιασμένα ερωτήματα:EXISTS

- Η SQL επιτρέπει να δούμε αν ένα σύνολο είναι κενό ή όχι -η κενότητα ή όχι του εμφωλιασμένου ερωτήματος ελέγχεται με τον τελεστή: **EXISTS**



EXISTS <εντολή SQL>	
True	Αν το αποτέλεσμα της εντολής δεν είναι κενο
False	Αν το αποτέλεσμα της εντολής είναι κενο



NOT EXISTS <εντολή SQL>	
True	Αν το αποτέλεσμα της εντολής είναι κενο
False	Αν το αποτέλεσμα της εντολής δεν είναι κενο

Παράδειγμα

- ΥΠ (κωδυ, όνομα, φύλο, κωδεξ)
- ΕΞΑΡΤΩΜΕΝΟΣ (ΑΔΤ, όνομα, φύλο)

- Θέλουμε το όνομα κάθε υπαλλήλου που έχει έναν εξαρτώμενο με το ίδιο όνομα και φύλο με αυτόν:

- **SELECT** υ.όνομα
- **FROM**ΥΠ υ, ΕΞΑΡΤΩΜΕΝΟΣ ε
- **WHERE** υ.κωδεξ=ε.ΑΔΤ **AND** υ.όνομα=ε.όνομα **AND**υ.φύλο=ε.φύλο

Παράδειγμα - IN

- ΥΠ (κωδυ, όνομα, φύλο, κωδεξ)
- ΕΞΑΡΤΩΜΕΝΟΣ (ΑΔΤ, όνομα, φύλο)

- Θέλουμε το όνομα κάθε υπαλλήλου που έχει έναν εξαρτώμενο με το ίδιο όνομα και φύλο με αυτόν:

- **SELECT** υ.όνομα
- **FROM** ΥΠ υ
- **WHERE** υ.κωδεξ **IN**

Όπου η τιμή κωδεξ είναι μέσα στο σύνολο των ΑΔΤ που προκύπτει από το εσωτερικό ερώτημα

(**SELECT** ε.ΑΔΤ **FROM** ΕΞΑΡΤΩΜΕΝΟΣ ε **WHERE** υ.όνομα=ε.όνομα **AND** υ.φύλο=ε.φύλο)

Παράδειγμα - EXISTS

- ΥΠ (κωδυ, όνομα, φύλο, κωδεξ)
- ΕΞΑΡΤΩΜΕΝΟΣ (ΑΔΤ, όνομα, φύλο)

- Θέλουμε το όνομα κάθε υπαλλήλου που έχει έναν εξαρτώμενο με το ίδιο όνομα και φύλο με αυτόν:

- **SELECT** υ.όνομα
- **FROM** ΥΠ υ
- **WHERE EXISTS**

(**SELECT** * **FROM** ΕΞΑΡΤΩΜΕΝΟΣ ε **WHERE** υ.κωδεξ=ε.ΑΔΤ **AND** υ.όνομα=ε.όνομα **AND** υ.φύλο=ε.φύλο)

Για κάθε πλειάδα υπαλλήλου, δηλαδή όλους τους υπαλλήλους έναν-έναν, δες αν υπάρχει στο σύνολο που προκύπτει από το εσωτερικό ερώτημα

Παράδειγμα – NOT EXISTS

- ΥΠ (κωδυ, όνομα, φύλο, κωδεξ)
- ΕΞΑΡΤΩΜΕΝΟΣ (ΑΔΤ, όνομα, φύλο)

- Θέλουμε το όνομα των υπαλλήλων που δεν έχουν εξαρτώμενα μέλη:

Παράδειγμα – NOT EXISTS

- ΥΠ (κωδυ, όνομα, φύλο, κωδεξ)
- ΕΞΑΡΤΩΜΕΝΟΣ (ΑΔΤ, όνομα, φύλο)

- Θέλουμε το όνομα των υπαλλήλων που δεν έχουν εξαρτώμενα μέλη:

