Κεφάλαιο 5 Η γλώσσα SQL

5.1 Εισαγωγή

Η γλώσσα **SQL (Structured Query Language) ε**ίναι η πιο διαδεδομένη διαλογική γλώσσα ερωταπαντήσεων που χρησιμοποιείται για την επικοινωνία του χρήστη με σχεσιακές ΒΔ. Πρόκειται για μία μη-διαδικαστική γλώσσα τέταρτης γενιάς, στην οποία ο χρήστης διατυπώνει διάφορα αιτήματα και το ΣΔΒΔ αναλαμβάνει να τα ικανοποιήσει. Η SQL δίνει τη δυνατότητα στο χρήστη να δημιουργήσει, να τροποποιήσει και να ενημερώσει τους πίνακες της βάσης, καθώς και να αναζητήσει πληροφορίες από τη βάση εφαρμόζοντας σύνθετα κριτήρια αναζήτησης.

Η γνώση της SQL είναι απαραίτητη ακόμη κι αν χρησιμοποιούμε ένα ΣΔΒΔ με γραφικό περιβάλλον (όπως η Access), καθώς πολλά συστήματα ΒΔ της μορφής πελάτη-εξυπηρετητή (client-server) έχουν SQL server.

Η SQL αποτελείται από δύο υποσύνολα, τη DDL και τη DML.

DDL (Data Definition Language): Γλώσσα ορισμού δεδομένων

Αποτελείται από τις εντολές με τις οποίες καθορίζουμε τη λογική οργάνωση των δεδομένων της βάσης, δηλ. δημιουργούμε τους πίνακες και τις μεταξύ τους σχέσεις.

DML (Data Manipulation Language): Γλώσσα χειρισμού δεδομένων

Αποτελείται από τις εντολές με τις οποίες ενημερώνουμε τα δεδομένα της βάσης και δημιουργούμε ερωτήματα για ανάκληση πληροφοριών από τη βάση.

Πρότυπα της SQL

Τα περισσότερα σχεσιακά ΣΔΒΔ (συμπεριλαμβανομένης και της Microsoft Access 2000) χρησιμοποιούν το πρότυπο ANSI / ISO SQL-92, υπάρχει όμως και το νέο πρότυπο ANSI / ISO SQL-99 που χρησιμοποιείται κυρίως από αντικειμενοστρεφή ΣΔΒΔ.

5.2 Access και SQL

Για να πληκτρολογήσουμε μία εντολή SQL μέσα από το περιβάλλον της Access 2000, δημιουργούμε ένα *νέο ερώτημα* για τη ΒΔ που μας ενδιαφέρει (βλ. ενότητα 8.2) και αφού μεταβούμε στην Προβολή σχεδίασης, στη συνέχεια επιλέγουμε την **Προβολή SQL** από το μενού ή το εικονίδιο της Προβολής.

Αφού πληκτρολογήσουμε την εντολή μας μπορούμε να την εκτελέσουμε κανονικά, όπως και ένα ερώτημα σε γραφικό περιβάλλον, από το εικονίδιο της **Εκτέλεσης** ή μέσω του μενού **Ερώτημα.**

Επίσης, αν δημιουργήσουμε ένα ερώτημα χρησιμοποιώντας το γραφικό περιβάλλον της Access, μπορούμε επιλέγοντας την **Προβολή SQL**, να δούμε τον κώδικα SQL που δημιουργεί η Access για να εκτελέσει το ερώτημά μας.

Οι τύποι δεδομένων της SQL που υποστηρίζει η Access 2000 είναι οι ακόλουθοι:

CHAR(n)	Αλφαριθμητικό με **n** χαρακτήρες (n = 0 έως 255)
TEXT	Κείμενο μέχρι 2.14GB
BYTE	Ακέραιος από 0 έως 255
SMALLINT	Ακέραιος μέχρι $0(10^5)$
INTEGER	Ακέραιος μέχρι $0(10^9)$
REAL	Πραγματικός μέχρι $0(10^{38})$
FLOAT	Πραγματικός διπλής ακρίβειας μέχρι $0(10^{308})$
DECIMAL(n, m)	Δεκαδικός συνολικά με **n** ψηφία, **m** από αυτά, δεκαδικά
MONEY	Μεγάλος δεκαδικός $0(10^{15})$ με 4 δεκαδικά ψηφία
COUNTER	Αυτόματη αρίθμηση
DATETIME	Ημερομηνία ή ώρα
BIT	Yes ή No
IMAGE	Αρχείο οποιασδήποτε μορφής μέχρι 2.14GB

Στη συνέχεια παρουσιάζουμε μόνο τις εντολές SQL που μπορούν να εκτελεστούν μέσα από το περιβάλλον της Access 2000. Σε όλες τις ενότητες που ακολουθούν, οι δεσμευμένες λέξεις της SQL γράφονται με κεφαλαία γράμματα, ενώ τα ονόματα των πεδίων, των πινάκων κλπ γράφονται με πεζά, χωρίς όμως αυτός ο κανόνας να είναι υποχρεωτικός κατά τη σύνταξη εντολών SQL.

Προσοχή: Οι αγκύλες δηλώνουν τα προαιρετικά τμήματα των εντολών και δεν αποτελούν μέρος των εντολών.

5.3 Εντολές για τη διαχείριση της δομής της ΒΔ

Με τις εντολές που ακολουθούν διαχειριζόμαστε τη δομή της ΒΔ, δηλαδή δημιουργούμε, τροποποιούμε και διαγράφουμε τους πίνακες, τα πεδία και τις μεταξύ τους συσχετίσεις. Είναι προτιμότερο, τέτοιου είδους εντολές να χρησιμοποιούνται μόνο κατά τον αρχικό σχεδιασμό, όταν η βάση είναι κενή δεδομένων. Αν είναι απαραίτητο να γίνουν αλλαγές στη δομή κατά τη διάρκεια της λειτουργίας της βάσης, αυτές πρέπει να γίνουν με ιδιαίτερη προσοχή.

