Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων

Σύστημα εξέτασης σχολής οδηγών (PHP&MySQL)

Ηράκλειο – Μάρτιος 2013

ΚΕΦΑΛΑΙΟ 1 - Εισαγωγή

Ευχαριστίες:

Θα ήθελα να ευχαριστήσω την οικογένεια μου και τους φίλους μου για την στήριξη και την καθοδήγησή τους καθ' όλη της διάρκεια συγγραφής της εργασίας αυτής.

Abstract:

The following paper's objective is to create a web application using PHP & MySQL programming. The technology used is the one dominating the World Wide Web (www/internet) and is used to develop a platform where Driver Licence tests are taken and instantly valuated.

In the following chapters of this paper the theoretical part of those technologies is provided, as well as a presentation of the application on how it functions (screenshots provided) and part of the code created.

Σύνοψη:

Ο στόχος της παρακάτω διατριβής είναι να δημιουργήσει μια διαδικτυακή εφαρμογή χρησιμοποιόντας προγραμματισμό σε PHP & MySQL. Η τεχνολογία που χρησιμοποιήθηκε είναι μια από τις επικρατέστερες στο Διαδύκτιο (WWW) και τη χρησιμοποιήσαμε για να κατασκευάσουμε μια πλατφόρμα όπου θα είναι δυνατή η άμεση εξέτασης για Δίπλωμα Οδήγησης και θα αξιολογείται άμεσα.

Στα παρακάτω κεφάλαια αυτής της εργασίας παρέχεται το θεωρητικό μέρος αυτών των τεχνολογιών, όπως και μια παρουσίαση του τρόπου λειτουργίας της εφαρμογής (παρέχονται εικόνες) και μέρος του κώδικα που δημιουργήθηκε.

Εισαγωγή:

Η παρούσα εργασία έγινε με κίνητρο την διευκόλυνση καθημερινών δραστηριοτήτων ή εργασιών με τη βοήθεια της τεχνολογίας. Στις μέρες μας η τεχνολογία μας επιτρέπει να κάνουμε διάφορες εργασίες με μεγαλύτερη ταχύτητα και πολύ μεγαλύτερη ακρίβεια.

Σε ακολουθία με το παραπάνω σκεπτικό, αποφάσισα για τη δημιουργία μιας διαδικτυακής εφαρμογής που θα επιτρέπει στον χρήστη να πραγματοποιεί εξέταση αντίστοιχη των σχολών οδηγών και να εμφανίζει άμεσα τα αποτελέσματα στον εξεταζόμενο. Έτσι αποφεύγεται η επίπονη διαδικασία της εξέτασης, της διόρθωσης, όπως και της αναμονής για τα αποτελέσματα.

Παρακάτω θα δούμε πόσο αποτελεσματικά και γρήγορα μπορεί να συμβεί αυτό, δίνοντας ένα πολύ καλό παράδειγμα για το πόσο χρήσιμες είναι οι νέες τεχνολογίες και το πόσα μπορούμε να κερδίσουμε από αυτές...

Πίνακας Περιεχομένων:

ΚΕΦΑΛΑΙΟ 2 – Ανάλυση τεχνολογιών

2.1 HTML

Η HTML (ακρωνύμιο του αγγλικού HyperText Markup Language ή αλλιώς Γλώσσα Σήμανσης Υπερκειμένου) είναι η κύρια γλώσσα σήμανσης για τις ιστοσελίδες, και τα στοιχεία της είναι τα βασικά δομικά στοιχεία των ιστοσελίδων.
Η HTML γράφεται υπό μορφή στοιχείων HTML τα οποία αποτελούνται από ετικέτες, οι οποίες περικλείονται μέσα σε σύμβολα «μεγαλύτερο από» και «μικρότερο από» (για παράδειγμα <html>), μέσα στο περιεχόμενο της ιστοσελίδας. Οι ετικέτες HTML συνήθως λειτουργούν ανά ζεύγη, με την πρώτη να ονομάζεται ετικέτα έναρξης και τη δεύτερη ετικέτα λήξης. Ανάμεσα στις ετικέτες, οι σχεδιαστές ιστοσελίδων μπορούν να τοποθετήσουν κείμενο, πίνακες, εικόνες κλπ.

World Wide Web
Από τα τέλη του 1940, οι άνθρωποι είχαν ονειρευτεί μια παγκόσμια βάση δεδομένων με την έννοια, ότι θα μπορούσε οποιοσδήποτε από οποιοδήποτε μέρος του κόσμου να αντλεί πληροφορίες και να συνδέεται με πολλές πηγές δεδομένων έτσι ώστε κάθε σημαντική πληροφορία να γίνεται άμεσα προσιτή στον χρήστη.
Το πιο δημοφιλές σήμερα τέτοιο σύστημα είναι το World Wide Web (WWW). Ο επίσημος ορισμός του είναι "μια πρωτοβουλία άντλησης πληροφορίας ευρέος περιεχομένου με σκοπό την παγκόσμια πρόσβαση σε ένα μεγάλο πλήθος αρχείων". Πιο απλά είναι ένα δίκτυο υπολογιστών βασισμένο στο Internet, που επιτρέπει στους χρήστες ενός υπολογιστή να έχουν πρόσβαση σε πληροφορίες που είναι αποθηκευμένες σε κάποιο άλλο υπολογιστή μέσω αυτού του παγκόσμιου δικτύου.
Το WWW ξεκίνησε το Μάρτιο του 1989 στο CERN, το Ευρωπαϊκό Εργαστήριο Φυσικής Σωματιδίων, που βρίσκεται στη Γενεύη. Το CERN είναι το σημείο συνάντησης φυσικών από όλο το κόσμο που συνεργάζονται σε μεγάλα προγράμματα φυσικής, μηχανολογίας και διεκπεραίωσης πληροφορίας. Επομένως, η μεγάλη γεωγραφική διασπορά των Πανεπιστημίων και Ινστιτούτων που συμμετέχουν στα προγράμματα αυτά και η ανάγκη για γρήγορο "μοίρασμα" της πληροφορίας, δημιούργησε την ανάγκη του συστήματος WWW.
Το 1995, 6 χρόνια μετά το CERN παρέδωσε το μέλλον του WWW στο World Wide Web Consortium που αποτελείται από το Γαλλικό Εθνικό Ινστιτούτο Έρευνας Επιστήμης Υπολογιστών και Ελέγχου (INRIA) και το Εργαστήριο Επιστήμης Υπολογιστών του Τεχνολογικού Ινστιτούτου της Μασαχουσέτης (MIT).

Είναι σημαντικό να αναφέρουμε ότι το WWW δεν θα υπήρχε καν χωρίς το Internet. Το Internet ξεκίνησε ως ένα ερευνητικό πρόγραμμα του Υπουργείου Άμυνας των ΗΠΑ στα 1970 και εξελίχθηκε σ' ένα παγκόσμιο δίκτυο-δικτύων, συνδέοντας πάνω από 100 χώρες. Το προσόν του είναι ότι καθοδηγείται από τον χρήστη και εξελίσσεται με Δημοκρατικό και Δαρβινικό τρόπο: τα καλά προϊόντα του δικτύου επιζούν ενώ τα φτωχά εξαφανίζονται. Το WWW αντικατέστησε πολυσύνθετες εντολές με απλές κινήσεις του ποντικιού, κάνοντας το Internet προσιτό σε όλους.

Document Type Declaration

Σύμφωνα με τα (X)HTML standards κάθε HTML αρχείο που χρησιμοποιούμε για να κατασκευάσουμε ιστοσελίδες – websites χρειάζεται ένα Document Type Declaration. Η «δήλωση τύπου εγγράφου», όπως είναι στα ελληνικά, είναι μία γραμμή κώδικα που τοποθετείται στο πάνω μέρος του κάθε HMTL αρχείου της ιστοσελίδας το οποίο ενημερώνει το W3C validator ποιά έκδοση (X)HTML χρησιμοποιείται.

Η παρουσία του DOCTYPE σε ένα HTML αρχείο είναι απαραίτητη γιατί πρώτον εγγυάται σωστό render στα προγράμματα πλοήγησης του διαδικτύου (browsers) και δεύτερον χωρίς αυτό δεν μπορεί η ιστοσελίδα σας να είναι έγκυρη ως προς το W3C. Το W3C είναι ο διεθνής οργανισμός προτύπων υπεύθυνος για το World Wide Web. Με την έλλειψη ή με τη λάθος σύνταξη του DOCTYPE, τα προγράμματα πλοήγησης του διαδικτύου διαβάζουν την ιστοσελίδα σας ως παλαιό και λανθασμένο κώδικα.

HTML5

Πρόκειται για την εξελιγμένη μορφή της αρχικής γλώσσας προγραμματισμού ιστοσελίδων, της HTML (HyperText Markup Language) η οποία δημιουργήθηκε το 1990. Η HTML5 ακόμα και σήμερα βρίσκεται υπό ανάπτυξη ή πιο απλά “υπό κατασκευή” μιας και μιλάμε για μια συνεχώς εξελισσόμενη τεχνολογία που ποτέ δεν θα είναι απόλυτα ολοκληρωμένη, σύμφωνα με τους ειδικούς.

Η ομάδα Web Hypertext Application Technology Working Group (WHATWG) ανέλαβε τον Ιούνιο του 2004 με μέλη της Apple, Mozilla Foundation και Opera Software την ανάπτυξη της HTML 5.

Η HTML5 δημιουργήθηκε με σκοπό να εξαλειφθεί η ανάγκη χρησιμοποίησης ιδιόκτητων προσθέτων και άλλων πανίσχυρων εφαρμογών όπως το Flash της Adobe και το SilverLight της Microsoft. Η HTML5 προοριζόταν για αντικατάσταση της HTML 4.01, της XHTML 1.0, και της DOM Level 2 HTML.

Το πρότυπο HTML5 υιοθετήθηκε ως αρχικό βήμα για τις εργασίες της νέας ομάδας εργασίας HTML του W3C το 2007. Αυτή η ομάδα εργασίας δημοσίευσε το Πρώτο Δημόσιο Working Draft του προτύπου στις 22 Ιανουαρίου 2008.

Οι συντάκτες της HTML5 είναι ο Ίαν Χίκσον της εταιρίας Google και ο Ντέιβ Χίατ της εταιρίας Apple.

Το μεγαλύτερο πλεονέκτημα της HTML5 είναι η χρησιμοποίηση SVG γραφικών, δηλαδή γραφικών που δεν αποτελούνται από pixels, αλλά από μαθηματικές συναρτήσεις δίνοντας την δυνατότητα στις εικόνες να μην επηρεάζονται από τις αναλύσεις των συσκευών των χρηστών. Αυτή η δυνατότητα παρέχεται μέσω της ετικέτας *(tag)* svg και του στοιχείου *(element)* canvas.

Το μεγαλύτερο μειονέκτημα της HTML5 είναι η ασυμβατότητα με τους παλιούς browsers *(browsers incombatibility)*, που όμως με την εξέλιξη της HTML5 δείχνει να εξαλείφεται.

Κανόνες

Κάποιοι βασικοί κανόνες που έχουν οριστεί για την HTML5 είναι:

- Βάση για τα νέα χαρακτηριστικά να είναι οι HTML, CSS, DOM, και η JavaScript
- Ελαχιστοποίηση των plugins (όπως το Flash)
- Καλύτερη λειτουργία εντοπισμού λαθών
- Περισσότερο markup για να αντικατασταθεί το scripting
- Πλήρη συμβατότητα ανεξαρτήτως συσκευής

Χαρακτηριστικά

Κάποια από τα νέα χαρακτηριστικά της HTML5 είναι:

- Το στοιχείο canvas για το drawing
- Τα στοιχεία video και audio για αναπαραγωγή πολυμέσων
- Νέα στοιχεία περιεχομένου όπως τα footer, header, nav και section
- Νέα στοιχεία δημιουργίας φόρμας όπως τα calendar, date, time, email, url και search

Το ιστορικό της HTML 5

2004: Η HTML 5 γεννιέται από την WHATWG (WHAT WORKING GROUP) που μέλη της είναι η Mozilla Foundation, η Apple και Opera Software.

2006: Η W3C ανακοινώνει την συνεργασία της με την WHATWG.

2008: Το πρώτο προσχέδιο της HTML 5 είναι γεγονός και είναι γραμμένο από τον Ίαν Χίκσον (Ian Hickson).

2008: Ο Firefox 3 γίνεται συμβατός με την HTML 5, λίγο αργότερα ακολουθούν google chrome, opera safari και φυσικά internet explorer.

2010: Το YouTube προσφέρει τον HTML 5 player.

2010: Ο Steve Jobs υποβαθμίζει με ανοιχτή επιστολή την τεχνολογία Flash και εξηγεί γιατί η Apple δεν πρόκειται να χρησιμοποιήσει την τεχνολογία αυτή στις συσκευές της.

> *" Η τεχνολογία Flash σχεδιάστηκε για υπολογιστές που χρησιμοποιούνται με «ποντίκια» και όχι για οθόνες αφής που χρησιμοποιούνται με δάχτυλα. "*

2011: Σπουδαίες εταιρίες, όπως η Disney, η Pandora και η Amazon, ξεκινούν να χρησιμοποιούν την HTML 5 όλο και περισσότερο.

2011: Η Adobe σταματάει την δημιουργία Flash για κινητά τηλέφωνα.

2012: Μεγαθήρια στον χώρο του internet όπως το Flickr.com και το Linkedin.com χρησιμοποιούν την HTML 5.

2013: Αναμένεται μέχρι τον Ιανουάριο του 2013 να πωληθούν πάνω από 1.000.000.000 κινητά συμβατά με HTML 5.

Το μέλλον της HTML

Η τρέχουσα έκδοση της HTML υπάρχει για περισσότερα από 10 χρόνια και δεν έχει καταφέρει να ακολουθήσει τους ρυθμούς πολλών εξελίξεων στην τεχνολογία. Το συμβούλιο για το World Wide Web (W3C), ο οργανισμός προτύπων που είναι υπεύθυνος για τη συντήρηση και ενημέρωση της HTML και άλλων προτύπων του web, εργάζεται για την ενημέρωση της γλώσσας και κυκλοφόρησε μια λειτουργική πρόχειρη έκδοση της HTML, την HTML5, τον

Οκτώβριο του 2009. Μια τελική έκδοση δεν έχει προγραμματιστεί ακόμα για τα επόμενα χρόνια.
Τα website και οι προγραμματιστές αλλάζουν και προσαρμόζονται στις τρέχουσες τεχνολογίες και τις πραγματικότητες των αγορών γρήγορα, αλλά οι υποκείμενες τεχνολογίες προχωρούν με πιο ψυχρό τρόπο. Οι κατασκευαστές προγραμμάτων περιήγησης είναι σίγουρο ότι θα υποστηρίζουν στα νέα τους έργα την HTML5. Όσοι το κάνουν νωρίς θα προσελκύσουν προγραμματιστές και χρήστες που τους ενδιαφέρει να έχουν ότι πιο καινούριο. Κάποιοι λένε ότι η πλήρης μετάβαση θα γίνει τουλάχιστον το 2020,ίσως και αργότερα. Σε κάθε περίπτωση, η συμβατότητα με την HTML 4.01θα παρέχεται στο μέλλον, ώστε οι παλιές σελίδες και τα παλιά site να μην εξαφανιστούν ξαφνικά.

2.2 CSS

Η CSS (Cascading Style Sheets-Διαδοχικά Φύλλα Στυλ) ή (αλληλουχία φύλλων στύλ) είναι μια γλώσσα υπολογιστή που ανήκει στην κατηγορία των γλωσσών φύλλων στυλ που χρησιμοποιείται για τον έλεγχο της εμφάνισης ενός εγγράφου που έχει γραφτεί με μια γλώσσα σήμανσης. Χρησιμοποιείται δηλαδή για τον έλεγχο της εμφάνισης ενός εγγράφου που γράφτηκε στις γλώσσες HTML και XHTML, δηλαδή για τον έλεγχο της εμφάνισης μιας ιστοσελίδας και γενικότερα ενός ιστοτόπου. Η CSS είναι μια γλώσσα υπολογιστή προορισμένη να αναπτύσσει στυλιστικά μια ιστοσελίδα δηλαδή να διαμορφώνει περισσότερα χαρακτηριστικά, χρώμματα, στοίχιση και δίνει περισσότερες δυνατότητες σε σχέση με την html. Για μια όμορφη και καλοσχεδιασμένη ιστοσελίδα η χρήση της CSS κρίνεται ως απαραίτητη.

Με την χρήση CSS μπορούμε να ορίζουμε χρώματα και μεγέθη οργανωμένα σε στυλ και έπειτα να εφαρμόζουμε τα στυλ αυτά στα στοιχεία των σελίδων του site μας. Με αυτόν τον τρόπο, κάθε φορά που αλλάζουμε το χρώμα ενός στυλ, αλλάζει το χρώμα όλων των στοιχείων που έχουν αναφορά στο στυλ αυτό. Έτσι αν έχουμε ορίσει ένα στυλ για το κύριο μενού του site, τότε θα χρειάζεται να αλλάξουμε το χρώμα του στυλ αυτού και αυτόματα θα εφαρμοστεί σε όλες τις σελίδες.

Εκτός από την ευκολία στην διαχείριση ενός site, ένα άλλο σημαντικό πλεονέκτημα της χρήσης CSS στις σελίδες είναι ο "καθαρότερος" κώδικας, χωρίς πολλές ιδιότητες στις ετικέτες οι οποίες τον κάνουν δυσανάγνωστο. Επιπλέον κάνει γρηγορότερη την πλοήγηση καθώς το αρχείο, μέσα στο οποίο ορίζονται τα στυλ, "διαβάζεται" από τον browser μόνο μια φορά και έπειτα αποθηκεύεται στην cache memory, μειώνοντας έτσι το μέγεθος της πληροφορίας που γίνεται download από τους browsers.

Η αλληλουχία εφαρμογής των φύλλων στύλ

Για ένα έγγραφο πχ xhtml θα υπάρχουν παραπάνω από ένα φύλλα στυλ τα οποία περιέχουν δηλώσεις για την εμφάνιση ενός συγκεκριμένου στοιχείου. Το Φύλλο στύλ που εφαρμόζεται σε ένα έγγραφο μπορεί να προέρχεται από :

- το συγγραφέα μιας ιστοσελίδας
- το χρήστη του πλοηγού
- τον ίδιο τον πλοηγό, αν έχει το δικό του προκαθορισμένο φύλλο στυλ

Συνεπώς για ένα xhtml στοιχείο θα υπάρχουν παραπάνω από μια δηλώσεις που πιθανόν να είναι συγκρουόμενες.Το πρότυπο css για να επιλύσει παρόμοιες συγκρούσεις έχει καθορίσει μια αλληλουχία-σειρά στην οποία θα μπούν αυτές οι δηλώσεις και με βάση την οποία θα επιλεγεί πχ η δήλωση που είναι πρώτη στη σειρά.

