ΕΙΣΑΓΩΓΗ και ΒΑΣΙΚΕΣ ΕΝΤΟΛΕΣ ΔΙΑΧΕΙΡΙΣΗΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

Αντώνιος Δεληγιαννάκης

Τι θα Δούμε …

- Βάση Δεδομένων: Τι είναι και τι όχι…
- Συστήματα Βάσεων Δεδομένων
  - Γιατί είναι χρήσιμα;
  - Βασικά στοιχεία τους
  - Βασικές Έννοιες
- Βασικές Εντολές Διαχείρισης μιας Βάσης Δεδομένων

Βάση Δεδομένων – Τι είναι

- Καλά οργανωμένη συλλογή και αποθήκευση σχετιζόμενων δεδομένων
  - Σχετικά με κοινή εφαρμογή
  - Η σωστή οργάνωση εξαρτάται από το πώς θέλουμε να τα επεξεργαστούμε/επερωτήσουμε

- Παραδείγματα Βάσεων Δεδομένων
  - Τράπεζες, συστήματα μισθοδοσίας, αεροπορικές κρατήσεις, προϊόντα σε ιστοσελίδες (πχ Amazon), νοσοκομεία, πανεπιστήμια…
  - Οι σοβαρές εφαρμογές που παρουσιάζουν/συσχετίζουν πληροφορία έχουν πίσω τους μια βάση.

Παράδειγμα – Τραπεζική Εφαρμογή

Τι δεδομένα χρειαζόμαστε;

- Στοιχεία πελατών, λογαριασμών, συναλλαγές, κάρτες, δάνεια κτλ
- Δεδομένα για αναλήψεις/καταθέσεις σε καταστήματα και ATM

- Σκεφτείτε…
  - Χιλιάδες συναλλαγές/sec
  - Ταυτόχρονη πρόσβαση (ATM, κατάστημα, online) στον ίδιο λογαριασμό

Γιατί όχι Αρχεία;

- Γιατί όχι απλά αρχεία;
  - Πχ, ένα αρχείο ανά μαθητή, λογαριασμό, κτλ
- Προβλήματα:
  - Τι συμβαίνει σε αποτυχία υλικού (δίσκου);
  - Γρήγορη αναζήτηση σε διάφορα πεδία της πληροφορίας
  - Γρήγορη εκτέλεση (διαφορετικών ειδών) ερωτήσεων;
  - Εύκολη γλώσσα για τις ερωτήσεις;
  - Χώρος/αρχείο;
  - Αν αλλάξουμε δομή στο αρχείο θα αλλάξουμε την εφαρμογή;
  - Ταυτόχρονη πρόσβαση…

Σύστημα Διαχείρισης Βάσεων Δεδομένων (ΣΔΒΔ/DBMS)

- Μια μεγάλη εφαρμογή που επιτρέπει την αποθήκευση μεγάλων όγκων πληροφορίας και τη διατήρησή της για μεγάλο χρονικό διάστημα
- Παραδείγματα από ΣΔΒΔ:
  - Oracle, IBM (DB2, Informix),
  - Microsoft (SQL Server, Access)
  - Sybase
  - Open source: MySQL (Sun/Oracle), PostgreSQL
  - Open source library: SQLite
- Θα επικεντρωθούμε σε σχεσιακά ΣΔΒΔ

Σχεσιακό Σύστημα; Τι είναι Αυτό;

- Προς το παρόν σκεφτείτε ότι αποτυπώνονται σε πίνακες οι βασικές οντότητες (λογαριασμοί, πελάτες, συναλλαγές) και οι σχέσεις τους
- Πχ, σε εφαρμογή τηλ/νων

Πίνακας CALLDETAILS

CustId	Call_start	Call_end	PhoneNo	CompanyId
100103	8:00	8:02	6995219694	COSMOPHONE
100105	7:55	8:20	6995811821	VONDOTE
100105	8:40	8:44	6991155123	VONDOTE
105812	8:55	8:59	6991058186	COSMOPHONE
…	…	…	…	…

Σχεσιακό Λογικό Μοντέλο

- Ένας μαθηματικός φορμαλισμός για την περιγραφή των δεδομένων
- Μας επιτρέπει να περιγράφουμε τα δεδομένα που αποθηκεύουμε, καθώς και λειτουργίες σε αυτά, ΧΩΡΙΣ να καθορίζουμε το πώς ακριβώς αποθηκεύονται

Πίνακας CALLDETAILS

CustId	Call_start	Call_end	PhoneNo	CompanyId
100103	8:00	8:02	6995219694	COSMOPHONE
100105	7:55	8:20	6995811821	VONDOTE
100105	8:40	8:44	6991155123	VONDOTE
105812	8:55	8:59	6991058186	COSMOPHONE
…	…	…	…	…

