Α Τ Ε Ι Μ Ε Σ Ο Λ Ο Γ Γ Ι Ο Υ

ΣΧΟΛΗ ΔΙΟΙΚΗΣΗΣ ΟΙΚΟΝΟΜΙΑΣ
ΤΜΗΜΑ ΕΦΑΡΜΟΓΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΣΤΗΝ ΔΙΟΙΚΗΣΗ ΚΑΙ ΟΙΚΟΝΟΜΙΑ

Π Τ Υ Χ Ι Α Κ Η Ε Ρ Γ Α Σ Ι Α

«ΔΙΑΧΕΙΡΙΣΗ ΠΡΟΓΡΑΜΜΑΤΟΣ ΕΠΔΟ – ΕΚΤΑΚΤΟΙ ΕΚΠΑΙΔΕΥΤΙΚΟΙ»

ΓΚΙΟΥΡΣΑΝΗ ΣΟΦΙΑ

ΤΕΙ ΜΕΣΟΛΟΓΓΙΟΥ
ΒΙΒΛΙΟΘΗΚΗ

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ

Δρ. ΔΡΟΣΟΣ ΛΑΜΠΡΟΣ

ΕΥΧΑΡΙΣΤΙΕΣ

Ολοκληρώνοντας αυτή την εργασία θα ήθελα να ευχαριστήσω θερμά τον επόπτη καθηγητή μου κ. Δρόσο Λάμπρο. Οι συζητήσεις μαζί του ήταν κάτι παραπάνω από επικοδομητικές. Είχαμε μια πολύ καλή συνεργασία και συνέβαλε στην ολοκλήρωση της εργασίας μου κάνοντας πολύ χρήσιμες παρατηρήσεις και επισημάνσεις.

Τέλος, θεωρώ υποχρέωση και δεν θα ήθελα να παραλείψω να τονίσω και να ευχαριστήσω πάρα πολύ για την αμέριστη συμπαράσταση της οικογένειάς μου η οποία ήταν εμφανής τόσο ψυχικά όσο και υλικά, καθ'όλη την διάρκεια προετοιμασίας της πτυχιακής μου εργασίας καθώς επίσης και κατά την διάρκεια των σπουδών μου.

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΡΟΛΟΓΟΣ.....1

ΚΕΦΑΛΑΙΟ 1

1.1 WORLD WIDE WEB (WWW) 2

1.1.1 Ανάγκη και λόγοι ύπαρξης του WWW 2
Η δημιουργία του WWW. Σύντομη ιστορική αναδρομή 3
Σε ποιους απευθύνεται το WWW.....3
Η λειτουργία του World Wide Web σε βάθος.....4
Τι είναι το WWW.....4
Πως λειτουργεί το WWW 5

1.1.2 Η έννοια του Hypertext και των Hypermedia.....7
Η έννοια του υπερκειμένου 7
Η έννοια των υπερμέσων.....9
Το πρωτόκολλο HTTP.....10

1.1.3 World Wide Web Servers.....11
NCSA httpd.....11
Apache.....11

1.1.4 Εισαγωγή στο Web Authoring.....12
Τα περιεχόμενα ενός Web document.....12
Η γλώσσα επισήμανσης υπερκειμένου HTML.....13
Γενικοί κανόνες σύνταξης.....14
Μορφοποίηση και οργάνωση του κειμένου.....15
Κατάλογοι.....16
Εικόνα.....17
Σύνδεσμοι.....17
Image Maps.....18
Ήχος και κίνηση.....18
Αμφίδρομη επικοινωνία.....18
CGI Scripts.....19
Πίνακες.....20
Τι είναι το Web Authoring.....20
Η εμπορική εταιρεία.....21
Ο μεμονωμένος χρήστης.....22

1.2 Επισκόπηση της Διαχείρισης Βάσεων Δεδομένων 23

1.2.1 Ένα εισαγωγικό παράδειγμα.....23

1.2.2 Τι είναι σύστημα βάσης δεδομένων 26
Δεδομένα.....26
Υλικό.....28

Λογισμικό....29
Χρήστες....29

1.2.3 Τι είναι βάση δεδομένων....31
Μόνιμα δεδομένα....31
Οντότητες και συσχετίσεις....32
Ιδιότητες....35

1.2.4 Γιατί βάση δεδομένων;....36
Εποπτεία διαχείρισης δεδομένων και βάσεων δεδομένων....37
Πλεονεκτήματα της χρήσης βάσεων δεδομένων....37

1.2.5 Ανεξαρτησία δεδομένων....41

1.2.6 Σχεσιακά και μη σχεσιακά συστήματα....47

1.2.7 Περίληψη....50

1.3 Τι είναι η PHP;....51

1.3.1 Εισαγωγή....51

1.3.2 Τι μπορεί να κάνει η PHP;....52

1.3.3 Τι είναι η PEAR;....54

ΚΕΦΑΛΑΙΟ 2

2.1 Βασικοί Κανόνες Σύνταξης της PHP....56

2.1.1 Βγαίνοντας από την HTML....56
Διαχωρισμός εντολών....57
Σχόλια....59

2.1.2 Μεταβλητές....58

2.1.3 Τύποι Δεδομένων....60
Αλλαγή του Τύπου Δεδομένων με την settype()....62
Γιατί Πρέπει να Ελέγχουμε τον Τύπο Δεδομένων;....63

2.1.4 Τελεστές και Εκφράσεις....63
Ο Τελεστής Εκχώρησης Τιμής....64
Αριθμητικοί Τελεστές....64
Ο Τελεστής Συνένωσης....65
Σύνθετοι Τελεστές Εκχώρησης Τιμής....65
Αυτόματη Αύξηση/Μείωση της Τιμής μιας Ακέραιας Μεταβλητής....66
Τελεστές Σύγκρισης....67

Δημιουργία Πολύπλοκων Εκφράσεων Ελέγχου με Λογικούς Τελεστές....69
Προτεραιότητα των τελεστών....70

2.1.5 Σταθερές....71
Προκαθορισμένες Σταθερές....72

2.1.6 Οι Εντολές Ελέγχου της Ροης Εκτελεσης....72
Η Εντολή if....73
Χρήση της Πρότασης else στην Εντολή if....73
Χρήση της Πρότασης elseif στην Εντολή if....74
Η Εντολή switch....74

2.1.7 Βρόχοι....75
Η Εντολή while....75
Η Εντολή do...while....76
Η Εντολή for....76
Έξοδος από έναν Βρόχο με την Εντολή break....77
Παράκαμψη μιας Επανάληψης με την Εντολή continue....78

2.1.8 Συναρτήσεις....78
Ορισμός μιας Συνάρτησης....78

2.1.9 Το Εύρος των Μεταβλητών....79
Προσπέλαση Μεταβλητών με την Εντολή global....79

2.1.10 Προκαθορισμένες Μεταβλητές....79

2.2 Βασικές Εντολές της MySQL....80

2.2.1 Οι Τύποι Δεδομένων της MySQL....81
Οι Αριθμητικοί Τύποι Δεδομένων....81
Τύποι Δεδομένων για Τιμές Ημερομηνίας/Ωρας....83
Τύποι Αλφαριθμητικών....84

2.2.2 Η Εντολή για την Δημιουργία Πινάκων....85

2.2.3 PHPMyadmin, διαχείριση της MySQL μέσω του web....89
Τί είναι η phpMyAdmin;....89

2.2.4 Χρήση της Εντολής INSERT....93
Αναλυτική Παρουσίαση της Εντολής INSERT....94

2.2.5 Χρήση της Εντολής SELECT....96
Ταξινόμηση των Αποτελεσμάτων που Επιστρέφει η Εντολή SELECT....98
Περιορίζοντας τα Αποτελέσματα που Παίρνουμε με την SELECT....100

2.2.6 Χρήση της Πρότασης WHERE στα Ερωτήματά μας 101
Χρήση Τελεστών σε Προτάσεις WHERE 103
Συγκρίσεις Αλφαριθμητικών με τον Τελεστή LIKE.....104

2.2.7 Επιλογή Δεδομένων από Πολλαπλούς Πίνακες 105

2.2.8 Χρήση της JOIN.....108

2.2.9 Χρήση της Εντολής UPDATE για την Τροποποίηση Εγγραφών.....112
Ενημέρωση Υπό Συνθήκες.....112

2.2.10 Χρήση της Εντολής REPLACE.....113

2.2.11 Χρήση της Εντολής DELETE 115
Διαγραφή Υπό Συνθήκες 116
Ανασκόπηση 117

ΚΕΦΑΛΑΙΟ 3

3.1 Εγκατάσταση και Διαμόρφωση των MySQL, Apache και PHP.....118

3.1.1 Εγκατάσταση της MySQL στα Windows.....118

3.1.2 Εγκατάσταση του Apache στα Windows.....118
Τρέχοντας τον Apache σε ένα Παράθυρο Κονσόλας.....119
Εκτελώντας τον Apache ως Υπηρεσία.....119

3.1.3 Εγκατάσταση και Διαμόρφωση της PHP στα Windows.....120

3.1.4 Εγκατάσταση της PEAR.....122

3.2 Επικοινωνία με Βάσεις Δεδομένων MySQL μέσω PHP.....124

3.2.1 Σύνδεση στην MySQL με την PHP 125
Χρήση της Συνάρτησης mysql_connect().....125
Εκτέλεση Ερωτημάτων.....126
Ανάκτηση Μηνυμάτων Σφάλματος.....128

3.2.2 Δουλεύοντας με Δεδομένα της MySQL.....129
Εισαγωγή Δεδομένων σε έναν Πίνακα μέσω PHP.....129
Ανασκόπηση 136

ΚΕΦΑΛΑΙΟ 4

4.1 Συγγραφή Κώδικα – Παρουσίαση Έργου Επεξεργασίας Έκτακτων Εκπαιδευτικών.....137
Δημιουργία Κεντρικής Σελίδας.....137
Ανασκόπηση 145
ΒΙΒΛΙΟΓΡΑΦΙΑ – ΔΙΚΤΥΑΚΟΙ ΤΟΠΟΙ 146

ΠΡΟΛΟΓΟΣ

Σκοπός της παρούσας εργασίας είναι να μελετήσει και να υλοποιήσει εφαρμογές σχετικά με τον τρόπο τον οποίο τα συστήματα διαχειρίζονται τις πληροφορίες μεσω του Web. Συγκεκριμένα, θα ασχοληθούμε με την διαχείριση του προγράμματος μαθημάτων, του τμήματος ΕΠΔΟ και με την επεξεργασία του πίνακα των έκτακτων εκπαιδευτικών.

Η εργασία αποτελείται από τέσσερα κεφάλαια και έχει σαν κεντρικό άξονα την εξέταση και υλοποίηση μιας εφαρμογής, της διαχείρισης προγράμματος της ΕΠΔΟ και της διαχείρισης έκτακτων εκπαιδευτικών.

Στο πρώτο κεφάλαιο, αναφέρονται οι βασικές έννοιες και η λειτουργία του World Wide Web (WWW), τα πρωτόκολλα που χρησιμοποιεί, οι WWW servers οι οποίοι είναι υπεύθυνοι για την διανομή των πληροφοριών μέσα στο Web(όπως το πρόγραμμα εξυπηρέτησης Apache) καθώς επισης και οι βασικοί κανόνες σύνταξης της γλώσσας HTML. Επίσης, ορίζει το γενικό πλαίσιο, εξηγώντας τί είναι βάση δεδομένων και γιατί είναι γενικά επιθυμητά τα συστήματα βάσεων δεδομένων. Περιγράφει επίσης σε συντομία τη διαφορά ανάμεσα στα σχεσιακά συστήματα βάσεων δεδομένων και τα υπόλοιπα. Επίσης αναφέρεται τι είναι η PHP και τι μπορεί να κάνει η scripting γλώσσα προγραμματισμού.

Στο δεύτερο κεφάλαιο, παρουσιάζονται τα βασικά στοιχεία της γλώσσας PHP και μας εξοικειώνει με την διαδικασία συγγραφής κώδικα, αποστολής αρχείων στον server και ελέγχου των αποτελεσμάτων. Επίσης αναφέρονται οι βασικές εντολές της MySQL και παρουσιάζονται τα βήματα για την δημιουργία της βάσης δεδομένων programma.

Στο τρίτο κεφάλαιο, παρουσιάζονται οι διαδικασίες εγκατάστασης και διαμόρφωσης των PHP, MySQL και Apache.Ένα μέρος αυτού του κεφαλαίου είναι αφιερωμένο στην συγγραφή κώδικα με την χρήση της MySQL σε συνδυασμό με την PHP.Καλύπτει διάφορα θέματα ανάπτυξης εφαρμογών, όπως η χρήση και διαχείριση φορμών.

Τέλος, στο κεφάλαιο τέσσερα, παρουσιάζεται η διαδικασία ανάπτυξης μιας απλής αλλά χρήσιμης εφαρμογής με βάση τις γνώσεις που έχουμε αποκτήσει στα προηγούμενα κεφάλαια. Συγκεκριμένα παρουσιάζεται η διαχείριση του πίνακα των έκτακτων εκπαιδευτικών (δηλ.εισαγωγή, διαγραφή και εμφάνιση των εκπαιδευτικών).

ΚΕΦΑΛΑΙΟ 1

1.1 WORLD WIDE WEB (WWW)

Εισαγωγή στο World Wide Web.

1.1.1 Ανάγκη και λόγοι ύπαρξης του WWW

Η απλούστερη περιγραφή που θα μπορούσε να δώσει κανείς για το Internet, η οποία συνοψίζει ταυτόχρονα και την ουσία της ύπαρξης και της λειτουργίας του, είναι ότι αυτό αποτελεί έναν "χώρο" όπου αποθηκεύεται η πληροφορία. Από την ίδια τη φύση της όμως, η πληροφορία δεν μπορεί να είναι στατική, διότι τότε χάνει την έννοιά της. Υφίσταται μόνο δυναμικά, υποκείμενη σε διαρκή μετάδοση και μεταλλαγή. Επεκτείνοντας λοιπόν την αρχική περιγραφή, ένας πιο ακριβής ορισμός θα έλεγε ότι το Internet είναι ένας χώρος αποθήκευσης αλλά ταυτόχρονα και ένα μέσο μετάδοσης, επεξεργασίας και μετεξέλιξης της πληροφορίας.

Κατά τη διάρκεια της εξέλιξης του δικτύου, εμφανίστηκαν και δοκιμάστηκαν πολλά εργαλεία, μέθοδοι και τρόποι αποθήκευσης και μετάδοσης της πληροφορίας, αρκετοί από τους οποίους πέτυχαν και κατάφεραν τελικά να καθιερωθούν. Το anonymous FTP για παράδειγμα, εισήγαγε πρώτο την έννοια του "αρχείου" στο επίπεδο του Internet, καθιστώντας έτσι δυνατή για οποιονδήποτε χρήστη την πρόσβαση σε προγράμματα και δεδομένα διασπαρμένα σε ολόκληρη την υφήλιο, ανεξάρτητα από τη γεωγραφική περιοχή στην οποία αυτός βρίσκεται. Παράλληλα, η προσθήκη εργαλείων αναζήτησης, όπως είναι π.χ. ο Archie, έδωσε τη δυνατότητα στους χρήστες να εντοπίζουν κάποιο συγκεκριμένο site όπου βρίσκεται αποθηκευμένο ένα αρχείο που αυτοί χρειάζονται.

Οπως παρατηρούμε όμως από τα παραπάνω, πρόκειται για δύο διαφορετικά πράγματα, δύο διαφορετικές υπηρεσίες του Internet οι οποίες είναι εντελώς ανεξάρτητες η μία από την άλλη. Εφόσον δηλαδή κάποιος χρήστης "εντοπίσει" ένα αρχείο της αρεσκείας του χρησιμοποιώντας ένα εργαλείο αναζήτησης όπως είναι ο Archie, θα πρέπει στη συνέχεια να χρησιμοποιήσει την υπηρεσία του FTP για να το μεταφέρει στο υπολογιστικό του σύστημα. Αν και στη συνέχεια αναπτύχθηκε μία πληθώρα νέων εργαλείων τα οποία επέκτειναν τις δυνατότητες αναζήτησης μίας πληροφορίας, π.χ. το WAIS το οποίο δίνει δυνατότητες εκτέλεσης αναζητήσεων πλήρους κειμένου (full-text) σε βάσεις δεδομένων ή ο Gopher ο οποίος ενσωματώνει δυνατότητες αναζήτησης και ανάκτησης αρχείων ή κειμένων, κατά βάση όλες οι λειτουργίες ή τα σχετικά εργαλεία εξακολουθούσαν να υφίστανται και να λειτουργούν ανεξάρτητα το ένα από το άλλο.

Το αποφασιστικό βήμα για την ολοκλήρωση (integration) της αναζήτησης της πληροφορίας στο Internet έγινε με την εισαγωγή μίας νέας υπηρεσίας που ονομάζεται World Wide Web ή WWW και η οποία διαδραμάτισε σημαντικότατο ρόλο στην εξάπλωση του δικτύου αυτού.

Η δημιουργία του WWW. Σύντομη ιστορική αναδρομή.

Το αποφασιστικό βήμα για την ολοκλήρωση (integration) της αναζήτησης της πληροφορίας στο Internet ξεκίνησε στα τέλη του 1991, όταν το ευρωπαϊκό ερευνητικό κέντρο για την πυρηνική φυσική και τη φυσική των στοιχειωδών σωματιδίων CERN, στη Γενεύη της Ελβετίας, εγκατέστησε τον πρώτο ηλεκτρονικό υπολογιστή που υποστήριζε την νέα υπηρεσία του WWW (Web server). Η προεργασία, βέβαια, είχε ξεκινήσει από τα μέσα του 1989, ενώ η αρχική ιδέα είχε διατυπωθεί πολύ πιο πριν.

Τριάντα περίπου χρόνια νωρίτερα, ένας φοιτητής του αμερικανικού πανεπιστημίου Harvard είχε πρώτος διατυπώσει την ανάγκη για μία τεχνολογία η οποία θα επέτρεπε στον χρήστη να έχει πρόσβαση σε πολλές διαφορετικές μορφές δεδομένων (κείμενο, εικόνα, ήχο, video κλπ.) μέσα από το ίδιο πρόγραμμα πελάτη (client program). Τα μέσα της εποχής εκείνης δεν ήταν βεβαίως επαρκή, αλλά η εισαγωγή του WWW ή W3, όπως συχνά συμβολίζεται, το έκανε τελικά πραγματικότητα, έστω και με καθυστέρηση τριάντα ολόκληρων χρόνων.

Σε ποιους απευθύνεται το WWW

Το Internet, αυτό το παγκόσμιο υπολογιστικό δίκτυο, έχει μπει για τα καλά στη ζωή μας. Είτε πρόκειται για επιχειρήσεις, είτε για ελεύθερους επαγγελματίες, είτε ακόμη και για απλούς ιδιώτες, υπάρχουν υπηρεσίες και πληροφορίες σε αυτόν τον απέραντο ηλεκτρονικό χώρο που ελκύουν τον καθένα. Σταδιακά λοιπόν, το Internet μετατρέπεται από δύσχρηστο εργαλείο στα χέρια των ακαδημαϊκών ερευνητών, των ανθρώπων της πληροφορικής και των στρατιωτικών σε πεδίο δράσης ακόμη και των απλών ανθρώπων, μεμονωμένων χρηστών που θέλουν να αξιοποιήσουν τα όσα θαυμαστά αυτό προσφέρει.

Σημαντικό ρόλο σε αυτήν την εξάπλωση διαδραμάτισε ο περίφημος "Παγκόσμιος Ιστός" της πληροφόρησης ή World Wide Web, ο οποίος απευθύνεται πρακτικά σε όλον τον ενεργό πληθυσμό, αν και βέβαια απέχει αρκετά ακόμη από το σημείο της αποδοχής του από το ευρύτερο κοινό. Οι πληροφορίες που εμπεριέχονται στον WWW ανταποκρίνονται σε ένα ευρύτατο φάσμα ενδιαφερόντων, από τέχνη, πολιτική, αθλητικά και πολιτιστικά μέχρι επιστημονικές εργασίες, δελτία καιρού, χρηματοοικονομικά στοιχεία, πληροφορική, βιολογία κλπ., δηλαδή οτιδήποτε μπορεί να φανταστεί κανείς, και αυτό δεν είναι υπερβολή αλλά η πραγματικότητα.

Παλαιότερα, οι κυριότεροι χρήστες του Web, αλλά και του Internet γενικότερα, ήταν κατά μείζονα λόγο άνθρωποι που είχαν σχετική εμπειρία στο χώρο των υπολογιστών. Τώρα πλέον που οι υπολογιστές έχουν εισέλθει δυναμικά στην αγορά των καταναλωτικών προϊόντων και συμπληρώνουν τον οικιακό εξοπλισμό πολλών κατοικιών, όλο και περισσότεροι χρήστες αποφασίζουν να αποκτήσουν πρόσβαση σε αυτήν την ηλεκτρονική κοινωνία. Ένας γιατρός, για παράδειγμα, μπορεί άνετα να συνδεθεί με νοσοκομεία και εργαστήρια του εξωτερικού ώστε να ενημερωθεί για νέους ιούς, ασθένειες, εμβόλια και να συγκεντρώσει διάφορα ιατρικά papers με επιστημονικό υλικό, "κατεβάζοντας" παράλληλα και εικόνες που μπορεί να αποθηκεύσει στον υπολογιστή του. Ένας μηχανικός θα πληροφορηθεί την πορεία των

projects που βρίσκονται σε εξέλιξη και θα δει on-line σχέδια και διαγράμματα. Ένας μαθητής ή φοιτητής θα επισκεφθεί κόμβους με εκπαιδευτικό περιεχόμενο και χάρη στη νέα multimedia φύση του Web (ιδιαίτερη αναφορά για την οποία γίνεται παρακάτω) μπορεί να "περιηγηθεί" σε ηλεκτρονικές εγκυκλοπαίδειες και να ακούσει συνοδευτικά αρχεία ήχου. Ένας απλός χρήστης θα "περιπλανηθεί" απλώς στον μαγευτικό κόσμο του Web για να συλλέξει στοιχεία που ελκύουν την προσοχή του ή για να περάσει λίγες ώρες με ευχάριστο τρόπο. Ολα αυτά καθιστούν το World Wide Web ιδιαίτερα δημοφιλή και ωθούν όλο και περισσότερους χρήστες να συνδεθούν με το παγκόσμιο δίκτυο που λέγεται Internet.

Η λειτουργία του World Wide Web σε βάθος.

Στην ακόλουθη ενότητα περιγράφεται με λεπτομέρειες τι ακριβώς είναι το World Wide Web και πως αυτό λειτουργεί.

Τι είναι το WWW

Η μεγάλη δημοσιότητα που γνωρίζει τον τελευταίο καιρό το Internet οφείλεται κατά ένα μεγάλο ποσοστό στο World Wide Web (WWW). Το Web, όπως το ονομάζουν οι χρήστες του χάριν συντομίας, είναι μία σχετικά νέα υπηρεσία του δικτύου Internet που του δίνει εξαιρετικά φιλική όψη αλλά και δυνατότητες Multimedia. Εάν θέλαμε να δώσουμε έναν ορισμό για το τι ακριβώς είναι το Web, θα λέγαμε ότι πρόκειται για ένα δίκτυο υπολογιστών/διανομέων (servers) που χρησιμοποιεί συνδέσμους υπερκειμένου για την προσπέλαση HTML εγγράφων. Η περιγραφή αυτή δεν είναι αρκετά σαφής για έναν αρχάριο χρήστη, αλλά ευθύς αμέσως ακολουθεί μία πιο αναλυτική περιγραφή για τα συνθετικά στοιχεία του Web και τον ρόλο τους.

Ας υποθέσουμε λοιπόν ότι έχουμε πολλούς υπολογιστές διασυνδεδεμένους μέσω καλωδιώσεων σε ένα είδος δικτύου ευρείας περιοχής WAN (Wide Area Network), που είναι ουσιαστικά η ίδια ανάπτυξη που διακρίνουμε σε όλο το Internet. Κάθε ένας από αυτούς τους υπολογιστές καλείται **εξυπηρετητής** ή **διανομέας Web** (Web server) και η ιδιαιτερότητά του είναι ότι έχει αποθηκευμένα ηλεκτρονικά έγγραφα γραμμένα σε μία γλώσσα που ονομάζεται **HTML**, από τα αρχικά των αγγλικών όρων HyperText Markup Language (Γλώσσα Επισήμανσης Υπερκειμένου), για την οποία γίνεται ιδιαίτερη αναφορά στην αντίστοιχη ενότητα. Τα έγγραφα αυτά μπορούν να περιέχουν ένα πλούσιο φάσμα δεδομένων, όπως π.χ. κείμενο, πίνακες, φόρμες, γραφικά, ήχο και video, και αποκαλούνται Web documents ή Web pages ή HTML pages. Πρόκειται, δηλαδή, για ένα αλληλεπιδραστικό multimedia περιβάλλον που προσφέρει τη δυνατότητα στον χρήστη να δημιουργήσει εντυπωσιακά ηλεκτρονικά έγγραφα και να τα "δημοσιεύσει" στον Web, καθιστώντας τα έτσι προσβάσιμα για οποιονδήποτε ενδιαφερόμενο.

Αυτό που κάνει τα έγγραφα ξεχωριστά είναι οι περίφημοι **υπερσύνδεσμοι** (hyperlinks). Αυτοί δεν είναι παρά περιοχές σε ένα έγγραφο που όταν "ενεργοποιηθούν" από τον χρήστη τον οδηγούν σε κάποιο άλλο Web document, όχι

απαραίτητα στον ίδιο υπολογιστή, το οποίο περιέχει πιθανότατα περισσότερες πληροφορίες για το θέμα του υπερσυνδέσμου που ενεργοποιήθηκε. Για παράδειγμα, σε ένα ηλεκτρονικό κείμενο σχετικό με τις Γλώσσες Προγραμματισμού εμφανίζεται με διαφορετικό χρώμα η λέξη *Μεταγλωττιστής*. Το διαφορετικό αυτό χρώμα δηλώνει συνήθως ότι πρόκειται για σύνδεσμο και επιτρέπει στον ενδιαφερόμενο χρήστη να επιλέξει τη λέξη αυτή. Η επιλογή της λέξης *Μεταγλωττιστής* λοιπόν από έναν χρήστη θα τον οδηγήσει αυτόματα σε ένα άλλο έγγραφο, ίσως και σε έναν διαφορετικό Web server, όπου θα αναγράφονται αναλυτικότερες πληροφορίες για τους μεταγλωττιστές των διάφορων γλωσσών προγραμματισμού. Με αυτόν τον τρόπο, τα διάφορα έγγραφα συνδέονται σε ένα τεράστιο πληροφοριακό πλέγμα, γεγονός που δικαιολογεί και την ιδιόρρυθμη ονομασία της υπηρεσίας του World Wide Web (Παγκόσμια Επεκτεινόμενος Ιστός).

Ο Web είναι δομημένος σε μία αρχιτεκτονική client/server. Σε μία απλουστευμένη προσέγγιση, ο server κρατά τα ηλεκτρονικά έγγραφα μέσα στα οποία βρίσκονται οι διάφοροι σύνδεσμοι και "διοχετεύει" τις πληροφορίες προς τον client, χειριζόμενος ταυτόχρονα και τις αιτήσεις για επικοινωνία και διασύνδεση με άλλους απομακρυσμένους υπολογιστές (hosts). Το πρωτόκολλο που χρησιμοποιείται από τον Web για την εκτέλεση όλων αυτών των διεργασιών είναι το **HTTP**, από τα αρχικά των αγγλικών όρων HyperText Transfer Protocol (Πρωτόκολλο Μεταφοράς Υπερκειμένου), για το οποίο αναπτύσσονται περισσότερα στην αντίστοιχη ενότητα.

Πως λειτουργεί το WWW

Η ουσία της λειτουργίας του WWW είναι -στη βάση της- ιδιαίτερα απλή. Ο χρήστης έχει μπροστά του ένα κείμενο. Κάποια σημεία αυτού του κειμένου που ονομάζονται **λέξεις-κλειδιά** (keywords) έχουν διασυνδέσεις (links) με κάποια άλλα κείμενα. Οταν λοιπόν ο χρήστης επιλέξει, με διάφορους τρόπους, κάποια από αυτές τις λέξεις ή φράσεις, ενεργοποιείται αυτόματα η διασύνδεση και αμέσως έχει αυτός μπροστά του το νέο κείμενο. Πρόκειται δηλαδή για μια εφαρμογή **υπερκειμένου** (hypertext) μέσω της οποίας ανατρέπεται η "παραδοσιακή" γραμμική ροή της πληροφορίας και ο χρήστης ασκεί ο ίδιος τον έλεγχο αναφορικά με τη δομή της παρουσίασης και της πρόσληψής της. Το υπερκείμενο δεν είναι τίποτε άλλο από ένα κείμενο όπου ορισμένες λέξεις-κλειδιά έχουν την ιδιότητα να οδηγούν σε άλλο επεξηγηματικό κείμενο, εικόνα, ήχο ή ακόμη και video.

Σε όλους μας έχει τύχει να συναντήσουμε κάποια εφαρμογή hypertext- τα αρχεία ή μενού βοήθειας (help menus) των Windows αποτελούν ίσως το πιο κοινό παράδειγμα- και έτσι η έννοια μπορεί να θεωρηθεί ήδη γνωστή (αν και σε αντίστοιχη ενότητα η έννοια αυτή περιγράφεται λεπτομερώς). Υπάρχουν όμως κάποιες βασικές διαφορές στην εφαρμογή της στο Internet μέσω του World Wide Web :

1. Η κατάληξη ενός link (το κείμενο δηλαδή το οποίο εμφανίζεται όταν το ενεργοποιήσει ο χρήστης) δεν είναι μόνο τοπική, στον ίδιο υπολογιστή. Το WWW συνιστά μία δομή η οποία "αγκαλιάζει" ολόκληρο το δίκτυο και συνδέει μεταξύ τους πηγές πληροφοριών (documents με την ευρεία έννοια του όρου στα αγγλικά) που βρίσκονται σε απομακρυσμένους υπολογιστές. Από εκεί προέρχεται και όρος Web που σημαίνει Ιστός. Συνεπώς, ο χρήστης

μπορεί να αρχίσει με ένα κείμενο το οποίο βρίσκεται σε κάποιον υπολογιστή στην Αθήνα και από εκεί να μεταφερθεί σε κάποια άλλα κείμενα που βρίσκονται στη Γενεύη, από εκεί στη Νέα Υόρκη κ.ο.κ.

2. Μέσω του World Wide Web, ο χρήστης δεν έχει πρόσβαση μόνο σε κείμενο αλλά σε ηλεκτρονικά έγγραφα με ευρύτερο περιεχόμενο. Κάποιο link, δηλαδή, μπορεί να τον διασυνδέσει με κάποιο αρχείο ήχου, video ή animation το οποίο, με το κατάλληλο client πρόγραμμα, μπορεί ο χρήστης να ακούσει ή να δει αντίστοιχα. Επιπλέον, ο Web έχει τη δυνατότητα να διασυνδέσει τον χρήστη με άλλους πόρους ή εργαλεία του Internet. Αυτό σημαίνει ότι ένα link μπορεί να ξεκινήσει μία διεργασία FTP για να μεταφέρει ο χρήστης κάποιο αρχείο στον υπολογιστή του, μπορεί να του δώσει πρόσβαση στα μενού μίας υπηρεσίας Gopher, να τον οδηγήσει σε κάποια βάση δεδομένων του WAIS, να τον συνδέσει με έναν απομακρυσμένο υπολογιστή μέσω Telnet κ.λ.π.

3. Πρέπει, τέλος, να επισημανθεί ότι όλες αυτές οι διαδικασίες επιτελούνται εντελώς διάφανα. Ο χρήστης απλώς ενεργοποιεί το link (με ένα κλικ του ποντικιού, κινούμενος με τα arrow keys και πατώντας <ENTER> ή απλά επιλέγοντας κάποιον αριθμό) και στη συνέχεια όλα γίνονται αυτόματα. Επιπρόσθετα, είναι προφανές ότι ο χρήστης δεν χρειάζεται να γνωρίζει τη διεύθυνση των διαδοχικών sites, με τα οποία αυτός συνδέεται ακολουθώντας κάποια links, μια και οι διευθύνσεις αυτές είναι "ενσωματωμένες" στα διάφορα ηλεκτρονικά έγγραφα και το client πρόγραμμα τις "διαβάζει" από μόνο του.

Κάθε αντικείμενο στο Internet καθορίζεται βάσει μίας διεύθυνσης που ονομάζεται URL (Uniform Resource Locator) και η οποία παρέχει τα απαραίτητα στοιχεία για την ανάκτηση των εγγράφων/πληροφοριών. Για παράδειγμα, η διεύθυνση **http://www.ntua.gr/info.html** προσδιορίζει την ανάκτηση ενός αρχείου-εγγράφου με όνομα **info.html**, το οποίο είναι γραμμένο στην γλώσσα HTML, από τον κόμβο **www.ntua.gr** του δικτύου χρησιμοποιώντας το πρωτόκολλο HTTP, στο οποίο στηρίζεται η μεταφορά δεδομένων μέσω του Web. Ο εξυπηρετητής (server) του δικτύου είναι ο απομακρυσμένος αυτός υπολογιστής που παραδίδει τα επιθυμητά δεδομένα. Ο πελάτης (client) είναι ο τελικός χρήστης που αναζητά τα δεδομένα αυτά. Αυτός "ανοίγει" μία σύνδεση προς τον server, υποβάλλει μία αίτηση και λαμβάνει την αντίστοιχη απάντηση. Στη συνέχεια, η συγκεκριμένη σύνδεση "κλείνει". Μόνο ένα μικρό τμήμα του URL δίνεται από τον client στον server. Το υπόλοιπο είναι γνωστό. Όταν ένα πλήρες URL περνάει από τον client στον server, τότε λέμε ότι έχουμε μία αίτηση proxy (proxy request), την οποία πρέπει να διαβάσει ο server εκ μέρους του client. Με απλά λόγια, σε αυτήν την περίπτωση ο server λειτουργεί και αυτός με τη σειρά του ως client, με σκοπό την καλύτερη εξυπηρέτηση της αρχικής αίτησης του χρήστη.

Με αυτόν τον τρόπο, καθίσταται δυνατή η ενοποίηση μέσω του World Wide Web όλων σχεδόν των υπηρεσιών του Internet. Ετσι, μπορούμε να έχουμε ηλεκτρονικά έγγραφα που μέσω του πρωτοκόλλου HTTP έχουν τη δυνατότητα να συνεργάζονται με άλλα πρωτόκολλα. Δηλαδή ο χρήστης, μέσω των proxy servers, μπορεί να συνδέεται με mail servers για την ανάγνωση των μηνυμάτων ηλεκτρονικού ταχυδρομείου, με FTP servers για την ανάκτηση αρχείων, με Gopher servers για την άντληση πληροφοριών κ.λ.π. Επίσης, οι proxy servers παρέχουν επιπρόσθετα και το

πλεονέκτημα ενός τοπικού disk cache, που επιταχύνει την πρόσβαση σε URL που χρησιμοποιούνται συχνά. Ο WWW server είναι υπεύθυνος για την αντιστοίχηση ενός παρεχόμενου URL με κάποιο αντικείμενο ή την εμφάνιση ενός μηνύματος λάθους στην περίπτωση που το αντικείμενο αυτό δεν υπάρχει. Είναι σχεδιασμένοι για να αποκρύπτουν τις λεπτομέρειες υλοποίησης από τον τελικό χρήστη και παρέχουν ευκολίες μέσω configuration files για την καλύτερη διαχείριση των καταλόγων.

Βάσει όσων αναφέρθηκαν παραπάνω, το WWW ξεφεύγει πλέον από το απλό hypertext και γίνεται ένα πλήρες εργαλείο υπερμέσων, ή hypermedia εργαλείο (οι όροι hypertext και hypermedia αναπτύσσονται λεπτομερώς σε ειδική ενότητα). Από τη στιγμή μάλιστα που επιτρέπει στον χρήστη, για παράδειγμα, να συμπληρώνει φόρμες (ερωτηματολόγια, on-line εγγραφές, συνδρομές κ.λ.π.), να επιλέγει μετακινώντας το ποντίκι σε διάφορα σημεία ενός γεωγραφικού χάρτη ή γραφήματος γενικότερα, να διασκεδάζει παίζοντας on-line παιχνίδια, τότε ίσως θα έπρεπε να μιλούμε για ένα αλληλεπιδραστικό περιβάλλον πολυμέσων και υπερμέσων (interactive multimedia και hypermedia εργαλείο), αν και οπωσδήποτε υπολείπονται αρκετά ακόμη στον τομέα αυτό.

Συνοψίζοντας, θα μπορούσε να δοθεί μία πιο "τεχνική" περιγραφή του WWW ως μία προσπάθεια ενοποίησης του τεράστιου όγκου πληροφοριών διαφορετικής μορφής που υπάρχουν σε ένα δίκτυο με τη βοήθεια ενός απλού πρωτοκόλλου. Συνακόλουθα, πρόκειται για μία προσπάθεια ενοποίησης διάφορων εργαλείων δικτύου (network tools) και βελτιστοποίησης της χρήσης των πόρων του δικτύου.

1.1.2 Η έννοια του Hypertext και των Hypermedia.

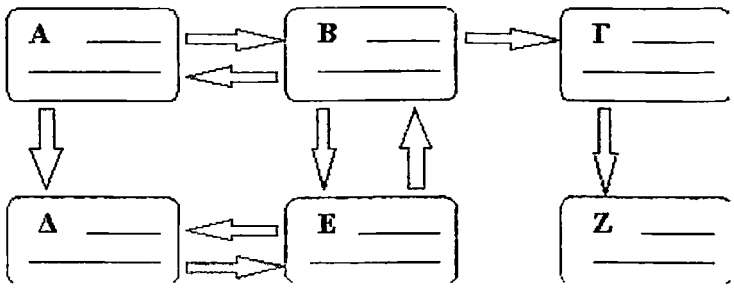
Στην ενότητα αυτή κρίνεται σκόπιμο να αναπτυχθούν με λεπτομέρειες οι έννοιες του υπερκειμένου (hypertext) και των υπερμέσων (hypermedia), έτσι ώστε ο ενδιαφερόμενος αναγνώστης να κατανοήσει καλύτερα τους βασικούς αυτούς όρους πάνω στους οποίους στηρίζεται ολόκληρη η λειτουργία του World Wide Web.

Η έννοια του υπερκειμένου

Ο απλούστερος ίσως τρόπος για να ορίσουμε το υπερκείμενο (hypertext) είναι να το αντιπαραθέσουμε και να το συγκρίνουμε με το κατεξοχήν παραδοσιακό κείμενο, όπως είναι αυτό ενός π.χ. βιβλίου. Το παραδοσιακό κείμενο, είτε αυτό είναι τυπωμένο είτε με τη μορφή αρχείων ενός υπολογιστή, είναι γραμμικό ή αλλιώς σειριακό, το οποίο απλά σημαίνει ότι υπάρχει ένας μοναδικός τρόπος που ορίζει τη σειρά με την οποία το κείμενο θα διαβαστεί. Ετσι, αρχικά διαβάζουμε την πρώτη σελίδα, στη συνέχεια τη δεύτερη, μετά την τρίτη κ.ο.κ. Με άλλα λόγια, δεν χρειάζεται να σκεφτούμε ως μαθηματικοί για να βρούμε έναν γενικό τρόπο, μία συγκεκριμένη φόρμουλα, βάσει του οποίου θα διαβάσουμε το κείμενο.

Το υπερκείμενο, αντίθετα, είναι μη γραμμικό ή αλλιώς μη σειριακό. Εδώ, δεν υπάρχει κάποιος συγκεκριμένος τρόπος που να προσδιορίζει τη σειρά με την οποία θα

διαβαστεί το κείμενο αυτό. Το ακόλουθο σχήμα αποτελεί ένα απλό παράδειγμα υπερκειμένου :



Σχ. Η δομή του υπερκειμένου

Ας υποθέσουμε ότι ξεκινάμε διαβάζοντας πρώτα το απόσπασμα **Α** του κειμένου. Εδώ όμως, αντί για μία και μοναδική επιλογή, όπως θα είχαμε σε ένα παραδοσιακό κείμενο, αυτή η δομή υπερκειμένου έχει δύο διαφορετικούς εναλλακτικούς τρόπους ανάγνωσης για τον αναγνώστη: μετά το **Α** δηλαδή, μπορούμε να συνεχίσουμε με το απόσπασμα **Β** ή το **Δ**. Υποθέτοντας τώρα ότι αποφασίζουμε να διαβάσουμε το κείμενο **Β**, έχουμε στη συνέχεια τη δυνατότητα να πάμε στο **Γ** ή το **Ε**, και από το **Ε** να μεταφερθούμε στο απόσπασμα **Δ**. Από τη στιγμή που είχαμε τη δυνατότητα να πάμε απευθείας από το κείμενο **Α** στο κείμενο **Δ**, το παράδειγμα αυτό δείχνει ότι είναι πιθανό να υπάρχουν πολλά ενδιάμεσα διαφορετικά "μονοπάτια" που συνδέουν δύο στοιχεία σε μία δομή υπερκειμένου. Το υπερκείμενο προσφέρει πολλές διαφορετικές επιλογές στους αναγνώστες, και καθένας από αυτούς αποφασίζει την ώρα της ανάγνωσης ποια από τις επιλογές αυτές θα ακολουθήσει. Αυτό σημαίνει ότι ο συγγραφέας του υπερκειμένου έχει θέσει έναν αριθμό εναλλακτικών τρόπων ανάγνωσης αυτού στους αναγνώστες.

Οπως φαίνεται στο παραπάνω σχήμα, το υπερκείμενο αποτελείται από διασυνδεδεμένα μεταξύ τους κομμάτια κειμένου. Καθένα από τα κομμάτια αυτά, ή καλύτερα μονάδες πληροφορίας, ονομάζεται **κόμβος** (node). Οποιοδήποτε και αν είναι το μέγεθος των κόμβων, καθένας από αυτούς μπορεί να "δείχνει" προς κάποιους άλλους κόμβους, και οι "δείκτες" αυτοί ονομάζονται **σύνδεσμοι** (links). Ενας σύνδεσμος υπερκειμένου διασυνδέει δύο κόμβους και η κατεύθυνσή του είναι από τον έναν κόμβο, που ονομάζεται **κόμβος αναχώρησης** (anchor node), προς τον άλλο, ο οποίος ονομάζεται **κόμβος προορισμού** (destination node). Δεν υπάρχει προκαθορισμένος αριθμός συνδέσμων που διαθέτει ένας κόμβος, αλλά αυτός εξαρτάται από το περιεχόμενο του κάθε κόμβου. Μερικοί κόμβοι, δηλαδή, μπορεί να συσχετίζονται με πολλούς άλλους και γι'αυτό το λόγο θα έχουν αρκετούς συνδέσμους, ενώ άλλοι κόμβοι θα εξυπηρετούν απλά και μόνο ως τελικοί προορισμοί κάποιων άλλων, χωρίς έτσι να διαθέτουν συνδέσμους.

Το σχήμα δείχνει επίσης ότι ολόκληρη η δομή του υπερκειμένου συνιστά ένα δίκτυο (network) κόμβων και συνδέσμων. Η μετακίνηση των αναγνωστών μέσα σε αυτό το είδος δικτύου αποδίδεται συχνά με τους αγγλικούς όρους **browsing** ή **navigating** και όχι με τον όρο "ανάγνωση", με σκοπό να δοθεί έμφαση στο γεγονός ότι οι χρήστες αποφασίζουν μόνοι τους τη σειρά με την οποία θα διαβαστούν οι κόμβοι του υπερκειμένου. Όταν, τώρα, οι χρήστες ακολουθούν τους συνδέσμους μέσα σε αυτό το

δίκτυο υπερκειμένου, αισθάνονται συχνά την ανάγκη να επιστρέψουν σε κάποιον κόμβο που ήδη είχαν επισκεφθεί προηγουμένως.

Τα περισσότερα από τα συστήματα υπερκειμένου παρέχουν αυτή την ευκολία στους χρήστες μέσω μίας διαδικασίας που ονομάζεται **οπισθοδρόμηση** (backtrack facility). Για να γίνει αυτό πιο κατανοητό, ας υποθέσουμε ότι βρισκόμαστε στον κόμβο **Δ** του παραπάνω σχήματος στον οποίο είχαμε φτάσει μέσω της διαδρομής **Α→Β→Ε→Δ**. Μία πρώτη εντολή οπισθοδρόμησης τότε θα μας έφερνε στον κόμβο **Ε**, μία δεύτερη στον κόμβο **Β**, και μία τρίτη στον αρχικό κόμβο **Α**. Εάν όμως, εναλλακτικά, είχαμε πάει απευθείας από τον κόμβο **Α** στον κόμβο **Δ**, τότε η διαδικασία της οπισθοδρόμησης θα μας πήγαινε επίσης απευθείας από τον τελευταίο κόμβο στον κόμβο **Α**. Αυτό το παράδειγμα δείχνει ότι η διαδικασία της οπισθοδρόμησης εξαρτάται τόσο πολύ από την κίνηση και τις επιλογές του μεμονωμένου χρήστη όσο εξαρτιόταν και η σειρά με την οποία αυτός είχε αρχικά επισκεφθεί τους κόμβους.

Η έννοια των υπερμέσων

Από τον παραδοσιακό ορισμό του όρου "υπερκείμενο" απορρέει ότι αυτό αποτελεί ένα σύστημα το οποίο είναι άρρηκτα συνδεδεμένο και συνυφασμένο με την έννοια του απλού κειμένου. Από τη στιγμή όμως που πολλά νεότερα συστήματα προσφέρουν τη δυνατότητα εργασίας με γραφικά και διάφορα άλλα μέσα αναπαράστασης της πληροφορίας, μερικοί άνθρωποι προτιμούν να χρησιμοποιούν τον όρο **υπερμέσα** (hypermedia) για να δώσουν ιδιαίτερη έμφαση στα χαρακτηριστικά και τις δυνατότητες πολυμέσων που διαθέτει το σύστημά τους. Σε κάθε περίπτωση πάντως, το υπερκείμενο αποτελεί μία τεχνική για την υποστήριξη και εφαρμογών πολυμέσων, αφού αυτό στηρίζεται στη διασύνδεση κόμβων που μπορούν να περιέχουν διαφόρων ειδών μορφές πληροφορίας. Ορισμένα από τα τυπικότερα ίσως μέσα για την αναπαράσταση πληροφοριών που χρησιμοποιούνται σε κόμβους υπερμέσων είναι το κείμενο, τα γραφικά, ο ήχος και το video.

Τα γραφικά που χρησιμοποιούνται σε εφαρμογές υπερμέσων μπορεί να είναι είτε ψηφιοποιημένες, δηλαδή "σαρωμένες", φωτογραφίες (digitized or scanned images) είτε αντικειμενοστραφείς εικόνες (object-oriented pictures) κατασκευασμένες από κάποιον αλγόριθμο γραφικών σε υπολογιστή. Τα γραφικά αυτά μπορούν να χρησιμοποιηθούν απλά και μόνο για εικονογράφηση ή μπορούν τα ίδια να διαθέτουν συνδέσμους προς κάποιους άλλους κόμβους πληροφοριών. Για παράδειγμα, τα διάφορα τμήματα σε έναν χάρτη της Πολυτεχνειούπολης Ζωγράφου θα μπορούσαν να αποτελούν συνδέσμους προς κάποιους άλλους κόμβους μίας εφαρμογής υπερμέσων, στους οποίους πιθανότατα θα αναγράφονταν τα ονόματα όλων των μελών ΔΕΠ του κάθε τμήματος.

Διάφορα άλλα μέσα αναπαράστασης της πληροφορίας που χρησιμοποιούνται συχνά ως τύποι δεδομένων σε κόμβους υπερμέσων είναι ο ήχος και οι κινούμενες εικόνες με τη μορφή video ή animation. Ο πιο κοινός τρόπος για να αναπαραστήσουμε ήχο και video ως κόμβους πληροφοριών είναι να χρησιμοποιήσουμε ένα απλό κείμενο ως κόμβο αναχώρησης, το οποίο θα οδηγεί στην κλήση ενός αρχείου ήχου ή video. Με τον τρόπο αυτό, το κείμενο λειτουργεί ως σύνδεσμος προς το παίξιμο ενός μουσικού κομματιού ή ενός video clip. Σε αντίθεση όμως με το κείμενο και τα γραφικά, ο ήχος

και το video παρουσιάζουν μία σημαντική δυσκολία. Αυτές οι δύο μορφές πληροφορίας δεν μπορούν να χρησιμοποιηθούν εύκολα ως κόμβοι αναχώρησης σε ένα δίκτυο υπερμέσων. Για παράδειγμα, ενώ ένα κείμενο για τη ζωή του Mozart μπορεί εύκολα να περιέχει συνδέσμους προς κάποια αρχεία ήχου με αποσπάσματα από τις σημαντικότερες συνθέσεις του, το αντίθετο είναι αρκετά πιο δύσκολο. Παρόλο αυτά όμως, πρόσφατες έρευνες έχουν δώσει κάποιες λύσεις και σε αυτό το πρόβλημα.

Το πρωτόκολλο HTTP

Το πρωτόκολλο HTTP είναι το πιο συνηθισμένο στον ηλεκτρονικό χώρο του World Wide Web. Η ονομασία του προέρχεται από τα αρχικά των αγγλικών λέξεων **HyperText Transfer Protocol** (Πρωτόκολλο Μεταφοράς Υπερκειμένου). Το πρωτόκολλο αυτό χρησιμοποιείται από τη συγκεκριμένη υπηρεσία του δικτύου Internet από το 1990. Το HTTP αποτελεί ένα πρωτόκολλο του επιπέδου εφαρμογών στα δίκτυα υπολογιστών και χρησιμοποιείται κυρίως σε διανεμημένα πληροφορικά συστήματα υπερμέσων. Είναι ένα γενικό, αντικειμενοστραφές πρωτόκολλο που μπορεί να χρησιμοποιηθεί σε ένα πλήθος εφαρμογών, για παράδειγμα σε εξυπηρετητές (servers) και διανεμημένα συστήματα διαχείρισης αντικειμένων. Το βασικότερο και πιο σημαντικό ίσως χαρακτηριστικό του πρωτοκόλλου αυτού είναι ότι επιτρέπει στα διάφορα συστήματα μετάδοσης δεδομένων να υφίστανται ανεξάρτητα από τα δεδομένα που αυτά μεταφέρουν.

Τα πιο διαδεδομένα πρωτόκολλα που χρησιμοποιούνται έως σήμερα στο WWW είναι αυτά της οικογένειας HTTP 1.x. Από την αρχή της δημιουργίας τους και της εφαρμογής τους στο World Wide Web, τα πρωτόκολλα της οικογένειας αυτής περνούν συνεχώς από διάφορα στάδια αναθεώρησης με σκοπό να καλυφθούν τυχόν αδυναμίες των αρχικών εκδόσεων του HTTP. Μία λύση που έχει ήδη προταθεί για την βελτίωση, ή καλύτερα επέκταση, του αρχικού πρωτοκόλλου είναι ένα πρωτόκολλο που φέρει την ονομασία **HTTP-NG**, δηλαδή HyperText Transfer Protocol-New Generation, και το οποίο ενδέχεται να καλύψει όσες αδυναμίες έχουν παρουσιαστεί μέχρι σήμερα στη λειτουργία της αρχική έκδοσης. Η σύνταξη του πρωτοκόλλου HTTP είναι η συνηθισμένη που ακολουθείται και από όλα τα υπόλοιπα URLs. Για παράδειγμα, μία σωστή σύνταξη θα ήταν : **http://www.ntua.gr/pub/multimedia/audio.htm**. Πάντως, ειδικά στο HTTP πρωτόκολλο, ο χρήστης μπορεί να δώσει μόνο τη διεύθυνση του server ή το όνομα ενός directory.

1.1.3 World Wide Web Servers.

Με τον όρο WWW servers δεν εννοούμε μόνο τους υπολογιστές-εξυπηρετητές που είναι υπεύθυνοι για τη διανομή των πληροφοριών μέσα στο World Wide Web, αλλά και το αντίστοιχο λογισμικό (software) που βρίσκεται εγκατεστημένο σε αυτούς και υποβοηθά την παραπάνω λειτουργία της εξυπηρέτησης. Οι WWW servers συναντώνται συχνά και με την ονομασία HTTP servers ακριβώς επειδή πρωταρχικός τους σκοπός είναι η διακίνηση δεδομένων βάσει του πρωτοκόλλου HTTP του Web. Τα περισσότερα από αυτά τα προγράμματα εξυπηρέτησης μπορούν να χρησιμοποιηθούν χωρίς καμία οικονομική επιβάρυνση των χρηστών και διατίθενται δωρεάν από τους κατασκευαστές τους για την πλειοψηφία των λειτουργικών συστημάτων (κυρίως όμως για Unix και Microsoft Windows). Ακολούθως, παρουσιάζονται δύο από τα γνωστότερα προγράμματα εξυπηρέτησης, το NCSA httpd και το Apache, χωρίς αυτό βέβαια να σημαίνει ότι δεν υπάρχουν άλλα αξιόλογα προγράμματα που θα μπορούσαν να συμπεριληφθούν επίσης στην παρουσίαση. Τα προγράμματα αυτά διανέμονται εντελώς δωρεάν, όχι όμως και για εμπορική χρήση.

NCSA httpd

Το πρόγραμμα αυτό αποτελεί έναν WWW server συμβατό με το πρωτόκολλο HTTP και εξυπηρετεί τη διάθεση πληροφοριακών εγγράφων (κυρίως υπερκειμένου) μέσω του World Wide Web. Το πρόγραμμα διατίθεται δωρεάν μόνο για ακαδημαϊκούς, ερευνητικούς ή ενδοεπιχειρησιακούς σκοπούς. Ο συγκεκριμένος αυτός server διακρίνεται για ένα πλήθος ιδιαίτερων χαρακτηριστικών, μερικά από τα οποία είναι: το σχετικά μικρό μέγεθός του, η ταχύτητα με την οποία επεξεργάζεται τον πληροφοριακό όγκο στο Web, το αυξημένο ποσοστό αξιοπιστίας και ασφάλειας που παρέχει και, τέλος, η συμβατότητα και μεταφορά του προγράμματος που του επιτρέπουν να υποστηρίζεται από έναν μεγάλο αριθμό πρωτοκόλλων της οικογένειας HTTP (παλαιότερων και νεότερων εκδόσεων). Ο ενδιαφερόμενος χρήστης μπορεί να αναζητήσει περισσότερες πληροφορίες μέσω της υπηρεσίας του World Wide Web στη διεύθυνση: **http://hoohoo.ncsa.uiuc.edu**.

Apache

Το ερευνητικό πρόγραμμα Apache οργανώθηκε σε μία προσπάθεια για την επίλυση διάφορων προβλημάτων σχετικά με την ανάπτυξη ενός δημόσιας χρήσης και αποδοχής (public domain) HTTP server για υπολογιστικά συστήματα που στηρίζονται στο λειτουργικό σύστημα UNIX. Η αρχική έκδοση του προγράμματος βασίστηκε στον κώδικα και τις ιδέες που συναντώνται και στον NCSA httpd. Από τότε, το συγκεκριμένο πρόγραμμα εξελίχθηκε σε ένα ανώτερο σύστημα, ικανό να συναγωνιστεί στη λειτουργικότητα, την αποτελεσματικότητα και την ταχύτητα οποιονδήποτε άλλο HTTP server για περιβάλλον Unix. Ο Apache αποτελεί σήμερα τον πιο δημοφιλή Web server. Καλύπτει έναν μεγάλο αριθμό μειονεκτημάτων που παρουσιάζονται στον NCSA httpd και, επιπλέον, έχει αποδειχτεί ταχύτερος και πιο αποτελεσματικός από μερικές εκδόσεις του προηγούμενου server. Περισσότερες

πληροφορίες για τον Apache server μπορεί να αναζητήσει κάποιος στη διεύθυνση: **http://www.apache.com.**

1.1.4 Εισαγωγή στο Web Authoring

Τα περιεχόμενα ενός Web document

Ας εξετάσουμε τώρα αναλυτικά τα διάφορα είδη δεδομένων που μπορεί να περιέχει ένα έγγραφο, ή αλλιώς μία σελίδα, στο World Wide Web. Η αρχιτεκτονική του Web είναι μία αρχιτεκτονική πελάτη/εξυπηρετητή (client/server architecture), όπως άλλωστε αναφέρθηκε και παραπάνω. Τα δεδομένα, που αποκαλούνται Web documents, Web pages ή HTML documents, βρίσκονται αποθηκευμένα στους Web servers, δηλαδή τους υπολογιστές του δικτύου που λειτουργούν ως εξυπηρετητές-διανομείς για τους χρήστες. Ο χρήστης που συνδέεται στο Internet για να προσπελάσει αυτά τα έγγραφα είναι ο client, δηλαδή ο πελάτης. Με τον όρο HTML documents εννοούμε αρχεία τα οποία συνήθως αποθηκεύονται με προέκταση.htm ή.ht3 (για την έκδοση 3.0 της γλώσσας HTML) και το περιεχόμενο των οποίων είναι δομημένο σύμφωνα με τις προδιαγραφές της γλώσσας αυτής. Η γλώσσα HTML (HyperText Markup Language),δηλαδή, είναι μία γλώσσα περιγραφής εγγράφων.

Βασικό στοιχείο της φιλοσοφίας του υπερκειμένου είναι η ύπαρξη των λεγόμενων υπερσυνδέσμων. Οπως αναφέρθηκε ήδη και στη σχετική ενότητα, αυτοί δεν είναι τίποτε άλλο από ειδικοί "σύνδεσμοι" που οδηγούν τον χρήστη σε κάποια άλλη πληροφορία, είτε μέσα στο ίδιο έγγραφο είτε συνηθέστερα σε κάποιο άλλο. Με τον τρόπο αυτό, δημιουργείται ένα πλέγμα από ηλεκτρονικά έγγραφα, όπου το καθένα έχει συνδέσμους που οδηγούν το χρήστη σε κάποιο άλλο σημείο. Πιο συγκεκριμένα, ένα έγγραφο υπερκειμένου θα έχει όλες τις λέξεις κανονικά με ένα χρώμα, εκτός από κάποιες οι οποίες θα είναι συνήθως πιο έντονες. Επιλέγοντας μία από αυτές τις τελευταίες λέξεις, ο χρήστης θα οδηγηθεί κάπου αλλού, σε μία σελίδα του WWW με περιεχόμενο που προσδιορίζει η λέξη που αυτός επέλεξε. Μέσω αυτής της διαδικασίας, ο χρήστης οδηγείται κάθε φορά σε διαφορετικές πηγές πληροφοριών, συλλέγοντας έτσι στοιχεία για θέματα που τον ενδιαφέρουν. Αυτή είναι και η σημαντικότερη δυνατότητα που προσφέρει το Web στους χρήστες του. Ενα τεράστιο δίκτυο εύκολα προσπελάσιμων πληροφοριακών εγγράφων.