Ο αλγόριθμος δημιουργίας αυτής της σειράς-αλληλουχίας είναι ο ακόλουθος:

1. Βρες όλες τις δηλώσεις που εφαρμόζονται στο στοιχείο που μας ενδιαφέρει. Οι δηλώσεις εφαρμόζονται στο στοιχείο αν ο επιλογέας του το επιλέξει (ταιριάζει με αυτό).

2. Ταξινόμησε με βάση τη σημασία (κανονική ή σημαντική) και προέλευση (συγγραφέας , χρήστη ή πλοηγός χρήστη). Με αύξουσα σειρά προτεραιότητας:
   - Δηλώσεις πλοηγού χρήστη
   - Κανονικές δηλώσεις χρήστη
   - Κανονικές δηλώσεις συγγραφέα
   - Σημαντικές δηλώσεις συγγραφέα

3. Σημαντικές δηλώσεις χρήστη

4. Ταξινόμησε τις δηλώσεις ίδιας σημασίας και προέλευσης με κριτήριο την εξειδίκευση του επιλογέα: οι πιο εξειδικευμένοι επιλογείς υπερισχύουν των πιο γενικών. Τα ψευδό-στοιχεία και οι ψευδο-κλάσεις λογαριάζονται σαν κανονικά στοιχεία και κλάσεις αντίστοιχα.

5. Τέλος ταξινόμησε ανάλογα με τη σειρά καθορισμού: αν δύο δηλώσεις έχουν το ίδιο βάρος , προέλευση και εξειδίκευση , αυτή που προσδιορίστηκε τελευταία επικρατεί. Οι δηλώσεις σε εισαγώμενα φύλλα στυλ θεωρούνται ότι δηλώνονται πριν από τις δηλώσεις στο ίδιο το φύλλο στυλ .

Αφού λοιπόν προκύψει μια σειρά-αλληλουχία κανόνων εμφάνισης που αφορούν το ίδιο στοιχείο θα επιλεγεί προς εφαρμογή (για την αποφυγή συγκρούσεων) η δήλωση που θα είναι τελευταία στην σειρά που αναλύθηκε πιο πάνω.

2.3 PHP

Εισαγωγή στην PHP

Η PHP είναι μια γλώσσα προγραμματισμού ειδικά για την κατασκευή δυναμικών ιστοσελίδων. Με τον όρο δυναμική εννοείται μια ιστοσελίδα που αλλάζει αυτόματα, ανάλογα με τα στοιχεία του θεατή της. Στοιχεία όπως το λειτουργικό του σύστημα η διεύθυνση IP του κ.ά. Η PHP χρησιμοποιείται όχι για την διακόσμηση μιας ιστοσελίδας αλλά για τον χειρισμό των λειτουργιών και εργασιών που θα διεκπεραιώνει. Συνεπώς, ο κώδικας που γράφεται για μια ιστοσελίδα σε γλώσσα PHP δεν γίνεται άμεσα αντιληπτός αλλά μετά από την επέμβαση του θεατή στην ιστοσελίδα. Για να γίνει αυτό κατανοητό: η PHP χρησιμοποιείται ευρέως για τον χειρισμό ιστοσελίδων με δυνατότητες όπως η εγγραφή χρηστών (user registration), τα φόρουμ κ.ά. Λειτουργεί με την βοήθεια της HTML και πλέον και με την XHTML(νέα αναθεωρημένη έκδοση της HTML). Σε συνδυασμό και με την MySQL μπορεί να χρησιμοποιηθεί κάλλιστα για την διαχείριση δεδομένων μέσα σε βάσεις.

Ιστορία της PHP

Η ιστορία της PHP ξεκινά από το 1995, όταν ένας φοιτητής, ο Rasmus Lerdorf, δημιούργησε χρησιμοποιώντας τη γλώσσα προγραμματισμού Perl ένα απλό script με όνομα php.cgi για προσωπική χρήση. Το script αυτό είχε σαν σκοπό να διατηρεί μια λίστα στατιστικών για τα άτομα που έβλεπαν το online βιογραφικό του σημείωμα. Αργότερα, αυτό το script το διέθεσε και σε φίλους του οι οποίοι άρχισαν να του ζητούν να προσθέσει περισσότερες δυνατότητες. Η γλώσσα τότε ονομαζόταν PHP/FI από τα αρχικά Personal Home Page/Form Interpreter. Το 1997 η PHP/FI έφθασε στην έκδοση 2.0, βασιζόμενη αυτή τη φορά στη γλώσσα C και αριθμώντας περισσότερους από 50.000 ιστότοπους που τη χρησιμοποιούσαν, ενώ αργότερα την ίδια χρονιά οι Andi Gutmans και Zeev Suraski ξαναέγραψαν τη γλώσσα από την αρχή, βασιζόμενοι όμως αρκετά στην PHP/FI 2.0. Έτσι η PHP έφθασε στην έκδοση 3.0 η οποία θύμιζε περισσότερο τη σημερινή μορφή της. Στη συνέχεια, οι Zeev και Andi δημιούργησαν την εταιρεία Zend (από τα αρχικά των ονομάτων τους) η οποία συνεχίζει μέχρι και σήμερα την ανάπτυξη και εξέλιξη της γλώσσας PHP. Ακολούθησε το 1998 η έκδοση 4.0 της PHP, τον Ιούλιο του 2004 διατέθηκε η έκδοση 5.0, ενώ αυτή τη στιγμή έχουν ήδη διατεθεί και οι πρώτες δοκιμαστικές εκδόσεις της επερχόμενης PHP 6.0, για οποιονδήποτε προγραμματιστή θέλει

να τη χρησιμοποιήσει. Οι περισσότεροι ιστότοποι επί του παρόντος χρησιμοποιούν κυρίως τις εκδόσεις 4.0 και 5.0 της PHP.

Δυνατότητες της PHP

Η PHP επικεντρώνεται κυρίως στο server-side scripting. Έτσι, μπορούμε να κάνουμε ότι μπορεί να κάνει ένα άλλο CGI πρόγραμμα, όπως να μαζέψει δεδομένα, να παράγει δυναμικό περιεχόμενο σελίδων ή να στείλει και να λάβει 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 τρέξετε τον webserver, με μια συνδεδεμένη εγκατάσταση της PHP. Μπορείτε να προσπελάσετε τα αποτελέσματα του PHP προγράμματος με ένα web browser, βλέποντας την σελίδα PHP μέσα από τον server.

- **Command line scripting**. Μπορούμε να φτιάξουμε ένα PHP script για να το τρέξουμε χωρίς server ή browser. Χρειαζόμαστε μόνο τον PHP μεταγλωττιστή για να την χρησιμοποιήσουμε. 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 πολύ καλά PHP και θέλουμε να χρησιμοποιήσουμε κάποια προχωρημένα χαρακτηριστικά της PHP στις client-side εφαρμογές μας, μπορουμε 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.

Με την PHP δεν είμαστε περιορισμένοι να εξάγουμε HTML. 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.

Τα πλεονεκτήματα της PHP

PHP Hypertext Processor είναι ένα server-side web γλώσσα προγραμματισμού που μπορεί να ενσωματωθεί στην HTML. PHP χρησιμοποιεί είναι ευρέως διαδεδομένες, και μπορεί να περιλαμβάνει κάθε είδους λειτουργίες του διακομιστή που λαμβάνει εισόδου και οθόνες χρήστη ή χειρίζεται την είσοδο. PHP μπορεί να τρέξει και στα δύο UNIX και Windows servers, η οποία καθιστά πιο προσιτό από τα Windows (ASP). Αυτή η γλώσσα scripting αυξάνεται μέρα με τη μέρα. PHP5 πλήρως αντικειμενοστραφή γλώσσα και την ανεξαρτησία πλατφόρμας και την ταχύτητά του σε διακομιστή Linux βοηθά στη δημιουργία μεγάλων και πολύπλοκων εφαρμογών web.

PHP είναι μια ιδιαίτερα χρήσιμη γλώσσα προγραμματισμού, επειδή επιτρέπει την προηγμένη προγραμματισμό και είναι εύκολο να ενσωματωθούν με τις ιστοσελίδες. Ένα άλλο συν της PHP είναι ότι η γλώσσα διεπαφών πολύ καλά με MySQL, ένα δημοφιλές είδος της online βάση δεδομένων. Η MySQL είναι μια εμπορική εφαρμογή βάσης δεδομένων βαθμό που διατίθεται δωρεάν στο πλαίσιο του Open Source σε κανέναν. Ένα άλλο συν της PHP είναι ότι είναι ανοικτού κώδικα. Ο πραγματικός κώδικας που είναι η PHP είναι διαθέσιμη στο κοινό για δωρεάν, ενώ ο πηγαίος κώδικας για προϊόντα όπως τα ASP δεν είναι. Έτσι PHP είναι πολύ φθηνή. Επειδή η PHP είναι ανοικτή πηγή, υπάρχει μια μεγάλη κοινότητα της PHP προγραμματιστές που βοηθούν ο ένας τον άλλον με κωδικό. Αυτό σημαίνει PHP προγραμματιστές μπορούν να βασίζονται ο ένας στον άλλο, χρησιμοποιώντας επαναχρησιμοποιήσιμα κομμάτια κώδικα που ονομάζονται συναρτήσεις και τάξεις αντί να προσπαθούν να ανακαλύψουν ξανά τον τροχό. Αυτό μπορεί να περιορίσει δραματικά την ώρα παραγωγής.

PHP βασίζεται σε C + + γλώσσα και τη σύνταξη που χρησιμοποιείται στην PHP είναι αρκετά παρόμοια με C / C + +. C / C + + εξακολουθεί να θεωρείται η καλύτερη γλώσσα προγραμματισμού από πολλούς προγραμματιστές και τους ανθρώπους που αγαπούν αυτή τη γλώσσα θα ήταν σίγουρα αισθάνονται πιο άνετα με τη σύνταξη της PHP.

PHP και MySQL είναι άριστη επιλογή για webmasters που αναζητούν να αυτοματοποιήσουν τις ιστοσελίδες τους. Τώρα αράχνες αναζήτησης «βλέπει» όλο το περιεχόμενο σε μια σελίδα PHP, με τον ίδιο τρόπο που προβάλλεται σε ένα πρόγραμμα περιήγησης. Η δημιουργία ενός php-καλάθι αγορών είναι εκπληκτικά απλή και, όταν γίνει με ακρίβεια θα μπορούσε να μεταφραστεί σε μια εξαιρετικά αποτελεσματική και παγκοσμίως αποδεκτό καλάθι php-shopping.

2.4 Javascript

Η JavaScript είναι μια γλώσσα σε μια μορφή script που χρησιμοποιείται για να επιτρέψει την πρόσβαση μέσω προγραμματισμού σε αντικείμενα τόσο στην εφαρμογή-πελάτη όσο και άλλες εφαρμογές. Χρησιμοποιείται κυρίως με τη μορφή client-side JavaScript, ενσωματωμένη ως ένα ολοκληρωμένο στοιχείο του web browser, επιτρέποντας έτσι την ανάπτυξη ενισχυμένων διεπαφών χρήστη και δυναμικών ιστοσελίδων. Η JavaScript είναι μια διάλεκτος του προτύπου ECMAScript 10 και χαρακτηρίζεται ως δυναμική και ασθενώς δακτυλογραφημένη. Η JavaScript επηρεάστηκε από πολλές γλώσσες και έχει σχεδιαστεί να μοιάζει με τη Java, αλλά να είναι ευκολότερο για τους μη προγραμματιστές να εργαστούν με αυτή.

Χαρακτηριστικά της JavaScript

Τα ακόλουθα χαρακτηριστικά είναι κοινά σε όλες τις εφαρμογές σύμφωνα με το ECMAScript,εκτός και αν ορίζεται διαφορετικά.

- **Αναγκαστική και δομημένη**:
  Η JavaScript υποστηρίζει όλη τη δομημένη σύνταξη προγραμματισμού σε C (π.χ., δηλώσεις if, while βρόχοι, δηλώσεις switch, κλπ.), με μία μερική εξαίρεση την οριοθέτηση των πεδίων: πχ. το C-block style-scoping επίπεδο δεν υποστηρίζεται (αντ 'αυτού, η JavaScript έχει ένα functional-scoping επίπεδο). Η JavaScript 1.7, ωστόσο, υποστηρίζει το block-επίπεδο scoping χρησιμοποιώντας την λέξη let. Όπως και η C, έτσι και η JavaScript κάνει διάκριση μεταξύ των εκφράσεων(expressions) και των δηλώσεων(statements).

- **Δυναμική**:
  - Δυναμική δακτυλογράφηση: Όπως στις περισσότερες γλώσσες προγραμματισμού, οι τύποι σχετίζονται με τις αξίες, και όχι με τις μεταβλητές αυτές καθ’ αυτές. Για παράδειγμα, μια μεταβλητή x θα μπορούσε να δεσμεύεται σε έναν αριθμό, και αργότερα σε μια λέξη. Η JavaScript υποστηρίζει διάφορους τρόπους για τη δοκιμή του τύπου ενός αντικειμένου.
  - Βασισμένη στο αντικείμενο: Η JavaScript είναι σχεδόν εξ ολοκλήρου βασισμένη στο αντικείμενο. Τα αντικείμενα της JavaScript είναι συστοιχίες, επαυξημένες με τα πρωτότυπα. Τα ονόματα των ιδιοτήτων των αντικειμένων είναι λέξεις κλειδιά : Για παράδειγμα το obj.x = 10 και obj ["x"] = 10 είναι ισοδύναμα, με την τελεία στη σύνταξη να χρησιμοποιείται για να διαβάζεται ή να εκφράζεται ευκολότερα το

αντικείμενο. Οι ιδιότητες και οι τιμές των αντικειμένων, μπορούν να προστεθούν, μεταβληθούν ή και να διαγραφούν κατά το χρόνο εκτέλεσης καθώς οι περισσότερες ιδιότητες ενός αντικειμένου (και εκείνων που αφορούν την πρωτότυπη αλυσίδα κληρονομίας) μπορούν να καταμετρηθούν με τη χρήση ενός βρόχου for. Η JavaScript έχει ένα μικρό αριθμό ενσωματωμένων αντικείμενων όπως το Function και το Date.

- Αξιολόγηση του χρόνου εκτέλεσης: Η JavaScript περιλαμβάνει τη λειτουργία eval που μπορεί να εκτελέσει τα statements που έχουν δοθεί σαν strings κατά το χρόνο εκτέλεσης.

- **Λειτουργίες:**
  - Πρώτης κλάσης λειτουργίες: Τα functions ανήκουν στην πρώτη κλάση και υπάγονται στη κατηγορία των αντικείμενων και ως εκ τούτου, έχουν ιδιότητες και μπορούν να χρησιμοποιηθούν όπως κάθε άλλο αντικείμενο.
  - Εσωτερικές λειτουργίες(functions) και closures: Τα εσωτερικά functions (functions που ορίζονται σε άλλα functions) δημιουργούνται κάθε φορά που γίνεται επίκληση στην εξωτερική function, και οι μεταβλητές των εξωτερικών functions εξακολουθούν να υφίστανται στο βαθμό που οι εσωτερικές functions εξακολουθούν να υπάρχουν, ακόμη και μετά την ολοκλήρωση της επίκλησης (π.χ. αν η εσωτερική function επιστράφηκε, εξακολουθεί να έχει πρόσβαση στις μεταβλητές της εξωτερικής function). Αυτός είναι ο μηχανισμός πίσω από τα closures εντός της JavaScript.

- **Βασισμένη στα πρωτότυπα:**
  - Πρωτότυπα: Η JavaScript χρησιμοποιεί πρωτότυπα αντί των κλάσεων κληρονομικότητας. Είναι δυνατό να προσομοιωθούν πολλά χαρακτηριστικά που χρησιμοποιούν ως βάση τις κλάσεις χρησιμοποιώντας τα πρωτότυπα στη JavaScript.
  - Λειτουργίες(functions) ως κατασκευαστές αντικειμένων: Οι λειτουργίες διπλασιάζονται σαν κατασκευαστές αντικειμένων μαζί με το τυπικό ρόλο τους. Προκαθορίζοντας μια κλήση συνάρτησης με τη λέξη new δημιουργείτε ένα νέο αντικείμενο και οι κλήσεις του λειτουργούν με την τοπική λέξη-κλειδί this δεσμεύοντας έτσι αυτό το αντικείμενο για την εν λόγω επίκληση. Η ιδιότητα prototype του κατασκευαστή καθορίζει το αντικείμενο που χρησιμοποιείται για το εσωτερικό πρωτότυπο ενός νέου αντικειμένου. Οι ενσωματωμένοι κατασκευαστές που υπάρχουν στην JavaScript, όπως για παράδειγμα το Array, έχουν

επίσης πρωτότυπα τα οποία μπορούν να τροποποιηθούν.

- Λειτουργίες ως μέθοδοι: Σε αντίθεση με πολλές αντικειμενοστραφείς γλώσσες, δεν υπάρχει διάκριση μεταξύ του ορισμού τής λειτουργίας και του ορισμού της μεθόδου. Αντίθετα, η διάκριση γίνεται κατά τη διάρκεια της λειτουργίας, έτσι μια function μπορεί να κληθεί σαν να ήταν μια μέθοδος. Όταν μια function καλείται σαν μέθοδος ενός αντικειμένου, η λειτουργία της τοπικής λέξης this είναι υποχρεωτική για την εν λόγω επίκληση στο αντικείμενο.

- **Διάφορα:**
  - Περιβάλλον run-time: Η JavaScript συνήθως βασίζεται στο χρόνο εκτέλεσης του περιβάλλοντος (π.χ. σε ένα web browser) για να παρέχει αντικείμενα και μεθόδους μέσω των οποίων τα οποία scripts μπορούν να αλληλεπιδράσουν με "τον έξω κόσμο". Στην πραγματικότητα, εναπόκειται από το περιβάλλον, έτσι ώστε να παρέχετε η δυνατότητα να συμπεριλάβει / scripts εισαγωγής (π.χ. HTML <script> στοιχεία).
  - Λειτουργίες variadic: Ένας αόριστος αριθμός παραμέτρων μπορεί να περάσει σε μια λειτουργία. Η λειτουργία αυτή μπορεί να έχει πρόσβαση σε αυτά μέσω των επίσημων παραμέτρων και των αντικειμένων τοπικών επιχειρημάτων.
  - Πίνακες και αντικείμενα literals: Όπως πολλές γλώσσες προγραμματισμού, έτσι και εδώ ,οι πίνακες και τα αντικείμενα (associative arrays σε άλλες γλώσσες), μπορούν να δημιουργηθούν το καθένα με μια συνοπτική συντόμευση της σύνταξης. Στην πραγματικότητα, αυτές οι literals αποτελούν τη βάση για τη μορφή των δεδομένων JSON.
  - Σύνηθες εκφράσεις: Η JavaScript υποστηρίζει επίσης, σύνηθες εκφράσεις με παρόμοιο τρόπο με τη Perl, οι οποίες παρέχουν μια πιο συνοπτική και ισχυρή σύνταξη για τη χειραγώγηση του κειμένου που είναι πιο εξελιγμένη από ό, τι οι ενσωματωμένες συναρτήσεις συμβολοσειράς.