- **SELECT** υ.όνομα
- **FROM** ΥΠ υ
- **WHERE NOT EXISTS**

(**SELECT** * **FROM** ΕΞΑΡΤΩΜΕΝΟΣ ε **WHERE** υ.κωδεξ=ε.ΑΔΤ)

Για κάθε πλειάδα, δηλαδή για όλους τους υπαλλήλους, δες έναν-έναν. Αν ΔΕΝ υπάρχει στο σύνολο που προκύπτει από το εσωτερικό ερώτημα, δηλείναι κενό, τότε κρατάμε την πλειάδα

Επιστρέφει όλους τους υπαλλήλους που έχουν εξαρτώμενα μέλη

Παράδειγμα 2

- ΥΠ (κωδυ, όνομα, μισθός, κωδτ)
- ΤΜ (κωδτ, όνομα, όροφος, διευθυντής)

- Θέλουμε το όνομα του διευθυντή ΟΛΩΝ (∀) των τμημάτων του 1ου ορόφου:

- **SELECT** υ.όνομα
- **FROM** ΥΠ υ
- **WHERE NOT EXISTS**
  - (**SELECT*** **FROM**ΤΜ τ **WHERE** τ.όροφος=1 **AND** υ.κωδυ<> τ.διευθυντής)

Δηλ. να ∄τμήμα στον 1ο όροφο που να μην έχει τον υ που εξετάζουμε διευθυντή

Παράδειγμα 2

κωδυ	όνομα	μισθός	κωδτ
50	Μαρία	…	12
60	Μήτσος	…	14
70	Νίκος	…	16

κωδτ	όνομα	όροφος	διευθυντής
12	ψάρια	2	70
14	κρέατα	1	50
16	λαχανικά	1	50

SELECT υ.όνομα **FROM** ΥΠ υ
WHERE NOT EXISTS
(**SELECT*** **FROM** ΤΜ τ
WHERE τ.όροφος=1 **AND** υ.κωδυ<> τ.διευθυντής)

Παράδειγμα 2

κωδυ	όνομα	κωδτ	κωδτ	όνομα	όροφος	διευθυντής
50	Μαρία	12	12	ψάρια	2	70
50	Μαρία	12	14	κρέατα	1	50
50	Μαρία	12	16	λαχανικά	1	50
60	Μήτσος	14	12	ψάρια	2	70
60	Μήτσος	14	14	κρέατα	1	50
60	Μήτσος	14	16	λαχανικά	1	50
70	Νίκος	16	12	ψάρια	2	70
70	Νίκος	16	14	κρέατα	1	50
70	Νίκος	16	16	λαχανικά	1	70

(**SELECT*** **FROM**ΤΜ τ
WHERE τ.όροφος=1 **AND**υ.κωδυ != τ.διευθυντής)

Παράδειγμα 2

κωδυ	όνομα	κωδτ	κωδτ	όνομα	όροφος	διευθυντής
50	Μαρία	12	12	ψάρια	2	70
50	Μαρία	12	14	κρέατα	1	50
50	Μαρία	12	16	λαχανικά	1	50
60	Μήτσος	14	12	ψάρια	2	70
60	Μήτσος	14	14	κρέατα	1	50
60	Μήτσος	14	16	λαχανικά	1	50
70	Νίκος	16	12	ψάρια	2	70
70	Νίκος	16	14	κρέατα	1	50
70	Νίκος	16	16	λαχανικά	1	70

(**SELECT** * **FROM**ΤΜ τ
WHERE τ.όροφος=1 **AND** υ.κωδυ!= τ.διευθυντής)

Παράδειγμα 2

κωδυ	όνομα	κωδτ	κωδτ	όνομα	όροφος	διευθυντής
60	Μήτσος	14	16	κρέατα	1	50
60	Μήτσος	14	16	λαχανικά	1	50
70	Νίκος	16	14	κρέατα	1	50

κωδυ	όνομα	μισθός	κωδτ
50	Μαρία	…	12
60	Μήτσος	…	14
70	Νίκος	…	16

SELECT υ.όνομα **FROM** ΥΠ υ
WHERE NOT EXISTS
(**SELECT** * **FROM** ΤΜ τ
WHERE τ.όροφος=1 **AND**υ.κωδυ!= τ.διευθυντής)

Μόνο η 1η πλειάδα δεν υπάρχει μέσα στον πίνακα του αποτελέσματος, δηλ επιστρέφει κενό, άρα το NOT EXISTS είναι true, άρα την κρατάμε και επιστρέφουμε το όνομα.