Πίνακας (Table) / Σχέση (Relation)

- Βασική έννοια αποθήκευσης δεδομένων

Πίνακας CALLDETAILS

CustId	Call_start	Call_end	PhoneNo	CompanyId
100103	8:00	8:02	6995219694	COSMOPHONE
100105	7:55	8:20	6995811821	VONDOTE
100105	8:40	8:44	6991155123	VONDOTE
105812	8:55	8:59	6991058186	COSMOPHONE
…	…	…	…	…

Πλειάδες (Tuples) ή Εγγραφές (Records)

- Τα αντικείμενα που αποθηκεύουμε

Πίνακας CALLDETAILS

CustId	Call_start	Call_end	PhoneNo	CompanyId
100103	8:00	8:02	6995219694	COSMOPHONE
100105	7:55	8:20	6995811821	VONDOTE
100105	8:40	8:44	6991155123	VONDOTE
105812	8:55	8:59	6991058186	COSMOPHONE
…	…	…	…	…

tuple → 100103 8:00 8:02 6995219694 COSMOPHONE

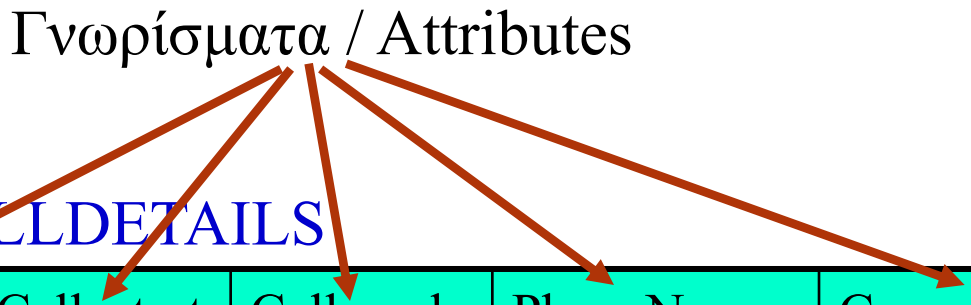
tuple → 100105 7:55 8:20 6995811821 VONDOTE

tuple → 100105 8:40 8:44 6991155123 VONDOTE

tuple → 105812 8:55 8:59 6991058186 COSMOPHONE

Γνωρίσματα (Attributes)

- Attributes: Τα πεδία με τα οποία περιγράφουμε την κάθε πλειάδα
- Το κάθε γνώρισμα έχει κάποιο τύπο δεδομένων
  - π.χ.: INTEGER, CHAR(20), VARCHAR(20), FLOAT, DATETIME



Πίνακας CALLDETAILS

CustId	Call_start	Call_end	PhoneNo	CompanyId
100103	8:00	8:02	6995219694	COSMOPHONE
100105	7:55	8:20	6995811821	VONDOTE
…	…	…	…	…

Σχήμα (Schema) Βάσης

- Σχήμα: Μια περιγραφή των δεδομένων που αποθηκεύονται

Πίνακας CALLDETAILS

CustId	Call_start	Call_end	PhoneNo	CompanyId
100103	8:00	8:02	6995219694	COSMOPHONE
100105	7:55	8:20	6995811821	VONDOTE
100105	8:40	8:44	6991155123	VONDOTE
105812	8:55	8:59	6991058186	COSMOPHONE
…	…	…	…	…

Στιγμιότυπο (Instance) Βάσης

- Στιγμιότυπο μιας βάσης: Τα δεδομένα που αυτή τη στιγμή είναι αποθηκευμένα

Πίνακας CALLDETAILS

CustId	Call_start	Call_end	PhoneNo	CompanyId
100103	8:00	8:02	6995219694	COSMOPHONE
100105	7:55	8:20	6995811821	VONDOTE
100105	8:40	8:44	6991155123	VONDOTE
105812	8:55	8:59	6991058186	COSMOPHONE
…	…	…	…	…

Κλειδί (Key) ενός Πίνακα

- Κλειδί (Key): Ένα ελάχιστο σύνολο γνωρισμάτων που η τιμή τους είναι, ΑΝΕΞΑΡΤΗΤΑ από το ποια δεδομένα είναι αποθηκευμένα, μοναδική στον πίνακα
- Πχ, αν ΠΑΝΤΑ το ΑΜΚΑ ενός φοιτητή είναι μοναδικό, το ΑΜΚΑ είναι ένα κλειδί

Πίνακας myStudent

AM	Name	LastName	AMKA	PHONE
2023000013	Γιώργος	Παύλου	29029900134	…
2023000015	Αθηνά	Παυλή	13059900134	…
2023000016	Γιώργος	Δημητρίου	21029900077	…
2023000019	Δημήτρης	Παύλου	25049900115	…
…	…	…	…	…