Χρήση της JavaScript στις ιστοσελίδες

Η κύρια χρήση της JavaScript είναι να γράφει λειτουργίες που είναι ενσωματωμένες ή περιλαμβάνονται στις HTML σελίδες και αλληλεπιδρούν με το Document Object Model (DOM) της σελίδας. Μερικά απλά παραδείγματα αυτής της χρήσης είναι:

- Το άνοιγμα ενός νέου παραθύρου με προγραμματιστικό έλεγχο του μεγέθους, της θέσης και των χαρακτηριστικών του νέου παραθύρου (δηλαδή εάν το μενού, η γραμμών εργαλείων, κλπ. θα είναι ορατά).

- Επικύρωση των τιμών μιας διαδικτυακής φόρμας για τη βεβαίωση ότι οι τιμές αυτές θα γίνουν δεκτές προτού αυτές υποβληθούν στο διακομιστή.

- Αλλαγή εικόνων καθώς ο κέρσορας του ποντικιού κινείται από πάνω τους: Η τεχνική αυτή χρησιμοποιείται συχνά για να επιστήσει την προσοχή του χρήστη σε σημαντικές συνδέσεις οι οποίες εμφανίζονται ως γραφικά στοιχεία.

Επειδή ο κώδικας της JavaScript μπορεί να λειτουργεί τοπικά σε ένα πρόγραμμα περιήγησης του χρήστη (και όχι σε ένα απομακρυσμένο server) μπορεί να ανταποκρίνεται άμεσα στις ενέργειες χρηστών, κάνοντας τις εφαρμογές πιο διαδραστικές. Επιπλέον, ο κώδικας της JavaScript μπορεί να ανιχνεύσει ενέργειες χρηστών που η HTML δεν μπορεί να ανιχνεύσει από μόνη της, όπως παραδείγματος χάριν ατομικές πληκτρολογήσεις. Εφαρμογές, όπως το Gmail επωφελούνται από αυτό αφού ένα μεγάλο μέρος της λογικής της διεπαφής του χρήστη είναι γραμμένο σε JavaScript, και η JavaScript αποστέλλει αιτήσεις για παροχή πληροφοριών (όπως το περιεχόμενο ενός μηνύματος ηλεκτρονικού ταχυδρομείου) στο διακομιστή. Η ευρύτερη τάση του προγραμματισμού Ajax εκμεταλλεύεται με τον ίδιο τρόπο αυτή την δύναμη.

Ένας μηχανισμός της JavaScript (επίσης γνωστός και ως διερμηνέας της JavaScript ή JavaScript implementation) είναι ένας διερμηνέας που μεταφράζει τον JavaScript κώδικα και τον εκτελεί αναλόγως. Η πρώτη μηχανή JavaScript δημιουργήθηκε από Brendan Eich στο Netscape Communications Corporation, για την εφαρμογή περιήγησης Netscape Navigator web.

Ένα πρόγραμμα περιήγησης στο διαδίκυτο είναι μακράν το πιο κοινό περιβάλλον υποδοχής για τη JavaScript. Τα προγράμματα περιήγησης στο

διαδίκτυο χρησιμοποιούν συνήθως API για τη δημιουργία "αντικειμένων υποδοχής" τα οποία είναι υπεύθυνα για την αντανάκλαση του DOM σε JavaScript. Ο web server είναι μια άλλη κοινή εφαρμογή του μηχανισμού της JavaScript.

Ένας JavaScript web Server εκθέτει τα αντικείμενα εκείνα που αντιπροσωπεύουν μια αίτηση HTTP καθώς και αντικείμενα απάντησης, τα οποία ένα πρόγραμμα JavaScript στη συνέχεια χειραγωγεί, δημιουργώντας μια δυναμική ιστοσελίδα.

Ένα παράδειγμα μιας ιστοσελίδας που περιέχει JavaScript είναι το παρακάτω:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">

