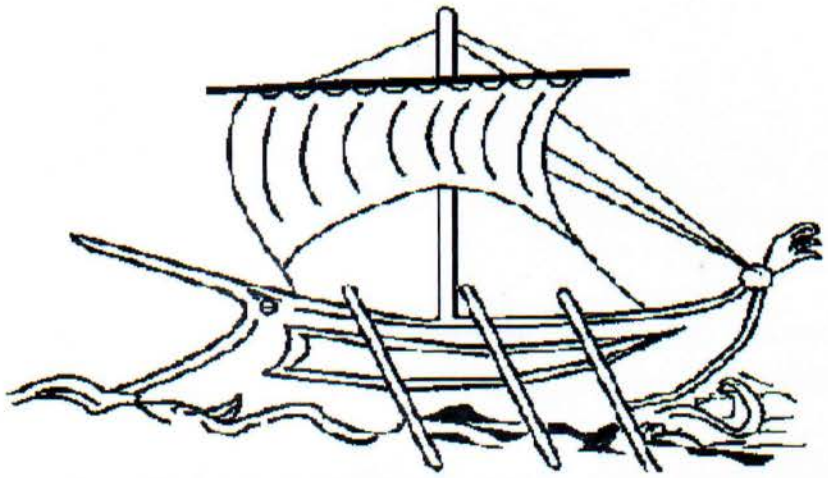
Α.Τ.Ε.Ι ΠΕΙΡΑΙΑ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ

ΤΜΗΜΑ ΑΥΤΟΜΑΤΙΣΜΟΥ



ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

"ΥΛΟΠΟΙΗΣΗ ΔΥΝΑΜΙΚΗΣ ΙΣΤΟΣΕΛΙΔΑΣ ON-LINE ΚΑΤΑΣΤΗΜΑΤΟΣ ΠΛΕΙΣΤΗΡΙΑΣΜΟΥ ΠΑΛΑΙΩΝ ΑΝΤΙΚΕΙΜΕΝΩΝ "

***ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ*:**
ΑΓΓΕΛΟΠΟΥΛΟΣ ΙΩΑΝΝΗΣ

***ΣΠΟΥΔΑΣΤΗΣ*:**
ΑΝΤΩΝΙΑΔΗΣ ΠΕΤΡΟΣ

ΑΘΗΝΑ - 2012

...αφιερώνεται στους φίλους και τους γονείς μου για την τεράστια συμπαράσταση τους όλα τα χρόνια που διήρκησε η φοίτησή μου.

Επίσης, στον επιβλέπων καθηγητή μου κ. Αγγελόπουλο Ιωάννη που με βοήθησε σημαντικά κατά τη δημιουργία αυτής της πτυχιακής εργασίας.

ΠΕΡΙΛΗΨΗ ΠΤΥΧΙΑΚΗΣ

Σκοπός της πτυχιακής εργασίας είναι η υλοποίηση δυναμικής ιστοσελίδας η οποία θα παρουσιάζει ένα ηλεκτρονικό κατάστημα παλαιοπωλείου(με το όνομα e-antiquery).

Μέσω της ιστοσελίδας αυτής θα διατίθενται διάφορα αντικείμενα προς πώληση,με τη μορφή πλειστηριασμού.Για κάθε αντικείμενο ορίζουμε μία αρχική τιμή εκκίνησης,μία τιμή άμεσης αγοράς καθώς και μία συγκεκριμένη χρονική στιγμή λήξης του πλειστηριασμού.

Για τη συμμετοχή του στα παραπάνω ο χρήστης θα πρέπει να προχωρήσει στη δημιουργία λογαριασμού(account) με τη συμπλήρωση μιας φόρμας στην οποία δηλώνει το username και password που επιθυμεί να χρησιμοποιήσει για την είσοδό του στην ιστοσελίδα καθώς και κάποια επιπλέον προσωπικά του στοιχεία.
Κατόπιν μπορεί να συμμετάσχει σε έναν ή περισσότερους απ΄τους πλειστηριασμούς με δύο τρόπους.Είτε με νέο ποντάρισμα μεγαλύτερο του ελάχιστου ποσού,είτε να προχωρήσει σε άμεση αγορά του αντικειμένου στην τιμή που έχει προκαθοριστεί (εκτός απ'την περίπτωση που η προσφορά κάποιου άλλου χρήστη έχει ξεπεράσει το ποσό άμεσης αγοράς,οπότε και ο πλειστηριασμός συνεχίζεται κανονικά χωρίς αυτήν τη δυνατότητα πλέον).

Όλα τα παραπάνω αφορούν τον χρήστη(user).Οι ενέργειες και οι λειτουργίες που μπορεί να εκτελέσει ο διαχειριστής(admin) της ιστοσελίδας περιγράφονται αναλυτικά στο κεφάλαιο IV που αποτελεί και το κατασκευαστικό μέρος της εργασίας.

Κλείνοντας,να αναφέρουμε τις εφαρμογές που χρησιμοποιήθηκαν και μας βοήθησαν στην υλοποίηση της συγκεκριμένης ιστοσελίδας:

- Adobe Dreamweaver
- Apache Server
- Βάση Δεδομένων MySQL
- Mail Server Mercury

ΚΕΦΑΛΑΙΟ Ι

ΘΕΩΡΗΤΙΚΟ ΜΕΡΟΣ
(ΕΙΣΑΓΩΓΗ)

ΕΙΣΑΓΩΓΗ:

ΔΥΝΑΜΙΚΕΣ ΙΣΤΟΣΕΛΙΔΕΣ

Οι *δυναμικές* ιστοσελίδες αντικαθιστούν τον HTML κώδικα με ένα σετ οδηγιών, οι οποίες χρησιμοποιούνται για την παραγωγή του HTML την ώρα που ο χρήστης κάνει αίτηση. Έτσι η ιστοσελίδα παράγεται δυναμικά, και τα βήματα που ακολουθεί είναι τα εξής:

1. Ο δημιουργός της ιστοσελίδας φτιάχνει ένα σετ οδηγιών δημιουργίας HTML και τις σώζει σε ένα αρχείο.
2. Ο χρήστης κάνει αίτηση της ιστοσελίδας στον browser του η οποία στην συνέχεια περνάει από τον browser στον Web Server.
3. Ο Web Server εντοπίζει το αρχείο με τις οδηγίες.
4. Ο Web Server ακολουθεί τις οδηγίες για να παράγει την HTML.
5. Ο Web Server στην συνέχεια στέλνει την HTML, η οποία έχει μόλις δημιουργηθεί, μέσω του δικτύου στον browser.
6. Ο browser προσπελαύνει την HTML.

Παρατηρούμε λοιπόν ότι ο κώδικας HTML που καθορίζει την ιστοσελίδα παράγεται μόνο μετά από απαίτηση του χρήστη. Υπάρχουν τεράστιες δυνατότητες σε μια εφαρμογή οι οποίες είναι αδύνατες με την χρήση μόνο HTML κώδικα. Για παράδειγμα μπορούν να αποθηκευτούν πληροφορίες όπως:

- Η ταυτότητα του χρήστη και προσωπικές εκτιμήσεις.
- Τον τύπο του browser που χρησιμοποιεί ο χρήστης.
- Πληροφορίες που παρέχονται από την αίτηση του χρήστη.
- Πληροφορίες που περιέχονται σε βάσεις δεδομένων, αρχεία κειμένου, XML αρχεία, κ.τ.λ.

Ανόμοια με τις *στατικές* ιστοσελίδες, τα δυναμικά scripts δίνουν ζωή σε ένα Web site, παρέχοντάς του χαρακτηριστικά διαλογικότητας και αμφίδρομης επικοινωνίας. Τα εγγενή αντικείμενα μιας δυναμικής εφαρμογής διευκολύνουν την επικοινωνία με τον server, την εκκίνηση, τον έλεγχο και την διαχείριση του site σαν μια συνεκτική εφαρμογή, και τον χειρισμό μεμονωμένων χρηστών κατά την διάρκεια ξεχωριστών συνόδων εργασίας τους.

Η στενή σχέση που υπάρχει μεταξύ των ASP/HTML, VBScript/JavaScript και βάσεων δεδομένων/SQL σημαίνει ότι θα πρέπει να έχετε έναν συγκεκριμένο βαθμό εξοικείωσης με όλες αυτές τις τεχνολογίες· φυσικά, εάν έχετε οποιοδήποτε υπόβαθρο στον προγραμματισμό, θα διαπιστώσετε ότι είναι πολύ εύκολο να κατανοήσετε και να χρησιμοποιήσετε τις εντολές και την σύνταξη τους σε κάθε γλώσσα/περιβάλλον. Εάν όχι, η εκμάθηση των βασικών δομών προγραμματισμού αυτών των γλωσσών όχι μόνο θα

σας βοηθήσει, αλλά θα αποδειχθεί χρήσιμη και για οποιαδήποτε άλλη γλώσσα προγραμματισμού επιθυμείτε να μάθετε.

Σύγκριση στατικών-Δυναμικών ιστοσελίδων

Οι στατικές ιστοσελίδες φέρουν αυτό το χαρακτηρισμό επειδή δεν παράγονται "κατ΄ απαίτηση". Το περιεχόμενο τους δεν αλλάζει παρά μόνο όταν ενσωματώνει νέες εκδόσεις ή αναθεωρήσεις ο υπεύθυνος του site(Webmaster). Οι ιστοσελίδες αυτές περιέχουν συνήθως τα ακόλουθα:

- Μία ενότητα κεφαλίδας(Header) η οποία, εκτός από τον τίτλο, περιέχει επίσης άλλες πληροφορίες αόρατες για τον χρήστη.
- Τον "κορμό"(Body), στον οποίο τοποθετείτε το μεγαλύτερο μέρος του περιεχομένου μιας ιστοσελίδας.
- Κώδικα JavaScript ή VBScript για την υλοποίηση δυνατοτήτων μέσω προγραμματισμού.
- Γραφικά ή πολυμέσα για την επίτευξη όσο το δυνατόν πιο ελκυστικής εμφάνισης.