Κλειδί (Key) ενός Πίνακα

- Τι εννοούμε «… ελάχιστο σύνολο γνωρισμάτων…» ;
- Ο συνδυασμός ΑΜΚΑ και Name είναι επίσης μοναδικός
  - Όμως το Name **δεν απαιτείται** για να αναγνωρίσεις τον φοιτητή – αρκεί μόνο το ΑΜΚΑ
  - Ο συνδυασμός ΑΜΚΑ,Name **ΔΕΝ** είναι κλειδί

Πίνακας myStudent

AM	Name	LastName	AMKA	PHONE
2023000013	Γιώργος	Παύλου	29029900134	…
2023000015	Αθηνά	Παυλή	13059900134	…
2023000016	Γιώργος	Δημητρίου	21029900077	…
2023000019	Δημήτρης	Παύλου	25049900115	…
…	…	…	…	…

Πρωτεύον Κλειδί (Primary Key)

- Ένας πίνακας μπορεί να έχει πολλά κλειδιά. Πχ:
  - ΑΜ
  - ΑΜΚΑ
  - Name,LastName,PHONE

- Ένα από τα κλειδιά το δηλώνουμε ως το Πρωτεύον Κλειδί (Primary Key) του πίνακα
  - Μας διευκολύνει αν περιέχει λιγότερα γνωρίσματα
  - Μας διευκολύνει αν έχει τιμή που συγκρίνεται εύκολα (πχ, αριθμητική τιμή)

Query Language

- Γλώσσα Επερωτήσεων
- Μας επιτρέπει να διαχειριζόμαστε τα δεδομένα
  - Δημιούργησε έναν πίνακα
  - Διέγραψε έναν πίνακα
  - Κάνε εισαγωγή εγγραφών σε έναν πίνακα
  - … κτλ
- Μας επιτρέπει σύνθετες ανακτήσεις δεδομένων που έχουν αποθηκευτεί

- SQL: Structured Query Language
  - Πρότυπη γλώσσα επερωτήσεων σε Σχεσιακά ΣΔΒΔ

Βασικές Εντολές Διαχείρισης

Λειτουργία Εντολής	Εντολή
Δημιούργησε έναν πίνακα	CREATE TABLE
Διέγραψε έναν πίνακα	DROP TABLE
Κάνε κάποια αλλαγή στην περιγραφή ενός πίνακα	ALTER TABLE
Εισαγωγή εγγραφών σε έναν πίνακα	INSERT INTO
Διαγραφή εγγραφών από πίνακα	DELETE FROM

ΣΗΜΕΙΩΣΗ: Το επόμενο σετ παραδειγμάτων **δεν είναι πλήρες**
Θα επανέλθουμε αργότερα στο μάθημα σε κάποιες εντολές, με επιπλέον λεπτομέρειες
Θα δείτε επίσης παραδείγματα στο Εργαστήριο

CREATE TABLE

- CREATE TABLE tableName (
  columnName1 columnType columnConstraints,
  columnName1 columnType columnConstraints,
  …
)
- CREATE TABLE myStudent (
  AM CHAR(10),
  Name VARCHAR(20),
  LastName VARCHAR(30),
  AMKA INT PRIMARY KEY,
  PHONE CHAR(10)
)

CREATE TABLE

- CREATE TABLE myStudent (
  AM CHAR(10),
  Name VARCHAR(20),
  LastName VARCHAR(30),
  AMKA INT PRIMARY KEY,
  PHONE CHAR(10)
)

Αποθηκεύει 10 χαρακτήρες πάντα

Αποθηκεύει το πολύ 20 χαρακτήρες

Απλό πρωτεύον κλειδί

- Τα πεδία χωρίζονται με το ‘,’
- Δεν έχει σημασία πόσα κενά βάζετε για διαχωριστικά
- Οι λέξεις κλειδιά της SQL δεν είναι case sensitive
- Τα ονόματα των πινάκων και των attributes ίσως είναι case sensitive (εξαρτάται από το ΣΔΒΔ)

CREATE TABLE

- CREATE TABLE myStudent (

```
        AM CHAR(10),
        Name VARCHAR(20),
        LastName VARCHAR(30),
        AMKA      INT,
        PHONE CHAR(10),
        PRIMARY KEY(Name, LastName, PHONE)
)
```

- Αν επιλέξετε σύνθετο Κλειδί (= έχει >1 attributes), τότε ΠΡΕΠΕΙ να δηλωθεί σε ξεχωριστή γραμμή
- Απλά πρωτεύοντα κλειδιά μπορούν να δηλωθούν στην ίδια γραμμή όπου δηλώνεται το γνώρισμα

DROP TABLE

- DROP TABLE tableName;

- DROP TABLE myStudent;