Ενα HTML έγγραφο μπορεί να περιέχει μία σειρά από δεδομένα, ξεκινώντας από τα πλέον απλά, όπως είναι το κείμενο, και καταλήγοντας σε multimedia δεδομένα, όπως είναι ο ήχος (audio) και το video. Φυσικά, σε ένα τέτοιο έγγραφο μπορούν να ενσωματωθούν γραφικά και φωτογραφίες, πίνακες καθώς και φόρμες εισαγωγής στοιχείων από τους χρήστες. Το ίδιο το έγγραφο, βέβαια, δεν εμπεριέχει πάντοτε όλα αυτά τα δεδομένα. Στην περίπτωση του ήχου και των video clips, για παράδειγμα, απλώς καθορίζονται στο HTML αρχείο τα αντίστοιχα αρχεία ήχου (με πιο συνηθισμένες προεκτάσεις τις.**wav** και.**au**) και video (με προέκταση.**avi** ή.**mov**) τα οποία είναι αποθηκευμένα σε κάποιον υποκατάλογο (directory) του server. Παρόμοια, τα γραφικά και οι διάφορες εικόνες είναι συγκεκριμένα αρχεία με συνήθεις καταλήξεις.**gif** ή.**jpg** τα οποία μέσω διάφορων υπερσυνδέσμων είναι διαθέσιμα στον χρήστη. Το τελικό αποτέλεσμα σε μία σελίδα του Web είναι ο συνδυασμός όλων αυτών των στοιχείων, αλλά το ίδιο το HTML document είναι της τάξης μερικών Kbytes σε μέγεθος. Εξαίρεση αποτελεί το κείμενο, το οποίο όπως είναι φυσικό ενσωματώνεται στο ίδιο το έγγραφο.

Οι διάφοροι υπερσύνδεσμοι παρέχουν ένα πλούσιο ρεπερτόριο επιλογών. Συνήθως αυτοί θα είναι διευθύνσεις κάποιας άλλης σελίδας του Web, αλλά μπορούν επίσης να είναι διευθύνσεις ηλεκτρονικού ταχυδρομείου που θα επιτρέπουν σε έναν χρήστη να στέλνει μηνύματα σε κάποιον άλλο, διευθύνσεις για FTP sites και Gopher sites, σύνδεσμοι για κάποιο αρχείο του ίδιου server που βρίσκεται ο χρήστης, σύνδεσμοι αρχείων ήχου/video κ.λ.π. Δεν είναι βέβαια απαραίτητο ένα κείμενο μόνο να παριστά έναν υπερσύνδεσμο. Μία εικόνα επίσης μπορεί να είναι ένας υπερσύνδεσμος, την οποία όταν ο χρήστης "ενεργοποιήσει" κατάλληλα (συνήθως κάνοντας κλικ με το ποντίκι του σε γραφικό περιβάλλον) να οδηγείται σε μία άλλη διεύθυνση.

Οπως είναι κατανοητό, αυτό το πλούσιο περιβάλλον πολυμέσων, το οποίο παράλληλα είναι και ισχυρά αλληλεπιδραστικό (interactive), αποτελεί ιδανική πλατφόρμα εργασίας για οποιονδήποτε ενδιαφέρεται να συλλέξει στοιχεία για κάποιο θέμα, να "περιπλανηθεί" στον ηλεκτρονικό χώρο του Internet αναζητώντας εικόνες, να πραγματοποιήσει on-line αγορές ή απλά να περάσει ευχάριστα το χρόνο του. Οπως αναφέρεται παρακάτω, η δημιουργία μίας σελίδας του Web είναι ένα σχετικά εύκολο έργο, τουλάχιστον για απλές μορφοποιήσεις σελίδων, και απαιτεί ελάχιστα εργαλεία. Στην ουσία και ένας απλός συντάκτης κειμένου (text editor) αρκεί.

Η γλώσσα επισήμανσης υπερκειμένου HTML

Η δημιουργία προσωπικών σελίδων στο WWW είναι σίγουρα κάτι που ενδιαφέρει τους περισσότερους χρήστες του Internet, τη στιγμή που όλοι μας έχουμε θαυμάσει ορισμένες σελίδες πραγματικά αριστουργήματα. Κλειδί για τη δημιουργία τέτοιων σελίδων είναι μία γλώσσα που φέρει το όνομα **HTML**, από τα αρχικά των αγγλικών λέξεων HyperText Markup Language (Γλώσσα Επισήμανσης Υπερκειμένου), και στην οποία στηρίζεται ολόκληρο το δημιούργημα του World Wide Web. Η γλώσσα HTML αποτελεί υποσύνολο της γλώσσας SGML (Standard Generalized Markup Language) που επινοήθηκε από την IBM προκειμένου να λυθεί το πρόβλημα της μη τυποποιημένης εμφάνισης κειμένων σε διάφορα υπολογιστικά συστήματα. Η SGML αποτελείται από τρία μέρη. Στο πρώτο μέρος, ορίζεται η γραμματοσειρά που θα χρησιμοποιηθεί. Στο δεύτερο, ορίζονται τα **tags** (ετικέτες) που θα χρησιμοποιηθούν για τη μορφοποίηση του συγκεκριμένου εγγράφου. Το τρίτο μέρος περιέχει το κείμενο μαζί με τα tags, που δίνει και το οπτικό αποτέλεσμα. Η ενότητα αυτή αναφέρεται μόνο στο τρίτο μέρος που είναι η γλώσσα HTML.

Η προβολή και η διαχείριση σε επίπεδο χρήστη ενός εγγράφου HTML είναι αρκετά απλή υπόθεση, αφού ο κυριότερος στόχος της γλώσσας είναι η παραγωγή εγγράφων υπερκειμένου εύκολων στο χειρισμό τους. Η γλώσσα αυτή παρουσιάζει απλό κείμενο, εικόνες και προσφέρει ορισμένες μορφοποιήσεις για διακριτική παρουσίαση συγκεκριμένων σημείων (λ.χ. τίτλων). Βασικό στοιχείο της HTML αποτελούν οι σύνδεσμοι (links) που μπορεί αυτή να δημιουργήσει, δηλαδή συγκεκριμένα τμήματα του περιεχομένου της σελίδας τα οποία όταν επιλεγούν από τον χρήστη οδηγούν σε κάποιο άλλο σημείο του εγγράφου, ή ακόμη και σε ένα άλλο έγγραφο που μπορεί να βρίσκεται σε κάποιον άλλο υπολογιστή του Internet. Με αυτά τα βασικά δομικά στοιχεία και ορισμένα άλλα δευτερεύοντα υλοποιούνται όλες οι σελίδες του World Wide Web.

Σε επίπεδο προγραμματισμού, τα HTML αρχεία είναι απλά ASCII αρχεία κειμένου, τα οποία περιέχουν το κείμενο που εμφανίζεται καθώς και τις ειδικές εντολές της γλώσσας για τις συγκεκριμένες λειτουργίες. Για τα γραφικά υπάρχουν αναφορές σε αρχεία γραφικών τα οποία βρίσκονται εκτός του HTML κειμένου. Συνεπώς, η κατασκευή των HTML αρχείων, και κατ' επέκταση των σελίδων του WWW, μπορεί να γίνει με έναν απλό συντάκτη κειμένου. Αν και η γλώσσα είναι αρκετά απλή, έχουν κατά καιρούς αναπτυχθεί αρκετά βοηθητικά προγράμματα, κυρίως γραφικά, τα οποία υποβοηθούν την κατασκευή των HTML αρχείων. Τα προγράμματα αυτά διευκολύνουν σε μεγάλο βαθμό την όλη εργασία, αφού ο χρήστης δεν χρειάζεται να θυμάται τις χαρακτηριστικές εντολές και δηλώσεις της γλώσσας.

Η γλώσσα HTML, τη στιγμή που γράφεται η αναφορά αυτή, βρίσκεται ήδη στην τρίτη της έκδοση. Στην πρώτη έκδοση υπάρχει η δυνατότητα απεικόνισης στατικής και κινούμενης εικόνας, καθώς και η δυνατότητα ήχου. Στη δεύτερη έκδοση προστέθηκε η δυνατότητα εισαγωγής στοιχείων από τον χρήστη. Στην τελευταία τρίτη έκδοση έχει προστεθεί η δυνατότητα απεικόνισης πινάκων, μαθηματικών συμβόλων κ.ά. Σκοπός της ενότητας αυτής είναι να αποκτήσει ο ενδιαφερόμενος αναγνώστης ορισμένες γνώσεις γύρω από τα βασικά χαρακτηριστικά και τις δυνατότητες που προσφέρει η γλώσσα για την ανάπτυξη σελίδων στο Web. Παρακάτω αναφέρονται μερικά από τα χαρακτηριστικά αυτά:

Γενικοί κανόνες σύνταξης

Οπως προαναφέρθηκε, η γλώσσα HTML αποτελείται από κείμενο και **tags** (ετικέτες) τα οποία καθορίζουν την εμφάνιση και τις ιδιότητες του κειμένου. Το tag είναι ένα είδος εντολής που προσδίδει κάποια ιδιαίτερα χαρακτηριστικά στο κείμενο, για παράδειγμα επιλογή μεγέθους χαρακτήρων. Σχεδόν όλα τα tags εσωκλείονται σε "<" και ">" και διακρίνονται σε **απλά** και **διπλά**. Ένα παράδειγμα απλού tag είναι το
 που δηλώνει αλλαγή γραμμής στο κείμενο. Στη δεύτερη περίπτωση, τα διπλά tags σημαδεύουν την αρχή και το τέλος του κειμένου που θα επηρεάσουν, για παράδειγμα το <B> Hello World </B> που εμφανίζει τη φράση "Hello World" με έντονα γράμματα. Όπως φαίνεται και από το παράδειγμα, το δεύτερο tag που τερματίζει την ισχύ του πρώτου εμπεριέχει και τον χαρακτήρα "/", πράγμα που ισχύει γενικά με όλα τα διπλά tags. Ειδική περίπτωση απλού tag αποτελεί αυτό των σχολίων που ορίζεται ως εξής: <!- -Σχόλιο- ->. Τέλος, για να τυπωθούν διάφοροι ειδικοί χαρακτήρες, η γλώσσα HTML διαθέτει το tag χαρακτήρων (character entity). Η γραφή του tag αυτού αρχίζει με τον χαρακτήρα "&", συνεχίζει με το κωδικοποιημένο όνομα του ειδικού χαρακτήρα και καταλήγει με τον χαρακτήρα ";".

Στη συνέχεια παραθέτουμε ένα απλό παράδειγμα σε γλώσσα HTML για να φανούν καλύτερα οι κανόνες σύνταξης που πρέπει να ακολουθούνται. Το παράδειγμα αυτό μπορεί να γραφτεί, όπως άλλωστε και κάθε κομμάτι κώδικα της γλώσσας αυτής, σε έναν οποιοδήποτε συντάκτη (editor) :

<HTML>

<HEAD>

<TITLE> This is my first page </TITLE>

</HEAD>

<BODY>

<H1> Hello World! </H1>

</BODY>

</HTML>

Σχ. Παράδειγμα προγράμματος σε γλώσσα HTML

Από το παραπάνω παράδειγμα φαίνεται ότι κάθε κώδικας στη γλώσσα αυτή αρχίζει και τελειώνει με το διπλό tag **<HTML>**. Μεταξύ αυτού του διπλού tag γράφεται όλο το HTML κείμενο. Όπως κάθε HTML κείμενο, έτσι και το παραπάνω, χωρίζεται σε δύο βασικά τμήματα. Το τμήμα της επικεφαλίδας και το κυρίως σώμα της σελίδας. Η επικεφαλίδα προσδιορίζεται από το διπλό tag **<HEAD>** και παρέχει πληροφορίες για το κείμενο που ακολουθεί. Το πιο συνηθισμένο περιεχόμενό της είναι ο τίτλος της Web σελίδας, ο οποίος ορίζεται από το διπλό tag **<TITLE>**. Το κυρίως σώμα της σελίδας ακολουθεί αμέσως μετά την επικεφαλίδα και περικλείεται από το διπλό tag **<BODY>**. Σε αυτήν την περιοχή υπάρχουν όλα τα συστατικά στοιχεία της σελίδας: μορφοποιημένο κείμενο, εικόνες, ήχοι, σύνδεσμοι, φόρμες εισαγωγής στοιχείων, πίνακες κ.ά. Τα tags μπορούν να γράφονται είτε με πεζά είτε με κεφαλαία γράμματα. Συνήθως, όμως, αυτά γράφονται με κεφαλαίους χαρακτήρες έτσι ώστε ο κώδικας να είναι ευανάγνωστος. Τυχόν κενές γραμμές, καθώς και τα επιπλέον κενά μεταξύ λέξεων, δεν λαμβάνονται υπόψη.

Μορφοποίηση και οργάνωση του κειμένου

Η γλώσσα HTML διαθέτει αρκετά "εργαλεία" για την καλύτερη δυνατή μορφοποίηση και οργάνωση του κειμένου. Δυστυχώς, υπάρχουν και αρκετοί περιορισμοί που όμως τείνουν να μειωθούν με τις επερχόμενες εκδόσεις της γλώσσας. Υπάρχουν δύο διαφορετικών ειδών tags μορφοποίησης του κειμένου: αυτά που δίνουν συγκεκριμένες ιδιότητες στους χαρακτήρες (π.χ. μέγεθος, γραμματοσειρά κ.λ.π.) και αυτά που καθορίζουν την οργάνωση και τη δομή του κειμένου (π.χ. παράγραφοι, ταξινόμηση, λίστες κ.λ.π.).

Η γλώσσα HTML προσφέρει έναν σχετικά περιορισμένο αριθμό γραμματοσειρών. Ο χρήστης, όμως, έχει τη δυνατότητα να συνδυάσει δύο ή τρεις γραμματοσειρές μαζί έτσι ώστε να επιτύχει το επιθυμητό αποτέλεσμα. Για να καθορίσει το μέγεθος των χαρακτήρων, ο χρήστης μπορεί να επιλέξει ένα από τα έξι διαφορετικά μεγέθη που διαθέτει η γλώσσα αυτή. Ανάλογα με το μήκος τους, οι χαρακτήρες ταξινομούνται σε **courier** και σε **proportional**. Στη γραμματοσειρά courier όλοι οι χαρακτήρες

καταλαμβάνουν τον ίδιο χώρο στην οθόνη, ενώ αντίθετα, οι χαρακτήρες της γραμματοσειράς proportional έχουν μεταβλητό μήκος ανάλογα με την κατασκευή τους. Για παράδειγμα, ο χαρακτήρας "ι" είναι εκ κατασκευής μικρότερος από τον χαρακτήρα "ω" και έτσι καταλαμβάνει λιγότερο χώρο. Για τον λόγο αυτό, οι χαρακτήρες courier χρησιμοποιούνται σε κείμενα που περιέχουν πίνακες για να μην αλλοιωθεί η στοίχισή τους και γενικά σε κείμενα που είναι από πριν μορφοποιημένα.

Κατάλογοι

Στη συγγραφή μίας σελίδας του Web υπάρχει πολλές φορές η ανάγκη δημιουργίας καταλόγων στοιχείων (lists). Με τη βοήθεια των καταλόγων αυτών, ο αναγνώστης διευκολύνεται περισσότερο στην κατανόηση και την ταχύτερη απομνημόνευση των στοιχείων. Η γλώσσα HTML διαθέτει πέντε διαφορετικά είδη καταλόγων. Ανάλογα με τη συχνότητα εμφάνισης και τη χρήση τους σε ένα HTML έγγραφο μπορούν αυτοί να διακριθούν σε :

1. ***Αριθμητικούς καταλόγους*** *(Ordered or Numbered Lists)* : Είναι οι κατάλογοι που έχουν αριθμημένες εγγραφές κατά αύξουσα σειρά. Η αρίθμηση αυτή μπορεί να είναι είτε μία απλή καταμέτρηση είτε μία ιεραρχική ταξινόμηση των στοιχείων του καταλόγου.
2. ***Μη αριθμητικούς καταλόγους*** *(Unordered Lists)* : Οι κατάλογοι αυτοί περιέχουν μία απλή αναφορά στα στοιχεία τους, χωρίς να παίζει κανένα ιδιαίτερο ρόλο η ταξινόμησή τους.
3. ***Καταλόγους επιλογών*** *(Menu Lists)* : Αυτός ο τύπος καταλόγου είναι ίσως ο λιγότερο δημοφιλής. Εχει ελάχιστη διαφορά από τους μη αριθμητικούς καταλόγους και μάλιστα τείνει να εξαλειφθεί και από τις νεότερες εκδόσεις της γλώσσας. Και στην περίπτωση αυτού του καταλόγου δεν παίζει κανένα ιδιαίτερο ρόλο η ταξινόμηση των στοιχείων που εμφανίζονται. Είναι δηλαδή μία απλή αναφορά.
4. ***Καταλόγους αρχείων*** *(Directory Lists)* : Αρκετές φορές ο χρήστης έχει την ανάγκη να παρουσιάσει σε μία σελίδα του κάποιον κατάλογο αρχείων. Την ανάγκη αυτή έρχεται να καλύψει ο κατάλογος αρχείων.
5. ***Ερμηνευτικούς καταλόγους*** *(Definition or Glossary Lists)* : Μία πολύ ενδιαφέρουσα και χρήσιμη κατηγορία είναι αυτή των ερμηνευτικών καταλόγων. Οπως μπορεί κάποιος εύκολα να αντιληφθεί από την ονομασία, πρόκειται για καταλόγους όρων και των ερμηνειών τους, κάτι ανάλογο δηλαδή με ένα λεξικό.
6. ***Συνδυασμούς καταλόγων*** : Πολλές φορές είναι απαραίτητος ο συνδυασμός διαφόρων από τους παραπάνω καταλόγους έτσι ώστε να επιτύχει ο χρήστης το επιθυμητό αποτέλεσμα. Η γλώσσα HTML παρέχει αυτήν τη δυνατότητα.

Εικόνα

Μέχρι αυτό το σημείο, παρουσιάστηκαν μερικές από τις δυνατότητες της γλώσσας που είχαν σχέση με την παρουσίαση ενός απλού κειμένου. Όπως αναφέρθηκε και παραπάνω όμως, η υπηρεσία του World Wide Web γνωρίζει τρομερή επιτυχία και παγκόσμια αποδοχή λόγω της απλότητας, της χρησιμότητας και κυρίως λόγω της εντυπωσιακής εμφάνισης των κειμένων της. Το τελευταίο οφείλεται στις δυνατότητες **πολυμέσων** (multimedia) που προσφέρει η γλώσσα HTML σε συνεργασία με τα διάφορα προγράμματα ανάγνωσης σελίδων του Web, τους επονομαζόμενους **Web browsers**. Η πιο απλή μορφή αναπαράστασης της πληροφορίας, μετά το κείμενο πάντοτε, είναι η εικόνα και γενικότερα τα γραφικά. Η γλώσσα HTML παρέχει στους χρήστες τη δυνατότητα ενσωμάτωσης τέτοιων μορφών πληροφορίας στα κείμενά τους.

Για να αποθηκευτεί μία εικόνα στον υπολογιστή, υπάρχουν διάφοροι τρόποι κωδικοποίησης που ονομάζονται **formats**. Τα πιο δημοφιλή formats εικόνας είναι τα: **GIF**, **JPG**, **BMP**, **PCX** και **TIF**. Δυστυχώς, δεν υποστηρίζονται απευθείας όλα τα formats εικόνων από τους Web browsers. Πάντως, το σύνολο των browsers υποστηρίζει εκ κατασκευής τουλάχιστον τις εικόνες GIF. Αυτό απλά σημαίνει ότι μία τέτοια εικόνα μπορεί να αποτελέσει μέρος του HTML κειμένου και γι'αυτό ονομάζεται και **εσωτερική** εικόνα. Για τα υπόλοιπα formats, ο χρήστης μπορεί να ορίσει στον browser του τα αντίστοιχα προγράμματα εμφάνισης που ονομάζονται viewers. Οι εικόνες που ανήκουν σε formats που δεν υποστηρίζονται απευθείας από τον browser του χρήστη ονομάζονται **εξωτερικές**.

Σύνδεσμοι

Όπως αναφέρθηκε και προηγουμένως, HTML ή HyperText Markup Language σημαίνει Γλώσσα Επισήμανσης Υπερκειμένου. Το υπερκείμενο (hypertext) είναι ένα κείμενο με λέξεις-κλειδιά που ονομάζονται σύνδεσμοι (links) και μπορούν να οδηγήσουν τον χρήστη σε ένα άλλο επεξηγηματικό κείμενο του συνδέσμου. Η γλώσσα HTML προσφέρει ακόμη μία δυνατότητα. Οι σύνδεσμοι δεν περιορίζονται σε απλές λέξεις ή φράσεις αλλά μπορούν να είναι ακόμη και εικόνες, οπότε στην περίπτωση αυτή έχουμε τις επονομαζόμενες **εικόνες-συνδέσμους**. Αντίστοιχα, η σύνδεση δεν είναι απαραίτητο να γίνεται πάνω σε κείμενο αλλά και σε κάποια εικόνα, ήχο ή ακόμη και video. Το σημαντικότερο όμως πλεονέκτημα που προσφέρουν οι σύνδεσμοι σε έναν χρήστη είναι η δυνατότητα ενσωμάτωσης διάφορων URLs στο HTML κείμενό του, που του εξασφαλίζουν σύνδεση με άλλους servers στον υπόλοιπο κόσμο μέσω διάφορων υπηρεσιών όπως FTP, Telnet, Mail, Gopher, Usenet και φυσικά WWW. Οι σύνδεσμοι, τέλος, μπορούν να διακριθούν σε δύο κατηγορίες: σε συνδέσμους που συνδέουν τον χρήστη σε εξωτερικά αρχεία (κείμενα, εικόνες, ήχους, video) και σε συνδέσμους που τον οδηγούν σε κάποιο συγκεκριμένο σημείο ενός κειμένου HTML.

Image Maps

Η τεχνική του Image Map είναι μία πολύ ενδιαφέρουσα παραλλαγή της εικόνας-συνδέσμου. Για την ακρίβεια, πρόκειται για μία εικόνα όπου διάφορες περιοχές της μπορούν να είναι σύνδεσμοι. Ένα απλό παράδειγμα Image Map είναι ένας χάρτης της Ευρώπης, όπου οι συνοριακές γραμμές των κρατών είναι καθαρά οροθετημένες για διευκόλυνση του χρήστη. Έτσι, εάν ο χρήστης επιλέξει, π.χ. με το ποντίκι του, ένα κράτος θα μπορέσει αυτός να πάρει πληροφορίες για τον πληθυσμό, το θρήσκευμα, την έκταση της χώρας κ.ά. Για την κατασκευή ενός Image Map πρέπει ο χρήστης να ακολουθήσει μία αρκετά σύνθετη, αλλά όχι και δύσκολη, διαδικασία.

Οι εικόνες που χρησιμοποιούνται για την κατασκευή τέτοιων Image Maps είναι σχεδόν αποκλειστικά σε GIF format. Οι περιοχές-σύνδεσμοι της εικόνας καθορίζονται από τον συγγραφέα της σελίδας σε ένα αρχείο που θα βρίσκεται στον Web server του. Το αρχείο αυτό, που ονομάζεται Map file, περιέχει τις συντεταγμένες των περιοχών-συνδέσμων καθώς και τις διευθύνσεις στις οποίες θα οδηγηθεί ένας χρήστης επιλέγοντας αυτές. Η επεξεργασία του αρχείου αυτού γίνεται από ένα πρόγραμμα που επίσης βρίσκεται στον Web server.

Ήχος και κίνηση

Ενας σύνδεσμος σε κάποιο HTML έγγραφο μπορεί να οδηγήσει τον χρήστη όχι μόνο σε κάποιο άλλο κείμενο, του ίδιου ή άλλου απομακρυσμένου Web server, αλλά και σε οποιουδήποτε τύπου εξωτερικά αρχεία. Τέτοια αρχεία είναι, για παράδειγμα, αρχεία εικόνων, ήχων, video κ.ά. Όπως συμβαίνει με τις εικόνες έτσι και με τον ήχο ή το video, υπάρχουν διάφοροι τρόποι αποθήκευσής τους. Οι τρόποι αυτοί ονομάζονται και σε αυτή την περίπτωση formats, μερικά γνωστά από τα οποία είναι τα ακόλουθα: **WAV, AU, AIF** και **MID** για τον ήχο και **AVI, MOV** και **MPG** για το video. Η δυνατότητα ενσωμάτωσης παντός είδους αρχείων που προσφέρει η γλώσσα HTML δίνει στις σελίδες του Web ιδιότητες πολυμέσων. Ετσι, με λίγη φαντασία και αρκετή δουλειά, κάθε χρήστης έχει τη δυνατότητα να κατασκευάσει μία on-line multimedia εφαρμογή της αρεσκείας του.

Αμφίδρομη επικοινωνία

Μέχρι αυτό το σημείο της αναφοράς, έχουν αναφερθεί οι περισσότερες δυνατότητες της γλώσσας HTML που έχουν σχέση με την παρουσίαση δεδομένων στην οθόνη. Κατ'αυτόν τον τρόπο, η επικοινωνία που έχει ο αναγνώστης με μία σελίδα του Web είναι μονόδρομη και αυτό γιατί δέχεται μόνο στοιχεία. Αν όμως ο αναγνώστης είχε τη δυνατότητα να δίνει στοιχεία, όπως για παράδειγμα το όνομά του, τη διεύθυνσή του κ.ά., η επικοινωνία του με τη σελίδα θα ήταν αμφίδρομη. Για να γίνει πιο κατανοητή η έννοια της αμφίδρομης επικοινωνίας, αναφέρουμε το ακόλουθο χαρακτηριστικό παράδειγμα.

Ας υποθέσουμε ότι ένας χρήστης βρίσκεται στη σελίδα του Web μίας εταιρείας που πουλάει δίσκους μουσικής. Εχει στη διάθεσή του έναν τεράστιο κατάλογο με τη δισκογραφία εκατοντάδων συγκροτημάτων μαζί με τις φωτογραφίες των εξώφυλλων των δίσκων, τους τίτλους αλλά και τα λόγια των τραγουδιών. Βέβαια, η σελίδα είναι εξοπλισμένη με αρκετά εργαλεία αναζήτησης όπως σημεία, τα οποία μπορεί αυτός να επιλέξει με ταξινόμηση καταλόγων κατά αλφαβητική ή χρονολογική σειρά, εμφάνιση νέων παραγωγών αρχίζοντας από χρονική στιγμή που ορίζει ο χρήστης ή ακόμη και αναζήτηση με λέξεις κλειδιά που και πάλι αυτός ορίζει.

Αποφασίζει να αναζητήσει τη δισκογραφία του συγκροτήματος "U2" τα τελευταία τρία χρόνια. Ετσι, λοιπόν, αυτός διαλέγει, π.χ. με τη βοήθεια του ποντικιού, την επιλογή "ΟΧΙ ΑΡΓΟΤΕΡΑ ΑΠΟ" και πληκτρολογεί τον αριθμό "3" στο ανάλογο κουτάκι (πεδίο). Στη λίστα με τις επιλογές "ΗΜΕΡΕΣ", "ΜΗΝΕΣ", "ΧΡΟΝΙΑ" ο χρήστης διαλέγει την τελευταία επιλογή. Το σύστημα απαντά ότι βρήκε δύο δίσκους με τις προδιαγραφές που ο χρήστης όρισε. Στη συνέχεια, αυτός βλέπει τις φωτογραφίες των εξώφυλλων των δίσκων και δίπλα τους τίτλους των τραγουδιών που περιέχονται στον καθένα. Πατώντας κάποιο εικονικό πλήκτρο με το ποντίκι του, ο ενδιαφερόμενος χρήστης ζητά να αγοράσει τον πρώτο από τους δύο δίσκους. Το σύστημα εμφανίζει τότε μία φόρμα καταχώρησης στοιχείων για την εισαγωγή του ονοματεπώνυμου του υποψήφιου αγοραστή, τον τύπο, τον αριθμό και την ημερομηνία λήξης της πιστωτικής του κάρτας. Τέλος, ζητείται από τον χρήστη να επιβεβαιώσει την παραγγελία επιλέγοντας "ΝΑΙ" ή "ΟΧΙ" αντίστοιχα. Αυτό αποτελεί ένα από τα πιο χαρακτηριστικά παραδείγματα αμφίδρομης επικοινωνίας, που βασίζεται στις δυνατότητες που παρέχει η γλώσσα HTML στους χρήστες για την υλοποίηση και υποστήριξη τέτοιου είδους επικοινωνίας μέσω εγγράφων του World Wide Web.

CGI Scripts

Οπως έχει φανεί μέχρι τώρα, η γλώσσα HTML έχει τη δυνατότητα να παρουσιάζει δεδομένα με έναν πραγματικά εντυπωσιακό τρόπο, αλλά δεν μπορεί να τα επεξεργαστεί. Το κενό αυτό της γλώσσας καλύπτεται από άλλα βοηθητικά προγράμματα, τα οποία μπορούν να είναι γραμμένα σε μία άλλη γλώσσα, όπως **C**, **PERL** ή ακόμη και **Unix Scripts** (κάτι ανάλογο των batch files του DOS). Αυτά τα βοηθητικά προγράμματα ονομάζονται **CGI** (Common Gateway Interface) και επεξεργάζονται δεδομένα που παρέχονται από τη σελίδα του Web και μετά επιστρέφουν κάποιο αποτέλεσμα είτε σε HTML είτε σε οποιαδήποτε άλλη μορφή εμείς ορίσουμε.

Τα CGI Scripts βρίσκονται πάντοτε στη μεριά του Web server και για να εκτελεστούν θα πρέπει να το επιτρέπει ο διαχειριστής του συστήματος (Web master). Μία πολύ συνηθισμένη χρήση CGI Scripts είναι η αναζήτηση κάποιας λέξης σε μία βάση δεδομένων ή η εκτέλεση κάποιας εντολής στο λειτουργικό σύστημα. Τέλος, τα CGI Scripts χρησιμοποιούνται για την επεξεργασία και την αξιοποίηση των δεδομένων που εισάγουν οι χρήστες σε μία εφαρμογή HTML.

Πίνακες

Πολλές φορές, κατά τη συγγραφή μίας σελίδας για το World Wide Web, παρουσιάζεται η ανάγκη οργάνωσης διάφορων στοιχείων σε πίνακες. Οι νεότερες εκδόσεις της γλώσσας HTML παρέχουν στους χρήστες αυτή τη δυνατότητα. Οι πίνακες αποτελούν πλέον ένα από τα ισχυρότερα χαρακτηριστικά της γλώσσας αυτής. Ενας πίνακας στοιχείων αποτελείται από γραμμές και στήλες. Το σημείο τομής μίας γραμμής και μίας στήλης ονομάζεται **κυψέλη** (cell). Μέσα στις κυψέλες τοποθετούνται τα στοιχεία του πίνακα, είτε αυτά είναι δεδομένα είτε τίτλοι. Δεδομένα του πίνακα μπορούν βέβαια να είναι και άλλες κατηγορίες στοιχείων, όπως κατάλογοι, εικόνες, φόρμες ή ακόμη και υποπίνακες.

Η γλώσσα HTML εξελίσσεται διαρκώς και πολύ σύντομα μπορεί να μην ισχύουν αρκετά από τα παραπάνω. Άλλωστε, το κυριότερο ίσως πρόβλημα που αντιμετωπίζει ο προγραμματιστής είναι ότι οι προδιαγραφές της γλώσσας περνούν συνεχώς από διάφορα στάδια συνεχούς αναθεώρησης με αποτέλεσμα να προστίθενται ολοένα και περισσότερα νέα χαρακτηριστικά. Είναι όμως βέβαιο πως η HTML έχει πολλά ακόμη να προσφέρει. Οπως δείχνουν τα πράγματα, οι δυνατότητές της θα είναι σύντομα τέτοιες ώστε τα έντυπα (περιοδικά, εφημερίδες) θα μοιάζουν ξεπερασμένα.

Ηδη ένα "παρακλάδι" της HTML που προσφέρει δυνατότητες εικονικής πραγματικότητας (virtual reality) εδώ και καιρό την εμφάνισή του. Η γλώσσα αυτή ονομάζεται **VRML**, από τα αρχικά των αγγλικών λέξεων Virtual Reality Modeling Language. Στη γλώσσα VRML δεν υπάρχει κείμενο με λέξεις-συνδέσμους αλλά εικόνα με **αντικείμενα-συνδέσμους**. Ο χρήστης έχει τη δυνατότητα να κινηθεί στον χώρο και να επιλέξει το αντικείμενο που θα επεξεργαστεί. Η ταχύτητα με την οποία εξελίσσεται αυτή η τεχνολογία είναι ασύλληπτη και είναι δύσκολο, έως αδύνατο, για κάποιον να προσδιορίσει την νέα "τάξη πραγμάτων" που θα ακολουθήσει στο χώρο του World Wide Web.

Τι είναι το Web Authoring

Το World Wide Web αποτελεί σήμερα ένα ιδανικό μέσο προώθησης των προϊόντων διάφορων εμπορικών επιχειρήσεων αλλά και προβολής των μεμονωμένων χρηστών. Από τη στιγμή, λοιπόν, που μία επιχείρηση ή ένας χρήστης έχουν αποφασίσει ότι ο ηλεκτρονικός χώρος του Web τους ενδιαφέρει ως ένα εργαλείο προβολής και διαφήμισης, θα πρέπει αυτοί να καταλήξουν στο υλικό που θα ήθελαν να παρουσιάσουν καθώς και στον τρόπο με τον οποίο θα αναπτύξουν τις σελίδες τους. Ολο αυτό το project που έχει να κάνει με την οργάνωση και παρουσίαση του υλικού συνηθίζεται να αποκαλείται **Web authoring** και εμπεριέχει ως τμήμα του και την κωδικοποίηση των σελίδων που συνήθως ονομάζεται **HTML programming**.

Οι ανάγκες, βέβαια, μίας εμπορικής επιχείρησης είναι τελείως διαφορετικές από αυτές ενός μεμονωμένου χρήστη, και συνεπώς εντελώς διαφορετικές είναι και οι απαιτήσεις τους σε ο, τι αφορά την ανάπτυξη μίας σελίδας στο World Wide Web. Ακολούθως αναφέρονται οι διαφορές στη σχεδίαση μίας WWW σελίδας για μία εταιρεία και έναν απλό χρήστη. Η ανάλυση που ακολουθεί έχει ως άξονα μία μεσαίου

μεγέθους επιχείρηση, αλλά η φιλοσοφία εργασίας είναι παρόμοια και για μικρότερες ή μεγαλύτερες εταιρείες. Τα μεγέθη που αναφέρονται απλά κλιμακώνονται ανάλογα :

Η εμπορική εταιρεία

Οσο τέλεια και αν είναι η υλικοτεχνική υποδομή της επιχείρησης, όσο πλούσιοι και αν είναι οι διαθέσιμοι πόροι της, ο σημαντικότερος παράγοντας επιτυχίας εδώ είναι το ανθρώπινο δυναμικό που αυτή διαθέτει και η συνεργασία που επιτυγχάνεται μεταξύ των διάφορων ομάδων εργασίας της (workgroups). Μία τέτοια ομάδα εργασίας, που συνήθως αποκαλείται Web team, αναλαμβάνει το συνολικό έργο της παρουσίασης της επιχείρησης στο χώρο του World Wide Web και απαρτίζεται κυρίως από τα ακόλουθα μέλη:

Τον **Webmaster** ο οποίος είναι ένα στέλεχος με τεχνικές γνώσεις αναφορικά με δίκτυα υπολογιστών και πληροφοριακά συστήματα. Αυτός θα αναλάβει το τεχνικό μέρος του έργου, δηλαδή θα εγκαταστήσει τον υπολογιστή που προορίζεται για Web server, καθώς και το απαιτούμενο software, ενώ θα αναλάβει και τις διάφορες καλωδιώσεις/συνδέσεις που είναι απαραίτητες. Αυτό, βέβαια, στην περίπτωση που η επιχείρηση επιθυμεί να αναπτύξει τον δικό της Web server και όχι να καταβάλλει κάποια μηνιαία συνδρομή σε κάποια από τις εταιρείες που παρέχουν υπηρεσίες πρόσβασης στο Internet (Internet providers) για να προβάλλει τη σελίδα της στο Web.

1. Εναν ή περισσότερους **Docmasters** ή **Web authors**, όπως αποκαλούνται. Είναι οι άνθρωποι που θα αναλάβουν το σχεδιασμό των σελίδων, σε ανώτερο επίπεδο, κατανέμοντας το υλικό και αναλαμβάνοντας τη διαχείριση του συνολικού έργου (project management). Αυτοί θα έχουν την επιμέλεια όλων των θεμάτων δίνοντας τις κατευθυντήριες γραμμές. Είναι απαραίτητο γι' αυτούς να γνωρίζουν σε βάθος τη φιλοσοφία του Web και να έχουν σχετική εμπειρία από πολλά sites του εξωτερικού που έχουν ήδη αξιοποιήσει με επιτυχία τον Web. Πρέπει να είναι διαρκώς ενήμεροι για τις εξελίξεις σε διεθνές επίπεδο και να φροντίζουν να επισκέπτονται συχνά άλλους κόμβους με πρωτότυπο και λειτουργικό σχεδιασμό.
2. Τους προγραμματιστές σε γλώσσα HTML ή **HTML programmers** που θα αναλάβουν την κωδικοποίηση των σελίδων. Ο συνολικός αριθμός τους εξαρτάται από το πλήθος των πληροφοριών που θα εισαχθούν στις σελίδες του Web.
3. Εναν **γραφίστα/ειδικό DTP** που θα παρέχει συμβουλές για την καλύτερη δυνατή εμφάνιση των σελίδων και θα επιμεληθεί την αισθητική τους πλευρά, σχεδιάζοντας ενδεχομένως και κάποια γραφικά μέρη. Η παρουσία του στην ομάδα εξαρτάται σε μεγάλο βαθμό από την βαρύτητα που δίνει η επιχείρηση στο συνολικό έργο, αλλά σίγουρα προσφέρει και μία διαφορετική οπτική γωνία προσέγγισης που ασφαλώς είναι χρήσιμη.

Ο μεμονωμένος χρήστης

Οι απαιτήσεις φυσικά ενός απλού χρήστη, σε ότι αφορά την ανάπτυξη μίας σελίδας στο World Wide Web, δεν μπορούν να είναι τόσο μεγάλες όσο αυτές μίας εμπορικής επιχείρησης. Κατά συνέπεια, για τη σχεδίαση μίας τέτοιας σελίδας δεν απαιτείται μία ολόκληρη και οργανωμένη ομάδα εργασίας, αλλά πρέπει ο ενδιαφερόμενος χρήστης να διαθέτει έναν λογαριασμό πρόσβασης στο Web και καλή γνώση της γλώσσας HTML. Έτσι, κάθε χρήστης του δικτύου Internet έχει σήμερα τη δυνατότητα να φτιάξει μία σελίδα στο Web και τα διαθέσιμα μέσα προσφέρονται για δημιουργικές σχεδιάσεις.

Υπάρχουν όμως ορισμένα σημεία που αξίζει κάθε χρήστης να προσέξει, εάν επιθυμεί η σελίδα του να μεταφέρει στους υπόλοιπους το μήνυμά του με τρόπο εύληπτο, καλαίσθητο, οργανωμένο και προσιτό. Το Web authoring, όπως αποκαλείται ελεύθερα, είναι μία υπέροχη τέχνη. Αυτό, όμως, δεν σημαίνει πως ο χρήστης δεν μπορεί να δομήσει το υλικό του με κάποιες στοιχειώδεις αρχές και να επιτύχει έτσι τη βέλτιστη απόδοσή του. Ακολουθώντας κάποιες κατευθυντήριες γραμμές, ο απλός χρήστης μπορεί κάλλιστα να έχει μία επιτυχημένη παρουσία στον αχανή και μαγικό κόσμο του Web.

Η πρώτη δουλειά του χρήστη για μία καλή παρουσία στο World Wide Web είναι να καθορίσει επακριβώς το υλικό που θα εισαχθεί. Αν οι σελίδες που θα δημιουργήσει αυτός έχουν απλά προσωπικό περιεχόμενο, τότε δεν θα τον απασχολήσει ιδιαίτερα αυτό το ζήτημα και το πιθανότερο είναι ότι θα κινηθεί ελεύθερα. Εάν, όμως, οι σελίδες του πρόκειται να χρησιμοποιηθούν για επαγγελματικούς σκοπούς και επιδιώκει να έχει το μέγιστο όφελος, τότε το πρώτο αυτό στάδιο είναι εξαιρετικά σημαντικό.

Εφόσον, τώρα, ο χρήστης έχει στα χέρια του το περιεχόμενο που θέλει να εισάγει σε απτή μορφή, μπορεί να προχωρήσει στο δεύτερο στάδιο της ανάπτυξης που είναι η οργάνωση και σωστή δόμησή του. Η συνηθέστερη και πλέον επιτυχημένη τακτική είναι να ακολουθήσει ο χρήστης μία από *πάνω-προς τα κάτω* προσέγγιση (top-down method). Με απλά λόγια, ο χρήστης ξεκινάει από τα γενικά και προχωρεί προς τα ειδικά. Αυτό παράλληλα σημαίνει ότι οι σελίδες δεν πρέπει να αναπτύσσονται σε εύρος αλλά σε βάθος. Στα περισσότερα sites του εξωτερικού η τακτική αυτή έχει αποδειχτεί ιδιαίτερα επιτυχής και έχει αποδώσει σημαντικούς καρπούς. Αυτή επιτρέπει την εύκολη ενημέρωση των σελίδων, την αποτελεσματική μετακίνηση του επισκέπτη μέσα σε αυτές, την εύκολη ανεύρεση των πληροφοριών και την απλοποίηση της επέκτασης των ήδη υπαρχουσών σελίδων με προσθήκη συμπληρωματικών πληροφοριών. Τέλος, παράλληλα με το δεύτερο αυτό στάδιο ανάπτυξης, ο χρήστης θα πρέπει να προσδιορίσει και τη λειτουργική φυσιογνωμία της σελίδας του.

2	**Chardonnay**	**Buena Vista**	**92**	**1**	94
3	**Chardonnay**	**Geyser Peak**	**92**	**5**	94
6	**Chardonnay**	**Stonestreet**	**91**	**4**	93
12	**Jo. Riesling**	**Jekel**	**93**	**1**	94
21	**Fume Blance**	**Ch. St. Jean**	**92**	**4**	94
22	**Fume Blance**	**Robt. Mondavi**	**91**	**2**	93
30	**Gewurztraminer**	**Ch. St. Jean**	**93**	**3**	94
43	**Cab. Sauvignon**	**Windsor**	**86**	**12**	95
45	**Cab. Sauvignon**	**Geyser Peak**	**89**	**12**	97
48	**Cab. Sauvignon**	**Robt. Mondavi**	**88**	**12**	99
50	**Pinot Noir**	**Gary Farrell**	**91**	**3**	94
51	**Pinot Noir**	**Stemmler**	**88**	**3**	95
52	**Pinot Noir**	**Dehlinger**	**90**	**2**	93
58	**Merlot**	**Clos du Bois**	**89**	**9**	95
64	**Zinfandel**	**Lytton Spring**	**89**	**9**	98
72	Zinfandel	Rafanelli	90	2	98

ΕΙΚ. 1.1 Η βάση δεδομένων της κάβας κρασιών (αρχείο ΚΑΒΑ)

Ανάκληση δεδομένων:

SELECT ΚΡΑΣΙ, ΚΑΒΑ, ΠΑΡΑΓΩΓΟΣ
FROM ΚΑΒΑ
WHERE ΩΡΙΜΑΝΣΗ = 95 ;

Αποτέλεσμα (όπως *θα ερφανιζάταν π.χ. σε μια οθόνη):*

ΚΡΑΣΙ	ΡΑΦΙ	ΠΑΡΑΓΩΓΟΣ
Cab. Sauvignon	43	Windsor
Pinot Noir	51	Stemmler
Merlot	58	Clos du Bois

ΕΙΚ. 1.2 Παράδειγμα ανάκλησης δεδομένων από τη βάση δεδομένων της κάβας κρασιών.

Εισαγωγή νέων δεδομένων:

```
INSERT
INTO ΚΑΒΑ     (ΡΑΦΙ, ΚΡΑΣΙ, ΠΑΡΑΓΩΓΟΣ, ΕΤΟΣ, ΦΙΑΛΕΣ, ΩΡΙΜΑΝΣΗ)
       VALUES ( 53, 'Pinot Noir', 'Saintsbury', 92 , 1, 96) ;
```

Ενημέρωση υπαρχόντων δεδομένων:

```
UPDATE ΚΑΒΑ
SET       ΦΙΑΛΕΣ = 4
WHERE  ΡΑΦΙ = 3 ;
```

Διαγραφή υπαρχόντων δεδομένων:

```
DELETE
FROM ΚΑΒΑ
WHERE ΡΑΦΙ = 2 ;
```

ΕΙΚ. 1.3 Παραδείγματα εισαγωγής (INSERT), ενημέρωσης (UPDATE), και διαγραφής (DELETE)

- Δεύτερον, οι γραμμές ενός τέτοιου πίνακα μπορούμε να θεωρήσουμε ότι αντιπροσωπεύουν τις **εγγραφές** (records) του αρχείου - που μερικές φορές λέγονται λογικές εγγραφές (logical records), για να διακρίνονται από άλλα είδη εγγραφών που θα εξετάσουμε αργότερα. Με παρόμοιο τρόπο, μπορούμε να θεωρήσουμε ότι οι στήλες αντιπροσωπεύουν τα **πεδία** (fields) αυτών των λογικών εγγραφών. Θα χρησιμοποιούμε συνήθως τους όρους "εγγραφή" και "πεδίο" όταν αναφερόμαστε στα συστήματα βάσεων δεδομένων γενικά, και τους όρους "γραμμή" και "στήλη" όταν αναφερόμαστε στα σχεσιακά συστήματα ειδικότερα.

- Τρίτον, οι πράξεις επιλογής (SELECT), εισαγωγής (INSERT), ενημέρωσης (UPDATE) και διαγραφής (DELETE), που φαίνονται στις Εικόνες 1.2 και 1.3 παραπάνω, είναι στην πραγματικότητα εντολές μιας γλώσσας βάσεων δεδομένων που ονομάζεται **SQL**.Η **SQL** είναι η γλώσσα που υποστηρίζεται σήμερα από τα περισσότερα συστήματα βάσεων δεδομένων του εμπορίου· στην πράξη μάλιστα, είναι η επίσημη πρότυπη γλώσσα για τα σχεσιακά συστήματα (περισσότερες λεπτομέρειες στην Ενότητα 1.6, παρακάτω). Το όνομα "SQL" προήλθε από τα αρχικά των λέξεων "Structured Query Language" (Δομημένη Γλώσσα Ερωτημάτων), και στην αρχή προφερόταν "σίκουελ". Σήμερα όμως, που η γλώσσα έχει γίνει πλέον πρότυπο, το όνομά της δεν είναι παρά ένα όνομα — επίσημα, δεν είναι σύντμηση - και έχει επικρατήσει η προφορά "ες-κιου-ελ".

1.2.2 Τι είναι σύστημα βάσης δεδομένων

Επαναλαμβάνοντας τον ορισμό που δώσαμε στην Ενότητα 1.1, σύστημα βάσης δεδομένων είναι βασικά ένα ηλεκτρονικό σύστημα τήρησης εγγραφών· με απλά λόγια, είναι ένα σύστημα για υπολογιστές, που ο γενικός σκοπός του είναι να τηρεί πληροφορίες και να δίνει αυτές τις πληροφορίες όταν του ζητούνται. Οι πληροφορίες που τηρούνται σε ένα τέτοιο σύστημα μπορεί να είναι οτιδήποτε έχει σημασία για το άτομο ή τον οργανισμό που εξυπηρετεί το συγκεκριμένο σύστημα — με άλλα λόγια, οτιδήποτε χρειάζεται για την υποβοήθηση των εργασιών αυτού του ατόμου ή οργανισμού.

Σημείωση: Οι όροι "δεδομένα" (data) και "πληροφορίες" (information) θεωρούνται συνώνυμοι. Κάποιοι συγγραφείς προτιμούν να τους ξεχωρίζουν, χρησιμοποιώντας τον όρο "δεδομένα" για να αναφέρονται στις τιμές που είναι πραγματικά αποθηκευμένες στη βάση δεδομένων και τον όρο "πληροφορίες" για να αναφέρονται στη σημασία αυτών των τιμών όπως τις αντιλαμβάνεται κάποιος χρήστης. Η διάκριση αυτή είναι οπωσδήποτε σημαντική - τόσο σημαντική, που είναι μάλλον προτιμότερο να επισημαίνεται ρητά όπου χρειάζεται, παρά να βασίζεται κανείς σε μια μάλλον αυθαίρετη διαφοροποίηση της σημασίας δύο όρων που είναι στην ουσία όμοιοι.

Μια άποψη ενός συστήματος βάσης δεδομένων, απλοποιημένη σε πολύ μεγάλο βαθμό, είναι ένα σύστημα βάσης δεδομένων το οποίο απαρτίζεται από τέσσερα βασικό στοιχεία: τα **δεδομένα**, το **υλικό** (hardware), το **λογισμικό** (software), και τους **χρήστες** (users). Θα εξετάσουμε με συντομία αυτά τα τέσσερα στοιχεία στη συνέχεια.

Δεδομένα

Υπάρχουν συστήματα βάσεων δεδομένων για μηχανήματα που εκτείνονται σε όλο το φάσμα - από τους μικροϋπολογιστές (ακόμα και τα φορητά PC) μέχρι τα μεγαλύτερα συστήματα mainframe. Είναι περιττό να πούμε ότι οι υπηρεσίες που παρέχει το κάθε δεδομένο σύστημα εξαρτώνται ως ένα βαθμό από το μέγεθος και την ισχύ του

μηχανήματος στο οποίο είναι εγκατεστημένο. Ειδικότερα, τα συστήματα για μεγάλα μηχανήματα (τα "μεγάλα συστήματα") είναι συνήθως συστήματα πολλών χρηστών, ενώ εκείνα των μικρότερων μηχανημάτων (Τα "μικρά συστήματα") είναι συνήθως συστήματα ενός χρήστη. **Σύστημα ενός χρήστη** (single user system) είναι ένα σύστημα στο οποίο μόνο ένας χρήστης έχει πρόσβαση στη βάση δεδομένων οποιαδήποτε δεδομένη στιγμή· **σύστημα πολλών χρηστών** (multiuser system) είναι ένα σύστημα στο οποίο πολλοί χρήστες έχουν πρόσβαση στη βάση δεδομένων ταυτόχρονα. Συνήθως θα θεωρούμε ότι αναφερόμαστε στη δεύτερη περίπτωση, που είναι πιο γενική, αλλά στην πράξη αυτή η διάκριση δεν έχει σχεδόν καμία σημασία σε ό,τι αφορά τους περισσότερους χρήστες: ένας από τους πρωταρχικούς στόχους των περισσότερων συστημάτων πολλών χρηστών είναι να επιτρέπουν στον κάθε μεμονωμένο χρήστη να συμπεριφέρεται σαν να δούλευε σε σύστημα ενός χρήστη. Τα ειδικά προβλήματα των συστημάτων πολλών χρηστών είναι κυρίως προβλήματα εγγενή στο ίδιο το σύστημα, και δεν είναι ορατά στο χρήστη .

Με την ευκαιρία, συνήθως είναι βολικό να θεωρεί κανείς για χάρη της απλότητας ότι το σύνολο των δεδομένων που είναι αποθηκευμένα στο σύστημα τηρούνται σε μία μόνο βάση δεδομένων. Στην πράξη πάντως, ακόμα και σε ένα μικρό σύστημα, μπορεί να υπάρχουν σοβαροί λόγοι για τους οποίους τα δεδομένα θα πρέπει να μοιραστούν σε πολλές χωριστές βάσεις δεδομένων.

Γενικά, τα δεδομένα της βάσης δεδομένων - τουλάχιστον σε ένα μεγάλο σύστημα - είναι *ενοποιημένα* (integrated) και *μεριζόμενα* (shared — δηλαδή κοινόχρηστα). Όπως θα δούμε στην Ενότητα 1.4, αυτές οι δύο πλευρές, η ενοποίηση και ο μερισμός των δεδομένων, είναι από τα μεγαλύτερα πλεονεκτήματα των συστημάτων βάσεων δεδομένων σε ένα "μεγάλο" περιβάλλον. Επίσης, τουλάχιστον η ενοποίηση των δεδομένων μπορεί να είναι εξίσου σημαντική σε ένα "μικρό" περιβάλλον. Υπάρχουν βέβαια και πολλά άλλα πλεονεκτήματα (στα οποία Θα αναφερθούμε αργότερα), ακόμη και σε ένα μικρό περιβάλλον. Πρώτα, όμως, ας εξηγήσουμε τι εννοούμε με τους όρους "ενοποίηση" και "μερισμός".

- Με τον όρο **ενοποίηση δεδομένων** (data integration) εννοούμε ότι η βάση δεδομένων μπορεί να θεωρείται μια συνένωση πολλών αρχείων δεδομένων, που από κάθε άλλη άποψη είναι ξεχωριστά το ένα από το άλλο, ενώ κάθε πλεονασμός εξαιτίας της επανάληψης δεδομένων μεταξύ αυτών των αρχείων έχει εξαλειφθεί εντελώς **ή** κατά ένα μέρος. Για παράδειγμα, μια βάση δεδομένων μπορεί να περιλαμβάνει ένα αρχείο ΥΠΑΛΛΗΛΟΙ, που περιέχει τα ονόματα, τις διευθύνσεις, τα τμήματα, τους μισθούς κ.λπ. των υπαλλήλων, και ένα αρχείο ΣΥΜΜΕΤΟΧΕΣ, που αντιπροσωπεύει τις συμμετοχές υπαλλήλων σε κάποια εκπαιδευτικά σεμινάρια (δείτε στην Εικόνα *1.5)*. Ας υποθέσουμε τώρα ότι, για να υποστηριχθεί η διαχείριση των εκπαιδευτικών σεμιναρίων, πρέπει να είναι γνωστό το τμήμα στο οποίο ανήκει ο κάθε υπάλληλος που συμμετέχει. Είναι σαφές ότι, σε αυτή την περίπτωση, δεν υπάρχει λόγος να συμπεριληφθούν αυτές οι (πλεονάζουσες) πληροφορίες στο αρχείο ΣΥΜΜΕΤΟΧΕΣ, επειδή μπορεί κανείς να τις βρίσκει από το αρχείο ΥΠΑΛΛΗΛΟΙ.

ΥΠΑΛΛΗΛΟΙ

ΣΥΜΜΕΤΟΧΕΣ

ΕΙΚ. 1.5 Τα αρχεία ΥΠΑΛΛΗΛΟΙ και ΣΥΜΜΕΤΟΧΕΣ

- Με τον όρο **μερισμός δεδομένων** (data sharing) εννοούμε οτι τα μεμονωμένα στοιχεία δεδομένων της βάσης δεδομένων μπορούν να τα μοιράζονται πολλοί διαφορετικοί χρήστες, με την έννοια ότι καθένας από αυτούς τους χρήστες μπορεί να έχει πρόσβαση στο ίδιο στοιχείο δεδομένων (και οι διάφοροι χρήστες μπορούν να το χρησιμοποιούν για διαφορετικό σκοπό). Όπως αναφέραμε και νωρίτερα, οι διάφοροι χρήστες μπορούν να έχουν πρόσβαση στο ίδιο στοιχείο δεδομένων την ίδια στιγμή ("ταυτόχρονη προσπέλαση"). Αυτού του είδους ο μερισμός (ταυτόχρονος ή όχι) είναι σε ένα βαθμό συνέπεια του γεγονότος ότι η βάση δεδομένων είναι ενοποιημένη. Στο παραπάνω παράδειγμά μας, των υπαλλήλων και των συμμετοχών τους σε σεμινάρια, τις πληροφορίες του αρχείου ΥΠΑΛΛΗΛΟΙ για τα τμήματα στα οποία ανήκουν οι υπάλληλοι θα τις μοιράζονται κατά κανόνα οι χρήστες που ανήκουν στο Τμήμα Προσωπικού και στο Τμήμα Εκπαίδευσης και, όπως είδαμε προηγουμένως, αυτές οι δύο κατηγορίες χρηστών Θα χρησιμοποιούν κατά κανόνα αυτές τις πληροφορίες για διαφορετικό σκοπό.

Μια άλλη συνέπεια του γεγονότος ότι η βάση δεδομένων είναι ενοποιημένη είναι ότι κάθε δεδομένος χρήστης ενδιαφέρεται κατά κανόνα μόνο για ένα πολύ μικρό τμήμα της συνολικής βάσης δεδομένων (και επιπλέον, τα τμήματα των διαφορετικών χρηστών θα επικαλύπτονται με πολλούς και διάφορους τρόπους). Με άλλα λόγια, η αντίληψη που έχουν οι διαφορετικοί χρήστες για την ίδια βάση δεδομένων διαφέρει από πολλές απόψεις. Στην πραγματικότητα μάλιστα, ακόμη και όταν δύο χρήστες μοιράζονται το ίδιο τμήμα της βάσης δεδομένων, ο τρόπος που βλέπουν αυτό το τμήμα είναι πολύ πιθανό να διαφέρει σημαντικά σε ένα πιο λεπτομερές επίπεδο. Αυτό το τελευταίο σημείο αναλύεται εκτενέστερα στην Ενότητα 1.5 και στο επόμενο κεφάλαιο.

Υλικό

Τα μέρη του συστήματος που έχουν να κάνουν με το υλικό (hardware) είναι τα εξής:

- Τα μέσα δευτερεύουσας αποθήκευσης κατά κανόνα, μαγνητικοί δίσκοι με κινούμενες κεφαλές που χρησιμοποιούνται για την αποθήκευση των δεδομένων, καθώς και οι αντίστοιχες συσκευές εισόδου/εξόδου, όπως οι μονάδες δίσκων

(disk drives) κ.λπ., οι ελεγκτές συσκευών (device controllers), τα κανάλια εισόδου/εξόδου,κ.ο.κ.

- Ο επεξεργαστής ή οι επεξεργαστές (processor) και η κύρια μνήμη (main memory), που χρησιμοποιούνται για την εκτέλεση του λογισμικού του συστήματος βάσης δεδομένων.

Λογισμικό

Μεταξύ της ίδιας της φυσικής βάσης δεδομένων (δηλαδή, των δεδομένων όπως είναι αποθηκευμένα στην πραγματικότητα) και των χρηστών του συστήματος υπάρχει ένα επίπεδο λογισμικού (software), ο **διαχειριστής βάσεων δεδομένων** (database manager) ή, όπως είναι ευρύτερα γνωστό, το **σύστημα διαχείρισης βάσεων δεδομένων** (database management system, **DBMS**). Το **DBMS** διαχειρίζεται όλες τις αιτήσεις των χρηστών για προσπέλαση της βάσης δεδομένων· οι λειτουργίες που σκιαγραφήσαμε στην Ενότητα 1.1, για την προσθήκη και την αφαίρεση αρχείων (ή πινάκων), την ανάκληση και την ενημέρωση δεδομένων που είναι αποθηκευμένα σε τέτοια αρχεία ή πίνακες κ.λπ., είναι όλες υπηρεσίες που παρέχονται από το DBMS. Έτσι, μπορεί να πει κανείς ότι μια γενικότερη υπηρεσία που παρέχεται από το DBMS είναι η απομόνωση των χρηστών της βάσης δεδομένων από τις λεπτομέρειες που αφορούν το υλικό (με τον ίδιο τρόπο που τα συστήματα γλωσσών Προγραμματισμού απομονώνουν τους προγραμματιστές εφαρμογών από τις λεπτομέρειες που αφορούν το υλικό). Με άλλα λόγια, το DBMS παρέχει στους χρήστες μια άποψη της βάσης δεδομένων "ανυψωμένη" κατά κάποιον τρόπο πάνω από το επίπεδο του υλικού, και υποστηρίζει τις πράξεις των χρηστών (όπως οι πράξεις της SQL στις οποίες αναφερθήκαμε με συντομία στην Ενότητα 1.1), οι οποίες εκφράζονται με βάση αυτή την άποψη ανωτέρου επιπέδου. Θα περιγράψουμε αυτή τη λειτουργία, καθώς και άλλες λειτουργίες του **DBMS**, με πολύ περισσότερες λεπτομέρειες, στη συνέχεια.