NOT EXISTS–Παράδειγμα 3

- ΥΠ (κωδυ, όνομα, μισθός, κωδτ)
- ΤΜ (κωδτ, όνομα, όροφος, διευθυντής)

- Ονόματα υπαλλήλων που είναι μόνοι τους στο τμήμα τους:

- **SELECT** υ.όνομα **FROM** ΥΠ υ **WHERE NOT EXISTS**
- (**SELECT** * **FROM** ΥΠ υ2
  - **WHERE** υ2.κωδτ = υ.κωδτ **AND** υ2.κωδυ <> υ.κωδυ)

κωδυ	όνομα	μισθός	κωδτ
50	Μαρία	…	12
60	Μήτσος		14
70	Νίκος		14

κωδτ	όνομα	οροφ.
12	ψάρια	2
14	κρέατα	1

NOT EXISTS–Παράδειγμα 3

- ΥΠ (κωδυ, όνομα, μισθός, κωδτ)
- ΤΜ (κωδτ, όνομα, όροφος, διευθυντής)

- Ονόματα υπαλλήλων που είναι μόνοι του στο τμήμα τους:

- **SELECT** υ.όνομα **FROM** ΥΠ υ **WHERE NOT EXISTS**
- (**SELECT** * **FROM**ΥΠ υ2
  - **WHERE** υ2.κωδτ = υ.κωδτ **AND** υ2.κωδυ <> υ.κωδυ)

κωδυ	όνομα	μισθός	κωδτ	SELECT *…
50	Μαρία	…	12	είναι ∅
60	Μήτσος	14		ΔΕΝ είναι ∅
70	Νίκος	14		ΔΕΝ είναι ∅

κωδτ	όνομα	οροφ.
12	ψάρια	2
14	κρέατα	1

Σημασιολογία SELECT

- **SELECT**[**DISTINCT**] <λίστα αποτελέσματος> 5
- **FROM**<λίστα δηλώσεων μεταβλ. πλειάδων> 1
- [**WHERE**<συνθήκηπλειάδων>] 2
- [**GROUPBY**<πεδία ομαδοποίησης>] 3
- [**HAVING**<συνθήκη ομάδων>] 4
- [**ORDERBY**<λίστα ταξινόμησης>] 5

- Η σημασιολογία είναι η σειρά με την οποία σκεφτόμαστε. Είναι χρήσιμη και σημαντική για να επιβεβαιώνουμε, αν δεν είμαστε σίγουροι!

Σημασιολογία

4. Σε κάθε ομάδα του 3 εφαρμόζουμε τη συνθήκη και κρατάμε αυτές που την ικανοποιούν.
5. Προβολή των τιμών στη λίστα αποτελέσματος σε μια πλειάδα για κάθε ομάδα, ταξινομημένη σύμφωνα με τα πεδία ταξινόμησης και χωρίς διπλότυπα αν απαιτείται.

Σημασιολογία

1. Δημιουργούμε Χ-γινόμενο των πινάκων που διατρέχουν οι δηλούμενες μεταβλητές πλειάδων.
2. Σε κάθε πλειάδα του 1 εφαρμόζουμε τη συνθήκη και κρατάμε αυτές που την ικανοποιούν. Σε περίπτωση *εμφωλιασμένων εντολών* στη συνθήκη, εφαρμόζεται αυτό αναδρομικά προσθέτοντας στο Χ-γινόμενο μια στήλη για κάθε εμφωλιασμένη εντολή, για την αποθήκευση του αντίστοιχου αποτελέσματος
3. Ομαδοποίηση του αποτελέσματος του 2 σε ομάδες με βάση τις τιμές των πεδίων ομαδοποίησης.