<html>
    <head>
        <title>simple page</title>
    </head>
    <body>
        <script type="text/javascript">
            document.write('Hello World!');
        </script>
        <noscript>
        <p>Your browser either does not support JavaScript, or you have JavaScript turned off.</p>
        </noscript>
    </body>
</html>
```

2.5 Βάσεις Δεδομέων

Εισαγωγή στις Βάσεις Δεδομένων

Η αλματώδης ανάπτυξη της επιστήμης της πληροφορικής και των επικοινωνιών τα τελευταία χρόνια έχει καταστήσει την πληροφορία ως ένα από τα βασικότερα και πολυτιμότερα αγαθά. Είναι κοινός τόπος σήμερα η εκτίμηση ότι το αγαθό της πληροφορίας είναι επιθυμητό απ' όλους τους εργαζόμενους αλλά και τους εκπαιδευόμενους, ώστε να είναι πιο αποδοτικοί, ανταγωνιστικοί αλλά και παραγωγικοί στην εργασία τους.

Τα συστήματα βάσεων δεδομένων τα χρησιμοποιούμε για να μπορούμε να αποθηκεύσουμε, να επεξεργαστούμε αλλά και να εκμεταλλευτούμε αποδοτικά αυτόν τον τεράστιο όγκο των πληροφοριών που αυξάνονται με αλματώδεις ρυθμούς καθημερινά.

Τα Δεδομένα και οι Πληροφορίες

Με τον όρο *πληροφορία* αναφερόμαστε συνήθως σε ειδήσεις, γεγονότα και έννοιες που αποκτάμε από την καθημερινή μας επικοινωνία και τα θεωρούμε ως αποκτηθείσα γνώση, ενώ τα δεδομένα μπορούν να είναι μη κατάλληλα επεξεργασμένα και μη ταξινομημένα σύνολα πληροφοριών. Ένας αυστηρός ορισμός για το τι είναι δεδομένα και τι είναι πληροφορία, σύμφωνα με την επιτροπή ANSI των ΗΠΑ, είναι ο εξής :

- *Δεδομένα (data)* είναι μια παράσταση, όπως γράμματα, αριθμοί, σύμβολα κ.ά. στα οποία μπορούμε να δώσουμε κάποια σημασία (έννοια).
- *Πληροφορία (information)* είναι η σημασία που δίνουμε σ' ένα σύνολο από δεδομένα, τα οποία μπορούμε να επεξεργαστούμε βάσει προκαθορισμένων κανόνων και να βγάλουμε έτσι κάποια χρήσιμα συμπεράσματα. Με τις πληροφορίες περιορίζεται η αβεβαιότητα που έχουμε για διάφορα πράγματα και βοηθιόμαστε έτσι στο να λάβουμε σωστές αποφάσεις.

Τα δεδομένα μπορούν να θεωρηθούν ως τρόποι αναπαράστασης εννοιών και γεγονότων που μπορούν να υποστούν διαχείριση και επεξεργασία. Η συλλογή και αποθήκευση ενός τεράστιου όγκου δεδομένων όπως απαιτούν οι κοινωνικές συνθήκες σήμερα, δεν λύνει τελείως το πρόβλημα της σωστής οργάνωσης και ταξινόμησης των δεδομένων. Τα δεδομένα θα πρέπει να οργανωθούν με τέτοιο τρόπο έτσι ώστε να μπορούμε να τα εντοπίζουμε και να τα αξιοποιούμε εύκολα και γρήγορα και τη στιγμή που τα χρειαζόμαστε.

Ένα κλασικό παράδειγμα μη σωστής οργάνωσης δεδομένων θα ήταν για παράδειγμα ο τηλεφωνικός κατάλογος της πόλης της Θεσσαλονίκης, όπου οι συνδρομητές δεν θα ήταν καταχωρημένοι αλφαβητικά σύμφωνα με το επώνυμο και το όνομά τους, αλλά εντελώς τυχαία. Ένας τέτοιος τηλεφωνικός κατάλογος θα περιείχε μια τεράστια ποσότητα δεδομένων αλλά θα ήταν ουσιαστικά άχρηστος.

Ένα άλλο παράδειγμα μη σωστής οργάνωσης δεδομένων θα ήταν μια πολύ μεγάλη βιβλιοθήκη με χιλιάδες τόμους βιβλίων και χωρίς να διαθέτει κάποιο υποτυπώδες σύστημα οργάνωσης και ταξινόμησης των βιβλίων. Ούτε οι υπάλληλοι της βιβλιοθήκης θα μπορούσαν να κάνουν τη δουλειά τους αλλά ούτε και οι επισκέπτες θα μπορούσαν να αξιοποιήσουν την πληθώρα των πληροφοριών που περιέχονται στα βιβλία. Εκτός λοιπόν από τη μόνιμη αποθήκευση των δεδομένων, χρειαζόμαστε και κάποιους τρόπους ευέλικτης και αποδοτικής οργάνωσής τους.

Χαρακτηριστικά παραδείγματα δεδομένων που απαιτούν σωστή και αποδοτική οργάνωση είναι τα εξής :

- Τα στοιχεία υπαλλήλων, πελατών, προμηθευτών και παραγγελιών μιας εμπορικής επιχείρησης.
- Τα στοιχεία υλικών μιας αποθήκης.
- Τα στοιχεία ταινιών, πελατών και δανεισμών μιας βιντεολέσχης.
- Τα στοιχεία υπαλλήλων, γιατρών, ασθενών αλλά και υλικών ενός νοσοκομείου.
- Τα στοιχεία βιβλίων, χρηστών (δανειστών) και δανεισμών μιας βιβλιοθήκης.

Η Οργάνωση Αρχείων

Ο πιο γνωστός τρόπος οργάνωσης δεδομένων με τη χρήση ηλεκτρονικών υπολογιστών είναι σε αρχεία εγγραφών. Για να κατανοήσουμε καλύτερα ορισμένες έννοιες, θα εξετάσουμε την περίπτωση ενός αρχείου πελατών και παραγγελιών μιας εμπορικής επιχείρησης. Για να οργανώσουμε σωστά το αρχείο μας, θα πρέπει να δημιουργήσουμε καρτέλες για τους πελάτες, αλλά και για τις παραγγελίες τους αργότερα, που θα πρέπει να περιέχουν τα εξής στοιχεία ανά πελάτη :

- Κωδικός
- Επώνυμο
- Όνομα
- Διεύθυνση
- ΤΚ
- Πόλη

- Τηλέφωνο
- ΑΦΜ
- ΔΟΥ

Η αντιστοίχηση του παλιού τρόπου οργάνωσης με τις καρτέλες σε σχέση με τον σύγχρονο ηλεκτρονικό τρόπο οργάνωσης, έχει ως εξής :

- Συρτάρι – Αρχείο Δεδομένων
- Καρτέλα πελάτη – Εγγραφή του αρχείου δεδομένων
- Στοιχείο της καρτέλας – Πεδίο της εγγραφής

Ένα *αρχείο (file)* θα μπορούμε να το χαρακτηρίσουμε σαν ένα σύνολο που αποτελείται από οργανωμένα ομοειδή στοιχεία. Τα στοιχεία ενός αρχείου μπορούμε να τα οργανώσουμε σε λογικές ενότητες και το σύνολο των στοιχείων που περιέχει μια λογική ενότητα καλείται *εγγραφή (record)*. Το κάθε στοιχείο της εγγραφής καλείται *πεδίο (field)*. Το πεδίο αποτελεί και τη μικρότερη δυνατή υποδιαίρεση των στοιχείων ενός αρχείου. Ένα πεδίο χαρακτηρίζεται από τον μέγιστο αριθμό των χαρακτήρων (bytes) που απαιτούνται για την καταχώρησή του στη μνήμη του υπολογιστή και που αποκαλείται *μήκος του πεδίου (field length)*.

Σε μια οργάνωση αρχείου όπως είναι οι πελάτες μιας εμπορικής επιχείρησης που είδαμε νωρίτερα, τα αντίστοιχα πεδία όλων των εγγραφών καταλαμβάνουν τον ίδιο αριθμό σεbytes που είναι αυτός που έχουμε ορίσει κατά τη δημιουργία του αρχείου. Για παράδειγμα, αν ορίσαμε ότι το πεδίο Επώνυμο θα έχει μήκος 15 χαρακτήρες, τότε το πεδίο της εγγραφής του πελάτη με επώνυμο Παπαδόπουλος, αλλά και το πεδίο της εγγραφής του πελάτη με επώνυμο Βες θα καταλαμβάνουν από 15 bytes στη μνήμη του υπολογιστή, ενώ αν ένας πελάτης ονομάζεται Παπαχριστοδουλόπουλος, τότε θα γίνει αποκοπή του επωνύμου του και θα καταχωρηθούν στη μνήμη του υπολογιστή μόνο τα 15 πρώτα γράμματα, δηλ. τα Παπαχριστοδουλό.

Ένα πεδίο χαρακτηρίζεται ακόμη και από το είδος των δεδομένων που μπορεί να περιέχει, όπως :

- *Αλφαριθμητικό (alphanumeric)*, μπορεί να περιέχει γράμματα, ψηφία ή και ειδικούς χαρακτήρες.
- *Αριθμητικό (numeric)*, μπορεί να περιέχει μόνο αριθμούς.
- *Αλφαβητικό (alphabetic)*, μπορεί να περιέχει μόνο γράμματα (αλφαβητικούς χαρακτήρες).
- *Ημερομηνίας (date)*, μπορεί να περιέχει μόνο ημερομηνίες.
- *Δυαδικό (binary)*, μπορεί να περιέχει ειδικού τύπου δεδομένα, όπως εικόνες, ήχους κ.ά.
- *Λογικό (logical)*, μπορεί να περιέχει μόνο μία από δύο τιμές, οι οποίες αντιστοιχούν σε δύο διακριτές καταστάσεις και μπορούν

να χαρακτηρισθούν σαν 0 και 1 ή σαν αληθές (true) και ψευδές (false).
- *Σημειώσεων (memo),* μπορεί να περιέχει κείμενο με μεταβλητό μήκος, το οποίο μπορεί να είναι και αρκετά μεγάλο και συνήθως αποθηκεύεται σαν ξεχωριστό αρχείο από το κύριο αρχείο.

Όσον αφορά τις εγγραφές, χρήσιμοι ορισμοί είναι οι εξής :

- *Μήκος εγγραφής (record length)* καλείται το άθροισμα που προκύπτει από τα μήκη των πεδίων που την αποτελούν.
- *Δομή εγγραφής (record layout)* ή *γραμμογράφηση* καλείται ο τρόπος με τον οποίο οργανώνουμε τα πεδία μιας εγγραφής.
- *Διάβασμα (read)* από αρχείο σημαίνει τη μεταφορά των δεδομένων του αρχείου, που γίνεται συνήθως ανά μία εγγραφή, από το μέσο αποθήκευσης (σκληρό δίσκο ή δισκέτα) στην κεντρική μνήμη του υπολογιστή για επεξεργασία.
- *Γράψιμο (write)* σε αρχείο σημαίνει μεταφορά των δεδομένων του αρχείου, που γίνεται συνήθως ανά μία εγγραφή, από την κεντρική μνήμη του υπολογιστή στο μέσο αποθήκευσης (σκληρό δίσκο ή δισκέτα).

Προβλήματα της Οργάνωσης Αρχείων

Στα αρχικά στάδια της οργάνωσης αρχείων, ήταν πολύ συνηθισμένη πρακτική η δημιουργία ξεχωριστών εφαρμογών (προγραμμάτων) και ξεχωριστών αρχείων, όπως για παράδειγμα η δημιουργία ενός αρχείου πελατών και ενός άλλου ανεξάρτητου αρχείου για τις παραγγελίες των πελατών. Τα προβλήματα που προέκυψαν από την πρακτική αυτή είναι τα εξής :

- *Πλεονασμός των δεδομένων (data redundancy).* Υπάρχει η περίπτωση να έχουμε επανάληψη των ίδιων δεδομένων σε αρχεία διαφορετικών εφαρμογών. Για παράδειγμα, αν έχουμε ένα αρχείο πελατών και ένα αρχείο παραγγελιών αυτών των πελατών, είναι σχεδόν σίγουρο ότι θα υπάρχουν κάποια στοιχεία των πελατών που θα υπάρχουν και στα δύο αρχεία.
- *Ασυνέπεια των δεδομένων (data inconsistency).* Αυτό μπορεί να συμβεί όταν υπάρχουν τα ίδια στοιχεία των πελατών (πλεονασμός) και στο αρχείο πελατών και στο αρχείο παραγγελιών και χρειασθεί να γίνει κάποια αλλαγή στη διεύθυνση ή στα τηλέφωνα κάποιου πελάτη, οπότε είναι πολύ πιθανό να γίνει η διόρθωση μόνο στο ένα αρχείο και όχι και στο άλλο.

- *Αδυναμία μερισμού δεδομένων (data sharing).* Μερισμός δεδομένων σημαίνει δυνατότητα για κοινή χρήση των στοιχείων κάποιων αρχείων. Για παράδειγμα, ο μερισμός δεδομένων θα ήταν χρήσιμος αν με την παραγγελία ενός πελάτη μπορούμε να έχουμε πρόσβαση την ίδια στιγμή στο αρχείο πελατών για να δούμε το υπόλοιπο του πελάτη και μετά στο αρχείο της αποθήκης για να δούμε αν είναι διαθέσιμα τα προϊόντα που παρήγγειλε ο συγκεκριμένος πελάτης. Η αδυναμία μερισμού δεδομένων δημιουργεί καθυστέρηση στη λήψη αποφάσεων και στην εξυπηρέτηση των χρηστών.
- *Αδυναμία προτυποποίησης*. Έχει να κάνει με την ανομοιομορφία και με την διαφορετική αναπαράσταση και οργάνωση των δεδομένων στα αρχεία των εφαρμογών. Η αδυναμία αυτή δημιουργεί προβλήματα προσαρμογής των χρηστών καθώς και προβλήματα στην ανταλλαγή δεδομένων μεταξύ διαφορετικών συστημάτων.

Οι Βάσεις Δεδομένων και τα ΣΔΒΔ (DBMS)

Για να δοθεί μια λύση σ' όλα τα παραπάνω προβλήματα, και με βάση το γεγονός ότι η χρήση των ηλεκτρονικών υπολογιστών και συνεπώς η ηλεκτρονική καταχώρηση και επεξεργασία δεδομένων αυξήθηκε κατακόρυφα ήδη από τη δεκαετία του '70 στις μεγάλες επιχειρήσεις και άρα είχαμε πάρα πολλές εφαρμογές να επεξεργάζονται δεδομένα σε πάρα πολλά αρχεία ταυτόχρονα, προτάθηκε η συνένωση όλων των αρχείων μιας εφαρμογής. Εκτός, όμως, από τη συνένωση των αρχείων, απαιτείτο και μια σωστή οργάνωσή τους. Δημιουργήθηκαν έτσι οι Τράπεζες Πληροφοριών ή Βάσεις Δεδομένων (Data Bases).

Μια *Βάση Δεδομένων (ΒΔ)* είναι ένα σύνολο αρχείων με υψηλό βαθμό οργάνωσης τα οποία είναι συνδεδεμένα μεταξύ τους με λογικές σχέσεις, έτσι ώστε να μπορούν να χρησιμοποιούνται από πολλές εφαρμογές και από πολλούς χρήστες ταυτόχρονα. Υπάρχει ένα ειδικό λογισμικό το οποίο μεσολαβεί ανάμεσα στις αρχεία δεδομένων και τις εφαρμογές που χρησιμοποιούν οι χρήστες και αποκαλείται *Σύστημα Διαχείρισης Βάσης Δεδομένων (ΣΔΒΔ)* ή *DBMS (Data Base Management System).* Το ΣΔΒΔ είναι στην ουσία ένα σύνολο από προγράμματα και υπορουτίνες που έχουν να κάνουν με τον χειρισμό της βάσης δεδομένων, όσον αφορά τη δημιουργία, τροποποίηση, διαγραφή στοιχείων, με ελέγχους ασφαλείας κ.ά.

Οι χρήστες των εφαρμογών αντλούν τα στοιχεία που τους ενδιαφέρουν από τη βάση δεδομένων χωρίς να είναι σε θέση να γνωρίζουν με ποιο τρόπο είναι οργανωμένα τα δεδομένα σ' αυτήν. Το ΣΔΒΔ παίζει τον ρόλο του μεσάζοντα ανάμεσα στον χρήστη και τη βάση δεδομένων και μόνο μέσω του ΣΔΒΔ μπορεί ο χρήστης να αντλήσει πληροφορίες από τη βάση δεδομένων. Ένα ΣΔΒΔ μπορεί να είναι εγκατεστημένο σ' έναν μόνο υπολογιστή ή και σ' ένα δίκτυο υπολογιστών και μπορεί να χρησιμοποιείται από έναν χρήστη ή και από πολλούς χρήστες.

Ένα *Σύστημα Βάσης Δεδομένων (ΣΒΔ)* ή *DBS (Data Base System)* αποτελείται από το υλικό, το λογισμικό, τη βάση δεδομένων και τους χρήστες. Είναι δηλαδή ένα σύστημα με το οποίο μπορούμε να αποθηκεύσουμε και να αξιοποιήσουμε δεδομένα με τη βοήθεια ηλεκτρονικού υπολογιστή. Αναλυτικά :

- Το *υλικό (hardware)* αποτελείται όπως είναι γνωστό από τους ηλεκτρονικούς υπολογιστές, τα περιφερειακά, τους σκληρούς δίσκους, τις μαγνητικές ταινίες κ.ά., όπου είναι αποθηκευμένα τα αρχεία της βάσης δεδομένων αλλά και τα προγράμματα που χρησιμοποιούνται για την επεξεργασία τους.
- Το *λογισμικό (software)* είναι τα προγράμματα που χρησιμοποιούνται για την επεξεργασία των δεδομένων (στοιχείων) της βάσης δεδομένων.
- Η *βάση δεδομένων (data base)* αποτελείται από το σύνολο των αρχείων όπου είναι αποθηκευμένα τα δεδομένα του συστήματος. Τα στοιχεία αυτά μπορεί να βρίσκονται αποθηκευμένα σ' έναν φυσικό υπολογιστή αλλά και σε περισσότερους. Όμως, στον χρήστη δίνεται η εντύπωση ότι βρίσκονται συγκεντρωμένα στον ίδιο υπολογιστή. Τα δεδομένα των αρχείων αυτών είναι *ενοποιημένα (data integration)*, δηλ. δεν υπάρχει πλεονασμός (άσκοπη επανάληψη) δεδομένων και *μερισμένα (data sharing)*, δηλ. υπάρχει δυνατότητα ταυτόχρονης προσπέλασης των δεδομένων από πολλούς χρήστες. Ο κάθε χρήστης έχει διαφορετικά δικαιώματα και βλέπει διαφορετικό κομμάτι της βάσης δεδομένων, ανάλογα με τον σκοπό για τον οποίο συνδέεται.
- Οι *χρήστες (users)* μιας βάσης δεδομένων χωρίζονται στις εξής κατηγορίες :
    - *Τελικοί χρήστες (end users)*. Χρησιμοποιούν κάποια εφαρμογή για να παίρνουν στοιχεία από μια βάση δεδομένων, έχουν τις λιγότερες δυνατότητες επέμβασης στα στοιχεία της βάσης δεδομένων, χρησιμοποιούν ειδικούς κωδικούς πρόσβασης και το σύστημα τούς

επιτρέπει ανάλογα πρόσβαση σε συγκεκριμένο κομμάτι της βάσης δεδομένων.

- *Προγραμματιστές εφαρμογών (application programmers).* Αναπτύσσουν τις εφαρμογές του ΣΒΔ σε κάποια από τις γνωστές γλώσσες προγραμματισμού.
- *Διαχειριστής δεδομένων (data administrator – DA).* Έχει τη διοικητική αρμοδιότητα και ευθύνη για την οργάνωση της βάσης δεδομένων και την απόδοση δικαιωμάτων πρόσβασης στους χρήστες.
- *Διαχειριστής βάσης δεδομένων (database administrator – DBA).* Λαμβάνει οδηγίες από τον διαχειριστή δεδομένων και είναι αυτός που διαθέτει τις τεχνικές γνώσεις και αρμοδιότητες για τη σωστή και αποδοτική λειτουργία του ΣΔΒΔ.

Η Αρχιτεκτονική των ΣΔΒΔ

Όπως είδαμε νωρίτερα, ένα ΣΔΒΔ (Σύστημα Διαχείρισης Βάσης Δεδομένων) έχει σαν αποστολή τη διαχείριση των δεδομένων των αρχείων της βάσης, δηλ. την προσθήκη, διαγραφή, τροποποίηση εγγραφών, την αναζήτηση μέσα στις εγγραφές κ.ά.). Το ΣΔΒΔ δέχεται αιτήσεις από τους χρήστες των εφαρμογών και επικοινωνεί με τα αρχεία της βάσης δεδομένων για να τις διεκπεραιώσει.