Διαγράφει τη δήλωση του πίνακα και όλα τα δεδομένα του …

ALTER TABLE

- Τροποποιεί τη δήλωση ενός πίνακα

- Μπορείτε πχ
  - να προσθέσετε/διαγράψετε ένα attribute

    ALTER TABLE tableName

    ADD COLUMN columnName columnDataType;

    Πχ:

    ALTER TABLE myStudent

    ADD COLUMN ΤΑΥΤΟΤΗΤΑ CHAR(8);
  - να προσθέσετε/διαγράψετε κάποιον περιορισμό
  - και πολλά άλλα (θα τα δούμε σιγά σιγά)

INSERT INTO

INSERT INTO TABLE_NAME VALUES (A1, …, An)

INSERT INTO myStudent VALUES('2013000013',
'George', 'Smith', 2902720199, '2178899317');

Παρατηρήστε τα μονά εισαγωγικά στα string

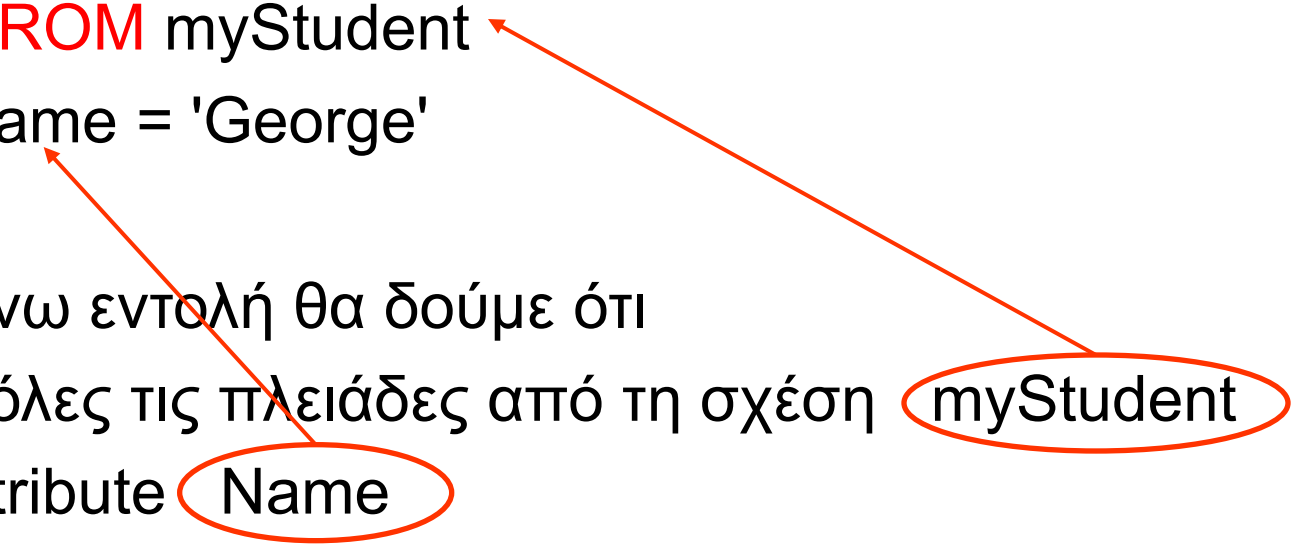
INSERT INTO Sells VALUES('bar5', 'AMSTEL3', 10,
'2016-03-16 15:00:02.5');

ΠΡΟΣΟΧΗ: Οι ημερομηνίες μπορούν να έχουν διαφορετική μορφή ανά σύστημα

DELETE FROM

DELETE FROM TABLE_NAME
WHERE conditions

DELETE FROM myStudent
WHERE Name = 'George'



Η παραπάνω εντολή θα δούμε ότι
διαγράφει όλες τις πλειάδες από τη σχέση myStudent
όπου το attribute Name
έχει τιμή ίση με 'George'

DELETE FROM

```
DELETE FROM myStudent
WHERE Name = 'Γιώργος'
```

Πίνακας myStudent

AM	Name	LastName	AMKA	PHONE
~~2023000013~~	~~Γιώργος~~	~~Παύλου~~	~~29029900134~~	…
2023000015	Αθηνά	Παυλή	13059900134	…
~~2023000016~~	~~Γιώργος~~	~~Δημητρίου~~	~~21029900077~~	…
2023000019	Δημήτρης	Παύλου	25049900115	…
…	…	…	…	…

Για τα Επόμενα Παραδείγματα …

Θα χρησιμοποιούμε ως παράδειγμα έναν πίνακα Student με το ακόλουθο Σχήμα

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

sID: Student ID / Αναγνωριστικό Μαθητή

sName: Όνομα Μαθητή

GPA: Μέσος όρος (βαθμολογίας – μέγιστη τιμή το 4.0)

hs: Πληθυσμός σχολείου στο οποίο φοιτά

SELECT

Εντολή για την ανάκτηση/αναζήτηση δεδομένων.