Άσκηση 1

- Έστω ότι θέλουμε μέσο μισθό υπαλλήλων κάτω από 30 για τμήματα με > 10 υπαλλήλους

- **SELECT** υ.κωδτ, AVG(υ.μισθός)
- **FROM** ΥΠ υ
- **WHERE** υ.ηλικία< 30
- **GROUP BY** υ.κωδτ
- **HAVING** COUNT(*)>10

Σωστό ή λάθος;

Άσκηση 1

- Έστω ότι θέλουμε μέσο μισθό υπαλλήλων κάτω από 30 για τμήματα με > 10 υπαλλήλους

- **SELECT** υ.κωδτ, AVG(υ.μισθός)
- **FROM** ΥΠ υ
- **WHERE** υ.ηλικία< 30
- **GROUP BY** υ.κωδτ
- **HAVING** COUNT(*)>10

1. Δημιουργούμε το Χ-γινόμενο

Άσκηση 1

- Έστω ότι θέλουμε μέσο μισθό υπαλλήλων κάτω από 30 για τμήματα με > 10 υπαλλήλους

- **SELECT** υ.κωδτ, AVG(υ.μισθός)
- **FROM** ΥΠ υ
- **WHERE** υ.ηλικία< 30
- **GROUP BY** υ.κωδτ
- **HAVING** COUNT(*)>10

1. Δημιουργούμε το Χ-γινόμενο
2. Εφαρμόζουμε τη συνθήκη στο Χ-γινόμενο

Άσκηση 1

- Έστω ότι θέλουμε μέσο μισθό υπαλλήλων κάτω από 30 για τμήματα με > 10 υπαλλήλους

- **SELECT** υ.κωδτ, AVG(υ.μισθός)
- **FROM** ΥΠ υ
- **WHERE** υ.ηλικία< 30
- **GROUP BY** υ.κωδτ
- **HAVING** COUNT(*)>10

1. Δημιουργούμε το Χ-γινόμενο
2. Εφαρμόζουμε τη συνθήκη στο Χ-γινόμενο
3. Ομαδοποιούμε το αποτέλεσμα σε ομάδες με βάση το τι υπάρχει στο GROUP BY

- Άρα τώρα έχουμε μια ομάδα για κάθε τμήμα και μέσα εκεί έχουμε τους υπαλλήλους κάτω από 30.

Άσκηση 1

- Έστω ότι θέλουμε μέσο μισθό υπαλλήλων κάτω από 30 για τμήματα με > 10 υπαλλήλους

- **SELECT** υ.κωδτ, AVG(υ.μισθός)
- **FROM** ΥΠ υ
- **WHERE** υ.ηλικία< 30
- **GROUP BY** υ.κωδτ
- **HAVING** COUNT(*)>10

1. Δημιουργούμε το Χ-γινόμενο
2. Εφαρμόζουμε τη συνθήκη στο Χ-γινόμενο
3. Ομαδοποιούμε το αποτέλεσμα σε ομάδες με βάση το τι υπάρχει στο GROUP BY
4. Εφαρμόζουμε στην ομάδα που έχουμε αυτή τη στιγμή το COUNT.

- «Έχεις, σαν τμήμα, πάνω από 10 ανθρώπους που είναι κάτω από 30;»

Άσκηση 1

- Έστω ότι θέλουμε μέσο μισθό υπαλλήλων κάτω από 30 για τμήματα με > 10 υπαλλήλους

- **SELECT** υ.κωδτ, AVG(υ.μισθός)
- **FROM** ΥΠ υ
- **WHERE** υ.ηλικία< 30
- **GROUP BY** υ.κωδτ
- **HAVING** COUNT(*)>10

Λάθος!