α) Δημιουργία πίνακα

Για να δημιουργήσουμε τη δομή ενός καινούργιου πίνακα, χρησιμοποιούμε την εντολή:

CREATE TABLE *Πίνακας*$_1$ **(***Πεδίο*$_1$ *Τύπος*$_1$*, Πεδίο*$_2$ *Τύπος*$_2$*, ..., Πεδίο*$_N$ *Τύπος*$_N$*,*

[CONSTRAINT *Περιορισμός*$_1$ **PRIMARY KEY (***Πεδίο*$_i$ *, Πεδίο*$_j$ *,...***)],**
[CONSTRAINT *Περιορισμός*$_2$ **NOT NULL (***Πεδίο*$_i$ *, Πεδίο*$_j$ *,...***)],**
[CONSTRAINT *Περιορισμός*$_3$ **UNIQUE (***Πεδίο*$_i$ *, Πεδίο*$_j$ *,...***)],**
[CONSTRAINT *Περιορισμός*$_4$ **FOREIGN KEY (***Πεδίο*$_i$**) REFERENCES** *Πίνακας*$_2$ **(***Πεδίο*$_k$**)]**
);

- Στις παραμέτρους *Πίνακας, Πεδίο, Περιορισμός* δίνουμε κάποιο όνομα. Τα ονόματα μπορούν να αποτελούνται και από ελληνικούς χαρακτήρες, καλύτερα κεφαλαίους για να αποφεύγονται οι τόνοι. Δεν πρέπει να περιέχουν κενά.

- Στην παράμετρο *Τύπος* δηλώνουμε έναν τύπο από αυτούς που περιγράψαμε στην ενότητα 5.2.

- Οι περιορισμοί (**CONSTRAINTS**) είναι προαιρετικοί.

- Ο περιορισμός **PRIMARY KEY** δηλώνει το πρωτεύον κλειδί, ο **NOT NULL** δηλώνει τα πεδία που δεν επιτρέπεται να μένουν κενά, ενώ ο **UNIQUE** δηλώνει τα πεδία που δεν επιτρέπεται να έχουν δύο ίδιες τιμές.

- Ο περιορισμός **FOREIGN KEY** δηλώνει ότι το *Πεδίο*$_i$ είναι ξένο κλειδί και συγκεκριμένα είναι το πρωτεύον κλειδί *Πεδίο*$_k$ του πίνακα *Πίνακας*$_2$, όπως δηλώνει η δεσμευμένη λέξη **REFERENCES.** Με τον τρόπο αυτό καθορίζουμε τη συσχέτιση που υπάρχει ανάμεσα στους πίνακες *Πίνακας*$_1$ και *Πίνακας*$_2$.

Παράδειγμα

Έστω ότι θέλουμε να δημιουργήσουμε μία ΒΔ για τα μουσικά CD της προσωπικής μας συλλογής και τα τραγούδια που περιέχει το καθένα. Υποθέτουμε ότι ένα τραγούδι χαρακτηρίζεται μοναδικά από το συνδυασμό του τίτλου του, των ερμηνευτών του και του CD στο οποίο ανήκει (καθώς μπορεί το ίδιο τραγούδι να εμφανίζεται σε περισσότερα από ένα CD). Μ' αυτή την υπόθεση, χρειάζονται μόνο δύο πίνακες για να υλοποιήσουμε τη ΒΔ:

CD

ΚΩΔΙΚΟΣ	**ΤΙΤΛΟΣ**	**ΕΤΑΙΡΕΙΑ**	**ΕΤΟΣ**

↓ Πρωτεύον κλειδί

ΤΡΑΓΟΥΔΙΑ

ΤΙΤΛΟΣ	**ΕΡΜΗΝΕΥΤΕΣ**	**ΚΩΔΙΚΟΣ_CD**

↓ Πρωτεύον κλειδί ↓ Πρωτεύον κλειδί ↓ Πρωτεύον κλειδί

Για τη δημιουργία των παραπάνω πινάκων, θα χρησιμοποιήσουμε τις εντολές SQL:

CREATE TABLE *CD* **(**
ΚΩΔΙΚΟΣ **CHAR(10) ,** *ΤΙΤΛΟΣ* **CHAR(50),** *ΕΤΑΙΡΕΙΑ* **CHAR(50),** *ΕΤΟΣ* **INTEGER,**
CONSTRAINT *ΚΛΕΙΔΙ* **PRIMARY KEY (***ΚΩΔΙΚΟΣ***));**

CREATE TABLE *ΤΡΑΓΟΥΔΙΑ* **(**
ΤΙΤΛΟΣ **CHAR(50),** *ΕΡΜΗΝΕΥΤΕΣ* **CHAR(60),** *ΚΩΔΙΚΟΣ_CD* **CHAR(10),**
CONSTRAINT *ΣΥΝΘΕΤΟ_ΚΛΕΙΔΙ* **PRIMARY KEY (***ΤΙΤΛΟΣ, ΕΡΜΗΝΕΥΤΕΣ***,** *ΚΩΔΙΚΟΣ_CD***),**
CONSTRAINT *ΞΕΝΟ_ΚΛΕΙΔΙ* **FOREIGN KEY (***ΚΩΔΙΚΟΣ_CD***) REFERENCES** *CD* **(***ΚΩΔΙΚΟΣ***));**

β) Τροποποίηση της δομής πίνακα

Με τις παρακάτω εντολές η SQL μας δίνει τη δυνατότητα προσθήκης ενός πεδίου στα δεξιά του πίνακα, αφαίρεσης κάποιου πεδίου, αλλαγής του τύπου του πεδίου, προσθήκης ή αφαίρεσης περιορισμού αντίστοιχα.

ALTER TABLE $Πίνακας_1$
ADD COLUMN $Πεδίο_1$ $Τύπος_1$**;**

ALTER TABLE $Πίνακας_1$
DROP COLUMN $Πεδίο_1$**;**

ALTER TABLE $Πίνακας_1$
ALTER COLUMN $Πεδίο_1$ $Τύπος_2$**;**

ALTER TABLE $Πίνακας_1$
ADD CONSTRAINT $Περιορισμός_1$ …. (όπως και η εντολή δημιουργίας περιορισμού)

ALTER TABLE $Πίνακας_1$
DROP CONSTRAINT $Περιορισμός_1$**;**

Παράδειγμα

Έστω ότι θέλουμε να προσθέσουμε στον πίνακα ΤΡΑΓΟΥΔΙΑ μία στήλη που περιγράφει τη χρονική διάρκεια του κάθε τραγουδιού σε λεπτά:

ALTER TABLE *ΤΡΑΓΟΥΔΙΑ* **ADD COLUMN** *ΔΙΑΡΚΕΙΑ* **INTEGER;**

Παρατήρηση: Όταν πρόκειται για αφαίρεση ή τροποποίηση κάποιου στοιχείου της δομής του πίνακα χρειάζεται ιδιαίτερη προσοχή, γιατί συνήθως χάνονται κάποια δεδομένα. Καλό είναι τέτοιου είδους αλλαγές να γίνονται μόνο όταν ο πίνακας είναι κενός δεδομένων.