Αυτή η κοινή διεπαφή (interface) των εφαρμογών με τα αρχεία αποκαλείται *λογική διεπαφή*. Οι εφαρμογές που δημιουργούμε δεν απασχολούνται με τον τρόπο που είναι αποθηκευμένα τα δεδομένα, πόσο χώρο καταλαμβάνουν κοκ και αυτή η ιδιότητα είναι γνωστή ως *ανεξαρτησία δεδομένων*.

Αυτό σημαίνει πρακτικά ότι οποιαδήποτε αλλαγή στον τρόπο οργάνωσης των αρχείων της βάσης δεδομένων δεν θα συνεπάγεται και αλλαγή στις εφαρμογές· ένα πρόβλημα που ταλαιπωρούσε πολύ τους προγραμματιστές παλαιοτέρων εποχών. Ακόμη, η προσθήκη, η κατάργηση ή και η τροποποίηση κάποιων εφαρμογών δεν θα έχει καμία επίπτωση στον τρόπο οργάνωσης των αρχείων της βάσης δεδομένων. Στα ΣΔΒΔ έχει επικρατήσει η λεγόμενη αρχιτεκτονική των τριών επιπέδων (βαθμίδων), όπου τα τρία επίπεδα είναι τα εξής :

- *Εσωτερικό επίπεδο (internal level),* έχει να κάνει με την αποθήκευση των αρχείων στον σκληρό δίσκο, δηλ. την πραγματική ή φυσική κατάστασή τους.
- *Εξωτερικό επίπεδο (external level),* έχει να κάνει με τους χρήστες είτε αυτοί είναι απλοί χειριστές, είτε προγραμματιστές ή και οι διαχειριστές της βάσης δεδομένων.

- *Εννοιολογικό επίπεδο (conceptual level)*, είναι ένα ενδιάμεσο επίπεδο που διασυνδέει τα δύο άλλα επίπεδα και έχει να κάνει με τη λογική σχεδίαση των αρχείων της βάσης δεδομένων.

Οι Οντότητες (Entities)

Με τον όρο *οντότητα (entity)* εννοούμε ένα αντικείμενο, ένα πρόσωπο, μια κατάσταση και γενικά ο,τιδήποτε μπορεί να προσδιορισθεί σαν ανεξάρτητη ύπαρξη (αυτόνομη μονάδα του φυσικού κόσμου). Για παράδειγμα, σε μια βάση δεδομένων μιας εμπορικής εταιρείας, οντότητες μπορεί να είναι οι εργαζόμενοι, οι πελάτες, οι προμηθευτές, οι παραγγελίες, τα είδη της αποθήκης (προϊόντα) κ.ά.

Το *Μοντέλο Οντοτήτων Συσχετίσεων (Entity Relationship Model, ER Model)* είναι μια διαγραμματική αναπαράσταση της δομής μιας βάσης δεδομένων και χρησιμοποιείται κατά τη φάση του λογικού σχεδιασμού της βάσης. Δηλαδή, δεν ασχολείται με τον τρόπο που αποθηκεύονται τα δεδομένα της βάσης, αλλά με την ταυτοποίηση των δεδομένων και με τον τρόπο με τον οποίο αυτά συσχετίζονται μεταξύ τους.

Θα δούμε ένα παράδειγμα μιας εταιρείας, η οποία περιέχει δεδομένα που αφορούν τους υπαλλήλους της (employees), τα τμήματά της (departments) και τα έργα (projects) που έχουν αναλάβει αυτά τα τμήματα. Ένα τμήμα της εταιρείας μπορεί να εποπτεύει ένα ή περισσότερα έργα (projects) και ένας υπάλληλος ανήκει σ' ένα μόνο τμήμα της εταιρείας αλλά μπορεί να απασχολείται ταυτόχρονα σε πολλά έργα, τα οποία δεν είναι υποχρεωτικό να παρακολουθούνται από το ίδιο τμήμα.

Οι Ιδιότητες (Attributes)

Με τον όρο *ιδιότητα* ή *χαρακτηριστικό* ή και *πεδίο (attribute)* μιας οντότητας, αναφερόμαστε σ' ένα από τα συστατικά της στοιχεία που την περιγράφουν και την κάνουν να ξεχωρίζει από τα άλλα στοιχεία της ίδιας οντότητας. Για παράδειγμα, η οντότητα ΠΕΛΑΤΗΣ μπορεί να έχει ως ιδιότητες (χαρακτηριστικά) τον κωδικό, το επώνυμο, το όνομα, τη διεύθυνση, το τηλέφωνο, το ΑΦΜ κ.ά., με τη βοήθεια των οποίων μπορούμε να ξεχωρίσουμε τους πελάτες μεταξύ τους.

Επίσης, η οντότητα ΠΑΡΑΓΓΕΛΙΑ μπορεί να έχει ως ιδιότητες (χαρακτηριστικά) τον κωδικό, τον αριθμό παραστατικού, την ημερομηνία, τον κωδικό πελάτη, το προϊόν κ.ά., με τη βοήθεια των οποίων μπορούμε να ξεχωρίσουμε τις παραγγελίες μεταξύ τους Στο παράδειγμα της εταιρείας, μπορούμε να ορίσουμε έναν τύπο οντότητας για τους υπαλλήλους της

εταιρείας (EMPLOYEE), έναν τύπο οντότητας για τα τμήματα που έχει η εταιρεία (DEPARTMENT) και έναν τύπο οντότητας για τα έργα που έχει αναλάβει η εταιρεία (PROJECT). Καθένας από τους παραπάνω τύπους οντοτήτων περιγράφεται από ένα όνομα και από το σύνολο των πεδίων που περιέχει. Οι πληροφορίες αυτές αποτελούν το *σχήμα (schema)* της οντότητας.

Τα Στιγμιότυπα (Snapshots)

Το κάθε διαφορετικό (αυτόνομο) στοιχείο μιας οντότητας αποκαλείται *στιγμιότυπο (snapshot)* ή και *εμφάνιση* της οντότητας. Για παράδειγμα, στην οντότητα ΠΕΛΑΤΗΣ, άλλο στιγμιότυπο είναι ο πελάτης με επώνυμο Παπαδόπουλος και άλλο στιγμιότυπο είναι ο πελάτης με επώνυμο Σουμπάσης.

Το Πρωτεύον Κλειδί (Primary Key)

Πρωτεύον κλειδί ή ***πεδίο κλειδί*** *(primary key)* μιας οντότητας καλείται εκείνη η ιδιότητα (ή ο συνδυασμός ιδιοτήτων) που έχει μοναδική τιμή για όλα τα στιγμιότυπα (εμφανίσεις) της οντότητας. Για παράδειγμα, στην οντότητα ΠΕΛΑΤΗΣ πρωτεύον κλειδί είναι ο κωδικός πελάτη, στην οντότητα ΠΑΡΑΓΓΕΛΙΑ πρωτεύον κλειδί μπορεί να είναι ο κωδικός παραγγελίας ή ο αριθμός παραστατικού κοκ.

Υπάρχουν περιπτώσεις όπου το πεδίο κλειδί ενός τύπου οντότητας μπορεί να μην είναι απλό αλλά σύνθετο, να αποτελείται δηλαδή από πολλά απλά πεδία και τότε η συνθήκη της μοναδικότητας για την τιμή του κλειδιού δεν εφαρμόζεται σε κάθε πεδίο του σύνθετου κλειδιού αλλά στο σύνολο του συνδυασμού αυτών των πεδίων.

Οι Συσχετίσεις (Relationships)

Με τον όρο *συσχέτιση (relationship)* αναφερόμαστε στον τρόπο σύνδεσης (επικοινωνίας) δύο ξεχωριστών οντοτήτων, ώστε να μπορούμε να αντλούμε στοιχεία (πληροφορίες) από τον συνδυασμό τους.

Για παράδειγμα, η οντότητα ΓΙΑΤΡΟΣ συσχετίζεται με την οντότητα ΑΣΘΕΝΗΣ αλλά και με την οντότητα ΚΛΙΝΙΚΗ στη βάση δεδομένων ενός νοσοκομείου. Μπορούμε να δεχθούμε ότι ένας γιατρός παρακολουθεί (συσχετίζεται με) πολλούς ασθενείς, αλλά ένας ασθενής παρακολουθείται από (συσχετίζεται με) έναν μόνο γιατρό και επίσης ένας γιατρός συσχετίζεται με

(ανήκει σε) μία μόνο κλινική, αλλά μια κλινική συσχετίζεται με (απασχολεί) πολλούς γιατρούς.

Στο παράδειγμα της εταιρείας, η οντότητα EMPLOYEE συσχετίζεται με την οντότητα DEPARTMENT και η οντότητα DEPARTMENT συσχετίζεται με την οντότητα PROJECTS. Ένας υπάλληλος ανήκει σ' ένα μόνο τμήμα και ένα τμήμα μπορεί να έχει πολλούς υπαλλήλους. Επίσης, ένα τμήμα εποπτεύει πολλά έργα αλλά ένα έργο εποπτεύεται από ένα μόνο τμήμα.

Το Ιεραρχικό Μοντέλο Βάσεων Δεδομένων

Υπάρχουν τρία βασικά μοντέλα που έχουν επικρατήσει στις βάσεις δεδομένων, το ιεραρχικό, το δικτυωτό και το σχεσιακό, και τα οποία αναπτύχθηκαν με βάση αντίστοιχες δομές. Το ιεραρχικό μοντέλο (hierarchical) έχει μια ιεραρχική δομή που θυμίζει δένδρο. Οι οντότητες μοιάζουν με απολήξεις από κλαδιά δένδρων και τοποθετούνται σε επίπεδα ιεραρχίας. Τα κλαδιά παριστάνουν τις συσχετίσεις ανάμεσα στις οντότητες.

Από μια οντότητα που βρίσκεται σ' ένα ανώτερο επίπεδο εκκινούν πολλά κλαδιά, καθένα από τα οποία καταλήγει σε μια οντότητα που βρίσκεται σ' ένα χαμηλότερο επίπεδο. Αλλά, σε κάθε οντότητα που βρίσκεται σ' ένα χαμηλότερο επίπεδο αντιστοιχεί μία και μόνο μία οντότητα που βρίσκεται σ' ένα ανώτερο επίπεδο. Το μοντέλο αυτό ήταν το πρώτο που εμφανίσθηκε αλλά σήμερα θεωρείται δύσχρηστο και ξεπερασμένο.

Το Δικτυωτό Μοντέλο Βάσεων Δεδομένων

Και στο δικτυωτό (network) μοντέλο, τα στοιχεία τοποθετούνται σ' ένα επίπεδο ιεραρχίας, αλλά κάθε στοιχείο μπορεί να συσχετισθεί με πολλά στοιχεία είτε σ' ένα κατώτερο ή σ' ένα ανώτερο επίπεδο.

Το Σχεσιακό Μοντέλο Βάσεων Δεδομένων

Το σχεσιακό (relational) μοντέλο έχει επικρατήσει σήμερα στην αναπαράσταση των δεδομένων καθώς διαθέτει σημαντικά πλεονεκτήματα ως προς τα άλλα δύο και οι βάσεις δεδομένων που σχεδιάζονται σύμφωνα μ' αυτό

αποκαλούνται σχεσιακές (relational databases). Με τις σχεσιακές βάσεις δεδομένων διαθέτουμε έναν σαφή, απλό και εύκολα κατανοητό τρόπο για να μπορέσουμε να αναπαραστήσουμε και να διαχειριστούμε τα δεδομένα μας. Υστερούν μόνο σε ταχύτητα υπολογισμών και σε χώρο αποθήκευσης, αλλά μόνο όταν έχουμε να κάνουμε με πολύ μεγάλες βάσεις δεδομένων.

Στο μοντέλο αυτό οι βάσεις δεδομένων περιγράφονται με αυστηρές μαθηματικές έννοιες και ο χρήστης βλέπει τις οντότητες και τις συσχετίσεις με τη μορφή πινάκων (tables) και σχέσεων (relations) αντίστοιχα.

Ένας *πίνακας (table)* αποτελείται από γραμμές (rows) και στήλες (columns), όπου τοποθετούμε τα στοιχεία σε οριζόντια και κάθετη μορφή. Η κάθε στήλη του πίνακα χαρακτηρίζει κάποια ιδιότητα της οντότητας και αποκαλείται *χαρακτηριστικό (attribute)* ή *πεδίο (field)*, ενώ η κάθε γραμμή του πίνακα περιέχει όλες τις πληροφορίες (στήλες) που αφορούν ένα στοιχείο της οντότητας και αποκαλείται *πλειάδα (tuple)* ή *εγγραφή (record)*.

Κάθε πεδίο του πίνακα μπορεί να πάρει ορισμένες μόνο τιμές, οι οποίες μπορεί να καθορίζονται από τον τύπο δεδομένων της ιδιότητας, όπως ονόματα ή αριθμοί για παράδειγμα, ή και από αυτό που εκφράζει, όπως το ότι δεν μπορούμε να έχουμε αρνητικό βάρος ή αρνητικό ΑΦΜ, για παράδειγμα. Το σύνολο των αποδεκτών τιμών μιας οντότητας αποκαλείται*πεδίο ορισμού (domain)*.

Για να μπορέσουμε να κατανοήσουμε τις σχεσιακές βάσεις δεδομένων, ένα πολύ χαρακτηριστικό παράδειγμα αποτελεί ένας πίνακας πελατών και ένας πίνακας παραγγελιών μιας εμπορικής εταιρείας.

Τα πεδία που μπορούμε να ορίσουμε στους πίνακες αυτούς είναι τα εξής :

ΠΙΝΑΚΑΣ (ΟΝΤΟΤΗΤΑ) ΠΕΛΑΤΕΣ

(ΚωδικόςΠελάτη, Επώνυμο, Όνομα, Διεύθυνση, ΤΚ, Πόλη, ΑΦΜ, Υπόλοιπο)

ΠΙΝΑΚΑΣ (ΟΝΤΟΤΗΤΑ) ΠΑΡΑΓΓΕΛΙΕΣ

(ΚωδικόςΠελάτη, ΚωδικόςΠαραγγελίας, Ημερομηνία, Είδος, Ποσότητα, ΤιμήΜονάδας)

Βλέπουμε ότι οι δύο πίνακες έχουν ένα κοινό πεδίο (στήλη), τον ΚωδικόςΠελάτη, και αυτό είναι απαραίτητο στις σχεσιακές βάσεις δεδομένων για να μπορέσουμε να κάνουμε τη δουλειά μας και να συνδυάσουμε πληροφορίες και από τους δύο πίνακες.

Όπως είναι εύκολα κατανοητό, η βασικότερη εργασία που έχουμε να κάνουμε κατά τον σχεδιασμό μιας σχεσιακής βάσης δεδομένων είναι να ορίσουμε τους πίνακες που θα χρησιμοποιήσουμε καθώς και τα πεδία που θα περιέχει ο καθένας απ' αυτούς. Η διαδικασία αυτή αποκαλείται κατασκευή του *σχήματος (schema)* μιας βάσης δεδομένων.

Οι κανόνες που πρέπει να ακολουθούμε πιστά κατά τον σχεδιασμό μιας σχεσιακής βάσης δεδομένων είναι οι εξής :

- Η κάθε οντότητα πρέπει να παριστάνεται ως ένας ξεχωριστός πίνακας.
- Η κάθε στήλη του πίνακα αντιστοιχεί σε μια ιδιότητα της οντότητας.
- Η κάθε γραμμή του πίνακα αντιστοιχεί σε μια εμφάνιση της οντότητας.
- Η κάθε γραμμή πρέπει να είναι μοναδική, δηλ. αποκλείεται να υπάρχουν δύο ή και περισσότερες γραμμές που να περιέχουν τα ίδια ακριβώς στοιχεία.
- Η σειρά εμφάνισης των γραμμών δεν έχει καμία σημασία.
- Η κάθε στήλη έχει μια δική της μοναδική ονομασία.
- Οι τιμές που ανήκουν στην ίδια στήλη πρέπει να είναι του ιδίου τύπου, δηλ. ή όλες αριθμοί ή όλες αλφαριθμητικές κοκ.
- Η στήλη που αποτελεί το πρωτεύον κλειδί (primary key) μιας οντότητας, δεν πρέπει να είναι ποτέ κενή (null).
- Αποκλείεται να υπάρχουν δύο ή και περισσότερες γραμμές που να περιέχουν την ίδια τιμή στο πρωτεύον κλειδί.
- Το πρωτεύον κλειδί μιας οντότητας αποκαλείται ξένο κλειδί (foreign key) σε μια άλλη οντότητα, με την οποία υπάρχει συσχετισμός.
- Μπορεί να υπάρχουν πολλές γραμμές που να έχουν την ίδια τιμή στο ξένο κλειδί.

Τα Σχεσιακά ΣΔΒΔ (RDBMS)

Τα Σχεσιακά Συστήματα Διαχείρισης Βάσεων Δεδομένων (ΣΣΔΒΔ) ή RBMS (Relational DataBase Management Systems) αναπτύχθηκαν με βάση το σχεσιακό μοντέλο και έχουν επικρατήσει πλήρως στον χώρο. Κατά τον σχεδιασμό και τη δημιουργία μιας σχεσιακής βάσης δεδομένων, οι πίνακες αποτελούν το μοναδικό δομικό και απαραίτητο στοιχείο για μπορέσουν να αναπαρασταθούν οι πληροφορίες που περιέχονται στη βάση δεδομένων.

Για να μπορέσουμε να προσθέσουμε, διαγράψουμε ή τροποποιήσουμε τα στοιχεία που περιέχονται σε μια βάση δεδομένων, χρησιμοποιούμε ειδικές γλώσσες προγραμματισμού που αποκαλούνται *γλώσσες ερωταπαντήσεων*

(query languages). Η γλώσσα που αποτελεί σήμερα ένα διεθνές πρότυπο για την επικοινωνία των χρηστών με τα Σχεσιακά ΣΔΒΔ είναι η*SQL (Structured Query Language)* ή *Δομημένη Γλώσσα Ερωτημάτων*. Μπορεί να λειτουργήσει αυτόνομα αλλά και σε συνεργασία μ' άλλες γλώσσες προγραμματισμού.

Μια άλλη, φιλική προς τον χρήστη γλώσσα προγραμματισμού για να μπορούμε να υποβάλουμε ερωτήματα σε σχεσιακές βάσεις δεδομένων και να λαμβάνουμε απαντήσεις είναι η*QBE (Query By Example)*, η οποία χρησιμοποιεί φόρμες για τη γραφική απεικόνιση των ερωτημάτων μας.

Σήμερα, υπάρχουν εξελιγμένα εργαλεία διαχείρισης σε γραφικό και φιλικό προς τον χρήστη περιβάλλον για να κάνουμε τα εξής :

- Δημιουργία πινάκων
- Δημιουργία φορμών
- Δημιουργία ερωτημάτων
- Δημιουργία εκθέσεων (αναφορών)

Τα Σχεσιακά ΣΔΒΔ τα διακρίνουμε στα *μεγάλα*, τα οποία αφορούν κυρίως μεγάλους οργανισμούς και επιχειρήσεις, έχουν τεράστιο όγκο δεδομένων και πολλούς χρήστες ταυτόχρονα, και τέτοια συστήματα είναι τα Oracle, Ingres, Informix, SQL Server κ.ά. και τα *μικρά*, τα οποία αφορούν κυρίως απλούς χρήστες, όπως είναι η Microsoft Access, ηParadox, η FoxPro κ.ά.

Το Μοντέλο Οντοτήτων–Συσχετίσεων

Το μοντέλο που έχει επικρατήσει σήμερα για να παραστήσει τις έννοιες ή τη δομή μιας βάσης δεδομένων είναι το *Μοντέλο Οντοτήτων–Συσχετίσεων (ΟΣ)*. Οι βασικές (θεμελιώδεις) έννοιες του μοντέλου αυτού είναι οι εξής :

- Οντότητες
- Ιδιότητες ή Χαρακτηριστικά
- Συσχετίσεις

Για να αναπαραστήσουμε ένα Μοντέλο Οντοτήτων – Συσχετίσεων χρησιμοποιούμε ειδικά διαγράμματα, όπου τα ορθογώνια συμβολίζουν τις οντότητες, οι ρόμβοι τις συσχετίσεις και οι ελλείψεις τις ιδιότητες. Με ευθείες γραμμές συνδέουμε τις οντότητες που συσχετίζονται με κάποιο τρόπο μεταξύ τους. Όλα τα παραπάνω αποτελούν τη λογική δομή μιας βάσης δεδομένων, μια εργασία που είναι απαραίτητο να γίνει πριν από την καταχώριση και την επεξεργασία των στοιχείων (πληροφοριών) της βάσης δεδομένων.

Το μοντέλο οντοτήτων–συσχετίσεων αποτελεί μια γενική περιγραφή των γενικών στοιχείων που απαρτίζουν μια βάση δεδομένων και απεικονίζει την αντίληψη που έχουμε για τα δεδομένα (εννοιολογικό), χωρίς να υπεισέρχεται σε λεπτομέρειες υλοποίησης.

Οι Οντότητες

Με τον όρο *οντότητα (entity)* αναφερόμαστε σε κάθε αντικείμενο, έννοια, πρόσωπο ή κατάσταση που έχει μια ανεξάρτητη ύπαρξη. Είναι κάτι που ξεχωρίζει και μπορούμε να συγκεντρώσουμε πληροφορίες (στοιχεία) γι' αυτό. Η οντότητα είναι αντίστοιχη με την έννοια της εγγραφής που συναντάμε στα αρχεία και στους πίνακες αλλά και με την έννοια του αντικειμένου στις σύγχρονες αντικειμενοστραφείς γλώσσες προγραμματισμού.

Παραδείγματα οντοτήτων είναι τα εξής : ΠΕΛΑΤΗΣ, ΠΑΡΑΓΓΕΛΙΑ, ΠΡΟΜΗΘΕΥΤΗΣ, ΑΠΟΘΗΚΗ, ΜΑΘΗΤΗΣ, ΚΑΘΗΓΗΤΗΣ, ΑΘΛΗΤΗΣ, ΑΓΩΝΙΣΜΑ, ΧΩΡΑ, ΠΟΛΕΙΣ κ.ά.

Μια βάση δεδομένων μπορεί να περιέχει πολλές διαφορετικές οντότητες, οι οποίες απεικονίζονται με ορθογώνια παραλληλόγραμμα και συσχετίζονται μεταξύ τους ανά δύο.