Σημείωση: Είναι σαφές ότι το **DBMS** είναι το σημαντικότερο στοιχείο λογισμικού ολόκληρου του συστήματος, αλλά δεν είναι το μόνο. Άλλα τέτοια στοιχεία είναι τα βοηθητικά προγράμματα, τα εργαλεία ανάπτυξης εφαρμογών, τα σχεδιαστικά βοηθήματα, τα εργαλεία σύνταξης αναφορών, κ.ο.κ.

Χρήστες

Διακρίνουμε τρεις γενικές κατηγορίες χρηστών:

- Στην πρώτη κατηγορία ανήκουν οι **προγραμματιστές εφαρμογών** (application programmers), οι οποίοι είναι υπεύθυνοι για το γράψιμο προγραμμάτων

εφαρμογών που χρησιμοποιούν τη βάση δεδομένων, κατά κανόνα σε κάποια γλώσσα όπως η **COBOL** ή η **PL/I** ή σε κάποια πιο σύγχρονη όπως η C ή η Pascal Αυτά τα προγράμματα επενεργούν στα δεδομένα με όλους τους συνηθισμένους τρόπους — ανακαλώντας πληροφορίες που υπάρχουν ήδη, προσθέτοντας νέες πληροφορίες, και διαγράφοντας ή αλλάζοντας υπάρχουσες πληροφορίες. 'Ολες αυτές οι ενέργειες εκτελούνται φυσικά με την υποβολή καταλλήλων αιτήσεων στο DBMS. Τα ίδια τα προγράμματα μπορεί να είναι συμβατικές εφαρμογές ομαδικής επεξεργασίας (batch applications) ή εφαρμογές **άμεσης επεξεργασίας** (online applications), και στόχος τους είναι να υποστηρίζουν τον τελικό χρήστη (δείτε στην επόμενη παράγραφο) που προσπελάζει τη βάση δεδομένων από κάποιο σταθμό εργασίας ή τερματικό. Οι περισσότερες σημερινές εφαρμογές ανήκουν σε αυτή την τελευταία κατηγορία.

- Στη δεύτερη κατηγορία χρηστών ανήκουν σι **τελικοί χρήστες**, σι οποίοι αλληλεπιδρούν με το σύστημα μέσω συνδεδεμένων σταθμών εργασίας ή

τερματικών. 'Ενας τελικός χρήστης μπορεί να προσπελάζει τη βάση δεδομένων μέσω κάποιας από τις εφαρμογές άμεσης επεξεργασίας που αναφέραμε στην προηγούμενη παράγραφο, ή να χρησιμοποιεί κάποια διασύνδεση (interface) που είναι οργανικό μέρος του λογισμικού του συστήματος βάσης δεδομένων. Βέβαια, τέτοιες διασυνδέσεις υποστηρίζονται και από εφαρμογές άμεσης επεξεργασίας· όμως, οι εφαρμογές αυτές είναι **ενσωματωμένες** (built-in) και όχι γραμμένες από κάποιο χρήστη. Τα περισσότερα συστήματα διαθέτουν τουλάχιστον μία τέτοια ενσωματωμένη εφαρμογή, δηλαδή έναν αλληλεπιδραστικό **επεξεργαστή γλώσσας ερωτημάτων** (interactive query language processor). Με τη βοήθεια αυτής της ενσωματωμένης εφαρμογής, ο χρήστης έχει τη δυνατότητα να δίνει στο DBMS διαταγές ή εντολές υψηλού επιπέδου (όπως οι **SELECT, INSERT** κ.λπ.). Η γλώσσα **SQL**, στην οποία αναφερθήκαμε στην Ενότητα 1.1, μπορεί να θεωρηθεί τυπικό παράδειγμα γλώσσας για την υποβολή ερωτημάτων στη βάση δεδομένων.

Σημείωση.· Αν και ο όρος "γλώσσα υποβολής ερωτημάτων" ή απλώς "γλώσσα ερωτημάτων" (query language) — χρησιμοποιείται πολύ συχνά, είναι μάλλον ανακριβής, επειδή η λέξη "ερώτημα" ("query" στα Αγλικά) υπονοεί μόνο την *ανάκληση* δεδομένων, ενώ οι γλώσσες ερωτημάτων κατά κανόνα δίνουν επίσης τη δυνατότητα για πράξεις ενημέρωσης, εισαγωγής και διαγραφής (ή ίσως και άλλες).

Τα περισσότερα συστήματα παρέχουν επίσης στους χρήστες τους πρόσθετες ενσωματωμένες διασυνδέσεις, όπου οι χρήστες δε δίνουν ρητές διαταγές όπως η SELECT, αλλά δουλεύουν π.χ. επιλέγοντας στοιχεία από ένα μενού ή συμπληρώνοντας πλαίσια σε μια φόρμα. Αυτού του είδους οι διασυνδέσεις που **οδηγούνται από μενού** ή από **φόρμες** (menu-driven ή forms-driven interfaces) τείνουν να είναι πολύ πιο εύχρηστες για τους χρήστες που δεν έχουν τυπική εκπαίδευση στην τεχνολογία πληροφοριών. Αντίθετα, οι διασυνδέσεις που **οδηγούνται από διαταγές** (command-driven interfaces) — όπως οι γλώσσες ερωτημάτων — απαιτούν ένα ορισμένο επίπεδο επαγγελματικής κατάρτισης στην τεχνολογία πληροφοριών, αν και όχι πολύ υψηλό (προφανώς όχι τόσο όσο χρειάζεται για το γράψιμο ενός προγράμματος εφαρμογής σε μια γλώσσα σαν την COBOL). Από την άλλη όμως, μια διασύνδεση οδηγούμενη από διαταγές είναι

μάλλον πιο ευέλικτη από μια διασύνδεση οδηγούμενη από μενού ή φόρμες, επειδή οι γλώσσες ερωτημάτων διαθέτουν συνήθως και ορισμένες λειτουργίες που δεν υποστηρίζονται από αυτές τις άλλες διασυνδέσεις.

- Στην τρίτη κατηγορία χρηστών ανήκει ο **υπεύθυνος διαχείρισης βάσεων δεδομένων** (database administrator, DBA). Η περιγραφή της δουλειάς του DBA, καθώς και η σχετική (και πολύ σημαντική) δουλειά του υπεύθυνου διαχείρισης δεδομένων (data administrator, DA) περιγράφονται στις Ενότητες 1.4 .

Εδώ, ολοκληρώθηκε η προκαταρκτική παρουσίαση των κυριότερων στοιχείων ενός συστήματος βάσης δεδομένων. Θα εξετάσουμε τώρα αυτές τις έννοιες με κάπως περισσότερες λεπτομέρειες.

1.2.3 Τι είναι βάση δεδομένων

Μόνιμα δεδομένα

Συνηθίζεται να αναφερόμαστε στα δεδομένα μιας βάσης δεδομένων με τον όρο "μόνιμα δεδομένα" (persistent data) — αν και μπορεί να μην παραμείνουν μόνιμα για πολύ! Με τον όρο "μόνιμα" εννοούμε ότι τα δεδομένα της βάσης δεδομένων έχουν ποιοτική διαφορά από τα άλλα — τα πιο εφήμερα — δεδομένα, όπως είναι τα δεδομένα εισόδου, τα δεδομένα εξόδου, οι εντολές ελέγχου, οι ουρές εργασιών, τα μπλοκ ελέγχου του λογισμικού, τα ενδιάμεσα αποτελέσματα, και, γενικά, όλα τα δεδομένα που είναι παροδικά (transient) από τη φύση τους. Ας σταθούμε λίγο στους όρους "δεδομένα εισόδου" και "δεδομένα εξόδου":

- "Δεδομένα εισόδου" (input data) είναι οι πληροφορίες που εισέρχονται στο σύστημα για πρώτη φορά (συνήθως από κάποιο τερματικό ή σταθμό εργασίας). Οι πληροφορίες αυτές μπορεί να προκαλέσουν κάποια αλλαγή στα μόνιμα δεδομένα (μπορεί να ενταχθούν στα μόνιμα δεδομένα), αλλά οι ίδιες δεν αποτελούν μέρος της βάσης δεδομένων αρχικά.

- Με ανάλογο τρόπο, "δεδομένα εξόδου" είναι τα μηνύματα και τα αποτελέσματα που εξέρχονται από το σύστημα (και κατά κανόνα τυπώνονται σε χαρτί ή εμφανίζονται στην οθόνη). Και αυτές οι πληροφορίες μπορεί να προέρχονται από τα μόνιμα δεδομένα αλλά οι ίδιες δε θεωρούνται μέρος της βάσης δεδομένων.

Βέβαια, η διάκριση ανάμεσα στα μόνιμα και τα παροδικά δεδομένα δεν είναι ούτε απόλυτη ούτε ξεκάθαρη — εξαρτάται ως ένα βαθμό και από τα συμφραζόμενα (δηλαδή, από τον τρόπο με τον οποίο χρησιμοποιούνται τα δεδομένα). Ωστόσο, αν δεχτούμε ότι αυτή η διάκριση έχει κάποιο νόημα εμπειρικά, μπορούμε τώρα να δώσουμε έναν κάπως ακριβέστερο ορισμό του όρου "βάση δεδομένων":

- Μια **βάση δεδομένων** αποτελείται από κάποια συλλογή μόνιμων δεδομένων που χρησιμοποιούνται από τα συστήματα των εφαρμογών μιας δεδομένης επιχείρησης.

Εδώ, ο όρος "επιχείρηση" δεν είναι παρά ένας βολικός γενικός όρος που σημαίνει οποιονδήποτε αυτόνομο εμπορικό, επιστημονικό, τεχνικό, ή άλλο οργανισμό. Επιχείρηση μπορεί να είναι ένα μεμονωμένο άτομο (που διαθέτει μια μικρή ιδιωτική βάση δεδομένων), μια ολόκληρη εταιρεία ή ένας παρόμοιος οργανισμός (που διαθέτει μια πολύ μεγάλη μεριζόμενη βάση δεδομένων), ή οτιδήποτε ανάμεσα σε αυτά τα δύο άκρα. Ας δούμε μερικά παραδείγματα:

1. Μια κατασκευαστική εταιρεία
2. Μια τράπεζα
3. Ένα νοσοκομείο
4. Ένα πανεπιστήμιο
5. Μια κρατική υπηρεσία

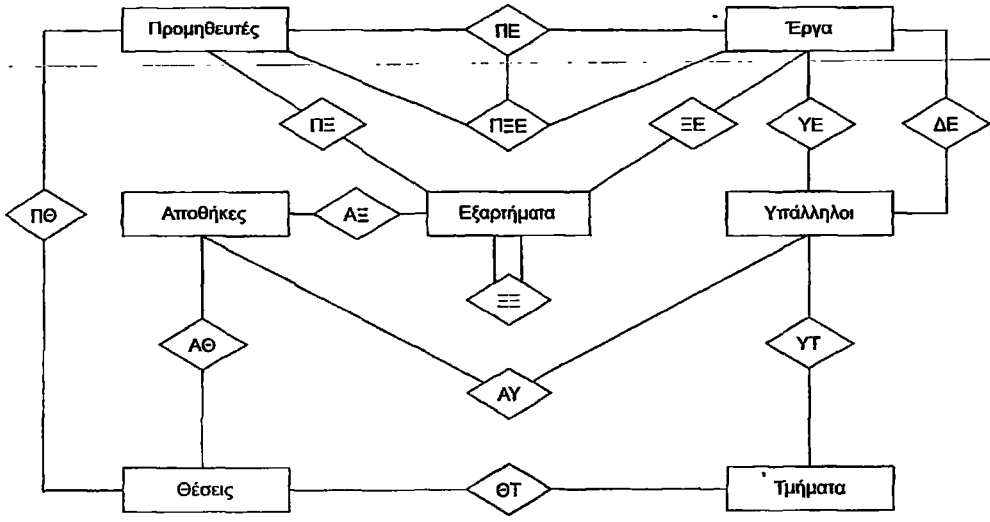
Κάθε επιχείρηση πρέπει αναγκαστικά να τηρεί πολλά δεδομένα που αφορούν τη λειτουργία της. Αυτά είναι τα "μόνιμα δεδομένα" στα οποία αναφερθήκαμε παραπάνω. Οι επιχειρήσεις που παραθέσαμε νωρίτερα θα πρέπει λογικά να περιλαμβάνουν στα μόνιμα δεδομένα τους και τα εξής:

1. Δεδομένα για προϊόντα
2. Δεδομένα για λογαριασμούς
3. Δεδομένα για ασθενείς
4. Δεδομένα για φοιτητές
5. Δεδομένα για σχεδιασμούς

Σημείωση:Παλιότερα χρησιμοποιούσαμε τον όρο "επιχειρησιακά δεδομένα" (operational data) αντί για τον όρο "μόνιμα δεδομένα". Αυτός ο παλιότερος όρος αντανακλούσε την έμφαση που δινόταν αρχικά από τα συστήματα βάσεων δεδομένων για τις **επιχειρησιακές** εφαρμογές ή εφαρμογές **παραγωγής** — δηλαδή, για τις εφαρμογές ρουτίνας με υψηλό βαθμό επανάληψης, που εκτελούνταν διαρκώς για να υποστηρίζουν τις καθημερινές δραστηριότητες μιας επιχείρησης (όπως, για παράδειγμα, μια εφαρμογή για την υποστήριξη των αναλήψεων και καταθέσεων χρημάτων σε ένα τραπεζικό σύστημα). Σήμερα, που οι βάσεις δεδομένων χρησιμοποιούνται όλο και περισσότερο και για άλλου είδους εφαρμογές — όπως είναι οι εφαρμογές υποστήριξης αποφάσεων (decision support) — ο όρος "επιχειρησιακά δεδομένα" δεν είναι πια εντελώς κατάλληλος. Στην πράξη μάλιστα, οι επιχειρήσεις διατηρούν στις μέρες μας δύο ξεχωριστές βάσεις δεδομένων, μία για τα επιχειρησιακά δεδομένα και άλλη μία για τα δεδομένα υποστήριξης αποφάσεων. Η βάση δεδομένων υποστήριξης αποφάσεων αποτελείται συχνά από συγκεντρωτικές πληροφορίες (π.χ. Πληροφορίες για συνολικές ποσότητες και μέσους όρους), οι οποίες εξάγονται περιοδικά (για παράδειγμα, καθημερινά ή εβδομαδιαία) από την επιχειρησιακή βάση δεδομένων.

Οντότητες και συσχετίσεις

Ας εξετάσουμε το παράδειγμα μιας κατασκευαστικής εταιρείας σε λίγο μεγαλύτερο βάθος. Μια τέτοια επιχείρηση θα πρέπει λογικά να τηρεί πληροφορίες σχετικά με τα έργα που έχει αναλάβει, τα εξαρτήματα που χρησιμοποιούνται σε αυτά τα έργα, τους προμηθευτές που της προμηθεύουν αυτά τα εξαρτήματα, τις αποθήκες όπου φυλάγονται αυτά τα εξαρτήματα, τους υπαλλήλους που απασχολούνται σε αυτά τα έργα, κ.ο.κ. Τα έργα, τα εξαρτήματα, οι προμηθευτές, κ.λπ. είναι λοιπόν σι βασικές **οντότητες** (entities) για τις οποίες πρέπει να τηρεί η επιχείρηση πληροφορίες. (Ο όρος "οντότητα" χρησιμοποιείται ευρέως στον κόσμο των βάσεων δεδομένων για να περιγράψει καθε διακεκριμένο αντικείμενο που αναπαρίσταται στη βάση δεδομένων,) Δείτε στην Εικόνα 1.6.



ΕΙΚ. 1.6 Ένα απλό διάγραμμα οντοτήτων/συσχετίσεων

Είναι πολύ σημαντικό να κατανοήσετε ότι, εκτός από τις ίδιες τις βασικές οντότητες, υπάρχουν και **συσχετίσεις** (relationships) που συνδέουν αυτές τις βασικές οντότητες μεταξύ τους. Αυτές οι συσχετίσεις παριστάνονται στην Εικόνα 1.6 με ρόμβους και συνδετικές γραμμές. Για παράδειγμα, υπάρχει μια συσχέτιση (η "ΠΞ") μεταξύ προμηθευτών και εξαρτημάτων: ο κάθε προμηθευτής προμηθεύει την επιχείρηση με ορισμένα εξαρτήματα και, αντίστροφα, κάθε εξάρτημα το παρέχουν ορισμένοι προμηθευτές (για την ακρίβεια, ο κάθε προμηθευτής παρέχει ορισμένα είδη εξαρτημάτων και κάθε είδος εξαρτημάτων το παρέχουν ορισμένοι προμηθευτές). Με ανάλογο τρόπο, τα εξαρτήματα χρησιμοποιούνται σε έργα και, αντίστροφα, τα έργα χρησιμοποιούν εξαρτήματα (συσχέτιση "ΞΕ"), τα εξαρτήματα φυλάγονται σε αποθήκες και οι αποθήκες περιέχουν εξαρτήματα (συσχέτιση "ΑΞ") κ.ο.κ. Προσέξτε ότι όλες αυτές οι συσχετίσεις είναι διπλής κατεύθυνσης — δηλαδή, μπορούν να διατρέχονται και προς τις δύο κατευθύνσεις. Για παράδειγμα, η συσχέτιση ΠΞ μεταξύ των προμηθευτών και των εξαρτημάτων μπορεί να δώσει απαντήσεις και στα δύο επόμενα ερωτήματα:

- Με δεδομένο έναν προμηθευτή, να βρεθούν τα εξαρτήματα που παρέχονται από αυτόν τον προμηθευτή.

- Με δεδομένο ένα εξάρτημα, να βρεθεί ο προμηθευτής που παρέχει αυτό το εξάρτημα.

Το σημαντικό σε αυτή τη συσχέτιση, και σε όλες τις υπόλοιπες που παρουσιάζονται σε αυτή την εικόνα, είναι ότι αποτελούν *τμήμα* των *δεδομένων,* όπως ακριβώς *και* οι βασικές οντότητες. Πρέπει λοιπόν να αναπαριστώνται στη βάση δεδομένων, όπως και οι βασικές οντότητες. Στη συνέχεια, θα εξετάσουμε τους τρόπους με τους οποίους μπορεί να γίνει αυτό.

Με την ευκαιρία, η Εικόνα 1.6 είναι ένα απλό παράδειγμα αυτού που (για προφανείς λόγους) ονομάζεται **διάγραμμα οντοτήτων/συσχετίσεων** (entity/relationship diagram, E/R diagram).

Η Εικόνα 1.6, όμως, φέρνει στην επιφάνεια και ορισμένα άλλα σημεία:

1. Το γεγονός ότι οι περισσότερες συσχετίσεις του διαγράμματος αφορούν δύο τύπους οντοτήτων — δηλαδή, είναι *δυαδικές* (binary) συσχετίσεις — δε σημαίνει με κανέναν τρόπο ότι όλες οι συσχετίσεις πρέπει υποχρεωτικά να είναι δυαδικές. Στο παράδειγμά μας υπάρχει μια συσχέτιση (η "ΠΞΕ") που αφορά τρεις τύπους οντοτήτων (προμηθευτές, εξαρτήματα, και έργα) — πρόκειται, δηλαδή, για τριαδική (ternary) συσχέτιση. Η ερμηνεία αυτής της συσχέτισης είναι ότι ορισμένοι προμηθευτές παρέχουν ορισμένα εξαρτήματα που χρησιμοποιούνται σε ορισμένα έργα. Πρέπει να δώσουμε ιδιαίτερη προσοχή στο ότι αυτή η τριαδική συσχέτιση ("προμηθευτές παρέχουν εξαρτήματα σε έργα") δεν είναι γενικά ισοδύναμη με το συνδυασμό των τριών δυαδικών συσχετίσεων "προμηθευτές παρέχουν εξαρτήματα", "εξαρτήματα χρησιμοποιούνται σε έργα", και "προμηθευτές προμηθεύουν έργα". Για παράδειγμα, η πρόταση

   (α) "ο Γεωργίου εφοδιάζει με κατσαβίδια το έργο Ακράτα"

   μας λέει περισσότερα από τις τρεις προτάσεις

   (β) "ο Γεωργιου διαθετει κατσαβιδια" ,

   (γ) *"στο έργο Ακράτα χρησιμοποιούνται κατσαβίδια", και*

   (δ) "ο Γεωργίου προμηθεύει το έργο Ακράτα",

   επειδή δεν μπορούμε να συμπεράνουμε βάσιμα το (α) γνωρίζοντας μόνο τα (β), (γ), και (δ). Για την ακρίβεια, αν γνωρίζουμε τα (β), (γ), και (δ), τότε μπορούμε να συμπεράνουμε ότι ο Γεωργίου εφοδιάζει με κατσαβίδια *κάποιο* έργο (ας πούμε το έργο Ez, οτι κάποιος προμηθευτής (ας πούμε ο (προμηθευτής Πx) εφοδιάζει το έργο Ακράτα με κατσαβίδια, και ότι ο Γεωργίου εφοδιάζει το έργο Ακράτα με κάποιο εξάρτημα (ας πούμε, το εξάρτημα Ξy) — αλλά δεν μπορούμε να συμπεράνουμε βάσιμα ότι ο Πx, είναι ο Γεωργίου, ότι το Ξy είναι τα κατσαβίδια, ή ότι το Ez είναι το έργο Ακράτα.

Οι λανθασμένοι συλλογισμοί αυτού του είδους είναι παράδειγμα της λεγόμενης **παγίδας των συσχετίσεων** (connection trap).

2. Το διάγραμμα περιλαμβάνει επίσης μια συσχέτιση (την "ΞΞ") που αφορά μόνο έναν τύπο οντότητας (τα εξαρτήματα). Αυτή η συσχέτιση ορίζει ότι κάποια εξαρτήματα περιέχουν άλλα εξαρτήματα ως άμεσα μέρη τους (είναι η λεγόμενη συσχέτιση **κατάστασης υλικών** — bill-of-materials). Για παράδειγμα, μια βίδα είναι μέρος ενός μεντεσέ, ο οποίος θεωρείται και αυτός εξάρτημα, και μπορεί με τη σειρά του να είναι μέρος κάποιου εξαρτήματος ανώτερου επιπέδου, όπως ένα παραθυρόφυλλο. Προσέξτε ότι αυτή η συσχέτιση εξακολουθεί να είναι δυαδική· η μόνη διαφορά είναι ότι οι δύο τύποι οντοτήτων που συνδέονται με αυτή τη συσχέτιση, τα εξαρτήματα και τα εξαρτήματα, τυχαίνει να είναι ο ίδιος τύπος οντότητας.

3. Γενικά, ένα δεδομένο σύνολο τύπων οντοτήτων μπορεί να συνδέονται μεταξύ τους με οσεσδήποτε διαφορετικές *συσχετίσεις*. *Στο* διάγραμμα του παραδείγματός μας, υπάρχουν δύο συσχετίσεις μεταξύ έργων και υπαλλήλων: η μία (η **'YE")** δείχνει ότι οι υπάλληλοι κατανέμονται σε έργα, και η άλλη (η "ΔΕ") ότι οι υπάλληλοι διαχειρίζονται έργα.

Σημείωση: Μια *συσχέτιση μπορεί* να θεωρηθεί οντότητα *και* η *ίδια*. Αν θεωρήσουμε ότι οντότητα είναι "οποιοδήποτε αντικείμενο για το οποίο θέλουμε να τηρούμε πληροφορίες", τότε οπωσδήποτε μια συσχέτιση ανταποκρίνεται σε αυτόν τον ορισμό. Για παράδειγμα, η πρόταση "το εξάρτημα Ξ4 φυλάγεται στην αποθήκη Α8" είναι μια οντότητα για την οποία μπορεί να θέλουμε να καταγράφουμε πληροφορίες — για παράδειγμα, την αντίστοιχη ποσότητα. Εξάλλου, υπάρχουν σαφή πλεονεκτήματα τα οποία μπορούμε να έχουμε αν αποφύγουμε να κάνουμε αδικαιολόγητες διακρίσεις μεταξύ οντοτήτων και συσχετίσεων.

Ιδιότητες

Σύμφωνα με τα πρηγούμενα, θεωρούμε οντότητα οποιοδήποτε αντικείμενο για το οποίο χρειάζεται να τηρούμε πληροφορίες. Με άλλα λόγια, οι οντότητες (και κατά συνέπεια, οι σχέσεις) έχουν **ιδιότητες** (properties). Για παράδειγμα, οι προμηθευτές έχουν έδρα, τα εξαρτήματα έχουν βάρος, τα έργα έχουν προτεραιότητα, οι αναθέσεις έχουν ημερομηνία έναρξης, κ.ο.κ. Αυτές οι ιδιότητες πρέπει λοιπόν να αναπαριστώνται στη βάση δεδομένων. Για παράδειγμα, θα μπορούσε η βάση δεδομένων να περιέχει έναν τύπο εγγραφής Π που θα αντιπροσωπεύει τον τύπο οντότητας "προμηθευτής", και αυτός ο τύπος εγγραφής θα μπορούσε να περιέχει με τη σειρά του έναν τύπο πεδίου ΠΟΛΗ που θα αντιπροσωπεύει την ιδιότητα "έδρα".

Οι ιδιότητες, με τη σειρά τους, μπορεί να είναι πολύ απλές ή να έχουν εσωτερική δομή οσοδήποτε σύνθετη. Για παράδειγμα, η ιδιότητα "έδρα προμηθευτή" είναι μάλλον πολύ απλή (αποτελείται από ένα όνομα πόλης) και μπορεί να αναπαρίσταται στη βάση δεδομένων με ένα απλό αλφαριθμητικό (character string). Αντίθετα, μια

αποθήκη μπορεί να έχει μια ιδιότητα "σχέδιο ορόφου", η οποία θα μπορούσε να είναι ιδιαίτερα πολύπλοκη και να αποτελείται από ένα ολόκληρο αρχιτεκτονικό σχέδιο καθώς και από συνοδευτικό περιγραφικό κείμενο. Τα σύγχρονα προϊόντα βάσεων δεδομένων είναι στην πλειοψηφία τους ανεπαρκή σε ό,τι αφορά τη διαχείριση σύνθετων ιδιοτήτων (όπως είναι τα σχέδια ή το κείμενο). Θα θεωρούμε σε γενικές γραμμές (όπου.έχει σημασία) ότι όλες οι ιδιότητες είναι "απλές" και μπορούν να παρασταθούν στη βάση δεδομένων με "απλούς" τύπους δεδομένων.Παραδείγματα τέτοιων "απλών" τύπων δεδομένων είναι οι αριθμοί, τα αλφαριθμητικά ,οι ημερομηνίες, οι ώρες, κ.λπ.

1.2.4 Γιατί βάση δεδομένων;

Γιατί, όμως, να χρησιμοποιήσει κανείς ένα σύστημα βάσης δεδομένων; Τι πλεονεκτήματα έχει; Ως ένα βαθμό, οι απαντήσεις σε αυτά τα ερωτήματα εξαρτώνται από το αν το συγκεκριμένο σύστημα είναι ενός χρήστη (single user) ή πολλών χρηστών (multiuser) . Στην περίπτωση του συστήματος πολλών χρηστών υπάρχουν πολλά πρόσθετα πλεονεκτήματα. Ας εξετάσουμε πρώτα την περίπτωση του συστήματος ενός χρήστη.

Θα επανέλθουμε για μία ακόμη φορά στο παράδειγμα της κάβας κρασιών (Εικόνα 1.1), το οποίο και μπορούμε να θεωρήσουμε τυπική περίπτωση βάσης δεδομένων ενός χρήστη. Αυτή η συγκεκριμένη βάση δεδομένων είναι τόσο μικρή και τόσο απλή που τα πλεονεκτήματα δε γίνονται αμέσως φανερά. Φανταστείτε, όμως, την περίπτωση μιας παρόμοιας βάσης δεδομένων για ένα μεγάλο εστιατόριο, με κελάρι που περιέχει ίσως χιλιάδες φιάλες, στο οποίο γίνονται πολύ συχνά αλλαγές· ή φανταστείτε την περίπτωση ενός καταστήματος πώλησης ποτών, που έχει επίσης πολύ μεγάλο απόθεμα και υψηλό βαθμό αντικαταστάσεων σε αυτό το απόθεμα. (Βέβαια, και αυτές οι περιπτώσεις αφορούν συστήματα ενός χρήστη, παρόλο το μέγεθος της βάσης δεδομένων.) Τα πλεονεκτήματα ενός συστήματος βάσης δεδομένων, σε σύγκριση με τις παραδοσιακές μεθόδους παρακολούθησης αποθεμάτων με χαρτί και μολύβι, είναι μάλλον φανερά σε αυτά τα παραδείγματα. Ας δούμε μερικά από αυτά τα πλεονεκτήματα:

- ***Οικονομία χώρου:*** Δεν υπάρχει ανάγκη για ογκώδη παραδοσιακά αρχεία με φακέλους και έγγραφα.

- ***Ταχύτητα.*** Το μηχάνημα μπορεί να ανακαλεί και να αλλάζει τα δεδομένα πολύ πιο γρήγορα από τον άνθρωπο. Ειδικότερα, τα περιστασιακά ερωτήματα της στιγμής (για παράδειγμα, "'Εχουμε περισσότερες φιάλες Zinfandel από ό,τι Pinot Noir;") μπορούν να απαντηθούν γρήγορα, χωρίς να χρειάζονται χρονοβόρες χειρωνακτικές ή οπτικές αναζητήσεις.

- ***Λιγότερος κόπος:*** Καταργείται σε μεγάλο βαθμό ο "μπελάς" της τήρησης αρχείων "με το χέρι". Οι μηχανικές εργασίες γίνονται πάντοτε καλύτερα από τα μηχανήματα.

- ***'Αμεση πληροφόρηση:*** Ακριβείς και ενημερωμένες πληροφορίες είναι διαθέσιμες ανά πάσα στιγμή, αμέσως μόλις ζητηθούν.

Τα παραπάνω πλεονεκτήματα είναι βέβαια ακόμη μεγαλύτερα σε ένα περιβάλλον πολλών χρηστών, όπου η βάση δεδομένων είναι πολύ πιθανό να είναι πολύ μεγαλύτερη και πολύ πιο σύνθετη από ό,τι στην περίπτωση του συστήματος ενός χρήστη. Υπάρχει όμως και ένα ακόμα πλεονέκτημα σε ένα τέτοιο περιβάλλον: *Το σύστημα βάσης δεδομένων παρέχει στην επιχείρηση κεντρικό έλεγχο των* δεδομένων της (τα οποία, όπως είδαμε στην Ενότητα 1.3, είναι ένα από τα πολυτιμότερα περιουσιακά της στοιχεία). Αυτό έρχεται σε έντονη αντίθεση με την περίπτωση μιας επιχείρησης που δε χρησιμοποιεί σύστημα βάσης δεδομένων, όπου κατά κανόνα κάθε εφαρμογή έχει δικά της αρχεία — και αρκετά συχνά, δικές της ταινίες και δίσκους — με αποτέλεσμα τα δεδομένα να είναι διάσπαρτα σε πολύ μεγάλο βαθμό, και έτσι να δυσχεραίνεται ο έλεγχός τους με οποιονδήποτε συστηματικό τρόπο.

Εποπτεία διαχείρισης δεδομένων και βάσεων δεδομένων

Ας σταθούμε λίγο περισσότερο στην έννοια του κεντρικού ελέγχου. Αυτή η έννοια υπονοεί ότι (σε μια επιχείρηση που χρησιμοποιεί σύστημα βάσης δεδομένων) θα υπάρχει κάποιος άνθρωπος που θα έχει την κεντρική ευθύνη για τα δεδομένα. Αυτό το άτομο είναι ο **υπεύθυνος διαχείρισης δεδομένων** (data administrator, DA) στον οποίο αναφερθήκαμε με συντομία στο τέλος της Ενότητας 1.2. Εφόσον (όπως είδαμε παραπάνω) τα δεδομένα είναι ένα από τα πολυτιμότερα περιουσιακά στοιχεία της επιχείρησης, είναι απαραίτητο να υπάρχει κάποιος άνθρωπος που θα κατανοεί αυτά τα δεδομένα και τις ανάγκες της επιχείρησης σε ό,τι αφορά τα δεδομένα, και αυτό το άτομο να βρίσκεται σε κάποιο ανώτερο διοικητικό επίπεδο. Το άτομο αυτό είναι ο υπεύθυνος διαχείρισης δεδομένων. Έτσι, είναι δουλειά του υπεύθυνου διαχείρισης δεδομένων να αποφασίσει εξαρχής ποια δεδομένα θα πρέπει να αποθηκευτούν στη βάση δεδομένων και να ορίσει την πολιτική για την τήρηση και το χειρισμό αυτών των δεδομένων από τη στιγμή που θα αποθηκευτούν. Ένα παράδειγμα τέτοιας πολιτικής είναι να καθοριστεί ποια άτομα επιτρέπεται να εκτελούν ποιες πράξεις σε ποια δεδομένα και κάτω από ποιες περιστάσεις — με άλλα λόγια, μια πολιτική για την *ασφάλεια* των δεδομένων (data security):

Θα πρέπει να δοθεί ιδιαίτερη έμφαση στο γεγονός ότι ο υπεύθυνος διαχείρισης δεδομένων είναι διοικητικό στέλεχος και όχι Τεχνικός (αν και θα πρέπει ασφαλώς να έχει κάποια ιδέα για τις δυνατότητες των συστημάτων βάσεων δεδομένων σε τεχνικό επίπεδο). **Ο** τεχνικός *που* έχει την ευθύνη για την υλοποίηση των αποφάσεων του υπεύθυνου διαχείρισης δεδομένων είναι ο **υπεύθυνος διαχείρισης βάσεων δεδομένων** (database administrator, DBA). Έτσι, ο DBA, αντίθετα από τον υπεύθυνο διαχείρισης δεδομένων, πρέπει να είναι κάποιος *επαγγελματίας* της *τεχνολογίας* πληροφοριών. Δουλειά του DBA είναι να δημιουργήσει την ίδια τη βάση δεδομένων και να υλοποιήσει τους τεχνικούς ελέγχους που είναι απαραίτητοι για να επιβάλλονται οι διάφορες αποφάσεις πολιτικής που έχει πάρει ο υπεύθυνος διαχείρισης δεδομένων.Ο DBA έχει επίσης την ευθύνη να εξασφαλίζει την αποδοτική λειτουργία του συστήματος και να παρέχει διάφορες άλλες σχετικές τεχνικές υπηρεσίες. **Ο** DBA πλαισιώνεται συνήθως από μερικούς προγραμματιστές συστημάτων και άλλους τεχνικούς (δηλαδή, η δουλειά του DBA στην πράξη γίνεται κατά κανόνα από μια πολυμελή ομάδα και όχι από ένα μόνο άτομο)· για απλοποίηση όμως, είναι βολικό να θεωρούμε ότι ο DBA είναι ένα μόνο άτομο.

Πλεονεκτήματα της χρήσης βάσεων δεδομένων

Θα ολοκληρώσουμε αυτή την ενότητα παραθέτοντας τα συγκεκριμένα πλεονεκτήματα που προκύπτουν από τον κεντρικό έλεγχο των δεδομένων.

- Ο πλεονασμός (redundancy) μπορεί να μειωθεί στο ελάχιστο.

  Στα συμβατικά συστήματα (εκείνα που δεν είναι συστήματα βάσεων δεδομένων), η κάθε εφαρμογή έχει τα δικά της αρχεία. Αυτό το γεγονός οδηγεί πολύ συχνά σε υψηλό βαθμό πλεονασμού (επανάληψης) για τα αποθηκευμένα δεδομένα, με αποτέλεσμα τη σπατάλη αποθηκευτικού χώρου. Για παράδειγμα, μια εφαρμογή παρακολούθησης προσωπικού και μια εφαρμογή καταγραφής εκπαιδευτικών δραστηριοτήτων μπορεί να έχουν από ένα αρχείο με πληροφορίες που αφορούν τα τμήματα της επιχείρησης στα οποία ανήκουν οι υπάλληλοι. Όπως είδαμε στην Ενότητα 1.2, αυτά τα δύο αρχεία μπορούν να ενοποιηθούν ώστε να εξαλειφθεί ο πλεονασμός, αν ο υπεύθυνος διαχείρισης δεδομένων γνωρίζει τις απαιτήσεις που έχουν σε ό,τι αφορά τα δεδομένα οι δύο εφαρμογές — δηλαδή, αν η επιχείρηση έχει τον απαραίτητο γενικό έλεγχο.

  Θα πρέπει εδώ να ξεκαθαρίσουμε ότι αυτό δε σημαίνει πως είναι πάντα δυνατό να εξαλειφθούν *όλοι* οι πλεονασμοί, ούτε πως είναι πάντα επιθυμητό. Μερικές φορές υπάρχουν σοβαροί επιχειρηματικοί ή τεχνικοί λόγοι που επιβάλλουν να τηρούνται ξεχωριστά αντίγραφα των ίδιων αποθηκευμένων δεδομένων. Είναι όμως απαραίτητο αυτός ο πλεονασμός να είναι καλά *ελεγχόμενος* — δηλαδή, το DBMS θα πρέπει να είναι ενήμερο γι' αυτή την κατάσταση, αν υπάρχει, και θα πρέπει να έχει την ευθύνη για τις "διαδιδόμενες ενημερώσεις" δεδομένων (propagating updates — δείτε στην επόμενη παρατήρηση).

- Η ασυνέπεια μπορεί να αποφευχθεί (ως ένα βαθμό).

  Στην ουσία, αυτό είναι συνέπεια της προηγούμενης παρατήρησης. Ας υποθέσουμε ότι ένα δεδομένο γεγονός του πραγματικού κόσμου — ας πούμε, το γεγονός ότι ο υπάλληλος Υ3 εργάζεται στο τμήμα Τ8 — αναπαρίσταται στην αποθηκευμένη βάση δεδομένων από δύο ξεχωριστές εγγραφές. Ας υποθέσουμε επίσης ότι το **DBMS** δεν είναι ενήμερο γι' αυτή τη "διπλοεγγραφή" (δηλαδή, ότι ο πλεονασμός δεν είναι ελεγχόμενος). Τότε, είναι βέβαιο ότι θα υπάρξουν περιπτώσεις που σι δύο καταχωρίσεις δε θα συμφωνούν μεταξύ τους — περιπτώσεις που η μία καταχώριση θα έχει ενημερωθεί ενώ η άλλη όχι. Σε αυτές τις περιπτώσεις, λέμε ότι η βάση δεδομένων είναι ασυνεπής (inconsistent). Είναι προφανές ότι μια ασυνεπής βάση δεδομένων μπορεί να δώσει στους χρήστες της λανθασμένες ή αντιφατικές πληροφορίες.

  Είναι επίσης ξεκάθαρο ότι, αν το συγκεκριμένο γεγονός αναπαρίσταται από μία μόνο καταχώριση (δηλαδή, αν δεν υπάρχει πλεονασμός), τέτοια ασυνέπεια δεν μπορεί να παρουσιαστεί. Εναλλακτικά, αν υπάρχει πλεονασμός, αλλά είναι ελεγχόμενος (δηλαδή, είναι γνωστός στο **DBMS**), το DBMS θα μπορεί να εγγυηθεί ότι η βάση δεδομένων δε θα είναι ποτέ

ασυνεπής στα μάτια τον χρήστη, εξασφαλίζοντας ότι κάθε αλλαγή που θα γίνεται σε οποιαδήποτε από τις δύο καταχωρίσεις θα γίνεται αυτόματα και στην άλλη. Αυτή η διαδικασία είναι γνωστή με το όνομα **διάδοση ενημερώσεων** (propagating updates) — όπου (όπως πάντα) ο όρος "ενημέρωση" θεωρείται ότι συμπεριλαμβάνει όλες τις πράξεις εισαγωγής, διαγραφής, και τροποποίησης δεδομένων. Σημειώστε πάντως ότι ελάχιστα είναι τα συστήματα που κυκλοφορούν σήμερα στην αγορά, τα οποία έχουν τη δυνατότητα να εκτελούν τέτοιες διαδιδόμενες ενημερώσεις αυτόματα, τα περισσότερα από τα σημερινά προϊόντα δεν υποστηρίζουν τον ελεγχόμενο πλεονασμό με κανέναν τρόπο, εκτός ίσως από μερικές ειδικές περιστάσεις.

- Τα δεδομένα μπορούν να είναι κοινόχρηστα.

  Εξετάσαμε αυτό το ζήτημα στην Ενότητα 1.2, αλλά για λόγους πληρότητας το αναφέρουμε και εδώ. Ο μερισμός (sharing — κοινή χρήση) δε σημαίνει μόνο ότι οι υπάρχουσες εφαρμογές μπορούν να μοιράζονται τα δεδομένα της βάσης δεδομένων αλλά και ότι είναι δυνατή η ανάπτυξη νέων εφαρμογών που θα μπορούν να χρησιμοποιούν τα ίδια αποθηκευμένα δεδομένα. Με άλλα λόγια, είναι δυνατό να ικανοποιούνται οι απαιτήσεις νέων εφαρμογών σε ό,τι αφορά τα δεδομένα. χωρίς να είναι αναγκαία η δημιουργία νέων αποθηκευμένων δεδομένων.

- Μπορούν να επιβάλλονται πρότυπα.

  Με τον κεντρικό έλεγχο της βάσης δεδομένων, ο υπεύθυνος διαχείρισης βάσεων δεδομένων (DBA) — κάτω από την επίβλεψη του υπεύθυνου διαχείρισης δεδομένων — μπορεί να εξασφαλίσει ότι θα τηρούνται όλα τα σχετικά πρότυπα για την αναπαράσταση των δεδομένων. Τα πρότυπα αυτά μπορεί να είναι πρότυπα της επιχείρησης, του εγκατεστημένου συστήματος, του τμήματος της επιχείρησης, του τεχνολογικού τομέα, ή εθνικά και διεθνή πρότυπα. Η τυποποίηση της αναπαράστασης των δεδομένων διευκολύνει ιδιαίτερα την *ανταλλαγή δεδομένων* (data interchange), δηλαδή τη μεταφορά δεδομένων μεταξύ διαφορετικών συστημάτων (αυτό το θέμα αποκτά ιδιαίτερη σημασία με την έλευση της τεχνολογίας της κατανεμημένης επεξεργασίας (distributed processing). Τα πρότυπα ονομασίας και τεκμηρίωσης των δεδομένων είναι επίσης πολύ επιθυμητά για να διευκολύνεται ο μερισμός και η καλύτερη κατανόηση των δεδομένων.

- Μπορούν να εφαρμόζονται περιορισμοί ασφάλειας.

  'Εχοντας πλήρη δικαιοδοσία πάνω στη βάση δεδομένων, ο DBA (α) μπορεί να εξασφαλίσει ότι η πρόσβαση στη βάση δεδομένων θα μπορεί να γίνεται μόνο μέσω των κατάλληλων καναλιών και, κατά συνέπεια, (β) μπορεί να ορίσει κανόνες ασφάλειας με βάση τους οποίους θα γίνεται έλεγχος κάθε φορά που θα υπάρχει απόπειρα προσπέλασης εμπιστευτικών δεδομένων (και πάλι, σύμφωνα με τις οδηγίες του υπεύθυνου διαχείρισης δεδομένων). Είναι δυνατό να καθοριστούν διαφορετικοί κανόνες για το κάθε είδος προσπέλασης (ανάκληση, εισαγωγή, διαγραφή, κ.λπ.) σε κάθε στοιχείο πληροφοριών της βάσης δεδομένων. Σημειώστε, όμως, ότι χωρίς τέτοιους κανόνες η ασφάλεια των δεδομένων μπορεί να διατρέχει μεγαλύτερο κίνδυνο από ό,τι σε ένα

παραδοσιακό σύστημα αρχειοθέτησης (με γεωγραφικά απομακρυσμένα αρχεία)· δηλαδή, η κεντρική φύση ενός συστήματος βάσης δεδομένων απαιτεί κατά κάποιον τρόπο την ύπαρξη ενός καλού συστήματος ασφάλειας.

- Μπορεί να διατηρείται η ακεραιότητα.

  Το πρόβλημα της ακεραιότητας (integrity) είναι να εξασφαλίζεται ότι τα δεδομένα της βάσης δεδομένων είναι ακριβή. Η ασυμφωνία μεταξύ δύο καταχωρίσεων που υποτίθεται ότι αντιπροσωπεύουν το ίδιο "γεγονός" είναι ένα παράδειγμα έλλειψης ακεραιότητας· φυσικά, αυτό το συγκεκριμένο πρόβλημα μπορεί να παρουσιαστεί μόνο αν υπάρχει πλεονασμός στα αποθηκευμένα δεδομένα. Ακόμη και αν δεν υπάρχει πλεονασμός όμως, πάλι υπάρχει περίπτωση να περιέχει η βάση δεδομένων λανθασμένες πληροφορίες. Για παράδειγμα, ένας υπάλληλος μπορεί να εμφανίζεται ότι έχει δουλέψει 400 ώρες μέσα σε μία εβδομάδα, αντί για 40, ή ότι ανήκει στο τμήμα Τ9 ενώ δεν υπάρχει τέτοιο τμήμα. Ο κεντρικός έλεγχος της βάσης δεδομένων μπορεί να βοηθήσει να αποφευχθούν τέτοιου είδους προβλήματα — στο βαθμό που είναι δυνατό να αποφευχθούν — επιτρέποντας στον υπεύθυνο διαχείρισης δεδομένων να ορίζει (και στον DBA να υλοποιεί) κανόνες ακεραιότητας με βάση τους οποίους θα γίνεται έλεγχος κάθε φορά που επιχειρείται κάποια πράξη ενημέρωσης. (Και πάλι χρησιμοποιούμε τον όρο "ενημέρωση" γενικά, για να καλύψουμε όλες τις πράξεις εισαγωγής, διαγραφής, και τροποποίησης δεδομένων.)

  Αξίζει να επισημάνουμε ότι η ακεραιότητα των δεδομένων έχει πολύ μεγαλύτερη σημασία σε ένα σύστημα βάσης δεδομένων πολλών χρηστών από ό,τι σε ένα περιβάλλον "ιδιωτικών αρχείων", ακριβώς επειδή η βάση δεδομένων είναι μεριζόμενη. Αυτό συμβαίνει γιατί, χωρίς τους κατάλληλους ελέγχους, μπορεί ένας χρήστης να ενημερώσει τη βάση δεδομένων με εσφαλμένο τρόπο, δημιουργώντας με αυτόν τον τρόπο λανθασμένα δεδομένα και "μολύνοντας" τους υπόλοιπους "αθώους" χρήστες με αυτά τα δεδομένα. Θα πρέπει επίσης να πούμε ότι τα περισσότερα προϊόντα βάσεων δεδομένων είναι μάλλον ανεπαρκή ως προς την υποστήριξη ελέγχων της ακεραιότητας, αν και έχουν γίνει τελευταία κάποιες βελτιώσεις σε αυτόν τον τομέα.

- Οι αντικρουόμενες απαιτήσεις μπορούν να εξισορροπούνται.

  Γνωρίζοντας τις συνολικές απαιτήσεις της επιχείρησης — σε αντιδιαστολή με τις απαιτήσεις των μεμονωμένων χρηστών — ο DBA (πάντα με τις οδηγίες του υπεύθυνου διαχείρισης δεδομένων) μπορεί να δομήσει το σύστημα με τέτοιον τρόπο ώστε να παρέχει γενικές υπηρεσίες που να είναι "βέλτιστες για την επιχείρηση". Για παράδειγμα, είναι δυνατό να επιλεχθεί μια αναπαράσταση των αποθηκευμένων δεδομένων που θα παρέχει γρήγορη πρόσβαση στις σημαντικότερες εφαρμογές (ίσως σε βάρος της απόδοσης ορισμένων άλλων εφαρμογών).

  Τα περισσότερα από τα πλεονεκτήματα που παραθέσαμε εδώ είναι ίσως αρκετά προφανή. Θα πρέπει όμως να προσθέσουμε στη λίστα ένα ακόμη σημείο, που ίσως δεν είναι και τόσο προφανές (αν και στην πραγματικότητα συνάγεται από μερικά από τα υπόλοιπα): *την ανεξαρτησία των δεδομένων*

(data independence). (Για να είμαστε ακριβείς, η ανεξαρτησία των δεδομένων είναι στόχος των συστημάτων βάσεων δεδομένων, και όχι απαραίτητα πλεονέκτημα.)

1.2.5 Ανεξαρτησία δεδομένων

Η έννοια της ανεξαρτησίας δεδομένων μπορεί να γίνει πιο εύκολα κατανοητή αν εξετάσουμε πρώτα το αντίθετό της. Οι εφαρμογές που υλοποιούνταν σε παλιότερα συστήματα είχαν την τάση να εξαρτώνται από τα δεδομένα (data-dependent). Αυτό σημαίνει ότι ο τρόπος με τον οποίο ήταν οργανωμένα τα δεδομένα στα μέσα δευτερεύουσας αποθήκευσης και η τεχνική που εφαρμοζόταν για την προσπέλασή τους καθορίζονταν από τις απαιτήσεις της συγκεκριμένης εφαρμογής, και — το σημαντικότερο — ότι η γνώση του συγκεκριμένου τρόπου οργάνωσης των δεδομένων και της συγκεκριμένης τεχνικής προσπέλασης ήταν ενσωματωμένη στη λογική και τον κώδικα της εφαρμογής

- ***Παράδειγμα:*** Ας υποθέσουμε ότι έχουμε μια εφαρμογή που επεξεργάζεται το αρχείο ΥΠΑΛΛΗΛΟΙ και ότι, για λόγους απόδοσης, έχει αποφασιστεί ότι το αρχείο θα αποθηκευτεί με ευρετήριο που βασίζεται στο πεδίο "όνομα υπαλλήλου". Σε ένα παλιότερο σύστημα, η εφαρμογή θα πρέπει κατά κανόνα να γνωρίζει ότι υπάρχει αυτό το ευρετήριο, καθώς και τη διάταξη του αρχείου όπως ορίζεται από το συγκεκριμένο ευρετήριο, και η εσωτερική δομή της εφαρμογής θα πρέπει να έχει δομηθεί πάνω σε αυτή τη γνώση. Ειδικότερα, η ακριβής μορφή που θα έχουν οι διάφορες διαδικασίες της εφαρμογής, για την προσπέλαση των δεδομένων και για τον έλεγχο των εξαιρέσεων, θα εξαρτώνται σε καθοριστικό βαθμό από τις λεπτομέρειες της διασύνδεσης με την οποία παρουσιάζεται στην εφαρμογή το λογισμικό διαχείρισης δεδομένων.

Λέμε ότι μια εφαρμογή σαν αυτή του παραδείγματος **εξαρτάται από τα δεδομένα** επειδή είναι αδύνατο να τροποποιηθεί η αποθηκευτική δομή (το πώς είναι αποθηκευμένα τα δεδομένα στο φυσικό μέσο) ή η τεχνική της προσπέλασης (το πώς προσπελάζονται τα δεδομένα) χωρίς να γίνουν σημαντικές τροποποιήσεις στην ίδια την εφαρμογή. Για παράδειγμα, δε θα ήταν δυνατό να αντικατασταθεί το ευρετηριασμένο αρχείο του παραπάνω παραδείγματος από κάποιο αρχείο με διευθύνσεις κατακερματισμού (hash-addressed file) χωρίς να γίνουν σημαντικές τροποποιήσεις στην εφαρμογή. Και το σημαντικότερο, τα τμήματα της εφαρμογής που θα έπρεπε να τροποποιηθούν σε μια τέτοια περίπτωση είναι ακριβώς εκείνα τα τμήματα που επικοινωνούν με το λογισμικό διαχείρισης των δεδομένων· οι δυσκολίες που παρουσιάζει αυτό δεν έχουν καμία σχέση με το πρόβλημα για το οποίο είχε γραφτεί η εφαρμογή — πρόκειται, δηλαδή, για δυσκολίες που προκαλούνται από τη φύση της διασύνδεσης της διαχείρισης των δεδομένων.

Σ' ένα σύστημα βάσης δεδομένων, οι εφαρμογές δεν πρέπει να εξαρτώνται από τα δεδομένα με κανέναν τρόπο, για δύο τουλάχιστον λόγους:

1. Οι διάφορες εφαρμογές πρέπει να βλέπουν διαφορετικές απόψεις των ίδιων δεδομένων. Για παράδειγμα, υποθέστε ότι πριν εγκαταστήσει η επιχείρηση την ενοποιημένη βάση δεδομένων της υπήρχαν δύο εφαρμογές, η ***A*** και η ***B***, που η κάθε μία είχε ένα ιδιωτικό αρχείο που περιλάμβανε το πεδίο "υπόλοιπο πελάτη". Υποθέστε, όμως, ότι η εφαρμογή ***A*** καταχώριζε σε αυτό το πεδίο δεκαδικούς αριθμούς ενώ η εφαρμογή ***B*** δυαδικούς. Υπάρχει η δυνατότητα να ενοποιηθούν αυτά τα δύο αρχεία και να εξαλειφθεί ο πλεονασμός, αρκεί το DBMS να έχει την ετοιμότητα και την ικανότητα να εκτελεί όλες τις απαραίτητες μετατροπές ανάμεσα στην αναπαράσταση που επιλέχθηκε για τα αποθηκευμένα δεδομένα (που μπορεί να είναι η δεκαδική ή η δυαδική ή κάποια άλλη μορφή) και τη μορφή στην οποία θέλει να βλέπει τα δεδομένα η κάθε εφαρμογή. Για παράδειγμα, αν αποφασιστεί να αποθηκεύονται οι πληροφορίες του συγκεκριμένου πεδίου σε δεκαδική μορφή, τότε κάθε προσπέλαση των δεδομένων από την εφαρμογή ***B*** θα απαιτεί τη μετατροπή τους προς ή από τη δυαδική μορφή. Αυτό είναι ένα μάλλον απλοϊκό παράδειγμα για τη διαφορά που μπορεί να υπάρχει σε ένα σύστημα βάσης δεδομένων ανάμεσα στο πώς βλέπει τα δεδομένα μια συγκεκριμένη εφαρμογή και το πώς είναι στην πραγματικότητα αποθηκευμένα τα δεδομένα. Αργότερα, θα δούμε πολλές άλλες πιθανές διαφορές.

2. Ο υπεύθυνος διαχείρισης βάσεων δεδομένων (DBA) πρέπει να έχει τη δυνατότητα να αλλάζει την αποθηκευτική δομή ή την τεχνική προσπέλασης, ώστε να ανταποκρίνεται στις μεταβαλλόμενες απαιτήσεις, χωρίς να χρειάζεται να τροποποιηθούν οι υπάρχουσες εφαρμογές. Για παράδειγμα, μπορεί να προστεθούν στη βάση δεδομένων νέα είδη δεδομένων, να υιοθετηθούν νέα πρότυπα, να αλλάξουν σι προτεραιότητες των εφαρμογών (και επομένως οι απαιτήσεις για τη σχετική απόδοσή τους), να εγκατασταθούν νέοι τύποι συσκευών αποθήκευσης, κ.ο.κ. Αν οι εφαρμογές εξαρτώνται από τα δεδομένα, αυτού του είδους οι αλλαγές οδηγούν κατά κανόνα σε αντίστοιχες αλλαγές στα προγράμματα, δεσμεύοντας με αυτόν τον τρόπο ανθρώπινο δυναμικό (προγραμματιστές) που αλλιώς θα ήταν διαθέσιμο για τη δημιουργία νέων εφαρμογών. Ακόμη και σήμερα πάντως, δεν είναι ασυνήθιστο να αναλώνεται ένα ποσοστό της τάξης του 25 τοις εκατό (ή και ακόμη μεγαλύτερο) του δυναμικού των προγραμματιστών σε τέτοιου είδους δραστηριότητες συντήρησης εφαρμογών — που σημαίνει σαφώς μια σπατάλη πολύτιμων πόρων.

Από τα προηγουμένα, προκύπτει ότι η ανεξαρτησία των δεδομένων είναι ένας βασικός στόχος για τα συστήματα βάσεων δεδομένων. Η ανεξαρτησία δεδομένων μπορεί να οριστεί ως **ανοσία των εφαρμογών στις αλλαγές της αποθηκευτικής δομής και της τεχνικής της προσπέλασης δεδομένων** — που σημαίνει, φυσικά, ότι οι συγκεκριμένες εφαρμογές δεν πρέπει να εξαρτώνται από καμία αποθηκευτική δομή ή τεχνική προσπέλασης.

Θα ξεκινήσουμε ορίζοντας τρεις έννοιες: το *αποθηκευμένο πεδίο, την αποθηκευμένη εγγραφή,* και το *αποθηκευμένο αρχείο.*

- Το **αποθηκευμένο πεδίο** (stored field) είναι η μικρότερη μονάδα αποθηκευμένων δεδομένων. Η βάση δεδομένων θα περιέχει κατά κανόνα πολλές **παρουσίες** (occurences ή instances) καθενός από τους πολλούς **τύπους** (types) αποθηκευμένων πεδίων. Για παράδειγμα, μια βάση δεδομένων που περιέχει πληροφορίες για εξαρτήματα θα έχει κατά πάσα πιθανότητα έναν τύπο αποθηκευμένου πεδίου που θα ονομάζεται "κωδικός εξαρτήματος", και θα υπάρχει μία παρουσία αυτού του αποθηκευμένου πεδίου για το κάθε είδος εξαρτήματος (βίδα, μεντεσές, καπάκι, κ.λπ.).

- Μια **αποθηκευμένη εγγραφή** (stored record) είναι μια συλλογή σχετικών μεταξύ τους αποθηκευμένων πεδίων. Και πάλι, κάνουμε διάκριση μεταξύ τύπου και παρουσίας. Μια **παρουσία** κάποιας αποθηκευμένης εγγραφής αποτελείται από μια ομάδα σχετικών μεταξύ τους παρουσιών αποθηκευμένων πεδίων. Για παράδειγμα, μια παρουσία αποθηκευμένης εγγραφής στη βάση δεδομένων των εξαρτημάτων θα μπορούσε να αποτελείται από μια παρουσία καθενός από τα παρακάτω αποθηκευμένα πεδία: κωδικός εξαρτήματος, όνομα εξαρτήματος, χρώμα εξαρτήματος, και βάρος εξαρτήματος. Γι' αυτόν το λόγο, λέμε ότι η βάση δεδομένων περιέχει πολλές παρουσίες του τύπου αποθηκευμένης εγγραφής "εξάρτημα" (και πάλι, υπάρχει μία παρουσία για το κάθε διαφορετικό είδος εξαρτήματος).