Αρχικό συντακτικό (θα επεκταθεί σύντομα)

(3) Αν ισχύουν οι συνθήκες του WHERE, εμφάνισε τις τιμές των attributes που εμφανίζονται εδώ

```
SELECT attribute1, attribute2, …
FROM TABLE_NAME
WHERE conditions
```

(1) Θα ελέγξουμε τις πλειάδες αυτού του πίνακα

(2) Για κάθε μία πλειάδα θα ελέγξουμε αν ισχύουν οι συνθήκες του WHERE

SELECT - FROM - WHERE

Εμφάνισε το αναγνωριστικό (sID) και το όνομα (sName) κάθε μαθητή που έχει μέσο όρο πάνω από 3.4

```
SELECT  sID, sName
FROM    Student
WHERE   GPA > 3.4
```

(3) Ποια γνωρίσματα των επιλεγμένων μαθητών θέλω να εμφανίσω;

(1) Σε ποιο πίνακα πρέπει να ψάξω;

(2) Τι συνθήκες πρέπει να ικανοποιεί ο κάθε μαθητής που θα επιλέξω;

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

SELECT – Αποτέλεσμα

```
SELECT  sID, sName
FROM    Student
WHERE   GPA > 3.4
```

sID	sName

Κάθε SQL ερώτημα επιστρέφει
ένα προσωρινό πίνακα (χωρίς όνομα)

Ισχύει ότι GPA > 3.4 ; →

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

SELECT – Αποτέλεσμα

```
SELECT  sID, sName
FROM    Student
WHERE   GPA > 3.4
```

sID	sName
2	Peter

Κάθε SQL ερώτημα επιστρέφει
ένα προσωρινό πίνακα (χωρίς όνομα)

Ισχύει ότι GPA > 3.4 ; →

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

SELECT – Αποτέλεσμα

```
SELECT  sID, sName
FROM    Student
WHERE   GPA > 3.4
```

sID	sName
2	Peter
3	Mary

Κάθε SQL ερώτημα επιστρέφει
ένα προσωρινό πίνακα (χωρίς όνομα)

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

Ισχύει ότι GPA > 3.4 ; 

SELECT - FROM - WHERE

Εμφάνισε το αναγνωριστικό (sID) και το όνομα (sName) κάθε μαθητή που πηγαίνει σε σχολείο με πληθυσμό (hs) 800 μαθητών

```
SELECT  sID, sName
FROM    Student
WHERE   hs = 800
```

Προσοχή: Ο έλεγχος ισότητας γίνεται με = και όχι με το ==
Σημείωση: Ο έλεγχος ανισότητας γίνεται είτε με το != είτε με το <>

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

SELECT – Αποτέλεσμα

```
SELECT  sID, sName
FROM    Student
WHERE   hs = 800
```

sID	sName
1	Jim

Ισχύει ότι hs = 800 ; →

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

SELECT – Αποτέλεσμα

```
SELECT  sID, sName
FROM    Student
WHERE   hs = 800
```

sID	sName
1	Jim
2	Peter

Ισχύει ότι hs = 800 ;

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

SELECT – Αποτέλεσμα

```
SELECT  sID, sName
FROM    Student
WHERE   hs = 800
```

sID	sName
1	Jim
2	Peter

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

Ισχύει ότι hs = 800 ;

SELECT *

Εμφάνισε όλα τα στοιχεία των μαθητών που έχουν μέσο όρο πάνω από 3.4

sID	sName	GPA	hs
2	Peter	3.7	800
3	Mary	3.5	1200

```
SELECT  *
FROM    Student
WHERE   GPA > 3.4
```

Με SELECT * επιστρέφουμε στο αποτέλεσμα όλα τα γνωρίσματα του πίνακα στο FROM

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

Συνθήκες σε String

Συνθήκη	Σημασία
sName = 'Peter'	Το γνώρισμα sName έχει τιμή ίση με 'Peter'
sName != 'Peter'	Το γνώρισμα sName έχει τιμή διαφορετική από 'Peter'
sName <> 'Peter'	Το γνώρισμα sName έχει τιμή διαφορετική από 'Peter'
sName like 'P%'	Το γνώρισμα sName ξεκινάει από 'P'
sName not like '%P%'	Το γνώρισμα sName δεν έχει μέσα του 'P'

Μονά εισαγωγικά για τις τιμές ενός string

Διπλά εισαγωγικά στα ονόματα Πινάκων ή γνωρισμάτων, αν πρέπει να διατηρήσουμε τα κεφαλαία
"Student"
"sName"