Οι Ιδιότητες (Χαρακτηριστικά) των Οντοτήτων

Με τον όρο *ιδιότητες (properties)* ή *χαρακτηριστικά (attributes)* αναφερόμαστε στα συστατικά (δομικά) στοιχεία που προσδιορίζουν (αποτελούν) μια οντότητα. Η ιδιότητα είναι αντίστοιχη με την έννοια του πεδίου που συναντάμε στα αρχεία και στους πίνακες αλλά και με την έννοια της μεταβλητής στις γλώσσες προγραμματισμού.

Για παράδειγμα, η οντότητα ΓΙΑΤΡΟΣ μπορεί να αποτελείται από τις ιδιότητες (χαρακτηριστικά) ΑριθμόςΜητρώου, Επώνυμο, Όνομα, Πατρώνυμο, Ειδικότητα, Βαθμός, ΈτοςΓέννησης, Διεύθυνση, ΑΦΜ, Τηλέφωνο, Κινητό κ.ά., ενώ η οντότητα ΑΘΛΗΤΗΣ μπορεί να αποτελείται από τις ιδιότητες (χαρακτηριστικά) ΚωδικόςΑθλητή, Επώνυμο, Όνομα, Πατρώνυμο, Αγώνισμα, Επίδοση, Σύλλογος, ΈτοςΓέννησης, Διεύθυνση, Τηλέφωνο, Κινητό κ.ά.

Απ' όλες τις ιδιότητες μιας οντότητας, υπάρχει μία μόνο ιδιότητα, και σπανιότερα ένας συνδυασμός δύο ή και περισσοτέρων ιδιοτήτων, η τιμή της οποίας είναι μοναδική και προσδιορίζει την κάθε εμφάνιση (στιγμιότυπο) της οντότητας και αποκαλείται ***πρωτεύον κλειδί*** *(primary key).* Το πρωτεύον

κλειδί εμφανίζεται στα διαγράμματα με υπογράμμιση ή με έντονη γραφή ή έχει ως πρόθεμα τον χαρακτήρα #.

Στο διάγραμμα οντοτήτων–συσχετίσεων οι ιδιότητες απεικονίζονται με σχήματα ελλειπτικής μορφής, τα οποία ενώνονται με ευθείες γραμμές με την οντότητα στην οποία ανήκουν.

Τα Κλειδιά

Όπως είδαμε και νωρίτερα, με τον όρο *κλειδί (key)* ή πιο σωστά *πρωτεύον κλειδί (primary key)* αναφερόμαστε σε μια ιδιότητα (πεδίο), ή σπανιότερα σ' ένα σύνολο ιδιοτήτων (πεδίων), η τιμή της οποίας είναι μοναδική σ' ολόκληρη την οντότητα (πίνακας). Στην πράξη, το πρωτεύον κλειδί έχει διαφορετική τιμή για κάθε εμφάνιση της οντότητας ή για κάθε γραμμή (εγγραφή) του πίνακα και ποτέ δεν μπορεί να έχει μηδενική (κενή) τιμή (null). Προσοχή, άλλο πράγμα είναι ο αριθμός 0 και άλλο πράγμα είναι η κενή τιμή (null), δηλ. η μη ύπαρξη τιμής.

Ο συνδυασμός δύο ή και περισσοτέρων ιδιοτήτων (πεδίων) για τη δημιουργία ενός πρωτεύοντος κλειδιού αποκαλείται *σύνθετο κλειδί*. Ένα παράδειγμα σύνθετου κλειδιού θα μπορούσε να είναι ο συνδυασμός των ιδιοτήτων Επώνυμο, Όνομα και Πατρώνυμο, εφόσον φυσικά είμαστε απολύτως βέβαιοι ότι δεν υπάρχουν δύο ή και περισσότερα άτομα με κοινές τιμές στις παραπάνω ιδιότητες.

Ξένο κλειδί αποκαλείται μια ιδιότητα (πεδίο) που είναι πρωτεύον κλειδί σε μια οντότητα (πίνακας) αλλά που υπάρχει και σε μια άλλη οντότητα (πίνακας) σαν απλή ιδιότητα. Τα ξένα κλειδιά είναι απαραίτητα για να μπορέσουμε να κάνουμε τις συσχετίσεις (συνδέσεις, επικοινωνίες) ανάμεσα στις οντότητες (πίνακες).

Για παράδειγμα, στην οντότητα ΣΥΛΛΟΓΟΣ, το πεδίο ΚωδικόςΣυλλόγου είναι πρωτεύον κλειδί, ενώ στην οντότητα ΑΘΛΗΤΗΣ, το πεδίο ΚωδικόςΣυλλόγου είναι ξένο κλειδί και είναι απαραίτητο για να μπορέσουμε να υλοποιήσουμε τη συσχέτιση ΑΝΗΚΕΙ, δηλ. να αντλήσουμε την πληροφορία ποιοι αθλητές ανήκουν σε ποιους συλλόγους. Προφανώς, στην οντότητα ΣΥΛΛΟΓΟΣ, το πεδίο Κωδικός Συλλόγου θα έχει μοναδικές τιμές, ενώ στην οντότητα ΑΘΛΗΤΗΣ, το πεδίο Κωδικός Συλλόγου θα έχει επαναλαμβανόμενες τιμές και αυτό γιατί πολλοί αθλητές θα ανήκουν στον ίδιο σύλλογο, αλλά ένας αθλητής ανήκει υποχρεωτικά σ' έναν και μόνο έναν σύλλογο.

Σ' ένα άλλο παράδειγμα, στην οντότητα ΓΙΑΤΡΟΣ, το πεδίο ΚωδικόςΓιατρού είναι πρωτεύον κλειδί, ενώ στην οντότητα ΑΣΘΕΝΗΣ, το πεδίο

ΚωδικόςΓιατρού είναι ξένο κλειδί και είναι απαραίτητο για να μπορέσουμε να υλοποιήσουμε τη συσχέτιση ΠΑΡΑΚΟΛΟΥΘΕΙΤΑΙ, δηλ. να αντλήσουμε την πληροφορία ποιοι ασθενείς παρακολουθούνται από ποιους γιατρούς. Προφανώς, στην οντότητα ΓΙΑΤΡΟΣ, το πεδίο ΚωδικόςΓιατρού θα έχει μοναδικές τιμές, ενώ στην οντότητα ΑΣΘΕΝΗΣ, το πεδίο ΚωδικόςΓιατρού θα έχει επαναλαμβανόμενες τιμές και αυτό γιατί πολλοί ασθενείς θα παρακολουθούνται από τον ίδιο γιατρό, αλλά ένας ασθενής παρακολουθείται μόνο από έναν γιατρό.

Αυτό αποτελεί βέβαια μια παραδοχή που κάνουμε για να μπορέσουμε να υλοποιήσουμε μια συσχέτιση σαν την παραπάνω σε μια βάση δεδομένων ενός Νοσοκομείου, αλλά μπορεί να θεωρήσει κάποιος ότι ένας ασθενής μπορεί να παρακολουθείται από πολλούς γιατρούς ταυτόχρονα, οπότε θα πρέπει να μεταβάλλουμε και τον τρόπο συσχέτισης των παραπάνω οντοτήτων.

Οι Συσχετίσεις Μεταξύ Οντοτήτων

Ο σωστός σχεδιασμός και προσδιορισμός των οντοτήτων και των ιδιοτήτων τους αποτελούν τα θεμελιώδη βήματα για τη σωστή σχεδίαση και υλοποίηση μιας βάσης δεδομένων. Μια συσχέτιση συνδέει δύο ή και περισσότερες οντότητες μεταξύ τους και παριστάνεται στο διάγραμμα οντοτήτων–συσχετίσεων μ’ έναν ρόμβο.

Οι συσχετίσεις είναι απαραίτητες για να μπορέσουμε να αντλήσουμε πληροφορίες που αφορούν δύο ή και περισσότερες οντότητες, όπως για παράδειγμα ποιοι πελάτες έκαναν παραγγελίες κάποια συγκεκριμένη χρονική περίοδο (συσχέτιση ΠΑΡΑΓΓΕΛΝΕΙ) ή ποιοι αθλητές ανήκουν σε ποιους συλλόγους (συσχέτιση ΑΝΗΚΕΙ) ή ποιοι αθλητές έλαβαν μέρος σε αγωνίσματα μια συγκεκριμένη χρονιά (συσχέτιση ΣΥΜΜΕΤΕΧΕΙ) κοκ.

Όταν οι οντότητες που συμμετέχουν σε μια συσχέτιση είναι δύο, η συσχέτιση αποκαλείται *διμελής* ή *δυαδική*. Ο βαθμός μιας συσχέτισης είναι ίσος με το πλήθος των οντοτήτων που συμμετέχουν σ’ αυτήν. Μια συσχέτιση μπορεί και η ίδια να έχει ιδιότητες που να περιγράφουν ορισμένα χαρακτηριστικά της, όπως για παράδειγμα η συσχέτιση ΠΑΡΑΓΓΕΛΙΑ ανάμεσα στις οντότητες ΠΕΛΑΤΗΣ και ΠΡΟΪΟΝ μπορεί να περιέχει τις ιδιότητες (πεδία) ΚωδικόςΠελάτη, ΚωδικόςΠροϊόντος, ΚωδικόςΠαραγγελίας, ΗμερομηνίαΠαραγγελίας, Ποσότητα κ.ά.

Στην περίπτωση αυτή το σωστό είναι να δημιουργήσουμε μια ακόμα οντότητα, την οντότητα ΠΑΡΑΓΓΕΛΙΑ, η οποία και θα περιέχει όλες τις παραπάνω ιδιότητες, και να μετονομάσουμε την προηγούμενη συσχέτιση από ΠΑΡΑΓΓΕΛΙΑ σε ΣΥΝΑΛΛΑΓΗ, που δεν θα περιέχει τώρα ιδιότητες. Έτσι, η παραπάνω συσχέτιση θα μετατραπεί από διμελή σε τριμελή.

Όταν σχεδιάζουμε μια βάση δεδομένων, θα πρέπει να εκχωρούμε ιδιότητες μόνο στις οντότητες και να έχουμε τις συσχετίσεις απλά και μόνο για να κατανοούμε τις λογικές συνδέσεις ανάμεσα στις οντότητες.

Οι Διμελείς Συσχετίσεις

Οι διμελείς συσχετίσεις μεταξύ οντοτήτων είναι αυτές που θα μας απασχολήσουν ιδιαίτερα και υπάρχουν τρία βασικά είδη συνδέσεων σ' αυτές, τα εξής :

- *Ένα-προς-ένα (1:1),* όπου μια εμφάνιση της μιας οντότητας συνδέεται με μία και μόνο μία εμφάνιση της άλλης οντότητας. Για παράδειγμα, η οντότητα ΣΥΛΛΟΓΟΣ έχει έναν μόνο προπονητή, ενώ η οντότητα ΠΡΟΠΟΝΗΤΗΣ συνδέεται μ' έναν και μόνο έναν σύλλογο. Σ' ένα άλλο παράδειγμα, η οντότητα ΝΟΜΟΣ έχει μία μόνο πόλη σαν πρωτεύουσα, ενώ η οντότητα ΠΡΩΤΕΥΟΥΣΑ αντιστοιχεί σ' έναν και μόνο έναν νομό. Στην περίπτωση των διμελών συσχετίσεων του τύπου ένα-προς-ένα, μπορούμε να ενώσουμε τα στοιχεία και των δύο ιδιοτήτων και να δημιουργήσουμε μια μοναδική οντότητα (πίνακα).
- *Ένα-προς-πολλά (1:Μ),* όπου μια εμφάνιση της μιας οντότητας συνδέεται με πολλές εμφανίσεις της άλλης οντότητας αλλά κάθε εμφάνιση της δεύτερης οντότητας συνδέεται με μία και μόνο μία εμφάνιση της πρώτης οντότητας. Για παράδειγμα, ένας ΠΕΛΑΤΗΣ κάνει πολλές παραγγελίες, αλλά μια ΠΑΡΑΓΓΕΛΙΑ αντιστοιχεί σ' έναν και μόνο έναν πελάτη. Σ' ένα άλλο παράδειγμα, ένας ΣΥΛΛΟΓΟΣ έχει πολλούς αθλητές, αλλά ένας ΑΘΛΗΤΗΣ ανήκει σ' έναν και μόνο έναν σύλλογο. Οι διμελείς συσχετίσεις του τύπου ένα-προς-ένα είναι οι πιο συχνά συναντώμενες και οι πιο βολικές στη διαχείριση.
- *Πολλά-προς-πολλά (Μ:Ν),* όπου σε μια εμφάνιση της μιας οντότητας αντιστοιχούν πολλές εμφανίσεις της άλλης οντότητας και σε κάθε εμφάνιση της δεύτερης οντότητας αντιστοιχούν πολλές εμφανίσεις της πρώτης οντότητας. Για παράδειγμα, ένας ΑΘΛΗΤΗΣ συμμετέχει σε πολλούς αγώνες αλλά και σ' έναν ΑΓΩΝΑ λαμβάνουν μέρος πολλοί αθλητές. Σ' ένα άλλο παράδειγμα, ένας ΚΑΘΗΓΗΤΗΣ διδάσκει σε πολλούς μαθητές αλλά και ένας ΜΑΘΗΤΗΣ διδάσκεται από πολλούς καθηγητές. Για να μπορέσουμε να διαχειριστούμε μια διμελή σχέση του τύπου πολλά-προς-πολλά, θα πρέπει να δημιουργήσουμε έναν

τρίτο πίνακα που θα περιέχει δύο μόνο ιδιότητες (πεδία), δηλ. τα πεδία κλειδιά των δύο οντοτήτων, οπότε ο συνδυασμός τους θα είναι και το πεδίο κλειδί (σύνθετο κλειδί) του νέου πίνακα.

Το Διάγραμμα Οντοτήτων Συσχετίσεων

Για να μπορέσουμε να διαμορφώσουμε το διάγραμμα οντοτήτων συσχετίσεων, θα πρέπει να ακολουθήσουμε τα εξής βήματα :

- Να ορίσουμε τις οντότητες (πίνακες) που θα ανήκουν στη βάση δεδομένων που θέλουμε να δημιουργήσουμε.
- Να ορίσουμε τις ιδιότητες (πεδία) και τα πρωτεύοντα κλειδιά της κάθε οντότητας (πίνακα).
- Να ορίσουμε τις συσχετίσεις ανάμεσα στις οντότητες.
- Δημιουργούμε το διάγραμμα οντοτήτων συσχετίσεων, όπου θα απεικονίσουμε τις οντότητες, τις ιδιότητές τους και τις συσχετίσεις τους.

Θα δούμε το διάγραμμα οντοτήτων συσχετίσεων για μια βάση δεδομένων με ομάδες (συλλόγους) ποδοσφαίρου, όπου θα έχουμε τις οντότητες ΑΘΛΗΤΗΣ, ΣΥΛΛΟΓΟΣ, ΠΡΟΠΟΝΗΤΗΣ και ΑΓΩΝΑΣ. Οι συσχετίσεις ανάμεσα στις οντότητες αυτές θα είναι οι εξής :

- ΑΘΛΗΤΗΣ – ΣΥΛΛΟΓΟΣ : ένα-προς-πολλά (1:Μ)
- ΣΥΛΛΟΓΟΣ – ΠΡΟΠΟΝΗΤΗΣ : ένα-προς-ένα (1:1)
- ΑΘΛΗΤΗΣ – ΑΓΩΝΑΣ : πολλά-προς-πολλά (Μ:Ν)

Λογικός Σχεδιασμός μιας Βάσης Δεδομένων

Αφού έχουμε δημιουργήσει το διάγραμμα οντοτήτων συσχετίσεων και έχουμε επιλέξει το σχεσιακό μοντέλο δεδομένων για την υλοποίηση της βάσης δεδομένων, ακολουθούμε τη διαδικασία της κανονικοποίησης και είμαστε έτοιμοι για την καταχώριση των στοιχείων της βάσης δεδομένων. Ανάλογα τώρα με το είδος της διμελούς συσχέτισης, διακρίνουμε τις εξής περιπτώσεις ως προς τον λογικό σχεδιασμό που θα πρέπει να ακολουθήσουμε :

Αν η συσχέτιση των δύο πινάκων είναι ένα-προς-ένα, τότε μπορούμε είτε να συνενώσουμε τους δύο πίνακες, με τις αντίστοιχες εγγραφές φυσικά, ή να προσθέσουμε το ένα από τα δύο πεδία κλειδιά σαν ξένο κλειδί στον άλλον πίνακα ή τέλος να δημιουργήσουμε έναν καινούργιο πίνακα με μόνα πεδία τα πεδία κλειδιά των δύο πινάκων (σύνθετο κλειδί). Η προτιμότερη διαδικασία είναι η πρώτη, δηλ. η συνένωση των δύο πινάκων σ' έναν ενιαίο πίνακα.

Αν η συσχέτιση των δύο πινάκων είναι ένα-προς-πολλά, τότε μπορούμε είτε να προσθέσουμε το ένα από τα δύο πεδία κλειδιά σαν ξένο κλειδί στον άλλον πίνακα ή να δημιουργήσουμε έναν καινούργιο πίνακα με μόνα πεδία τα πεδία κλειδιά των δύο πινάκων (σύνθετο κλειδί). Η προτιμότερη διαδικασία είναι η πρώτη, δηλ. η προσθήκη του ξένου κλειδιού στην πλευρά 'πολλά' της σχέσης.

Αν η συσχέτιση των δύο πινάκων είναι πολλά-προς-πολλά, τότε το μόνο που μπορούμε και πρέπει να κάνουμε είναι να δημιουργήσουμε έναν καινούργιο πίνακα με μόνα πεδία τα πεδία κλειδιά των δύο πινάκων (σύνθετο κλειδί), όπου το κάθε πεδίο κλειδί από μόνο του γίνεται ξένο κλειδί. Οι δύο αρχικοί πίνακες δεν μεταβάλλονται.

Η Κανονικοποίηση

Τα προβλήματα που είναι πιθανό να παρουσιασθούν κατά τη διαδικασία της υλοποίησης του σχεδιασμού μιας βάσης δεδομένων είναι η περιττή (άσκοπη) επανάληψη πληροφοριών, που είναι γνωστή με τον όρο redundancy, καθώς και δυσκολίες στην ενημέρωση της βάσης δεδομένων. Τα παραπάνω προβλήματα είναι γνωστά ως *πλεονασμοί δεδομένων* και*ανωμαλίες ενημέρωσης* και για να αντιμετωπιστούν με επιτυχία, θα πρέπει να διασπάσουμε τις μεγάλες σχέσεις σε μικρότερες. Αυτό γίνεται με τη διαδικασία της κανονικοποίησης, έτσι ώστε η βάση δεδομένων να είναι έτοιμη για καταχώριση στοιχείων.

Η *κανονικοποίηση (normalization)* είναι μια τεχνική που ασχολείται με την ανάλυση των σχέσεων (συσχετίσεων) σε μια βάση δεδομένων, όπου κάνουμε μετατροπή των αρχικών μεγάλων σχέσεων σε μικρότερες.

Πλεονασμός Δεδομένων και Ανωμαλίες Ενημέρωσης

Με τον όρο πλεονασμός δεδομένων (data redundancy) εννοούμε την άσκοπη επανάληψη στοιχείων (πληροφοριών). Τα προβλήματα που προκύπτουν από τον πλεονασμό δεδομένων είναι γνωστά με τον όρο ανωμαλίες ενημέρωσης (update anomalies). Για να μπορέσουμε να κατανοήσουμε τις παγίδες του πλεονασμού δεδομένων, θα δούμε ένα παράδειγμα με τους πίνακες ΠΕΛΑΤΗΣ και ΠΑΡΑΓΓΕΛΙΑ. Η σχέση μεταξύ τους είναι ένα-προς-πολλά, δηλ. ένας

πελάτης μπορεί να κάνει πολλές παραγγελίες, αλλά μια παραγγελία γίνεται μόνο από έναν πελάτη.

Αν αποφασίσουμε να δημιουργήσουμε έναν μόνον πίνακα, όπου σε κάθε γραμμή (εγγραφή) του θα υπάρχουν όλα τα στοιχεία της παραγγελίας και δίπλα όλα τα στοιχεία του πελάτη που έχει κάνει την παραγγελία, τότε θα έχουμε πέσει στην παγίδα του πλεονασμού δεδομένων και αυτό γιατί τα στοιχεία του κάθε πελάτη θα επαναλαμβάνονται για κάθε παραγγελία που έχει κάνει.

Ανωμαλία εισαγωγής έχουμε στην περίπτωση που θελήσουμε να καταχωρήσουμε μια καινούργια παραγγελία, οπότε θα πρέπει να καταχωρήσουμε εκ νέου κι όλα τα στοιχεία του πελάτη που έκανε τη συγκεκριμένη παραγγελία, κάτι που είναι κουραστικό, χρονοβόρο και περιέχει τον κίνδυνο λαθών.

Ένα άλλο πρόβλημα με ανωμαλία εισαγωγής έχουμε στην περίπτωση που θελήσουμε να καταχωρήσουμε ένα καινούργιο πελάτη ο οποίος δεν έχει κάνει ακόμα καμία παραγγελία, οπότε η βάση δεδομένων δεν θα μας το επιτρέψει και αυτό γιατί δεν δέχεται κενή τιμή (null) για το πεδίο κλειδί Κωδικός Παραγγελίας.

Ανωμαλία διαγραφής έχουμε στην περίπτωση που θελήσουμε να διαγράψουμε μια παραγγελία που είναι η μοναδική ενός πελάτη, οπότε θα χάσουμε και όλες τις πληροφορίες του συγκεκριμένου πελάτη.

Ανωμαλία τροποποίησης έχουμε στην περίπτωση που θελήσουμε να αλλάξουμε κάποιο στοιχείο ενός πελάτη, όπως τη διεύθυνση ή το τηλέφωνό του, οπότε θα πρέπει να τροποποιήσουμε όλες τις εγγραφές του πίνακα όπου εμφανίζεται ο συγκεκριμένος πελάτης. Αν δεν κάνουμε την αλλαγή σ' όλες τις εγγραφές, τότε ο πελάτης θα εμφανίζεται να έχει δύο διευθύνσεις ή δύο τηλέφωνα κοκ. Μιλάμε τότε για *μη συνεπή (inconsistent)* βάση δεδομένων.

Ως γνωστόν, η λύση στο παραπάνω πρόβλημα είναι να δημιουργήσουμε έναν πίνακα με τα στοιχεία των πελατών και έναν ξεχωριστό πίνακα με τα στοιχεία των παραγγελιών, όπου θα υπάρχει και σαν πεδίο (ξένο κλειδί) ο ΚωδικόςΠελάτη.

Η Διαδικασία της Κανονικοποίησης

Η μέθοδος της κανονικοποίησης βοηθάει στον λογικό σχεδιασμό μιας βάσης δεδομένων και είναι συμπληρωματική του μοντέλου οντοτήτων συσχετίσεων. Το κέρδος για μας είναι ότι δεν υπάρχουν προβλήματα συνέπειας, πλεονασμού και εγκυρότητας των πληροφοριών της βάσης δεδομένων. Ακολουθώντας τη διαδικασία της κανονικοποίησης κάνουμε συνεχείς διασπάσεις των πινάκων σε