  Με την ευκαιρία, σημειώνουμε ότι συνηθίζεται να παραλείπονται οι προσδιορισμοί "τύπος" και "παρουσία", και η διάκρισή τους βασίζεται στα συμφραζόμενα. Αν και υπάρχει ένας μικρός κίνδυνος σύγχυσης, αυτή η πρακτική είναι πολύ βολική και θα την υιοθετούμε και εμείς .

- Τέλος, ένα **αποθηκευμένο αρχείο** (stored file) είναι η συλλογή όλων των παρουσιών ενός τύπου αποθηκευμένων εγγραφών. (Σημείωση: Αγνοούμε σκόπιμα την περίπτωση να περιέχει ένα αποθηκευμένο αρχείο περισσότερους από έναν τύπους αποθηκευμένων εγγραφών. Είναι άλλη μία παραδοχή που κάνουμε για λόγους απλούστευσης, που δεν επηρεάζει σοβαρά κανένα από τα ζητήματα που θα εξετάσουμε στη συνέχεια.)

Από την άλλη μεριά, στα συμβατικά συστήματα (εκείνα που δεν είναι συστήματα βάσεων δεδομένων), πολύ συχνά η λογική εγγραφή (logical record) μιας εφαρμογής ταυτίζεται με κάποια αντίστοιχη αποθηκευμένη εγγραφή. Όπως είδαμε όμως, αυτό δεν είναι απαραίτητο να συμβαίνει σε ένα σύστημα βάσης δεδομένων, επειδή ο υπεύθυνος διαχείρισης βάσεων δεδομένων (DBA) μπορεί να χρειαστεί να κάνει αλλαγές στην αποθηκευτική δομή — δηλαδή, στα αποθηκευμένα πεδία, εγγραφές, και αρχεία — χωρίς να μεταβληθεί η αντίστοιχη λογική εγγραφή. Για παράδειγμα, το πεδίο "βάρος εξαρτήματος" που αναφέραμε νωρίτερα μπορεί να αποθηκευτεί σε δυαδική μορφή για να εξοικονομηθεί αποθηκευτικός χώρος, ενώ μια εφαρμογή γραμμένη σε COBOL μπορεί να το βλέπει ως στοιχείο τύπου PICTURE (δηλαδή, ως αλφαριθμητικό). Και αργότερα, ο DBA μπορεί να αποφασίσει για κάποιο λόγο να αλλάξει την αποθηκευμένη αναπαράσταση αυτού του πεδίου από δυαδική σε δεκαδική και η εφαρμογή να μπορεί και πάλι να το βλέπει σε μορφή χαρακτήρων.

Όπως είδαμε παραπάνω, μια διαφορά σαν αυτή, όπου γίνεται μετατροπή του τύπου δεδομένων ενός συγκεκριμένου πεδίου σε κάθε προσπέλαση, είναι σχετικά μικρή· γενικά όμως, η διαφορά ανάμεσα σε αυτό που βλέπει η εφαρμογή και σε αυτό που είναι πραγματικά αποθηκευμένο μπορεί να είναι πολύ μεγάλη. Για να ενισχύσουμε αυτή την παρατήρηση, παρουσιάζουμε στη συνέχεια μια λίστα με διάφορα στοιχεία της αποθηκευτικής δομής που είναι δυνατό να μεταβάλλονται. Εξετάζουμε στην κάθε περίπτωση τι θα πρέπει να κάνει το DBMS για να προστατεύσει μια εφαρμογή από τέτοιου είδους αλλαγές (και αν αυτή η προστασία είναι πάντα εφικτή).

- Αναπαράσταση των αριθμητικών δεδομένων

  Ένα αριθμητικό πεδίο μπορεί να αποθηκεύεται σε κάποια εσωτερική αριθμητική μορφή (για παράδειγμα, σε συνεπτυγμένη δεκαδική μορφή — packed decimal) ή ως αλφαριθμητικό. Και στις δύο περιπτώσεις, ο DBA πρέπει να επιλέξει κατάλληλη αριθμητική βάση (για παράδειγμα, δυαδική ή δεκαδική), κλίμακα (σταθερή τιμή ή κινητής υποδιαστολής), είδος (πραγματική ή μιγαδική τιμή), και ακρίβεια (πλήθος ψηφίων). Οποιοδήποτε από αυτά τα στοιχεία μπορεί να τροποποιηθεί για να βελτιωθεί η απόδοση, για να συμμορφωθεί η εφαρμογή με κάποιο πρότυπο, ή για πολλούς άλλους λόγους.

- Αναπαράσταση των δεδομένων κειμένου (χαρακτήρων)

  Ένα αλφαριθμητικό πεδίο μπορεί να αποθηκευτεί με ένα από τα πολλά διαφορετικά κωδικοποιημένα συνόλα χαρακτήρων (character set) όπως το **ASCII**, το **EBCDIC**, κ.λπ.

- Μονάδες των αριθμητικών δεδομένων

  Οι μονάδες ενός αριθμητικού πεδίου μπορεί να αλλάξουν — για παράδειγμα, από ίντσες σε εκατοστά του μέτρου, κατά τη διάρκεια μιας διαδικασίας αλλαγής μετρικού συστήματος.

- Κωδικοποίηση των δεδομένων

  Σε ορισμένες περιπτώσεις μπορεί να είναι επιθυμητή η αναπαράσταση των αποθηκευμένων δεδομένων με κωδικοποιημένες τιμές. Για παράδειγμα, το πεδίο "χρώμα εξαρτήματος", το οποίο μια εφαρμογή το βλέπει ως αλφαριθμητικό ("Κόκκινο", "Μπλε", "Πράσινο" κ.λπ.), μπορεί να αποθηκεύεται ως ένα δεκαδικό ψηφίο που ερμηνεύεται κάθε φορά σύμφωνα με το μηχανισμό κωδικοποίησης 1 = "Κόκκινο", 2 = "Μπλε", κ.ο.κ

- Πραγμάτωση των δεδομένων

Στην πράξη, τα λογικά πεδία που βλέπει μια εφαρμογή αντιστοιχούν συνήθως σε κάποια συγκεκριμένα αποθηκευμένα πεδία (αν και, όπως είδαμε, μπορεί να υπάρχουν διαφορές στον τύπο των δεδομένων, τις μονάδες μέτρησης, κ.ο.κ.). Σε μια τέτοια περίπτωση, η διαδικασία πραγμάτωσης (materialization) — δηλαδή, η κατασκευή μιας παρουσίας του λογικού πεδίου από την αντίστοιχη

παρουσία του αποθηκευμένου πεδίου, για να χρησιμοποιηθεί από την εφαρμογή — μπορεί να θεωρηθεί ότι είναι άμεση. Μερικές φορές όμως, ένα λογικό πεδίο δεν έχει κάποιο αντίστοιχο αποθηκευμένο πεδίο, αλλά οι τιμές του πραγματώνονται με κάποιον υπολογισμό που εκτελείται σε πολλές παρουσίες αποθηκευμένων πεδίων. Για παράδειγμα, οι τιμές του λογικού πεδίου "συνολική ποσότητα" μπορεί να πραγματώνονται με την άθροιση πολλών μεμονωμένων αποθηκευμένων τιμών ποσοτήτων. Εδώ, το πεδίο "συνολική ποσότητα" είναι ένα παράδειγμα εικονικού πεδίου (virtual field) και η διαδικασία πραγμάτωσής του είναι έμμεση. Σημειώστε, πάντως, ότι ο χρήστης μπορεί να διαπιστώσει ότι υπάρχει διαφορά ανάμεσα στα πραγματικά και τα εικονικά πεδία, από το γεγονός ότι είναι πιθανό να μην μπορεί (άμεσα) να εισαγάγει ή να τροποποιήσει μια παρουσία εικονικού πεδίου.

- Δομή των αποθηκευμένων εγγραφών

Δύο υπάρχουσες αποθηκευμένες εγγραφές θα μπορούσαν να συνδυαστούν σε μία. Για παράδειγμα, οι αποθηκευμένες εγγραφές

Κωδικός Εξαρτήματος	Χρώμα

Και

Κωδικός Εξαρτήματος	Βάρος

Θα μπορούσαν να συνδυαστούν για να σχηματίσουν την εγγραφή

Κωδικός Εξαρτήματος	Χρώμα	Βάρος

Μια τέτοια αλλαγή θα μπορούσε να γίνει σε περίπτωση που αποφασιστεί να ενταχθούν στο σύστημα βάσης δεδομένων κάποιες ξένες εφαρμογές. Η αλλαγή αυτή σημαίνει, βέβαια, ότι η λογική εγγραφή μιας εφαρμογής μπορεί να είναι υποσύνολο της αντίστοιχης αποθηκευμένης εγγραφής — δηλαδή, ορισμένα από τα πεδία αυτής της αποθηκευμένης εγγραφής να είναι αόρατα στη συγκεκριμένη εφαρμογή.

Εναλλακτικά, ένας τύπος αποθηκευμένης εγγραφής θα μπορούσε να χωριστεί στα δύο. Αν αντιστρέψουμε το προηγούμενο παράδειγμα, ο τύπος αποθηκευμένης εγγραφής

Κωδικός Εξαρτήματος	Χρώμα	Βάρος

Θα μπορούσε να χωριστεί στα εξής:

Και

Μια τέτοια υποδιαίρεση θα μπορούσε, για παράδειγμα, να επιτρέψει την αποθήκευση κάποιων τμημάτων της αρχικής εγγραφής, που δε χρησιμοποιούνται και τόσο συχνά, σε μια πιο αργή συσκευή αποθήκευσης. Αυτό σημαίνει ότι η λογική εγγραφή μιας εφαρμογής θα μπορούσε να περιέχει πεδία από πολλές διαφορετικές αποθηκευμένες εγγραφές — δηλαδή, να είναι υπερσύνολο οποιασδήποτε από αυτές τις αποθηκευμένες εγγραφές.

- Δομή των αποθηκευμένων αρχείων

  Ένα αποθηκευμένο αρχείο μπορεί να υλοποιηθεί στο φυσικό μέσο αποθήκευσης με πολλούς διαφορετικούς τρόπους. Για παράδειγμα, μπορεί να περιέχεται ολόκληρο σε μία μόνο μονάδα αποθήκευσης (δηλαδή, σε ένα μόνο δίσκο) ή να είναι διαμοιρασμένο σε πολλές συσκευές αποθήκευσης διαφορετικών τύπων· οι αποθηκευμένες εγγραφές του μπορεί να είναι ή να μην είναι ταξινομημένες με τη φυσική σειρά, σύμφωνα με τις τιμές ενός αποθηκευμένου πεδίου· μπορεί να είναι ή να μην είναι ταξινομημένο με έναν ή περισσότερους άλλους τρόπους (για παράδειγμα, σύμφωνα με ένα ή περισσότερα ευρετήρια, ή με μία ή περισσότερες ενσωματωμένες αλυσίδες δεικτών, ή και με τους δύο αυτούς τρόπους)· μπορεί να είναι ή να μην είναι προσπελάσιμο μέσω διευθύνσεων κατακερματισμού (hash-addressing)· οι αποθηκευμένες εγγραφές του μπορεί να είναι ή να μην είναι φυσικά οργανωμένες σε μπλοκ (δηλαδή, η κάθε φυσική εγγραφή να περιέχει περισσότερες από μία αποθηκευμένες εγγραφές), κ.ο.κ. Όμως, κανένα από αυτά τα ζητήματα δε θα πρέπει να επηρεάζει τις εφαρμογές με κανέναν τρόπο (εκτός, φυσικά, από την απόδοσή τους).

Εδώ ολοκληρώνεται η λίστα μας με τα στοιχεία της αποθηκευτικής δομής τα οποία είναι δυνατό να μεταβάλλονται. Αυτή η λίστα σημαίνει (μεταξύ άλλων) ότι η βάση δεδομένων θα πρέπει να έχει τη δυνατότητα να **αναπτύσσεται,** χωρίς να επηρεάζονται σι υπάρχουσες εφαρμογές· μάλιστα, αυτή η δυνατότητα της βάσης δεδομένων να αναπτύσσεται χωρίς να επηρεάζονται λογικά οι υπάρχουσες εφαρμογές είναι ίσως ο πιο βασικός λόγος που απαιτείται η ανεξαρτησία των δεδομένων. Για παράδειγμα, θα πρέπει να υπάρχει η δυνατότητα να επεκταθεί μια υπάρχουσα αποθηκευμένη εγγραφή, με την προσθήκη νέων αποθηκευμένων πεδίων που θα αντιπροσωπεύουν κάποιες πρόσθετες πληροφορίες για έναν υπάρχοντα τύπο οντότητας (για παράδειγμα, στην αποθηκευμένη εγγραφή "εξάρτημα" θα μπορούσε να προστεθεί το αποθηκευμένο πεδίο "κόστος μονάδας"). Αυτά τα νέα πεδία θα πρέπει να είναι αόρατα στις υπάρχουσες εφαρμογές. Με ανάλογο τρόπο, θα πρέπει να υπάρχει η δυνατότητα να προστίθενται εντελώς νέοι τύποι αποθηκευμένων εγγραφών (και νέων αποθηκειμένων αρχείων), και πάλι χωρίς να χρειάζεται να τροποποιηθούν οι εφαρμογές που υπάρχουν ήδη.Αυτές οι εγγραφές θα πρέπει να αντιπροσωπεύουν νέους τύπους οντοτήτων (για παράδειγμα στη *βάση δεδομένων των* εξαρτημάτων θα

μπορούσε να προστεθεί ένας τύπος εγγραφών "προμηθετής")· Και πάλι, αυτές οι προσθήκες θα πρέπει να ειναι αόρατες στις υπάρχουσες εφαρμογές.

Η αναξαρτησία των δεδομένων δεν είναι κάτι απόλυτο — τα διάφορα συστήματα την παρέουν σε διαφορετικό βαθμό. Για να το θέσουμε διαφορετικά, ελάχιστα είναι τα συστήματα (για να μη πούμε κανένα) που δεν παρέχουν κάποιου είδους ανεξαρτησία δεδομένων,απλώς ορισμένα συστήματα παράχουν μικρότερη ανεξαρτησία δεδομένων από κάποια άλλα.Τα σημερινά συστήματα τείνουν να παρεχουν μεγαλύτερη ανεξαρτησία δεδομένων από τα παλιότερα αλλά εξακολουθούν να απέχουν από το τέλειο.

1.2.6 Σχεσιακά και μη σχεσιακά συστήματα

Όλα σχεδόν τα προϊόντα διαχείρισης βάσεων δεδομένων που αναπτύχθηκαν από τα τέλη της δεκαετίας του 1970 και μετά βασίζονται στη λεγόμενη **σχεσιακή προσέγγιση** (relational approach)· ακόμα, η συντριπτική πλειοψηφία των ερευνών που έγιναν για τις βάσεις δεδομένων τα τελευταία 25 χρόνια βασίζονται επίσης σε αυτή την προσέγγιση — αν και έμμεσα σε ορισμένες περιπτώσεις. Είναι γεγονός ότι η σχεσιακή προσέγγιση αποτελεί σήμερα την κυρίαρχη τάση στην αγορά, και ότι το "σχεσιακό μοντέλο" είναι το σημαντικότερο επίτευγμα σε ολόκληρη την ιστορία των βάσεων δεδομένων. Για όλους αυτούς τους λόγους, και για τον πρόσθετο λόγο ότι το σχεσιακό μοντέλο έχει στερεές μαθηματικές βάσεις και, επομένως, είναι ένα ιδανικό μέσο για την διδασκαλία των εννοιών και των αρχών των συστημάτων βάσεων δεδομένων.

Τι εννοούμε, λοιπόν, όταν λέμε ότι ένα σύστημα είναι σχεσιακό; Δυστυχώς, δεν είναι δυνατό να απαντήσουμε πλήρως αυτό το ερώτημα σε αυτό το αρχικό στάδιο· ωστόσο, είναι δυνατό αλλά και επιθυμητό να δώσουμε μια πρόχειρη απάντηση που θα τη βελτιώσουμε και θα την ολοκληρώσουμε αργότερα. Με δύο λόγια, λοιπόν, σχεσιακό είναι ένα σύστημα που:

1. Ο χρήστης αντιλαμβάνεται τα δεδομένα ως πίνακες (και μόνο ως πίνακες).

2. Οι τελεστές πράξεων που έχει στη διάθεσή του ο χρήστης (για παράδειγμα, για την ανάκληση δεδομένων) είναι τελεστές που δημιουργούν νέους πίνακες με βάση κάποιους παλιότερους. Για παράδειγμα, υπάρχει ένας τελεστής για την εξαγωγή ενός υποσυνόλου των γραμμών ενός πίνακα, και ένας άλλος τελεστής για την εξαγωγή ενός υποσυνόλου των στηλών του — και φυσικά, ένα υποσύνολο των γραμμών και ένα υποσύνολο των στηλών ενός πίνακα μπορούν και τα δύο να θεωρηθούν επίσης πίνακες.

Ο λόγος για τον οποίο ονομάζονται αυτά τα συστήματα "σχεσιακά" (relational) είναι ότι ο όρος "σχέση" (relation) είναι ουσιαστικά ένα όρος των μαθηματικών που σημαίνει "πίνακας". Πρακτικά μάλιστα, στις περισσότερες οι όροι "σχέση" και "πίνακας" μπορούν να θεωρηθούν συνώνυμοι.

Θα διατυπώσουμε πολύ πιο συγκεκριμένα αυτόν τον ορισμό αργότερα· για την ώρα πάντως, μας αρκεί ο ορισμός που δώσαμε πιο πάνω. Στην Εικόνα 1.8 μπορείτε να

δείτε ένα παράδειγμα. Τα δεδομένα — δείτε στο τμήμα (α) της εικόνας — αποτελούνται από ένα μόνο πίνακα, που ονομάζεται **ΚΑΒΑ** (στην πράξη πρόκειται για μια συντομευμένη παραλλαγή του πίνακα **ΚΑΒΑ** που είδαμε στην Εικόνα 1.1, μικρότερη σε μέγεθος, για λόγους ευκολίας). Στο τμήμα (β) της εικόνας παρουσιάζονται ενδεικτικά δύο ανακλήσεις δεδομένων — στη μία γίνεται μια πράξη εξαγωγής υποσυνόλου γραμμών και στην άλλη μια πράξη εξαγωγής υποσυνόλου στηλών. Σημείωση: Οι δύο ανακλήσεις δεδομένων είναι στην πραγματικότητα παραδείγματα χρήσης της εντολής **SELECT** της γλώσσας **SQL** που συναντήσαμε για πρώτη φορά στην Ενότητα 1.1.

α) *Δεδομένος πίνακας:* ΚΑΒΑ

ΚΡΑΣΙ	ΕΤΟΣ	**ΦΙΑΛΕΣ**
Chardonnay	91	4
Fume Blanc	91	2
Pinet Noir	88	3
Zinfandel	89	9

β) Τελεστές (παραδείγματα):
1. Υποσύνολο γραμμών: Αποτέλεσμα:

```
SELECT    ΚΡΑΣΙ, ΕΤΟΣ, ΦΙΑΛΕΣ
FROM       ΚΑΒΑ
WHERE    ΕΤΟΣ > 90 ;
```

ΚΡΑΣΙ	ΕΤΟΣ	**ΦΙΑΛΕΣ**
Chardonnay	91	4
Fume Blanc	91	2

2. *Υποσύνολο* στηλών: Αποτέλεσμα:

```
SELECT    ΚΡΑΣΙ, ΦΙΑΛΕΣ
FROM       ΚΑΒΑ ;
```

ΚΡΑΣΙ	**ΦΙΑΛΕΣ**
Chardonnay	4
Fume Blanc	2
Pinet Noir	3
Zinfandel	9

ΕΙΚ. 1.8 Δομή δεδομένων και τελεστές σε ένα σχεσιακό σύστημα (παραδείγματα)

Μπορούμε τώρα να κάνουμε την εξής διάκριση ανάμεσα στα σχεσιακά και τα μη σχεσιακά συστήματα: Όπως είδαμε, ο χρήστης ενός σχεσιακού συστήματος βλέπει τα δεδομένα ως πίνακες και μόνο ως πίνακες. Αντίθετα, ο χρήστης ενός μη σχεσιακού συστήματος βλέπει και άλλες δομές δεδομένων, αντί για τους πίνακες ενός σχεσιακού συστήματος ή εκτός από αυτούς. Αυτές οι άλλες δομές, με τη σειρά τους, χρειάζονται άλλους τελεστές πράξεων για το χειρισμό των δεδομένων τους. Για παράδειγμα, σε ένα **ιεραρχικό** σύστημα (hierarchic system) τα δεδομένα παρουσιάζονται στο χρήστη με τη μορφή ενός συνόλου δενδροειδών δομών (ιεραρχιών), και στους τελεστές για το χειρισμό αυτών των δομών περιλαμβάνονται τελεστές για να διατρέχονται οι ιεραρχικές διαδρομές προς την κορυφή ή προς τη βάση των δένδρων.

Ας σταθούμε σε αυτό το ζήτημα λίγο περισσότερο: Τα συστήματα βάσεων δεδομένων μπορούν να καταταχθούν σε κατηγορίες, σύμφωνα με τις δομές δεδομένων και τους τελεστές που παρέχουν στο χρήστη. Τα παλιότερα συστήματα (τα προ-σχεσιακά) χωρίζονται σε τρεις μεγάλες κατηγορίες: τα συστήματα **αντιστραμμένης λίστας** (inverted list system), τα **ιεραρχικά** συστήματα (hierarchical system), και τα **δικτυωτά** συστήματα (network system). Παραδείγματα προϊόντων του εμπορίου που ανήκουν σε αυτές τις τρεις κατηγορίες είναι τα παρακάτω:

Συστήματα αντιστραμμένης λίστας: CA-DATACOM/ DB της Computer Associates International Inc. (παλιότερα γνωστό ως DATACOM/DB της Applied Data Research)

Ιεραρχικά συστήματα: IMS της IBM Corporation

Δικτυωτά συστήματα: CA-IDMS/DB της Computer Associates International Inc. (παλιότερα γνωστό ως IDMS της Cullinet Software Imc.)

Τα πρώτα σχεσιακά συστήματα άρχισαν να εμφανίζονται στην αγορά στα τέλη της δεκαετίας του 1970 και στις αρχές της δεκαετίας του 1980. Παραδείγματα τέτοιων προϊόντων είναι το **DB2** της IBM Corporation, το Rdb/VMS της Digital Equipment Corporation, το ORACLE της Oracle Corporation, το INGRES της Ingres Division of The ASK Group Inc., το SYBASE της Sybase Inc. και πολλά άλλα.

Τα τελευταία χρόνια, έχουν γίνει έρευνες σε μια μεγάλη ποικιλία συστημάτων που θα μπορούσαν να χαρακτηριστούν "μετα-σχεσιακά", ορισμένα από τα οποία βασίζονται σε "συμβατές προς τα πάνω" επεκτάσεις της αρχικής σχεσιακής προσέγγισης, ενώ άλλα βασίζονται σε απόπειρες να δημιουργηθεί κάτι εντελώς διαφορετικό. Θα περιοριστούμε σε μια απλή ονομαστική αναφορά ορισμένων από αυτές τις πιο σύγχρονες προσεγγίσεις:

- Συμπερασματικά **DBMS** (deductive **DBMS**)
- Εμπειρα **DBMS** (expert **DBMS**)

- Επεκτάσιμα **DBMS** (Extendable DBMS)
- Αντικειμενοστρεφή **DBMS** (Object-oriented DBMS)
- Σημασιολογικά **DBMS** (Semantic DBMS)
- **DBMS** καθολικής σχέσης (Universal relation DBMS)

Στην περίπτωση των **αντικειμενοστρεφών** (object-oriented) συστημάτων, έχουν εμφανιστεί ήδη μερικά συστήματα — όπως το GemStone της Servio Corporation, το ObjectStore της Object Design Corporation, και το OpenDB της Hewlett-Packard Corporation.

1.2.7 Περίληψη

Θα ολοκληρώσουμε αυτό το εισαγωγικό κεφάλαιο συνοψίζοντας τα κυριότερα ζητήματα που εξετάσαμε. Καταρχήν, **σύστημα βάσης δεδομένων** είναι ένα σύστημα τήρησης εγγραφών με υπολογιστή. 'Ενα τέτοιο σύστημα περιλαμβάνει τα ίδια τα **δεδομένα** (που είναι αποθηκευμένα σε μια **βάση δεδομένων**), το **υλικό**, το **λογισμικό** (και ειδικότερα, το **σύστημα διαχείρισης βάσεων δεδομένων** ή **DBMS**), και — το σημαντικότερο! — τους **χρήστες**. Οι χρήστες, με τη σειρά τους, μπορούν να υποδιαιρεθούν στους **προγραμματιστές εφαρμογών**, τους **τελικούς χρήστες**, και τον **υπεύθυνο διαχείρισης βάσεων δεδομένων (DBA)**. Ο DBA είναι υπεύθυνος για τη διαχείριση της βάσης δεδομένων και του συστήματος βάσης δεδομένων, σύμφωνα με την πολιτική που έχει καθορίσει ο **υπεύθυνος διαχείρισης δεδομένων**.

Οι βάσεις δεδομένων είναι **ενοποιημένες**, και συνήθως **μεριζόμενες** (κοινόχρηστες), και χρησιμοποιούνται για την αποθήκευση **μόνιμων** δεδομένων. Τα δεδομένα αυτά θα μπορούσαμε να θεωρήσουμε (κάπως άτυπα) ότι αντιπροσωπεύουν **οντότητες** καθώς και **συσχετίσεις** μεταξύ αυτών των οντοτήτων — αν και, στην πραγματικότητα, μια συσχέτιση δεν είναι παρά ένας ειδικός τύπος οντότητας. Επίσης, εξετάσαμε με συντομία την ιδέα του **διαγράμματος οντοτήτων/συσχετίσεων.**

Τα συστήματα βάσεων δεδομένων παρέχουν στους χρήστες τους πολλά πλεονεκτήματα, ένα από τα σημαντικότερα από τα οποία είναι η **ανεξαρτησία δεδομένων** (η ανοσία των εφαρμογών σε αλλαγές του τρόπου αποθήκευσης και προσπέλασης των δεδομένων).

Τέλος, τα συστήματα βάσεων δεδομένων μπορεί να βασίζονται σε πολλές διαφορετικές προσεγγίσεις, ανάμεσα στις οποίες και η **σχεσιακή προσέγγιση**. Τόσο από οικονομική όσο και από θεωρητική άποψη, η σχεσιακή προσέγγιση είναι σαφώς η σημαντικότερη (και αυτό δε φαίνεται να αλλάζει στο κοντινό μέλλον). Σε ένα σχεσιακό σύστημα, οι χρήστες βλέπουν τα δεδομένα σαν **πίνακες**, και οι τελεστές που είναι διαθέσιμοι στους χρήστες για να επενεργούν στα δεδομένα είναι τελεστές διαχείρισης πινάκων. Είδαμε μερικά απλά παραδείγματα της **SQL** της πρότυπης γλώσσας για τα σχεσιακά συστήματα.

1.3 Τι είναι η PHP;

1.3.1 Εισαγωγή

Η PHP, της οποίας τα αρχικά αντιπροσωπεύουν το "PHP: Hypertext Preprocessor" είναι μια ευρέως χρησιμοποιούμενη, ανοιχτού κώδικα, γενικού σκοπού scripting γλώσσα προγραμματισμού, η οποία είναι ειδικά κατάλληλη για ανάπτυξη εφαρμογών για το Web και μπορεί να ενσωματωθεί στην HTML.

Απλή απάντηση, αλλά τι σημαίνει; Ένα παράδειγμα:

Παράδειγμα 1-1. Ένα εισαγωγικό παράδειγμα

```
<html>
    <head>
        <title>Example</title>
    </head>
    <body>

        <?php
        echo "Hi, I'm a PHP script!";
        ?>

    </body>
</html>
```

Παρατηρήστε πως αυτό είναι διαφορετικό από ένα script γραμμένο σε άλλες γλώσσες προγραμματισμού όπως η Perl ή η C : Αντί να γράφουμε ένα πρόγραμμα με πολλές εντολές για να εξάγουμε HTML, γράφουμε ένα HTML script με κάποιο ενσωματωμένο κώδικα για να κάνει κάτι (σε αυτή την περίπτωση, να εμφανίζει κάποιο κείμενο). Ο κώδικας PHP είναι εσώκλειστος σε ειδικά tags (αρχές και τέλους) που μας επιτρέπουν να μεταφερόμαστε μέσα και έξω από το "PHP mode" (PHP τρόπο λειτουργίας).

Αυτό που διαχωρίζει την PHP από κάτι σαν client-side Javascript είναι ότι ο κώδικας εκτελείται στον server (εξηπηρετητή). Αν είχαμε ένα script σαν το παραπάνω στον server μας, ο client θα έπαιρνε τα αποτελέσματα της εκτέλεσης αυτού του script, χωρίς να υπάρχει κανένας τρόπος να καταλάβει τι κώδικας υπάρχει από κάτω. Μπορούμε ακόμη να ρυθμίσουμε τον web server μας να χειρίζεται όλα τα HTML αρχεία μας με την PHP, και τότε πραγματικά δεν υπάρχει τρόπος ο χρήστης να καταλάβει τι έχουμε κάτω από το μανίκι μας.

Τα καλύτερο πράγμα στην PHP είναι ότι είναι εξαιρετικά απλή για ένα νεοφερμένο αλλά προσφέρει πολλές προηγμένα χαρακτηριστικά για ένα επαγγελματία προγραμματιστή. Δεν πρέπει να τρομάζουμε όταν διαβάζουμε την μακροσκελή λίστα με τα χαρακτηριστικά της PHP. Μπορούμε να εξοικειωθούμε μέσα σε πολύ λίγο χρόνο και να αρχίσουμε να γράφουμε απλά script σε λίγες ώρες.

Αν και η ανάπτυξη της PHP εστιάζεται σε server-side scripting, μπορούμε να κάνουμε πολύ περισσότερα με αυτή. Διαβάστε παρακάτω και δείτε περισσότερα στην παράγραφο [illegible].

1.3.2 Τι μπορεί να κάνει η PHP;

Οτιδήποτε. Η PHP επικεντρώνεται κυρίως στο server-side scripting, έτσι μπορούμε να κάνουμε οτιδήποτε ένα άλλο CGI πρόγραμμα μπορεί να κάνει, όπως να μαζέψει δεδομένα, να παράγει δυναμικό περιεχόμενο σελίδων, ή να στείλει και να πάρει cookies. Αλλά η PHP μπορεί να κάνει πολύ περισσότερα.

Υπάρχουν τρεις κύριοι τομείς που χρησιμοποιείται ένα PHP script.

- Server-side scripting. Αυτό είναι το πιο παραδοσιακό και το κύριο πεδίο για την PHP. Χρειάζεστε τρία πράγματα για να δουλέψει αυτό. Τον PHP μεταγλωττιστή (parser) (CGI ή server module), ένα webserver (εξηπηρετητή σελίδων) και ένα web browser ("φυλλομετρητή"). Πρέπει να τρέξουμε τον webserver, με μια συνδεδεμένη εγκατάσταση της PHP. Μπορούμε να προσπελάσετε τα αποτελέσματα του PHP προγράμματος με ένα web browser, βλέποντας την σελίδα PHP μέσα από τον server. Για περισσότερες πληροφορίες, θα δούμε στην παράγραφο [illegible].
- Command line scripting. Μπορούμε να φτιάξουμε ένα PHP script για να το τρέχουμε χωρίς server ή browser. Χρειαζόμαστε μόνο τον PHP μεταγλωττιστή για να την χρησιμοποιήσουμε με αυτό τον τρόπο. Αυτός ο τύπος είναι ιδανικός για script που εκτελούνται συχνά με τη χρήση της cron (σε *nix ή Linux) ή με τον Task Scheduler (στα Windows). Αυτά τα script μπορούν επίσης να χρησιμοποιηθούν για απλές εργασίες επεξεργασίες κειμένου.
- Εγγραφή client-side GUI εφαρμογών (Γραφικά περιβάλλοντα χρηστών). Η PHP ίσως να μην είναι η πιο καλή γλώσσα για να γράψει κανείς παραθυριακές εφαρμογές, αλλά αν ξέρουμε PHP πολύ καλά και θέλουμε να χρησιμοποιήσουμε κάποια προχωρημένα χαρακτηριστικά της PHP στις client-side εφαρμογές μας, μπορούμε επίσης να χρησιμοποιήσουμε το PHP-GTK για αυτού του είδους τα προγράμματα. Έχουμε επίσης τη δυνατότητα να γράφουμε cross-platform εφαρμογές με αυτό τον τρόπο. Το PHP-GTK είναι μια επέκταση της PHP και δεν συμπεριλαμβάνεται στην κύρια διανομή. Αν ενδιαφέρεστε για το PHP-GTK, επισκεφτείτε [illegible].

Η PHP μπορεί να χρησιμοποιηθεί σε όλα τα κύρια λειτουργικά συστήματα, συμπεριλαμβανομένου του Linux, πολλών εκδοχών του Unix (HP-UX, Solaris και OpenBSD), Microsoft Windows, Mac OS X, RISC OS και πιθανώς σε άλλα. Η PHP υποστηρίζει επίσης τους Apache, Microsoft Internet Information Server, Personal Web Server, Netscape και iPlanet servers, Oreilly Website Pro server, Caudium, Xitami, OmniHTTPd, και πολλούς άλλους webserver. Για την πλειοψηφία των server η PHP έχει ένα module, για τους υπόλοιπους η PHP μπορεί να λειτουργήσει ως ένας CGI επεξεργαστής.

Έτσι με την PHP έχουμε την ελευθερία επιλογής ενός λειτουργικού συστήματος και ενός web server. Επιπλέον, έχουμε επίσης την ελευθερία να χρησιμοποιήσουμε συναρτησιακό (procedural) ή αντικειμενοστρεφή (object oriented) προγραμματισμό ή μια ανάμειξη τους. Αν και η παρούσα έκδοση δεν υποστηρίζει όλα τα πρότυπα χαρακτηριστικά, μεγάλες βιβλιοθήκες κώδικα και μεγάλες εφαρμογές (συμπεριλαμβανομένης και της βιβλιοθήκης PEAR) είναι γραμμένες μόνο με αντικειμενοστρεφή κώδικα.

Με την PHP δεν είμαστε περιορισμένοι να εξάγουμε HTML. Οι δυνατότητες της PHP συμπεριλαμβάνουν την εξαγωγή εικόνων, αρχείων PDF, ακόμη και ταινίες Flash (χρησιμοποιώντας τα libswf και Ming) παράγονται αμέσως. Μπορούμε επίσης να εξάγουμε εύκολα οποιοδήποτε κείμενο όπως XHTML και οποιοδήποτε άλλο XML αρχείο. Η PHP μπορεί να δημιουργεί αυτόματα αυτά τα αρχεία και να τα αποθηκεύει στο σύστημα αρχείων, αντί να τα εκτυπώνει, αποτελώντας έτσι μια server-side cache για το δυναμικό μας περιεχόμενο.

Ένα από τα πιο δυνατά και σημαντικά χαρακτηριστικά της PHP είναι η υποστήριξη που έχει για ένα μεγάλο σύνολο βάσεων δεδομένων. Η συγγραφή μιας σελίδας που υποστηρίζει βάσεις δεδομένων είναι εξαιρετικά απλή. Οι εξής βάσεις δεδομένων υποστηρίζονται μέχρι στιγμής:

Adabas D	Ingres	Oracle (OCI7 and OCI8)
dBase	InterBase	Ovrimos
Empress	FrontBase	PostgreSQL
FilePro (read-only)	mSQL	Solid
Hyperwave	Direct MS-SQL	Sybase
IBM DB2	MySQL	Velocis
Informix	ODBC	Unix dbm

Έχουμε επίσης μια αφαιρετική επέκταση DBX βάσεων δεδομένων (DBX database abstraction extension) που μας επιτρέπει διάφανα να χρησιμοποιήσουμε οποιαδήποτε βάση δεδομένων υποστηρίζεται από αυτή την επέκταση. Επιπλέον η PHP υποστηρίζει το ODBC, το Open Database Connection standard (Ανοιχτό πρότυπο Σύνδεσης Βάσεων δεδομένων) έτσι μπορούμε να συνδεθούμε σε οποιαδήποτε βάση δεδομένων που υποστηρίζει αυτό το παγκόσμιο πρότυπο.

Η PHP έχει επίσης υποστήριξη για επικοινωνία με άλλες υπηρεσίες χρησιμοποιώντας πρωτόκολλα όπως LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM (στα Windows) και αμέτρητα άλλα. Μπορούμε επίσης να ανοίξουμε raw network sockets και να αλληλεπιδράσουμε με οποιοδήποτε άλλο πρωτόκολλο. Η PHP έχει ακόμη υποστήριξη για την περίπλοκη ανταλλαγή δεδομένων WDDX μεταξύ σχεδόν όλων των Web programming γλωσσών. Μιλώντας για δια-επικοινωνία, η PHP υποστηρίζει instantiation αντικειμένων Java και τα χρησιμοποιεί διάφανα σαν αντικείμενα PHP. Μπορούμε επίσης να χρησιμοποιήσουμε την CORBA επέκταση μας για να προσπελάσουμε remote (απομακρυσμένα) αντικείμενα.

Η PHP έχει εξαιρετικά χρήσιμα χαρακτηριστικά επεξεργασίας κειμένων, από την POSIX επέξταση ή τις Perl regular expressions μέχρι XML parsing αρχείων. Για τη μεταγλώττιση και την πρόσβαση αρχείων XML, υποστηρίζουμε τα πρότυπα SAX και DOM. Μπορούμε να χρησιμοποιήσουμε την XSLT επέκταση μας για να μετατρέψουμε τα XML αρχεία σε άλλες μορφές.

Καθώς χρησιμοποιούμε την PHP στον τομέα του ecommerce, θα βρούμε τις Cybercash payment, CyberMUT, VeriSign Payflow Pro και CCVS συναρτήσεις χρήσιμες για τα online προγράμματα πληρωμής μας.

Τελευταίο αλλά σημαντικό, έχουμε πολλές άλλες ενδιαφέρουσες επεκτάσεις, τις mnoGoSearch search engine συναρτήσεις, πολλά εργαλεία συμπίεσης (gzip, bz2), μετατροπές ημερολογίου, μεταφράσεις...

1.3.3 Τι είναι η PEAR;

Η PEAR είναι συντομογραφία της "PHP Επέκταση και Εφαρμογή Αποθήκης "("PHP Extension and Application Repository") και προφέρεται ακριβώς όπως το φρούτο. Ο σκοπός του PEAR είναι να παρέχει:

- Μια δομημένη βιβλιοθήκη από τον ανοικτός-πηγάζοντα κώδικα για PHP χρήστες
- Ένα σύστημα για τη διανομή κώδικα και συντήρηση πακέτων
- Ένα τυποποιημένο ύφος για τον κώδικα γραμμένο σε PHP
- PHP Foundation Classes(PFC).(Τις PHP Κατηγορίες Θεμελίωσης)
- The PHP Extension Community Library (PECL).Την PHP Επέκταση Κοινοτικής Βιβλιοθήκης
- Ένα web site, κατάλογους διευθύνσεων και μεταφορτώσεις καθρεφτών για την υποστήριξη της PHP/PEAR κοινότητας .

Η PEAR είναι ένα κοινοτικός-οδηγημένο πρόγραμμα με PEAR Group σαν σώμα κυβέρνησης. Το πρόγραμμα έχει ιδρυθεί από τον Stig S. Bakken το 1999 και αρκετοί άνθρωποι έχουν συμμετάσχει στο πρόγραμμα από τότε.

Δομημένη βιβλιοθήκη PHP Κώδικα

Ο κώδικας στην PEAR χωρίζεται σε "πακέτα". Κάθε πακέτο είναι ένα χωριστό πρόγραμμα με την ομάδα ανάπτυξής του, αριθμός έκδοσης, κύκλος απελευθέρωσης, τεκμηρίωση και μια καθορισμένη σχέση με άλλες συσκευασίες (συμπεριλαμβανομένων των εξαρτήσεων). Τα πακέτα διανέμονται με ένα αρχείο περιγραφής μέσα, και εγκαθίστανται στο τοπικό σύστημά σας με τη χρησιμοποίηση του PEAR installer.

Υπάρχουν δύο τύποι πακέτων: πακέτα πηγής (περιέχοντας τα αρχεία πηγής μόνο), και δυαδικά πακέτα(περιορισμός της πλατφόρμας - συγκεκριμένα δυαδικά αρχεία, και

πιθανά αρχεία πηγής). Εγκαθηστόντας τα πακέτα πηγής με κώδικα C προφανώς απαιτεί ένα C περιβάλλον χτισίματος.

Διανομή κώδικα και Συντήρηση Πακέτων

Όλα τα πακέτα PEAR καταχωρούνται μέσα και φορτώνονται σε μια κεντρική βάση δεδομένων διαθέσιμα στην pear.php.net. Ανοικτός-πηγάζων τρίτο πακέτο μπορεί επίσης να καταχωρηθεί και να φορτωθεί. Πακέτα με closed-source μπορούν να εγκατασταθούν από τον PEAR installer, αλλά η PEAR βάση δεδομένων είναι για open-source κώδικα μόνο.

Pear.php.net θα παράσχει και μία ανθρώπινη-φιλική (HTML) και μηχανή-φιλική (currently XML-RPC) διεπαφή στην PEAR βάση δεδομένων . Η μεταφόρτωση πακέτων γίνεται με HTTP. Άλλες λειτουργίες που το pear.php.net site προσφέρει είναι:

- διαχείριση απολογισμού χρηστών (ενσωματωμένο με τον CVS server)
- διαχείριση πακέτων
- διαχείριση αποδέσμευσης

The PHP Foundation Classes(PFC)

Το PHP Foundation Classes είναι ένα υποσύνολο του PEAR το οποίο εστιάζει στην ποιότητα, γενικότητα, διαλειτουργικότητα και συμβατότητα. Εάν η PHP πρόκειται να συνεχίσει να χτίζει PEAR πακέτα πέρα από τον installer, θα συσσωρεύσει το PFC.

The PHP Extension Community Library (PECL)

PECL (προφέρεται "pickle") συνήθηζε να είναι μέρος της αποθήκης της PEAR για C επεκτάσεις αλλά εκείνα διανεμήθηκαν με την PHP 4, στην πραγματικότητα έχοντας κάπου να μετακινήσεις τις επεκτάσεις από την PHP ήταν ένα από τα κίνητρα όταν δημιουργήθηκε η PECL. Οι επεκτάσεις στην PECL ακολουθούν τα PHP πρότυπα κωδικοποίησης παρά τα PEAR's, αλλά διανέμονται και εγκαθίστανται όπως τα PEAR πακέτα.

Gtk Packages

Gtk packages είναι πακέτα που παρέχουν το λογισμικό το οποίοπ χρησιμοποιεί η τεχνολογία του προγράμματος . Κώδικας σε αυτήν την υπο--αποθήκη ακολουθεί τα PEAR's .

ΚΕΦΑΛΑΙΟ 2

2.1 Βασικοί Κανόνες Σύνταξης της PHP

2.1.1 Βγαίνοντας από την HTML

Όταν η PHP μεταγλωτίζει (parses) ένα αρχείο, απλά κάνει ένα πέρασμα στο κείμενο του αρχείου μέχρι να συναντήσει ένα από τα ειδικά tags που της λένε να αρχίσει να μεταφράζει το κείμενο ως κώδικα PHP. Ο parser (μεταγλωττιστής) τότε εκτελεί ολόκληρο τον κώδικα που βρίσκει, μέχρι να συναντήσει το επόμενο PHP tag κλεισίματος, το οποίο λέει στον parser να αρχίσει να κάνει ξανά, απλά ένα πέρασμα στο κείμενο. Αυτός είναι ο μηχανισμός που μας επιτρέπει να προσθέσουμε PHP κώδικα μέσα σε HTML: οτιδήποτε βρίσκεται έξω από τα tags της PHP μένει τελείως μόνο, ενώ οτιδήποτε μέσα μεταγλωττίζεται ως κώδικας.

Υπάρχουν τέσσερα σύνολα από tags που μπορούν να χρησιμοποιηθούν για να δηλώσουμε τα κομμάτια που έχουν κώδικα σε PHP. Από αυτά, μόνο δύο (<?php. . .?> and <script language="php">. . .</script>) είναι πάντα διαθέσιμα. Τα άλλα μπορούν να ενεργοποιηθούν και να απενεργοποιηθούν από το `php.ini` αρχείο ρυθμίσεων. Ενώ τα short-form tags και τα tags που μοιάζουν με αυτά της ASP μπορεί να είναι βολικά, δεν είναι τόσο portable όσο οι μακρύτερες εκδόσεις. Επίσης, αν σκοπεύουμε να προσθέσουμε PHP κώδικα σε XML ή XHTML, θα χρειαστεί να χρησιμοποιήσουμε την <?php. . .?> φόρμα για να προσαρμοστεί στην XML.

Τα tags που υποστηρίζονται από την PHP είναι:

Παράδειγμα 2-1. Τρόποι για να βγούμε (escape) από την HTML

```
1.  <?php echo("if you want to serve XHTML or XML documents, do
like this\n"); ?>

2.  <? echo ("this is the simplest, an SGML processing
instruction\n"); ?>
    <?= expression ?> This is a shortcut for "<? echo expression
?>"

3.  <script language="php">
        echo ("some editors (like FrontPage) don't
              like processing instructions");
    </script>

4.  <% echo ("You may optionally use ASP-style tags"); %>
    <%= $variable; # This is a shortcut for "<% echo . . ." %>
```

Ο πρώτος τρόπος, <?php. . .?>, είναι και ο προτιμότερος, καθώς επιτρέπει τη χρήση της PHP σε κώδικα συμβατό με την XML όπως η XHTML.

Ο δεύτερος τρόπος δεν είναι πάντα διαθέσιμος. Τα σύντομα tags είναι διαθέσιμα μόνο όταν έχουν ενεργοποιηθεί. Αυτό μπορεί να γίνει μέσω της συνάρτησης **short_tags()** (μόνο στην PHP 3), ενεργοποιώντας την επιλογή ρύθμισης

στο αρχείο ρυθμίσεων της PHP, ή κάνοντας compile την PHP με την επιλογή --enable-short-tags στο **configure**. Ακόμη και αν είναι ενεργοποιημένο ως προεπιλογή στο php.ini-dist, η χρήση των short tags δεν προτιμάται.

Ο τέταρτος τρόπος είναι διαθέσιμος μόνο αν τα ASP-style tags έχουν ενεργοποιηθεί χρησιμοποιώντας την επιλογή ρυθμίσεων.

Σημείωση: Υποστήριξη για τα ASP-style tags προστέθηκε στην έκδοση 3.0.4.

Σημείωση: Η χρήση των short tags θα πρέπει να αποφεύγεται κατά την ανάπτυξη εφαρμογών ή βιβλιοθηκων (libraries) που προορίζονται για διανομή (redistribution), ή εφαρμογή σε PHP servers που δεν τους χειριζόμαστε οι ιδιοι, επειδη τα short tags μπορεί να μην υποστηρίζονται από τον τελικό server. Για μεταφέρσιμο (portable), κώδικα που θα προορίζεται για χρήση και από άλλους, πρέπει να βεβαιωθούμε ότι δεν κάνουμε χρήση των short tags.

Το tag κλεισίματος για το block θα συμπεριλάβει το αμέσως επόμενο trailing newline αν υπάρχει. Επίσης, το tag κλεισίματος αυτόματα υποδηλώνει και ένα ερωτηματικό. Δεν χρειάζεται να έχουμε ερωτηματικό για να τερματίσουμε την τελευταία γραμμή ενός PHP block.

Η PHP μας επιτρέπει να χρησιμοποιήσουμε δομές σαν αυτή:

Παράδειγμα 2-2. Προχωρημένος τρόπος για να κάνουμε escape

```
<?php
if ($expression) {
    ?>
    <strong>This is true.</strong>
    <?php
} else {
    ?>
    <strong>This is false.</strong>
    <?php
}
?>
```

Αυτό λειτουργεί όπως περιμέναμε, επειδή όταν η PHP φτάνει στα ?> tags κλεισίματος, απλά αρχίζει να εμφανίζει ο,τιδήποτε βρει μέχρι να συνατήσει ένα άλλο tag ανοίγματος. Το παράδειγμα που δόθηκε εδώ έχει επινοηθεί, φυσικά, με σκοπό να εμφανίσουμε μεγάλα blocks κειμένου, αφού το να ξεφεύγουμε από τη μεταγλώττιση της PHP είναι γενικά πιο αποτελεσματικό από το να στέλνουμε ολόκληρο το κείμενο μέσω της συνάρτησης ή της ή κάτι τέτοιο.

Διαχωρισμός εντολών

Οι εντολές διαχωρίζονται με τον ίδιο τρόπο όπως και στην C ή την Perl - τερματίζουμε κάθε εντολή με ένα ερωτηματικό.

Το tag κλεισίματος (?>) επίσης υποδηλώνει το τέλος μιας έκφρασης-δηλωσης, συνεπώς τα ακόλουθα είναι ισοδύναμα:

```
<?php
    echo "This is a test";
?>

<?php echo "This is a test"
?>
```

Σχόλια

Η PHP υποστηρίζει σχόλια σαν της 'C', 'C++' και του Unix shell. Για παράδειγμα:

```
<?php
    echo "This is a test"; // This is a one-line c++ style
comment
    /* This is a multi line comment
       yet another line of comment */
    echo "This is yet another test";
    echo "One Final Test"; # This is shell-style style comment
?>
```

Το σχόλιο "μια γραμμής" σχολιάζει μόνο μέχρι το τέλος μιας γραμμής ή του τρέχοντος block στον κώδικα της PHP, όποιο είναι πρώτο.

```
<h1>This is an <?php # echo "simple";?>
example.</h1>
<p>The header above will say 'This is an example'.
```

Πρέπει να προσέχουμε να μην εμφωλεύουμε σχόλια σαν αυτά της 'C', κάτι που είναι πιθανό να συμβεί όταν σχολιάζουμε μεγάλα blocks.

```
<?php
 /*
    echo "This is a test"; /* This comment will cause a problem
*/
 */
?>
```

Το σχόλιο μιας γραμμής στην πραγματικότητα σχολιάζει μέχρι το τέλος μιας γραμμής ή του τρέχοντος block στον κώδικα της PHP, όποιο είναι πρώτο. Αυτό σημαίνει πως ο HTML κώδικας που ακολουθεί το // ?> Θα εκτυπωθεί. Το tag ?> ξεφεύγει από την κατάσταση της PHP και επιστρέφει στην HTML, και το // δεν μπορεί να το επηρεάσει.

2.1.2 Μεταβλητές

Μια μεταβλητή (variable) είναι μία ειδική θέση την οποία μπορούμε να ορίσουμε εμείς για την αποθήκευση μιας τιμής. Οι μεταβλητές είναι ζωτκής σημασίας για τον προγραμμαυισμό. Χωρίς τις μεταβλητές , ήμασταν υποχρεωμένοι να ενσωματώνουμε στον κώδικα των script όλες τις τιμές που χρειαζόμαστε. Προσθέτοντας δύο αριθμούς και εκτυπώνοντας το αποτέλεσμα, μπορούμε να επιτύχουμε κάτι χρήσιμο:

```
print  (2 + 4) ;
```

Ωστόσο, το παραπάνω script θα είναι χρήσιμο μόνο για τους ανθρώπους που θέλουν να ξέρουν το άθροισμα των τιμών 2 και 4. Για να αντιπαρέλθουμε αυτό τον περιορισμό, θα μπορούσαμε να γράψουμε ένα script για την εύρεση του αθροίσματος ενός άλλου ζεύγους αριθμών π.χ των 3 και 5. Ωστόσο, αυτή η προσέγγιση στον προγραμματισμό είναι παράλογη και σ'αυτό ακριβώς το σημείο καλούνται να παίξουν τον σημαντικό ρόλο τους οι μεταβλητές.

Οι μεταβλητές μας δίνουν την δυνατότητα να δημιουργούμε "πρότυπα" για την εκτέλεση διάφορων ενεργειών (π.χ. την πρόσθεση δύο αριθμών) χωρίς να ανησυχούμε για τις τιμές που περιέχουν οι μεταβλητές. Οι τιμές ανατίθενται στις μεταβλητές κατά την εκτέλεση του script – πιθανώς από τον ίδιο τον χρήστη, ή μέσω ενός ερωτήματος το οποίο εκτελείται σε μία βάση δεδομένων.

Θα πρέπει να χρησιμοποιούμε μεταβλητές οποτεδήποτε τα δεδομένα που χρησιμοποιούνται για την εκτέλεση μιας λειτουργίας στο script μας υπόκεινται σε αλλαγή από την μία εκτέλεση του script στην επόμενη, ή κατά την διάρκεια ζωής του script.

Μία μεταβλητή αποτελείται από ένα όνομα της επιλογής μας, το οποίο έχει σαν πρόθεμα ένα σύμβολο δολαρίου ($). Τα ονόματα των μεταβλητών μπορούν να περιλαμβάνουν γράμματα, αριθμούς και τον χαρακτήρα της κάτω παύλας (_). Τα ονόματα των μεταβλητών δεν μπορούν να περιλαμβάνουν κενά διαστήματα. Πρέπει πάντα να ξεκινούν με ένα γράμμα , ή με τον χαρακτήρα της κάτω παύλας . Το ακόλουθο απόσπασμα κώδικα ορίζει ορισμένες έγκυρες μεταβλητές :

```
$a ;
$a_longish_variable_name ;
$2453 ;
$sleepyzzzz ;
```

ΣΗΜΕΙΩΣΗ: Θα πρέπει να χρησιμοποιούμε ένα σταθερό στύλ για την ονομασία των μεταβλητών μας. Επίσης τα ονόματα των μεταβλητών θα πρέπει να είναι περιγραφικά. Για παράδειγμα, εάν το script μας χειρίζεται ονόματα χρηστών και κωδικούς πρόσβασης, δεν πρέπει να δημιουργήσουμε μία μεταβλητή $n για το όνομα και μία μεταβλητή $p για τον κωδικό πρόσβασης – τα ονόματα αυτά δεν είναι περιγραφικά.

Εάν ασχοληθούμε με το ίδιο script λίγους μήνες αργότερα, τα $n και $p μπορεί να μην μας λένε τίποτα για την λειτουργία των μεταβλητών.

Το ελληνικό ερωτηματικό (;) – γνωστό επίσης σαν χαρακτήρας τερματισμού εντολών – χρησιμοποιείται για τον τερματισμό μιας εντολής της PHP. Οι χαρακτήρες ελληνικού ερωτηματικού στο παραπάνω απόσπασμα κώδικα δεν είναι μέρος των ονομάτων των μεταβλητών.

ΣΗΜΕΙΩΣΗ:Μια μεταβλητή (variable) είναι ένας χώρος κτακράτησης ενός τύπου δεδομένων. Μία μεταβλητή μπορεί να αποθηκεύει αριθμούς, ακολουθίες χαρακτήρων (αλφαριθμητικά, strings), αντικείμενα, διατάξεις (arrays) ή λογικές τιμές (boolean). Τα περιεχόμενα μιας μεταβλητής μπορούν να αλλάζουν ανά πάσα στιγμή.

Όπως βλέπουμε, έχουμε στην διάθεσή μας άφθονες επιλογές για την ονομασία μεταβλητών. Για να δηλώσουμε μία μεταβλητή, το μόνο που χρειάζεται είναι να την συμπεριλάβουμε στο script μας. Όταν δηλώνουμε μία μεταβλητή, συνήθως θα αναθέτουμε μία τιμή σ'αυτήν, στην ίδια εντολή, όπως βλέπουμε στην συνέχεια:

```
$num1 = 8 ;
$num2 = 23 ;
```

Οι παραπάνω γραμμές κώδικα δηλώνουν δύο μεταβλητές χρησιμοποιώντας τον τελεστή εκχώρησης τιμής (=) για να τους αναθέσουν επίσης τιμές. Αφού δώσουμε τιμές στις μεταβλητές μας, μπορούμε να τις αντιμετωπίσουμε σαν να ήταν οι ίδιες οι τιμές . Με άλλα λόγια, η εντολή

```
print $num1 ;
```

ισοδυναμεί με την

```
print 8
```

εφόσον η μεταβλητή $num1 περιέχει την τιμή 8.

2.1.3 Τύποι Δεδομένων

Διαφορετικοί τύποι δεδομένων δεσμεύουν διαφορετικά ποσά μνήμης και μπορεί να αντιμετωπίζονται διαφορετικά μέσα σ'ένα script. Για τον λόγο αυτό, ορισμένες γλώσσες προγραμματισμού απαιτούν από τον προγραμματιστή να δηλώνει εκ των προτέρων τον τύπο των δεδομένων που θα αποθηκεύει μία μεταβλητή. Εν αντιθέσει η PHP είναι σχετικά χαλαρή όσον αφορά στο θέμα του τύπου δεδομένων, πράγμα το οποίο σημαίνει ότι εξακριβώνει τους τύπους δεδομένων όταν ανατίθενται δεδομένα σε κάθε μεταβλητή. Αυτό είναι ταυτόχρονα πλεονέκτημα και μειονέκτημα. Από τη μία σημαίνει ότι οι μεταβλητές μπορούν να χρησιμοποιούνται με ευέλικτο τρόπο, αποθηκεύοντας ένα αλφαριθμητικό την μία φορά και έναν ακέραιο την άλλη. Από την άλλη αυτή η προσέγγιση μπορεί να δημιουργήσει προβλήματα σε μεγαλύτερα scripts, εάν αναμένουμε ότι μία μεταβλητή αποθηκεύει έναν συγκεκριμένο τύπο

δεδομένων, όταν στην πραγματικότητα αποθηκεύει κάτι εντελώς διαφορετικό. Για παράδειγμα υποθέστε ότι δημιουργήσαμε κώδικα σχεδιασμένο έτσι ώστε να χρησιμοποιεί μία μεταβλητή για τον χειρισμό διατάξεων. Εάν αντί μιας διάταξης η συγκεκριμένη μεταβλητή περιέχει μία αριθμητική τιμή, θα προκληθούν σφάλματα όταν ο κώδικάς μας θα επιχειρήσει να εκτελέσει οποιεσδήποτε ενέργειες με την μεταβλητή οι οποίες εφαρμόζονται μόνο σε διατάξεις.
Ο Πίνακας παρουσιάζει τους έξι στάνταρ τύπους δεδομένων που υποστηρίζει η PHP.

ΠΙΝΑΚΑΣ 2.1.1 Οι Στάνταρ Τύποι Δεδομένων που Υποστηρίζει η PHP

Integer	5	Ακέραιοι αριθμοί
Double	3.234	Αριθμοί κινητής υποδιαστολής
String	"hello"	Μία ακολουθία χαρακτήρων(συμβολοσειρά)
Boolean	true	Μία από τις δύο λογικές τιμές: true(αληθές) ή false(ψευδές)
Object		Ένα υπόδειγμα (instance) αντικειμένου μιας κλάσης
Array		Ένα διατεταγμένο σύνολο κλειδιών και τιμών

Η PHP υποστηρίζει επίσης δύο ειδικούς τύπους δεδομένων , οι οποίοι παρουσιάζονται στον Πίνακα 2.1.2.