Άρα το παραπάνω απαντά στο μέσος μισθός υπαλλήλων <30 για τμήματα με >10 ***τέτοιους*** υπαλλήλους

Παράδειγμα1 - Σωστό

- Έστω ότι θέλουμε μέσο μισθό υπαλλήλων κάτω από 30 για τμήματα με > 10 υπαλλήλους

- **SELECT** υ.κωδτ, AVG (υ.μισθός)
- **FROM** ΥΠ υ
- **WHERE** υ.ηλικία< 30 **AND** υ.κωδτ **IN**
  - **(SELECT** υ.κωδτ **FROM** ΥΠυ **GROUP BY** υ.κωδτ **HAVING** COUNT(*)>10)

Παράδειγμα 2

- ΙΣ (κωδι, όνομα, επίπεδο)
- ΣΚ (κωδσ, όνομα, χρώμα)
- ΚΡ (κωδι, κωδσ, ημερομηνία)

- Ονόματα ιστιοπλόων στο υψηλότερο επίπεδο:

Λάθος

- **SELECT** ι.όνομα
- **FROM** ΙΣ ι
- **WHERE** ι.επίπεδο= **MAX**(ι.επίπεδο)

Παράδειγμα 2

- ΙΣ (κωδι, όνομα, επίπεδο)
- ΣΚ (κωδσ, όνομα, χρώμα)
- ΚΡ (κωδι, κωδσ, ημερομηνία)

- Ονόματα ιστιοπλόων στο υψηλότερο επίπεδο:

Σωστό

SELECT ι.όνομα
FROM ΙΣ ι
WHERE ι.επίπεδο **IN**
(**SELECT MAX**(ι.επίπεδο) **FROM**ΙΣ ι)

Ευχαριστώ!

- https://eclass.uoa.gr/courses/DIND136/ Έγγραφα > Διαλέξεις
- Βιβλία:
  - Κεφάλαιο 7.1.7:*Συναθρ. Συν.*
  - 7.1.8: *Ομαδοποίηση*

SQL Ενημερώσεις ή Τροποποιήσεις

SQL: βασικές εντολές

- **Select** → για ερωτήματα
- Insert
- Delete
- Update

  → για ενημερώσεις
- Create/Drop table → για λογική σχεδίαση
- Create view → για σχεδίαση εξωτ. σχημάτων
- Create index → για φυσική σχεδίαση

- Κάθε ερώτημα εφαρμόζεται σε ένα σύνολο σχέσεων και παράγει **μια** σχέση

Δημιουργία-Διαγραφή πίνακα

```
CREATE TABLE ΥΠΑΛΛΗΛΟΣ(
ΑΦΜ CHAR(9),
ΌνομαVARCHAR(15) NOTNULL,
Αρχικό_ονόματος_πατρός CHAR,
Επώνυμο VARCHAR(15) NOT NULL,
Ημερ_Γέννησης DATE,
Φύλο CHAR(1),
Μισθός DECIMAL(10,2),
PRIMARY KEY(ΑΦΜ)
);
```

CREATE: τύποι δεδομένων πεδίου ορισμού

Για τον ορισμό του πεδίου ορισμού, οι διαθέσιμοι built-in τύποι περιλαμβάνουν:

- **CHAR**(n) (αλφαριθμητικό σταθερού μήκους)
- **VARCHAR**(n) (αλφαριθμητικό με μήκος το πολύ ηχαρακτήρες)
- **INT** ή **INTEGER** & **SHORTINT**(ακέραιες τιμές & μικρότερες ακ.τ.)
- **DECIMAL** (n, d) (dαπό τα ηψηφία είναι στα δεξιά της υποδιαστολής)
- **FLOAT**(n)ή **REAL** & **DOUBLE PRECISION**
- **DATE** 10 θέσεις YYYY-MM-DD
- **TIME**(αλφαριθμητικό με ειδική μορφή)
- ……..

CREATE: Δημιουργία πίνακα από άλλον με την SQL

CREATE TABLE ΠΡΟΪΣΤΑΜΕΝΟΣ **AS**

SELECT στήλη 1, στήλη 2,...