γ) Διαγραφή πίνακα

Η εντολή αυτή θέλει επίσης ιδιαίτερη προσοχή γιατί διαγράφει ολόκληρο τον πίνακα με τα δεδομένα του.

DROP TABLE $Πίνακας_1$ **;**

5.4 Εντολές για τη διαχείριση των δεδομένων της ΒΔ

Με τις επόμενες εντολές διαχειριζόμαστε τα δεδομένα της βάσης, εφόσον έχει ήδη καθοριστεί η δομή της.

α) Εισαγωγή δεδομένων

INSERT INTO *Πίνακας*$_1$ **(** *Πεδίο*$_i$, *Πεδίο*$_j$, ... **) VALUES (** *Τιμή*$_i$, *Τιμή*$_j$, ... **);**

- Μία εντολή INSERT μπορεί να προσθέσει μία μόνο γραμμή στο τέλος του πίνακα.
- Η INSERT μας επιτρέπει να βάλουμε επιλεκτικά τιμές μόνο σε ορισμένα από τα πεδία. της τελευταίας γραμμής. Τα υπόλοιπα μπορούν να μείνουν κενά.
- Για πεδία χαρακτήρων οι τιμές δίνονται μέσα σε απλά εισαγωγικά (π.χ. 'Κώστας') ενώ για πεδία ημερομηνίας δίνονται μέσα σε # (π.χ. #25/3/2000#).

Παράδειγμα

INSERT INTO *CD* **(***ΚΩΔΙΚΟΣ, ΤΙΤΛΟΣ***) VALUES (***'4567 6788 5', 'Harvest'***);**

Παρατήρηση: Η εντολή INSERT σπάνια χρησιμοποιείται στην πράξη, καθώς δεν επιτρέπει τη μαζική εισαγωγή δεδομένων. Η εισαγωγή δεδομένων γίνεται πολύ πιο εύκολα από ένα γραφικό περιβάλλον (π.χ. της Access) και μέσω των φορμών εισαγωγής δεδομένων (βλ. ενότητες 10.2, 10.3).

β) Τροποποίηση δεδομένων

UPDATE *Πίνακας*$_1$ **SET** *Πεδίο*$_i$ =*Τιμή*$_i$, *Πεδίο*$_j$ =*Τιμή*$_j$, ... **WHERE** *Συνθήκη***;**

- Ως *Τιμή* μπορεί να μπει και μία αριθμητική παράσταση (π.χ. *Πεδίο*$_i$ = *Πεδίο*$_k$ *+100*)
- Η *Συνθήκη* μπορεί να περιέχει τελεστές σύγκρισης (>, <, =, κλπ) και λογικούς τελεστές (**NOT, AND, OR**). Επίσης μπορεί να περιέχει τους τελεστές **IS**, **IN, BETWEEN … AND** και **LIKE** (βλ. ενότητα 5.5).
- Μ' αυτό τον τρόπο μπορούμε να μεταβάλλουμε επιλεκτικά τα δεδομένα ενός πίνακα.

Παράδειγμα

UPDATE *CD* **SET** *ΕΤΑΙΡΕΙΑ='Pendragon', ΕΤΟΣ=1999* **WHERE** *ΤΙΤΛΟΣ='Harvest'***;**

γ) Διαγραφή γραμμών

DELETE FROM *Πίνακας*$_1$ [**WHERE** *Συνθήκη*]**;**

- Αν δεν υπάρχει η επιλογή WHERE διαγράφονται όλες οι γραμμές του πίνακα.

- Δεν μπορούμε να διαγράφουμε επιλεκτικά τιμές από ορισμένα πεδία του πίνακα

Παράδειγμα

DELETE FROM *CD* **WHERE** *ΤΙΤΛΟΣ* **IS NULL;**

5.5 Ερωτήματα επιλογής δεδομένων

Με την εντολή **SELECT** που ακολουθεί, δε μεταβάλλουμε τα δεδομένα της βάσης, αλλά επιλέγουμε να εμφανίσουμε κάποια από αυτά με βάση κάποια κριτήρια. Κατά την επιλογή μπορούν να γίνουν πράξεις πάνω στα δεδομένα, ή να γίνει ομαδοποίηση των δεδομένων και να εξαχθούν κάποια συγκεντρωτικά στοιχεία. Όπως θα διαπιστώσουμε στη συνέχεια από τα παραδείγματα αυτής και της επόμενης ενότητας, η SELECT μπορεί να είναι πολύ χρήσιμη για τη λήψη αποφάσεων με χρήση των δεδομένων της βάσης.

Η γενική της σύνταξη είναι η ακόλουθη:

SELECT [**DISTINCT**] [**TOP** *n*]
***** ή *Πίνακας*$_1$*.Πεδίο*$_i$ [**AS** *Όνομα*$_i$]*, Πίνακας*$_1$*.Πεδίο*$_j$ [**AS** *Όνομα*$_j$]*, ...,*
Πίνακας$_2$*.Πεδίο*$_k$ [**AS** *Όνομα*$_k$]*, ...,* ή *Παράσταση*$_i$ *με κάποια πεδία* [**AS** *Όνομα*$_i$]
FROM *Πίνακας*$_1$ *, Πίνακας*$_2$*, ...*
[**WHERE** *Συνθήκη*$_1$]
[**GROUP BY** *Πίνακας*$_1$*.Πεδίο*$_j$*, Πίνακας*$_2$*.Πεδίο*$_k$*,...*]
[**HAVING** *Συνθήκη*$_2$]
[**ORDER BY** *Πίνακας*$_1$*.Πεδίο*$_i$ [**DESC**]*, Πίνακας*$_2$*.Πεδίο*$_k$ [**DESC**]*, ...*]**;**

α) Κύριο τμήμα της SELECT

Είναι το απαραίτητο κομμάτι της SELECT που περιλαμβάνει τις δεσμευμένες λέξεις **SELECT** και **FROM**.