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

Συνθήκες σε String

Εμφάνισε το αναγνωριστικό (sID) και το όνομα (sName) κάθε μαθητή που δε λέγεται Mary

```
SELECT  sID, sName
FROM    Student
WHERE   sName != 'Mary'
```

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

SELECT – Αποτέλεσμα

```
SELECT  sID, sName
FROM    Student
WHERE   sName != 'Mary'
```

sID	sName
1	Jim

Ισχύει ότι sName != 'Mary' ;

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

SELECT – Αποτέλεσμα

```
SELECT  sID, sName
FROM    Student
WHERE   sName != 'Mary'
```

sID	sName
1	Jim
2	Peter

Ισχύει ότι sName != 'Mary' ;

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

SELECT – Αποτέλεσμα

```
SELECT  sID, sName
FROM    Student
WHERE   sName != 'Mary'
```

sID	sName
1	Jim
2	Peter

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

Ισχύει ότι sName != 'Mary' ;

Περισσότερες Συνθήκες: AND, OR

Στο WHERE μπορούν να εμφανίζονται πολλές συνθήκες που συνδυάζονται με AND ή OR

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800

```
SELECT   *
FROM     Student
WHERE    sName = 'Jim' OR
         GPA > 3.4 AND hs < 900
```

ισοδύναμο με …

```
SELECT   *
FROM     Student
WHERE    sName = 'Jim' OR
         (GPA > 3.4 AND hs < 900)
```

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

SELECT DISTINCT

Εμφάνισε όλα τα ονόματα των μαθητών

```
SELECT  sName
FROM    Student
```

sName
Jim
Peter
Jim

*Εμφάνισε όλα τα **διακριτά** ονόματα των μαθητών*

```
SELECT  DISTINCT sName
FROM    Student
```

sName
Jim
Peter

Το SELECT DISTINCT αφαιρεί τα διπλότυπα

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Jim	3.5	1200

ORDER BY: Ταξινόμηση

Εμφάνισε όλα τα διακριτά ονόματα των μαθητών ταξινομημένα

```
SELECT    DISTINCT sName
FROM      Student
ORDER BY  sName
```

sName
Jim
Mary
Peter

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

ORDER BY: Ταξινόμηση

Εμφάνισε τα ονόματα και τον πληθυσμό των σχολείων των μαθητών ταξινομημένα κατά φθίνουσα σειρά του hs, και μετά κατά αύξουσα σειρά ονόματος

```
SELECT    sName, hs
FROM      Student
ORDER BY  hs DESC,
          sName ASC
```

Σημείωση: Το ASC είναι προαιρετικό

sName	hs
Mary	1200
Jim	800
Peter	800

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

LIMIT: Όριο Εγγραφών στο Αποτέλεσμα

Εμφάνισε τα ονόματα και τον πληθυσμό των σχολείων των μαθητών ταξινομημένα κατά φθίνουσα σειρά του hs, και μετά κατά αύξουσα σειρά ονόματος.
Εμφάνισε το πολύ 2 εγγραφές

```
SELECT    sName, hs
FROM      Student
ORDER BY  hs DESC,
          sName ASC
LIMIT     2
```

sName	hs
Mary	1200
Jim	800

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

Αποθήκευση Αποτελέσματος

Κάθε SQL ερώτημα επιστρέφει

ένα προσωρινό πίνακα (χωρίς όνομα)

Μπορούμε να δημιουργήσουμε έναν πίνακα για να αποθηκεύσουμε το αποτέλεσμα ενός SQL ερωτήματος

- Τα ονόματα των attributes του πίνακα και ο τύπος των γνωρισμάτων του εξαρτώνται από τα δεδομένα του SQL ερωτήματος
- Ο πίνακας ΔΕ θα έχει κλειδιά ή περιορισμούς
- Ο πίνακας ΔΕΝ πρέπει να υπάρχει ήδη

```
CREATE TABLE myTableName AS
SELECT sID, sName
FROM   Student
```

myTableName

sID	sName
1	Jim
2	Peter
3	Mary

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

Αποθήκευση Αποτελέσματος

Κάθε SQL ερώτημα επιστρέφει

ένα προσωρινό πίνακα (χωρίς όνομα)

Μπορούμε να κάνουμε εισαγωγή των εγγραφών του αποτελέσματος σε έναν πίνακα που ήδη υπάρχει

- Ο πίνακας πρέπει να έχει συμβατό αριθμό και τύπο γνωρισμάτων με το SQL ερώτημα
- Τα ονόματα των γνωρισμάτων στον πίνακα μπορεί να διαφέρουν

```
INSERT INTO myTable
SELECT sID, sName
FROM   Student
```