Η HTML 5.0 είναι η πιο πρόσφατη έκδοση της γλώσσας(μπορείτε να βρείτε περισσότερες πληροφορίες για τις προδιαγραφές της HTML στην διεύθυνση *http://www.w3.org/MarkUp/0*).

Τα βασικά tags της HTML είναι δισυπόστατα, με ένα tag αρχής και ένα tag τέλους. Η μόνη διαφορά μεταξύ τους είναι η ύπαρξη του συμβόλου / στο tag τέλους(για παράδειγμα, τα <B> και </B> είναι τα tags αρχής και τέλους για την μορφοποίηση έντονης γραφής), θεωρήστε τα tags αρχής και τέλους σαν οριοθέτες οι οποίοι προειδοποιούν την εφαρμογή browser να ξεκινήσει την εφαρμογή έντονης γραφής στην θέση του tag αρχής και να την ολοκληρώσει στην θέση του tag τέλους.

Όμοια με τους περισσότερους άλλους κανόνες, υπάρχουν και εξαιρέσεις. Μία τέτοια είναι το tag που χρησιμοποιείται για την ενσωμάτωση εικόνων σε ιστοσελίδες(<img scr = "image.gif">),το οποίο είναι μονοδιάστατο και δεν διαθέτει tag τέλους. Για ορισμένα tags της HTML το tag τέλους είναι υποχρεωτικό, για άλλα προαιρετικό, ενώ για ορισμένα άλλα δεν υπάρχει καθόλου(και απαγορεύεται η χρήση του).

Υπό κανονικές συνθήκες χρήσης, ορισμένα tags της HTML παρέχουν στους χρήστες την δυνατότητα να επικοινωνούν με τον server.Για παράδειγμα, εάν ενσωματώσετε μια σύνδεση στην ιστοσελίδα σας, ο χρήστης-επισκέπτης θα μπορεί να κάνει click πάνω της για να ζητήσει την μεταφορά μιας διαφορετικής ιστοσελίδας από τον server ή απ΄ οποιονδήποτε Web server του κόσμου. Τα αρχεία γραφικών προσπελάζονται μέσω ενός παρόμοιου μηχανισμού, με το tag img. Στα tags δημιουργίας φορμών μπορείτε να καθορίζετε την διαδρομή και το όνομα αρχείου ενός προγράμματος ή script το οποίο θα εκτελείτε όταν ο χρήστης κάνει click στο πλήκτρο Submit για την υποβολή(αποστολή) της φόρμας.

Τα ASP εκμεταλλεύονται αυτούς τους μηχανισμούς σαν ένα μέσο για την εκτέλεση των scripts.
Έτσι λοιπόν μια στατική ιστοσελίδα HTML λειτουργεί σε έναν client browser ως εξής:

1. Ο δημιουργός της ιστοσελίδας την κατασκευάζει και την αποθηκεύει σε ένα .htm αρχείο.
2. Ο χρήστης κάνει αίτηση(Request) της ιστοσελίδας στον browser του, και η αίτηση μεταφέρεται από τον browser στον Web Server.
3. Ο Web Server εντοπίζει το αρχείο .htm .
4. Ο Web Server στην συνέχεια στέλνει την HTML μέσω του δικτύου στον browser.
5. Ο browser προσπελαύνει την HTML.

Υπάρχουν όμως *περιορισμοί* στις στατικές ιστοσελίδες. Για παράδειγμα αν θέλαμε να εμφανίσουμε στην οθόνη του χρήστη την τωρινή ώρα, θα έπρεπε να γραφτεί ένας κώδικας HTML ο οποίος κάθε φορά που προσπελάζεται η σελίδα, θα εμφανίζει την ώρα, πράγμα που δεν μπορεί να γίνει, διότι το αποτέλεσμα θα είναι μία ιστοσελίδα που θα εμφανίζει συνέχεια την ίδια ώρα.

<hl> In Web server the hour is 14:20 </hl>

Το πρόβλημα θα λυνόταν αν μπορούσε να βρεθεί ένας τρόπος να παράγεται ο κώδικας HTML μετά την αίτηση του χρήστη, δηλ. να γίνει αντικατάσταση του κώδικα HTML με ένα σετ από οδηγίες οι οποίες θα καθορίζουν και θα παράγουν τον HTML κώδικα την ώρα που ο χρήστης κάνει αίτηση, πράγμα το οποίο επιτυγχάνεται μέσω των δυναμικών ιστοσελίδων και της τεχνολογίας PHP, όπως θα δούμε παρακάτω.

Γλώσσες σεναρίων-Scripting Languages

Οι *scripting languages* είναι οι γλώσσες προγραμματισμού που αποτελούν την βάση των δυναμικών εφαρμογών. Τα HTML δεν είναι γλώσσα προγραμματισμού, για αυτό οι εντολές γράφονται σε άλλες γλώσσες. Ο όρος scripting language περιγράφει τις γλώσσες στις οποίες τα σενάρια(scripts) είναι γραμμένα.

Το πρότυπο HTML επιτρέπει την συμπερίληψη script σε οποιοδήποτε σημείο του HTML κώδικα. Συγκεκριμένα, όταν ο κώδικας HTML παράγεται, κάθε script μέσα στην σελίδα στέλνεται σε έναν *script host*, δηλαδή μια εφαρμογή που επικοινωνεί με διαφορετικές μηχανές διερμήνευσης script.Το script host στην συνέχεια οδηγεί την κατάλληλη μηχανή διερμήνευσης script(*script engine*) να ερμηνεύσει το script.Κάθε γλώσσα(scripting language) έχει έναν συγκεκριμένο script engine.Δηλαδή ένα πρόγραμμα ή script που είναι γραμμένο σε VBScript, θα σταλεί στην VBScript script engine(μηχανή διερμήνευσης VBScript).

Υπάρχουν πολλές scripting language, εκ των οποίων οι πιο γνωστές είναι η VBScript και η JavaScript. Η JavaScript ήταν η πρώτη γλώσσα εφαρμογής σε πλευρά Client. Η VBScript είναι βασισμένη στην γλώσσα προγραμματισμού Visual Basic.

Επίσης υπάρχουν δυο τρόποι με τους οποίους το script μπορεί να προσπελαστεί. Οι τρόποι αυτοί είναι σε μεριά server και σε μεριά client:

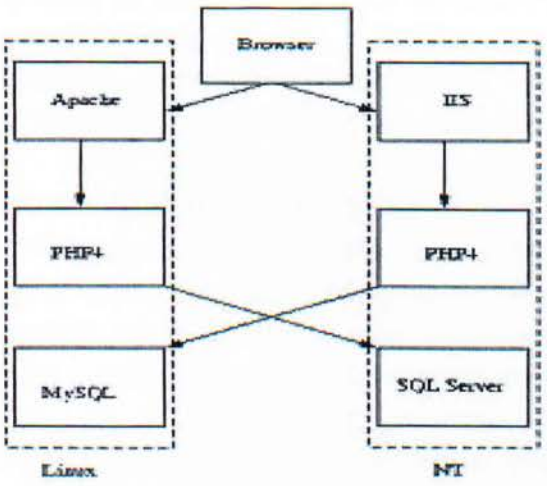
- Το script το οποίο μεταφράζεται από τον server ονομάζεται *server-side script*(σε μεριά server). Το server-side script είναι μία οδηγία η οποία προσπελάζεται από τον server, και παράγει HTML, το οποίο στην συνέχεια στέλνεται σαν μέρος του HTML Response στον browser.
- Το script το οποίο μεταφράζεται από τον browser ονομάζεται *client-side script*(σε μεριά client). Το client-side script είναι μια οδηγία η οποία δεν προσπελάζεται από τον server, αλλά στέλνεται σαν μέρος του HTML Response στον browser, και στην συνέχεια προσπελάζεται από τον browser.

Οι δυο δημοφιλέστεροι Web Servers

Αυτοί είναι οι:

- ***IIS(Internet Information Server)***

  Ο Internet Information Server είναι ένα προϊόν της Microsoft και μπορεί να εκτελεστεί σε όλες τις πλατφόρμες που υποστηρίζουν NT 4.0. Αυτό σημαίνει ότι μπορούν να χρησιμοποιηθούν μηχανήματα που βασίζονται στους επεξεργαστές x86, DEC Alpha AXP, MIPS ή Power PC.
  Η εγκατάσταση αυτού του server είναι πολύ εύκολη και υποστηρίζει υπηρεσίες web, ftp και gopher οι οποίες μπορούν να ενεργοποιηθούν ή να απενεργοποιηθούν πολύ εύκολα.
  Ο IIS είναι ολοκληρωμένος με το σύστημα server που χρησιμοποιούν τα Windows, γεγονός που του επιτρέπει να χρησιμοποιεί τον ίδιο κατάλογο χρηστών και έτσι να παρέχει υψηλή ασφάλεια, ελέγχοντας με αυτόν τον τρόπο τα δικαιώματα που έχει κάθε χρήστης σε εφαρμογές internet ή intranet.Υποστηρίζει SLL και παρέχει μια σειρά από λειτουργίες που αυξάνουν την απόδοση του και τον καθιστούν ικανό να ανταποκριθεί σε οποιαδήποτε απαίτηση για server.
  Το δυνατότερο σημείο του IIS είναι η υποστήριξη που παρέχει στην εύκολη ανάπτυξη δυναμικών ιστοσελίδων στο web αφού είναι ο server που υποστηρίζει το περιβάλλον ανάπτυξης εφαρμογών στο web: ASP(active server pages).