- Η επιλογή **DISTINCT** δηλώνει ότι δε θέλουμε στο αποτέλεσμα της SELECT να εμφανιστούν δύο ίδιες γραμμές.

- Η επιλογή **TOP** *n* δηλώνει ότι θέλουμε το αποτέλεσμα της SELECT να περιέχει μόνο τις πρώτες *n* γραμμές που προκύπτουν από την αναζήτηση που κάναμε (συνήθως συνδυάζεται με κάποιου είδους ταξινόμηση, δηλ. με την εντολή **ORDER BY** - βλ. παράδ. 4**)**.

- Το αποτέλεσμα της SELECT θα έχει τόσες στήλες όσα και τα πεδία, εκφράσεις κλπ που δηλώνουμε στο κύριο τμήμα της SELECT.

- Ο χαρακτήρας ***** αντί των διαφόρων πεδίων, δηλώνει ότι στο αποτέλεσμα της SELECT θα εμφανιστούν *όλα* τα πεδία των πινάκων που δηλώνουμε στη **FROM.**

- Με τη δεσμευμένη λέξη **FROM** δηλώνουμε τους πίνακες που θα χρησιμοποιηθούν για τη λήψη των δεδομένων. Όταν δηλώνουμε περισσότερους από έναν πίνακες, είναι απαραίτητο, μπροστά από τα ονόματα των πεδίων που χρησιμοποιούμε σε οποιοδήποτε τμήμα της SELECT, να γράφουμε και το όνομα του πίνακα στον οποίο ανήκει το κάθε πεδίο, ακολουθούμενο από τελεία π.χ. $\textit{Πίνακας}_1.\textit{Πεδίο}_j$, $\textit{Πίνακας}_2.\textit{Πεδίο}_k$ (βλ. παράδ. 6, 7).

- Αν όμως χρησιμοποιούμε στοιχεία από ένα μόνο πίνακα, μπορούμε να αναφερόμαστε στα πεδία του μόνο με το όνομά τους.

- Η επιλογή **AS** $\textit{Όνομα}_i$ χρησιμοποιείται μόνο αν θέλουμε στο αποτέλεσμα της SELECT, το συγκεκριμένο πεδίο ή παράσταση να εμφανιστεί με άλλο όνομα από αυτό που έχει στον πίνακα. Το $\textit{Όνομα}_i$ ονομάζεται **ψευδώνυμο (alias)** του πεδίου ή της παράστασης. Η **AS** είναι πολύ χρήσιμη όταν πρόκειται για μία παράσταση πεδίων, γιατί σ' αυτήν την περίπτωση το όνομα που δίνει η SQL στη στήλη που προκύπτει είναι συνήθως δυσνόητο. Με την επιλογή **AS** μπορούμε να δώσουμε στην παράσταση ό,τι όνομα επιθυμούμε (βλ. παραδείγματα 8-12).

- Μία παράσταση πεδίων μπορεί να περιέχει, εκτός από αριθμητικές πράξεις (αν τα πεδία είναι αριθμητικά), συναρτήσεις που αφορούν μία ή περισσότερες τιμές ενός πεδίου. Παραδείγματα τέτοιων συναρτήσεων είναι οι **ABS** και **YEAR**, που εφαρμόζονται σε κάθε μία από τις τιμές ενός αριθμητικού πεδίου ή πεδίου ημερομηνίας/ώρας αντίστοιχα, ενώ υπάρχουν και οι συναρτήσεις ομαδοποίησης δεδομένων, για τις οποίες θα μιλήσουμε παρακάτω.

<u>Παραδείγματα</u>

Έστω η βάση για τα μουσικά CD που δημιουργήσαμε και τροποποιήσαμε στην ενότητα 5.3.

CD

ΚΩΔΙΚΟΣ	**ΤΙΤΛΟΣ**	**ΕΤΑΙΡΕΙΑ**	**ΕΤΟΣ**

↓

Πρωτεύον κλειδί

ΤΡΑΓΟΥΔΙΑ

ΤΙΤΛΟΣ	**ΕΡΜΗΝΕΥΤΕΣ**	**ΚΩΔΙΚΟΣ_CD**	**ΔΙΑΡΚΕΙΑ**

↓ ↓ ↓

Π ρ ω τ ε ύ ο ν κ λ ε ι δ ί

1) Στο ερώτημα: "Ποιες εταιρείες είναι καταχωρημένες στη βάση;" η απάντηση δίνεται εκτελώντας την εντολή:

SELECT DISTINCT *ΕΤΑΙΡΕΙΑ* **FROM** *CD***;**

2) Να εμφανιστούν τα στοιχεία των τραγουδιών της βάσης:

SELECT * FROM *ΤΡΑΓΟΥΔΙΑ***;**

β) Ταξινόμηση

Μπορούμε να ταξινομήσουμε τα αποτελέσματα μιας SELECT ως προς ένα ή περισσότερα πεδία, χρησιμοποιώντας τη δεσμευμένη λέξη **ORDER BY**.

- Η **ORDER BY** ταξινομεί τα αποτελέσματα της αναζήτησης σε αύξουσα σειρά, με βάση τη σειρά των πεδίων που δηλώνονται σ' αυτήν. Π.χ. η **ORDER BY** ΕΠΩΝΥΜΟ, ΟΝΟΜΑ ταξινομεί πρώτα κατά αύξουσα σειρά επωνύμου, και μόνο αν βρεθούν δύο ή περισσότερες γραμμές με το ίδιο ΕΠΩΝΥΜΟ, αυτές ταξινομούνται σε αύξουσα σειρά ονόματος. Αν θέλουμε φθίνουσα σειρά στην ταξινόμηση με βάση ένα πεδίο, τότε δεξιά του ονόματός του προσθέτουμε τη δεσμευμένη λέξη **DESC (**είναι το αντίστοιχο του **ASC (**αύξουσα σειρά), που συνήθως παραλείπεται).

Παραδείγματα

3) Να εμφανιστούν οι τίτλοι των CD, η εταιρεία και η χρονολογία, ταξινομημένα από το πιο πρόσφατο CD ως το πιο παλιό και με αλφαβητική σειρά πρώτα εταιρείας και κατόπιν τίτλου.