ΠΡΙΝ την εισαγωγή myTable

id	name
7	Tony
9	Helen

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

ΜΕΤΑ την εισαγωγή myTable

id	name
7	Tony
9	Helen
1	Jim
2	Peter
3	Mary

Ενημερώσεις – UPDATES

```
UPDATE TABLE_NAME
SET A1=Expr1, …, An=Exprn
WHERE conditions
```

```
UPDATE Student SET hs = 1000
WHERE sName = 'Jim'
```

Πρέπει να αλλάξει το hs σε 1000

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	800
3	Mary	3.5	1200

Ενημερώσεις – UPDATES

UPDATE TABLE_NAME
SET A1=Expr1, …, An=Exprn
WHERE conditions

UPDATE Student SET hs = 1000
WHERE sName = 'Jim'

Άλλαξε το hs σε 1000 →

Student

sID	sName	GPA	hs
1	Jim	3.0	1000
2	Peter	3.7	800
3	Mary	3.5	1200

NULL Τιμές

Μπορεί μία τιμή ενός attribute να έχει NULL τιμή

Αυτό σημαίνει ότι η τιμή του είναι άγνωστη/μη καθορισμένη

Τι θα δούμε:

- Γιατί να συμβαίνει/επιτρέπεται αυτό;
- Πώς δηλώνεται ένα attribute το οποίο επιτρέπει NULL τιμές;
- Πώς επηρεάζεται η λογική άλγεβρα αν υπάρχουν NULL;

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	NULL
3	Mary	3.5	1200

NULL Τιμές – Γιατί να Υπάρχουν;

Λόγος 1: Μπορεί όταν κάνουμε εισαγωγή ένα tuple είτε:

- να μη γνωρίζουμε την τιμή κάποιου attribute, ή
- να μην είναι απαραίτητο να έχει τιμή κάποιο attribute (γενικώς)

Π.χ.:

- Μπορεί να μη γνωρίζουμε τον πληθυσμό του σχολείου
- Μπορεί να αποθηκεύουμε στοιχεία για φοιτητές, να υπάρχει attribute "Επιβλέπων Καθηγητής" στον φοιτητή και ο φοιτητής να μην έχει πάρει ακόμα διπλωματική

Student

sID	sName	GPA	hs
1	Jim	3.0	800
2	Peter	3.7	NULL
3	Mary	3.5	1200

Attributes που επιτρέπουν NULL Τιμές

```
CREATE TABLE myStudent (
        AM CHAR(10) UNIQUE NOT NULL,
        Name VARCHAR(20) NOT NULL,
        LastName VARCHAR(30) NOT NULL,
        AMKA   INT  PRIMARY KEY,
        PHONE CHAR(10) NULL,
        advisorID INT
)
```

Όχι NULL

Επιτρέπεται NULL

Κλειδιά που δεν επιλέγονται για PRIMARY KEY δηλώνονται ως UNIQUE

Πού δεν επιτρέπονται NULL;

- Τα attributes του PRIMARY KEY δεν επιτρέπουν NULL
- Attribute δηλωμένα με τον περιορισμό NOT NULL

Πού επιτρέπονται NULL;

- Attribute δηλωμένα με τον περιορισμό NULL
- Attribute που δεν είναι μέλος του PRIMARY KEY και δεν έχουμε δηλώσει NOT NULL ή NULL (default τιμή σε πολλά ΣΔΒΔ, **αλλά όχι σε όλα τα ΣΔΒΔ**)

Σύγκριση με NULL

sID	sName	GPA	hs
1	Jim	3.5	1100

Η σύγκριση ενός attribute που έχει NULL τιμή επιστρέφει UNKNOWN

```
SELECT * FROM STUDENT
WHERE sName <= 'Mac'
```

Ισχύει ότι sName <= 'Mac' ; → TRUE

Student

sID	sName	GPA	hs
1	Jim	3.5	1100
2	Peter	3.7	800
3	NULL	3.6	1200
4	Jim	NULL	NULL
5	Mac	NULL	900

Σύγκριση με NULL

sID	sName	GPA	hs
1	Jim	3.5	1100

Η σύγκριση ενός attribute που έχει NULL τιμή επιστρέφει UNKNOWN

```
SELECT * FROM STUDENT
WHERE sName <= 'Mac'
```

Ισχύει ότι sName <= 'Mac' ; → FALSE

Student

sID	sName	GPA	hs
1	Jim	3.5	1100
2	Peter	3.7	800
3	NULL	3.6	1200
4	Jim	NULL	NULL
5	Mac	NULL	900

Σύγκριση με NULL

Η σύγκριση ενός attribute που έχει NULL τιμή επιστρέφει UNKNOWN

```
SELECT * FROM STUDENT
WHERE sName <= 'Mac'
```

sID	sName	GPA	hs
1	Jim	3.5	1100

Ισχύει ότι sName <= 'Mac' ; → UNKNOWN

Student

sID	sName	GPA	hs
1	Jim	3.5	1100
2	Peter	3.7	800
3	NULL	3.6	1200
4	Jim	NULL	NULL
5	Mac	NULL	900