πιο απλές και συμπαγείς μορφές, με στόχο πάντα να αποφύγουμε τον πλεονασμό (επανάληψη) των δεδομένων. Αφαιρούμε πεδία από τις αρχικές μεγάλες σχέσεις και τα τοποθετούμε σε νέες σχέσεις έτσι ώστε να μπορούμε να έχουμε τις ίδιες πληροφορίες και με τις νέες σχέσεις.

Μπορούμε να πούμε ότι *κανονικοποίηση (normalization)* είναι η διαδικασία μετατροπής των δεδομένων κάποιων σχέσεων (πινάκων) σε πιο απλές και πιο σαφείς σχέσεις, χωρίς πλεονασμούς (επαναλήψεις) των δεδομένων. Οι βασικές μορφές της κανονικοποίησης είναι τρεις, η πρώτη (1^{η} NF), η δεύτερη (2^{η} NF) και η τρίτη (3^{η} NF). Θα ξεκινήσουμε τη μελέτη μας με μια βάση δεδομένων που θέλουμε να κατασκευάσουμε για τους αθλητές στο αγώνισμα των 100 μέτρων στίβου, οι οποίοι προέρχονται από διάφορους συλλόγους διαφόρων χωρών και όπου μας ενδιαφέρουν οι επιδόσεις τους σε διάφορους διεθνείς αγώνες.

Αρχικά, χωρίς καμία μελέτη, θα μπορούσε κάποιος να θεωρήσει ότι τα δεδομένα για τους αθλητές και τις επιδόσεις τους στους αγώνες θα ήταν κάπως έτσι :

Κωδικός_Αθλητή	**Επώνυμο_Όνομα**	**Κωδικός_Συλλόγου**	**Ονομασία_Συλλόγου**	**Κωδικός_Αγώνα**	**Αγώνας**	**Επίδοση**	**Κωδικός_Χώρας**	**Ονομασία_Χώρας**
100	Lewis Carl	200	America	01 02 03	LA 2004 MO 2003 RO 2002	10.08 10.04 10.07	300	USA
101	Wells John	201	Rangers	01 03 04	LA 2004 RO 2002 GE2001	10.04 10.05 10.03	301	Britain

Παρατηρούμε ότι τα πεδία *Κωδικός_Αγώνα*, *Αγώνας* και *Επίδοση* έχουν περισσότερες από μία τιμές. Αυτό είναι αντίθετο με τις αρχές της σχεσιακής θεωρίας, γιατί κάθε σχέση του σχεσιακού μοντέλου θα πρέπει να έχει πεδία με μία και μοναδική τιμή σε κάθε σειρά (εγγραφή).

Δεν μπορούμε φυσικά να προσθέσουμε κι άλλα πεδία, δηλ. ένα πεδίο για κάθε αγώνα, γιατί στο σχεσιακό μοντέλο δεν μπορεί μια σχέση να έχει μεταβαλλόμενο αριθμό πεδίων και ούτε είμαστε σε θέση να γνωρίζουμε σε πόσους αγώνες πήρε μέρος ένας αθλητής.

Μια πρώτη λύση θα ήταν να μην έχουμε επαναλαμβανόμενες τιμές στην ίδια σειρά και να προσπαθήσουμε να δημιουργήσουμε έτσι έναν νέο πίνακα σε κάθε κελί του οποίου να περιέχεται μία μόνο τιμή, όπως φαίνεται στο παρακάτω σχήμα :

Κωδικός_Αθλητή	**Επώνυμο_Όνομα**	**Κωδικός_Συλλόγου**	**Ονομασία_Συλλόγου**	**Κωδικός_Αγώνα**	**Αγώνας**	**Επίδοση**	**Κωδικός_Χώρας**	**Ονομασία_Χώρας**
100	Lewis Carl	200	America	01	LA 2004	10.08	300	USA
100	Lewis Carl	200	America	02	MO 2003	10.04	300	USA
100	Lewis Carl	200	America	03	RO 2002	10.07	300	USA
101	Wells John	201	Rangers	01	LA 2004	10.04	301	Britain
101	Wells John	201	Rangers	03	RO 2002	10.05	301	Britain
101	Wells John	201	Rangers	04	GE2001	10.03	301	Britain

Η σχέση ή ο πίνακας που επεξεργαζόμαστε βρίσκεται τώρα στην $1^{η}$ κανονική μορφή ($1^{η}$ NF). Ο ορισμός λέει ότι μια σχέση (πίνακας) βρίσκεται στην *$1^{η}$ κανονική μορφή* όταν περιέχει σταθερό και όχι μεταβλητό αριθμό πεδίων (στηλών) και κάθε πεδίο της σχέσης δεν περιέχει επαναλαμβανόμενες τιμές. Επίσης, κάθε κελί της σχέσης (διασταύρωση γραμμής και στήλης) θα πρέπει να περιέχει μία μόνο τιμή. Τα μειονεκτήματα που βλέπουμε αμέσως ότι προκύπτουν από τη νέα μορφή που πήρε ο πίνακας είναι ότι έχουμε τώρα περισσότερες γραμμές για να απεικονίσουμε τα ίδια ακριβώς δεδομένα και φυσικά έχουμε περιττή επανάληψη τιμών.

Αν είμαστε στην $1^{η}$ κανονική μορφή, για να μπορέσουμε να προχωρήσουμε στην $2^{η}$ και στην $3^{η}$ κανονική μορφή, θα πρέπει να ορίσουμε πρώτα ένα πρωτεύον κλειδί, δηλ. ένα πεδίο ή έναν συνδυασμό από δύο ή περισσότερα πεδία (σύνθετο κλειδί) για να μπορούμε να προσδιορίζουμε μονοσήμαντα την κάθε γραμμή (εγγραφή ή και πλειάδα).

Στο παραπάνω παράδειγμα, παρατηρούμε ότι ο πιο κατάλληλος συνδυασμός πεδίων για να προσδιορίσει μονοσήμαντα την κάθε γραμμή είναι τα πεδία *Κωδικός_Αθλητή* και*Κωδικός_Αγώνα*. Τώρα, αν το κλειδί που έχουμε ορίσει είναι σύνθετο, δηλ. αποτελείται από δύο ή περισσότερα πεδία, θα πρέπει να συνεχίσουμε με την $2^{η}$ κανονική μορφή ($2^{η}$ NF), αλλιώς θα πρέπει να συνεχίσουμε με την $3^{η}$ κανονική μορφή ($3^{η}$ NF).

Συνεχίζοντας τώρα με την $2^{η}$ κανονική μορφή, ψάχνουμε να βρούμε τα πεδία εκείνα που να συσχετίζονται με (αφορούν, εξαρτώνται από) ολόκληρο το σύνθετο κλειδί. Παίρνουμε τα πεδία που συγκροτούν το σύνθετο κλειδί και από τα πεδία αυτά δημιουργούμε έναν καινούργιο πίνακα.

Στον πίνακά μας, το μόνο πεδίο που έχει σχέση με τον συνδυασμό των πεδίων που συγκροτούν το σύνθετο κλειδί, δηλ. με τα πεδία *Κωδικός_Αθλητή* και *Κωδικός_Αγώνα*, είναι προφανώς το πεδίο *Επίδοση*. Δημιουργούμε τώρα τον παρακάτω πίνακα.

Κωδικός_Αθλητή	**Κωδικός_Αγώνα**	**Επίδοση**
100	01	10.08
100	02	10.04
100	03	10.07
101	01	10.04
101	03	10.05
101	04	10.03

Συνεχίζοντας με την 2η κανονική μορφή, προσπαθούμε τώρα να βρούμε ποια από τα υπόλοιπα πεδία του πίνακα εξαρτώνται από κάθε ξεχωριστό πεδίο του σύνθετου κλειδιού. Παίρνουμε αυτό το ξεχωριστό πεδίο ως πρωτεύον κλειδί και με τα πεδία που έχουν σχέση μ' αυτό δημιουργούμε και από έναν καινούργιο πίνακα κάθε φορά.

Στο παράδειγμά μας, θα προκύψουν οι εξής δύο καινούργιοι πίνακες :

Κωδικός_ Αθλητή	**Επώνυμο _Όνομα**	**Κωδικός_ Συλλόγου**	**Ονομασία_ Συλλόγου**	**Κωδικός _Χώρας**	**Ονομασία _Χώρας**
100	Lewis Carl	200	America	300	USA
100	Lewis Carl	200	America	300	USA
100	Lewis Carl	200	America	300	USA
101	Wells John	201	Rangers	301	Britain
101	Wells John	201	Rangers	301	Britain
101	Wells John	201	Rangers	301	Britain

Κωδικός_Αγώνα	**Αγώνας**
01	LA 2004
02	MO 2003
03	RO 2002
01	LA 2004
03	RO2002
04	GE2001

Φυσικά, στον δεύτερο πίνακα που έχει σχέση με τους Αγώνες, θα μπορούσαμε να είχαμε συμπεριλάβει και πεδία που να αφορούν έναν συγκεκριμένο αγώνα, όπως Πόλη, Χώρα, Ημερομηνία, Διεθνής ή Φιλικός κ.ά. Βλέπουμε ότι έχουμε ήδη εξαλείψει ένα μεγάλο μέρος του πλεονασμού δεδομένων που είχαμε στον αρχικό πίνακα. Η σχέση που επεξεργαζόμαστε βρίσκεται τώρα στην $2^{η}$ κανονική μορφή ($2^{η}$ NF). Ο ορισμός λέει ότι μια σχέση βρίσκεται στην *$2^{η}$ κανονική μορφή* όταν έχει προέλθει από σχέση της $1^{ης}$κανονικής μορφής και ακόμη τα πεδία που δεν ανήκουν στο κλειδί έχουν σχέση μόνο με το κλειδί.

Βλέπουμε, όμως, ότι υπάρχουν ακόμη πλεονασμοί δεδομένων, όπως συμβαίνει με τα πεδία *Ονομασία_Συλλόγου* και *Ονομασία_Χώρας* στον πίνακα που έχει ως πεδίο κλειδί τον*Κωδικό_Αθλητή*. Θα πρέπει συνεπώς να προχωρήσουμε σε δύο ακόμη διασπάσεις του πίνακα αυτού για να αποφύγουμε αυτές τις επαναλήψεις τιμών.

Κωδικός_Αθλητή	**Επώνυμο_Όνομα**	**Κωδικός_Συλλόγου**
100	Lewis Carl	200
100	Lewis Carl	200
100	Lewis Carl	200
101	Wells John	201
101	Wells John	201
101	Wells John	201

Κωδικός_Συλλόγου	**Ονομασία_Συλλόγου**	**Κωδικός_Χώρας**
200	America	300
200	America	300
200	America	300
201	Rangers	301
201	Rangers	301
201	Rangers	301

Κωδικός_Χώρας	**Ονομασία_Χώρας**
300	USA
300	USA
300	USA
301	Britain
301	Britain
301	Britain

Έχουμε φθάσει σ' ένα σημείο που δεν χρειάζεται περαιτέρω διάσπαση των πινάκων καθώς στους πίνακες που έχουμε καταλήξει δεν υπάρχουν πεδία που να περιγράφουν κάτι που να έχει σχέση με κάποιο άλλο πεδίο εκτός από το πεδίο κλειδί. Η σχέση που επεξεργαζόμαστε βρίσκεται τώρα στην $3^{η}$ κανονική μορφή ($3^{η}$ NF). Ο ορισμός λέει ότι μια σχέση βρίσκεται στην *$3^{η}$ κανονική μορφή* όταν ικανοποιεί τις απαιτήσεις της $1^{ης}$ και της $2^{ης}$ κανονικής μορφής και ακόμη δεν υπάρχει κάποιο πεδίο στον πίνακα που να εξαρτάται από κάποιο άλλο πεδίο διαφορετικό του πρωτεύοντος κλειδιού.

Apache Web Server

Λειτουργίες και χαρακτηριστικά

Ο Apache Web Server είναι αυτό ακριβώς που δηλώνει το όνομά του. Πρόκειται δηλαδή για έναν εξυπηρετητή (server) του παγκόσμιου Ιστού (Web). Με τον όρο server το μυαλό μας πηγαίνει ίσως σε ηλεκτρονικούς υπολογιστές που φιλοξενούν ιστοσελίδες και όχι άδικα. Ο όρος αυτός χρησιμοποιείται και για το μηχάνημα εξυπηρετητή (hardware) αλλά και για το πρόγραμμα (software). Στο άρθρο αυτό θα ασχοληθούμε μόνο με το software και συγκεκριμένα με τον Apache.

Ο Apache εγκαθίσταται σε έναν υπολογιστή ο οποίος μπορεί να χρησιμοποιεί διάφορα λειτουργικά συστήματα όπως Linux, Unix, Microsoft Windows, GNU, FreeBSD, Solaris, Novell NetWare, Mac OS X, OS/2, TPF. Ο ρόλος του Apache είναι να αναμένει αιτήσεις από διάφορα προγράμματα – χρήστες (clients) όπως είναι ένας ο φυλλομετρητής (browser) ενός χρήστη και στη συνέχεια να εξυπηρετεί αυτές τις αιτήσεις “σερβίροντας” τις σελίδες που ζητούν είτε απευθείας μέσω μιας ηλεκτρονικής διεύθυνσης (URL), είτε μέσω ενός συνδέσμου (link). Ο τρόπος με τον οποίο ο Apache εξυπηρετεί αυτές τις αιτήσεις, είναι σύμφωνος με τα πρότυπα που ορίζει το πρωτόκολλο HTTP (Hypertext Transfer Protocol).

Το ξεκίνημα και η εξέλιξη του Apache Web Server

Η περίοδος έναρξης της δημιουργίας του προγράμματος χρονολογείται στις αρχές του 1990, όταν άρχισε να αναπτύσσεται από τον Robert McCool, ως ένα project του National Center for Supercomputing Applications (NCSA) με το όνομα HTTPd (HTTP daemon). Το 1994 ο Robert McCool αποχώρησε από το NCSA με αποτέλεσμα το NCSA HTTPd να μείνει σχεδόν εγκαταλειμμένο, πέρα από κάποιες διορθώσεις (patches) που ανέπτυσσαν και διένειμαν εκτός από τον McCool και άλλοι προγραμματιστές,. Το 1995 ανέλαβε το πρόγραμμα το Ίδρυμα Λογισμικού Apache (Apache Software Foundation), το οποίο διατηρεί την εποπτεία του έως και σήμερα.

Ο Apache σήμερα

Ο Apache HTTP αναπτύσσεται από την "Κοινότητα Ανοιχτού Λογισμικού" και η εποπτεία, υποστήριξη, και διάθεση του προγράμματος γίνεται από το Apache Software Foundation. Το πρόγραμμα είναι ανοιχτού κώδικα (open source), κάτι που σημαίνει ότι σύμφωνα με την άδεια χρήσης του (license), διατίθεται δωρεάν και μπορούν να γίνουν ελεύθερα από το χρήστη προσθήκες και τροποποιήσεις στον κώδικα του.

Η προέλευση του ονόματος του

Υπάρχουν δύο εκδοχές σχετικά με την προέλευση του ονόματος του. Η πρώτη εντοπίζεται στα πρώτα χρόνια της δημιουργίας του, τότε που ως NCSA HTTPd έπρεπε να αναπτύσσονται συνεχώς διορθώσεις (patches) για να ενσωματωθούν στον αρχικό του κώδικα με αποτέλεσμα να του δοθεί το όνομα a patchy server. Η δεύτερη εκδοχή σύμφωνα με το ίδρυμα Apache, αναφέρει ότι το όνομα αυτό δόθηκε προς τιμήν των ιθαγενών Ινδιάνων της Αμερικής και συμβολίζει το μαχητικό πνεύμα και την αντοχή.

Χαρακτηριστικά και λειτουργίες του Apache HTTP

Ο Apache διαθέτει ποικιλία χαρακτηριστικών και μπορεί να υποστηρίξει μια μεγάλη γκάμα εφαρμογών με τις οποίες και συνεργάζεται. Οι δυνατότητες του προγράμματος αυτού καθαυτού και τα χαρακτηριστικά του δεν είναι και τόσο πολλά. Ένα από τα βασικότερα χαρακτηριστικά του όμως, το οποίο και του δίνει μεγάλες δυνατότητες, είναι ότι μπορεί να προσαρμόσει επάνω του πολλές προσθήκες προγραμμάτων (modules), τα οποία με τη σειρά τους παρέχουν διαφορετικές λειτουργίες. Μερικά από τα πιο γνωστά modules του Apache HTTP είναι τα modules πιστοποίησης, όπως για παράδειγμα τα mod_access, mod_auth, mod_digest κ.λπ. Παρέχει επίσης SSL σε TLS μέσω των (mod_ssl), και proxy module (mod_proxy), πραγματοποιεί ανακατευθύνσεις διευθύνσεων (URL rewrites) μέσω του mod_rewrite, καταγραφές συνδέσεων μέσω του mod_log_config, συμπίεση αρχείων μέσω του mod_gzip και πολλά άλλα modules τα οποία διατίθενται είτε απ'ο το Apache Software Foundation, είτε από τρίτες εταιρίες λογισμικού.

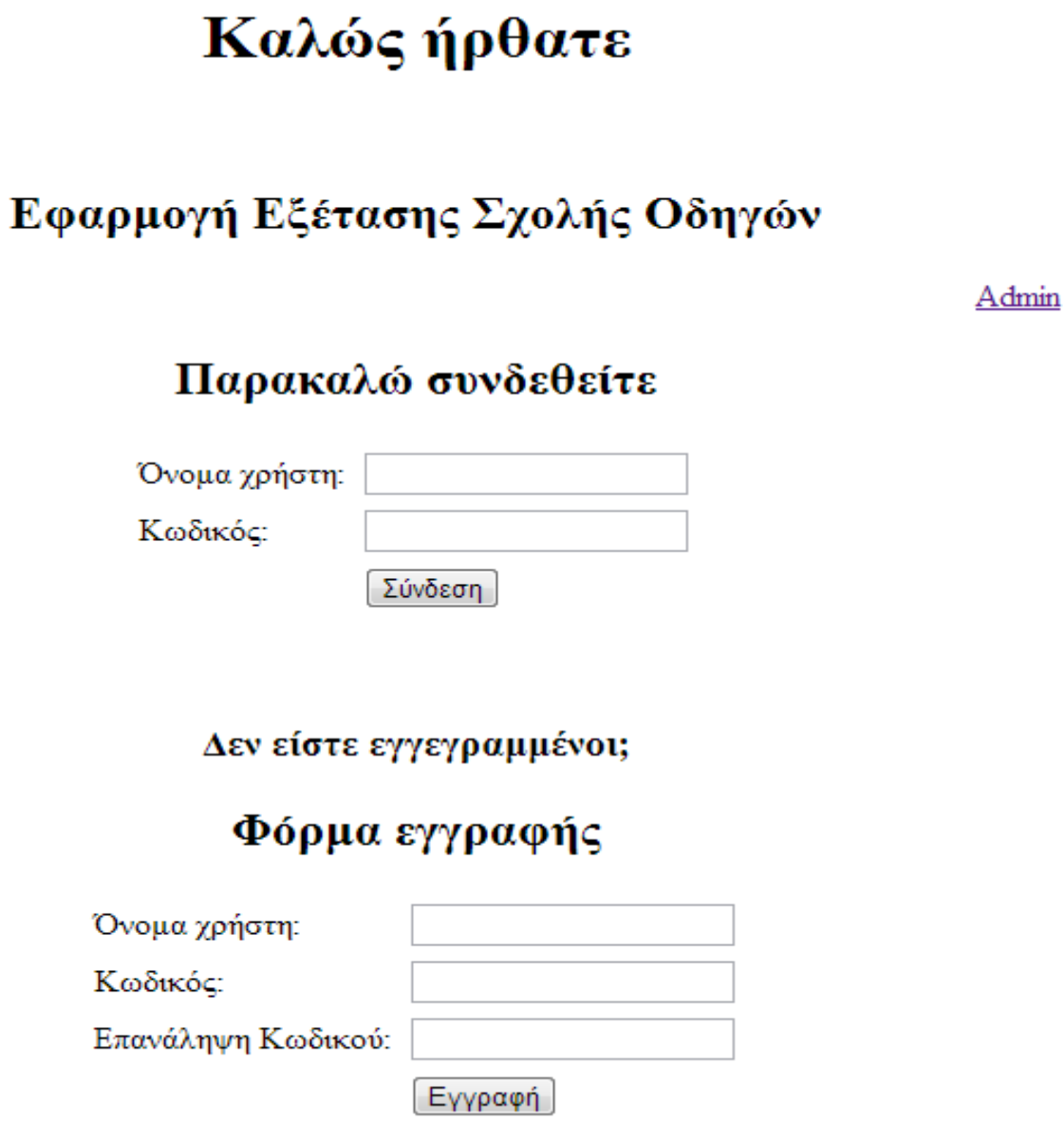
Ένα άλλο χαρακτηριστικό – δυνατότητα του Apache HTTP, όπως έχω αναφέρω πιο πάνω, είναι ότι μπορεί να εγκατασταθεί σε διάφορα λειτουργικά συστήματα. Ο Apache HTTP υποστηρίζει επίσης αρκετές διάσημες εφαρμογές και γλώσσες προγραμματισμού όπως MySQL, PHP, Perl, Python κ.λπ.

Αυτά είναι μερικά από τα χαρακτηριστικά και τις λειτουργίες του που κάνουν τον Apache τον πιο δημοφιλή Web Server από το 1996 έως τις μέρες μας. Περισσότερο από το 50% των ιστοχώρων του παγκόσμιου ιστού, χρησιμοποιεί τον Apache ως εξυπηρετητή. Το υπόλοιπο ποσοστό καλύπτουν αντίστοιχα προγράμματα, όπως το Microsoft Internet Information Services (IIS), ο Sun Java System Web Server, ο Zeus Web Server κα.

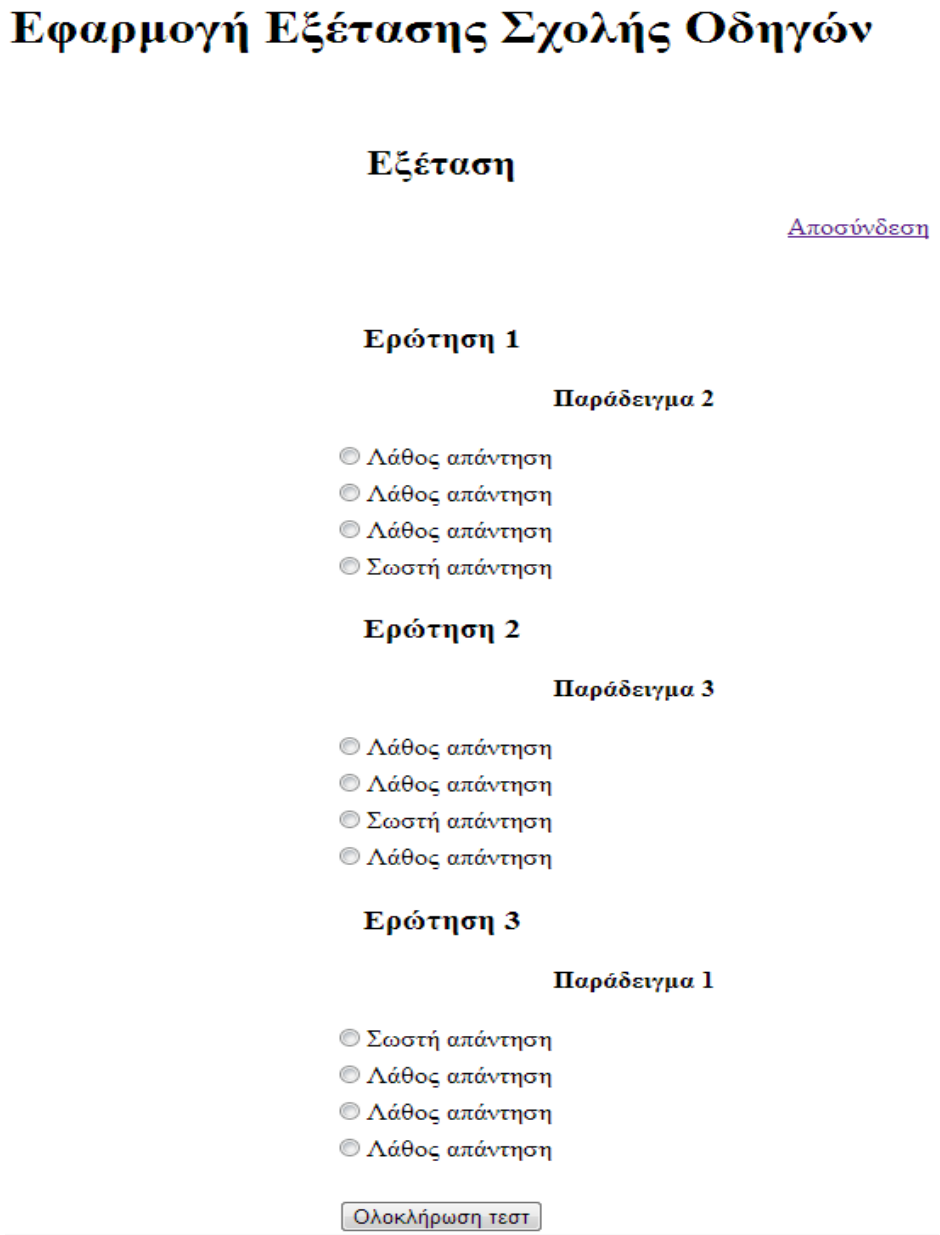
ΚΕΦΑΛΑΙΟ 3 – Παρουσίαση site

3.2 Λειτουργίες και screenshots

- Κεντρική (homepage):
  Εδώ έχουμε την πρώτη οθόνη που βλέπει ο χρήστης όταν εισέρχεται στην εφαρμογή. Υπάρχουν οι δυνατότες εγγραφής (αν δεν έχει λογαριασμό) και σύνδεσης (log in). Ακόμη, υπάρχει σύνδεσμος (link) για τον διαχειρηστή του συστήματος(Admin).

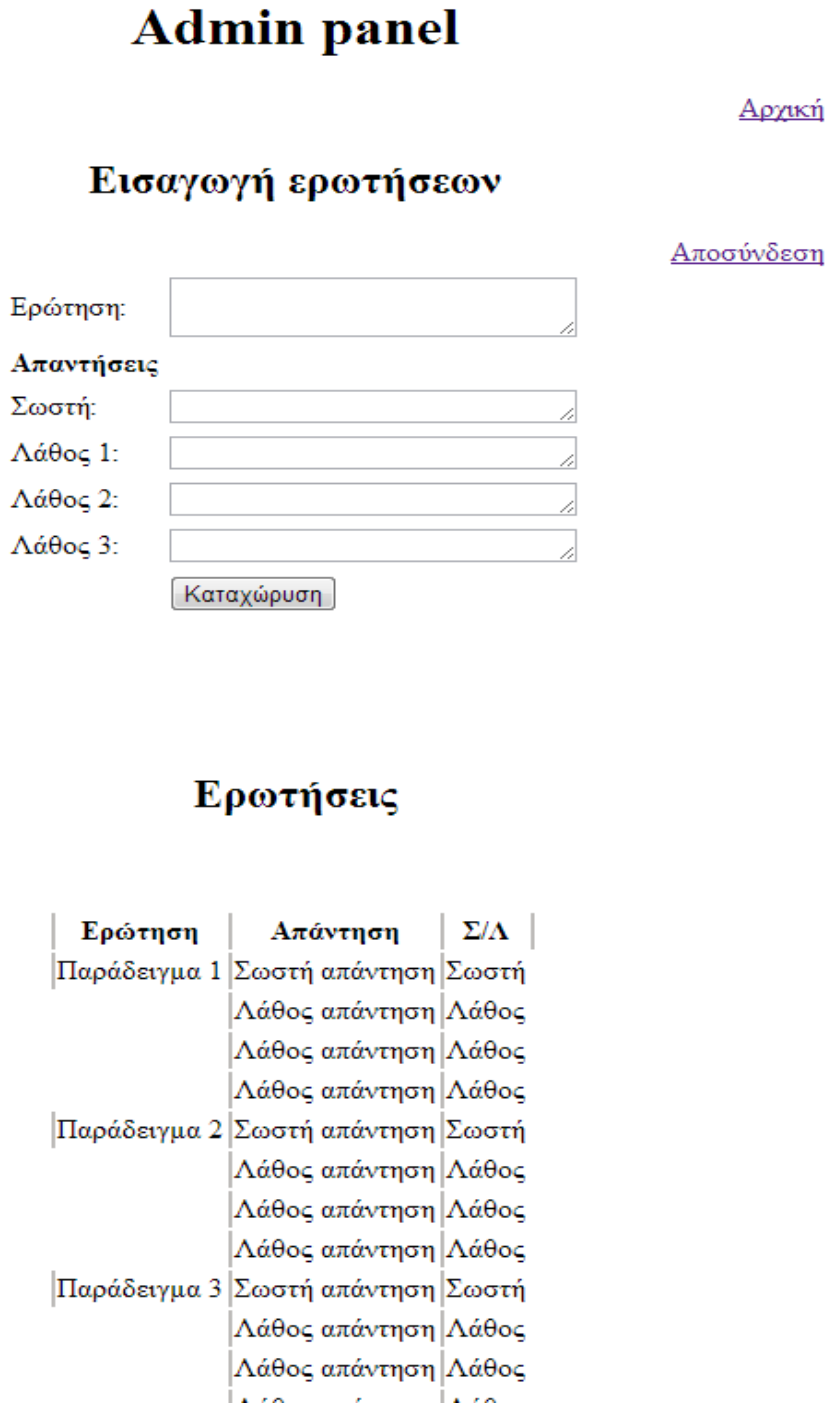


- Οθόνη τεστ:
  Εδώ βλέπουμε ένα παράδειγμα των ερωτήσεων του συστήματος. Το φορμάτ των ερωτήσεων είναι:
  - Ερώτηση (Παράδειγμα [νούμερο])
  - Απαντήσεις
  Παρακάτω μπορούμε να δούμε ότι οι ερωτήσεις εμφανίζονται με τυχαία σειρά κάθε φορά, όπως και οι αντίστοιχες απαντήσεις της κάθε ερώτησης. Στο συγκεκριμένο παράδειγμα χρησιμοποιήσαμε μόνο 3 ερωτήσεις για να εμφανίζεται ολόκληρη η φόρμα, κανονικά εμφανίζονται 10 ερωτήσεις από το σύνολο αυτών που έχουμε περάσει στο σύστημα.



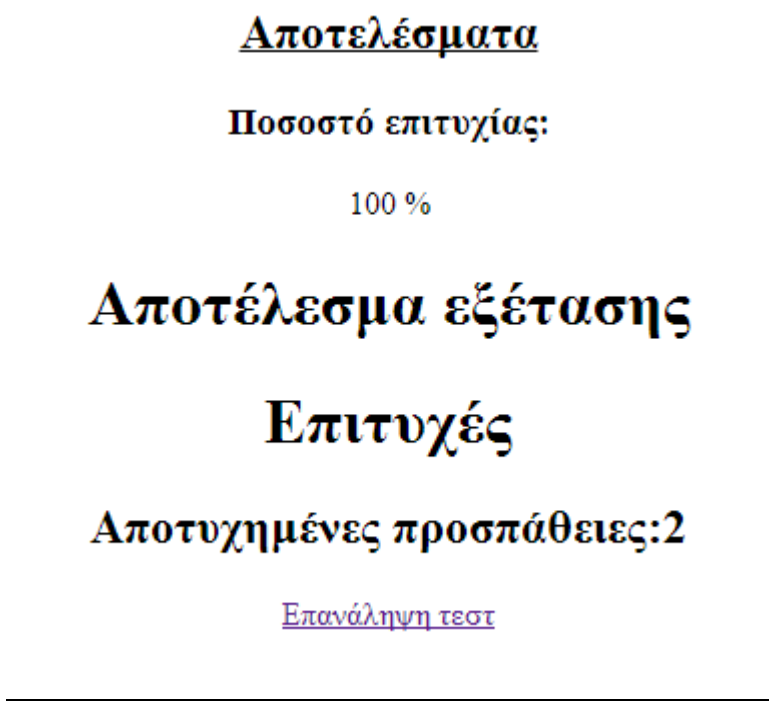
- Admin panel(1):
  Αυτή είναι η οθόνη εισαγωγής κωδικού για να μπορέσει να εισέλθει κανείς στη σελίδα εισαγωγής ερωτήσεων. Είναι στατικά προκαθορισμένος ο κωδικός στο “admin”. Επίσης, υπάρχει σύνδεσμος για επιστροφή στην αρχική σελίδα.

- Admin panel(2):
  Εδώ μπορεί ο χρήστης να εισάγει την ερώτηση που θέλει και στην συνέχεια να ορίσει τις τέσσερεις πιθανές απαντήσεις, μια σωστή και 3 λάθος.Παρακάτω μπορεί να δει όσες ερωτήσεις έχει εισάγει ως τώρα και τις απαντήσεις της κάθε ερώτησης.
  Επιπλέον υπάρχουν σύνδεσμοι για την επιστροφή στην κεντρική σελίδα ή για την αποσύνδεση από το admin panel.



- Οθόνη αποτελεσμάτων:
  Αυτή είναι η τελική οθόνη, όπου εμφανίζεται το ποσοστό επιτυχίας του χρήστη που έκανε το τεστ. Στη συνέχεια, του ανακοινώνεται αν πέτυχε ή απέτυχε στην εξέταση. Τέλος, εμφανίζονται οι αποτυχημένες προσπάθειές του. Το τελευταίο εμφανίζεται μονάχα αφού περάσει επιτυχημένα την εξέταση.

  Ακόμη, υπάρχει σύνδεσμος για να επαναλάβει το τεστ, εφόσον το επιθυμεί...



3.2 Επεξήγηση κώδικα

- Αρχείο test_page.php :

Σε αυτό το αρχείο βλέπουμε τον front-end κώδικα για το test.