FROM ΥΠΑΛΛΗΛΟΣ

WHERE... ;

DROP: Διαγραφή πίνακα με την SQL

DROP TABLE ΠΡΟΪΣΤΑΜΕΝΟΣ

Τόσο απλά, ο πίνακας ΠΡΟΪΣΤΑΜΕΝΟΣ δεν θα αποτελεί πια μέρος του σχήματος της ΒΔ και δεν θα μπορούμε να έχουμε πρόσβαση σε καμμιά από τις πλειάδες του…

Εισαγωγή - INSERT INTO

- Εισάγει σε μια σχέση μια ολόκληρη εγγραφή (πλειάδα).
- Σημασιολογία:

- **INSERT INTO** όνομα_σχέσης **VALUES**(λίστα τιμών)

- εισαγωγή τιμών στη λίστα τιμών ως μια νέα πλειάδα στη σχέση

Εισαγωγή - INSERT INTO

- Αντιγράφει”από μια σχέση δεδομένα (πλειάδα) και τα εισάγει σε άλλη.
- Σημασιολογία:

- **INSERT INTO** όνομα_σχέσης **SELECT**… **FROM**…

- εκτέλεση εντολής SELECT και εισαγωγή αποτελέσματος στη σχέση

INSERT INTO - Παράδειγμα

- ΦΟΙΤ (κωδφ, όνομα, μέσος_όρος)
- ΕΓΓΡ (κωδφ, όνομα, πιστοποίηση)

- **INSERT INTO** ΦΟΙΤ
- **VALUES** (33, ‘Albert Einstein’, 10);

- **INSERT INTO** ΦΟΙΤ
- **SELECT** κωδφ, όνομα
- **FROM** ΕΓΓΡ
- **WHERE** πιστοποίηση=“ΝΑΙ”;

INSERT INTO:Παράδειγμα 2

CREATE TABLE ΟΔΗΓΟΣ (

ΑρΔιπλώματος INT NOT NULL,

Επώνυμο CHAR(50),

Όνομα CHAR(50),

ΗμερΠρόσληψηςDATE,

PRIMARY KEY (ΑρΔιπλώματος)

);

- **INSERT INTO** ΟΔΗΓΟΣ **VALUES**(100000, ‘Παππά’, ‘Μαρία’,‘10/12/2016’);
- **INSERT INTO** ΟΔΗΓΟΣ **VALUES**(100001, ‘Λαλάκης’, ‘Λάκης’,‘20/7/2016’);

INSERT INTO: Περιορισμοί

- Ο αριθμός των (τιμών) δεδομένων στο VALUES πρέπει να είναι = με τον αριθμό των στηλών της σχέσης.
- Η σειρά με την οποία δίνονται τα δεδομένα θα πρέπει να αντιστοιχεί με τη σειρά ορισμού των στηλών της σχέσης όταν δημιουργήθηκε (CREATE TABLE).
- Αντίστοιχα και οι τύποι δεδομένων θα πρέπει να συμφωνούν.
- Δεδομένα τύπου CHAR ή VARCHAR περικλείονται μέσα σε ‘ ’.

INSERT INTO: Περιορισμοί

Αν η τιμή μια στήλης δεν είναι γνωστή, μπορεί να της δοθεί η τιμή ***NULL εάν***:

- Η στήλη *δεν* είναι πρωτεύον κλειδί της σχέσης
- Δεν έχει οριστεί με NOT NULL στην εντολή CREATE TABLE
- Αν μια τιμή υπάρχει ήδη, δεν μπορούμε να προσθέσουμε διπλότυπα ως πρωτεύον κλειδί

INSERT INTO ΟΔΗΓΟΣ
VALUES(100044, 'Λαλάκης', 'Κίτσος',NULL);

INSERT INTO ΟΔΗΓΟΣ
VALUES(100044, 'Αστερίου', 'Αστέριος',NULL);

INSERTINTO:με λίστα στηλών

- **INSERT INTO** όνομα_σχέσης(στήλη, στήλη, …) **VALUES**(λίστα τιμών)

- Το ΣΔΒΔ ταιριάζει κάθε τιμή με κάθε στήλη της σχέσης, με τη σειρά με την οποία αναφέρονται.
- Δεν χρειάζεται να αναφέρονται όλες οι στήλες, αρκεί οι στήλες που δεν αναφέρονται να επιτρέπουν τιμές NULL.