Σύγκριση με NULL

Η σύγκριση ενός attribute που έχει NULL τιμή επιστρέφει UNKNOWN

sID	sName	GPA	hs
1	Jim	3.5	1100
4	Jim	NULL	NULL

SELECT * FROM STUDENT
WHERE sName <= 'Mac'

Ισχύει ότι sName <= 'Mac' ; → TRUE

Student

sID	sName	GPA	hs
1	Jim	3.5	1100
2	Peter	3.7	800
3	NULL	3.6	1200
4	Jim	NULL	NULL
5	Mac	NULL	900

Σύγκριση με NULL

Η σύγκριση ενός attribute που έχει NULL τιμή επιστρέφει UNKNOWN

sID	sName	GPA	hs
1	Jim	3.5	1100
4	Jim	NULL	NULL
5	Mac	NULL	900

```
SELECT * FROM STUDENT
WHERE sName <= 'Mac'
```

Student

sID	sName	GPA	hs
1	Jim	3.5	1100
2	Peter	3.7	800
3	NULL	3.6	1200
4	Jim	NULL	NULL
5	Mac	NULL	900

Ισχύει ότι sName <= 'Mac' ; → TRUE

Σύγκριση με NULL

Η σύγκριση ενός attribute που έχει NULL τιμή επιστρέφει UNKNOWN

SELECT * FROM STUDENT
WHERE sName > 'Mac'

sID	sName	GPA	hs
2	Peter	3.7	800

Ισχύει ότι sName > 'Mac' ; → TRUE

Student

sID	sName	GPA	hs
1	Jim	3.5	1100
2	Peter	3.7	800
3	NULL	3.6	1200
4	Jim	NULL	NULL
5	Mac	NULL	900

Σύγκριση με NULL – Παράδοξο;;;

```
SELECT * FROM STUDENT
WHERE sName > 'Mac' or
      sName <= 'Mac'
```

sID	sName	GPA	hs
1	Jim	3.5	1100
2	Peter	3.7	800
4	Jim	NULL	NULL
5	Mac	NULL	900

Θα περιμένατε το OR να επιστρέψει όλες τις πλειάδες!

Όμως για τον τρίτο μαθητή και οι 2 συγκρίσεις επιστρέφουν UNKNOWN

Ισχύει ότι sName > 'Mac' or sName <= 'Mac' ; → UNKNOWN

Student

sID	sName	GPA	hs
1	Jim	3.5	1100
2	Peter	3.7	800
3	NULL	3.6	1200
4	Jim	NULL	NULL
5	Mac	NULL	900

Σωστός Έλεγχος για NULL Τιμές

SELECT * FROM STUDENT

WHERE sName is NOT NULL

sID	sName	GPA	hs
1	Jim	3.5	1100
2	Peter	3.7	800
4	Jim	NULL	NULL
5	Mac	NULL	900

Student

sID	sName	GPA	hs
1	Jim	3.5	1100
2	Peter	3.7	800
3	NULL	3.6	1200
4	Jim	NULL	NULL
5	Mac	NULL	900

Σωστός Έλεγχος για NULL Τιμές

SELECT * FROM STUDENT
WHERE sName is NULL

sID	sName	GPA	hs
3	NULL	3.6	1200

Σημείωση: ΔΕΝ είναι σωστό να κάνουμε σύγκριση με '='

SELECT * FROM STUDENT
WHERE sName = NULL
δεν ικανοποιείται

Student

sID	sName	GPA	hs
1	Jim	3.5	1100
2	Peter	3.7	800
3	NULL	3.6	1200
4	Jim	NULL	NULL
5	Mac	NULL	900

NULL και Λογική Άλγεβρα 3-Τιμών

Η σύγκριση ενός attribute που έχει NULL τιμή επιστρέφει UNKNOWN

TRUE or FALSE → TRUE

TRUE and FALSE → FALSE

UNKNOWN or UNKNOWN → UNKNOWN

UNKNOWN and UNKNOWN → UNKNOWN

UNKNOWN or FALSE → UNKNOWN

UNKNOWN and FALSE → FALSE

UNKNOWN or TRUE → TRUE

UNKNOWN and TRUE → UNKNOWN

NOT UNKNOWN → UNKNOWN

NULL και Λογική Άλγεβρα 3-Τιμών

Πράξεις με NULL έχουν ως αποτέλεσμα NULL

A+NULL → NULL

A-NULL → NULL

A*NULL → NULL

A/NULL → NULL

NULL/A → NULL

Το Α παραπάνω μπορεί να είναι πραγματικός αριθμός ή NULL