```
<? session_start(); ?>

<html>

	<head>

		<meta charset="greek">

		<title>Σχολή Οδηγών</title>

	</head>
```

```
<body>
		<center>
		<h1>Εφαρμογή Εξέτασης Σχολής Οδηγών</h1>
		<p> </p>
		<h2>Εξέταση</h2>
		<p><div align="right"><a href="logout.php">Αποσύνδεση</a></div></p>
		<p> </p>
		<p><form action="results.php" method="POST" />
			<?php include("test.php"); // Κλήση αρχείου back-end που περιέχει το τεστ ?>
			<br /><input type="submit" value="Ολοκλήρωση τεστ" />
		</form></p>
		</center>
	</body>
</html>
```

- Αρχείο test.php :

Εδώ έχουμε τον κώδικα από το αρχείο που καλείται από το test_page.php και το οποίο ουσιαστικά παρέχει τη λειτουργία του τεστ.

```
<?php
	include("db_con.php"); // Σύνδεση με βάση δεδομένων


	$q = mysql_query("SELECT * FROM questions"); // Επιλογή ερωτήσεων από τη βάση δεδομένων
	$rows = mysql_num_rows($q); // Αποθήκευση αριθμού "γραμμών" του πίνακα questions
	$i = 1; // Αρχικοποίηση μετρητή
	$arr = array(); // Αρχικοποίηση array
	$counter = 1; // Αρχικοποίηση μετρητή
	do
	{
		do
```

```
{
    $var = rand(1, $rows); // Εκχώρηση στη $var ενός τυχαίου αριθμού από το 1 μέχρι και τον αριθμό "γραμμών" του πίνακα questions
}while(in_array($var, $arr)); // Έλεγχος αν ο τυχαίος αριθμός έχει ξαναπροκύψει

echo '<h3>Ερώτηση '.$counter.'</h3>'; // Αριθμός ερώτησης
echo '<table>';
// Επιλογή από τη βάση δεδομένων της ερώτησης με την τιμή της $var
$query2 = mysql_query("SELECT * FROM questions as q, answers as a WHERE (".$var."=q.questions_id OR a.belong=".$var.") AND q.questions_id=a.belong");
while($row2 = mysql_fetch_array($query2))
{
    if(!isset($flag)){ echo '<p style="width:800px"><b>'.$row2['question'].'</b></p>'; $flag = 1; } // Εμφάνιση της επιλεγμένης ερώτησης
}
```

```
##====================== Τυχαια εμφάνιση απαντήσεων σε κάθε ερώτηση ======================##

        $query2 = mysql_query("SELECT a.answer, a.value FROM questions as q, answers as a WHERE (".$var."=q.questions_id OR a.belong=".$var.") AND q.questions_id=a.belong");

        while($row2 = mysql_fetch_array($query2))

        {

            mysql_query("INSERT INTO temp (ans, val) VALUES ('".$row2['answer']."', '".$row2['value']."')"); // Καταχώρηση σε προσωρινό πίνακα των απαντήσεων

        }

        $query3 = mysql_query("SELECT * FROM temp");

        $rows2 = mysql_num_rows($query3); // Αποθήκευση αριθμού "γραμμών" του πίνακα temp

        $j = 1; // Αρχικοποίηση μετρητή

        $arr2 = array(); // Αρχικοποίηση array

        do // Αρχή επανάληψης για τυχαία εμφάνιση απαντήσεων

        {
```

```
do
{
    $var2 = rand(1, $rows2); // Εκχώρηση στη $var ενός τυχαίου αριθμού από το 1 μέχρι και τον αριθμό "γραμμών" του πίνακα questions
}while(in_array($var2, $arr2)); // Έλεγχος αν ο τυχαίος αριθμός έχει ξαναπροκύψει
$query3 = mysql_query("SELECT * FROM temp WHERE id=".$var2);
while($row3 = mysql_fetch_array($query3))
{
    echo '<tr><td><input type="radio" name="q'.$counter.'" value="'.$row3['val'].'" required />'.$row3['ans'].'</td></tr>'; // Εμφάνιση απαντήσεων ερώτησης
}

array_push($arr2, $var2); // Εκχώρηση του τυχαίου αριθμού στο $arr
$j++; // Αύξηση του μετρητή κατά 1
}while($j<=4 && $j<=$rows2);
```

```
            mysql_query("TRUNCATE temp"); // Διαγραφή και επαναδημιουργία πίνακα temp
            echo '</table>';
            unset($flag);
            array_push($arr, $var); // Εκχώρηση του τυχαίου αριθμού στο $arr

        ##=============================================================================##
            $i++; // Αύξηση του μετρητή κατά 1
            $counter++;
        }while ($i<=10 && $i<=$rows); // Έλεγχος να εκτελούνται μέχρι 10 επαναλήψεις(όσες οι max ερωτήσεις που θέλουμε να εμφανίζονται) ή μέχρι τα $rows που έχουμε βρει ότι έχει ο πίνακας temp

        mysql_close(); // Κλείσιμο σύνδεσης με βάση
?>
```

- Αρχείο results:

```
<?php
	session_start();
	$sum = 0;
	for($i=1;$i<=10;$i++)
	{
		$sum = $sum + $_POST['q'.$i];
	}
	$result = ($sum/10)*100;

	if($result >= 80)
	{
		$outcome = 'Επιτυχές';
	}
```

```
        else
        {
                $outcome = 'Αποτυχία';
                if(!isset($_SESSION['tries']))
                {
                        $_SESSION['tries'] = 1;
                }
                else
                {
                        $_SESSION['tries'] = $_SESSION['tries'] + 1;
                }
        }
?>
<html>
        <head>
```

```
        <meta charset="greek">
        <title>Σχολή Οδηγών</title>
</head>

<body>
        <center>
                <h2><u>Αποτελέσματα</u></h2>
                <h3>Ποσοστό επιτυχίας:</h3><?=$result?> %
                <h1>Αποτέλεσμα εξέτασης</h1>
                <?php
                        if($outcome == 'Επιτυχές')
                        {
                                echo '<h1>'.$outcome.'</h1>';
                                echo '<h2>Αποτυχημένες προσπάθειες:'.$_SESSION['tries'].'</h2>';
                                unset($_SESSION['tries']);
```

```
                }
                else
                {
                        echo '<h1>'.$outcome.'</h1>';
                }
        ?>
        <p><a href="index.php">Επανάληψη τεστ<a/></p>
    </center>
</body>
</html>
```

ΚΕΦΑΛΑΙΟ 4 – Βιβλιογραφία

4.1 Πηγές Internet

HTML

http://gocreations.gr/html5-hypertext-markup-language-5/

http://www.greektuts.net/html5-intro-part1/

==========

CSS

http://www.wlearn.gr/index.php/home-css-83

http://el.wikipedia.org/wiki/CSS

==========

PHP

http://el.wikipedia.org/wiki/PHP

http://phpmanual.web.fc2.com/greek/intro-whatcando.html

http://gr.ixarticle.com/articles/1016367/

==========

Javascript

http://en.wikipedia.org/wiki/JavaScript

==========

DB

http://dide.flo.sch.gr/Plinet/Tutorials/Tutorials-DataBasesTheory.html

---

Apache Server

http://mytwocents.gr/apache-web-server/