ΠΙΝΑΚΑΣ 2.1.2 Ειδικοί Τύποι Δεδομένων

Resource	Αναφορά σε έναν πόρο (π.χ. μία βάση δεδομένων)
NULL	Μία μη_αρχικοποιημένη μεταβλητή(μία μεταβλητή της οποίας η τιμή είναι απροσδιόριστη)

Οι τύποι πόρων επιστρέφονται συχνά από συναρτήσεις οι οποίες ασχολούνται με εξωτερικές εφαρμογές ή αρχεία. Ο τύπος NULL δεσμεύεται για τις μεταβλητές που δεν έχουν αρχικοποιηθεί (δηλαδή, τις μεταβλητές στις οποίες δεν έχουμε αναθέσει ακόμη μία τιμή).

Μπορούμε να χρησιμοποιούμε την εγγενή συνάρτηση gettype() της PHP για να ελέγχουμε τον τύπο δεδομένων οποιασδήποτε μεταβλητής . Εάν τοποθετήσουμε το όνομα μιας μεταβλητής μέσα στις παρενθέσεις σε μία κλήση αυτής της συνάρτησης , η gettype() θα επιστρέψει ένα αλφαριθμητικό, το οποίο αντιπροσωπεύει τον τύπο δεδομένων της μεταβλητής. Ο κώδικας της Λίστας 2.1.1 αναθέτει πέντε διαφορετικούς τύπους δεδομένων σε μία μεταβλητή και την ελέγχει με την συνάρτηση gettype() κάθε φορά.

ΛΙΣΤΑ 2.1.1 Έλεγχος του Τύπου Δεδομένων μιας Μεταβλητής

```
1: <html>
2: <head>
```

```
 3:  <title>Listing 2.1.1 Testing the type of a
variable</title>
 4: </head>
 5: <body>
 6: <?php
 7:  $testing; / / δήλωση της μεταβλητής χωρις εκχώρηση
τιμής
 8: print gettype ($testing ) ; / / null (απροσδιόριστη
τιμή)
  9: print "<br>";
10: $testing = 5 ;
11: print gettype ($testing ) ; / / integer (ακέραιος )
12: print "<br>";
13: $testing = "five" ;
14: print gettype ($testing ) ; / / string (αλφαριθμητικό
)
15: print "<br>";
16: $testing = 5.0 ;
17: print gettype ($testing ) ; / / double (τιμή κινητής
υποδιαστολής )
18: print "<br>";
19: $testing = true ;
20: print gettype ($testing ) ; / / boolean (λογική τιμή
)
21: print "<br>";
22: ?>
23: </body>
24: </html>
```

Εισάγαμε τον παραπάνω κώδικα σε ένα αρχείο κειμένου με όνομα gettype.php και τοποθετούμε αυτό το αρχείο στον αρχικό κατάλογο εγγράφων του Web server μας. Όταν προσπελάζουμε το παραπάνω script με μια εφαρμογή Web browser, παράγει την ακόλουθη έξοδο:

```
NULL
integer
string
double
boolean
```

Αλλαγή του Τύπου Δεδομένων με την settype()

Η PHP διαθέτει την συνάρτηση settype() για την αλλαγή του τύπου δεδομένων μιας μεταβλητής. Για να χρησιμοποιήσουμε την συνάρτηση settype(), θα πρέπει να τοποθετήσουμε μέσα στις παρενθέσεις την μεταβλητή της οποίας τον τύπο δεδομένων θέλουμε να αλλάξουμε, καθώς και τον τύπο δεδομένων στον οποίο θέλουμε να αλλάξουμε την μεταβλητή.

```
Π.χ.$undecided = 3.14 ;
    print gettype ($undecided) ; / / double (τιμή κινητής
                                    υποδιαστολής )
    print "is $undecided <br>"; / / 3.14
    settype($undecided , 'string' ) ;
```

Όταν προσπελάζουμε το παραπάνω script με μια εφαρμογή Web browser, παράγει την ακόλουθη έξοδο:

```
double is 3.14
```

Γιατί Πρέπει να Ελέγχουμε τον Τύπο Δεδομένων;

Σε ποιές περιπτώσεις είναι χρήσιμο να γνωρίζουμε τον τύπο δεδομένων μιας μεταβλητής; Στον προγραμματισμό,παρουσιάζονται συχνά περιπτώσεις στις οποίες τα δεδομένα θα περνιούνται στον δικό μας κώδικα από μία άλλη πηγή . Οι συναρτήσεις μπορούν να δέχονται πληροφορίες από τον κώδικα που τις καλεί με την μορφή ορισμάτων (arguments). Για να μπορέσει να λειτουργήσει σωστά η συνάρτηση με τα δεδομένα που της έχουν παρασχεθεί, συχνά είναι καλή ιδέα να ελέγχουμε πρώτα ότι στην συνάρτηση έχουν περαστεί τιμές με σωστό τύπο δεδομένων. Για παράδειγμα, μία συνάρτηση η οποί αναμένει ένα όρισμα πόρου, δεν θα δουλέψει σωστά εάν της περάσουμε ένα αλφαριθμητικό.

2.1.4 Τελεστές και Εκφράσεις

Με όσα έχουμε μάθει μέχρι τώρα, μπορούμε να εκχωρούμε τιμές δεδομένων σε μεταβλητές. Μπορούμε επίσης να εξετάσουμε και να αλλάξουμε τον τύπο δεδομένων μιας μεταβλητής. Ωστόσο, μία γλώσσα προγραμματισμού δεν μπορεί να είναι χρήσιμη εάν δεν μας παρέχει κάποιο τρόπο για να χειριζόμαστε τα δεδομένα που αποθηκεύουμε. Οι τελεστές (operators) είναι σύμβολα τα οποία μας δίνουν την δυνατότητα να χρησιμοποιούμε μία ή περισσότερες τιμές για να παράγουμε μια νέα τιμή. Μία τιμή στην οποία επιδρά ένας τελεστής αναφέρεται με τον όρο τελεστέος (operand).

ΣΗΜΕΙΩΣΗ: Ένας τελεστής (operator) είναι ένα σύμβολο ή μία σειρά συμβόλων, τα οποία όταν χρησιμοποιούνται σε συνδυασμό με τιμές, εκτελούν μία ενέργεια και συνήθως παράγουν μία νέα τιμή.Ένας τελεστέος (operand) είναι μία τιμή η οποία χρησιμοποιείται μαζί με έναν τελεστή. Συνήθως χρησιμοποιούνται δύο τελεστέοι με έναν τελεστή.

Ας συνδυάσουμε τώρα δύο τελεστέους με έναν τελεστή για να παράγουμε μία νέα τιμή:

```
4 + 5
```

Οι τιμές 4 και 5 είναι οι τελεστέοι. Σ'αυτούς τους τελεστέους επιδρά ο τελεστής της πρόσθεσης (+), παράγοντας το αποτέλεσμα 9. Οι τελεστές σχεδόν πάντα

τοποθετούνται μεταξύ δύο τελεστέων, αν και παρακάτω θα δούμε ορισμένες εξαιρέσεις σ'αυτό τον κανόνα.

Ο συνδυασμός των τελεστέων με έναν τελεστή για την παραγωγή ενός αποτελέσματος ονομάζεται έκφραση (expression). Αν και στις περισσότερες περιπτώσεις οι τελεστές αποτελούν την βάση των εκφράσεων, μία έκφραση δεν είναι απαραίτητο να περιέχει έναν τελεστή. Στην πραγματικότητα, η PHP ορίζει σαν έκφραση οτιδήποτε μπορεί να χρησιμοποιηθεί σαν μία τιμή (στην θέση μιας τιμής). Σ'αυτό το ευρύ ορισμό περιλαμβάνονται οι σταθερές τιμές ακεραίων, όπως η 654, οι μεταβλητές, όπως η $user, και οι κλήσεις συναρτήσεων, όπως η gettype(). Για παράδειγμα , το (4+5) είναι μία έκφραση η οποία απαρτίζεται από δύο επιμέρους εκφράσεις και έναν τελεστή. Όταν μία έκφραση παράγει μία τιμή συχνά λέγεται ότι 'αποτιμάται' σ'αυτή την τιμή. Δηλαδή, αφού υπολογιστούν όλες οι επιμέρους εκφράσεις που περιέχει, η έκφραση μπορεί να αντιμετωπιστεί σαν να ήταν η ίδια η τιμή.

ΣΗΜΕΙΩΣΗ: Μία έκφραση (expression) είναι οποιοσδήποτε συνδυασμός συναρτήσεων, τιμών και τελεστών ο οποίος αποτιμάται σε μία τιμή. Γενικά ,εάν μπορούμε να χρησιμοποιήσουμε κάτι στην θέση μιας τιμής , αυτό θεωρείται ότι είναι έκφραση.

Τώρα που έχουμε ξεμπερδέψει με τις βασικές έννοιες και αρχές , είναι πλέον καιρός για μία ξενάγηση στους πιο κοινά χρησιμοποιούμενους τελεστές της PHP.

Ο Τελεστής Εκχώρησης Τιμής

Έχουμε δει ήδη τον τελεστή εκχώρησης τιμής, κάθε φορά που αρχικοποιούσαμε μία μεταβλητή σε μία τιμή. Ο τελεστής εκχώρησης τιμής αντιπροσωπεύεται από υον χαρακτήρα = . Ο τελεστής εκχώρησης τιμής παίρνει την τιμή του δεξιού τελεστέου του και την εκχωρεί στον αριστερό τελεστέο του:

```
$name = "matt" ;
```

Αριθμητικοί Τελεστές

Οι αριθμητικοί τελεστές λειτουργούν με τον γνωστό και αναμενόμενο τρόπο – εκτελούν αριθμητικές πράξεις .Ο Πίνακας 2.2.3 παρουσιάζει τους αριθμητικούς τελεστές. Ο τελεστής της πρόσθεσης προσθέτει τον δεξιό με τον αριστερό τελεστέο του. Ο τελεστής της αφαίρεσης αφαιρεί τον δεξιό τελεστέο από τον αριστερό. Ο τελεστής της διαίρεσης τον αριστερό τελεστέο με τον δεξιό. Ο τελεστής πολλαπλασιασμού πολλαπλασιάζει τον αριστερό τελεστέο με τον δεξιό.Ο τελεστής ακέραιου υπολοίπου(modulus) επιστρέφει το ακέραιο υπόλοιπο της διαίρεσης του αριστερού τελεστέου με τον δεξιό.

ΠΙΝΑΚΑΣ 2.1.3 Οι Αριθμητικοί Τελεστές.

+	Πρόσθεση	10+3	13
-	Αφαίρεση	10-3	7
/	Διαίρεση	10/3	3.33333333333333
*	Πολλαπλασιασμός	10*3	30
%	Ακέραιο υπόλοιπο	10%3	1

Ο Τελεστής Συνένωσης

Ο τελεστής συνένωσης (concatenation) αντιπροσωπεύεται από τον χαρακτήρα της τελείας . Αντιματωπίζοντας και τους δύο τελεστέους σαν αλφαριθμητικά, προσαρτά τον δεξιό τελεστέο στο τέλος του αριστερού. Συνεπώς ο κώδικας

```
"hello" , "world"
```

επιστρέφει το αποτέλεσμα

```
"hello world"
```

Ανεξάρτητα από τους τύπους δεδομένων τους , οι τελεστέοι αντιμετωπίζονται σαν αλφαριθμητικά και το αποτέλεσμα είναι πάντα ένα αλφαριθμητικό.

```
$centimeters = 212 ;
print "the width is " . ($centimeters/100). " metres";
```

Σύνθετοι Τελεστές Εκχώρησης Τιμής

Αν και στην πραγματικότητα υπάρχει μόνο ένας τελεστής εκχώρησης τιμής, η PHP υποστηρίζει ορισμένους σύνθετους τελεστές, οι οποίοι μεταβάλουν τον αριστερό τελεστέο τους και επιστρέφουν ένα αποτέλεσμα. Γενικά, οι τελεστές χρησιμοποιούν τους τελεστέους τους χωρίς να αλλάζουν τις τιμές τους. Οι τελεστές εκχώρησης που θα παρουσιάσουμε σ'αυτήν την ενότητα παραβιάζουν αυτό τον κανόνα. Ένας σύνθετος τελεστής εκχώρησης απαρτίζεται από ένα στάνταρ σύμβολο τελεστή το οποίο ακολουθείται από το σύμβολο ίσον. Οι σύνθετοι τελεστές εκχώρησης είναι μία συντόμευση και μας εξοικονομούν κόπο και χρόνο, επειδή δεν χρειάζεται να χρησιμοποιήσουμε ρητά δύο ξεχωριστούς τελεστές. Για παράδειγμα, ο κώδικας

```
$x = 4;
$x = $x + 4; / / η $x ισούτε τώρα με 8
```

μπορεί να γραφεί ως εξής:

```
$x = 4;
$x += 4; / / η $x ισούτε τώρα με 8
```

Υπάρχει ένας σύνθετος τελεστής εκχώρησης για κάθε έναν από τους αριθμητικούς τελεστές και ένας για τον τελεστή συνένωσης . Ο Πίνακας 2.1.4 παρουσιάζει τους σύνθετους τελεστές εκχώρησης.

ΠΙΝΑΚΑΣ 2.1.4 Σύνθετοι Τελεστές Εκχώρησης Τιμής

+=	$x + = 5	$x = $x + 5
-=	$x - = 5	$x = $x – 5
/=	$x / = 5	$x = $x / 5
*=	$x * = 5	$x = $x * 5
%=	$x % = 5	$x = $x % 5
.=	$x . = " test "	$x = $x . " test "

Κάθε ένα από τα παραδείγματα του Πίνακα 2.1.4 μετασχηματίζει με κάποιο τρόπο την τιμή της $x χρησιμοποιώντας την τιμή του δεξιού τελεστέου.

Αυτόματη Αύξηση/Μείωση της Τιμής μιας Ακέραιας Μεταβλητής

Όταν γράφουμε κώδικα με την PHP ,συχνά θα είναι απαραίτητο να αυξήσουμε ή να μειώσουμε την τιμή μιας ακέραιας μεταβλητής. Συνήθως θα θέλουμε να κάνουμε κάτι τέτοιο όταν μετραμε το πλήθος των επαναλήψεων ενός βρόχου(loop).

Γνωρίζουμε ήδη δύο τρόπους με τους οποίους γίνεται αυτό. Μπορούμε να αυξήσουμε την ακέραια τιμή που περιέχει η $x με τον τελεστή της πρόσθεσης

```
$x = $x + 1; / / η τιμή της $x έχει αυξηθεί
```

ή χρησιμοποιώντας ένα σύνθετο τελεστή εκχώρησης

```
$x + = 1; / / η τιμή της $x έχει αυξηθεί
```

Και στις δύο περιπτώσεις , ο παραγόμενος ακέραιος εκχωρείται στην μεταβλητή $x. Επειδή οι εκφράσεις αυτού του είδους είναι πολύ κοινές, η PHP παρέχει ορισμένους ειδικούς τελεστές οι οποίοι μας επιτρέπουν να προσθέτουμε ή να αφαιρούμε την ακέραια τιμή 1 από μία ακέραια μεταβλητή, εκχωρώντας ταυτόχρονα το αποτέλεσμα στην ίδια την μεταβλητή. Για τον λόγο αυτό, αναφέρονται σαν τελεστές αύξησης ή μείωσης της τιμής. Ο τελεστής αύξησης τιμής αποτελείται από δύο σύμβολα συν, τα οποία προσαρτώνται στο όνομα μιας μεταβλητής :

```
$x++; / / η τιμή της $x έχει αυξηθεί
```

Η έκφραση αυτή αυξάνει την τιμή της μεταβλητής $x κατά ένα. Χρησιμοποιώντας δύο σύμβολα μείον κατά τον ίδιο τρόπο μπορούμε να μειώσουμε την τιμή της μεταβλητής :

```
$x--; / / η τιμή της $x μειώνεται
```

Εάν χρησιμοποιούμε τους τελεστές αύξησης ή μείωσης τιμής σε συνδυασμό με έναν τελεστή εκτέλεσης υπό συνθήκες, ο τελεστέος τροποποιείται μόνο αφού έχει ολοκληρωθεί ο έλεγχος του τελεστή εκτέλεσης υπό συνθήκες:

```
$x = 3;
$x++ < 4; / / true
```

Στο προηγούμενο παράδειγμα, η $x περιέχει την τιμή 3 όταν ελέγχεται έναντι του 4 με τον τελεστή μικρότερο από, οπότε η έκφραση ελέγχου επιστρέφει true (αληθές). Αφού ολοκληρωθεί αυτός ο έλεγχος, η τιμή της $x αυξάνεται.

Σε ορισμένες περιπτώσεις, μπορεί να θέλουμε να αυξήσουμε ή να μειώσουμε την τιμή μιας μεταβλητής σε μία έκφραση ελέγχου πρίν εκτελεστεί ο έλεγχος. Η PHP παρέχει επίσης τελεστές αύξησης και μείωσης τιμής για το σκοπό αυτό.Αυτοί οι τελεστές συμπεριφέρονται με τον ίδιο ακριβώς τρόπο όπως και οι τελεστές αύξησης/μείωσης που είδαμε παραπάνω, αλλά γράφονται με τα σύμβολα σύν ή μείον πρίν από το όνομα της μεταβλητής :

```
++$x; / / η τιμή της $x αυξάνεται
--$x ; / / η τιμή της $x μειώνεται
```

Εάν αυτοί οι τελεστές χρησιμοποιηθούν, σαν μέρος μιας έκφρασης ελέγχου, η λειτουργία τους (αύξηση ή μείωση της τιμής) λαμβάνει χώρα πρίν εκτελεστεί ο έλεγχος .

```
$x = 3;
++$x < 4 ; / / false
```

Στο προηγούμενο απόσπασμα κώδικα, η τιμή $x αυξάνεται πριν ελεγχθεί έναντι του 4. Για τον λόγο αυτό η έκφραση ελέγχου στην δεύτερη γραμμή επιστρέφει false (ψευδές), επειδή το 4 δεν είναι μικρότερο από το 4.

Τελεστές Σύγκρισης

Οι τελεστές σύγκρισης εκτελούν ελέγχους στους τελεστέους τους. Επιστρέφουν μία λογική (τύπου Boolean) τιμή: true (αληθές) εάν ο έλεγχος είναι επιτυχής, ή false (ψευδές) σε κάθε άλλη περίπτωση. Αυτό το είδος εκφράσεων είναι χρλησιμο σε δομές ελέγχου, όπως αυτές που δημιουργούνται με τις εντολές if και while.

Για παράδειγμα, εάν θέλουμε να ελέγξουμε εάν η τιμή που περιέχει η $x είναι μικρότερη από 5, μπορούμε να χρησιμοποιήσουμε τον τελεστή "μικρότερο από":

```
$x < 5
```

Εάν η $x περιέχει την τιμή 3, η παραπάνω έκφραση επιστρέφει την τιμή true.Εάν η $x περιέχει την τιμή 7, η παραπάνω έκφραση αποτιμάται σε false.
Ο Πίνακας 2.1.5 παρουσιάζει τους τελεστές σύγκρισης .

ΠΙΝΑΚΑΣ 2.1.5 Οι Τελεστές Σύγκρισης

= =	Ισοδυναμία.Ο αριστερός τελεστέος είναι ισοδύναμος με τον δεξιό.	$x = = 5	False
!=	Μη-ισοδυναμία.Ο αριστερός τελεστέος δεν είναι ισοδύναμος με τον δεξιό.	$x != 5	True
= = =	Ταυτοποίηση.Ο αριστερόςτελεστέος είναι ισοδύναμος με τον δεξιό και έχουν τον ίδιο τύπο δεδομένων.	$x = = = "7"	False
>	Μεγαλύτερο από. Ο αριστερός τελεστέος είναι μεγαλύτερος από τον δεξιό.	$x > 4	False
>=	Μεγαλύτερο από ή ίσο με. Ο αριστερός τελεστέος είναι μεγαλύτερος από ή ίσος με τον δεξιό.	$x >= 4	True
<	Μικρότερο από. Ο αριστερός τελεστέος είναι μικρότερος από τον δεξιό.	$x < 4	False
<=	Μικρότερο από ή ίσο με. Ο αριστερός τελεστέος είναι μικρότερος από ή ίσος με τον δεξιό.	$x <= 4	True

Οι παραπάνω τελεστές χρησιμοποιούνται ευρέως με ακέραιες τιμές (integer), ή τιμές κινητής υποδιαστολής (double), αν και ο τελεστής ισοδυναμίας χρησιμοποιείται επίσης για την σύγκριση αλφαριθμητικών. Ο τελεστής = = ελέγχει για ισοδυναμία, ενώ ο τελεστής = εκχωρεί μία τιμή.

Δημιουργία Πολύπλοκων Εκφράσεων Ελέγχου με Λογικούς Τελεστές

Οι λογικοί τελεστές ελέγχουν συνδυασμούς λογικών τιμών (τιμών τύπου Boolean). Για παράδειγμα, ο τελεστής or (διάζευξη), ο οποίος αντιπροσωπεύεται από δύο χαρακτήρες καθέτου (| |) ή απλά από την λέξη or, επιστρέφει true εάν είτε ο αριστερός, είτε ο δεξιός τελεστέος του είναι true:

```
true | | false
```

Η παραπάνω έκφραση επιστρέφει true.

Ο τελεστής and (σύζευξη), ο οποίος αντιπροσωπεύεται από δύο χαρακτήρες && ή απλώς από την λέξη and, επιστρέφει true μόνο εάν αμφότεροι οι τελεστέοι του είναι true:

```
true && false
```

Η παραπάνω έκφραση επιστρέφει false. Ωστόσο, σπάνια θα χρησιμοποιούμε έναν λογικό τελεστή για να ελέγχουμε δύο σταθερές τιμές. Θα ήταν πιό λογικό να ελέγχουμε δύο ή περισσότερες εκφράσεις οι οποίες αποτιμώνται σε μία λογική τιμή.Για παράδειγμα η έκφραση

```
( $x > 2 ) && ( $x < 15 )
```

επιστρέφει true εάν η μεταβλητή $x περιέχει μία τιμή μεγαλύτερη από 2 και μικρότερη από15. Έχουμε συμπεριλάβει τις παρενθέσεις για να κάνουμε πιο ευανάγνωστο τον κώδικα. Ο Πίνακας 2.1.6 παρουσιάζει τους λογικούς τελεστές.

ΠΙΝΑΚΑΣ 2.1.6 Οι Λογικοί Τελεστές

||	Or (διάζευξη). Ο αριστερός ή δεξιός τελεστέος είναι true.	true || false	true
or	Or (διάζευξη). Ο αριστερός ή δεξιός τελεστέος είναι true	true or false	true
xor	Xor (αποκλειστική διάζευξη). Ο αριστερός ή δεξιός τελεστέος είναι true αλλά όχι ταυτόχρονα και οι δύο.	true xor false	false
&&	And (σύζευξη). Ο αριστερός και ο δεξιός τελεστέος είναι true.	true && false	false

and	And (σύζευξη). Ο αριστερός και ο δεξιός τελεστέος είναι true.	true and false	false
!	Not (άρνηση). Ο ένας και μοναδικός τελεστέος δεν είναι true.	!true	false

Γιατί υπάρχουν δύο εκδόσεις των τελεστών or και and; Η απάντηση έγκειται στην προτεραιότητα των τελεστών.

Προτεραιότητα των τελεστών

Όταν χρησιμοποιούμε έναν τελεστή, ο μηχανισμός εκτέλεσης της PHP συνληθως διαβάζει την έκφρασή μας από τ'αριστερά προς τα δεξιά. Ωστόσο, για τις πολύπλοκες εκφράσεις που περιλαμβάνουν περισσότερους από έναν τελεστές, τα πράγματα μπορεί να γίνουν λίγο πιο δύσκολα. Κατ'αρχήν ας δούμε μια απλή περίπτωση:

```
4 + 5
```

Σ'αυτή την περίπτωση δεν υπάρχει κίνδυνος σύγχυσης. Η PHP προσθέτει απλώς το 4 με το 5. Τί γίνεται όμως με το ακόλουθο απόσπασμα;

```
4 + 5 * 2
```

Εδώ προκύπτει ένα πρόβλημα. Τι σημαίνει αυτή η ασαφής έκφραση – υπολογισμό του αθροίσματος του 4 και 5, πολλαπλασιασμός επί 2 και αποτέλεσμα 18; Ή μήπως σημαίνει πρόσθεση του 4 στο αποτέλεσμα του 5 επί 2, με τελικό αποτέλεσμα την τιμή 14; Εάν διαβάζουμε την παραπάνω έκφραση από τ' αριστερά προς τα δεξιά, θα ίσχυε η πρώτη περίπτωση. Ωστόσο, η PHP ορίζει διαφορετική προτεραιότητα για κάθε τελεστή. Επειδή ο τελεστής του πολλαπλασιασμού έχει υψηλότερη προτεραιότητα από τον τελεστή της πρόσθεσης, η σωστή λύση στο παραπάνω πρόβλημα είναι η δεύτερη.

Μπορούμε να χρησιμοποιήσουμε παρενθέσεις για να υποχρεώσουμε την PHP να εκτελέσει την έκφραση της πρόσθεσης πρίν από την έκφραση του πολλαπλασιασμού:

```
( 4 + 5 ) * 2
```

Οποιαδήποτε κι αν είναι η προτεραιότητα των τελεστών μιας πολύπλοκης έκφρασης, καλή ιδέα είναι να χρησιμοποιούμε πάντα παρενθέσεις για να κάνουμε σαφέστερο τον κώδικά μας και για να αποφεύγουμε δύσκολα στον εντοπισμό σφάλματα. Ακολουθεί μία λίστα των τελεστών κατά σειρά προτεραιότητας (οι τελεστές με την υψηλότερη προτεραιότητα παρουσιάζονται πρώτοι):

```
++, --, (ρητή δήλωση τύπου)
/, * , %
```

```
+, -
<, <=, =>, >
= =, = = =, ! =
&&
| |
=, + =, - =, / = , *=, %=, . =
and
xor
or
```

Όπως βλέπουμε, ο τελεστής or έχει χαμηλότερη προτεραιότητα από τον | |, ενώ ο τελεστής and έχει χαμηλότερη προτεραιότητα από τον &&. Αυτό σημαίνει ότι μπορούμε να χρησιμοποιούμε τους χαμηλότερης προτεραιότητας λογικούς τελεστές για να αλλάζουμε τον τρόπο με τον οποίο διαβάζουμε και αποτιμάμε μια πολύπλοκη έκφραση ελέγχου. Οι δύο ακόλουθες εκφράσεις είναι ισοδύναμες, αλλά η δεύτερη είναι πολύ πιο ευανάγνωστη :

```
$x and $y | | $z
$x && ( $y | | $z )
```

Όπως επίσης ισοδύναμη έκφραση είναι και η εξής :

```
$x and ( $y or $z)
```

Η σειρά προτεραιότητας είναι ο μόνος λόγος για τον οποίο υπάρχουν δύο μορφές για τους λογικούς τελεστές σύζευξης και διάζευξης : && και and, | | και or. Ωστόσο, στις περισσότερες περιπτώσεις (αν όχι σ'όλες) η χρήση των παρενθέσεων κάνει σαφέστερο τον κώδικά μας και ελαχιστοποιεί την πιθανότητα σφαλμάτων σε σύγκριση με την συγγραφή κώδικα ο οποίος βασίζεται μόνο στην προτεραιότητα αυτών των τελεστών.

2.1.5 Σταθερές

Οι μεταβλητές μας παρέχουν έναν ευέλικτο τρόπο αποθήκευσης δεδομένων. Μπορούμε να αλλάζουμε τις τιμές τους και τον τύπο των δεδομένων που αποθηκεύουν ανά πάσα στιγμή. Ωστόσο, εάν θέλουμε να δουλέψουμε με μία τιμή η οποία δε θέλουμε να αλλάξει καθ'όλη την διάρκεια εκτέλεσης ενός script, μπορούμε να ορίσουμε μία σταθερά (constant). Για την δημιουργία μιας σταθεράς πρέπει να χρησιμοποιούμε την εγγενή συνάρτηση define() της PHP.Αφού χρησιμοποιήσουμε την define(), η σταθερά που ορίζουμε δεν μπορεί να αλλάξει. Για να χρησιμοποιήσουμε την define(), θα πρέπει να τοποθετήσουμε το όνομα της σταθεράς και την τιμή που θέλουμε να της αναθέσουμε μέσα στο ζεύγος παρενθέσεων της συνάρτησης. Οι τιμές αυτές πρέπει να χωρίζονται μεταξύ τους με ένα κόμμα :

```
define(  "CONSTANT_NAME" , 42 ) ;
```

Η τιμή της σταθεράς μπορεί να είναι ένας αριθμός, ένα αλφαριθμητικό, ή μία λογική (Boolean) τιμή. Το όνομα της σταθεράς αναγράφεται με όλους τους χαρακτήρες κεφαλαίους. Οι σταθερές προσπελάζονται μόνο βάσει του ονόματός τους , δεν

απαιτείται η χρήση του συμβόλου $.Ο κώδικας της Λίστας 2.1.2 ορίζει και προσπελάζει μία σταθερά.

ΛΙΣΤΑ 2.1.2 Ορισμός μιας Σταθεράς

```
1: <html>
2: <head>
3: <title>Listing 2.2.2 Defining a constant </title>
4: </head>
5: <body>
6: <?php
7: define ("USER" , "Gerald") ;
8: print "Welcome " .USER ;
9: ?>
10: </body>
11: </html>
```

Παρατηρούμε ότι στην γραμμή 8 χρησιμοποιούμε τον τελεστή συνένωσης για να προσαρτήσουμε την τιμή της σταθεράς στο αλφαριθμητικό "Welcome". Αυτό γίνεται επειδή ο μηχανισμός εκτέλεσης της PHP δεν έχει κανένα τρόπο για να κάνει διαχωρισμό μεταξύ μιας σταθεράς και ενός αλφαριθμητικού το οποίο περικλείεται σε εισαγωγικά.

Εισάγουμε τον παραπάνω κώδικα σε ένα αρχείο κειμένου με όνομα constant.php και το τοποθετούμε αυτό το αρχείο στον αρχικό κατάλογο εγγράφων του Web server μας. Όταν προσπελάζουμε αυτό το script με μία εφαρμογή Web browser, παράγει την ακόλουθη έξοδο:

```
Welcome Gerald
```

Προκαθορισμένες Σταθερές

Η PHP παρέχει ορισμένες προκαθορισμένες σταθερές τις οποίες μπορούμε να χρησιμοποιούμε. Για παράδειγμα, η _FILE_επιστρέφει το όνομα του αρχείου που διαβάζει επί του παρόντος ο μηχανισμός εκτέλεσης της PHP. Η_LINE_επιστρέφει τον αριθμό γραμμής του αρχείου. Αυτές οι σταθερές είναι χρήσιμες για την δημιουργία μηνυμάτων σφάλματος. Μπορούμε επίσης να εξακριβώσουμε ποιά έκδοση της PHP διερμηνεύει το script μας με την προκαθορισμένη σταθερά PHP_VERSION. Αυτό μπορεί να είναι χρήσιμο εάν θέλουμε να συμπεριλάβουμε πληροφορίες έκδοσης στην έξοδο του script όταν στέλνουμε μία αναφορά σφάλματος.

2.1.6 Οι Εντολές Ελέγχου της Ροης Εκτελεσης

Τα περισσότερα scripts αποτιμούν συνθήκες και αλλάζουν ανάλογα την συμπεριφορά τους. Η δυνατότητα λήψης αποφάσεων κάνει δυναμικές τις βασιζόμενες στην PHP σελίδες μας, ικανές να αλλάζουν το αποτέλεσμά τους ανάλογα με τις ισχύουσες συνθήκες. Όμοια με τις περισσότερες γλώσσες προγραμματισμού, η PHP μας

επιτρέπει να ενσωματώνουμε δυναμική συμπεριφορά στα scripts χρησιμοποιώντας την εντολή if.

Η Εντολή if

Η εντολή if μας παρέχει έναν τρόπο για να ελέγχουμε την εκτέλεση μιας άλλης εντολής (ή ενός τμήματος κώδικα το οποίο αποτελείται από περισσότερες εντολές και περικλείεται μέσα σε άγκιστρα). Η εντολή if αποτιμά μία έκφραση ελέγχου η οποία περικλείεται μέσα σε παρενθέσεις. Εάν αυτή η έκφραση επιστρέψει τιμή true, η εντολή εκτελείται. Αλλιώς η εντολή παρακάμπτεται. Η εντολή αυτή επιτρέπει στα scripts να λαμβάνουν αποφάσεις με βάση οποιονδήποτε αριθμό παραγόντων :

```
if  (έκφραση )  {
     / / κώδικας που θα εκτελεστεί εάν η έκφραση ελέγχου
αποτιμάται σε true
}
```

Στην Λίστα 2.1.3, ένα τμήμα κώδικα εκτελείται μόνο εάν μία μεταβλητή περιέχει το αλφαριθμητικό "happy".

ΛΙΣΤΑ 2.1.3 Χρήση της Εντολής if

```
1: <html>
2: <head>
3: <title>Listing 2.2.3</title>
4: </head>
5: <body>
6: <?php
7: $mood = "happy";
8: if ($mood = = "happy") {
9:     print "Hooray, I'm in a good mood";
10: }
11: ?>
12: </body>
13: </html>
```

Εδώ χρησιμοποιούμε τον τελεστή σύγκρισης = = για να συγκρίνουμε την τιμή της μεταβλητής $mood με το αλφαριθμητικό "happy". Εάν ταιριάζουν, η έκφραση αποτιμάται σε true και το τμήμα κώδικα κάτω από την εντολή if εκτελείται.

Χρήση της Πρότασης else στην Εντολή if

Όταν δουλεύουμε με την εντολή if, συχνά θα θέλουμε να ορίσουμε ένα εναλλακτικό τμήμα κώδικα, το οποίο θα εκτελείται όταν η έκφραση ελέγχου αποτιμάται σε

false.Αυτό μπορούμε να το κάνουμε προσθέτοντας την πρόταση else στην εντολή if, ακολουθούμενη από ένα δεύτερο τμήμα κώδικα:

```
if  (έκφραση )  {
      / / κώδικας που θα εκτελεστεί εάν η έκφραση ελέγχου
            αποτιμάται σε true
} else {
     / / ο κώδικας που θα εκτελεστεί σε όλες τις άλλες
περιπτώσεις
}
```

Χρήση της Πρότασης elseif στην Εντολή if

Μπορούμε να χρησιμοποιούμε μία δομή if....elseif....if για να ελέγχουμε πολλαπλές εκφράσεις πρίν παρέχουμε ένα "προεπιλεγμένο" τμήμα κώδικα:

```
if  (έκφραση ) {
  / / κώδικας που θα εκτελεστεί εάν η έκφραση ελέγχου αποτιμάται σε true
} elseif ( δεύτερη έκφραση ) {
   / / ο κώδικας που θα εκτελεστεί εάν η προηγούμενη έκφραση αποτιμάται σε false
   / / και αυτή η έκφραση αποτιμάται σε true
} else  {
  / / ο κώδικας που θα εκτελεστεί σε όλες τις άλλες περιπτώσεις
}
```

Εάν η πρώτη έκφραση ελέγχου δεν αποτιμάται σε true, το πρώτο τμήμα κώδικα αγνοείται. Κατόπιν, στη πρόταση elseif, αποτιμάται μία δεύτερη έκφραση ελέγχου. Εάν η δεύτερη έκφραση ελέγχου αποτιμάται σε true,θα εκτελεστεί το δεύτερο τμήμα κώδικα. Αλλιώς θα εκτελεστεί το τμήμα κώδικα της πρότασης else. Μπορούμε να συμπεριλάβουμε όσες προτάσεις elseif θέλουμε σε μία εντολή if, ενώ εάν δε χρειαζόμαστε μία προεπιλεγμένη ενέργεια μπορούμε να παραλείψουμε την πρόταση else.

ΣΗΜΕΙΩΣΗ:Η πρόταση elseif μπορεί επισης να γράφεται σαν δύο ξεχωριστές λέξεις (else if). Μπορούμε να χρησιμοποιούμε οποιαδήποτε από τις δύο μορφές θέλουμε – είναι απλά θέμα προσωπικής προτίμησης.

Η Εντολή switch

Η εντολή switch μας παρέχει έναν εναλλακτικό τρόπο για να αλλάζουμε την ροή εκτέλεσης ενός προγράμματος ανάλογα με το αποτέλεσμα μιας έκφρασης. Υπάρχουν ορισμένες σημαντικές διαφορές μεταξύ των εντολών switch και if. Χρησιμοποιώντας την εντολή if σε συνδυασμό με την elseif, μπορούμε να εξετάσουμε πολλαπλές συνθήκες. Η εντολή switch εξετάζει μόνο μία συνθήκη και εκτελεί διαφορετικά τμήματα κώδικα ανάλογα με το αποτέλεσμα αυτού του ελέγχου, εφόσον η έκφραση ελέγχου αποτιμάται σε έναν απλό τύπο δεδομένων (έναν αριθμό, ένα αλφαριθμητικό,ή μία τιμή τύπου Boolean) .Σε μία εντολή if η έκφραση ελέγχου

αποτιμάται πάντα σαν true ή false. Σε μία εντολή switch η έκφραση ελέγχου παράγει ένα αποτέλεσμα το οποίο μπορεί να ελεγχθεί έναντι οποιουδήποτε αριθμού τιμών:

```
switch (έκφραση ελέγχου)  {
    case αποτέλεσμα1:
        // ο κώδικας αυτός εκτελείται εάν η έκφραση ελέγχου επιστρέφει το
αποτέλεσμα1
    break;
    case αποτέλεσμα2:
        // ο κώδικας αυτός εκτελείται εάν η έκφραση ελέγχου επιστρέφει το
αποτέλεσμα2
    break;
    default;
      // ο κώδικας που θα εκτελεστεί εάν δεν εκτελέστηκε μέχρι τώρα μία εντολή
break
}
```

Συχνά η έκφραση ελέγχου της εντολής switch είναι απλώς μια μεταβλητή. Μέσα στο τμήμα κώδικα της εντολής switch θα υπάρχουν μία ή περισσότερες εντολές case, οι οποίες αντιπροσωπεύουν τις εναλλακτικές περιπτώσεις. Κάθε μία από αυτές τις case ελέγχει μία τιμή έναντι του αποτελέσματος της έκφρασης ελέγχου τηα εντολής switch. Εάν βρεθεί μία ισότητα,εκτελείται ο κώδικας που ακολουθεί μετά από την συγκεκριμένη εντολή case. Η εντολή break τερματίζει ολοκληρωτικά την εκτέλεση της switch. Εάν παραληφθεί, εξετάζεται η έκφραση ελέγχου της επόμενης case. Εάν η ροή εκτέλεσης φτάσει στην προεραιτική εντολή default, εκτελείται ο κώδικάς της.

ΠΡΟΣΟΧΗ: Μην ξεχνάτε ότι πρέπει να συμπεριλαμβάνουμε μία εντολή break στο τέλος οποιουδήποτε κώδικα ο οποίος πρόκειται να εκτελείται σαν μέρος μιας εντολής case. Χωρίς την break, η ροή εκτέλεσης του προγράμματος θα συνεχίσει στην επόμενη case και τελικά θα φτάσει στην default. Στις περισσότερες περιπτώσεις , αυτή δεν είναι η επιθυμητή συμπεριφορά.

2.1.7 Βρόχοι

Μέχρι τώρα εξετάσαμε τις αποφάσεις που μπορεί να λάβει ένα script σχετικά με τον κώδικα που θα εκτελέσει στην συνέχεια. Τα scripts μπορούν επίσης να αποφασίζουν πόσες φορές θα εκτελέσουν ένα τμήμα κώδικα. Για την εκτέλεση επαναλαμβανόμενων ενεργειών μέσα από ένα script, θα χρησιμοποιούμε τις εντολές δημιουργίας βρόχων. Σχεδόν χωρίς εξαίρεση, ένας βρόχος συνεχίζει να επαναλαμβάνεται μέχρι να ικανοποιηθεί μια συνθήκη, ή μέχρι να επιλέξουμε εμείς ρητά έξοδο από τον βρόχο.

Η Εντολή while

Η δομή της εντολής while δείχνει παρόμοια με την δομή μιας απλής εντολής if :

```
while ( έκφραση )  {
        // τμήμα κώδικα.....
}
```

Για όσο χρόνο η έκφραση της while αποτιμάται σε true, εκτελείται το τμήμα κώδικα κατ'επανάληψη. Κάθε εκτέλεση του τμήματος κώδικα ενός βρόχου αποκαλείται επανάληψη (iteration). Μέσα στο τμήμα κώδικα συνήθως θα αλλάζουμε κάτι το οποίο επηρεάζει την έκφραση της εντολής while, αλλιώς ο βρόχος θα επαναλαμβάνει την εκτέλεσή του επ'άπειρον.

Η Εντολή do...while

Θα μπορούσαμε να πούμε ότι η εντολή do...while είναι παρόμοια με μία αντίστροφη εντολή while. Η σημαντική διαφορά μεταξύ των δύο είναι ότι το τμήμα κώδικα εκτελείται πρίν από τον έλεγχο και όχι μετά:

```
do {
   // ο κώδικας που εκτελείται...
} while (έκφραση) ;
```

ΥΠΟΔΕΙΞΗ: Η έκφραση ελέγχου μιας εντολής do..while θα πρέπει πάντα να τερματίζεται με τον χαρακτήρα του ελλήνικού ερωτηματικού.

Η εντολή do...while είναι χρήσιμη όταν θέλουμε το τμήμα κώδικα να εκτελείται τουλάχιστον μία φορά, ακόμη κι αν η έκφραση ελέγχου της while αποτιμάται σε false.

Η Εντολή for

Χρησιμοποιώντας την εντολή for δεν μπορούμε να επιτύχουμε τίποτα περισσότερο από αυτά που μπορούμε να κάνουμε με μία εντολή while. Ωστόσο, η εντολή for είναι συχνά ένας καλύτερος και ασφαλέστερος τρόπος για να επιτύχουμε το ίδιο αποτέλεσμα. Η εντολή for μας επιτρέπει να επιτύχουμε τα ίδια πράγματα με την εντολη while, με μία και μόνο εντολή κώδικα. Αυτό μας δίνει την δυνατότητα να γράφουμε πιό συμπαγή κώδικα και μειώνει τις πιθανότητες να ξεχάσουμε να αυξήσουμε την μεταβλητή που χρησιμοποιούμε σαν μετρητή, πράγμα το οποίο θα είχε σαν αποτέλεσμα την δημιουργία ενός ατελείωτου βρόχου:

```
for    ( έκφραση αρχικοποίησης; έκφραση ελέγχου; έκγραση
       τροποποίησης; )  {
   // κώδικας που θα εκτελεστεί
}
```

ΣΗΜΕΙΩΣΗ: Ένας αέναος βρόχος είναι, όπως υποδηλώνει το όνομά του, ένας βρόχος ο οποίος συνεχίζει να εκτελείται επ'άπειρον. Εάν δημιουργήσουμε έναν αέναο βρόχο, το script μας θα συνεχίσεινα

τρέχει επ'άπειρον. Αυτό είναι ιδιαίτερα οδυνηρό για τον Web server μας και καθιστά άχρηστη την συγκεκριμένη ιστοσελίδα.

Οι εκφράσεις που περιέχονται μέσα στις παρενθέσεις της εντολής for χωρίζονται μεταξύ τους με τον χαρακτήρα του ελληνικού ερωτηματικού. Συνήθως η πρώτη έκφραση αρχικοποιεί μία μεταβλητή, η οποία λειτουργεί σαν μετρητής. Η δεύτερη έκφραση αντιπροσωπεύει την συνθήκη ελέγχου για τον βρόχο, ενώ η τρίτη έκφραση αυξάνει την τιμή της μεταβλητής μετρητή.

Έξοδος από έναν Βρόχο με την Εντολή break

Οι εντολές while και for διαθέτουν μία εγγενή έκφραση ελέγχου με την οποία μπορούμε να τερματίσουμε έναν βρόχο. Ωστόσο, η εντολή break μας επιτρέπει να τερματίσουμε έναν βρόχο βασιζόμενοι στα αποτελέσματα επιπλέον ελέγχων. Αυτό μπορεί να μας παρέχει έναν μηχανισμό ασφάλειας έναντι πιθανών σφαλμάτων. Για παράδειγμα δημιουργούμε έναν απλό βρόχο for ο οποίος διαιρεί έναν μεγάλο αριθμό με την τιμή μιας μεταβλητής η οποία αυξάνεται σε κάθε επανάληψη του βρόχου και εκτυπώνει το αποτέλεσμα στην οθόνη. Γενικά, η διαίρεση δια του μηδενός είναι πολύ κακή ιδέα στον προγραμματισμό και η Λίστα 2.1.4 μας διασφαλίζει έναντι αυτού του γεγονότος, επιβάλλοντας την έξοδο από τον βρόχο εάν η τιμή της μεταβλητής $counter ισούται με μηδέν.

ΛΙΣΤΑ 2.1.4 Χρήση της Εντολής break για την Έξοδο από έναν Βρόχο.

```
1: <html>
2: <head>
3: <title>Listing 2.2.4</title>
4: </head>
5: <body>
6: <?php
7: $counter = - 4;
8: for ( ; $counter <= 10; $counter++ ) {
9:     if ( $counter = = 0 )
10:        break;
11:   $temp = 4000/$counter;
12:      print "4000 divided by $counter is . . .
$temp<br>";
13: }
14: ?>
15: </body>
16: </html>
```

ΣΗΜΕΙΩΣΗ: Στην PHP, η διαίρεση ενός αριθμού διά του μηδενός δεν προκαλεί ένα "μοιραίο" σφάλμα (μία κατάσταση σφάλματος από την οποία δεν μπορεί να ανακάμψει το πρόγραμμά μας). Αντ'αυτού, η PHP εμφανίζει ένα προειδοποιητικό μήνυμα και η εκτέλεση του κώδικά μας συνεχίζεται.

Παράκαμψη μιας Επανάληψης με την Εντολή continue

Η εντολή continue τερματίζει την εκτέλεση της τρέχουσας επανάληψης ενός βρόχου, αλλάδεν τερματίζει τον ίδιο τον βρόχο. Αντίθετα, ξεκινά αμέσως η επόμενη επανάληψη του βρόχου. Η χρήση της εντολής break με τον τρόπο που παρουσιάσαμε, είναι μάλλον ένα δραστικό μέτρο. Χρησιμοποιώντας την εντολή continue μπορούμε να αποφεύγουμε το σφάλμα διαίρεσης διά του μηδενός χωρίς να τερματίζουμε ολοκληρωτικά τον βρόχο.

2.1.8 Συναρτήσεις

Θεωρήστε μία συνάρτηση (function) σαν μία μηχανή. Μία μηχανή παίρνει τα ακατέργαστα υλικά που τροφοδοτούμε σ'αυτήν και τα χειρίζεται με κάποιον τρόπο για να επιτύχει έναν σκοπό ή για να παράγει ένα προ'ι'όν. Μία συνάρτηση δέχεται τιμές (δεδομένα) από εμάς, τις επεξεργάζεται και κατόπιν εκτελεί μία ενέργεια (π.χ. εκτύπωση στο παράθυρο μιας εφαρμογής browser), επιστρέφει μία τιμή, ή και τα δύο.

Μία συνάρτηση είναι ένα αυτόνομο τμήμα κώδικα το οποίο μπορούν να καλούν τα script μας. Όταν καλείται η συνάρτηση εκτελείται ο κώδικάς της. Σε μία συνάρτηση μπορούμε να περνάμε τιμές (δεδομένα) και η συνάρτηση χειρίζεται αυτές τις τιμές με κάποιο τρόπο. Όταν ολοκληρώνει την εκτέλεσή της, η συνάρτηση μπορεί να επιστρέφει μία τιμή στον κώδικα που την κάλεσε.

Ορισμός μιας Συνάρτησης

Μπορούμε να ορίσουμε μία συνάρτηση χρησιμοποιώντας την εντολή function:

```
function όνομα_συνάρτησης ( $όρισμα1, $όρισμα2 ) {
 / / κώδικας της συνάρτησης
}
```

Το όνομα της συνάρτησης ακολουθεί μετά από την εντολή function και συνοδεύεται από ένα ζεύγος παρενθέσεων. Εάν η συνάρτησή μας απαιτεί ορίσματα, θα πρέπει να τοποθετήσουμε ονόματα μεταβλητών διαχωρισμένα με κόμματα μέσα στις παρενθέσεις. Αυτές οι μεταβλητές τροφοδοτούνται με τις τιμές που περνάμε στην συνάρτηση. Ακόμη κι αν η συνάρτησή μας δεν απαιτεί ορίσματα, οι παρενθέσεις είναι υποχρεωτικές .

Ο κώδικας της Λίστας 2.1.5 δημιουργεί μία συνάρτηση η οποία επιστρέφει το άθροισμα δύο αριθμών.

ΛΙΣΤΑ 2.1.5 Μία Συνάρτηση η Οποία Επιστρέφει μία Τιμή

```
1: <html>
2: <head>
3: <title>Listing 2.2.5</title>
```

```
4: </head>
5: <body>
6: <?php
7: function addNums ( $firstnum, $secondnum ) {
8:         $result = $firstnum + $secondnum;
9:         return $result;
10: }
11: print addNums (3,5) ;
12:  / / θα εκτυπώσει το "8"
13: ?>
14: </body>
15: </html>
```

Εισάγουμε τον παραπάνω κώδικα σε ένα αρχείο κειμένου με όνομα addnums.php και τοποθετούμε αυτό το αρχείο στον αρχικό κατάλογο εγγράφων του Web server μας. Όταν προσπελάζουμε αυτό το script μέσα από το περιβάλλον μιας εφαρμογής Web browser, παράγει την ακόλουθη έξοδο:

8

2.1.9 Το Εύρος των Μεταβλητών

Μία μεταβλητή η οποία δηλώνεται μέσα σε μία συνάρτηση παραμένει τοπική(local) σ'αυτή την συνάρτηση. Με άλλα λόγια, δεν είναι διαθέσιμη έξω από αυτή την συνάρτηση ή μέσα σε άλλες συναρτήσεις. Σε μεγαλύτερα έργα, η τεχνική αυτή μπορεί να αποτρέψει την κατά λάθος αντικατάσταση των περιεχομένων μιας μεταβλητής όταν δηλώνουμε δύο μεταβλητές με το ίδιο όνομα σε ξεχωριστές συναρτήσεις .

Προσπέλαση Μεταβλητών με την Εντολή global

Εξ'ορισμού, μέσα από μία συνάρτηση δεν είναι δυνατόν να προσπελάσουμε μία μεταβλητή η οποία έχει οριστεί σε κάποια άλλη θέση. Εάν επιχειρήσουμε να χρησιμοποιήσουμε μία μεταβλητή με το ίδιο όνομα, το αποτέλεσμα είναι ο ορισμός ή η προσπέλαση μιας τοπικής μεταβλητής .

2.1.10 Προκαθορισμένες Μεταβλητές

Πριν κατασκευάσουμε μία φόρμα και την χρησιμοποιήσουμε για να λάβουμε δεδομένα από τους χρήστες, θα πρέπει να κάνουμε μία μικρή απόκλιση για να επανεξετάσουμε το θέμα των γενικών μεταβλητών. Μία γενική μεταβλητή (global variable) είναι οποιαδήποτε μεταβλητή δηλώνεται στο κορυφαίο επίπεδο ενός script – δηλαδή, έξω από οποιαδήποτε συνάρτηση. Όλες οι γενικές μεταβλητές καθίστανται διαθέσιμες σε μία διάταξη στοιχείων (array) με όνομα $GLOBALS.

Η PHP διαθέτει αρκετές προκαθορισμένες μεταβλητές, οι οποίες χαρακτηρίζονται σαν "superglobals" (υπερ-γενικές)· ουσιαστικά, ο χαρακτηρισμός αυτός σημαίνει ότι οι μεταβλητές αυτές είναι πάντα παρούσες και διαθέσιμες στα script μας. Κάθε μία από τις ακόλουθες μεταβλητές είναι στην πραγματικότητα μία διάταξη άλλων μεταβλητών·

- `$_GET` - Περιέχει οποιεσδήποτε μεταβλητές παρέχονται σε ένα script μέσω της μεθόδου GET.
- `$_POST` - Περιέχει οποιεσδήποτε μεταβλητές παρέχονται σε ένα script μέσω της μεθόδου POST.
- `$_COOKIE` - Περιέχει οποιεσδήποτε μεταβλητές παρέχονται σε ένα script μέσω ενός cookie.
- `$_FILES` - Περιέχει οποιεσδήποτε μεταβλητές παρέχονται σε ένα script μέσω αρχείων.
- `$_ENV` - Περιέχει οποιεσδήποτε μεταβλητές παρέχονται σε ένα script σαν μέρος του περιβάλλοντος του server.
- `$_REQUEST`- Περιέχει οποιεσδήποτε μεταβλητές παρέχονται σε ένα script μέσω οποιουδήποτε μηχανισμού εισαγωγής δεδομένων από τον χρήστη.
- `$_SESSION` - Περιέχει οποιεσδήποτε μεταβλητές είναι επί του παρόντος δηλωμένος σε μία σύνοδο (session) επικοινωνίας.

2.2 Βασικές Εντολές της MySQL

Αυτό το κεφάλαιο αποκλίνει από την PHP με την οποία ασχοληθήκαμε μέχρι τώρα και παρουσιάζει μία σύντομη εισαγωγή στην σύνταξη της SQL της γλώσσας που θα χρησιμοποιούμε για να δημιουργήσουμε και για να χειριζόμαστε πίνακες βάσεων δεδομένων MySQL.Το κεφάλαιο αυτό περιλαμβάνει πολλά πρακτικά παραδείγματα και υποθέτει ότι γνωρίζουμε τους τρόπους δημιουργίας και εκτέλεσης ερωτημάτων σε βάσεις δεδομένων μέσω του MySQL monitor στα Windows ή σε *Linux/Unix.* Εναλλακτικά, εάν χρησιμοποιούμε ένα εργαλείο MySQL με γραφικό περιβάλλον, υποθέτουμε ότι γνωρίζουμε πώς να εκτελούμε ερωτήματα σε βάσεις δεδομένων μέσω αυτού του περιβάλλοντος.

Συγκεκριμένα, σ' αυτό το κεφάλαιο θα ασχοληθούμε με τα ακόλουθα:

- Οι βασικοί τύποι δεδομένων που υποστηρίζει η MySQL.
- Χρήση της εντολής `CREATE TABLE` για την δημιουργία ενός πίνακα.
- Χρήση της εντολής `INSERT` για την εισαγωγή εγγραφών σε πίνακες.

- Χρήση της εντολής SELECT για την ανάκτηση εγγραφών από πίνακες βάσεων δεδομένων.

- Χρήση βασικών συναρτήσεων και των προτάσεων WHERE και GROUP BY σε εκφράσεις της εντολής SELECT.

- Επιλογή δεδομένων από πολλαπλούς πίνακες με την χρήση της εντολής JOIN.

- Χρήση των εντολών UPDATE και REPLACE για την τροποποίηση υπαρχουσών εγγραφών.

- Χρήση της εντολής DELETE για την διαγραφή εγγραφών

2.2.1 Οι Τύποι Δεδομένων της MySQL

Ο σωστός ορισμός των πεδίων ενός πίνακα είναι σημαντικός για την συνολική βελτιστοποίηση μιας βάσης δεδομένων. Θα πρέπει πάντα να χρησιμοποιούμε τον τύπο δεδομένων και το μέγεθος που χρειαζόμαστε πραγματικά για κάθε πεδίο.

Η MySQL χρησιμοποιεί πολλούς διαφορετικούς τύπους δεδομένων, οι οποίοι ταξινομούνται σε τρεις κατηγορίες: αριθμητικοί, ημερομηνίας/ώρας και αλφαριθμητικών. Θα πρέπει να δίνουμε ιδιαίτερη προσοχή στο θέμα των τύπων δεδομένων, επειδή ο ορισμός του σωστού τύπου δεδομένων είναι πιο σημαντικός από οποιαδήποτε άλλη άποψη της διαδικασίας δημιουργίας ενός πίνακα.

Οι Αριθμητικοί Τύποι Δεδομένων

Η MySQL χρησιμοποιεί όλους τους στάνταρ αριθμητικούς τύπους δεδομένων της ANSI SQL. Συνεπώς, εάν ερχόμαστε στην MySQL από ένα διαφορετικό σύστημα βάσεων δεδομένων, οι ακόλουθοι ορισμοί θα πρέπει να μας φανούν γνωστοί. Στην ακόλουθη λίστα παρουσιάζονται οι πιο κοινοί τύποι δεδομένων, μαζί με τις περιγραφές τους.

ΣΗΜΕΙΩΣΗ: Κατά την παρουσίαση των αριθμητικών τύπων δεδομένων χρησιμοποιούνται σι όροι "προσημασμένος" και "μη-προσημασμένος". Ένας προσημασμένος ακέραιος είναι ένας θετικός ή αρνητικός ακέραιος αριθμός (δηλαδή φέρει πρόσημο), ενώ ένας μη-προσημασμένος ακέραιος είναι ένας μη-αρνητικός ακέραιος αριθμός.

- **INT** - Ενας ακέραιος κανονικού μεγέθους, ο οποίος μπορεί να είναι είτε προσημασμένος, είτε μη-προσημασμένος. Εάν είναι προσημασμένος, το υποστηριζόμενο πεδίο τιμών είναι από -2147483648 έως 2147483647. Εάν είναι μη-προσημασμένος, το υποστηριζόμενο πεδίο τιμών είναι από 0 έως 4294967295. Το μέγιστο μήκος του είναι 11 φηφία.

ΣΗΜΕΙΩΣΗ: Τα INT και INTEGER είναι συνώνυμα. Εάν ο όρος INTEGER μας βοηθά να θυμόμαστε καλύτερα τον τύπο δεδομένων, μπορούμε να τον χρησιμοποιούμε.

- **TINYINT** - Ενας πολύ μικρός ακέραιος, ο οποίος μπορεί να είναι προσημασμένος ή μη-προσημασμένος. Εάν είναι προσημασμένος, το υποστηριζόμενο πεδίο τιμών είναι από -128 έως 127. Εάν είναι μη-προσημασμένος, το υποστηριζόμενο πεδίο τιμών είναι από 0 έως 255. Το μέγιστο μήκος του είναι 4 ψηφία.

- **SMALLINT** - Ενας μικρός ακέραιος, ο οποίος μπορεί να είναι προσημασμένος ή μηπροσημασμένος. Εάν φέρει πρόσημο, το υποστηριζόμενο πεδίο τιμών είναι από - 32768 έως 32767. Εάν δεν φέρει πρόσημο, το υποστηριζόμενο πεδίο τιμών είναι από 0 έως 65535. Το μέγιστο μήκος του είναι 5 ψηφία.

- **MEDIUMINT** - Ενας μεσαίου μεγέθους ακέραιος, ο οποίος μπορεί να είναι προσημασμένος ή μη-προσημασμένος. Εάν φέρει πρόσημο, το υποστηριζόμενο πεδίο τιμών είναι από -8388608 έως 8388607. Εάν δεν φέρει πρόσημο, το υποστηριζόμενο πεδίο τιμών είναι από 0 έως 16777215. Το μέγιστο μήκος του είναι 9 ψηφία.