INSERTINTO: με λίστα στηλών -παράδειγμα

- **CREATE TABLE** ΦΡΟΥΤΑ(
- κωδπ INT AUTO_INCREMENT PRIMARY KEY,
- όνομα VARCHAR (50) NOT NULL,
- τιμήINT DEFAULT100,
- νομόςVARCHAR (50)
-);

- **INSERT INTO** ΦΡΟΥΤΑ **VALUES**(1, ‘Αχλάδι’, 100, ‘Αρκαδία’);
- **INSERT INTO** ΦΡΟΥΤΑ(κωδπ, όνομα, τιμή, νομός) **VALUES**(2, ‘Κεράσι’, 100, ‘Ημαθία’);
- **INSERT INTO** ΦΡΟΥΤΑ(όνομα, νομός, κωδπ, τιμή) **VALUES**(‘Λεμόνι’, ‘Λακωνία’, 3, 20);
- **INSERT INTO** ΦΡΟΥΤΑ(όνομα, νομός)**VALUES**(‘Μήλο’, ‘Μαγνησία’);

INSERT INTO: με λίστα στηλών -παράδειγμα

κωδπ	όνομα	τιμή	νομός
1	Αχλάδι	100	Αρκαδία
2	Κεράσι	100	Ημαθία
3	Λεμόνι	20	Λακωνία
4	Μήλο	100	Μαγνησία

INSERT:μαζική εισαγωγή

- Κάθε **INSERT INTO** εισάγει μόνο μία εγγραφή (πλειάδα) στη σχέση της ΒΔ.
- Για να εισάγουμε **περισσότερες εγγραφές**:
- **INSERT INTO** όνομα_πίνακα(a,b,c)
- **VALUES**(1,2,3),(4,5,6),(7,8,9);
- ή παίρνουμε τις εγγραφές από άλλον πίνακα:

- **INSERT INTO** όνομα_πίνακα1
- **SELECT** * **FROM** όνομα_πίνακα2

Διαγραφή - DELETE FROM

Διαγραφή - DELETE FROM

- DELETE FROM: Διαγραφή δεδομένων με την SQL
- **DELETE FROM** όνομα_σχέσης [**WHERE**<συνθήκη>]

- Σημασιολογία:
- Εκτέλεση του ερωτήματος SELECT … FROM σχέση WHERE<συνθήκη>
- Απαλοιφή των πλειάδων του αποτελέσματος από τη σχέση.
- Αν δεν υπάρχει το WHERE, τότε διαγράφονται όλες οι πλειάδες της σχέσης

DELETEFROM:Παράδειγμα 1

- **DELETE FROM** ΟΔΗΓΟΣ **WHERE** ΑρΔιπλώματος=100000;
- **DELETE FROM** ΟΔΗΓΟΣ **WHERE** Επώνυμο=‘Λαλάκης’;

ΑρΔιπλώματος	Επώνυμο	Όνομα	ΗμερΠρόσληψης
100000	Παππά	Μαρία	10/12/2016
100001	Λαλάκης	Λάκης	20/7/2016

ΑρΔιπλώματος	Επώνυμο	Όνομα	ΗμερΠρόσληψης

DELETE FROM:Παράδειγμα 2

- ΦΟΙΤ (κωδφ, όνομα, τμήμα, μέσος_όρος)
- ΔΗΛ (κωδφ, κωδμ, βαθμός)
- ΜΑΘ (κωδμ, τίτλος, τμήμα)
- ΤΜ (κωδτ, όνομα)

- Ερώτημα: διαγραφή φοιτητών του τμήματος DIND Ιπου έχουν πάρει <7 σε μάθημα του τμήματος:
- **DELETE FROM** ΦΟΙΤ
- **WHERE** τμήμα='DIND' **AND** βαθμός <7;
- Σωστό ή λάθος;

Λάθος

DELETE FROM:Παράδειγμα 2

- ΦΟΙΤ (κωδφ, όνομα, τμήμα, μέσος_όρος)
- ΔΗΛ (κωδφ, κωδμ, βαθμός)
- ΜΑΘ (κωδμ, τίτλος, τμήμα)
- ΤΜ (κωδτ, όνομα)