SELECT *ΤΙΤΛΟΣ, ΕΤΑΙΡΕΙΑ, ΕΤΟΣ* **FROM** *CD*
ORDER BY *ΕΤΟΣ* **DESC,** *ΕΤΑΙΡΕΙΑ, ΤΙΤΛΟΣ***;**

4) Να βρεθεί το τραγούδι με τη μικρότερη διάρκεια

SELECT TOP 1 *ΤΙΤΛΟΣ, ΔΙΑΡΚΕΙΑ* **FROM** *ΤΡΑΓΟΥΔΙΑ* **ORDER BY** *ΔΙΑΡΚΕΙΑ***;**

γ) Συνθήκες επιλογής γραμμών

Ενώ στο κύριο τμήμα της SELECT επιλέγουμε τις στήλες των πινάκων που μας ενδιαφέρουν, χρησιμοποιώντας την επιλογή **WHERE** ή **HAVING** επιλέγουμε ορισμένες μόνο γραμμές από τις παραπάνω στήλες. Οι γραμμές αυτές θα πρέπει να πληρούν τα κριτήρια (*Συνθήκες*) που θέτουμε στη WHERE ή τη HAVING. Η **HAVING** χρησιμοποιείται μόνο σε συνδυασμό με συναρτήσεις ομαδοποίησης και την επιλογή **GROUP BY**, γι' αυτό θα αναφερθούμε σ' αυτήν παρακάτω.

- Η *Συνθήκη*$_1$ της **WHERE** είναι λογική συνθήκη που μπορεί να περιέχει οποιαδήποτε από τα πεδία των πινάκων που εμφανίζονται στη **FROM**, τελεστές πράξεων (+,-,*,/), τελεστές σύγκρισης (>, <, =, κλπ) αλλά και λογικούς τελεστές (NOT, AND, OR).

- Όταν η σύγκριση γίνεται με την κενή τιμή (**NULL**), δε χρησιμοποιείται ο τελεστής **"="** αλλά ο τελεστής **IS**.

- Μία συνθήκη μπορεί επίσης να περιέχει τους τελεστές **IN, BETWEEN … AND** και **LIKE**, που χρησιμοποιούνται ως εξής:

Πεδίο$_i$ **IN** (*Τιμή*$_1$*,Τιμή*$_2$*,…,Τιμή*$_N$ *)*
Πεδίο$_i$ **BETWEEN** *Τιμή*$_1$ **AND** *Τιμή*$_2$
Πεδίο$_i$ **LIKE** *Αλφαριθμητικό*

όπου το *Αλφαριθμητικό* περιέχει τους χαρακτήρες *"wildcards" (μπαλαντέρ)* ***** ή **?**. Ο χαρακτήρας ***** αντικαθιστά οποιοδήποτε αριθμό χαρακτήρων (ή και 0 χαρακτήρες), ενώ ο χαρακτήρας **?** αντικαθιστά ακριβώς 1 χαρακτήρα.

π.χ. *ΕΠΩΝΥΜΟ* **LIKE** *'Ι*'* σημαίνει επώνυμο που αρχίζει από Ι, ενώ
ΕΠΩΝΥΜΟ **LIKE** *'*ΟΠΟΥΛΟ*'* σημαίνει επώνυμο που περιέχει τους χαρακτήρες 'ΟΠΟΥΛΟ'. Τέλος,
ΚΩΔΙΚΟΣ **LIKE** *'???'* σημαίνει ότι ο κωδικός αποτελείται από ακριβώς τρεις χαρακτήρες.

- Αν δεν υπάρχει η **WHERE** τότε εμφανίζονται όλες οι τιμές των στηλών που έχουμε δηλώσει στο κύριο τμήμα της SELECT. Χρειάζεται όμως προσοχή στην περίπτωση που στη **FROM** έχουμε δηλώσει περισσότερους του ενός πίνακες: συνήθως οι πίνακες αυτοί έχουν κάποιο κοινό πεδίο που δηλώνει τη μεταξύ τους συσχέτιση. Η ταύτιση των δύο πεδίων πρέπει να δηλωθεί στη WHERE, αλλιώς το αποτέλεσμα θα είναι το καρτεσιανό γινόμενο των αντίστοιχων τμημάτων των δύο πινάκων! (βλ. παράδ. 6).

Παραδείγματα

5) Να εμφανιστούν τα τραγούδια που έχουν διάρκεια μεταξύ 3' και 4'

SELECT *ΤΙΤΛΟΣ, ΔΙΑΡΚΕΙΑ* **FROM** *ΤΡΑΓΟΥΔΙΑ*
WHERE *ΔΙΑΡΚΕΙΑ* **BETWEEN** *3* **AND** *4* **ORDER BY** *ΔΙΑΡΚΕΙΑ*;

6) Να εμφανιστούν οι τίτλοι όλων των CD στα οποία υπάρχουν τραγούδια του ερμηνευτή "Sting"

SELECT *CD.ΤΙΤΛΟΣ, ΤΡΑΓΟΥΔΙΑ.ΕΡΜΗΝΕΥΤΕΣ* **FROM** *CD, ΤΡΑΓΟΥΔΙΑ*
WHERE *CD. ΚΩΔΙΚΟΣ=ΤΡΑΓΟΥΔΙΑ.ΚΩΔΙΚΟΣ_CD*
AND *ΤΡΑΓΟΥΔΙΑ.ΕΡΜΗΝΕΥΤΕΣ* **LIKE** *'* Sting *'*;

(με την επιλογή **LIKE** *'* Sting *'* εντοπίζουμε και όλα τα τραγούδια που ερμηνεύει ο "Sting" μαζί με άλλους ερμηνευτές).

Παρατήρηση: Αν στην παραπάνω συνθήκη της WHERE είχε παραλειφθεί η συνθήκη *CD.ΚΩΔΙΚΟΣ=ΤΡΑΓΟΥΔΙΑ.ΚΩΔΙΚΟΣ_CD* η οποία δηλώνει τη συσχέτιση των πινάκων *CD* και *ΤΡΑΓΟΥΔΙΑ*, θα παίρναμε ως αποτέλεσμα το καρτεσιανό γινόμενο (δηλαδή όλους τους δυνατούς συνδυασμούς) των τιμών του πεδίου *CD.ΤΙΤΛΟΣ* επί εκείνες τις τιμές του πεδίου *ΤΡΑΓΟΥΔΙΑ.ΕΡΜΗΝΕΥΤΕΣ*, που περιέχουν τη συμβολοσειρά *'Sting'*. Έτσι θα εμφανίζονταν όλοι οι τίτλοι των CD που υπάρχουν στη βάση, και όχι μόνο αυτοί που σχετίζονται με το συγκεκριμένο ερμηνευτή.