- ***Apache***

  Ο Apache είναι σήμερα ο πιο δημοφιλής server στον κόσμο και υπάρχουν τρείς λόγοι για αυτό:

  1. είναι δωρεάν
  2. είναι συμβατός με πληθώρα λειτουργικών συστημάτων όπως σε Unix, Linux και σε OS/2
  3. είναι πάρα πολύ γρήγορος

Αρκεί απλώς να αναφέρουμε ότι το Artificial Intelligent Lab του Πανεπιστημίου M.I.T, το οποίο δέχεται πάνω από ένα εκατομμύριο κλήσεις την ημέρα κατά μέσο όρο, χρησιμοποιεί τον apache σαν server.

Το μόνο μειονέκτημα αυτού του server είναι ότι δεν παρέχει αξιόλογα εργαλεία διαχείρισης και έτσι είναι απαραίτητη η βαθιά γνώση του λειτουργικού συστήματος που χρησιμοποιείται καθώς και της γλώσσας προγραμματισμού C.

Επικοινωνία μεταξύ Βάσεων Δεδομένων & Web

Μία λύση για εφαρμογές οι οποίες θα επιτρέψουν την διασύνδεση εφαρμογών του web με βάσεις δεδομένων και θα επιτρέψουν την δυναμική αλληλεπίδραση μεταξύ client και server, αποτελούν οι τεχνολογίες CGI και ISAPI.

CGI (common gateway interface)

Το CGI είναι ένα πρότυπο, το οποίο επιτρέπει την αμφίδρομη ροή πληροφοριών μεταξύ browser και server με την χρήση ενός εκτελέσιμου προγράμματος(gateway) το οποίο υπάρχει στον HTTP-Server και καλείται από αυτόν σαν αποτέλεσμα κάποιων ενεργειών του χρήστη. Έτσι ένα gateway σε έναν SQL-Server θα μπορούσε είτε να μας δώσει πληροφορίες για την τιμή κάποιου προϊόντος, είτε να μας εμφανίσει τα αποτελέσματα κάποιας αναζήτησης.

Χαρακτηριστικό παράδειγμα μιας τέτοιας CGI εφαρμογής είναι οι μηχανές αναζήτησης πληροφοριών(Google,Yahoo, AltaVista κ.τ.λ.)στις οποίες το gateway κάνει μια αναζήτηση στην βάση δεδομένων, η οποία αντιπροσωπεύει την γνώση της αντίστοιχης μηχανής.

Δυστυχώς το μοντέλο του CGI παρουσιάζει ένα μεγάλο μειονέκτημα. Το μοντέλο αυτό μπορεί να παρομοιαστεί με ένα μεγάλο πολυκατάστημα στο οποίο υπάρχει μόνο ένα ταμείο για την εξυπηρέτηση όλων των πελατών του. Τέτοιου είδους εφαρμογές δεν μπορούν να εξυπηρετήσουν μεγάλο αριθμό ταυτόχρονων αιτήσεων. Ο μέγιστος αριθμός εξυπηρέτησης ταυτόχρονων αιτήσεων, με χρήση της τεχνολογίας CGI, κυμαίνεται μεταξύ 50 και 100 ενώ παρατηρείται σημαντική επιβράδυνση όταν απαιτείται η εξυπηρέτηση περισσότερων από 100 ταυτόχρονων αιτήσεων.

Η αιτία αυτού του μειονεκτήματος έγκειται στο ότι οι εφαρμογές CGI δεν αποτελούν ενσωματωμένες λειτουργίες του server και έτσι απαιτείται να φορτωθεί και να εκτελεστεί το αρχείο .exe για κάθε αίτηση ξεχωριστά με αποτέλεσμα να καλυφτούν πολύ γρήγορα οι πόροι του server.

ISAPI (Internet Server Application Interface)

Η αντιμετώπιση των προβλημάτων που παρουσίαζε η τεχνολογία CGI έγινε με χρήση της τεχνολογίας ISAPI(Internet Server Application Interface).Στην τεχνολογία αυτή τα CGI προγράμματα έχουν αντικατασταθεί από αρχεία dll(Dynamic Link Library) τα οποία είναι ενσωματωμένες λειτουργίες του server και φορτώνονται κάθε φορά μαζί του. Έτσι αυτή η τεχνολογία παρουσιάζει γρηγορότερη εξυπηρέτηση των αιτήσεων σε σχέση με την τεχνολογία CGI.

Η τεχνολογία ISAPI αν και δίνει λύσεις σε ορισμένα προβλήματα, δεν μπόρεσε να ανταποκριθεί στις μεγάλες απαιτήσεις που υπήρχαν για να δημιουργηθούν πιο ολοκληρωμένες δυναμικές σελίδες στο web, διότι τα dll προγράμματα, αν και αποτελούν ενσωματωμένες λειτουργίες του server, δεν παύουν να είναι προγράμματα γραμμένα σε γλώσσες όπως C++, Visual Basic, Perl κ.τ.λ., γεγονός που απαιτεί αφενός καλή γνώση των τελευταίων, αφετέρου βαθιές γνώσεις των πρωτοκόλλων του internet.

Θα έπρεπε να υπήρχαν ενσωματωμένα αντικείμενα του server, με την απλή χρήση των οποίων θα ήταν δυνατή η απόκτηση των πληροφοριών που εισάγει ο χρήστης όταν συμπληρώσει και στείλει μια φόρμα ή αντικείμενα, τα οποία θα επέτρεπαν άμεσα την διασύνδεση με μια βάση δεδομένων και θα έδιναν έτσι την δυνατότητα διαχειρίσεως της βάσης από πολλά μεταξύ τους απομακρυσμένα σημεία. Τις παραπάνω απαιτήσεις και όχι μόνο ήρθαν με την σειρά τους να εκπληρώσουν τα *active server pages(ASP)* και κατ' επέκταση οι δυναμικές εφαρμογές, όπως είναι η PHP.

Το Περιβάλλον Ανάπτυξης

Το περιβάλλον στο οποίο γίνεται η ανάπτυξη και συντήρηση των web sites αποκαλείται περιβάλλον ανάπτυξης. Ο όρος *περιβάλλον ανάπτυξης(development environment)* αναφέρεται συγκεκριμένα στην πλατφόρμα, στην διαδικασία και στα εργαλεία που χρησιμοποιούνται για την σχεδίαση, την κατασκευή και τη συντήρηση όλων των αρχείων/αντικειμένων/δεδομένων που αποτελούν τα επιμέρους συστατικά ενός σύγχρονου Web site.

Η πολυπλοκότητα των web sites καλύπτει μία πολύ μεγάλη κλίμακα – από μερικές ιστοσελίδες διασυνδεδεμένες μεταξύ τους, έως πλήρεις εφαρμογές web με μια πληθώρα τύπων αρχείων, αντικειμένων και άλλων συστατικών τα οποία αλληλεπιδρούν και συνεργάζονται αρμονικά.

Γενικά, τα περιβάλλοντα ανάπτυξης μπορούν να διαχωριστούν σε δύο ευρείες κατηγορίες: "*χειροκίνητα*" και "*αυτοματοποιημένα*".

"Χειροκίνητη" Μεθοδολογία Ανάπτυξης

Ορισμένοι προγραμματιστές προτιμούν να γράφουν όλο τον κώδικα μόνοι τους. Πράγματι, είναι εφικτή η χειροκίνητη ανάπτυξη ακόμη και ενός πολύπλοκου web site με εργαλεία τόσο απλά, όσο το Notepad ή το WordPad Ωστόσο, εργαλεία όπως αυτά που προαναφέραμε χρησιμοποιούνται συνήθως για σύντομες διορθωτικές εργασίες υπαρχόντων αρχείων, και όχι για την κατασκευή ενός ολόκληρου web site εκ του μηδενός.

"Αυτοματοποιημένη" Μεθοδολογία Ανάπτυξης

Σε αντίθεση με τους απλούς συντάκτες κειμένου, τα αυτοματοποιημένα περιβάλλοντα ανάπτυξης παρέχουν στον χρήστη ένα πλούσιο σύνολο εργαλείων τα οποία καθιστούν ευκολότερες, γρηγορότερες, λιγότερο επιρρεπείς σε λάθη και συχνά πιο διαισθητικές τις κοινές εργασίες ανάπτυξης HTML και ASP κώδικα. Ορισμένα εργαλεία όπως το FrontPage ή το Joomla παρέχουν υψηλότερο βαθμό αυτοματοποίησης σε σύγκριση με άλλα, με αντίτιμο αυξημένες παρασκηνιακές διεργασίες, οι οποίες δεν γίνονται πάντα εύκολα κατανοητές. Με την υπερβολική αυτοματοποίηση μπορεί να βρεθείτε κυριολεκτικά στο έλεος ενός προγράμματος, όταν τα πράγματα δεν δείχνουν να λειτουργούν με τον αναμενόμενο τρόπο και θα θέλατε να πάρετε στα χέρια σας τον έλεγχο.