- **BIGINT** - Ενας μεγάλος ακέραιος, ο οποίος μπορεί να είναι προσημασμένος ή μη προσημασμένος. Εάν φέρει πρόσημο, το υποστηριζόμενο πεδίο τιμών είναι από - 9223372036854775808 έως 9223372036854775807. Εάν δεν φέρει πρόσημο, το υποστηριζόμενο πεδίο τιμών είναι από 0 έως 18446744073709551615. Το μέγιστο μήκος του είναι 11 ψηφία.

- **FLOAT(M,D)** - Ένας αριθμός κινητής υποδιαστολής, ο οποίος δεν μπορεί να είναι μη προσημασμένος. Μπορούμε να ορίσουμε το μέγεθος εμφάνισης (M) και το πλήθος των δεκαδικών (D). Αυτές οι παράμετροι δεν είναι απαραίτητες· εάν δεν τις καθορίσουμε, χρησιμοποιούνται οι τιμές 10,2, όπου το 2 είναι ο αριθμός των δεκαδικών και το 10 είναι ο συνολικός αριθμός ψηφίων (συμπεριλαμβανομένων των δεκαδικών). Η ακρίβεια μιας τιμής τύπου FLOAT μπορεί να φτάσει τα 24 δεκαδικά ψηφία.

- **DOUBLE(M,D)** - Ενας αριθμός κινητής υποδιαστολής διπλής ακρίβειας, ο οποίος δεν μπορεί να είναι μη-προσημασμένος. Μπορούμε να ορίσουμε το μέγεθος εμφάνισης (M) και το πλήθος των δεκαδικών (D). Αυτές οι παράμετροι δεν είναι απαραίτητες· εάν δεν τις καθορίσουμε, χρησιμοποιούνται οι τιμές 16,4, όπου το 4 είναι ο αριθμός των δεκαδικών. Η ακρίβεια μιας τιμής τύπου DOUBLE μπορείνα φτάσει τα 53 δεκαδικά ψηφία. Ο όρος REAL είναι συνώνυμος του όρου DOUBLE.

- **DECIMAL(M,D)** - Ενας μη-συμπιεσμένος αριθμός κινητής υποδιαστολής, ο οποίος δεν μπορεί να είναι μη-προσημασμένος. Στους μη-συμπιεσμένους αριθμούς τύπου DECIMAL, κάθε δεκαδικό ψηφίο αντιστοιχεί σε ένα byte. Είναι υποχρεωτικό να ορίσουμε το μέγεθος εμφάνισης (M) και το πλήθος των δεκαδικών (D). Το NUMERIC είναι ένα συνώνυμο για τον τύπο DECIMAL.

Απ' όλους τους αριθμητικούς τύπους δεδομένων που υποστηρίζει η MySQL, κατά πάσα πιθανότητα θα χρησιμοποιούμε πιο συχνά τον INT. Εάν ορίσουμε τα πεδία μας σε μικρότερο μέγεθος απ' όσο πραγματικά χρειαζόμαστε, μπορεί να συναντήσουμε διάφορα προβλήματα· για παράδειγμα, εάν ορίσουμε ένα πεδίο αποθήκευσης κωδικών με τύπο δεδομένων TINYINT (μη-προσημασμένο), δεν θα μπορέσουμε να προσθέσουμε την 256η εγγραφή στον πίνακά μας εάν το πεδίο αυτό είναι πρωτεύον κλειδί(και, κατά συνέπεια, απαιτούμενο).

Τύποι Δεδομένων για Τιμές Ημερομηνίας/Ωρας

Η MySQL υποστηρίζει αρκετούς τύπους δεδομένων για την αποθήκευση τιμών ημερομηνίας/ώρας, και αυτοί οι τύποι δεδομένων είναι ιδιαίτερα ευέλικτοι όσον αφορά στην είσοδο δεδομένων από τον χρήστη. Με άλλα λόγια, μπορούμε να εισάγουμε ημερομηνίες σι οποίες δεν είναι πραγματικές, όπως για παράδειγμα η 30 Φεβρουαρίου. Μπορούμε επίσης να αποθηκεύουμε ημιτελείς τιμές ημερομηνίας - τιμές από τις οποίες λείπουν κάποια στοιχεία. Εάν ξέρουμε ότι κάποιος γεννήθηκε κάποια ημέρα του Νοεμβρίου του 1980, μπορούμε να χρησιμοποιήσουμε την τιμή ημερομηνίας 1980-11-00, όπου το "00" αντιπροσωπεύει την άγνωστη ημέρα.

Η ευελιξία των τύπων δεδομένων ημερομηνίας/ώρας της MySQL σημαίνει επίσης ότι όλη η ευθύνη για τον έλεγχο των ημερομηνιών επαφίεται στον δημιουργό της εφαρμογής. Η MySQL ελέγχει μόνο την εγκυρότητα δύο στοιχείων: ότι η τιμή για τον μήνα ανήκει στο πεδίο τιμών 0-12 και η τιμή για την ημέρα ανήκει στο πεδίο τιμών 0-31.

Η MySQL δεν ελέγχει αυτόματα εάν η 30η ημέρα του δεύτερου μήνα (Φεβρουαρίου) είναι μία έγκυρη ή μη-έγκυρη ημερομηνία.
Οι τύποι δεδομένων που υποστηρίζει η MySQL για τιμές ημερομηνίας/ώρας είναι:

- **DATE** - Μία τιμή ημερομηνίας με την μορφή EEEE-MM-HH, μεταξύ 1000-01-01 και 9999-12-31. Για παράδειγμα, η 30η Δεκεμβρίου 1973 θα αποθηκευτεί σαν 1973-12-30.

- **DATETIME** - Ενας συνδυασμός τιμών ημερομηνίας και ώρας με την μορφή EEEE-MM-HH ΩΩ:ΛΛ:ΔΔ, μεταξύ 1000-01 -01 00:00:00 και 9999-12-31 23:59:59. Για παράδειγμα., η ώρα 3:30 το μεσημέρι της 3Ονς Δεκεμβρίου 1973 αποθηκεύεται σαν1973-12- 30 15:30:00.

- **TIMESTAMP** - Μία χρονική ένδειξη ανάμεσα στα μεσάνυχτα της 1ης Ιανουαρίου 1970 και στο έτος 2037. Μπορούμε να ορίζουμε πολλαπλά μήκη στο πεδίο TIMESTAMP, το οποίο συσχετίζεται άμεσα με την οποιαδήποτε τιμή αποθηκεύεται σ' αυτό. Το προεπιλεγμένο μήκος για τον τύπο δεδομένων TIMESTAMP είναι 14 και αποθηκεύει τιμές της μορφής EEEEMMHHΩΩΛΛΔΔ. Η μορφή αυτή δείχνει παρόμοια με την μορφή του τύπου δεδομένων DATETIME, χωρίς τις παύλες ανάμεσα στους αριθμούς· η ώρα 3:30 το μεσημέρι της 3Ονς Δεκεμβρίου του 1973 αποθηκεύεται σαν 19731230153000. Αλλα μεγέθη διαθέσιμα για τον τύπο δεδομένων

TIMESTAMP είναι τα 12 (ΕΕΜΜΗΗΩΩΛΛΔΔ), 8 (ΕΕΕΕΜΜΗΗ) και 6 (ΕΕΜΜΗΗ).

- **TIME** - Αποθηκεύει την ώρα σε μορφή ΩΩ:ΛΛ:ΔΔ.

- **YEAR(M)** - Αποθηκεύει ένα έτος σε διψήφια ή τετραφήφια μορφή. Εάν η παράμετρος μεγέθους (M) οριστεί σε 2 (π.χ. YEAR(2)), το έτος μπορεί να είναι από 1970 έως 2069 (70 έως 69). Εάν η παράμετρος μεγέθους οριστεί σε 4, το έτος μπορεί να είναι από 1901 έως 2155. Το προεπιλεγμένο μέγεθος είναι 4.

Κατά πάσα πιθανότητα θα χρησιμοποιούμε τους τύπους δεδομένων DATETIME ή DATE πιο συχνά από οποιονδήποτε άλλο σχετιζόμενο με ημερομηνίες/ώρες τύπο δεδομένων.

Τύποι Αλφαριθμητικών

Αν και οι τύποι δεδομένων για τους αριθμούς και τις τιμές ημερομηνίας/ώρας είναι χρήσιμοι τα περισσότερα δεδομένα που αποθηκεύουμε θα είναι σε μορφή αλφαριθμητικών (string).Η ακόλουθη λίστα περιγράφει τους πιο κοινούς τύπους δεδομένων για αλφαριθμητικά που υποστηρίζει η MySQL.

- **CHAR(M)** - Ενα αλφαριθμητικό σταθερού μεγέθους, με μέγεθος μεταξύ 1 και 255 χαρακτήρες (για παράδειγμα, CHAR(5)), στο οποίο προστίθενται κενά διαστήματα στα δεξιά μέχρι να φτάσει στο προσδιοριζόμενο μήκος. Ο ορισμός μήκους δεν είναι απαραίτητος, αλλά το προεπιλεγμένο μήκος είναι 1.

- **VARCHAR(M)** - Ενα αλφαριθμητικό μεταβλητού μεγέθους, με μήκος από 1 έως 255 χαρακτήρες (για παράδειγμα, VARCHAR (25)). Θα πρέπει να ορίζουμε την παράμετρο μήκους όταν δημιουργούμε ένα πεδίο VARCHAR.

- **BLOB ή TEXT** - Ενα πεδίο με μέγιστο μέγεθος 65535 χαρακτήρες. Ο όρος BLOB είναι ακρωνύμιο του Binary Large Objects, που σημαίνει μεγάλα αντικείμενα δυαδικής μορφής. Ο τύπος BLOB χρησιμοποιείται για την αποθήκευση μεγάλων ποσοτήτων δυαδικών δεδομένων, όπως για παράδειγμα εικόνες ή άλλοι τύποι αρχείων. Τα πεδία που ορίζονται με τύπο TEXT αποθηκεύουν επίσης μεγάλες ποσότητες δεδομένων· η διαφορά μεταξύ των δύο είναι ότι οι λειτουργίες ταξινόμησης και σύγκρισης που εκτελούνται στα αποθηκευμένα δεδομένα κάνουν διάκριση μεταξύ κεφαλαίων/πεζών χαρακτήρων για τα δεδομένα τύπου BLOB, ενώ αυτό δεν ισχύει για τα πεδία με τύπο δεδομένων TEXT. Οι τύποι δεδομένων BLOB και TEXT δεν απαιτούν από εμάς να καθορίσουμε μέγεθος.

- **TINYBLOB ή TINYTEXT** - Ενα πεδίο τύπου BLOB ή TEXT με μέγιστο μέγεθος 255 χαρακτήρες. Για τους τύπους TINYBLOB ή TINYTEXT δεν καθορίζουμε μέγεθος.

- **MEDIUMBLOB ή MEDIUMTEXT** - Ενα πεδίο τύπου BLOB ή TEXT με μέγιστο μέγεθος 16777215 χαρακτήρες. Για τους τύπους MEDIUMBLOB ή MEDIUMTEXT δεν καθορίζουμε μέγεθος.

- **LONGBLOB ή LONGTEXT** - Ενα πεδίο τύπου BLOB ή TEXT με μέγιστο μέγεθος 4294967295 χαρακτήρες. Για τους τύπους LONGBLOB ή LONGTEXT δεν καθορίζουμε μέγεθος.

- **ENUM** - Μία απαρίθμηση - δηλαδή, μία λίστα. Οταν ορίζουμε έναν τύπο ENUM - ουσιαστικά δημιουργούμε μία λίστα στοιχείων, από τα οποία πρέπει να επιλεχθεί μία τιμή (ή μπορεί να είναι NULL). Για παράδειγμα, εάν θέλαμε το πεδίο μας να περιέχει μία από τις τιμές **"A"**, "B", ή "C", θα έπρεπε να το ορίσουμε σαν ENUM ('A', 'B', 'C'), έτσι ώστε μόνο αυτές σι τιμές (ή η τιμή NULL) να μπορούν να εισαχθούν στο πεδίο. Ενας τύπος ENUM μπορεί να περιέχει έως 65535 διαφορετικές τιμές.

ΣΗΜΕΙΩΣΗ: Ο τύπος SET (σύνολο) είναι παρόμοιος με τον ENUM στο ότι ορίζεται σαν λίστα. Ωστόσο, ο τύπος SET αποθηκεύεται σαν μία πλήρης τιμή και όχι σαν ένας δείκτης προς μία τιμή, όπως ισχύει με τους τύπους απαρίθμησης.

Κατά πάσα πιθανότητα θα χρησιμοποιούμε πεδία με τύπους δεδομένων VARCHAR και TEXT πιο συχνά από τους άλλους τύπους δεδομένων.

2.2.2 Η Εντολή για την Δημιουργία Πινάκων

Η εντολή για την δημιουργία ενός πίνακα απαιτεί τα ακόλουθα στοιχεία:

- Το όνομα του πίνακα
- Τα ονόματα των πεδίων
- Εναν ορισμό για κάθε πεδίο

Η γενική σύνταξη για την εντολή δημιουργίας πίνακα είναι:

```
CREATE TABLE όνομα_πίνακα (όνομα_στήλης τύπος_στήλης);
```

Το όνομα του πίνακα είναι δική μας επιλογή, αλλά θα πρέπει να αντικατοπτρίζει την χρήση ή τον ρόλο του πίνακα. Παρόμοια, τα ονόματα των πεδίων θα πρέπει να είναι όσο το δυνατόν πιο περιγραφικά και σχετιζόμενα με τον ρόλο των πεδίων ή με τα δεδομένα που αποθηκεύουν.

Η βάση δεδομένων με την οποία θα ασχοληθούμε στα επόμενα κεφάλαια σχετίζεται με το πρόγραμμα μαθημάτων του τμήματος ΕΠΔΟ του ΤΕΙ Μεσολογγίου.Πριν δημιουργήσουμε τους πίνακες,θα δημιουργήσουμε πρώτα τη βάση μας.Η σύνταξή της είναι η εξής:

CREATE DATABASE όνομα_βάσης ;

Δηλαδή **mysql> create database programma;**

Ένα πρόγραμμα μαθήμάτων αποτελείται από τις ώρες και ημέρες, το μάθημα, το είδος μαθήματος (θεωρία, ασκήσεις/πράξεις ,εργαστήριο), τον καθηγητή που το διδάσκει και την αίθουσα οπου γίνεται το μάθημα.Γι'αυτό το λόγο στη βάση δεδομένων μας με το όνομα programma, θα δημιουργήσουμε πίνακες που κρατάνε τις παραπάνω πληροφορίες με τα επιμέρους πεδία τους.

Δημιουργούμε πίνακα με το όνομα μάθημα(**mathima**).Τα πεδία από τα οποία αποτελείται είναι 1) ο κωδικός μαθήματος (**id_math**),όπου τον δηλώνουμε ως int διότι θα κρατάει ακέραιες αριθμητικές τιμές,not null που σημαίνει μη κενό πεδίο και auto_increment που σημαίνει αυτόματη αύξηση τιμής 2) ο τίτλος μαθήματος (**titlos**), που τον δηλώνουμε ως αλφαριθμητικό (varchar) με μέγεθος χαρακτήρων 30 και μη κενό πεδίο(not null) 3) το εξάμηνο στο οποίο ανήκει το μάθημα (**exam**), που το δηλώνουμε ως αλφαριθμητικό (varchar) με μέγεθος χαρακτήρων 10 και μη κενό πεδίο(not null) 4) πόσες ώρες είναι η θεωρία,οι ασκήσεις/πράξης, το εργαστήριο (**thew**),(**ask**),(**erg**). Σ'αυτά τα πεδία κρατάμε ακέραιες σριθμητικές τιμές γιάυτό τα δηλώνουμε ως int και το πεδίο thew ως μη κενό. 7) Και το σύνολο ωρών(**synolo**), που κρατάει τις συνολικές ώρες του μαθήματος και το δηλώνουμε ως int και not null.Επίσης το πεδίο για τον κωδικό μαθήματος (id_math) το ορίζουμε σαν πρωτεύον κλειδί (Primary key).

```
mysql>create table mathima(
        ->id_math int not null auto_increment,
        ->titlos varchar(30) not null,
        ->exam varchar(10) not null,
        ->thew int not null,
        ->ask int,
        ->erg int,
        ->synolo int not null,
        ->Primary key (id_math));
Query OK, 0 rows affected ( 0.02  sec)
```

Δημιουργούμε πίνακα με το όνομα είδος μαθήματος (**eidos_mathimatos**).Τα πεδία από τα οποία αποτελείται είναι 1) ο κωδικός είδους μαθήματος (**id_eidous**),όπου τον δηλώνουμε ως int διότι θα κρατάει ακέραιες αριθμητικές τιμές, not null και auto_increment που σημαίνει αυτόματη αύξηση τιμής, σαν χαρακτηριστικό του πεδίου,λέμε στην MySQL να εισάγει αυτόματα τον επόμενο διαθέσιμο αριθμό στο πεδίο id_eidous κάθε εγγραφής που εισάγουμε. 2) το είδος μαθήματος (**eidos_mathimatos**), που το δηλώνουμε ως αλφαριθμητικό (varchar) με μέγεθος χαρακτήρων 20. Σαν πρωτεύον κλειδί (Primary key) στον πίνακα αυτό ορίζουμε τον κωδικό είδους (**id_eidous**).

```
mysql>create table eidos_mathimatos(
        ->id_eidous int not null auto_increment,
        ->eidos_mathimatos varchar(20),
        ->Primary key (id_eidous));
Query OK, 0 rows affected ( 0.02  sec)
```

Παρόμοια, θα δημιουργήσουμε πίνακα με όνομα εκπαιδευτικος (**ekpaideutikos**).Αποτελείται από τα πεδία κωδικός εκπαιδευτικού (**id_ekpaid**), ονοματεπώνυμο (**onomatepwnymo**), το e_mail του καθηγητή (**e_mail**) και τον κωδικό κατηγορίας του εκπαιδευτικού δηλαδή σε πια βαθμίδα ανήκει (**id_kathg_ekpaid**).Ορίζουμε σαν πρωτεύων κλειδί (**Primary key**) τον κωδικό του εκπαιδευτικου (**id_ekpaid**) και σαν ξένο κλειδί (**Foreign key**) τον κωδικό κατηγορίας εκπαιδευτικού(**id_kathg_ekpaid**) διότι το πεδίο αυτό αναφέρεται ως πρωτεύων κλειδί στον πίνακα κατηγορία εκπαιδευτικού γιάυτό ορίζεται ως ξένο.

```
mysql>create table ekpaideutikos(
        ->id_ekpaid int not null auto_increment,
        ->onomatepwnymo varchar(30) not null,
        ->e_mail varchar(30),
        ->id_kathg_ekpaid int not null,
        ->Primary key (id_ekpaid),
        ->index (id_kathg_ekpaid),
        ->Foreign key (id_kathg_ekpaid),
        ->References kathgoria_ekpaid (id_kathg_ekpaid));
Query OK, 0 rows affected ( 0.02  sec)
```

Ο πίνακας κατηγορία εκπαιδευτικού(**kathgoria_ekpaid**) αναφέρεται στην βαθμίδα των εκπαιδευτικών δηλαδή αν είναι επιστημονικοί με πλήρη προσόντα,αναπληρωτές καθηγητές ,καθηγητές εφαρμογών κ.τ.λ.

```
mysql>create table kathgoria_ekpaid(
        ->id_kathg_ekpaid int not null auto_increment,
        ->kathgoria varchar(30) not null,
        ->Primary key (id_kathg_ekpaid));
Query OK, 0 rows affected ( 0.02  sec)
```

Ο πίνακας ώρες (**wres**) αναφέρεται στο ωρολόγιο εβδομαδιαίο πρόγραμμα μαθημάτων με πεδία τον κωδικό της ώρας (**id_wras**),την ώρα (**wra**) που τη δηλώνουμε ως κειμενο(**text**), την ημέρα (**hmera**) και ορίζουμε ως πρωτεύων κλειδί(**Primary key**) τον κωδικό της ώρας(**id_wras**).

```
mysql>create table wres(
        ->id_wras int not null auto_increment,
        ->wra text,
         ->hmera varchar(10),
         ->Primary key (id_wras));
Query OK, 0 rows affected ( 0.02  sec)
```

Δημιουργούμε πίνακα με όνομα αίθουσα (**ai8ousa**) ο οποίος περιέχει ως πεδία τον κωδικό της αίθουσας (**id_ai8ousas**), την αίθουσα(**ai8ousa**) και το πεδίο για τον κωδικό της αίθουσας ορίζεται σαν πρωτεύων κλειδί .

```
mysql>create table ai8ousa(
         ->id_ai8ousas int not null auto_increment,
         ->ai8ousa varchar(10) not null,
         ->Primary key (id_ai8ousas));
```

```
Query OK, 0 rows affected ( 0.02  sec)
```

Για να γίνει συσχέτηση μεταξύ των πινάκων της βάσης δεδομένων μας programma ,δημιουργούμε έναν πίνακα που τον ονομάζουμε **endiamesos** ,ο οποίος έχει σαν πεδία τους κωδικούς όλων των προηγούμενων πινάκων που δημιουργήσαμε, δηλαδή **id_math, id_ekpaid, id_eidous, id_wras, id_ai8ousas.** Έχει σαν πρωτεύωντα κλειδιά όλα τα πεδία του και ξένο κλειδί τον κωδικό εκπαιδευτικού**(id_ekpaid),**του πίνακα ekpaideutikos.

```
mysql>create table endiamesos(
          ->id_math int,
          ->id_ekpaid int,
          ->id_eidous int,
          ->id_wras int,
          ->id_ai8ousas int,
          ->Primary key (id_math, id_ekpaid, id_eidous,
id_wras, id_ai8ousas),
          ->Foreign key (id_ekpaid),
          ->References ekpaideutikos (id_ekpaid));
Query OK, 0 rows affected ( 0.02  sec)
```

ΣΗΜΕΙΩΣΗ : Ο MySQL server ανταποκρίνεται με το μήνυμα Query OK κάθε φορά που εκτελείται επιτυχώς ένα ερώτημα, ανεξαρτήτως του τύπου του. Σε κάθε άλλη περίπτωση εμφανίζεται ένα μήνυμα σφάλματος.

2.2.3 PHPMyadmin, διαχείριση της MySQL μέσω του web

Τί είναι η phpMyAdmin;

Η phpMyAdmin είναι ένα παραστατικό σύστημα για την διαχείριση βάσεων δεδομένων σε MySQL. Είναι γραμμένη σε PHP και εξυπηρετεί στο να επιδείξει τα περιεχόμενα των βάσεων δεδομένων στον server (ή πελάτη) πάνω στον οποίο η MySQL έχει εγκατασταθεί. Εξ'αιτίας αυτής της διασύνδεσης μπορούμε να

δημιουργήσουμε νέες βάσεις δεδομένων, να τροποποιήσουμε υπάρχουσες βάσεις και να τροποποιήσουμε τα περιεχόμενα των πεδίων.

Σε περίπτωση που θέλουμε να την εγκαταστήσουμε στα Windows για να διαβάζουμε τις βάσεις δεδομένων οι οποίες βρίσκονται online, θα πρέπει να σχηματίσουμε τα πάντα επεξεργάζοντας το αρχείο config.inc.php όπως παρακάτω:

Έστω ότι:

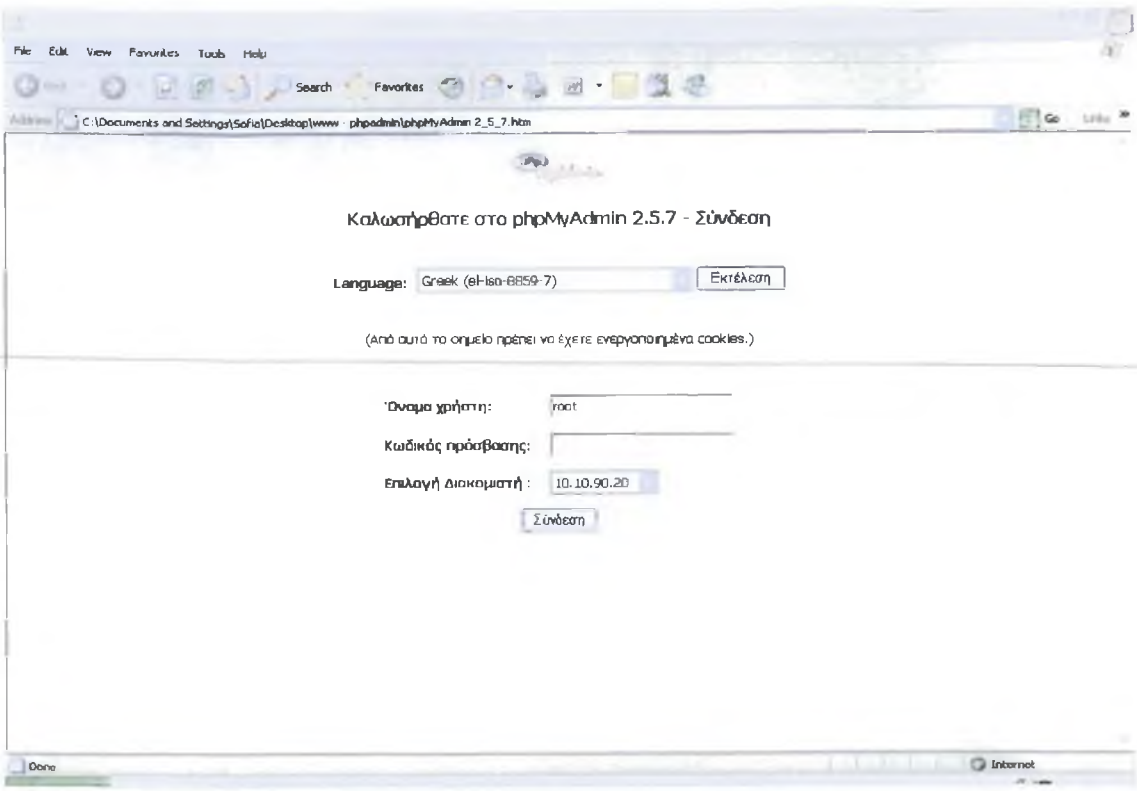
- IP DB Server: 10.10.90.20
- User: root
- Password: 0000
- Database Name: programma

Τότε:

```
$cfgServers[1]["host"] = "10.10.90.20"; // MySQL hostname
$cfgServers[1]["port"] = ""; // MySQL port - leave blank for default
port
$cfgServers[1]["adv_auth"] = false; // Use advanced authentication?
$cfgServers[1]["stduser"] = ""; // MySQL standard user (only needed
with advanced auth)
$cfgServers[1]["stdpass"] = ""; // MySQL standard password (only
needed with advanced auth)
$cfgServers[1]["user"] = "root"; // MySQL user (only needed with
basic auth)
$cfgServers[1]["password"] = "0000"; // MySQL password (only needed
with basic auth)
$cfgServers[1]["only_db"] = "programma"; // If set to a db-name,
only this db is accessible
$cfgServers[1]["verbose"] = ""; // Verbose name for this host -
leave blank to show the hostname
$cfgServers[1]["bookmarkdb"] = ""; // Bookmark db - leave blank for
no bookmark support
$cfgServers[1]["bookmarktable"] = ""; // Bookmark table - leave
blank for no bookmark support
```

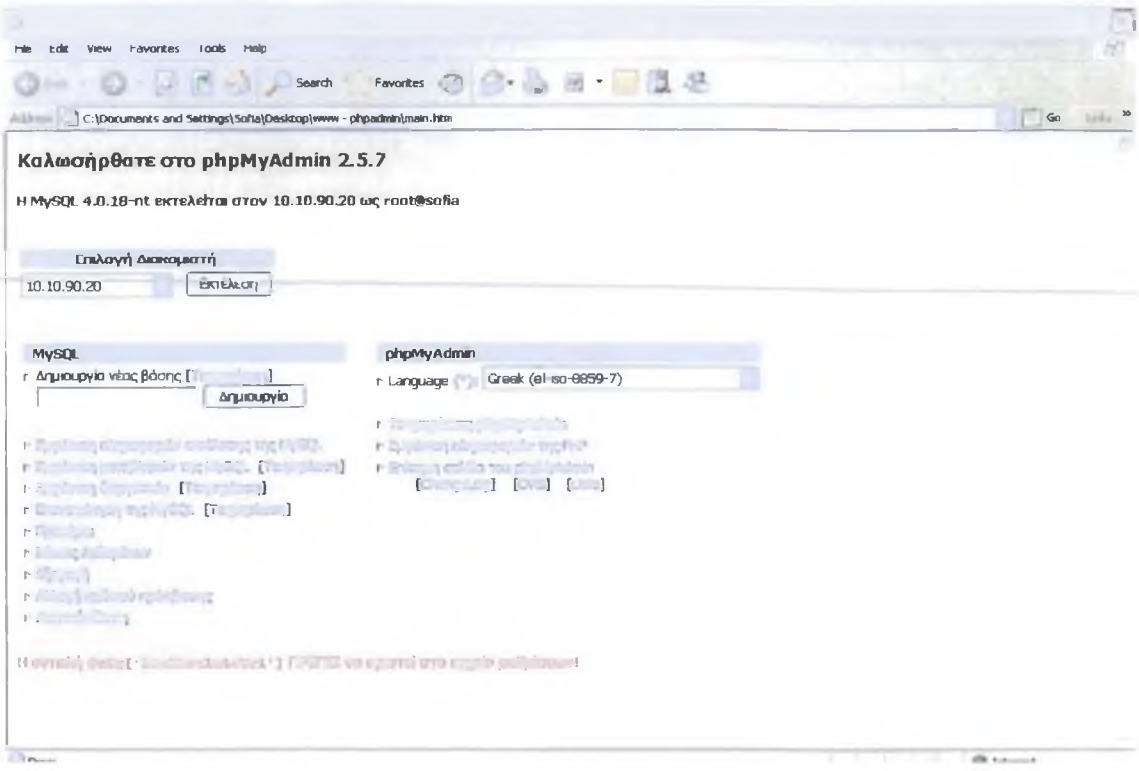
Στο αρχείο config.inc.php θα βρούμε πολλές παραμέτρους οι οποίες επαναλαμβάνονται . Αυτές εξυπηρετούν να διαχειρίζονται τις DBs σε πολλούς hosts με την ίδια διασύνδεση.

- Τώρα είμαστε έτοιμοι να μπούμε στο περιβάλλον της phpMyAdmin απο την διεύθυνση http//:localhost/ phpMyAdmin . Όπως φαίνεται στην παρακάτω εικόνα δηλώνουμε το όνομα χρήστη(root) και τον κωδικό πρόσβασης για να συνδεθούμε:



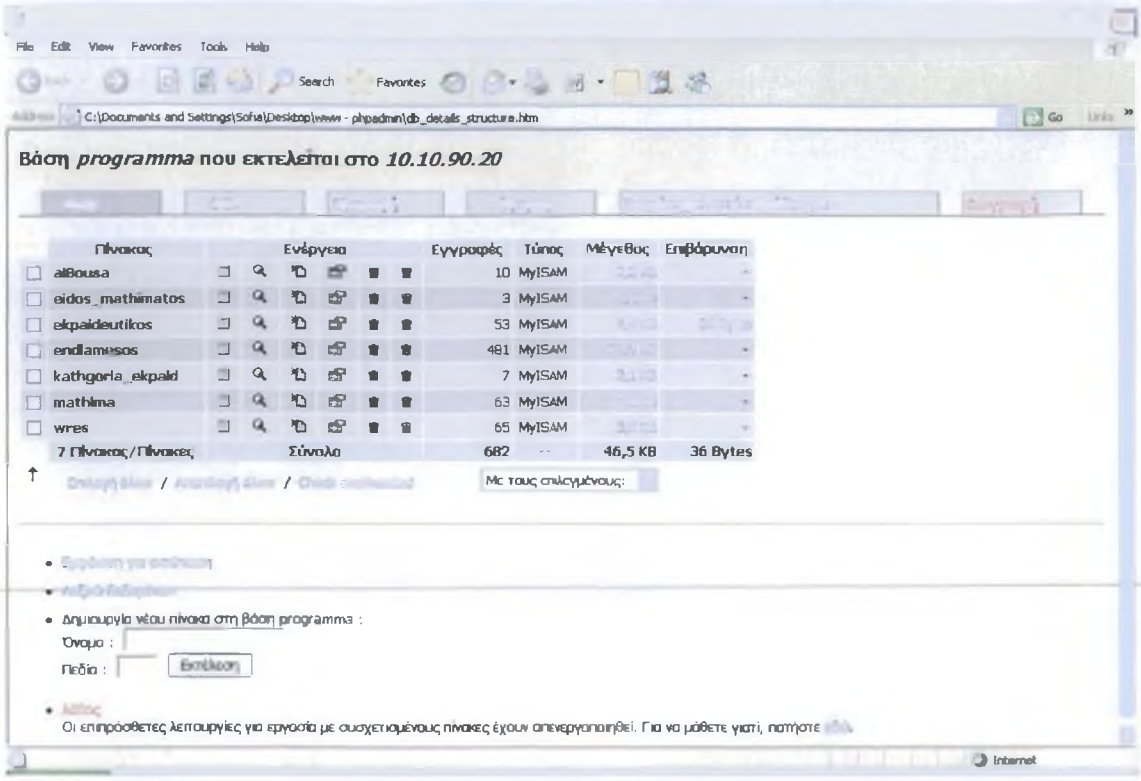
ΕΙΚ. Πρόσβαση στην phpmyadmin

- Πολύ εύκολα τώρα μπορούμε να δημιουργήσουμε τη βάση δεδομένων μας, δηλώνοντας το όνομα της βάσης, που είναι το programma στο παράδειγμά μας, απλά πατώντας το κουμπί Δημιουργία, όπως φαίνεται παρακάτω:



ΕΙΚ. Δημιουργία νέας βάσης δεδομένων

- Δημιουργούμε τους πίνακες δηλώνοντας το όνομα και τον αριθμό των πεδίων και πατώντας το κουμπί Εκτέλεση μπορούμε να δηλώσουμε τον τύπο των πεδίων.Τώρα είμαστε έτοιμοι να επεξεργαστούμε τους πίνακες, να κάνουμε INSERT, DELETE,UPDATE κ.τ.λ



ΕΙΚ. Δημιουργία νέου πίνακα

Όπως βλέπουμε η phpMyAdmin είναι πολύ εύκολή στην χρήση,χρησιμοποιούνται οι ίδιες εντολές της MySQL ,το μόνο που αλλάζει είναι το περιβάλλον επεξεργασίας .

2.2.4 Χρήση της Εντολής INSERT

Αφού δημιουργήσουμε μερικούς πίνακες, θα χρησιμοποιήσουμε την εντολή INSERT της SQL για να προσθέσουμε νέες εγγραφές σ'αυτούς .Η βασική σύνταξη της εντολής INSERT είναι:

```
INSERT INTO όνομα_πίνακα(λίστα στηλών) VALUES (τιμές στηλών);
```

Στην λίστα των τιμών μέσα στις παρενθέσεις τα αλφαριθμητικά θα πρέπει να περικλείονται σε εισαγωγικά. Η SQL χρησιμοποιεί εξ ορισμού αποστρόφους, αλλά η MySQL μας επιτρέπει να επιλέξουμε εάν θα χρησιμοποιήσουμε αποστρόφους ή εισαγωγικά. Δεν πρέπει να ξεχνάμε ότι θα πρέπει να χρησιμοποιούμε τον χαρακτήρα \ πριν από τα εισαγωγικά που περιλαμβάνονται στο ίδιο το αλφαριθμητικό.

ΣΗΜΕΙΩΣΗ: Οταν εισάγουμε ακέραιες τιμές, δεν χρειάζεται να τις περικλείουμε σε εισαγωγικά ή αποστρόφους.

Ακολουθεί ένα παράδειγμα αλφαριθμητικού στο οποίο είναι απαραίτητη η χρήση του χαρακτήρα \:

O'Connor said "Boo"

Εάν περικλείουμε τα αλφαριθμητικά μας σε εισαγωγικά, η εντολή INSERT θα δείχνει ως εξής:

```
INSERT INTO table_name (column_name)  VALUES ("O'Connor
said \ "Boo\" ");
```

Εάν περικλείουμε τα αλφαριθμητικά μας σε αποστρόφους, η εντολή INSERT θα δείχνει ως εξής:

```
INSERT INTO table_name (column_name)  VALUES ('O\ 'Connor
said "Boo\" ');
```

Αναλυτική Παρουσίαση της Εντολής INSERT

Εκτός από το όνομα του πίνακα, δύο είναι τα βασικά μέρη της εντολής INSERT - η λίστα των στηλών και η λίστα των τιμών. Στην πραγματικότητα, η μόνη απαιτούμενη είναι η λίστα των τιμών, αλλά εάν παραλείψουμε την λίστα των στηλών, θα πρέπει να κατονομάζουμε ρητά κάθε στήλη με την σειρά στην λίστα των τιμών.

Χρησιμοποιώντας σαν παράδειγμα το πρόγραμμα μαθημάτων του τμήματος ΕΠΔΟ ,ο πίνακας mathima, έχει επτά πεδία: τα id_math, titlos, exam, thew, ask, erg, synolo.Για να εισάγουμε μία πλήρη εγγραφή σ' αυτό τον πίνακα, θα μπορούσαμε να χρησιμοποιήσουμε οποιαδήποτε από τις ακόλουθες εντολές:

1. Μία εντολή στην οποία κατονομάζονται όλες οι στήλες:

```
insert into mathima(id_math, titlos, exam, thew,
ask, erg, synolo) values ('1', 'Μαθηματικά
I','A','3','2','0','5');
```

2. Μία εντολή η οποία χρησιμοποιεί όλες τις στήλες, αλλά δεν τις κατονομάζει ρητά:

```
insert into mathima values ('2', 'Αρχές
Δικαίου','A','2','2','0','4');
```

Δοκιμάζουμε και τις δύο εντολές για να δούμε τι θα συμβεί. Θα πρέπει να πάρουμε αποτελέσματα παρόμοια με τα ακόλουθα:

```
mysql> insert into mathima
        ->(id_math, titlos, exam, thew, ask, erg,
synolo)
        ->values
        ->('1', 'Μαθηματικά Ι','Α','3','2','0','5');
Query OK, 1 rows affected ( 0.01  sec)

mysql>insert into mathima values ('2','Αρχές
Δικαίου','Α','2','2','0','4');
Query OK, 1 rows affected ( 0.01  sec)
```

Ας προχωρήσουμε τώρα σε ορισμένες πιο ενδιαφέρουσες μεθόδους χρήσης της εντολής INSERT. Επειδή η τιμή του πεδίου id_math είναι ένας αυτόματα αυξανόμενος ακέραιος, δεν χρειάζεται να τον συμπεριλάβουμε στην λίστα τιμών. Ωστόσο, εάν υπάρχει κάποια τιμή την οποία δεν θέλουμε να αναγράψουμε στην λίστα τιμών της INSERT (όπως η τιμή του πεδίου id_math στην περίπτωσή μας), θα πρέπει να αναφέρουμε όλες τις υπόλοιπες στήλες που χρησιμοποιούμε. Για παράδειγμα, επειδή η ακόλουθη εντολή δεν κατονομάζει τις στήλες και δεν περιλαμβάνει τιμή για το πεδίο id_math, Θα παράγει ένα σφάλμα:

```
mysql>insert into mathima values
        ->('Αγγλικά Ι','Α','2','2','0','4');
ERROR 1136: Column count doesn't match value count
at row 1
```

Επειδή δεν αναφέραμε ρητά τα ονόματα των στηλών, η MySQL αναμένει ότι έχουμε εισάγει τιμές για όλες τις στήλες στην λίστα τιμών της εντολής, πράγμα το οποίο προκαλεί μία κατάσταση σφάλματος στην προηγούμενη εντολή που παρουσιάσαμε. Εάν ο στόχος μας ήταν να αφήσουμε την MySQL να κάνει μόνη της αυτή την δουλειά αυξάνοντας αυτόματα την τιμή του πεδίου id_math, θα έπρεπε να χρησιμοποιήσουμε μία από τις ακόλουθες εντολές:

1. Μία εντολή στην οποία κατονομάζονται όλες σι στήλες εκτός της id_math

```
mysql> insert into mathima
        ->( titlos, exam, thew, ask, erg, synolo)
        ->values
```

```
->('Μακροοικονομική','B,'2','2','0','4');
```

2. Μία εντολή η οποία χρησιμοποιεί όλες τις στήλες, αλλά δεν τις κατονομάζει ρητά, και περιλαμβάνει την καταχώριση NULL για να υποδείξει την απουσία τιμής για το πεδίο id_math

```
mysql>insert into mathima values

        ->('NULL' 'Προγραμματισμός Η/Υ
II','B','2','0','2','4');
```

Χρησιμοποιούμε μία από τις δύο παραπάνω μορφές της εντολής για τον πίνακα mathima.Το ποια μορφή της εντολής θα επιλέξουμε δεν έχει σημασία για την MySQL, αλλά όπως ισχύει και με οτιδήποτε άλλο το οποίο επαφίεται στην προσωπική μας προτίμηση, θα πρέπει να επιλέξουμε μία μορφή και να παραμείνουμε σταθεροί σ' αυτήν καθ'όλη την διάρκεια ανάπτυξης της εφαρμογής μας. Η συνέπεια και η ομοιομορφία στις συμβάσεις και στις δομές που χρησιμοποιούμε είναι δύο πλεονεκτήματα τα οποία θα μας διευκολύνουν μελλοντικά κατά την αποσφαλμάτωση της εφαρμογής, επειδή θα ξέρουμε τι πρέπει να περιμένουμε σε κάθε περίπτωση.

2.2.5 Χρήση της Εντολής SELECT

Η SELECT είναι η εντολή της SQL που χρησιμοποιείται για την ανάκτηση εγγραφών από πίνακα. Η σύνταξη αυτής της εντολής μπορεί να είναι εντελώς απλή, ή εξαιρετικά πολύπλοκη. Καθώς θα εξοικειωνόμαστε όλο και περισσότερο με τον προγραμματισμό εφαρμογών βάσεων δεδομένων, θα μάθουμε πώς να εμπλουτίζουμε τις εντολές SELECT σε τόσο μεγάλο βαθμό, έτσι ώστε τελικά ένα μεγάλο μέρος της απαιτούμενης εργασίας να γίνεται από την ίδια την βάση δεδομένων, χωρίς να επιβαρυνόμαστε με συγγραφή πρόσθετου κώδικα.

Στην απλούστερη μορφή της, η σύνταξη της SELECT δείχνει κάπως έτσι:

```
SELECT εκφράσεις_και_στήλες FROM όνομα_πίνακα
[WHERE συνθήκη]
[ORDER BY στήλη [ASC | DESC ] ]
[LIMIT μετάθεση, γραμμές]
```

Ας ξεκινήσουμε με την πρώτη γραμμή:

```
SELECT εκφράσεις_και_στήλες FROM όνομα_πίνακα
```

Μία ιδιαίτερα βολική έκφραση είναι το σύμβολο * το οποίο αντιπροσωπεύει "τα πάντα". Συνεπώς, για να επιλέξουμε "τα πάντα" (όλες τις γραμμές και όλες τις

στήλες) από τον πίνακα mathima, η εντολή SELECT θα έπρεπε να έχει την ακόλουθη μορφή:

```
SELECT*FROM mathima;
```

Ανάλογα με την ποσότητα των δεδομένων που εισάγαμε στον πίνακα mathima παραπάνω, τα αποτελέσματα που θα πάρουμε μπορεί να διαφέρουν, αλλά σε γενικές γραμμές θα πρέπει να δείχνουν παρόμοια με την ακόλουθη λίστα:

mysql> select*from mathima;

```
+----------+---------------------------+---------+---------+--------+---------+-------------+
| id_math  |          titlos           |  exam   |  thew   |  ask   |  erg    |  synolo     |
+----------+---------------------------+---------+---------+--------+---------+-------------+
|    1     | Μαθηματικά I              |  A      |  3      |  2     |  0      |  5          |
|    2     | Αρχές Δικαίου             |  A      |  2      |  2     |  0      |  4          |
|    3     | Διοίκηση Επιχειρήσεων     |  A      |  2      |  2     |  0      |  4          |
+----------+---------------------------+---------+---------+--------+---------+-------------+
```

Οπως βλέπουμε, το αποτέλεσμα που παράγει η MySQL είναι ένας αρκετά ελκυστικός πίνακας με τα ονόματα των στηλών στην πρώτη γραμμή. Εάν θέλουμε να επιλέξουμε μόνο συγκεκριμένες στήλες, αντικαθηστούμε τον χαρακτήρα * με τα ονόματα των επιθυμητών στηλών διαχωρισμένο με κόμματα. Η ακόλουθη εντολή επιλέγει μόνο τα πεδία id_math, titlos, exam, synolo από τον πίνακα mathima.

id_math	titlos	exam	synolo
1	Μαθηματικά I	A	5
2	Αρχές Δικαίου	A	4
3	ΔιοίκησηΕπιχειρήσεων	A	4
4	Αγγλικά I	A	3
5	Μικροοικονομική	A	4
6	Εισαγωγή στην Πληροφο	A	2
7	Προγραμματισμός Η/ΥΙ	A	5
8	Ευρωπαι'κή Ένωση&Επι	A	2
9	Πληροφορική&Κοινωνία	A	2

9 rows in set (0.00 sec)

Ο πίνακας mathima αποτελείται απο 63 εγγραφές, όσα και τα μαθήματα που διδάσκονται στο τμήμα της ΕΠΔΟ.Ο πίνακας που παρουσιάσαμε παραπάνω είναι ένα δείγμα αυτού ώστε να κατανοήσουμε τις εντολές που χρησιμοποιούμε στη MySQL.

Ταξινόμηση των Αποτελεσμάτων που Επιστρέφει η Εντολή SELECT

Εξ ορισμού, τα αποτελέσματα των ερωτημάτων που εκτελούμε με την εντολή SELECT σε έναν πίνακα μιας βάσης δεδομένων εμφανίζονται με την σειρά με την οποία εισάγαμε τις εγγραφές στον πίνακα, χωρίς κάποιο είδος ταξινόμησης. Εάν θέλουμε να ταξινομήσουμε τα αποτελέσματα με έναν συγκεκριμένο τρόπο, όπως π.χ. κατά ημερομηνία, κωδικό, όνομα, κ.λ.π., θα πρέπει να καθορίσουμε αυτή την απαίτησή μας στην εντολή χρησιμοποιώντας την πρόταση ORDER BY. Στο αποτέλεσμα της ακόλουθης εντολής, σι εγγραφές θα εμφανιστούν ταξινομημένες κατά τίτλο μαθήματος (titlos) του πίνακα mathima:

```
mysql> select id_math, titlos, exam, synolo from mathima
    -> order by titlos;
```

1	Αγγλικά Ι	Α	3
2	Αρχές Δικαίου	Α	4
3	ΔιοίκησηΕπιχειρήσεων	Α	4
6	Εισαγωγή στην Πληροφο	Α	2
8	Ευρωπαι'κή Ένωση&Επι	Α	2
4	Μαθηματικά Ι	Α	5
5	Μικροοικονομική	Α	4
9	Πληροφορική&Κοινωνία	Α	2
7	Προγραμματισμός Η/ΥΙ	Α	5

:Οταν επιλέγουμε εγγραφές από έναν πίνακα χωρίς να καθορίσουμε μία σειρά ταξινόμησης, τα αποτελέσματα του ερωτήματός μας μπορεί να εμφανιστούν ταξινομημένα βάσει της τιμής του πεδίου-κλειδιού, αλλά μπορεί και όχι. Αυτό συμβαίνει επειδή η MySQL επαναχρησιμοποιεί τον χώρο που καταλάμβαναν σι διαγραμμένες γραμμές (εγγραφές). Με άλλα λόγια, εάν προσθέσουμε εγγραφές με κωδικούς *(id_math)* από 1 έως 5, διαγράψουμε την εγγραφή με κωδικό 4 και κατόπιν προσθέσουμε μία επόμενη εγγραφή (η οποία θα λάβει τον κωδικό 6), οι εγγραφές θα εμφανίζονται στον πίνακα μετην εξής σειρά: 1,2,3,6,5.

Η προεπιλεγμενη σειρά ταξινόμησης που χρησιμοποιεί η πρόταση **ORDER BY** είναι αύξουσα (ASC), τα αλφαριθμητικά ταξινομούνται αλφαβητικά, από το A ως το Z, σι ακέραισι αυξητικά ξεκινώντας από το 0 και σι ημερομηνίες από την παλαιότερη έως την νεότερη. Μπορούμε επίσης να καθορίσουμε φθίνουσα σειρά ταξινόμησης χρησιμοποιώντας την δεσμευμένη λέξη **DESC .**

mysql> select id_math, titlos, exam, synolo from mathima
-> order by titlos desc;

7	Προγραμματισμός Η/ΥΙ	A	5
9	Πληροφορική&Κοινωνία	A	2
2	Μικροοικονομική	A	4
1	Μαθηματικά Ι	A	5
8	Ευρωπαι'κή Ένωση&Επι	A	2
6	Εισαγωγή στην Πληροφο	A	2
3	ΔιοίκησηΕπιχειρήσεων	A	4
5	Αρχές Δικαίου	A	4
4	Αγγλικά Ι	A	3

Δεν περιοριζόμαστε σε ταξινόμηση βάσει ενός μόνο πεδίου - μπορούμε να καθορίσουμε όσα πεδία θέλουμε, διαχωρίζοντας τα ονόματά τους με κόμματα. Η σειρά ταξινόμησης αντικατοπτρίζει την σειρά με την οποία κατονομάζουμε τα πεδία.

Περιορίζοντας τα Αποτελέσματα που Παίρνουμε με την SELECT

Μπορούμε να χρησιμοποιήσουμε την πρόταση **LIMIT** για να ανακτάμε μόνο έναν συγκεκριμένο αριθμό εγγραφών στο αποτέλεσμα ενός ερωτήματος που εισάγαμε με την εντολή SELECT. Δύο είναι σι παράμετροι που πρέπει να καθορίζουμε όταν χρησιμοποιούμε την πρόταση LIMIT: η μετάθεση (offset) και το πλήθος των γραμμών. Η μετάθεση είναι η θέση έναρξης· το πλήθος των γραμμών είναι μάλλον αυτονόητο.

ΥΠΟΔΕΙΞΗ: Στην πλειοψηφία των περιπτώσεων, η μέτρηση στον προγραμματισμό ξεκινά από το 0 και όχι από το 1. Για παράδειγμα: 0, 1, 2, 3 αντί 1, 2, 3,4.

Υποθέστε ότι έχουμε περισσότερες από 2-3 εγγραφές στον πίνακα mathima και θέλουμε να επιλέξουμε τον κωδικό μαθήματος , τον τίτλο, το εξάμηνο και το σύνολο ωρών για τις πρώτες τρεις εγγραφές και να τις εμφανίσουμε ταξινομημένες βάσει συνόλου ωρών *(synolo)*. Με άλλα λόγια,θέλουμε να επιλέξουμε τις τρεις εγγραφές με το μικρότερο σύνολο ωρών. Η ακόλουθη πρόταση LIMIT καθορίζει ότι η επιλογή θα ξεκινήσει από την αρχή του πίνακα και θα φτάσει μέχρι την τρίτη εγγραφή:

```
mysql> select id_math, titlos, exam, synolo from mathima
    -> order by synolo limit 3;
```

2	Αρχές Δικαίου	A	4
3	ΔιοίκησηΕπιχειρήσεων	A	4
1	Μαθηματικά I	A	5

3 rows in set (0.00 sec)

Η πρόταση LIMIT μπορεί να αποδειχθεί ιδιαίτερα χρήσιμη σε μία πραγματική εφαρμογή. Για παράδειγμα, μπορούμε να χρησιμοποιήσουμε την πρόταση LIMIT σε μία σειρά εντολών SELECT για να εξετάζουμε τα αποτελέσματα "σελίδα προς σελίδα":

```
1. SELECT * FROM mathima ORDER BY synolo LIMIT 0 , 3;

2. SELECT * FROM mathima ORDER BY synolo LIMIT 3 , 3;

3. SELECT * FROM mathima ORDER BY synolo LIMIT 6 , 3;
```

Εάν καθορίσουμε μετάθεση και αριθμό γραμμών στο ερώτημά μας και δεν βρεθούν αποτελέσματα, δεν θα δούμε ένα μήνυμα σφάλματος - επιστρέφεται απλώς ένα κενό σύνολο εγγραφών *(empty set)*. Για παράδειγμα, εάν ο πίνακας mathima περιείχε μόνο 6 εγγραφές, το ερώτημα στο οποίο η πρόταση LIMIT έχει τιμή 6 στην παράμετρο μετάθεσης δεν θα παράγει αποτελέσματα:

```
mysql> select id_math, titlos, exam, synolo from mathima
    -> order by synolo limit 6 , 3;

Empty set (0.00 sec)
```

Στις βασιζόμενες στο Web εφαρμογές, όταν εμφανίζονται λίστες δεδομένων με συνδέσεις όπως οι "previous 10" και "next 10", μπορούμε εκ του ασφαλούς να υποθέσουμε ότι χρησιμοποιείται ένα ερώτημα με την πρόταση LIMIT.

2.2.6 Χρήση της Πρότασης WHERE στα Ερωτήματά μας

Μέχρι τώρα μάθαμε αρκετούς τρόπους για να ανακτάμε συγκεκριμένες στήλες από τους πίνακες της βάσης δεδομένων μας, αλλά όχι συγκεκριμένες γραμμές (εγγραφές). Αυτή ακριβώς την ανάγκη έρχεται να καλύφει η πρόταση **WHERE**. Από την βασική σύνταξη της SELECT, μπορούμε να δούμε ότι η πρόταση WHERE χρησιμοποιείται για τον καθορισμό μιας συγκεκριμένης συνθήκης:

```
SELECT εκφράσεις_και_στήλες FROM όνομα_πίνακα
[WHERE συνθήκη]
```

Ενα παράδειγμα θα ήταν η ανάκτηση των εγγραφών όλων των εκπαιδευτικών του τμήματος ΕΠΔΟ, οι οποίοι είναι Επίκουροι καθηγητές.Η βαθμίδα του εκπαιδευτικού έχει δηλωθεί στον πίνακα kathg_ekpaid από κάποιον κωδικό, όπως φαίνεται παρακάτω:

id_kathg_ekpaid	kathgoria
1	ΕΠ(Π)
2	ΕΠ(Ε)
3	ΕΡ(Π)
4	ΕΡ(Ε)
5	ΑΝ.Κ.
6	ΕΠ.Κ.
7	Κ.ΕΦ.

Άρα αναζητούμε όλους τους εκπαιδευτικούς οι οποίοι έχουν κωδικό κατηγορίας εκπαιδευτικού 6, δηλαδή είναι Επίκουροι καθηγητές.

mysql> select * from ekpaideutikos where id_kathg_ekpaid = 6;

id_ekpaid	onomatepwnymo	e_mail	id_kathg_ekpaid
2	**Μαυριδάκης Θεοφάνης**		**6**
4	**Μακρής Χρήστος**		**6**
5	**Συρμακέσης Σπύρος**		**6**

3 rows in set (0.00 sec)

Στο προηγούμενο παράδειγμα χρησιμοποιούμε έναν ακέραιο στην συνθήκη της πρότασης WHERE. Οταν χρησιμοποιούμε ακέραιες τιμές, δεν χρειάζεται να τις περικλείουμε σε εισαγωγικά. Τα εισαγωγικά είναι απαραίτητα μόνο για τα αλφαριθμητικά, όπου ισχύουν επίσηςοι ίδιοι κανόνες όσον αφορά στην χρήση του χαρακτήρα \. όπως αναφέραμε ατην προηγούμενη ενότητα για την εντολή INSERT.

Χρήση Τελεστών σε Προτάσεις WHERE

Σε προηγούμενα παραδείγματα χρησιμοποιήσαμε τον τελεστή ίσον (=) σε προτάσεις WHERE για να καθορίσουμε την αλήθεια μιας συνθήκης - ένα πράγμα είναι ίσο με ένα άλλο. Μπορούμε να χρησιμοποιήσουμε πολλούς τύπους τελεστών· οι δημοφιλέστεροι εξ αυτών είναι οι τελεστές σύγκρισης και οι λογικοί τελεστές.

Οι τελεστές σύγκρισης, οι οποίοι παρουσιάζονται στον Πίνακα 2.1, θα πρέπει να μας είναι γνωστοί από την άλγεβρα του γυμνασίου.

ΠΙΝΑΚΑΣ 2.1 Οι Βασικοί Τελεστές Σύγκρισης και η Σημασία τους

=	Ίσο με
!=	Άνισο (διάφορο)
<=	Μικρότερο από ή ίσο με
<	Μικρότερο από
>=	Μεγαλύτερο από ή ίσο με
>	Μεγαλύτερο από

Υπάρχει επίσης ένας βολικός τελεστής με όνομα BETWEEN, ο οποίος είναι χρήσιμος για συγκρίσεις ακέραιων και άλλων δεδομένων, επειδή αναζητά τα αποτελέσματά του μεταξύ (between) μιας ελάχιστης και μιας μέγιστης τιμής.

Υπάρχουν επίσης σι λογικοί τελεστές, οι οποίοι μας επιτρέπουν να χρησιμοποιήσουμε πολλαπλές συγκρίσεις στην πρόταση WHERE. Οι βασικοί λογικοί τελεστές είναι σι AND (σύζευξη) και OR (διάζευξη). Οταν χρησιμοποιούμε τον τελεστή AND, όλες οι συγκρίσεις που περιλαμβάνουμε στην πρόταση WHERE πρέπει να αποτιμηθούν σαν αληθείς. Από την άλλη, ο τελεστής OR απαιτεί να είναι αληθής τουλάχιστον μία από τις συγκρίσεις της πρότασης WHERE.

Συγκρίσεις Αλφαριθμητικών με τον Τελεστή LIKE

Μπορούμε να συγκρίνουμε αλφαριθμητικά μέσα σε μία πρόταση WHERE χρησιμοποιώντας τον τελεστή = ή !=, αλλά υπάρχει ακόμη ένας τελεστής, ιδιαίτερα χρήσιμος σε προτάσεις WHERE για την σύγκριση αλφαριθμητικών: ο LIKE. Ο τελεστής αυτός χρησιμοποιεί δύο χαρακτήρες μπαλαντέρ για τις αναζητήσεις του:

- % - Ταιριάζει με πολλαπλούς χαρακτήρες
- _ - Ταιριάζει με έναν ακριβώς χαρακτήρα

Εάν θέλουμε να βρούμε τις εγγραφές του πίνακα ekpaideutikos στις οποίες το ονοματεπώνυμο του εκπαιδευτικού ξεκινά με το γράμμα "Γ"", μπορούμε να χρησιμοποιήσουμε την ακόλουθη εντολή:

mysql> select * from ekpaideutikos where onomatepwnymo like 'Γ%';

id_ekpaid	onomatepwnymo	e_mail	id_kathg-ekpaid
7	**Γαρμπής Αριστογιάννης**		**7**
15	**Γάτσιος Γεώργιος**		**4**
16	**Γεωργίου Γεώργιος**		**1**
17	**Γιαννόπουλος Κων/νος**		**4**
18	**Γκέκα Ελένη**		**4**

ΣΗΜΕΙΩΣΗ: Εκτός κι αν εφαρμόζουμε τον τελεστή LIKE για την σύγκριση δυαδικών αλφαριθμητικών, η σύγκριση δεν κάνει διάκριση μεταξύ κεφαλαίων και πεζών χαρακτήρων.