1) Να εμφανιστούν τα τραγούδια που περιέχει το CD με τίτλο "Harvest"

SELECT *ΤΡΑΓΟΥΔΙΑ.ΤΙΤΛΟΣ* **FROM** *CD, ΤΡΑΓΟΥΔΙΑ* **WHERE**
CD.ΚΩΔΙΚΟΣ=ΤΡΑΓΟΥΔΙΑ.ΚΩΔΙΚΟΣ_CD **AND** *CD.ΤΙΤΛΟΣ='Harvest'*;

δ) Ομαδοποίηση γραμμών

Συχνά είναι απαραίτητο να εξάγουμε κάποια συγκεντρωτικά στοιχεία από τους πίνακες της βάσης μας, όπως αθροίσματα, μέσους όρους, μέγιστες ή ελάχιστες τιμές, πλήθος δεδομένων κλπ. Για το σκοπό αυτό η SQL διαθέτει τις **συναρτήσεις ομαδοποίησης SUM, AVG, MIN, MAX** και **COUNT**. Οι συναρτήσεις αυτές δεν εφαρμόζονται μεμονωμένα σε μία τιμή, αλλά σε ομάδες τιμών ενός πεδίου του πίνακα. Για κάθε ομάδα τιμών, η συνάρτηση ομαδοποίησης επιστρέφει μία μόνο τιμή (το άθροισμα των τιμών της ομάδας, το μέσο όρο των τιμών, την ελάχιστη τιμή, τη μέγιστη τιμή, ή το πλήθος των τιμών της ομάδας αντίστοιχα).

- Από τις παραπάνω συναρτήσεις διαφοροποιείται λίγο η συνάρτηση **COUNT**, καθώς δεν αναφέρεται σε συγκεκριμένο πεδίο αλλά στην ουσία μετράει το πλήθος των *γραμμών* της κάθε ομάδας. Γι' αυτό συνήθως χρησιμοποιείται χωρίς να περιέχει για όρισμα ένα συγκεκριμένο πεδίο, αλλά ως **COUNT(*)**.

Ο καθορισμός των ομάδων των γραμμών πάνω στις οποίες θα εφαρμοστεί μία συνάρτηση ομαδοποίησης γίνεται με την επιλογή **GROUP BY**.

- Αν δεν υπάρχει η **GROUP BY,** τότε ολόκληρη η στήλη στην οποία εφαρμόζεται η συνάρτηση ομαδοποίησης λαμβάνεται σα μία ομάδα. Έτσι π.χ. στη βάση με τα μουσικά CD, αν ζητάμε το άθροισμα της χρονικής διάρκειας όλων των τραγουδιών που υπάρχουν στη βάση, αρκεί να χρησιμοποιήσουμε τη συνάρτηση **SUM** στο πεδίο ΔΙΑΡΚΕΙΑ του πίνακα ΤΡΑΓΟΥΔΙΑ. Αν όμως θέλουμε το άθροισμα της διάρκειας των τραγουδιών κάθε CD χωριστά, θα πρέπει να χωρίσουμε τις γραμμές του πίνακα ΤΡΑΓΟΥΔΙΑ σε ομάδες ανάλογα με τον κωδικό του CD στον οποίο ανήκει το κάθε τραγούδι και να εφαρμόσουμε τη συνάρτηση SUM σε κάθε μία από αυτές τις ομάδες. Αυτό επιτυγχάνεται βάζοντας το πεδίο ΚΩΔΙΚΟΣ_CD μέσα στη **GROUP BY** (βλ. παράδ. 8-9).

- Μπορούμε να δηλώσουμε περισσότερα από ένα πεδία μέσα στη **GROUP BY**. Αυτό σημαίνει ότι μία ομάδα γραμμών, αποτελείται από εκείνες τις γραμμές του πίνακα που έχουν τον ίδιο συνδυασμό τιμών σε όλα τα πεδία που δηλώνονται στη **GROUP BY**. Έτσι, όσο πιο πολλά πεδία δηλώνουμε στη **GROUP BY**, τόσο περισσότερες (και μικρότερες) ομάδες δημιουργούμε.

- Αν χρησιμοποιήσουμε τη **GROUP BY** χωρίς να έχουμε συνάρτηση ομαδοποίησης, τότε απλά ταξινομούμε σε αύξουσα σειρά τον πίνακα, με βάση τα πεδία που είναι δηλωμένα στη **GROUP BY** (δηλ. η **GROUP BY** αντικαθιστά την **ORDER BY**).

- Η εντολή **HAVING** χρησιμοποιείται μόνο εφόσον έχει προηγηθεί η **GROUP BY** και επιλέγει κάποιες από τις ομάδες γραμμών που έχουν δημιουργηθεί με τη GROUP BY, ανάλογα με τη *Συνθήκη*$_2$. Στη *Συνθήκη*$_2$ περιέχονται συναρτήσεις ομαδοποίησης. Ο λόγος που η *Συνθήκη*$_2$ δεν μπορεί να ενσωματωθεί σε μία **WHERE**, είναι γιατί η **WHERE** εκτελείται πριν από τη **GROUP BY**. Έτσι δεν είναι γνωστό στη **WHERE** αν υπάρχουν ομάδες γραμμών και ποιες είναι. Για παράδειγμα, αν θέλαμε να εμφανίσουμε μόνο τα CD που έχουν διάρκεια μεγαλύτερη από 1 ώρα, τη συνθήκη αυτή θα έπρεπε να τη δηλώσουμε στη **HAVING** και όχι στη **WHERE**, καθώς περιέχει τη συνάρτηση **SUM** (βλ. παραδ. 11-12).