Ανάπτυξη On-line Εφαρμογών

Οι on-line εφαρμογές(ή web εφαρμογές) είναι web sites τα οποία παρέχουν πολλή περισσότερη λειτουργικότητα από τις στατικές ιστοσελίδες ή ένα συμβατικό web site.Σ'ένα συμβατικό Web site μπορείτε να συμπληρώσετε μία φόρμα και με το πάτημα του πλήκτρου "Υποβολή" ή "Submit" τα περιεχόμενα της φόρμας στέλνονται σε ένα CGI script και υφίστανται επεξεργασία, μετασχηματιζόμενα σε ένα μήνυμα ηλεκτρονικού

ταχυδρομείου ή πιθανώς σε ένα αρχείο κειμένου στον server.Οτιδήποτε άλλο δεν είναι τίποτα περισσότερο απο κείμενο ή γραφικά τα οποία εμφανίζονται στο παράθυρο μιας εφαρμογής browser, διανθισμένα με συνδέσεις υπερκειμένου.

Οι online εφαρμογές μοιάζουν με τα συμβατικά προγράμματα που χρησιμοποιείται, όπως π.χ. το Word, το Excel κ.α., εκτός από το γεγονός ότι δεν "τρέχουν" στον δικό σας υπολογιστή, αλλά online. Για παράδειγμα, εάν θέλατε να δημιουργήσετε μία εφαρμογή για την online συμπλήρωση φορολογικών δηλώσεων σ'ένα web site, θα έπρεπε να ακολουθήσετε έναν κύκλο ανάπτυξης παρόμοιο με αυτόν που απαιτείται για μία αντίστοιχη συμβατική εφαρμογή.

Το Default Web Site που δημιουργείται κατά την εγκατάσταση του IIS στο σύστημά σας είναι η θέση απο την οποία ξεκινούν οι εφαρμογές. Υποθέτοντας ότι γνωρίζετε ποιές λειτουργίες θέλετε να ενσωματώσετε στην εφαρμογή σας, μπορείτε να γράψετε τον κώδικα για την υλοποιησή τους με το ASP και την VBScript(ή οποιαδήποτε άλλη γλώσσα script προτιμάτε).Για λόγους ταχύτητας και εύκολης αναβάθμισης σε ευρύτερη κλίμακα, ίσως αποφασίσετε να υλοποιήσετε συγκεκριμένα τμήματα του προγράμματός σας σαν εκτελέσιμα συστατικά(ActiveX) χρησιμοποιώντας γλώσσες όπως είναι η C++ ή η Visual Basic.

Εναλλακτικά, θα μπορούσατε να αγοράσετε προκατασκευασμένα συστατικά τα οποία διαθέτουν την λειτουργικότητα που χρειάζεστε.Κατά πάσα πιθανότητα θα δημιουργείτε επίσης βάσεις δεδομένων για την αποθήκευση των δεδομένων των συνδρομητών ή των πελατών σας στο παρασκήνιο, καθώς και πίνακες με πληροφορίες οι οποίες δεν θα αλλάζουν συχνά, όπως για παράδειγμα οι πίνακες με τους φορολογικούς συντελεστές που αναφερθήκαμε προηγουμένως και οι φόρμες.

Τέλος, θα συνενώνετε τις λειτουργίες, τα συστατικά, τις βάσεις δεδομένων και τις συνόδους των χρηστών μεταξύ τους χρησιμοποιώντας αντικείμενα ASP και την γλώσσα script(π.χ.PHP) που προτιμάτε. Οι λειτουργίες του ASP διαχειρίζονται την επικοινωνία με τον χρήστη και καλούν τα άλλα συστατικά της εφαρμογής μέσα στο πεδίο δράσης που υπαγορεύουν οι παράμετροι της συνόδου/εφαρμογής που καθορίζετε.

Client/Server Εφαρμογές

Είναι εύκολο να σκέφτεται κανείς τους servers σαν υπολογιστές και όχι σαν υπηρεσίες υλοποιούμενες μέσω λογισμικού οι οποίες "τρέχουν" στους υπολογιστές. Αυτό οφείλεται στο γεγονός ότι συνήθως χρησιμοποιούμε τον όρο *server(διακομιστής)* για να χαρακτηρίσουμε ένα μηχάνημα και τον όρο *υπηρεσία* για να χαρακτηρίσουμε το λογισμικό.

Σε κάθε περίπτωση, η ευθύνη του server είναι η παροχή υπηρεσιών σε άλλους υπολογιστές – για παράδειγμα, η παράδοση περιεχομένου απο το Διαδίκτυο. Ένα *σύστημα-πελάτης(client)* ζητά υπηρεσίες από έναν server. Τυπικοί "πελάτες" είναι οι εφαρμογές Web browser και τα προγράμματα ηλεκτρονικού ταχυδρομείου. Ωστόσο, η διαχωριστική γραμμή μεταξύ των servers και των clients γίνεται ολοένα και πιο ασαφής, με τους servers να αναλαμβάνουν πλέον και ρόλους client ή το αντίστροφο.

Ο όρος client/server(πελάτης/διακομιστής) περιγράφει εύστοχα την αλληλεπίδραση που συμβαίνει μεταξύ μιας εφαρμογής Web browser και ενός server. Ο "πελάτης", σ'αυτή την περίπτωση η εφαρμογή Web browser, στέλνει μία αίτηση για μια ιστοσελίδα ή οποιονδήποτε άλλο πόρο ενός web site και ο server παρέχει ένα αντίγραφο της ιστοσελίδας(και των σχετιζόμενων με αυτή αρχείων) στον πελάτη. Προφανώς, ο server είναι κατασκευασμένος ώστε να έχει τη δυνατότητα να χειρίζεται ταυτόχρονα πολλαπλές αιτήσεις, προερχόμενες από πολλαπλούς "πελάτες".

Ένα άλλο, σχετιζόμενο με την αρχιτεκτονική client/server μοντέλο αλληλεπίδρασης είναι οι *εφαρμογές πολλαπλών βαθμίδων(multi-tiered applications)*. Οι εφαρμογές αυτές εμπεριέχουν πολλαπλά επίπεδα αλλήλεπίδρασης.
Για παράδειγμα, υποθέστε ότι χρησιμοποιείται την client εφαρμογή σας για να ζητήσετε δεδομένα από έναν server και ο server, με την σειρά του, πρέπει να ζητήσει λειτουργίες επεξεργασίας και δεδομένα από μία πηγή δεδομένων ή έναν ενδιάμεσο "διερμηνευτή" δεδομένων, ή ένα πρόγραμμα μετατροπής. Αυτοί οι "ενδιάμεσοι" αναφέρονται συνήθως σαν *μεσαία βαθμίδα(middle tier)* σε μια εφαρμογή πολλαπλών βαθμίδων και μπορούν να υπάρχουν αρκετά επίπεδα ενδιάμεσων μεταξύ του τελικού χρήστη και των δεδομένων ή της λειτουργικότητας που ζητά. Στην ιδανική περίπτωση οι ενδιάμεσες βαθμίδες θα πρέπει να είναι "διαφανείς", έτσι ώστε οι λειτουργίες της εφαρμογής να εκτελούνται με ομοιόμορφο τρόπο, χρησιμοποιώντας το σύστημα επικοινωνίας που κατανοεί η client εφαρμογή.

Οι εφαρμογές πολλαπλών βαθμίδων μπορούν να αποτελούνται από συστατικά διασκορπισμένα σε οποιαδήποτε online θέση ή σε ένα δίκτυο.Αυτό σημαίνει ότι οι εφαρμογές αυτές μπορούν να αναπτύσσονται ευκολότερα σε ευρεία κλίμακα και μπορούν να είναι πιο αξιόπιστες, ισχυρές και πλούσιες σε δεδομένα.

Σκεφτείται πόσα θα μπορούσατε να πετύχετε ακόμη και σ'έναν μεμονωμένο τομέα της αγοράς, εάν είχατε την δυνατότητα να συνδυάσετε την πληθώρα των πόρων που είναι διαθέσιμοι στο Internet σε μία γιγαντιαίων διαστάσεων, μονολιθική εφαρμογή...

Τα Web Sites σαν κανάλια επικοινωνίας

Από φυσικής απόψεως το Internet είναι απλώς ένα δίκτυο δικτύων, το οποίο χρησιμοποιεί κυρίως τα πρωτόκολλα TCP/IP για την επικοινωνία μεταξύ των διαφόρων κόμβων του.Ένας μηχανικός συστημάτων μπορεί να σας πεί για τα διάφορα επίπεδα(layers) ή πρωτόκολλα(protocols) που απαιτούνται για την επικοινωνία μεταξύ των κόμβων ενός δικτύου, πως διαχωρίζονται τα δεδομένα σε πακέτα(packets) τα οποία ακολουθούν μία ποικιλία διαδρομών από έναν υπολογιστή στον επόμενο διαμέσου του τοπικού δικτύου ή του Internet, και πολλά άλλα ενδιαφέροντα θέματα.

Είναι εύκολο να θεωρήσει κανείς το web σαν ένα υποσύνολο του Internet(πράγμα το οποίο ισχύει) και την επικοινωνία μέσω του web σαν μία ακόμη μέθοδο προβολής και διαφήμισης, όπως η τηλεόραση ή το ραδιόφωνο. Ωστόσο, η κατασκευή ενός επιτυχημένου web site – ή ακριβέστερα μιάς εφαρμογής web, εμπλέκει πολλούς άλλους παράγοντες, πέρα από τις απλές μηχανικές διεργασίες(συγγραφή του κώδικα, δημοσίευση κ.τ.λ.). Όχι μόνο θα πρέπει να ενσωματώσετε άφθονα και χρήσιμα χαρακτηριστικά διαλογικότητας, άλλα θα πρέπει επίσης να κατασκευάσετε το site σας με τέτοιο τρόπο, έτσι ώστε ο χρήστης να αισθάνεται άνετα σ'αυτό.