- Ερώτημα: διαγραφή φοιτητών του τμήματος DIND που έχουν πάρει <7 σε μάθημα του τμήματος:

- **DELETE FROM** ΦΟΙΤ
- **WHERE** τμήμα='DIND' **AND** κωδφ IN
- (**SELECT** δ.κωδφ **FROM** ΔΗΛ δ, ΜΑΘ μ
- **WHERE** δ.βαθμός<7 AND δ.κωδμ=μ.κωδμ AND μ.τμήμα='DIND')

Ενημερώσεις - UPDATE

- UPDATE: Ενημέρωση δεδομένων με την SQL
- **UPDATE** όνομα_σχέσης **SET** όνομα_στήλης [**WHERE**<συνθήκη>]

- Σημασιολογία:
- Εκτέλεση εντολής SELECT * FROM σχέση WHERE<συνθήκη>
- Εκτέλεση εντολής SELECT όνομα_στήλης, υπόλοιπα πεδία FROM σχέση WHERE<συνθήκη>
- Απαλοιφή των πλειάδων που θα αλλάξουν
- Εισαγωγή των νέων πλειάδων

- Αν δεν υπάρχει το WHERE, τότε τροποποιούνται όλες οι πλειάδες της σχέσης

UPDATE:Παράδειγμα 1

ΑρΔιπλώματος	Επώνυμο	Όνομα	ΗμερΠρόσληψης	Μισθός
100000	Παππά	Μαρία	10-12-2016	1000
100001	Λαλάκης	Λάκης	20-07-2016	2000

- **UPDATE** ΟΔΗΓΟΣ **SET** ΗμερΠρόσληψης=’01-06-2016’;

ΑρΔιπλώματος	Επώνυμο	Όνομα	ΗμερΠρόσληψης	Μισθός
100000	Παππά	Μαρία	01-06-2016	1000
100001	Λαλάκης	Λάκης	01-06-2016	2000

UPDATE:Παράδειγμα 2

ΑρΔιπλώματος	Επώνυμο	Όνομα	ΗμερΠρόσληψης	Μισθός
100000	Παππά	Μαρία	10-12-2016	1000
100001	Λαλάκης	Λάκης	20-07-2016	2000

- **UPDATE** ΟΔΗΓΟΣ **SET** ΗμερΠρόσληψης=’01-06-2016’;
- **UPDATE** ΟΔΗΓΟΣ **SET** Μισθός = Μισθός+(Μισθός*0.5) **WHERE** Μισθός<=1000;

ΑρΔιπλώματος	Επώνυμο	Όνομα	ΗμερΠρόσληψης	Μισθός
100000	Παππά	Μαρία	01-06-2016	1500
100001	Λαλάκης	Λάκης	01-06-2016	2000

INSERT–DELETE –UPDATE: Παράδειγμα

- **INSERT INTO** ΦΡΟΥΤΑ(κωδπ, όνομα, τιμή, νομός) **VALUES**(20, ‘Λεμόνι’, 20, Λακωνία);
- **DELETE FROM** ΦΡΟΥΤΑ**WHERE** όνομα=‘Πεπόνι’;
- **UPDATE** ΦΡΟΥΤΑ **SET** όνομα=‘Φυρίκι’ **WHERE** όνομα=‘Αχλάδι ’;

κωδπ	όνομα	τιμή	νομός	
1	Φυρίκι	100	Αρκαδία	Αλλαξε το Αχλάδι σε Φυρίκι
2	Φράουλα	200	Ηλεία	
~~3~~	~~Πεπόνι~~	~~800~~	~~Ηλεία~~	Αφαιρέθηκετο Πεπόνι
4	Μήλο	120	Αρκαδία	
20	Λεμόνι	20	Λακωνία	Προστέθηκετο Λεμόνι

Ευχαριστώ!

- https://eclass.uoa.gr/courses/DIND136/ Έγγραφα > Διαλέξεις
- Βιβλία:
  - Κεφάλαιο 6.4
    - Εντολές Εισαγωγής, Διαγραφής και Ενημέρωσης