Παραδείγματα

8) Να βρεθεί ο μέσος όρος διάρκειας όλων των τραγουδιών της βάσης

SELECT INT(AVG (*ΔΙΑΡΚΕΙΑ***)) AS** *ΜΕΣΗ_ΔΙΑΡΚΕΙΑ* **FROM** *ΤΡΑΓΟΥΔΙΑ***;**

(Η συνάρτηση **INT** επιστρέφει το ακέραιο μέρος του αριθμού)

9) Να βρεθεί ο μέσος όρος διάρκειας των τραγουδιών κάθε CD

SELECT *ΚΩΔΙΚΟΣ_CD*, **INT(AVG (***ΔΙΑΡΚΕΙΑ***)) AS** *ΜΕΣΗ_ΔΙΑΡΚΕΙΑ_ΤΡΑΓΟΥΔΙΩΝ*
FROM *ΤΡΑΓΟΥΔΙΑ* **GROUP BY** *ΚΩΔΙΚΟΣ_CD***;**

10) Να βρεθεί το CD με το μεγαλύτερο αριθμό τραγουδιών

SELECT TOP 1 *ΚΩΔΙΚΟΣ_CD***, COUNT(***ΚΩΔΙΚΟΣ_CD***) AS** *ΑΡΙΘΜΟΣ_ΤΡΑΓΟΥΔΙΩΝ*
FROM *ΤΡΑΓΟΥΔΙΑ* **GROUP BY** *ΚΩΔΙΚΟΣ_CD*
ORDER BY COUNT(*ΚΩΔΙΚΟΣ_CD***) DESC;**

11) Να βρεθούν τα CD με διάρκεια μεγαλύτερη από 50'

SELECT *ΚΩΔΙΚΟΣ_CD***, SUM(***ΔΙΑΡΚΕΙΑ***) AS** *ΔΙΑΡΚΕΙΑ_CD*
FROM *ΤΡΑΓΟΥΔΙΑ* **GROUP BY** *ΚΩΔΙΚΟΣ_CD* **HAVING SUM(***ΔΙΑΡΚΕΙΑ)>50***;**

12) Να βρεθούν οι ερμηνευτές που είναι καταχωρημένοι σε περισσότερα από δύο τραγούδια της βάσης

SELECT *ΕΡΜΗΝΕΥΤΕΣ***, COUNT(*) AS** *ΑΡΙΘΜΟΣ_ΤΡΑΓΟΥΔΙΩΝ*
FROM *ΤΡΑΓΟΥΔΙΑ* **GROUP BY** *ΕΡΜΗΝΕΥΤΕΣ* **HAVING COUNT(*)>***2***;**

ε) Σειρά εκτέλεσης των ενεργειών επιλογής

Συνολικά η σειρά με την οποία γίνονται ενέργειες επιλογής στα δεδομένα με βάση τα διάφορα τμήματα της εντολής SELECT είναι η ακόλουθη:

FROM	(επιλέγουμε τους σχετικούς πίνακες)
WHERE	(επιλέγουμε ορισμένες γραμμές)
GROUP BY	(ομαδοποιούμε αυτές τις γραμμές)
HAVING	(επιλέγουμε κάποιες από τις παραπάνω ομάδες γραμμών)
SELECT	(επιλέγουμε μόνο τις στήλες που μας ενδιαφέρουν)
ORDER BY	(ταξινομούμε τις παραπάνω στήλες)
DISTINCT	(αφαιρούμε τα διπλότυπα)
TOP *n*	(από τις γραμμές που προκύπτουν, κρατάμε τις πρώτες *n*)

5.6 Σύνθετα ερωτήματα επιλογής

Συχνά οι ανάγκες ενός ερωτήματος είναι τέτοιες ώστε χρειάζεται να χρησιμοποιηθούν περισσότερες από μία εντολές SELECT, φωλιασμένες η μία μέσα στην άλλη. Η δεύτερη SELECT ενσωματώνεται μέσα στη συνθήκη του τμήματος WHERE (ή HAVING) της πρώτης SELECT. Η δεύτερη SELECT μπαίνει σε παρένθεση και ακολουθεί έναν από τους τελεστές >, <, =, … , **IN**. Π.χ:

SELECT *Πεδίο*$_1$*, Πεδίο*$_2$*,...***FROM** *Πίνακας*$_1$
WHERE *Πεδίο*$_i$ **IN (SELECT** *Πεδίο*$_j$ **FROM** *Πίνακας*$_2$ **WHERE** *Συνθήκη***)**

Απαραίτητη προϋπόθεση είναι τα πεδία *Πεδίο*$_i$ και *Πεδίο*$_j$ να είναι του ίδιου τύπου.

- Σημειώνουμε ότι οι τελεστές σύγκρισης (<, >, = κλπ) μπορούν να χρησιμοποιηθούν μόνο αν η δεύτερη **SELECT** επιστρέφει μία μοναδική τιμή (π.χ. αν είναι **SELECT MAX(***Πεδίο*$_j$**) FROM**… ή **SELECT TOP** 1 *Πεδίο*$_j$ **FROM**…). Συνήθως όμως, περιμένουμε να πάρουμε από τη δεύτερη **SELECT** ένα πλήθος τιμών. Σ' αυτές τις περιπτώσεις χρησιμοποιούνται οι τελεστές σύγκρισης σε συνδυασμό με τους τελεστές **ALL** και **SOME** (ή **ANY**). Π.χ:

SELECT *Πεδίο*$_1$*, Πεδίο*$_2$*,...***FROM** *Πίνακας*$_1$
WHERE *Πεδίο*$_i$ **> ALL (SELECT** *Πεδίο*$_j$ **FROM** *Πίνακας*$_2$ **WHERE** *Συνθήκη***)**

δηλαδή το *Πεδίο*$_i$ είναι μεγαλύτερο από *όλα* τα αποτελέσματα της δεύτερης **SELECT**. Αν αντί για **ALL** γράφαμε **SOME** (ή **ANY**), τότε για να αληθεύει η συνθήκη της πρώτης **WHERE**, θα έπρεπε το *Πεδίο*$_i$ να είναι μεγαλύτερο από *τουλάχιστον ένα* από τα αποτελέσματα της δεύτερης **SELECT**.