Πιθανότατα θυμάστε την εποχή κατά την οποία ο όρος *φιλικό για τον χρήστη* σήμαινε τα πάντα. Ένα web site πρέπει να είναι τόσο φιλικό για τον χρήστη, έτσι ώστε ακόμή και ο πλέον αρχάριος να μπορεί να κατανοήσει αμέσως τον τρόπο με τον οποίο θα το χρησιμοποιήσει. Ταυτόχρονα, η φιλικότητα του δεν θα πρέπει να απωθεί τους πιο προχωρημένους χρήστες, στόχος όμως ο οποίος στην πράξη δεν είναι καθόλου εύκολος να επιτευχθεί.

Καθιέρωση Στόχων

Έχετε αποφασίσει ότι το web είναι το ενδεδειγμένο μέσο για την κοινοποίηση ενός συγκεκριμένου συνόλου πληροφοριών ή για την παροχή μιας συγκεκριμένης υπηρεσίας με διαλογικά χαρακτηριστικά, σε πραγματικό χρόνο.
Στο σημείο στο οποίο βρίσκεστε τώρα καλείστε να λάβετε μία πληθώρα άλλων αποφάσεων οι οποίες σχετίζονται με την διαμόρφωση του site, έτσι ώστε να μπορεί να εκτελεί τις λειτουργίες που απαιτούνται για την υλοποίηση των στόχων του. Ωστόσο, εάν αυτοί οι στόχοι είναι ασαφείς ή απροσδιόριστοι, η πορεία προς την υλοποίησή τους θα είναι πολύ πιο δύσκολη, κοπιαστική και με συχνές οπισθοχωρήσεις. Είναι προτιμότερο να ορίσετε από την αρχή συγκεκριμένους στόχους, οι οποίοι θα μπορούν να σας καθοδηγήσουν προς την σωστή κατεύθυνση για την ανάπτυξη και υλοποίηση του site σας.
Για παράδειγμα, εάν ένας από τους βασικούς στόχους του site σας είναι να δίνει στους χρήστες την δυνατότητα να κάνουν online αγορές, θα πρέπει να λάβετε υπόψη τα ακόλουθα:

- Η αγορά στην οποία απευθύνεται(αγορά - στόχος) και δημογραφικά στοιχεία για τους επισκέπτες που πρέπει να προσελκύει.
- Προβλεπόμενες πωλήσεις και πλήθος επισκεπτών(πιθανών αγοραστών) στο site.
- Ενοποίηση του online με το offline σύστημα διαχείρησης αποθεμάτων.
- Η διαδικασία αγοράς προϊόντων από το site.
- Ασφάλεια, προστασία της ιδιωτικότητας και τεχνική υποστήριξη των πελατών.

Παρακάτω θα αναλύσουμε διεξοδικά έναν προς ένα τους ανωτέρω στόχους:

Αγορά-Στόχος και δημογραφικά στοιχεία

Το να γνωρίζετε την αγορά στην οποία απευθύνεστε δεν είναι σημαντικό μόνο για τους καθαρά επιχειρησιακούς σκοπούς αλλά παίζει επίσης σημαντικό ρόλο στην δομή του web site που δημιουργείται. Εάν το τμήμα του πληθυσμού στο οποίο απευθύνεστε δεν έχει υπολογιστή ή δεν διαθέτει σύνδεση στο Internet, δεν πρόκειται να πετύχετε σημαντικές πωλήσεις μέσω του web site σας. Ανάλογα, εάν απευθύνεστε σε στελέχη επιχειρήσεων ή σε ακαδημαϊκούς οι οποίοι κατά κανόνα έχουν υψηλής ταχύτητας πρόσβαση στο Internet, μπορείται να χρησιμοποιήσετε πιο "εξειδικευμένο" περιεχόμενο(πιο "απαιτητικές" μορφές αρχείων) για να επικοινωνήσετε μαζί τους. Με το ίδιο κριτήριο, εάν το μεγαλύτερο μέρος του ακροτηρίου σας πρόκειται να συνδέεται στο site σας μέσω κινητών τηλεφώνων ή ψηφιακών προσωπικών βοηθών(Personal Digital Assistants ή αλλιώς PDA's), θα πρέπει να εξετάσετε την χρήση μιας ειδικής μετατροπής έτσι ώστε το site σας να είναι συμβατό μ'αυτές τις μορφές σύνδεσης.

Προβλεπόμενες πωλήσεις και πλήθος επισκεπτών

Το πλήθος των επισκεπτών(hits) που δέχεται το site σας είναι σημαντικό, επειδή έχει άμεση επίδραση στην διαθεσιμότητά του. Εάν χειρίζεστε εκατομμύρια επισκέπτες κάθε μέρα για να πετύχετε τους στόχους που έχουν τεθεί για το site σας, πιθανότατα θα χρειαστείτε επίσης πολλαπλούς servers και άφθονο εύρος ζώνης(bandwidth), έτσι ώστε το site σας να μπορεί να είναι άμεσα διαθέσιμο για την πλειονότητα των χρηστών(για να μην αναφέρουμε την αναγκαιότητα χρήσης ενός ποιοτικού συστήματος DBMS,καθώς και την σωστή διαμόρφωση του λειτουργικού συστήματος,των web servers καθώς και των εφαρμογών που τρέχουν στο παρασκήνιο).

Το διαθέσιμο εύρος ζώνης πρέπει να είναι επαρκές ώστε να υποστηρίζει το μέγιστο αναμενόμενο πλήθος επισκεπτών στο site σας. Ένα άλλο θέμα το οποίο σχετίζεται με το εύρος ζώνης είναι το ποσό που χρησιμοποιείται.Οι περισσότερες εταιρείες που παρέχουν υπηρεσίες "φιλοξενίας" web sites(ή αλλιώς *Web Hosting*) θέτουν κάποιο όριο όσον αφορά στην χρήση του εύρους ζώνης, ακόμη και όταν ισχυρίζονται ότι παρέχουν απεριόριστο εύρος

ζώνης στους πελάτες τους.Συνήθως η υπέρβαση αυτού του ορίου σημαίνει επιπλέον χρεώσεις ή τουλάχιστον επαναδιαπραγμάτευση του συμβολαίου.

Ενοποίηση του online με το offline σύστημα διαχείρισης αποθεμάτων

Έχουν γίνει γνωστές αρκετές περιπτώσεις εταιρειών οι οποίες δεν φρόντισαν για την ενοποίηση του online με το offline σύστημα διαχείρισης των αποθεμάτων τους. Φυσικά, οι πελάτες αναμένουν ότι εάν βρούν ένα προϊόν και το αγοράσουν online, θα μπορέσουν και να το αποκτήσουν. Αλλά εάν δεν υπάρχει συντονισμός των βάσεων δεδομένων που υποστηρίζουν το online και το offline σύστημα λήψης παραγγελιών και διαχείρισης αποθεμάτων, ένα προϊόν το οποίο δείχνει ότι online είναι διαθέσιμο, μπορεί υλικά(offline) να μην είναι. Τα καταστήματα λιανικής θα πρέπει να διασφαλίσουν ότι υπάρχει συνεχής σύνδεση μεταξύ των αποθεμάτων που εμφανίζονται online και των αγορών που πραγματοποιούνται offline. Πολλά καταστήματα συνδέουν τα πάντα στο online σύστημα, σε μια προσπάθεια να αποφύγουν τέτοιου είδους προβλήματα.

Η διαδικασία αγοράς

Η διαδικασία αγοράς θα πρέπει να είναι γρήγορη, εύκολη και ευχάριστη – όχι πολύπλοκη, κουραστική και βαρετή. Οι καταναλωτές θα πρέπει να μπορούν να βρίσκουν και να προσθέτουν στο καλάθι αγορών τους τα προϊόντα που αναζητούν, με άμεση ενημέρωση για το κόστος και τα έξοδα αποστολής, θα πρέπει επίσης να υπάρχει ένας εύκολος τρόπος για την αφαίρεση προϊόντων από το καλάθι αγορών τους, σε περίπτωση που αλλάξουν γνώμη. Τα βήματα για την πληρωμή με πιστωτική κάρτα θα πρέπει να καθορίζονται με σαφήνεια, ενώ θα πρέπει επίσης να παρέχονται συνδέσεις προς τον τομέα εξυπηρέτησης πελατών, τους όρους και τις συνθήκες χρήσης, την πολιτική προστασίας της ιδιωτικότητας καθώς και πληροφορίες ασφαλείας.

Ασφάλεια,προστασία της ιδιωτικότητας & τεχνική υποστήριξη των πελατών

Αν και δεν θα θέλετε να κοινοποιήσετε σε όλο τον κόσμο τα μέτρα ασφάλειας που χρησιμοποιείται, θα πρέπει τουλάχιστον να κάνετε μία γενική αναφορά στο θέμα της ασφάλειας για να καθησυχάσετε τους πελάτες σας. Επίσης, δεδομένου ότι το ενδιαφέρον για την προστασία των προσωπικών δεδομένων αυξάνεται καθημερινά, μία δήλωση για την πολιτική προστασίας της ιδιωτικότητας που ακολουθεί η εταιρεία είναι πολύ σημαντική για το site σας.