2.2.7 Επιλογή Δεδομένων από Πολλαπλούς Πίνακες

Δεν περιορίζόμαστε στην επιλογή δεδομένων μόνο από έναν πίνακα τη φορά. Εάν ίσχυε ένας τέτοιος περιορισμός, ο προγραμματισμός εφαρμογών θα ήταν μία ιδιαίτερα επίπονη και κουραστική εργασία! Οταν επιλέγουμε δεδομένα από περισσότερους του ενός πίνακες με μία εντολή SELECT, οι πίνακες ενώνονται (join) για την παραγωγή του αποτελέσματος.

Υποθέστε ότι έχουμε δύο πίνακες, τους fruit (φρούτα) και color (χρώμα). Μπορούμε να επιλέξουμε όλες τις γραμμές από τους δύο πίνακες χρησιμοποιώντας δύο ξεχωριστές εντολές SELECT:

mysql> select * from fruit;

id	fruitname
1	apple
2	orange
3	grape
4	banana

4 rows in set (0.00 sec)

mysql> select * from color;

id	colorname
1	red
2	orange
3	purple
4	yellow

4 rows in set (0.00 sec)

Οταν θέλουμε να επιλέξουμε δεδομένα ταυτόχρονα από δύο πίνακες, υπάρχουν ορισμένες διαφορές στην σύνταξη της εντολής SELECT. Κατ' αρχήν θα πρέπει να διασφαλίσουμε ότι όλοι οι πίνακες από τους οποίους θέλουμε να ανακτήσουμε δεδομένα περιλαμβάνονται στην πρόταση FROM της εντολής SELECT. Χρησιμοποιώντας σαν παράδειγμα τους πίνακες fruit και color εάν θέλαμε απλώς να επιλέξουμε όλα τα πεδία και όλες τις εγγραφές από τους δύο πίνακες ίσως νομίσουμε ότι θα έπρεπε να χρησιμοποιήσουμε την ακόλουθη εντολή SELECT:

mysql> select * from fruit , color;

id	fruitname	id	colorname
1	apple	1	red
2	orange	1	red
3	grape	1	red
4	banana	1	red
1	apple	2	orange
2	orange	2	orange
3	purple	2	orange
4	banana	2	orange
1	apple	3	purple
2	orange	3	purple
3	purple	3	purple
4	banana	3	purple
1	apple	4	yellow
2	orange	4	yellow
3	purple	4	yellow
4	banana	4	yellow

16 rows in set (0.00 sec)

Δεκαέξι γραμμές με επαναλαμβανόμενες πληροφορίες - όχι ακριβώς αυτό που ζητούσαμε! Τί έκανε το παραπάνω ερώτημα; Ουσιαστικά ένωσε μία γραμμή του πίνακα color με κάθε γραμμή του πίνακα fruit. Επειδή υπάρχουν τέσσερις εγγραφές στον πίνακα fruit και τέσσερις εγγραφές στον πίνακα color, το ερώτημα επέστρεψε 16 εγγραφές.

Οταν επιλέγουμε δεδομένα από πολλαπλούς πίνακες, θα πρέπει να δίνουμε ιδιαίτερη προσοχή στην σύνταξη της πρότασης WHERE για να διασφαλίσουμε ότι θα πάρουμε το σωστό αποτέλεσμα. Στην περίπτωση των πινάκων fruit και color, αυτό που θέλουμε πραγματικά είναι να εμφανίσουμε τα πεδία με τα ονόματα φρούτων και τα ονόματα χρωμάτων από αυτούς τους δύο πίνακες για τις εγγραφές με τους ίδιους κωδικούς στο πεδίο id. Αυτό μας φέρνει στο επόμενο σημαντικό ζήτημα για το ερώτημα πώς – πώς ακριβώς θα υποδείξουμε ποια πεδία θέλουμε όταν τα πεδία έχουν τα ίδια ακριβώς ονόματα στους δύο πίνακες!

Απλό· προσθέτουμε το όνομα του πίνακα στο όνομα του πεδίου, ως εξής:

όνομα_πίνακα.όνομα_πεδίου

Συνεπώς, το ερώτημα για την επιλογή των ονομάτων και των χρωμάτων των φρούτων από τους δύο πίνακες για τις εγγραφές με τον ίδιο κωδικό στο πεδίο id πρέπει να διατυπωθεί ως εξής:

mysql> select fruitname, colorname from fruit, color where fruit.id = color.id;

fruitname	colorname
apple	red
orange	orange
grape	purple
banana	yellow

4 rows in set (0.00 sec)

Ωστόσο, εάν επιχειρήσουμε να επιλέξουμε μία στήλη η οποία εμφανίζεται και στους δύο πίνακες με το ίδιο όνομα, θα πάρουμε ένα σφάλμα το οποίο υποδεικνύει την ασάφεια του ερωτήματός μας:

```
mysql> select id, fruitname, colorname from fruit, color
          -> where fruit.id = color.id;
ERROR 1052: Column: 'id' in field list is ambiguous
```

Εάν θέλουμε να επιλέξουμε το πεδίο id από τον πίνακα fruit, θα έπρεπε να χρησιμοποιήσουμε το ακόλουθο ερώτημα:

mysql> select fruit.id, fruitname, colorname from fruit, color
-> where fruit.id = color.id;

id	fruitname	colorname
1	apple	red
2	orange	orange
3	grape	purple
4	banana	yellow

4 rows in set (0.00 sec)

Το παραπάνω ερώτημα ήταν ένα απλό παράδειγμα ένωσης δύο πινάκων στην ίδια εντολή SELECT. Η JOIN είναι μία δεσμευμένη λέξη της γλώσσας SQL και μας δίνει την δυνατότητα να δημιουργούμε πολυπλοκότερα ερωτήματα.

2.2.8 Χρήση της JOIN

Μπορούμε να χρησιμοποιήσουμε αρκετούς τύπους ενώσεων (joins) στην MySQL, όλοι εκ των οποίων αναφέρονται στην σειρά με την οποία ενώνονται οι πίνακες και στην σειρά με την οποία εμφανίζονται τα αποτελέσματα. Ο τύπος της ένωσης που χρησιμοποιείται για τους πίνακες fruit και color ονομάζεται εσωτερική ένωση (INNER JOIN), αν και δεν αναφέρθηκε ρητά στο ερώτημα. Αναδιατυπώνοντας την εντολή SQL ώστε να χρησιμοποιεί την σωστή σύνταξη INNER JOIN, θα καταλήξουμε στην ακόλουθη εντολή:

mysql> select fruitname, colorname from fruit inner join color

-> on fruit.id = color.id;

fruitname	Colorname
apple	red
orange	orange
grape	purple
banana	yellow

4 rows in set (0.00 sec)

Σ' αυτή την περίπτωση η πρόταση ON αντικατέστησε την πρόταση WHERE, λέγοντας στην MySQL να ενώσει τις γραμμές των πινάκων οι οποίες έχουν ίδια τιμή στο πεδίο id. Όταν ενώνουμε πίνακες χρησιμοποιώντας την πρόταση ON, μπορούμε να χρησιμοποιούμε οποιεσδήποτε συνθήκες θα χρησιμοποιούσαμε και σε μία πρόταση WHERE, συμπεριλαμβανομένων των λογικών και αριθμητικών τελεστών.

Ενας άλλος κοινός τύπος ένωσης είναι η αριστερή ένωση (LEFT JOIN). Όταν ενώνουμε δύο πίνακες με την LEFT JOIN, επιστρέφονται όλες οι εγγραφές από τον πρώτο πίνακα, ανεξάρτητα από το εάν υπάρχουν αντίστοιχες εγγραφές στον δεύτερο πίνακα. Υποθέστε ότι έχουμε δύο πίνακες σε μία βάση δεδομένων με διευθύνσεις - τον master_name, ο οποίος περιέχει τις βασικές εγγραφές και το email, ο οποίος περιέχει εγγραφές με διευθύνσεις ηλεκτρονικού ταχυδρομείου. Οποιεσδήποτε εγγραφές του πίνακα email μπορούν να συσχετίζονται μέσω κωδικού (id) με μία εγγραφή του πίνακα master_name. Για παράδειγμα:

mysql> select name_id, firstname, lastname from master_name;

name_id	firstname	lastname
1	John	Smith
2	Jane	Smith
3	Jimbo	Jones
4	Andy	Smith

7	Chris	Jones
45	Anna	Bell
44	Jimmy	Carr
43	Albert	Smith
42	John	Doe

9 rows in set (0.00 sec)

mysql> select name_id, email from enail;

NAME_ID	EMAIL
42	[illegible]
45	annabell@aol.com

2 rows in set (0.00 sec)

Χρησιμοποιώντας την LEFT JOIN σ' αυτούς τους δύο πίνακες, θα διαπιστώσουμε ότι εάν δεν υπάρχει τιμή στον πίνακα email για μία εγγραφή του κύριου πίνακα (master_name), εμφανίζεται η τιμή NULL στην θέση της διεύθυνσης email στο αποτέλεσμα του ερωτήματος:

mysql> select firstname, lastname, email from master_name left join email

->on master_name.name_id = email.name_id;

FIRSTNAME	LASTNAME	EMAIL
John	Smith	NULL
Jane	Smith	NULL
Jimbo	Jones	NULL
Andy	Smith	NULL
Chris	Jones	NULL
Anna	Bell	[illegible]

Jimmy	Carr	NULL
Albert	Smith	NULL
John	Doe	

9 rows in set (0.01 sec)

Μία δεξιά ένωση (RIGHT JOIN) δουλεύει παρόμοια με την αριστερή ένωση (LEFT JOIN), αλλά αντιστρέφοντας την σειρά των πινάκων. Με άλλα λόγια, όταν χρησιμοποιούμε την RIGHT JOIN επιστρέφονται όλες οι γραμμές από τον δεύτερο πίνακα, ανεξάρτητα από το εάν υπάρχουν αντίστοιχες γραμμές στον πρώτο πίνακα. Ωστόσο, στην περίπτωση των πινάκων master_name και email, υπάρχουν μόνο δύο γραμμές στον πίνακα email, ενώ υπάρχουν εννέα γραμμές στον πίνακα master_name. Αυτό σημαίνει ότι θα επιστραφούν μόνο δύο από τις εννέα γραμμές του πίνακα master_name:

mysql> select firstname, lastname, email from master_name right join email

->on master_name.name_id = email.name_id;

FIRSTNAME	LASTNAME	EMAIL
John	Doe	
Anna	Bell	

2 rows in set (0.00 sec)

Η MySQL υποστηρίζει αρκετούς τύπους ενώσεων και σ' αυτή την ενότητα μάθαμε για τους πιο κοινά χρησιμοποιούμενους. Για να μάθουμε περισσότερα για τους διάφορους τύπους ενώσεων, όπως οι CROSS JOIN, STRAIGHT JOIN και NATURAL JOIN, ανατρέξτε στο εγχειρίδιο τη MySQL (MySQL Manual), στην διεύθυνση http://www.mysql.com/doc/J/O/JOIN.html

2.2.9 Χρήση της Εντολής UPDATE για την Τροποποίηση Εγγραφών

Η UPDATE είναι μία εντολή της SQL η οποία χρησιμοποιείται για την τροποποίηση του περιεχομένου μιας ή περισσότερων στηλών σε μία υπάρχουσα εγγραφή ενός πίνακα. Στην απλούστερη μορφή της, η σύνταξη της εντολής UPDATE είναι:

```
UPDATE όνομα πίνακα
SET στήλη1 = 'νέα τιμή1',
στήλη2='νέα τιμή2'
[WHERE συνθήκη]
```

Η γενική διαδικασία για την ενημέρωση μιας εγγραφής είναι παρόμοια με αυτή που χρησιμοποιείται για την εισαγωγή μιας εγγραφής - τα δεδομένα που εισάγαμε πρέπει να είναι κατάλληλα για τον τύπο δεδομένων του πεδίου που τροποποιούμε και θα πρέπει επίσης να περικλείουμε τα αλφαριθμητικά μας σε αποστρόφους ή εισαγωγικά, χρησιμοποιώντας χαρακτήρες \ εάν είναι απαραίτητο.

Για παράδειγμα, υποθέστε ότι έχουμε τον πίνακα με όνομα ekpaideutikos ο οποίος περιλαμβάνει τα πεδία , id_ekpaid, onomatepwnymo, e_mail, id_kathg_ekpaid και τα οποία περιέχουν, αντίστοιχα, δεδομένα για τον κωδικό, το ονοματεπώνυμο, το e_mail, και την βαθμίδα του εκπαιδευτικού:

Ενημέρωση Υπό Συνθήκες

Η διατύπωση της εντολής UPDATE με τέτοιο τρόπο ώστε να εκτελεί τις τροποποιήσεις υπό συνθήκες σημαίνει ότι θα πρέπει να χρησιμοποιήσουμε μία πρόταση WHERE για την επιλογή των κατάλληλων εγγραφών. Η χρήση μιας πρότασης WHERE σε μία εντολή UPDATE δεν διαφέρει από την χρήση μιας πράτασης WHERE σε μία εντολή SELECT- μπορούμε να χρησφοποιήσουμε όλους τους γνωστούς τελεστές (λογικούς, σύγκρισης, κ.λ.π.).

Υποθέστε ότι ο πίνακας ekpaideutikos, περιέχει μία εγγραφή η οποία έχει ένα ορθογραφικό λάθος ("Τριανταφύλου" αντί για "Τριανταφύλλου"). Η σωστή εντολή UPDATE για την διόρθωση αυτής της ανορθογραφίας είναι:

```
mysql>update ekpaideutikos set ekpaideutikos_onomatepwnymo = 'Τριανταφύλλου'

    ->where ekpaideutikos_onomatepwnymo = 'Τριανταφύλου';

Query OK, 1 row affected (0.00 sec)
Rows matched: 1 Changed: 1 Warnings: 0
```

Σ'αυτή την περίπτωση επιλέχθηκε και τροποποιήθηκε μόνο μία γραμμή. Ολος ο υπόλοιπος πίνακας ekpaideutikos παραμένει άθικτος και όλα τα ονόματα των εκπαιδευτικών είναι σωστά:

1	Δρόσος Λάμπρος	[illegible]	5
2	Μαυριδάκης Θεοφάνης	[illegible]	6
3	Τριανταφύλλου Βασίλης	[illegible]	5
4	Μακρής Χρήστος	[illegible]	6
5	Συρμακέσης Σπύρος	[illegible]	6

5 rows in set (0.00 sec)

2.2.10 Χρήση της Εντολής REPLACE

Μία άλλη μέθοδος για την τροποποίηση των εγγραφών ενός πίνακα είναι η εντολή REPLACE, η οποία έχει σημαντικές ομοιότητες με την εντολή INSERT.

REPLACE INTO όνομα_πίνακα (λίστα στηλών) VALUES (τιμές στηλών);

Η εντολή REPLACE λειτουργεί ως εξής: εάν η εγγραφή που εισάγουμε στον πίνακα περιέχει μία τιμή πρωτεύοντος κλειδιού η οποία ταιριάζει με την τιμή πρωτεύοντος κλειδιού μιας υπάρχουσας εγγραφής, η υπάρχουσα εγγραφή του πίνακα διαγράφεται και στην θέση της εισάγεται η νέα εγγραφή.

ΣΗΜΕΙΩΣΗ: Η εντολή REPLACE είναι μία επέκταση της MySQL· δεν περιλαμβάνεται ατην ANSI SQL. Η εντολή αυτή προσομοιώνει το αποτέλεσμα της συνδυασμένης χρήσης μιας εντολής DELETE και μιας εντολής INSERT για την διαγραφή και επανεισαγωγή μιας συγκεκριμένης εγγραφής. Δηλαδή, δύο εντολές στην τιμή της μιας.

Χρησιμοποιώντας σαν παράδειγμα τον πίνακα ai8ousa, η ακόλουθη εντολή θα αντικαταστήσει την εγγραφή για την αίθουσα:

mysql> replace into ai8ousa values

->(1, 'Λ5');
Query OK, 2 rows affected (0.00 sec)

Στο αποτέλεσμα της εντολής παρατηρούμε ότι αναφέρεται το , 2 rows affected (επηρεάστηκαν 2 γραμμές). Σ'αυτή την περίπτωση, επειδή το πεδίο id_ai8ousas είναι πρωτεύον κλειδί στον πίνακα και η τιμή 1 που χρησιμοποιήσαμε στην REPLACE υπάρχει ήδη στον πίνακα ai8ousa, διαγράφτηκε η αρχική εγγραφή μ' αυτή την τιμή στο πεδίο id_ai8ousas και αντικαταστάθηκε από μία νέα εγγραφή - συνεπώς, επηρεάστηκαν 2 γραμμές.

Χρησιμοποιούμε ένα ερώτημα για να επιλέξουμε τις εγγραφές και να βεβαιωθούμε ότι η αντικατάσταση έγινε σωστά.Η παλιά τιμή της αίθουσας πριν την αντικατάσταση , με κωδικό 1 ήταν 'Λ6':

id_ai8ousas	**ai8ousa**
1	Λ5
2	Ε1
3	Ε2
4	ΑΜΦ
5	ΕΡ1
6	ΕΡ2
7	ΕΡ3
8	ΕΡ4
9	ΕΡ5
10	ΕΡ6

Εάν χρησιμοποιήσουμε την εντολή REPLACE και η τιμή του πρωτεύοντος κλειδιού στην εισαγόμενη εγγραφή δεν ταιριάζει με μία υπάρχουσα τιμή πρωτεύοντος κλειδιού στον πίνακα, η νέα εγγραφή θα εισαχθεί χωρίς να επηρεαστεί καμία από τις υπάρχουσες γραμμές του πίνακα.

2.2.11 Χρήση της Εντολής DELETE

Η βασική σύνταξη για την εντολή DELETE είναι:

```
DELETE FROM όνομα_πίνακα
[WHERE συνθήκη]
[LIMIT γραμμές]
```

Παρατηρήστε ότι δεν προβλέπεται προδιαγραφή για τις στήλες στην εντολή DELETE- η εντολή αυτή διαγράφει ολόκληρες εγγραφές. Γι'αυτό θα πρέπει να είμαστε προσεκτικοί όταν χρησιμοποιείτε την εντολή DELETE.

Υποθέτουμε ότι έχουμε τον πίνακα ekpaideutikos και την ακόλουθη δομή του.Βέβαια παρουσιάζουμε ένα δείγμα του πίνακα για να εξηγήσουμε την χρήση των εντολών:

id_ekpaid	onomatepwnymo	e_mail	id_kathg-ekpaid
1	**Δρόσος Λάμπρος**		**5**
2	**Μαυριδάκης Θεοφάνης**		**6**
3	**Τριανταφύλλου Βασίλης**		**5**
4	**Μακρής Χρήστος**		**6**
5	**Συρμακέσης Σπύρος**		**6**

5 rows in set (0.00 sec)

Η ακόλουθη εντολή θα διαγράψει όλες τις εγγραφές του πίνακα:

mysql> delete from ekpaideutikos;
Query OK, 0 rows affected (0.00 sec)

Μπορούμε να επιβεβαιώσουμε το γεγονός της διαγραφής εκτελώντας μία εντολή SELECT για να επιλέξουμε δεδομένα από τον πίνακα:

mysql> select * from ekpaideutikos;
Empty set (0.00 sec)

Οπως βλέπετε, οι εγγραφές όλων των εκπαιδευτικών έχουν διαγραφεί.

Διαγραφή Υπό Συνθήκες

Μία εντολή DELETE η οποία εκτελείται υπό συνθήκες, όμοια με μία εντολή SELECT ή UPDATE η οποία εκτελείται υπό συνθήκες, χρησιμοποιεί μία πρόταση WHERE για την επιλογή συγκεκριμένων εγγραφών. Έχουμε στην διάθεσή μας ολόκληρη την γκάμα των τελεστών σύγκρισης και των λογικών τελεστών για να διατυπώσουμε την συνθήκη βάσει της οποίας θα επιλεχθούν οι εγγραφές που θέλουμε να διαγράψουμε.

Ενα παράδειγμα θα ήταν η διαγραφή όλων των εγγραφών που έχουν κωδικό κατηγορίας εκπαιδευτικού 6 από τον πίνακα ekpaideutikos:

mysql> delete from ekpaideutikos where id_kathg_ekpaid = 6 ;
Query OK , 3 rows affected (0.00 sec)

Οπως υποδεικνύει το μήνυμα, διαγράφτηκαν 3 εγγραφές.

id_ekpaid	onomatepwnymo	e_mail	id_kathg-ekpaid
1	**Δρόσος Λάμπρος**		**5**
4	**Τριανταφύλλου Βασίλης**		**5**

2 rows in set (0.00 sec)

Για τους χρήστες της MySQL 4.0 (ή μεταγενέστερης έκδοσης): μπορούμε επίσης να χρησιμοποιούμε την πρόταση ORDER BY σε μία εντολή DELETE. Ακολουθεί η σύνταξη της εντολής DELETE με την προσθήκη της πρότασης ORDER BY:

```
DELETE FROM όνομα πίνακα
[WHERE συνθήκη]
[ORDER BY όνομα_στήλης [ASC | DESC]]
[LIMIT γραμμές]
```

Με την πρώτη ματιά ίσως αναρωτηθείτε "Τι σημασία έχει η ταξινόμηση για τις εγγραφές που πρόκειται να διαγράψω;". Η πρόταση ORDER BY δεν καθορίζει την σειρά διαγραφής· χρησιμοποιείται για την ταξινόμηση των εγγραφών.

Ανασκόπηση

Σ'αυτό το κεφάλαιο μάθαμε τα βασικά στοιχεία της SQL - από την δημιουργία ενός πίνακα, μέχρι την τροποποίηση εγγραφών. Η εντολή για την δημιουργία πινάκων απαιτεί τρία σημαντικά στοιχεία πληροφορίας: το όνομα του πίνακα, τα ονόματα των πεδίων και τους ορισμούς των πεδίων. Οι ορισμοί των πεδίων είναι σημαντικοί, επειδή ένας καλοσχεδιασμένος πίνακας βοηθά ατην ταχύτερη λειτουργία και απόκριση της βάσης δεδομένων. Η MySQL υποστηρίζει τρεις διαφορετικές κατηγορίες τύπων δεδομένων: αριθμητικοί, ημερομηνίας/ώρας και αλφαριθμητικών.

Η εντολή INSERT χρησιμοποιείται για την προσθήκη εγγραφών σε έναν πίνακα. Η εντολή SELECT χρησιμοποιείται για την επιλογή (ανάκτηση) εγγραφών από συγκεκριμένους πίνακες. Ο χαρακτήρας * μας επιτρέπει να επιλέγουμε εύκολα όλα τα πεδία για όλες τις εγγραφές ενός πίνακα, αλλά μπορούμε επίσης να καθορίζουμε συγκεκριμένα ονόματα στηλών. Εάν το αποτέλεσμα είναι υπερβολικά μακροσκελές, η πρόταση LIMIT μας παρέχει μία απλή μέθοδο για να ανακτάμε μικρότερα τμήματα, υποδεικνύοντας μία θέση έναρξης και τον αριθμό των εγγροφών που θέλουμε να επιστρέψει το ερώτημα. Η πρόταση ORDER BY χρησιμοποιείται για την ταξινόμηση των δεδομένων. Η ταξινόμηση μπορεί να εκτελείται βάσει αριθμών, ημερομηνιών και αλφαριθμητικών, με αύξουσα ή φθίνουσα σειρά. Η προεπιλεγμένη σειρά ταξινόμησης είναι αύξουσα. Εάν δεν καθορίσουμε ταξινόμηση, τα αποτελέσματα του ερωτήματος εμφανίζονται με την σειρά με την οποία υπάρχουν οι εγγραφές στον πίνακα.

Μπορούμε να επιλέξουμε ποιες εγγραφές θέλουμε να επιστρέφει ένα ερώτημα χρησιμοποιώντας την πρόταση WHERE με μία συνθήκη ελέγχου. Στην συνθήκη της πρότασης WHERE μπορούμε να χρησιμοποιήσουμε λογικούς τελεστές ή τελεστές σύγκρισης, μεμονωμένα ή σε συνδυασμό, για την διατύπωση πολυπλοκότερων συνθηκών.Εάν θέλουμε να επιλέξουμε εγγραφές από πολλαπλούς πίνακες με την ίδια εντολή SELECT, θα πρέπει να ενώνουμε τους πίνακες χρησιμοποιώντας τον σωστό τύπο ένωσης (JOIN). Οι πιο κοινοί τύποι ένωσης είναι οι INNER JOIN (εσωτερική ένωση), LEFT JOIN (αριστερή ένωση) και RIGHT JOIN (δεξιά ένωση), αν και η MySQL υποστηρίζει και άλλα είδη ενώσεων.

Οι εντολές UPDATE και REPLACE χρησιμοποιούνται για την τροποποίηση υπαρχόντων δεδομένων ενός πίνακα. Η UPDATE είναι κατάλληλη για την αλλαγή τιμών σε συγκεκριμένες στήλες, ή για την αλλαγή τιμών σε πολλαπλές εγγραφές με βάση συγκεκριμένες συνθήκες.Η REPLACE είναι μία παραλλαγή της εντολής INSERT, η οποία διαγράφει και κατόπιν εισάγει εκ νέου μία εγγραφή με το ίδιο πρωτεύον κλειδί. Προσέξτε ιδιαίτερα όταν χρησιμοποιούμε την εντολή UPDATE για να αλλάξουμε τιμές σε μία στήλη, επειδή εάν δεν χρησιμοποιήσουμε την κατάλληλη συνθήκη για την επιλογή των σωστών εγγραφών, το αποτέλεσμα θα είναι ενημέρωση της στήλης για όλες τις εγγραφές του πίνακα.

Η εντολή DELETE είναι απλή - διαγράφει ολόκληρες εγγραφές από έναν πίνακα. μία ενδεχομένως επικίνδυνη εντολή, καλό θα είναι να παραχωρούμε δικαιώματα για την χρήση της DELETE μόνο στους χρήστες οι οποίοι μπορούν να έχουν αυτή την υπευθυνότητα.Όταν χρησιμοποιούμε την εντολή DELETE, μπορούμε να καθορίζουμε μία συνθήκη στην πρόταση WHERE για υην επιλογή συγκεκριμένων εγγραφών ενός πίνακα χρησιμοποιώντας την πρόταση LIMIT.

ΚΕΦΑΛΑΙΟ 3

3.1 Εγκατάσταση και Διαμόρφωση των MySQL, Apache και PHP

3.1.1 Εγκατάσταση της MySQL στα Windows

Η διαδικασία εγκατάστασης της Mysql στα Windows είναι αρκετά απλή και καθοδηγούμενη από έναν οδηγό (wizard) .Έχουμε μεταφέρει όλους τους πόρους ,αρχίζουμε κάνοντας unzip τα αρχεία στον κατάλογο temp και τρέχουμε το πρόγραμμα setup.exe.Ο προκαθορισμένος κατάλογος όπου θα εγκατασταθεί η ίδια η MySQL είναι ο κατάλογος C:\mysql .Η εταιρεία MySQLAB διαθέτει ένα εργαλείο διαχείρισης με γραφικό περιβάλλον για τους χρήστες των Windows, το οποίο ονομάζεται WinMySQLadmin
.Αφού εγκατασταθεί πρέπει να ξεκινήσει χρησιμοποιώντας το βοηθητικό πρόγραμμα Services Control Manager(SCM)(που βρίσκεται στο Control Panel) ή χρησιμοποιώντας την εντολή NET START MySQL.

Για να ελέγξουμε αν η MySQL δουλεύει , μπορούμε να εκτελέσουμε τις παρακάτω εντολές :

```
C:\ mysql\ bin \ mysqlshow
C:\ mysql\ bin \ mysqlshow -u root mysql
C:\ mysql\ bin \ mysqladmin version status proc
C:\ mysql\ bin \ mysqladmin -u root shutdown
```

Η MySQL θα δημιουργήσει δύο βάσεις δεδομένων , τις βάσεις δεδομένων mysql και test.Η βάση δεδομένων mysql θα χρησιμοποιηθεί για αποθήκευση των δικαωμάτων και πρόσβαση στον διακομιστή.Η βάση δεδομένων test δεν απαιτείται,αλλά μας δίνει ένα ασφαλές μέρος να εκτελούμε εντολές, για να δούμε αν όλα διαμορφώθηκαν σωστά

Είμαστε τώρα έτοιμοι να εγκαταστήσουμε τον Apache στα Windows!

3.1.2 Εγκατάσταση του Apache στα Windows

Το Apache 2.0 τρέχει στις περισσότερες πλατφόρμες των Windows και παρέχει αυξημένει απόδοση και σταθερότητα έναντι της έκδοσης 1.3 για Windows.Πριν εγκαταστήσουμε το Apache θα πρέπει να διασφαλίσουμε οτι δεν τρέχουμε κανέναν Web server στο σύστημά μας (π.χ. μια προηγούμενη έκδοση του Apache, τον

Microsoft Internet Information Server ή τον Microsoft Personal Web Server).Καλό θα ήταν να καταργήσουμε την εγκατάσταση των υπαρχόντων servers ή να τους απενεργοποιήσουμε με οποιονδήποτε άλλο τρόπο.Μπορούμε να τρέχουμε πολλαπλούς Web servers αλλά σε διαφορετικούς συνδιασμούς διευθύνσεων και θυρών.
Η διαδικασία εγκατάστασης είναι πολύ γνωστή,δείχνει παρόμοια με άλλες εφαρμογές εγκατάστασης των Windows.
Ο κατάλογος που θα εγκατασταθεί ο Apache είναι C:\ Program Files \ Apache Group \Apache.
Μετά την εγκατάσταση του Apache, θα πρέπει να τροποποιήσουμε τα αρχεία διαμόρφωσης που υπάρχουν στον κατάλογο conf .Θα δούμε την επεξεργασία του αρχείου διαμόρφωσης httpd.conf, αφού εγκαταστήσουμε την PHP.

Υπάρχουν δύο τρόποι που μπορούμε να τρέξουμε τον Apache:

- Από ένα παράθυρο κονσόλας
- Ως Windows υπηρεσία.

Τρέχοντας τον Apache σε ένα Παράθυρο Κονσόλας

Για να τρέξουμε τον Apache με ένα παράθυρο κονσόλας , επιλέγουμε την επιλογή Start Apache as console App, από το μενού Start. Αυτό θα ανοίξει ένα παράθυρο κονσόλας και θα ξεκινήσει τον Apache. Αυτό θα παραμείνει ενεργό και μέχρι να σταματήσουμε τον διακομιστή Apache.

Για να τον σταματήσουμε,θα πρέπει είτε να τρέξουμε την επιλογή Shutdown Apache as Consol App από το μενού Start ή να ανοίξουμε ένα άλλο παράθυρο εντολών και να πληκτρολογήσουμε το παρακάτω:
C:\Program Files\ Apache Group\ Apache> apache –k shutdown

Εκτελώντας τον Apache ως Υπηρεσία:

Πριν ξεκινήσουμε τον Apache ως υπηρεσία, πρέπει να τον εγκαταστήσουμε ως υπηρεσία.Για να εγκαταστήσουμε την προκαθορισμένη υπηρεσία Apache τρέχουμε την επιλογή Install Apache as Service από το μενού Start.Ανοίγουμε το παράθυρο Services στο Control Panel.επιλέγουμε Apache και κάνουμε κλικ στο Start.Ο Apache τρέχει τώρα ,κρυμμένος στο υπόβαθρο.Μπορούμε αργότερα να σταματήσουμε τον Apache,κάνοντας κλικ στο Stop.Ως εναλλακτική λύση στο παράθυρο Services, μπορούμε να αρχίσουμε και να σταματήσουμε την υπηρεσία Apache από την γραμμή εντολών με τα:

```
NET START apache
NET STOP apache
```

Ο Apache θα συνδεθεί με την θύρα 80 αφού ξεκινήσει.Για να συνδεθούμε στον διακομιστή και για να αποκτήσουμε πρόσβαση στην προκαθορισμένη σελίδα,ξεκινάμε έναν browser και δίνουμε αυτό υο URL:

http://localhost/

Αυτό θα πρέπει να απαντήσει με μια σελίδα καλωσορίσματος και μία σύνδεση στο εγχειρίδιο του Apache.Εάν ο κύριος υπολογιστής δεν είναι συνδεδεμένος στο Internet πρέπει να χρησιμοποιήσουμε αυτό με το URL:

http://127.0.0.1/

Αυτή είναι η IP διεύθυνση που σημαίνει localhost.
Αν έχουμε αλλάξει τον αριθμό θύρας από το 80, θα πρέπει να προσαρτήσουμε το :port_number στο τέλος του URL.

3.1.3 Εγκατάσταση και Διαμόρφωση της PHP στα Windows

Τώρα είμαστε έτοιμοι να εγκαταστήσουμε την PHP για Windows.Σταματάμε τον Apache πρίν ξεκινήσουμε την διαδικασία εγκατάστασης της PHP.Η διαδικασία είναι πολύ απλή , αν καταλάβουμε ότι, αντίθετα με την PHP3, η PHP4 έχει διαιρεθεί σε διάφορα συστατικά,που απαιτούν διάφορα DLL.Δηλαδή, δεν μπορούμε να τρέξουμε την PHP σε κατάσταση CGI, ως αυτόνομα εκτελέσιμο πρόγραμμα. Πρέπει να βεβαιωθούμε ότι τα DLL υπάρχουν σε έναν κατάλογο που είναι στην διαδρομή των Windows. Ο ευκολότερος τρόπος για να το κάνουμε είναι να αντιγράψουμε αυτό το DLL στο SYSTEM ή στο SYSTEM32, που είναι κάτω από τον κατάλογο των Windows.Τα DLL που πρέπει να αντιγραφούν είναι τα MSVCRT.DLL(μπορεί να είναι ήδη εκεί) και PHP4TS.DLL.

Προειδοποίηση
Ο Apache 2.0 σχεδιάστηκε για να τρέχει σε Windows NT 4.0, Windows 2000 ή Windows XP. Αυτή τη στιγμή, η υποστήριξη για Windows 9x είναι ατελής. Ο Apache 2.0 δεν αναμένεται να δουλεύει σε αυτά τα συστήματα τώρα.

Παρέχουμε το περίγραμμα εδώ ως ένα οδηγό βοήθειας. Δεν θα πρέπει να έχουμε πρόβλημα να εγκαταστήσουμε και να διαμορφώσουμε την PHP σε ένα υπολογιστή Windows.

1. Αρχίζουμε αντιγράφοντας το php.ini-dist στον κατάλογο '%WINDOWS%' και το μετονομάζουμε σε 'php.ini'.Η μεταβλητη '%WINDOWS%' συνήθως δείχνει στο C:\WINDOWS για τα Windows 9X και το C:\WINNT για διακομιστές NT.

2. Τροποποιούμε το αρχείο php.ini και αλλάζουμε την ρύθμηση extension_dir για να δείχνει στον κατάλογο που περιέχει τις DLL λειτουργικές μονάδες για τις επεκτάσεις(Δηλαδή C:\php\extensions) . Ορίζουμε το doc_root ώστε να δείχνει στο Web διακομιστή ρίζα ,δηλαδή στον ορατό κατάλογο ρίζα του διακομιστή.(Δηλαδή C:\apache\Apache2\htdocs).
3. Αφαιρούμε τα σχόλια από το αρχείο php.ini για τις λειτουργικές μονάδες που θέλουμε να φορτώνονται όταν ξεκινάμε την PHP. Αφαιρούμ ετα σχόλια από τις γραμμές extension=php_*dll , για να φορτώσουμε τις λειτουργικές μονάδες .Σημειώνουμε ότι κάποιες λειτουργικές μονάδες απαιτούν να εγκατασταθούν επιπλέον βιβλιοθήκες στο σύστημα , για να δουλέψει σωστά η λειτουργική μονάδα .Σημειώνουμε επίσης ,ότι η υποστήριξη για MySQL είναι τώρα ενσωματωμένη στην PHP4. Δεν απαιτεί φόρτωση μέσω αυτής της μεθόδου.

Όπως είπαμε υπάρχουν δύο τρόποι να ρυθμίσουμε την PHP να δουλεύει με τον Apache 2.0 στα Windows. Ο ένας είναι να χρησιμοποιήσουμε το CGI binary, ο άλλος είναι να χρησιμοποιήσουμε το Apache module DLL. Σε κάθε περίπτωση πρέπει να σταματήσουμε τον Apache server και να μορφοποιήσουμε το `httpd.conf` στον κατάλογο conf του Apache, για να ρυθμίσουμε τον Apache ώστε να δουλεύει με την PHP.

Πρέπει να εισάγουμε αυτές τις γραμμές στο `httpd.conf` αρχείο ρυθμίσεων του Apache μας για να ρυθμίσουμε το *CGI binary*:

Παράδειγμα 3-1. PHP και Apache 2.0 σαν CGI

```
ScriptAlias /php/ "c:/php/"
AddType application/x-httpd-php .php
AddType application/x-httpd-php .phtml
Action application/x-httpd-php "/php/php.exe"
```

Αν θέλουμε να χρησιμοποιήσουμε την PHP σαν ένα module στον Apache 2.0, μετακινούμε το `php4ts.dll` στο `winnt/system32` (για τα Windows NT/2000) ή στο `windows/system32` (for Windows XP), κάνοντας overwriting τυχόν παλαιότερο αρχείο. Πρέπει να εισάγουμε αυτές τις δυο γραμμές στο αρχείο ρυθμίσεων `httpd.conf` του Apache μας για να ρυθμίσουμε το *PHP-Module* για τον Apache 2.0:

Παράδειγμα 3-2. PHP και Apache 2.0 σαν Module

```
LoadModule php4_module "c:/php/sapi/php4apache2.dll"
AddType application/x-httpd-php .php
```

Σημείωση: Πρέπει να θυμηθούμε να αντικαταστήσουμε το `c:/php/` με το πραγματικό μας path στο `PHP` στα παραπάνω παραδείγματα. Πρέπει να χρησιμοποιήσουμε το `php4apache2.dll` στο LoadModule directive μας και *όχι* το `php4apche.dll`. Το τελευταίο έχει σχεδιαστεί για να τρέχει με τον Apache 1.3.x.

Προειδοποίηση

Μην αναγκατεύετε τις εγκαταστάσεις σας με αρχεία dll από *διαφορετικές εκδόσεις PHP* . Η μόνη σας επιλογή είναι να χρησιμοποιήσετε τα dll και τις επετκάσεις που έρχονται με την έκδοση της PHP που έχετε κάνει download.

3.1.4 Εγκατάσταση της PEAR

Αυτό το κεφάλαιο περιγράφει, πώς να εγκαταστήσουμε πακέτα από την PEAR.

Εισαγωγή

Αυτό το κεφάλαιο απαιτεί να είμαστε ήδη εξοικειωμένοι με την γενική δομή της PEAR.

Η βασική εγκατάσταση η οποία έρχεται με την PHP διανομή ως τμήμα της PEAR, περιέχει όλη την ουσία που απαιτείται για να τρέξουν τα PEAR εργαλεία εγκαταστάσης κ.λ.π. Εάν έχουμε μια πρόσφατη εγκατάσταση PHP, μπορούμε να χαλαρώσουμε: Η PEAR βασική εγκατάσταση είναι ήδη εκεί, εκτός αν έχουμε συντάξει την PHP με `./configure` flag `--` `χωρίς -pear`.

Τα πακέτα που δεν είναι μέρος της PEAR μπορούν να εγκατασταθούν με PEAR διαχειριστή πακέτων. Ο διαχειριστής είναι συγκρίσιμος με το Debian's "apt-get". Πάλι: Εάν τρέχουμε μια πρόσφατη έκδοση PHP (> 4.3.0) μπορούμε να πηδήξουμε το επόμενο τμήμα. Αλλά εάν τρέχουμε PHP 4.2.* ή νωρίτερα, πρέπει να το εγκαταστήσουμε χειρονακτηκά.

Εκτός από την εγκατάσταση των πακέτων, ο PEAR διαχειριστής πακέτων επίσης χειρίζεται μερικούς άλλους στόχους: Μπορεί να δημιουργήσει τα νέα πακέτα στη μηχανή μας, να διαχειρίσει ένα ληξιαρχείο από εγκατεστημένα πακέτα , να ελέγχει τις εξαρτήσεις και μπορεί να αλληλεπιδράσει με XML-RPC υπηρεσία στο pear.php.net για να εξυπηρετήσουν μερικούς άλλους στόχους.

Εγκατάσταση στα Windows

Αφότου έχουμε μεταφορτώσει και έχουμε εγκαταστήσει την PHP, πρέπει manually να εκτελέσουμε το αρχείο δεσμης το οποίο είναι τοποθετημένο μέσα στο π.χ.. `c:\php\go-pear.bat`. Το set up θα μας υποβάλει μερικές ερωτήσεις και κατόπιν PEAR Package Manager θα εγκατασταθεί στην διαδρομή, στην οποία έχουμε δηλώσει κατά τη διάρκεια της εγκατάστασης. Τελος πρέπει να προσθέσουμε εκείνη την διαδρομή εγκαταστάσης στο περιβάλλον PATH μας (Start > Control Panel > System > Environment). Μετά από αυτό μπορούμε να έχουμε πρόσβαση στο PEAR Package Manager τρέχοντας την εντολή `pear` σε ένα Windows Command Prompt.

Για να ενημερώσουμε την PEAR εγκατάσταση από go-pear.org, αίτημα http://go-pear.org/ στον διακομιστή μας και σώζουμε την έξοδο σε ένα τοπικό αρχείο `go-pear.php`. Μπορούμε έπειτα να το τρέξουμε

```
php go-pear.php
```

σε ένα Windows Command Prompt για να αρχίσουμε τη διαδικασία αναπροσαρμογών.

Manual εγκατάσταση

Η εγκατάσταση των πακέτων manually δεν είναι συνιστώμενος τρόπος, αλλά πολλοί άνθρωποι βρίσκουν προβλήματα χρησιμοποιώντας τους αυτοματοποιημένους τρόπους, όταν τα sites τους είναι τοποθετημένα σε περιβάλλοντα web hosting χωρίς πρόσβαση στο φλοιό.

Στις ακόλουθες παραγράφους, θα δείξουμε πώς να εγκαταστήσουμε τα πακέτα manually σε ένα site με το ακόλουθο σχέδιο πορείας : Το έγγραφο root από το site είναι `/var/www/www.example.com/htdocs/`. Στο ίδιο επίπεδο του `htdocs` υπάρχει ένας άλλος κατάλογος αποκαλούμενος `includes`. Αυτός ο κατάλογος δεν μπορεί να προσεγγιστεί via HTTP αλλά via FTP ή WebDAV.

Η εγκατάσταση αποτελείται από μερικά εύκολα να ακολουθηθούν βήματα:

1. Download το πακέτο:

   Μπορούμε να κατεβάσουμε το πακέτο από το PEAR homepage με τον web browser. Εάν δεν ξέρουμε την URL από τη σελίδα πληροφοριών πακέτου για το πακέτο, μπορούμε να χρησιμοποιήσουμε το package browser για να δούμε τα πρόσφατα διαθέσιμα πακέτα.

2. Φορτώνουμε την πηγή κώδικα του πακέτου

Αφού κατεβάσουμε το .tgz αρχείο του πακέτου, πρέπει να εξαγάγουμε το περιεχόμενο του αρχείου σε έναν προσωρινό κατάλογο στην τοπική μηχανή μας. Μετά από αυτό πρέπει να φορτώσουμε τον κώδικα πηγής via FTP, WebDAV ή μια άλλη μέθοδο στον κατάλογο `/var/www/www.example.com/includes/`, έτσι ώστε ο κώδικας πηγής για Mail_Mime τώρα π.χ.. βρίσκεται μέσα στο `/var/www/www.example.com/includes/Mail/`.

3. Ρυθμίζουμε την οδηγία include_path.

   Τώρα πρέπει να ρυθμίσουμε την PHP's include_path οδηγία έτσι ώστε να περιέχει τη θέση όπου φορτώσαμε ακριβώς τον κώδικα πηγής. Εάν έχουμε πρόσβαση στο `php.ini` configuration file από το site μας , πρέπει να προσθέσουμε `/var/www/www.example.com/includes/` στην οδηγία εκεί. Εάν δεν έχουμε πρόσβαση στη διαμόρφωση, πρέπει να θέσουμε το include_path σε κάθε script όπου θέλεουμε να χρησιμοποιήσουμε το πακέτο:

```
ini_set("include_path", '/var/www/www.example.com/includes/' .
PATH_SEPARATOR . ini_get("include_path"));
```

4. Αφού έχουμε τελειώσει την εγκατάσταση, μπορούμε τώρα να χρησιμοποιήσουμε το πακέτο:

```
require_once "Mail/Mime.php";

$mime = ...
```

3.2 Επικοινωνία με Βάσεις Δεδομένων MySQL μέσω PHP

Τώρα που γνωρίζουμε αρκετά πράγματα για την PHP και την MySQL, είμαστε έτοιμοι να χρησιμοποιήσουμε τις δύο αυτές γλώσσες σε συνδυασμό. Θεωρούμε την PHP σαν έναν "αγωγό", μέσω του οποίου μπορούμε να φτάσουμε στην MySQL - οι εντολές που μάθαμε στο προηγούμενο κεφάλαιο είναι οι ίδιες εντολές που θα στέλνουμε στην MySQL στο παρόν κεφάλαιο· αυτή την φορά, όμως, θα τις στέλνουμε με την PHP. Συγκεκριμένα, σ' αυτό το κεφάλαιο θα ασχοληθούμε με τα ακόλουθα:

- Σύνδεση στην MySQL με την χρήση της PHP

- Εισαγωγή και επιλογή δεδομένων μέσα από PHP scripts

3.2.1 Σύνδεση στην MySQL με την PHP

Εάν θέλουμε να χρησιμοποιήσουμε συναρτήσεις της PHP για να συνομιλήσουμε με την MySQL, θα πρέπει να τρέχουμε την MySQL σε μία θέση στην οποία μπορεί να συνδεθεί ο Web server μας (όχι κατ' ανάγκη στον ίδιο υπολογιστή στον οποίο τρέχει ο Web server μας). Θα πρέπει επίσης να έχουμε δημιουργήσει έναν χρήστη (με κωδικό πρόσβασης) και θα πρέπει να γνωρίζουμε το όνομα της βάσης δεδομένων στην οποία θέλουμε να συνδεθούμε. Βέβαια πρώτα θα πρέπει να έχουμε εγκαταστήσει την MySQL, PHP και APACHE σύμφωνα με τις οδηγίες τις οποίες έχουμε δώσει.

Σε όλα τα scripts που θα παρουσιάσουμε σ' αυτό το κεφάλαιο, το όνομα της βάσης δεδομένων είναι programma. Το όνομα του χρήστη είναι root και ο κωδικός πρόσβασής του είναι '0000'.

[redacted]: Μπορούμε να βρούμε την ενότητα του εγχειριδίου της PHP που καλύπτει όλες τις σχετιζόμενες με την MySQL συναρτήσεις στην διεύθυνση http://www.php.net/manual/en/ref.mysql.php.Σας προτείνω να το χρησιμοποιήσετε!

Χρήση της Συνάρτησης mysql_connect()

Η συνάρτηση mysql_connect() είναι η πρώτη συνάρτηση που πρέπει να καλέσουμε όταν χρησιμοποιούμε ένα PHP script για να συνδεθούμε στην MySQL - εάν δεν έχουμε μία ανοικτή σύνδεση με την MySQL, δεν πρόκειται να πάμε και πολύ μακριά! Η βασική σύνταξη για την υλοποίηση της σύνδεσης είναι:

mysql_connect ("όνομα_host", "όνομα_χρήστη", "κωδικός_πρόσβασης") ;

Με τις πραγματικές τιμές που θα χρησιμοποιούμε για τα ορίσματα στα παραδείγματα αυτού του κεφαλαίου, η συνάρτηση για την υλοποίηση της σύνδεσης δείχνει κάπως έτσι:

mysql_connect ("10.10.90.20", "root", "0000") ;

Η συνάρτηση αυτή επιστρέφει έναν δείκτη σύνδεσης (connection index) εάν η σύνδεση είναι επιτυχής, ή εάν η σύνδεση δεν υλοποιηθεί. Η Λίστα 3.1 παρουσιάζει ένα λειτουργικό παράδειγμα script σύνδεσης. Ο κώδικας εκχωρεί την τιμή του δείκτη σύνδεσης σε μία μεταβλητή με όνομα $conn και κατόπιν εκτυπώνει την τιμή της $conn σαν απόδειξη της επιτυχούς υλοποίησης της σύνδεσης.

ΛΙΣΤΑ 3.1 Ενα Απλό Script Σύνδεσης

```
1: <?php
2: $conn = mysql_connect ("10.10.90.20", "root", "0000")
;                                                    3:
echo "$conn";
4: ?>
```

Αποθηκεύουμε αυτό το script με όνομα mysqlconnect.php και το τοποθετούμε στην περιοχή εγγράφων του Web server μας. Εάν προσπελάσουμε αυτό το script μέσα από το περιβάλλον μιας εφαρμογής Web browser, θα δούμε κάτι όπως το ακόλουθο στο παράθυρο της εφαρμογής Web browser:

Resource id #1

Η σύνδεση στην MySQL με την χρήση της συνάρτησης mysql_connect () είναι απλή υπόθεση. Η σύνδεση κλείνει όταν το script ολοκληρώνει την εκτέλεσή του, αλλά εάν θέλουμε να κλείσουμε εμείς ρητά την σύνδεση, προσθέτουμε απλώς την συνάρτηση mysql_close() στο τέλος του script,
όπως βλέπετε στην Λίστα 3.2.

ΛΙΣΤΑ 3.2 Δεύτερη, Τροποποιημένη Εκδοση του Script Σύνδεσης

```
1: <?php
2: $conn = mysql_connect ("10.10.90.20", "root", "0000")
;                                                    3:
echo "$conn";
4: mysql_close ($conn) ;
5:?>
```

Αυτό ήταν όλο. Στην επόμενη ενότητα θα καλύψουμε τις συναρτήσεις εκτέλεσης ερωτημάτων, σι οποίες είναι πολύ πιο ενδιαφέρουσες από το απλό άνοιγμα ή κλείσιμο μιας σύνδεσης!

Εκτέλεση Ερωτημάτων

Εάν ξέρουμε πώς να γράφουμε ερωτήματα χρησιμοποιώντας εντολές SQL, έχουμε κερδίσει την μισή μάχη για την εκτέλεση ερωτημάτων MySQL μέσα από κώδικα PHP. Η συνάρτηση mysql_query() της PHP χρησιμοποιείται για την αποστολή ενός ερωτήματος SQL στην MySQL. Εάν είναι επιτυχής, επιστρέφει έναν δείκτη αποτελέσματος. Εάν αποτύχει, η συνάρτηση επιστρέφει false.

Οταν χρησιμοποιούμε την συνάρτηση mysql_query(), θα παρατηρήσουμε ότι λείπει ένα κομμάτι του παζλ· η επιλογή της βάσης δεδομένων που θα χρησιμοποιηθεί. Οταν συνδεόμαστε στην MySQL από την γραμμή εντολής, η βάση δεδομένων καθορίζεται στο αλφαριθμητικό σύνδεσης· εναλλακτικά, μπορούμε να την αλλάξουμε χειροκίνητα, αφού συνδεθούμε. Με την PHP αυτό γίνεται μέσω μιας διαφορετικής συνάρτησης με όνομα mysql_select_db(), η οποία έχει την ακόλουθη σύνταξη:

mysql_select_db (όνομα_βάσης_δεδομένων, δείκτης_σύνδεσης) ;

Για να συνδεθούμε σε μία βάση δεδομένων με όνομα programma, χρησιμοποιούμε κατ' αρχήν την συνάρτηση mysql_connect () και κατόπιν χρησιμοποιήούμε την mysql_select_db (), όπως βλέπετε στην Λίστα 3.3.

ΛΙΣΤΑ 3.3 Σύνδεση και Επιλογή μιας Βάσης Δεδομένων

```
1: <?php
2: $conn = mysql_connect ("10.10.90.20", "root", "0000")
;                                                    3:
mysql_select_db("programma", $conn") ;
4: ?>
```

Στο σημείο αυτό γνωρίσαμε δύο σημαντικές πληροφορίες: τον δείκτη σύνδεσης (ο οποίος είναι αποθηκευμένος στην μεταβλητή $conn) και το γεγονός ότι η PHP θα χρησιμοποιήσει την βάση δεδομένων programma καθ' όλη την διάρκεια ζωής του συγκεκριμένου script. Ο δείκτης σύνδεσης χρησιμοποιείται στην σύνταξη της συνάρτησης mysql_query():

mysql_query (ερώτημα, δείκτης_σύνδεσης) ;

Στο script μας, υλοποιούμε κατ' αρχήν την σύνδεση και κατόπιν εκτελούμε ένα ερώτημα. Το script της Λίστας 3.4 δημιουργεί έναν απλό πίνακα με όνομα ekpaideutikos.

ΛΙΣΤΑ 3.4 Ενα Script για την Δημιουργία ενός Πίνακα

```
1: <?php
2: / / άνοιγμα της σύνδεσης
3: $conn = mysql_connect ("10.10.90.20", "root", "0000")
;
4: / / επιλογή της βάσης δεδομένων
5: mysql_select_db("programma", $conn") ;
6: / / δημιουργία της εντολής SQL
7: $sql = CREATE TABLE ekpaideutikos (id_ekpaid int not
null primary key          8: auto_increment,
onomatepwnymo varchar(30) not null, e_mail varchar(30),
9: id_kathg_ekpaid int not null ) ;
```

```
10: / / εκτέλεση της εντολής SQL
11: $result = mysql_query ( $sql, $conn ) ;
12: / / εμφάνιση του αποτελέσματος
13: echo $result;
14: ?>
```

ΣΗΜΕΙΩΣΗ: Οταν εισάγουμε ερωτήματα χρησιμοποιώντας την συνάρτηση mysql_query(), το ελληνικό ερωτηματικό στο τέλος της εντολής SQL δεν είναι απαραίτητο. Το μόνο ελληνικό ερωτηματικό σ' αυτή την γραμμή θα πρέπει να εμφανίζεται στο τέλος της εντολής PHP.

Επειδή η συνάρτηση mysql_query επιστρέφει μόνο ένα αποτέλεσμα true (επιτυχία) ή false (αποτυχία), η έξοδος του παραπάνω script είναι:

1

Το1 ισούται με true και υποδεικνύει ότι το ερώτημα εκτελέστηκε επιτυχώς. Η τιμή 0 θα υποδείκνυε την αποτυχία εκτέλεσης του ερωτήματος. Προσπελάστε την MySQL από το περιβάλλον γραμμής εντολής για να επαληθεύσετε την δημιουργία του πίνακα ekpaideutikos:

mysql> describe ekpaideutikos;

2 rows in set (0.00 sec)

Συγχαρητήρια - έχετε δημιουργήσει επιτυχώς έναν πίνακα στην βάση δεδομένων MySQL χρησιμοποιώντας κώδικα της PHP!

Ανάκτηση Μηνυμάτων Σφάλματος

Πρέπει να αφιερώσουμε λίγο χρόνο για να εξοικειωθούμε με την συνάρτηση mysql_error(), επειδή θα γίνει ένας από τους καλύτερους φίλους μας. Οταν χρησιμοποιείται σε συνδυασμό με την συνάρτηση die() της PHP, η οποία τερματίζει απλώς το script στο σημείο στο οποίο εμφανίζεται, ησυνάρτηση mysql_error() επιστρέφει ένα επεξηγηματικό μήνυμα σφάλματος.

Για παράδειγμα, τώρα που έχουμε δημιουργήσει έναν πίνακα με όνομα ekpaideutikos στην βάση δεδομένων, δεν μπορούμε να εκτελέσουμε το ίδιο script ξανά χωρίς να εμφανιστεί ένα μήνυμα σφάλματος. Πριν προσπαθήσουμε να εκτελέσουμε το ίδιο script ξανά, θα το τροποποιήσουμε ώστε να περιλαμβάνει την συνάρτηση mysql_error() (δείτε την Λίστα 3.5).

ΛΙΣΤΑ 3.5 Το Script για την Δημιουργία ενός Πίνακα, με Εμφάνιση Μηνυμάτων Σφάλματος :

```
1: <?php
2: / / άνοιγμα της σύνδεσης
3: $conn = mysql_connect ("10.10.90.20", "root", "0000")
;
4: / / επιλογή της βάσης δεδομένων
5: mysql_select_db("programma", $conn") ;
6: / / δημιουργία της εντολής SQL
7: $sql = CREATE TABLE ekpaideutikos (id_ekpaid int not
null primary key          8: auto_increment,
onomatepwnymo varchar(30) not null, e_mail varchar(30),
9: id_kathg_ekpaid int not null ) ;
10: / / εκτέλεση της εντολής SQL
11: $result = mysql_query ( $sql, $conn ) or
die(mysql_error()) ;
12: / / εμφάνιση του αποτελέσματος
13: echo $result;
14: ?>
```

Αφού εκτελέσουμε το παραπάνω script, θα πρέπει να δούμε κάτι παρόμοιο με το ακόλουθο στο παράθυρο της εφαρμογής Web browser:

Table 'programma' alredy exists

Αυτό κι αν είναι συναρπαστικό! Στην επόμενη ενότητα θα αρχίσουμε να εισάγουμε δεδομένα στον πίνακά μας· πολύ σύντομα θα ανακτάμε και θα μορφοποιούμε τα δεδομένα μας μέσω PHP.

3.2.2 Δουλεύοντας με Δεδομένα της MySQL

Η εισαγωγή, η ενημέρωση, η διαγραφή και η ανάκτηση δεδομένων είναι εργασίες οι οποίες βασίζονται στην χρήση της συνάρτησης mysql_query() για την εκτέλεση ερωτημάτων SQL. Για τις INSERT, UPDATE και DELETE δεν απαιτείται επιπλέον κώδικας στο script μετά απότην εκτέλεση του ερωτήματος, επειδή δεν εμφανίζονται αποτελέσματα (εκτός κι αν το θελήσουμε εμείς ρητά). Όσον αφορά στην SELECT, έχουμε στην διάθεσή μας διάφορες επιλογές για την εμφάνιση των δεδομένων που ανακτά το ερώτημά μας. Αλλά ας ξεκινήσουμε από τα βασικά, εισάγοντας ορισμένα δεδομένα, έτσι ώστε να έχουμε κάτι να ανακτήσουμε αργότερα.

Εισαγωγή Δεδομένων σε έναν Πίνακα μέσω PHP

Η ευκολότερη μέθοδος για να εισάγουμε δεδομένα σε έναν πίνακα μέσω της PHP είναι η ενσωμάτωση των δεδομένων στην εντολή INSERT, όπως βλέπετε στην Λίστα 3.6.