Παραδείγματα

1) Να βρεθούν τα τραγούδια που έχουν διάρκεια μεγαλύτερη από το μέσο όρο διάρκειας όλων των τραγουδιών της βάσης

SELECT *ΤΙΤΛΟΣ, ΔΙΑΡΚΕΙΑ* **FROM** *ΤΡΑΓΟΥΔΙΑ*
WHERE *ΔΙΑΡΚΕΙΑ* **> (SELECT AVG(***ΔΙΑΡΚΕΙΑ***) FROM** *ΤΡΑΓΟΥΔΙΑ***);**

2) Ποιοι ερμηνευτές εμφανίζονται στα CD της εταιρείας "Pendragon";

SELECT DISTINCT *ΕΡΜΗΝΕΥΤΕΣ* **FROM** *ΤΡΑΓΟΥΔΙΑ* **WHERE** *ΚΩΔΙΚΟΣ_CD* **IN (SELECT** *ΚΩΔΙΚΟΣ* **FROM** *CD* **WHERE** *ΕΤΑΙΡΕΙΑ='Pendragon'***);**

3) Υπάρχουν στη βάση CD που δεν έχουν κανένα τραγούδι καταχωρημένο;

SELECT *ΚΩΔΙΚΟΣ* **FROM** *CD* **WHERE** *ΚΩΔΙΚΟΣ***<> ALL**
(SELECT *ΚΩΔΙΚΟΣ_CD* **FROM** *ΤΡΑΓΟΥΔΙΑ***);**

Τέλος, αναφέρουμε ότι δύο SELECT μπορούν να χρησιμοποιηθούν ταυτόχρονα και χωρίς να είναι φωλιασμένες, αλλά με την πράξη της ένωσης, υπό την προϋπόθεση ότι τα αντίστοιχα πεδία των δύο SELECT είναι του ίδιου τύπου:

(SELECT *Πεδίο$_i$*, *Πεδίο$_j$*, … **FROM** *Πίνακας$_1$* **WHERE** *Συνθήκη$_1$***)**
UNION [ALL]
(SELECT *Πεδίο$_i$*, *Πεδίο$_j$*, … **FROM** *Πίνακας$_2$* **WHERE** *Συνθήκη$_2$***)**

- Τα αποτελέσματα της δεύτερης SELECT μπαίνουν μετά από αυτά της πρώτης SELECT, γι' αυτό πρέπει τα αντίστοιχα πεδία των δύο SELECT να είναι συμβατά.

- Η επιλογή **ALL** δηλώνει ότι συμπεριλαμβάνονται και διπλότυπες γραμμές που μπορεί να προκύψουν από τις δύο **SELECT**. Χωρίς την **ALL**, τα διπλότυπα αφαιρούνται.

Παράδειγμα

Από τους πίνακες ΠΕΛΑΤΕΣ_Α και ΠΕΛΑΤΕΣ_Β της ενότητας 3.4, αν θέλουμε να εμφανίσουμε όλους τους πελάτες της Θεσσαλονίκης, χωρίς διπλότυπα:

(SELECT *ΕΠΩΝΥΜΟ*, *ΟΝΟΜΑ* **FROM** *ΠΕΛΑΤΕΣ_Α* **WHERE** *ΠΟΛΗ='Θεσ/κη'***)**
UNION
(SELECT *ΕΠΩΝΥΜΟ*, *ΟΝΟΜΑ* **FROM** *ΠΕΛΑΤΕΣ_Β* **WHERE** *ΠΟΛΗ='Θεσ/κη'***)**

Παρατήρηση: Αν *Πίνακας$_1$=Πίνακας$_2$*, τότε η ένωση των δύο **SELECT** μπορεί να αντικατασταθεί από μία **SELECT** ως εξής:
SELECT *Πεδίο$_i$*, *Πεδίο$_j$*, … **FROM** *Πίνακας$_1$* **WHERE (***Συνθήκη$_1$* **OR** *Συνθήκη$_2$***);**

5.7 Ασκήσεις

- Για τη βάση με τα μουσικά CD που περιγράφεται στην ενότητα 5.5 να δημιουργήσετε ερωτήματα SQL που να εκφράζουν τα παρακάτω:

1) Να εμφανιστούν μόνο οι τίτλοι των CD και τα αντίστοιχα τραγούδια που έχουν στον τίτλο τους (είτε των CD είτε των τραγουδιών) τη λέξη 'Blues'

2) Να βρεθούν οι τίτλοι όλων των CD στα οποία εμφανίζεται το συγκρότημα 'Dire Straits' (να γίνει με φωλιασμένες SELECT)

- Έστω οι δύο πίνακες πελατών ΠΕΛΑΤΕΣ_Α και ΠΕΛΑΤΕΣ_Β που περιγράφονται στην ενότητα 3.4.

3) Να υλοποιηθεί το παράδειγμα για τη διαφορά σχέσεων (δηλαδή να βρεθούν οι πελάτες οι αποκλειστικοί της εταιρείας Α) με φωλιασμένες SELECT.

4) Να υλοποιηθεί το παράδειγμα για την τομή σχέσεων (δηλαδή να βρεθούν οι κοινοί πελάτες των δύο εταιρειών) με φωλιασμένες SELECT.

5) Με χρήση της εντολής SELECT, να υλοποιηθεί το παράδειγμα της ενότητας 3.4 για το γινόμενο σχέσεων.

6) Με χρήση της εντολής SELECT, να υλοποιηθεί το παράδειγμα της ενότητας 3.4 για την προβολή σχέσεων.

7) Με χρήση της εντολής SELECT, να υλοποιηθεί το παράδειγμα της ενότητας 3.4 για την επιλογή από σχέση.

8) Με χρήση της εντολής SELECT, να υλοποιηθεί το παράδειγμα της ενότητας 3.4 για τη συνένωση σχέσεων.

- Έστω η βάση ΚΑΘΗΓΗΤΗΣ-ΜΑΘΗΜΑ-ΔΙΔΑΣΚΕΙ που περιγράφεται στην ενότητα 3.3.

9) Να γραφούν οι εντολές SQL για τη δημιουργία των τριών πινάκων.

10) Να γραφούν οι εντολές SQL για τη δημιουργία των τριών πινάκων και των μεταξύ τους συσχετίσεων.

11) Με την κατάλληλη εντολή SQL να προστεθεί στον πίνακα ΜΑΘΗΜΑ το πεδίο ΔΙΔΑΚΤΙΚΕΣ_ΜΟΝΑΔΕΣ

Να δημιουργηθούν τα παρακάτω ερωτήματα σε γλώσσα SQL:

12) Να εμφανιστούν όλοι οι διαφορετικοί Τομείς στους οποίους ανήκουν οι καθηγητές

13) Ποιοι καθηγητές δίδασκαν στο Τμήμα Μαθηματικών κατά το έτος 1998-1999;

14) Να βρεθεί το πλήθος όλων των μαθημάτων του Τμήματος.

15) Να βρεθεί το πλήθος όλων των μαθημάτων του κάθε Τομέα.

16) Να εμφανιστούν οι Τομείς που έχουν πλήθος μαθημάτων > 20.