Περαιτέρω, εάν οι πελάτες σας ενδιαφέρονται για την υποστήριξη που μπορείτε να τους παρέχετε είτε πρίν είτε μετά την πώληση, θα πρέπει να συμπεριλάβετε στοιχεία για την επικοινωνία τους με το τμήμα τεχνικής υποστήριξης της εταιρείας.

Φιλοξενία του Web site

Αν και το καλύτερο σενάριο όταν δημιουργείται εφαρμογές web είναι να έχετε απόλυτο έλεγχο στους servers, αυτό συχνά είναι ανέφικτο˙ κυρίως οι δημιουργοί web sites που δουλεύουν σαν ελεύθεροι επαγγελματίες, έχουν ελάχιστες επιλογές όσον αφορά αυτό το θέμα.Ορισμένες φορές θα είστε υποχρεωμένοι να χρησιμοποιήσετε τις υπηρεσίες φιλοξενίας μιας συγκεκριμένης εταιρείας παροχής υπηρεσιών Web, ενώ σε άλλες περιπτώσεις η εταιρεία για την οποία αναπτύσσετε το site μπορεί να μην έχει την δυνατότητα να αγοράσει ή να μισθώσει τους δικούς της servers. Η γνώση των διαφόρων υπηρεσιών φιλοξενίας web sites και του τρόπου με τον οποίο μπορείτε να τις εκμεταλλευτείτε καλύτερα είναι πολύ σημαντική για την επαγγελματική επιτυχία σας.
Η πολιτική, το hardware και το λογισμικό που χρησιμοποιεί μια εταιρεία φιλοξενίας web sites είναι στοιχεία τα οποία έχουν σημαντική επίδραση στις εφαρμογές που δημιουργείτε.

Αφού διεξάγετε μία έρευνα αγοράς, το καλύτερο που έχετε να κάνετε είναι να επιλέξετε μία εταιρεία η οποία ειδικεύεται στην φιλοξενία(στέγαση) web sites και όχι μία εταιρεία παροχής γενικών υπηρεσιών Internet η οποία διαθέτει και υπηρεσίες φιλοξενίας ιστοσελίδων. Οι εταιρείες παροχής Internet που παρέχουν φιλοξενία σε web sites σαν συμπληρωματική υπηρεσία μπορεί να μην παρέχουν την ίδια *ποιότητα εξυπηρέτησης* και το ίδιο *επίπεδο δυνατοτήτων* με μία ειδικευμένη εταιρεία παροχής υπηρεσιών web.

Επιλογή Τύπου Σύνδεσης

Τα βασικά θέματα τα οποία πρέπει να εξετάσετε όταν επιλέγετε έναν τύπο σύνδεσης ή όταν αξιολογείτε μία εταιρεία φιλοξενίας ιστοσελίδων, είναι το πλήθος των συνδέσεων στο Internet και η ταχύτητά τους. Σχεδόν κάθε εταιρεία φιλοξενίας ιστοσελίδων ή εταιρεία παροχής Internet αγοράζει συνδέσεις Internet από κάποια μεγαλύτερη εταιρεία(upstream provider), εκτός ίσως από τις πολύ μεγάλες εταιρείες.

Πολλές εταιρείες δηλώνουν ότι είναι *multi-homed*, πράγμα το οποίο σημαίνει ότι διαθέτουν συνδέσεις προς περισσότερες από μία μεγάλες ΕΠΙ. Αυτό είναι σημαντικό, επειδή ακόμη και οι μεγαλύτερες ΕΠΙ μπορεί να τεθούν εκτός λειτουργίας για ώρες, ημέρες ή ακόμη και εβδομάδες, αν και το τελευταίο είναι μάλλον σπάνιο εώς απίθανο! Οι καλύτερες εταιρείες φιλοξενίας ιστοσελίδων διαθέτουν τουλάχιστον τρείς εώς επτά συνδέσεις προς μεγάλες ΕΠΙ. Το διαθέσιμο εύρος ζώνης εξαρτάται απο τον τύπο των συνδέσεων που διαθέτει μία εταιρεία με το Internet. Το εύρος ζώνης που θα είναι διαθέσιμο σ'εσάς εξαρτάται από το πλήθος των sites που ανταγωνίζονται για το ίδιο εύρος ζώνης(είτε στον δικό σας Η/Υ είτε σε όλους τους Η/Υ της συστοιχίας στην οποία φιλοξενείται το site σας).

Για να υπολογίσετε το απαιτούμενο εύρος ζώνης για την υποστήριξη ενός δεδομένου αριθμού χρηστών, μπορείτε να ακολουθήσετε μία διαδικασία παρόμοια με την ακόλουθη:

1. Υπολογίστε το μέγεθος των μεταδιδόμενων αρχείων. Για HTML σελίδες, πολλαπλασιάστε το πλήθος των χαρακτήρων ανά γραμμή(περίπου 80) επί το πλήθος των bits ανά χαρακτήρα(8), επί το πλήθος των γραμμών ανά σελίδα(γύρω στις 66), επί το πλήθος των σελίδων του εγγράφου και επί έναν συντελεστή πρόσθετου φόρτου της τάξης του 1.5. Συνεπώς, για ένα HTML έγγραφο 10 σελίδων γεμάτο με κώδικα, το μέγεθος είναι: 10 σελίδες επί 66 γραμμές ανά σελίδα(=660), επί 80 χαρακτήρες ανά γραμμή(=52.800), επί 8 bits ανά χαρακτήρα(=422.400), επί τον συντελεστή 1.5. Τελικώς, το σύνολο των μεταδιδόμενων bits είναι 633.600 ή περίπου 2/3 ενός εκατομμυρίου bits.

2. Διαιρέστε το πλήθος των μεταδιδόμενων bits με το διαθέσιμο εύρος ζώνης για να εξακριβώσετε πόσος χρόνος θα απαιτείται για την μετάδοση του εγγράφου. Για ένα έγγραφο με το μέγεθος που υπολογίσαμε παραπάνω, ο χρόνος μετάδοσης του θα είναι περίπου 0.4 δευτερόλεπτα εάν χρησιμοποιείται μία γραμμή T-1(1.54 εκατομμύρια bits ανά δευτερόλεπτο). Αντίστοιχα, διαιρώντας την ταχύτητα μετάδοσης με το μέγεθος αρχείου του εγγράφου μπορείτε να βρείτε το πλήθος των εγγράφων που μπορούν να μεταδίδονται ανά δευτερόλεπτο(περίπου 2.4 στο δικό μας παράδειγμα).

   Ο παραπάνω υπολογισμός δεν είναι αντιπροσωπευτικός για τις τυπικές HTML σελίδες, επειδή οι περισσότερες από αυτές δεν είναι γεμάτες με κείμενο σε κάθε στήλη κάθε γραμμής. Ωστόσο, οι σελίδες περιλαμβάνουν συνήθως αρχεία γραφικών, τα οποία πρέπει να μεταδοθούν ξεχωριστά. Στους υπολογισμούς σας για το απαιτούμενο εύρος ζώνης θα πρέπει να συνυπολογίσετε όλα τα αρχεία τα οποία απαρτίζουν μία σελίδα. Επιπρόσθετα, θα πρέπει να θυμάστε ότι σχεδόν κανένας χρήστης δεν διαθέτει σύνδεση επιπέδου T-1.

3. Υπολογίστε το πλήθος των χρηστών που μπορείτε να υποστηρίξετε με το διαθέσιμο εύρος ζώνης. Το πλήθος των χρηστών υπολογίζεται συνήθως πολλαπλασιάζοντας το πλήθος των μεταδιδόμενων σελίδων ανά μέρα επί το τυπικό μέγεθος αρχείου(συμπεριλαμβανομένων όλων των αρχείων που σχετίζονται με κάθε σελίδα), σε kilobytes ή megabytes. Εάν διαθέτετε εύρος ζώνης 1.54 εκατομμύρια bits(δηλαδή μία γραμμή T-1 αποκλειστικά για τον web server σας), μπορείτε να μεταδίδετε(συνυπολογίζοντας τον συντελεστή πρόσθετου φόρτου) περίπου 11.088.000.000 bytes ανά ημέρα ή λίγο παραπάνω από 10

Gb. Εάν κάθε σελίδα έχει μέγεθος 30K, μπορείτε να υποστηρίξετε περίπου 333.000 αιτήσεις αποστολής σελίδων ανά ημέρα.

4. Εάν οι αιτήσεις αποστολής σελίδων ήταν ομοιόμορφα κατανεμημένες μεταξύ των 24 ωρών της ημέρας, κανένας χρήστης δεν θα αντιμετώπιζε πρόβλημα μειωμένης ταχύτητας. Ωστόσο, συνήθως οι χρήστες "συνωστίζονται" σ'ένα site κατά τις ώρες αιχμής. Συνεπώς, θα πρέπει επίσης να υπολογίσετε το πλήθος των χρηστών που μπορεί να υποστηρίξει ταυτόχρονα το site σας, διατηρώντας αποδεκτούς χρόνους απόκρισης.
   Αυτό γίνεται υπολογίζοντας το πλήθος των αιτήσεων αποστολής σελίδων που μπορούν να διεκπεραιωθούν μέσα σε ένα αυθαίρετα καθορισμένο χρονικό διάστημα. *Για παράδειγμα,* για να βρείτε πόσοι χρήστες θα λαμβάνουν τις σελίδες που ζήτησαν μέσα σε τρία το πολύ δευτερόλεπτα από την στιγμή που έκαναν click, θα πρέπει να εκτελέσετε τον ακόλουθο υπολογισμό:

   Ταχύτητα μετάδοσης ανά δευτερόλεπτο, επί αυθαίρετα καθορισμένο χρονικό διάστημα, διά μέσο μέγεθος σελίδας(μην ξεχάσετε να συμπεριλάβετε στον υπολογισμό τον συντελεστή πρόσθετου φόρτου).