ΛΙΣΤΑ 3.6 Ενα Script για την Εισαγωγή μιας Εγγραφής σε έναν Πίνακα

```
1: <?php
2: / / άνοιγμα της σύνδεσης
3: $conn = mysql_connect ("10.10.90.20", "root", "0000")
;
4: / / επιλογή της βάσης δεδομένων
5: mysql_select_db("programma", $conn") ;
6: / / δημιουργία της εντολής SQL
7: $sql = INSERT INTO ekpaideutikos values (' ', 'some
value' , ' ' , ' ' ) ;
8: / / εκτέλεση της εντολής SQL
9: $result = mysql_query ( $sql, $conn ) or
die(mysql_error()) ;
10: / / εμφάνιση του αποτελέσματος
11: echo $result;
12: ?>
```

Ίσως αναρωτιέστε γιατί χρειάζεται να εμφανίσουμε το αποτέλεσμα, εάν το μόνο που κάνει το script είναι να εισάγει κάποια δεδομένα. Δεν είμαστε υποχρεωμένοι να το κάνουμε· το περιλαμβάνουμε για λόγους πληρότητας. Θα μπορούσαμε να κάνουμε αυτό το script λίγο πιο συμπαγές προσαρμόζοντας την γραμμή εκτέλεσης του ερωτήματος έτσι ώστε να εκτελεί απλώς το ερώτημα και να εμφανίζει μία απλή δήλωση για το εάν η εκτέλεση ολοκληρώθηκε επιτυχώς ή όχι, όπως βλέπετε στην Λίστα 3.7.

ΛΙΣΤΑ 3.7 Το Τροποποιημένο Script για την Εισαγωγή μιας Εγγραφής

```
1: <?php
2: / / άνοιγμα της σύνδεσης
3: $conn = mysql_connect ("10.10.90.20", "root", "0000")
;
4: / / επιλογή της βάσης δεδομένων
5: mysql_select_db("programma", $conn") ;
6: / / δημιουργία της εντολής SQL
7: $sql = INSERT INTO ekpaideutikos values (' ', 'some
value' , ' ' , ' ' ) ;
8: / / εκτέλεση της εντολής SQL
9: if (mysql_query ( $sql, $conn ) ) {
10:             echo  "record added!" ;
11: } else  {
12:             echo " something went wrong ";
13: }
14: ?>
```

Η εκτέλεση αυτού του script θα έχει σαν αποτέλεσμα την προσθήκη μιας εγγραφής στον *πίνακα* ekpaideutikos. Για να εισάγουμε περισσότερες εγγραφές πέρα από αυτή που παρουσιάζεται στο script, μπορούμε να δημιουργήσουμε μία μακροσκελή λίστα εντολών SQL με ενσωματωμένα τα επιθυμητά δεδομένα και κατόπιν να χρησιμοποιήσουμε την συνάρτηση mysql_query () πολλές φορές για την εκτέλεση αυτών των εντολών. Εναλλακτικά, μπορούμε να δημιουργήσουμε μία φόρμα για την εισαγωγή των δεδομένων στο script.

Για να δημιουργήσουμε την φόρμα γι' αυτό το script χρειάζόμαστε 4 πεδία .Η ενέργεια (action) για την φόρμα είναι το όνομα του script προσθήκης εγγραφών· ας το ονομάσουμε neos_ekpaid1.php. Ο HTML κώδικας για την φόρμα θα πρέπει να δείχνει παρόμοιος με αυτόν της Λίστας 3.8.

ΛΙΣΤΑ 3.8 Μία Φόρμα για την Εισαγωγή Εγγραφών

```
<html>
<head>
<title>Neos Ekpaid</title>
</head>
<body text="#FFFFFF" style="background-attachment:
fixed">

<p align="center"><font face="Arial Black"><b><u><i><span
lang="el">

<font size="5" color="#000000"><span style="background-
color: #FFFFFF">ΝΕΟΣ
ΕΚΠΑΙΔΕΥΤΙΚΟΣ</span></font></span></i></u></b></font></p>

<table WIDTH="980">
<tr>

    <td align="left" bgcolor="#FF0000"><form
ACTION="neos_ekpaid1.php"

                        METHOD="POST">

<p align="center"><font size="4"><b>Κωδικός
εκπαιδευτικού:</b></font><font color="#0000FF"><br>

<inputTYPE="int"NAME="id_ekpaid"SIZE="20"MAXLENGTH="30"><
/font></p>

<p align="center"><font
size="4"><b>Όνοματεπώνυμο:</b></font><font
color="#0000FF"><br><font color="#00FFFF">
```

```
<inputTYPE="varchar"NAME="onomatepwnymo"SIZE="20"MAXLENGT
H="30"></font></font></p>

<p align="center"><font size="4"><b>Ηλεκτρονικό
Ταχυδρομείο:</b></font><font color="#0000FF"><br>

<input TYPE="varchar" NAME="e_mail" SIZE="20"
MAXLENGTH="30"></font></p>

<p align="center"><font size="4"><b>Κωδικός Κατηγορίας
Εκπαιδευτικού:</b></font><font color="#0000FF"><br><font
color="#00FFFF">

<input TYPE="int" NAME="id_kathg_ekpaid" SIZE="20"
MAXLENGTH="30"></font></font></p>

<p align="center"><font color="#0000FF"> <font
size="4"><input TYPE="submit" VALUE="Καταχώρησε"></font>

<input type="reset" value="Reset" name="B1"> </font></p>

</form>

</td>

</tr>

</table>

</body>

</html>
```

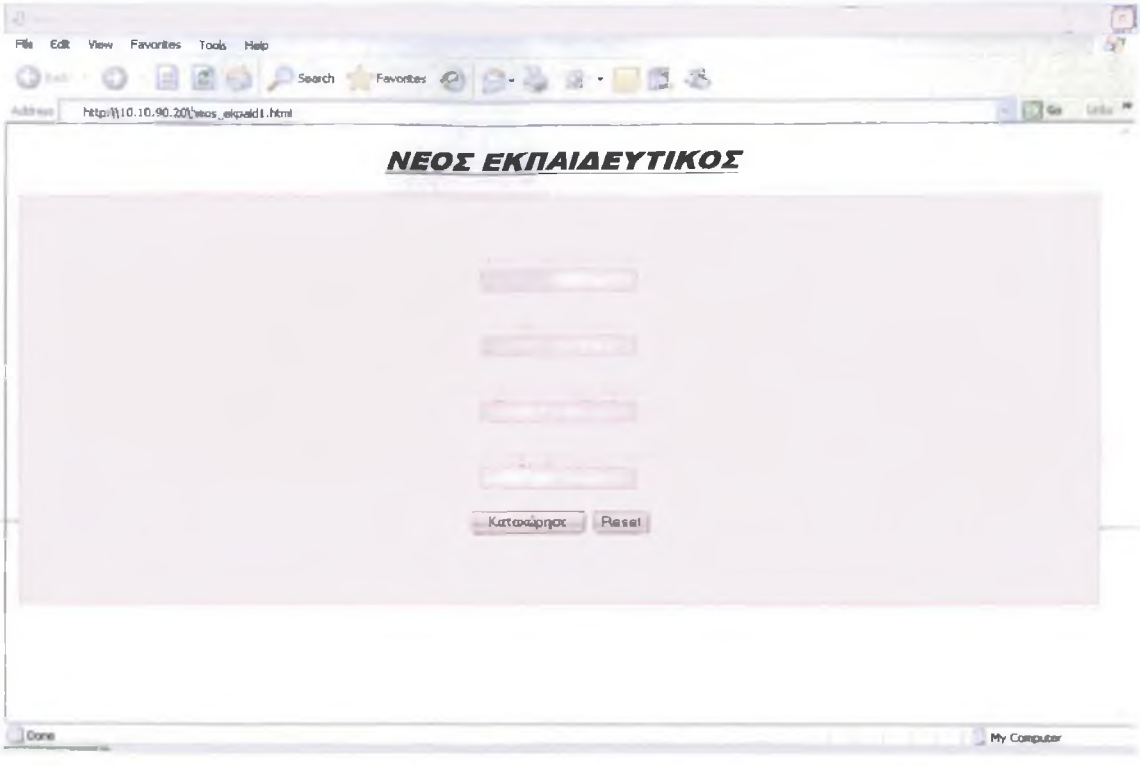
Αποθηκεύουμε αυτό το αρχείο με όνομα neos_ekpaid1.html και το τοποθετούμε στον αρχικό κατάλογο εγγράφων του Web server μας, δηλαδή στον κατάλογο htdocs του Apache. Στην συνέχεια δημιουργούμε το script neos_ekpaid1.php, όπως παρουσιάζεται στην Λίστα 3.9.Οι τιμές που εισάγει ο χρήστης στην φόρμα αντικαθιστά τις ενσωματωμένες στον κώδικα τιμές του ερωτήματος SQL με την βοήθεια μιας μεταβλητής η οποία ονομάζεται $_POST.

ΛΙΣΤΑ 3.9 Ενα Script Εισαγωγής Εγγραφών το Οποίο Παίρνει Δεδομένα από μία Φόρμα

```
<?php
$link = mysql_connect('10.10.90.20', 'root', '0000');

if (!$link) {

   die('Not connected : ' . mysql_error());

}
// make programma the current db
$db_selected = mysql_select_db('programma', $link);
if (!$db_selected) {

   die ('Can\'t use programma : ' . mysql_error());

}

$database="programma";
extract($_POST);
if ($id_ekpaid>53)

  {
$query = "insert into programma.ekpaideutikos values
('".$id_ekpaid."','".$onomatepwnymo."','".$e_mail."','".$
id_kathg_ekpaid."')";

  }

mysql_query($query) or die("Λάθος Στοιχεία.Δοκιμάστε
ξανά!");
print "Εκπαιδευτικός Καταχωρήθηκε!";

mysql_close($link);

?>
```

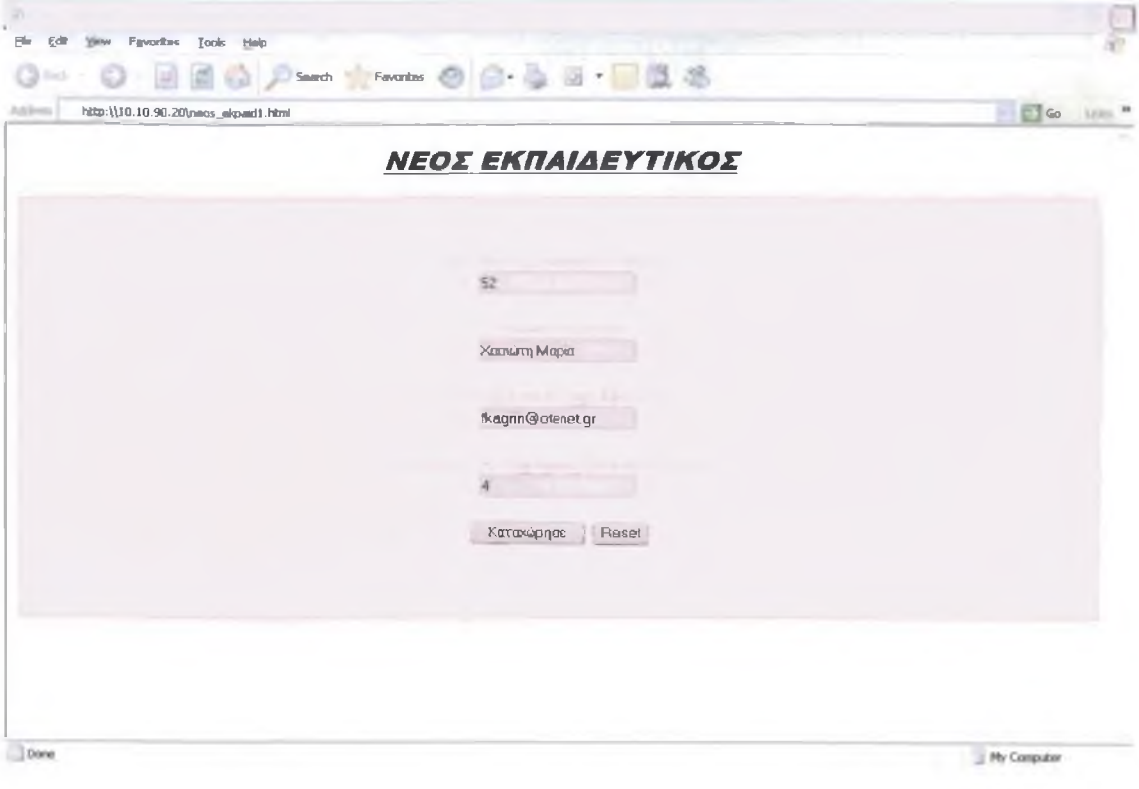
Αποθηκεύουμε το script με όνομα neos_ekpaid1.php, και το τοποθετούμε στον αρχικό κατάλογο εγγράφων του Web server μας, δηλαδή στον κατάλογο htdocs του Apache. Από το περιβάλλον μιας εφαρμογής Web browser, ανοίγουμε την HTML φόρμα που δημιουργήσαμε παραπάνω. Θα πρέπει να δείχνει παρόμοια με αυτή της Εικόνας 3.1.

ΕΙΚΟΝΑ 3.1 Η HTML φόρμα για την προσθήκη μιας εγγραφής.



Εισάγουμε τιμές στα πλαίσια της φόρμας , όπως βλέπετε στην Εικόνα 3.2.

ΕΙΚΟΝΑ 3.2 Πληκτρολόγηση κειμένου στο πεδίο της φόρμας.

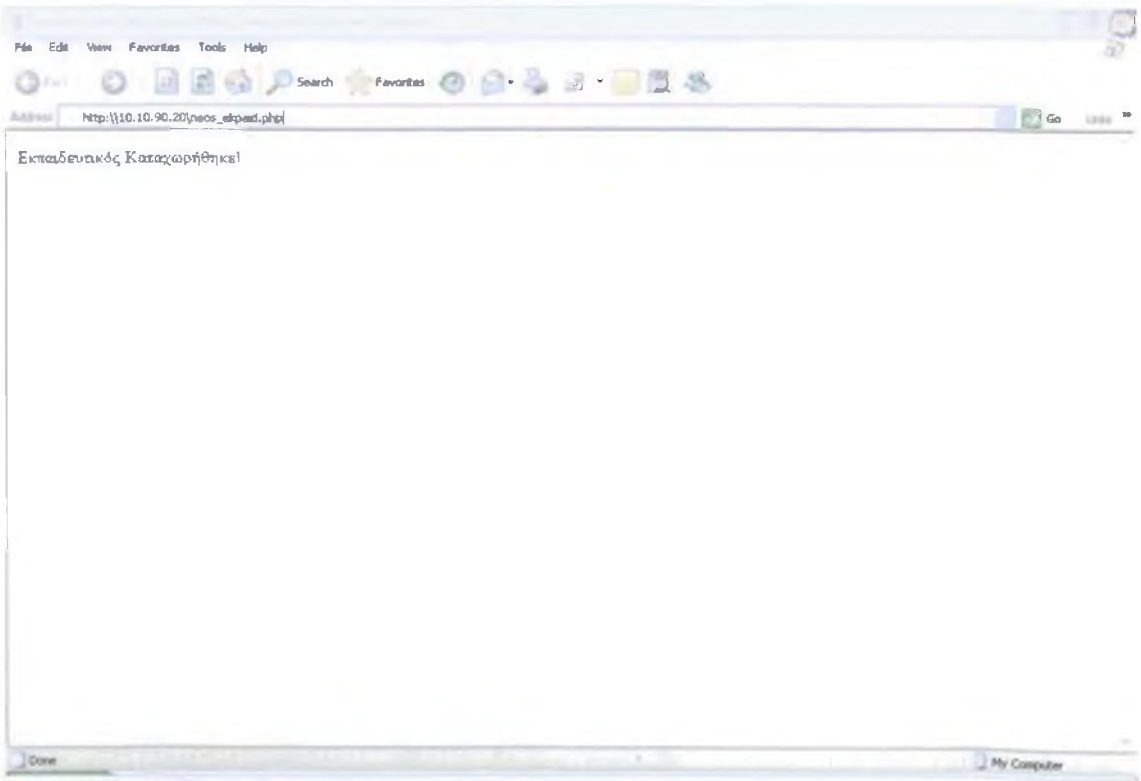


Τέλος, κάνουμε κλικ στο κουμπί Καταχώρησε (εισαγωγή εγγραφής) για να εκτελέσουμε το script και να εισάγουμε την εγγραφή στον πίνακα. Εάν η διαδικασία ολοκληρωθεί επιτυχώς θα δούμε αποτελέσματα παρόμοια με αυτά της Εικόνας 3.3. ενώ εάν έχουμε κάνει λάθος θα δούμε αποτελέσματα σαν αυτά της Εικόνας 3.4.

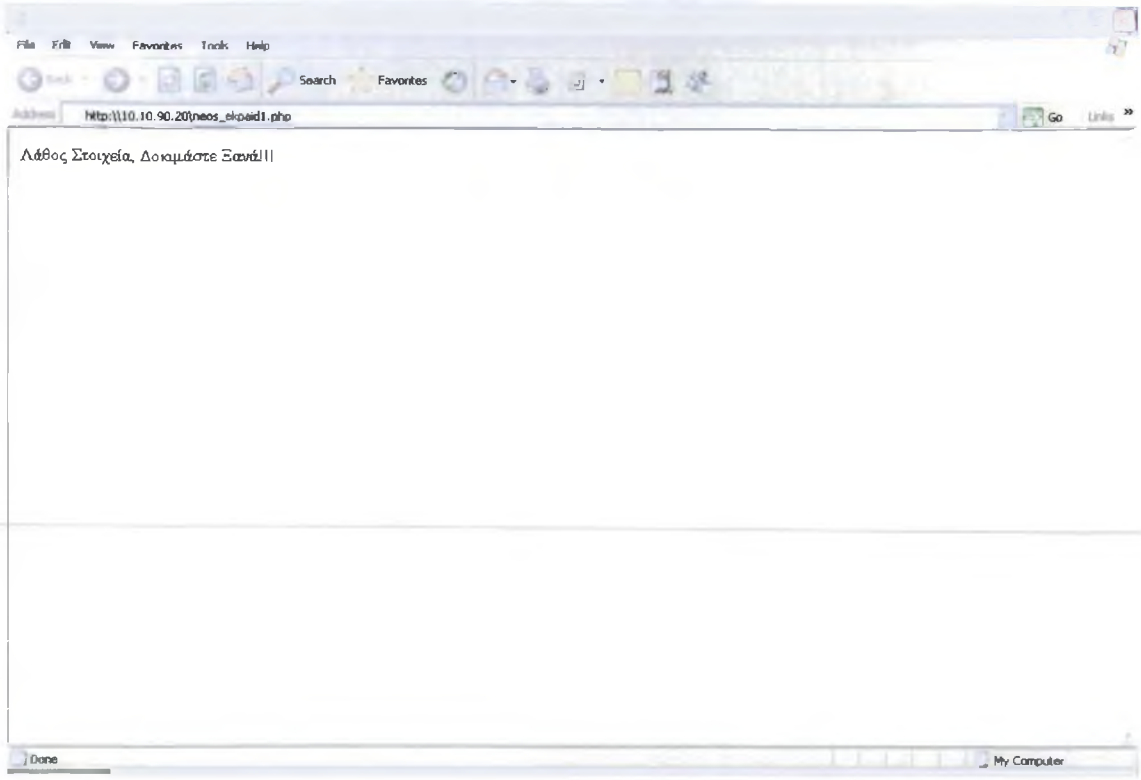
Για να ελέγξουμε την δουλειά μας, μπορούμε να χρησιμοποιήσουμε το περιβάλλον γραμμής εντολής της MySQL για να εξετάσουμε τις εγγραφές του πίνακα:

mysql> select * from ekpaideutikos ;

ΕΙΚΟΝΑ 3.3 Η εγγραφή έχει προστεθεί στον πίνακα.



ΕΙΚΟΝΑ 3.4 Η εγγραφή δεν έχει προστεθεί στον πίνακα.Μήνυμα λάθους.



Ανασκόπηση

Η συνδυασμένη χρήση των PHP και MySQL για την δημιουργία δυναμικών εφαρμογών Web με υποστήριξη βάσεων δεδομένων είναι εύκολη υπόθεση. Πρέπει να θυμόμαστε απλώς ότι οι συναρτήσεις της PHP είναι ουσιαστικά μία πύλη μέσω της οποίας επικοινωνούμε με τον server βάσεων δεδομένων· οτιδήποτε θα μπορούσαμε να εισάγουμε από το περιβάλλον γραμμής εντολής της MySQL, μπορεί επίσης να χρησιμοποιηθεί με την σινάρτηση mysql_query().

Για να συνδεθούμε στην MySQL χρησιμοποιώντας την PHP, θα πρέπει να γνωρίζουμε το όνομα χρήστη και τον κωδικό πρόσβασής μας για την MySQL, καθώς και το όνομα της βάσης δεδομένων. Χρησιμοποιώντας τις συναρτήσεις mysql_connect() και mysql_select_db(), μπορούμε να συνδεθούμε στην MySQL και να επιλέξουμε την βάση δεδομένων που θέλουμε να χρησιμοποιήσουμε καθ' όλη την διάρκεια ζωής του script.

Αφού συνδεθούμε, μπορούμε να εισάγουμε στάνταρ εντολές SQL χρησιμοποιώντας την συνάρτηση mysql_query(). Εάν έχουμε εισάγει μία εντολή SELECT, μπορούμε να χρησιμοποιήσουμε την συνάρτηση mysql_numrows() για να μετρήσουμε τις εγγραφές που επιστρέφει το ερώτημα. Εάν θέλουμε να εμφανίσουμε τα δεδομένα που ανακτήσαμε, μπορούμε να χρησιμοποιήσουμε την συνάρτηση mysql_fetch_array() για να ανακτήσουμε (με έναν βρόχο) τις εγγραφές μία προς μία από το αποτέλεσμα του ερωτήματος και να τις εμφανίσουμε στην οθόνη.

ΚΕΦΑΛΑΙΟ 4

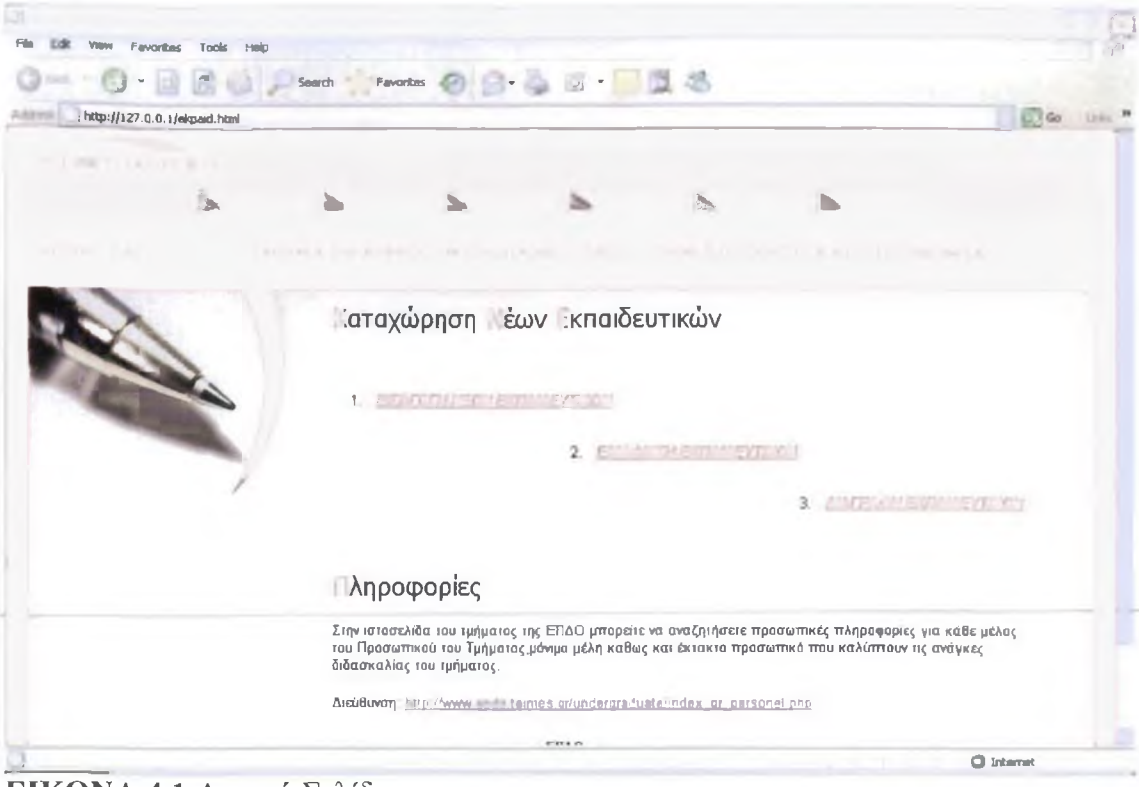
4.1 Συγγραφή Κώδικα – Παρουσίαση Έργου Επεξεργασίας Έκτακτων Εκπαιδευτικών

Σ' αυτό το κεφάλαιο θα ασχοληθούμε με τη δημιουργία μιας κεντρικής σελίδας HTML απ'όπου θα γίνεται καταχώρηση νέων έκτακτων εκπαιδευτικών στο τμήμα ΕΠΔΟ του ΤΕΙ Μεσολογγίου. Επίσης θα μπορούμε να πραγματοποιήσουμε και άλλες ενέργειες όπως διαγραφή εκπαιδευτικών και εμφάνιση του πίνακα με τους υπάρχοντες εκπαιδευτικούς. Όλα τα PHP scripts που παρουσιάσαμε θα μας φανούν πολύ χρήσιμα διότι γνωρίζουμε τα βασικά για την συγγραφή κώδικα και μπορούμε να χρησιμοποιούμε μεταβλητες και διατάξεις, να δημιουργούμε και να καλούμε συναρτήσεις και να συνδεόμαστε στην MySQL για να κάνουμε εκπληκτικά πράγματα με μία βάση δεδομένων. Στο Word Wide Web, οι φόρμες της HTML είναι το βασικό μέσο με το οποίο στέλνονται οι πληροφορίες από τον χρήστη στον server.

Δημιουργία Κεντρικής Σελίδας

Για την ευκολότερη περιήγησή μας στο περιβάλλον της επξεργασίας των έκτακτων εκπαιδευτικών, θα δημιουργήσουμε μία κεντρική ιστοσελίδα απ'όπου θα μπορούμε να περιηγηθούμε στις διάφορες λειτουργίες των εκπαιδευτικών όπως εισαγωγή νέων εκπαιδευτικών, διαγραφή και εμφάνιση της λίστας με τους υπάρχοντες εκπαιδευτικούς.

Όλες τις σελίδες HTML τις επεξεργαζόμαστε στο Microsoft Office FrontPage. Η κεντρική σελίδα μας θα είναι κάπως έτσι:



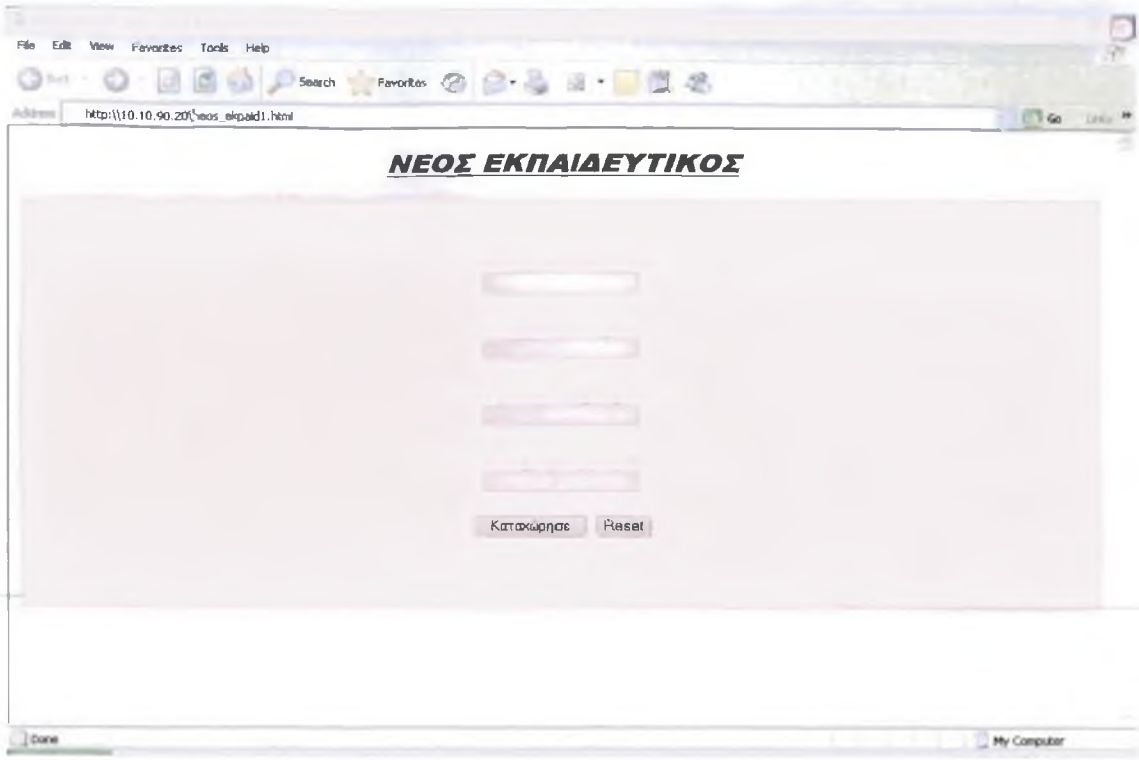
ΕΙΚΟΝΑ 4.1 Αρχική Σελίδα

Στον κώδικα της κεντρικής σελίδας προσθέσαμε τον παρακάτω κώδικα όπου δηλώνουμε τα hyperlink τα οποία θα μας οδηγήσουν σε άλλες σελίδες :

```
File Edit Format View Help
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-7">
</head>

<body >
<h2 align="center">ΚΑΤΑΧΩΡΗΣΗ ΝΕΩΝ ΕΚΠΑΙΔΕΥΤΙΚΩΝ</h2>
<p align="center"> </p>
<ol>
  <li>
    <div align="center"><a href="neos_ekpaid1.html">ΕΙΣΑΓΩΓΗ ΝΕΩΝ ΕΚΠΑΙΔΕΥΤΙΚΩΝ</a></div>
  </li>
  <li>
    <div align="center"><a href="show_ekpaid.php">ΕΜΦΑΝΙΣΗ ΕΚΠΑΙΔΕΥΤΙΚΩΝ</a></div>
  </li>
   <li>
    <div align="center"><a href="delete_ekpaid.php">ΔΙΑΓΡΑΦΗ ΕΚΠΑΙΔΕΥΤΙΚΩΝ</a></div>
  </li>
</ol>
</body>
</html>
```

ΕΙΚΟΝΑ 4.2. HTML Κώδικας των Υπερσυνδέσμων.

Πατώντας την επιλογή ΕΙΣΑΓΩΓΗ ΝΕΩΝ ΕΚΠΑΙΔΕΥΤΙΚΩΝ από την κεντρική σελίδα θα μας μεταφέρει σε μια άλλη σελίδα που περιλαμβάνει μια φόρμα καταχώρησης νέων εκπαιδευτικών. Ο κρυφός κώδικας που υπάρχει θα μας συνδέσει με την βάση μας και θα μεταφέρει ,όλα τα στοιχεία της φόρμας που εισάγουμε, απευθείας στον πίνακα ekpaideutikos της βάσης μας.



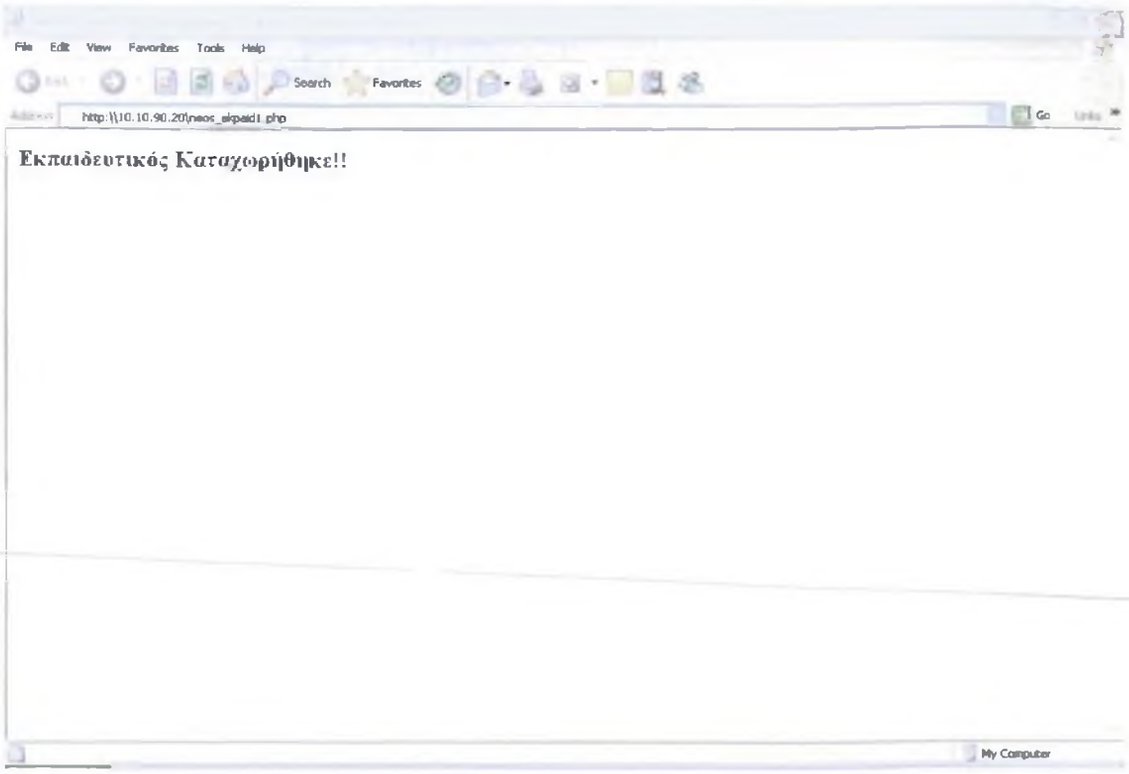
ΕΙΚΟΝΑ 4.3. Η HTML φόρμα για την προσθήκη μιας εγγραφής.

Όπως έχουμε ήδη αναφέρει σε προηγούμενο κεφάλαιο, δημιουργούμε το script neos_ekpaid1.php, όπως παρουσιάζεται στην παρακάτω ΕΙΚΟΝΑ, όπου οι τιμές που εισάγει ο χρήστης στην φόρμα αντικαθιστά τις ενσωματωμένες στον κώδικα τιμές του ερωτήματος SQL με την βοήθεια μιας μεταβλητής η οποία ονομάζεται $_POST. Αποθηκεύουμε αυτό το αρχείο με όνομα neos_ekpaid1.php και το τοποθετούμε στον αρχικό κατάλογο εγγράφων του Web server μας, δηλαδή στον κατάλογο htdocs του Apache.

```
File  Edit  Format  View  Help
<?php

$link = mysql_connect('194.177.216.138', 'root', '0000');
if (!$link) {
   die('Not connected : ' . mysql_error());
}

// make programma the current db
$db_selected = mysql_select_db('programma', $link);
if (!$db_selected) {
   die ('Can\'t use programma : ' . mysql_error());
}
$database="programma";
extract($_POST);
if ($id_ekpaid>53)
  {

$query = "insert into programma.ekpaideutikos values
        ('".$id_ekpaid."','".$onomatepwnymo."','".$e_mail."','".$id_kathg_ekpaid."')";
  }


mysql_query($query) or die("Λάθος Στοιχεία.Δοκιμάστε ξανά!");

print "Εκπαιδευτικός Καταχωρήθηκε!";


mysql_close($link);
?>
```

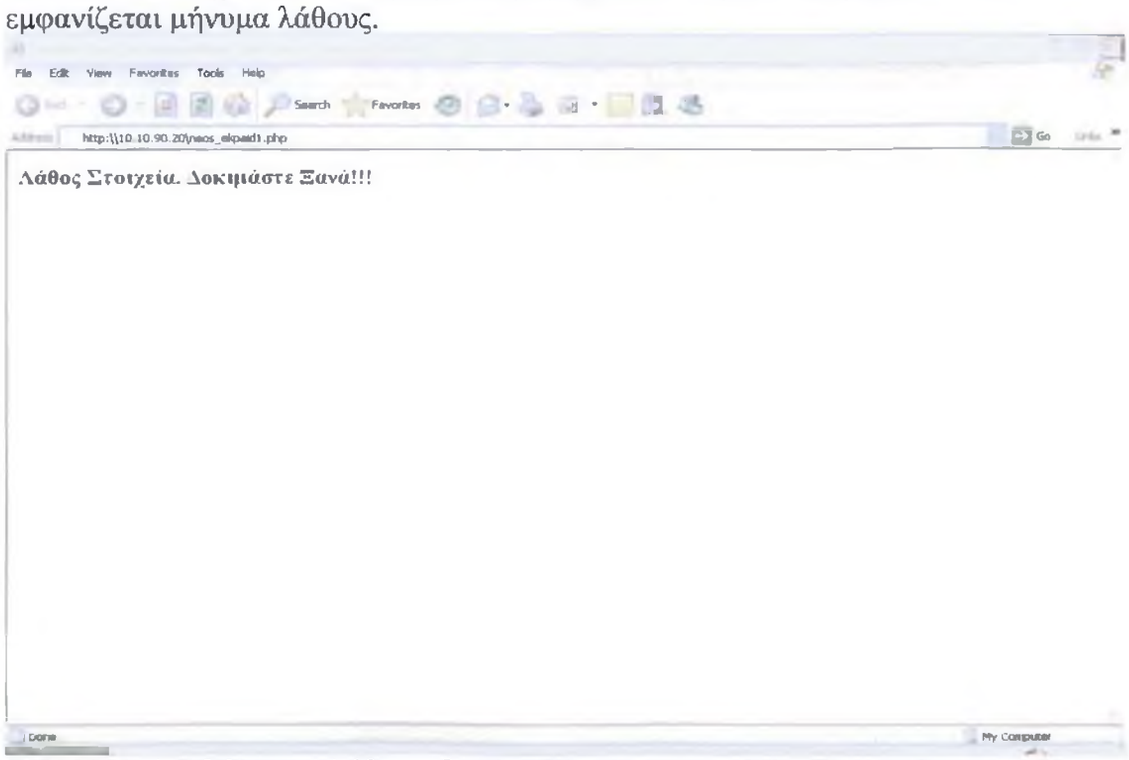
ΕΙΚΟΝΑ 4.4 Ένα Script Εισαγωγής Εγγραφών.

Άν τα στοιχεία τα οποία έχουμε εισάγει στη φόρμα είναι σωστά και τειρούν τις εντολές του script neos_ekpaid1.php τότε εμφανίζεται ένα μήνυμα που μας λέει ότι ο εκπαιδευτικός έχει καταχωρηθεί στη βάση δεδομένων μας ,όπως φαίνεται παρακάτω:



ΕΙΚΟΝΑ 4.5 Η Εγγραφή έχει Προστεθεί στον Πίνακα.

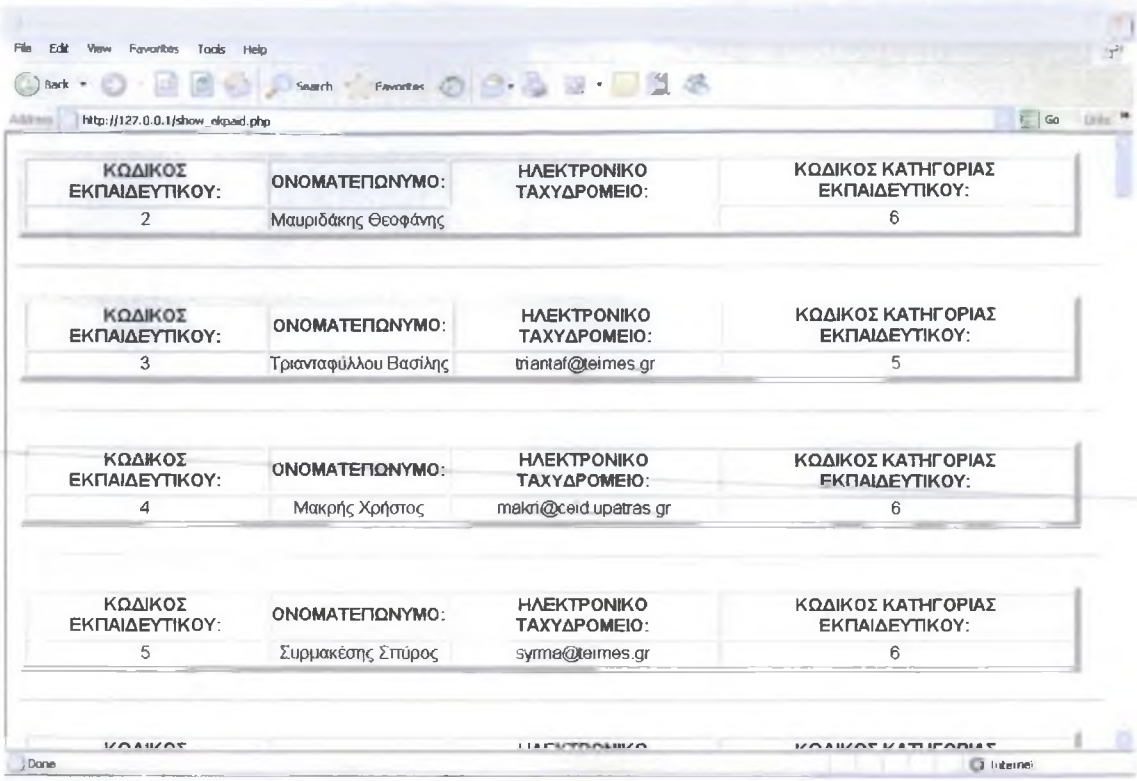
Αν πάλι τα στοιχεία που εισάγουμε στην φόρμα δεν τειρούν τους όρους τότε εμφανίζεται μήνυμα λάθους.



ΕΙΚΟΝΑ 4.6 Μήνυμα Λάθους, δεν έχει Πραγματοποιηθεί η Εγγραφή.

Αν θέλουμε να επιβεβαιώσουμε την εγγραφή μας και γενικά να δούμε την λίστα με τους καταχωρημένους εκπαιδευτικούς πατάμε την επιλογή ΕΜΦΑΝΙΣΗ

ΕΚΠΑΙΔΕΥΤΙΚΩΝ και εμφανίζεται ο πίνακας με όλες τις εγγραφές, όπως φαίνεται στην εικόνα:



ΚΩΔΙΚΟΣ ΕΚΠΑΙΔΕΥΤΙΚΟΥ:	ΟΝΟΜΑΤΕΠΩΝΥΜΟ:	ΗΛΕΚΤΡΟΝΙΚΟ ΤΑΧΥΔΡΟΜΕΙΟ:	ΚΩΔΙΚΟΣ ΚΑΤΗΓΟΡΙΑΣ ΕΚΠΑΙΔΕΥΤΙΚΟΥ:
2	Μαυριδάκης Θεοφάνης		6

ΚΩΔΙΚΟΣ ΕΚΠΑΙΔΕΥΤΙΚΟΥ:	ΟΝΟΜΑΤΕΠΩΝΥΜΟ:	ΗΛΕΚΤΡΟΝΙΚΟ ΤΑΧΥΔΡΟΜΕΙΟ:	ΚΩΔΙΚΟΣ ΚΑΤΗΓΟΡΙΑΣ ΕΚΠΑΙΔΕΥΤΙΚΟΥ:
3	Τριανταφύλλου Βασίλης	triantaf@teimes.gr	5

ΚΩΔΙΚΟΣ ΕΚΠΑΙΔΕΥΤΙΚΟΥ:	ΟΝΟΜΑΤΕΠΩΝΥΜΟ:	ΗΛΕΚΤΡΟΝΙΚΟ ΤΑΧΥΔΡΟΜΕΙΟ:	ΚΩΔΙΚΟΣ ΚΑΤΗΓΟΡΙΑΣ ΕΚΠΑΙΔΕΥΤΙΚΟΥ:
4	Μακρής Χρήστος	makri@ceid.upatras.gr	6

ΚΩΔΙΚΟΣ ΕΚΠΑΙΔΕΥΤΙΚΟΥ:	ΟΝΟΜΑΤΕΠΩΝΥΜΟ:	ΗΛΕΚΤΡΟΝΙΚΟ ΤΑΧΥΔΡΟΜΕΙΟ:	ΚΩΔΙΚΟΣ ΚΑΤΗΓΟΡΙΑΣ ΕΚΠΑΙΔΕΥΤΙΚΟΥ:
5	Συρμακέσης Σπύρος	syrma@teimes.gr	6

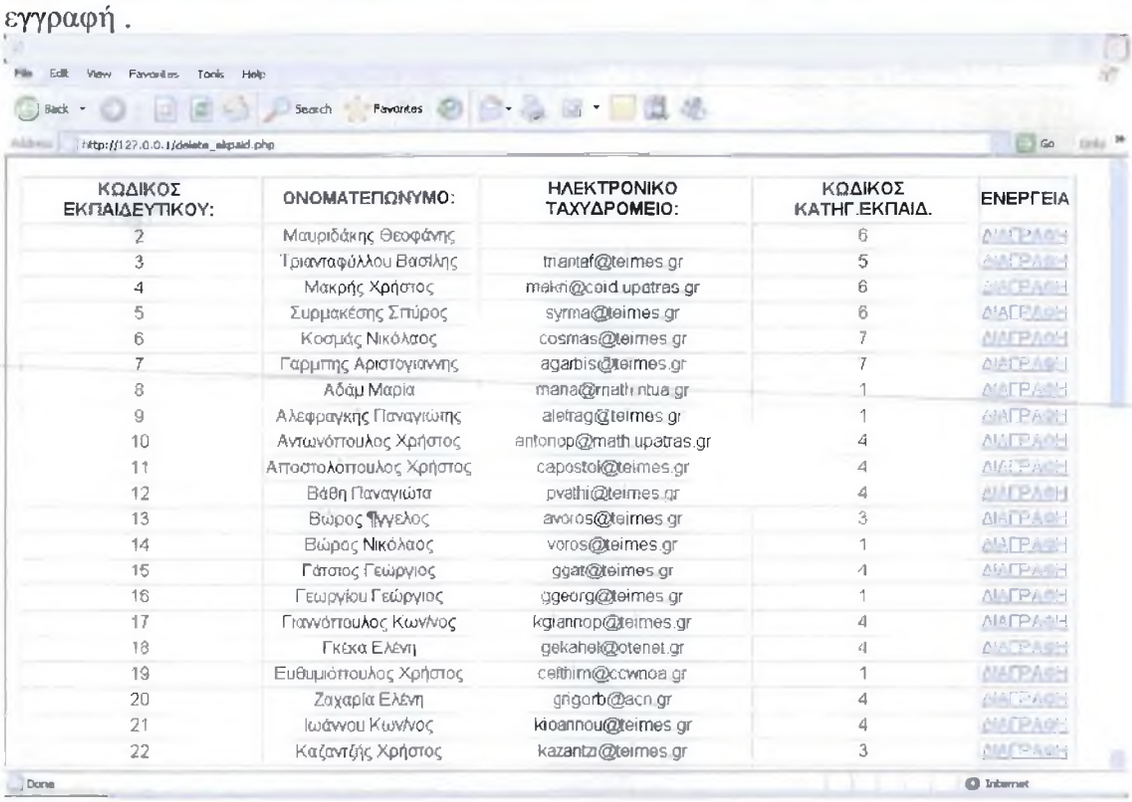
ΕΙΚΟΝΑ 4.7 Εμφάνιση Πίνακα Εκπαιδευτικών.

Το script που χρησιμοποιούμε το ονομάσαμε show_ekpaid.php, το αποθηκεύσαμε στον κατάλογο htdocs του Apache και είναι το εξής :

```
<HTML>
<HEAD>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-7">
</HEAD>
<BODY>
<?
mysql_connect('10.10.90.20','root','0000');
@mysql_select_db('programma') or die( "Unable to select database");

$get_list = ("SELECT *  FROM ekpaideutikos ORDER BY id_ekpaid " )  ;

$get_list_res = mysql_query($get_list) or die (mysql_error());

while ($line=mysql_fetch_array($get_list_res))
{
$id_ekpaid=$line["id_ekpaid"];
$onomatepwnymo=$line["onomatepwnymo"];
$e_mail=$line["e_mail"];
$id_kathg_ekpaid=$line["id_kathg_ekpaid"];

print"<table  celpadding=3 cellspacing=2 border=6 width=98%>
<tr>
    <th>ΚΩΔΙΚΟΣ ΕΚΠΑΙΔΕΥΤΙΚΟΥ:</th>
      <th>ΟΝΟΜΑΤΕΠΩΝΥΜΟ:</th>
            <th>ΗΛΕΚΤΡΟΝΙΚΟ ΤΑΧΥΔΡΟΜΕΙΟ:</th>
            <th>ΚΩΔΙΚΟΣ ΚΑΤΗΓΟΡΙΑΣ ΕΚΠΑΙΔΕΥΤΙΚΟΥ:</th>
            </tr>
    <td align=center>$id_ekpaid</td>

    <td align=center> $onomatepwnymo</td>

    <td align=center> $e_mail</td>

    <td  align=center> $id_kathg_ekpaid</td>
</table><br><hr><br>";

}
?>
</BODY>
</HTML>
```

ΕΙΚΟΝΑ 4.8 Το Script για την Εμφάνιση του Πίνακα Εκπαιδευτικών.

Μια άλλη επιλογή από την κεντρική σελίδα μας είναι η διαγραφή εκπαιδευτικών. Μπορούμε να διαγράψουμε εγγραφές που υπάρχουν καταχωρημένες στον πίνακα των εκπαιδευτικών.Όπως βλέπουμε στην εικόνα δίπλα από κάθε εγγραφή υπάρχει μια επιλογή που γράφει Διαγραφή και κάνοντας κλίκ πάνω της , σβύνει ολόκληρη την εγγραφή .



ΚΩΔΙΚΟΣ ΕΚΠΑΙΔΕΥΤΙΚΟΥ:	ΟΝΟΜΑΤΕΠΩΝΥΜΟ:	ΗΛΕΚΤΡΟΝΙΚΟ ΤΑΧΥΔΡΟΜΕΙΟ:	ΚΩΔΙΚΟΣ ΚΑΤΗΓ.ΕΚΠΑΙΔ.	ΕΝΕΡΓΕΙΑ
2	Μαυριδάκης Θεοφάνης		6	ΔΙΑΓΡΑΦΗ
3	Τριανταφύλλου Βασίλης	triantaf@teimes.gr	5	ΔΙΑΓΡΑΦΗ
4	Μακρής Χρήστος	makri@cord.upatras.gr	6	ΔΙΑΓΡΑΦΗ
5	Συρμακέσης Σπύρος	syrma@teimes.gr	6	ΔΙΑΓΡΑΦΗ
6	Κοσμάς Νικόλαος	cosmas@teimes.gr	7	ΔΙΑΓΡΑΦΗ
7	Γαρμπής Αριστογιαννης	agarbis@teimes.gr	7	ΔΙΑΓΡΑΦΗ
8	Αδάμ Μαρία	mana@rmath.ntua.gr	1	ΔΙΑΓΡΑΦΗ
9	Αλεφραγκής Παναγιώτης	aletrag@teimes.gr	1	ΔΙΑΓΡΑΦΗ
10	Αντωνόπουλος Χρήστος	antonop@math.upatras.gr	4	ΔΙΑΓΡΑΦΗ
11	Αποστολόπουλος Χρήστος	capostol@teimes.gr	4	ΔΙΑΓΡΑΦΗ
12	Βάθη Παναγιώτα	pvathi@teimes.gr	4	ΔΙΑΓΡΑΦΗ
13	Βώρος Άγγελος	avoros@teimes.gr	3	ΔΙΑΓΡΑΦΗ
14	Βώρας Νικόλαος	voros@teimes.gr	1	ΔΙΑΓΡΑΦΗ
15	Γάτσος Γεώργιος	ggat@teimes.gr	4	ΔΙΑΓΡΑΦΗ
16	Γεωργίου Γεώργιος	ggeorg@teimes.gr	1	ΔΙΑΓΡΑΦΗ
17	Γιαννόπουλος Κωνσνος	kgiannop@teimes.gr	4	ΔΙΑΓΡΑΦΗ
18	Γκέκα Ελένη	gekahek@otenet.gr	4	ΔΙΑΓΡΑΦΗ
19	Ευθυμιόπουλος Χρήστος	ceithim@ccwmoa.gr	1	ΔΙΑΓΡΑΦΗ
20	Ζαχαρία Ελένη	grigorb@acn.gr	4	ΔΙΑΓΡΑΦΗ
21	Ιωάννου Κωνσνος	kioannou@teimes.gr	4	ΔΙΑΓΡΑΦΗ
22	Καζαντζής Χρήστος	kazantzi@teimes.gr	3	ΔΙΑΓΡΑΦΗ

ΕΙΚΟΝΑ 4.9 Διαγραφή Εγγραφών από τον Πίνακα.

Για να γίνει αυτή η λειτουργία χρησιμοποιούμε ένα script με όνομα delete_ekpaid.php . Ο κώδικας που χρησιμοποιούμε είναι ο εξής :

```
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-7">
</head>
<body>
<?
mysql_connect('10.10.90.20','root','0000');
mysql_select_db('programma') or die( "Unable to select database");

$get_list = ("SELECT * FROM ekpaideutikos ORDER BY id_ekpaid " )  ;

$get_list_res = mysql_query($get_list) or die (mysql_error());

$display_block .= "
<table  celpadding=3 cellspacing=2 border=1 width=98%>
<tr>
<th>ΚΩΔΙΚΟΣ ΕΚΠΑΙΔΕΥΤΙΚΟΥ:</th>
<th>ΟΝΟΜΑΤΕΠΩΝΥΜΟ:</th>
<th>ΗΛΕΚΤΡΟΝΙΚΟ ΤΑΧΥΔΡΟΜΕΙΟ:</th>
<th>ΚΩΔΙΚΟΣ ΚΑΤΗΓ.ΕΚΠΑΙΔ.</th>
<th>ΕΝΕΡΓΕΙΑ</th>
</tr>";

while ($line=mysql_fetch_array($get_list_res))
{
$id_ekpaid=$line["id_ekpaid"];
$onomatepwnymo=$line["onomatepwnymo"];
$e_mail=$line["e_mail"];
$id_kathg_ekpaid=$line["id_kathg_ekpaid"];
```

Συνέχεια κώδικα:

```
File Edit Format View Help

$display_block .= " <tr>
<td align=center>$id_ekpaid<br></td>
<td align=center>$onomatepwnymo<br></td>
<td align=center>$e_mail<br></td>
<td align=center>$id_kathg_ekpaid<br></td>
<td align=center><a href=\"remove_from_ekpaid.php?id_ekpaid=$id_ekpaid\">ΔΙΑΓΡΑΦΗ</a></td>
</tr>";
}
$display_block .= "</table>";
?>
<?
print $display_block;
print "<br><br><a href=\"ekpaid.html\">ΕΠΙΣΤΡΟΦΗ ΣΤΗΝ ΑΡΧΙΚΗ ΣΕΛΙΔΑ</a>";
?>
</body>
</html>
```

ΕΙΚΟΝΑ 4.10 Το Script για την Διαγραφή Εγγραφών.

Στο script delete_ekpaid.php υπάρχει ένα ακόμα συνδεόμενο script με όνομα remove_from_ekpaid.php.Αυτό το script εμφανίζει μήνυμα επιτυχίας ή όχι της διαγραφής εκπαιδευτικού και επίσης περιλαμβάνει δύο link για να επιστρέψουμε στο πίνακα διαγραφής ή στην αρχική σελίδα.Ο κώδικας του script remove_from_ekpaid φαίνεται στην εικόνα :

```
File Edit Format View Help
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-7">
</head>
<body>
<?

mysql_connect('10.10.90.20','root','0000');
mysql_select_db('programma') or die( "Unable to select database");

if  ($_GET[id_ekpaid] != "") {
$delete_item = "DELETE FROM ekpaideutikos WHERE id_ekpaid=$_GET[id_ekpaid] ";
if (mysql_query($delete_item) )
{
echo "Η ΕΓΓΡΑΦΗ ΔΙΑΓΡΑΦΗΚΕ ΕΠΙΤΥΧΩΣ! <br><br><a href=\"delete_ekpaid.php\">ΕΠΙΣΤΡΟΦΗ ΣΤΗΝ
ΔΙΑΓΡΑΦΗ ΕΚΠΑΙΔΕΥΤΙΚΟΥ</a><br><br><a href=\"ekpaid.html\">ΕΠΙΣΤΡΟΦΗ ΣΤΗΝ ΑΡΧΙΚΗ
ΣΕΛΙΔΑ</a>";

}else {
echo "Η ΔΙΑΓΡΑΦΗ ΔΕΝ ΕΓΙΝΕ ΕΠΙΤΥΧΩΣ , ΥΠΑΡΧΕΙ ΣΦΑΛΜΑ!!<br><br><a
href=\"neos_ekpaid1.html\">ΕΠΙΣΤΡΟΦΗ ΣΤΗΝ ΕΙΣΑΓΩΓΗ  ΝΕΟΥ ΕΚΠΑΙΔΕΥΤΙΚΟΥ</a><br><br><a
href=\"ekpaid.html\">ΕΠΙΣΤΡΟΦΗ ΣΤΗΝ ΑΡΧΙΚΗ ΣΕΛΙΔΑ</a>";
}
}
?>
</body>
</html>
```

ΕΙΚΟΝΑ 4.11 Script για την Εμφάνιση Μηνυμάτων μετά την Διαγραφή.

Ανασκόπηση

Στο σημείο αυτό έχουμε στην διάθεσή μας τα εργαλεία που χρειαζόμαστε για να δημιουργήσουμε πραγματικά εξελιγμένα και διαλογικά περιβάλλοντα. Σ'αυτό το κεφάλαιο μάθαμε πώς να χειριζόμαστε τα δεδομένα που εισάγουν οι χρήστες σε φόρμες και πώς να στέλνουμε τα αποτελέσματα μιας φόρμας στην βάση δεδομένων μας μέσω μιας εφαρμογής Web browser με την βοηθεια ενός PHP script. Δημιουργήσαμε μία κεντρική σελίδα HTML που περιλαμβάνει επιλογές για την εισαγωγή, διαγραφή και εμφάνιση των έκτακτων εκπαιδευτικών καθώς επίσης και πληροφορίες σχετικά με το site του τμήματος ΕΠΔΟ. Επίσης δείχνουμε τον κώδικα που χρησιμοποιήσαμε για την δημιουργία φόρμας καταχώρησης εκπαιδευτικών και τα PHP scripts για την εισαγωγή, διαγραφή και εμφάνιση των έκτακτων εκπαιδευτικών του τμήματος ΕΠΔΟ.

ΒΙΒΛΙΟΓΡΑΦΙΑ – ΔΙΚΤΥΑΚΟΙ ΤΟΠΟΙ

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. Julie C. Meloni " ΜΑΘΕΤΕ PHP , MySQL και Apache σε 24 Ώρες "
2. Tomson , Welling " Ανάπτυξη Web Εφαρμογών με PHP και MySQL "
3. C. J. Date " Εισαγωγή στα Συστήματα ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ "

ΔΙΚΤΥΑΚΟΙ ΤΟΠΟΙ

1. www.phpwin.gr/downloads/manuals/php_manual_el.html
2. www.php.net
3. www.phpmyadmin.net
4. www.apache.com
5. www.mysql.com
6. www.google.gr/world_wide_web
7. www.lis.upatras