   Για παράδειγμα, εάν κάθε αίτηση αποστολής σελίδων απαιτεί την μετάδοση αρχείων μεγέθους 3.75KB κατά μέσο όρο και χρησιμοποιείτε μία γραμμή T-1, μπορείτε να εξυπηρετήσετε περίπου 100 χρήστες μέσα σε ένα χρονικό διάστημα τρίων δευτερολέπτων.

Σχεδίαση μιας Εφαρμογής για ένα Web Site

Η εργασία αυτή θα σας παρουσιάσει την αρχική φάση σχεδίασης μίας online εφαρμογής. Συγκεκριμένα καλύπτει τον καθορισμό των στόχων του web site και της εφαρμογής, τον έλεγχο ευχρηστίας, την δημιουργία πλάνων, την αντίστροφη μηχανική, τα θέματα που σχετίζονται με το hardware, το λογισμικό και το εύρος ζώνης, τις απαιτήσεις στον τομέα της ασφάλειας, τις προβλέψεις και τις αξιολογήσεις.

- Καθορισμός στόχων.Ζητήστε από τον πελάτη να σας παρέχει μία αναλυτική περιγραφή των βασικών λειτουργιών που θα πρέπει να εκτελεί το web site, συμπεριλαμβανομένων και οποιονδήποτε μεθόδων "μέτρησης απόδοσης"(για παράδειγμα, το πλήθος των επισκέψεων, των συναλλαγών ή των πωλήσεων ανά ημέρα κ.τ.λ.).Εάν οι στόχοι είναι δύσκολο να μετρηθούν(ή να ποσοτικοποιηθούν γενικότερα) ή ορίζονται με μη-απτούς όρους(π.χ. αύξηση της ικανοποίησης των πελατών), φροντίστε να συμφωνήσετε εκ των προτέρων με τον πελάτη σας σε κάποια αμοιβαία αποδεκτά κριτήρια μέτρησης.

Είναι πολύ σημαντικό να μπορείτε να δείξετε ότι το έργο σας έχει υλοποιηθεί και λειτουργεί σύμφωνα με αυτά που ζήτησε ο πελάτης.

- Καθορίστε επακριβώς τις λειτουργίες και τα χαρακτηριστικά που θα πρέπει να έχει το web site για την επίτευξη των στόχων που προσδιορίστηκαν στο προηγούμενο βήμα. Για παράδειγμα, εάν πρόκειται να υπάρχει μια λειτουργία αναζήτησης προϊόντων, προσδιορίστε ποιά θα είναι τα διαθέσιμα κριτήρια βάσει των οποίων θα μπορεί να εκτελεί την αναζήτηση ο χρήστης και ποιά πεδία πρέπει να συμπεριληφθούν στην βάση δεδομένων για να είναι δυνατές αυτές οι αναζητήσεις. Η αντίστροφη μηχανική είναι ένα πολύ καλό εργαλείο το οποίο μπορείτε να χρησιμοποιήσετε σ'αυτό το στάδιο.

- Καθορίστε πιθανά μοτίβα αναζήτησης τα οποία μπορεί να χρησιμοποιούν οι χρήστες για την ανεύρεση των δεδομένων που τους ενδιαφέρουν, καθώς και άλλα χαρακτηριστικά τα οποία θέλετε να επιδεικνύει το site και εκτελέστε τους απαιτούμενους ελέγχους ευχρηστίας για να διασφαλίσετε ότι οι "διαδρομές" αναζήτησης που χρησιμοποιείτε εσείς είναι "συμβατές" με την συμπεριφορά ενός τυπικού χρήστη.

- Καταλήξτε σε μία εκτίμηση για τα μεγέθη αρχείων, τους χρόνους μεταφοράς, το πλήθος των χρηστών που θα μπορούν να επισκέπτονται ταυτόχρονα το site και το απαιτούμενο εύρος ζώνης. Αξιολογήστε τις διαθέσιμες πλατφόρμες hardware/λογισμικού και εξακριβώστε τι απαιτείται για την υποστήριξη του web site ώστε να λειτουργεί στο επίπεδο που θεωρεί ικανοποιητικό ο πελάτης σας.

- Αναπτύξτε σε αρχικό επίπεδο την αρχιτεκτονική του site και ζητήστε απο τον πελάτη σας να εγκρίνει εγγράφως το έργο. Μην συμπεριλάβετε μόνο την διάταξη και την εμφάνιση του site, αντίθετα συμπεριλάβετε ένα αναλυτικό πλάνο για την χρήση των πόρων, εκτιμήσεις για το κόστος και τον χρόνο και τις εξαρτήσεις μεταξύ των εργασιών. Η χρήση ενός καλού προγράμματος διαχείρισης έργων μπορεί να σας βοηθήσει σημαντικά σ'αυτή τη φάση. Εάν υπάρχουν ειδικές απαιτήσεις στον τομέα της ασφάλειας(σχετιζόμενες με τη διεξαγωγή συναλλαγών ηλεκτρονικού εμπορίου ή με την διακίνηση ευαίσθητων επιχειρησιακών δεδομένων), συμπεριλάβετε και αυτές τις απαιτήσεις στην αρχιτεκτονική.

- Αναθέστε εργασίες στους συνεργάτες σας και ξεκινήστε την ανάπτυξη όλων των φάσεων του έργου σύμφωνα με το χρονοδιάγραμμα στο οποίο έχετε καταλήξει. Καθώς εξελίσσεται το έργο, επιλέξτε συγκεκριμένα "ορόσημα" στα οποία θα παρουσιάζετε την πρόοδο σας στον πελάτη, ζητώντας του αναλυτικό σχολιασμό.

Στο χώρο του Internet οι πελάτες ζητούν συνήθως πολλές αλλαγές, οπότε προετοιμαστείτε για την υλοποίηση ταχύτατων αλλαγών στην αρχιτεκτονική σας καθώς το έργο σας οδεύει προς την ολοκληρωσή του.

- Αφού ολοκληρωθεί το έργο, επανεξετάστε το σε συνεργασία με τον πελάτη και άλλους χρήστες, σε μία προσπάθεια να εντοπίσετε οποιαδήποτε σφάλματα ή αβλεψίες(ή αλλαγές που θα έπρεπε να γίνουν). Κάντε τις τελικές αλλαγές και κατόπιν φορτώστε το web site στον server "παραγωγής", με το κανονικό όνομα domain.

- Αναλύετε προσεκτικά τις γνώμες που λαμβάνετε από τους χρήστες κατά την αρχική φάση τη δημοσίευσης του site και κάντε οποιεσδήποτε προσαρμογές απαιτούνται, ανάλογα με τις πληροφορίες που θα λάβετε από αυτούς.

Η Διάταξη ενός Web Site

Αρχικά θα πρέπει να καθορίσετε την δομή και την διάταξη του site σας, ανάλογα με τους στόχους σας. Συχνά, το στάδιο αυτό παραβλέπεται από πολλούς δημιουργούς, στην βιασύνη τους να προλάβουν να κάνουν κάτι(οτιδήποτε) πρίν τους κλέψει κάποιος άλλος την ιδέα. Στην πραγματικότητα όμως οι περισσότερες ιδέες για web sites έχουν ήδη υλοποιηθεί, οπότε δεν υπάρχει τίποτα για να κλαπεί. *Αυτό που μετράει δεν είναι τόσο πολύ η ιδέα, όσο η καλή εκτέλεσή της.*

Σε τελική ανάλυση οι χρήστες εκτιμούν και είναι διατεθειμένοι να επιστρέψουν σ'ένα site μόνο εάν αυτό έχει οποιοδήποτε ή όλα τα παρακάτω χαρακτηριστικά:

-Ευκολία

Για να εξακριβώσετε εάν η δομή του site σας είναι εύχρηστη, εξετάστε τις υπάρχουσες διαδικασίες που χρησιμοποιούν οι άνθρωποι για να κάνουν την ίδια εργασία και βεβαιωθείτε ότι είναι πραγματικά πιο εύκολο να γίνει online. Για παράδειγμα, εάν έχετε υψηλής ταχύτητας σύνδεση στο web και ξέρετε τι κάνετε, η αγορά οποιουδήποτε προϊόντος online μπορεί να σας φανεί σαν το πιο βολικό πράγμα στον κόσμο. Αλλά για κάποιον ο οποίος έχει χαμηλής ταχύτητας σύνδεση στο Internet ή περιορισμένη εμπειρία στον online κόσμο, οι αγορές μέσω του web μπορεί να αποδειχθούν μπελάς, ενώ μία βόλτα στο τοπικό πολυκατάστημα θα ήταν πού πιο εύκολη.
Κάνοντας το Web site σας όσο το δυνατόν πιο εύκολο στην χρήση μπορείτε να πείσετε ακόμη και τους πιο "απρόθυμους" επισκέπτες.

-Κόστος

Το Internet και το Web έχουν σαφή επίδραση στις τιμές, ευτυχώς προς όφελος των καταναλωτών. Ο "ανταγωνισμός" από το Internet αποθαρρύνει τις πάσης φύσεως εταιρείες από το να πουλήσουν σε πολύ υψηλή τιμή.Αυτό εξηγεί εν μέρει την δημοτικότητα των sites που ασχολούνται με δημοπρασίες, τόσο στους κύκλους των αγοραστών, όσο και στους κύκλους των πωλητών. Το web μας παρέχει έναν αξιόπιστο μηχανισμό για την εύρεση των τιμών διαφόρων αγαθών και υπηρεσιών και εάν το site σας δίνει στους άλλους την εντύπωση ότι συνεισφέρει στην μείωση του κόστους, τότε θα είναι δημοφιλές.

-Εντυπωσιασμός

Το στοιχείο του εντυπωσιασμού χαρακτηρίζει συνήθως τα sites που διαθέτουν νέες λειτουργίες ή παρέχουν στους χρήστες πρωτοποριακές δυνατότητες(πιθανώς αμφιλεγόμενες σε ορισμένες περιπτώσεις). Ο εντυπωσιασμός μπορεί να διαρκέσει κάμποσο, αλλά συνήθως σβήνει γρήγορα. Ο εντυπωσιασμός είναι κάτι καλό, μακροπρόθεσμα όμως είναι προτιμότερο να δώσετε έμφαση στα υπόλοιπα χαρακτηριστικά που αναφέρουμε εδώ. Το site σας θα πρέπει να συνεχίσει να είναι δημοφιλές και παραγωγικό(με την επιχειρησιακή έννοια του όρου) ακόμη και μετά από τα 15 λεπτά δημοσιότητας που του αναλογούν!

-Πρωτοτυπία

Υπάρχουν εκατομμύρια web sites και είναι εύκολο να υποθέσει κανείς ότι όλες οι πραγματικά καλές ιδέες έχουν υλοποιηθεί με τον ένα ή τον άλλο τρόπο. Στην πραγματικότητα όμως θα υπάρχει πάντα χώρος για την ανάπτυξη πρωτότυπων sites από καινοτόμους σχεδιαστές. Κάθε νέα τεχνολογία και κάθε αλλαγή στη στάση και στην συμπεριφορά του κοινού είναι μία ευκαιρία για νέα, δημιουργικά web sites.

-Σταθερότητα & Αξιοπιστία

Όταν χαθεί ο εντυπωσιασμός, όταν θα αντιγράψουν την ιδέα σας εκατοντάδες άλλοι, όταν όλοι μειώνουν τις τιμές τους κατά 5% περισσότερο από εσάς, τί απομένει για να σας στηρίξει; Εάν βγήκατε πρώτοι και με διαφήμιση, μπορεί να έχετε ήδη ένα μερίδιο της αγοράς. Αλλά πως θα το αυξήσετε; Πώς θα το διατηρήσετε ενάντια στον ανταγωνισμό; Υποθέτοντας ότι κάνατε καλή δουλειά με την διαφήμιση, την ανάπτυξη των προϊόντων σας και την κάλυψη όλων των βασικών διαδικασιών της επιχείρησή σας, το κλειδί για την συνέχιση της επιτυχίας σας είναι η σταθερότητα και η αξιοπιστία. Βεβαιωθείτε ότι το site σας είναι πάντα λειτουργικό και ασφαλές. Βεβαιωθείτε επίσης ότι η εταιρεία που σας παρέχει υπηρεσίες web

δεν υπερφορτώνει τους servers της και δεν ξεχνάει να σας ειδοποιήσει για τα προγραμματισμένα διαστήματα "εκτός λειτουργίας".
Αυτό ισχύει ακόμη περισσότερο εάν στεγάζετε το site σας σε δικά σας συστήματα. Αφιερώστε επιπλέον χρόνο σ'αυτούς τους τομείς, ακόμη και αν αυτό σημαίνει ότι θα καθυστερήσει το ξεκίνημα σας στην αγορά. Μέχρι ενός ορίου, το καλό marketing μπορεί να αντισταθμίσει το καθυστερημένο ξεκίνημα μιας επιχειρησιακής δραστηριότητας, αλλά θα πρέπει να έχετε υπόψη ότι είναι σημαντικό να βρίσκεστε στις πρώτες θέσεις.

Στο σημείο αυτό θα πρέπει να έχετε σαφή άποψη σχετικά με το τι θέλετε να επιτύχετε με το site σας. Αφιερώστε λίγο χρόνο για να γράψετε μία απλή αλλά επικεντρωμένη αναφορά, η οποία θα περιγράφει συνοπτικά πως πρόκειται να επιτύχει αυτούς τους στόχους το web site που δημιουργείτε. Αργότερα, όταν θα φτάσετε στην φάση του προϋπολογισμού, δώστε ιδιαίτερη προσοχή στο αναμενόμενο κέρδος της επένδυσης. Να έχετε υπόψη ότι κάποια στιγμή ίσως κληθείτε να αιτιολογήσετε γιατί ξοδεύτηκαν χιλιάδες ευρώ σ'αυτές τις – ωραίες κατά τα άλλα – λειτουργίες που υλοποιήσατε με το ASP.

Η Δομή ενός Web Site

Η δομή ενός τυπικού web site μπορεί να απεικονιστεί με γραφικό τρόπο, σαν ένας χάρτης ο οποίος υποδεικνύει ποιές σελίδες συνδέονται με ποιές άλλες σελίδες. Χρησιμοποιώντας κάτι ανάλογο μ'ένα εικονογραφημένο σενάριο, μπορείτε να δημιουργήσετε έναν απλό χάρτη για το site σας σε 15 λεπτά, με χαρτί και μολύβι.

Ωστόσο η μέθοδος αυτή δεν είναι πλέον τόσο χρήσιμη όσο ήταν στο παρελθόν, κυρίως για web sites τα οποία στην πραγματικότητα είναι εφαρμογές. Χρησιμοποιήστε τη μέθοδο του εικονογραφημένου σεναρίου για τις στατικές ιστοσελίδες(αν και στο μέλλον οι στατικές ιστοσελίδες δεν θα είναι πλέον τόσο κοινές) και ακολουθήστε την "παραδοσιακή" μέθοδο ανάπτυξης λογισμικού(με το διάγραμμα ροής) για τις βασιζόμενες σε κώδικα σελίδες. Στην εφαρμογή σας μπορεί να υπάρχουν "διαδρομές" και "στροφές" οι οποίες σπανίως γίνονται ορατές, αλλά είναι απαραίτητες· είναι χρησιμότερο να ξέρετε κάτω απο ποιές περιπτώσεις θα εμφανιστούν, παρά να σχεδιάσετε ένα πλαίσιο από το οποίο θα διέρχονται αρκετές εκατοντάδες γραμμές σε ένα δισδιάστατο σχέδιο.

-Πλοήγηση

Η πλοήγηση – δηλαδή οι μετακινήσεις των χρηστών σε όλη την έκταση ενός web site – είναι ένα θέμα ζωτικής σημασίας για οποιοδήποτε site. Εάν οι χρήστες δε μπορούν να βρούν αυτό που αναζητούν, είναι πολύ απλό να κάνουν click για να μεταβούν σε οποιαδήποτε από τα μυριάδες άλλα sites που είναι διαθέσιμα στο web. Αυτό δεν σημαίνει ότι είστε υποχρεωμένοι να χρησιμοποιήσετε την ίδια διάταξη που χρησιμοποιούν όλοι οι άλλοι,

σημαίνει απλά ότι θα πρέπει να αφιερώσετε λίγο χρόνο για να διασφαλίσετε ότι η διάταξή σας μπορεί να χρησιμοποιηθεί εύκολα απ' οποιονδήποτε και πιθανώς να ελέγξετε την ευχρηστία και την αποτελεσματικότητά της ζητώντας την βοήθεια λίγων φίλων σας. Εάν ένα άτομο το οποίο δεν γνωίζει το site σας δεν μπορεί να κινηθεί αποτελεσματικά σ'αυτό, είναι προφανές ότι η διάταξη του site σας δεν είναι όσο διαισθητική ή εύχρηστη θα έπρεπε.

Να έχετε υπόψη ότι πολλοί οργανισμοί για τους οποίους η ροή της κυκλοφορίας είναι σημαντική, καθώς και πολλές επιχειρήσεις πληρώνουν αδρές αμοιβές σε ειδικευμένους συμβούλους για να βρούν ποιό είναι το βέλτιστο μοτίβο κυκλοφορίας ή η βέλτιστη διάταξη ενός π.χ. καταστήματος, που θα βοηθήσει στην επίτευξη των στόχων τους. Θα μπορούσατε να χρησιμοποιήσετε παρόμοιες τεχνικές ελέγχου ευχρηστίας για το δικό σας web site. Εάν δεν γνωρίζετε τον τρόπο, θα μπορούσατε να προσλάβετε έναν εξωτερικό σύμβουλο ο οποίος ειδικεύεται στον έλεγχο ευχρηστίας των web sites και στην προσαρμογή τους ώστε να είναι πιο φιλικά για τον χρήστη.

Από την άποψη του ASP, η χρήση των cookies μπορεί να σας δώσει μία ιδέα σχετικά με τα μοτίβα κυκλοφορίας των χρηστών (click-stream) καθώς διασχίζουν το site σας. Σε συνδυασμό με τα στατιστικά επισκέψεων(hint statistics), τα δεδομένα αυτά είναι ένα πολύτιμο εργαλείο για να εντοπίσετε τα πιθανά προβλήματα στην σχεδίαση του site σας, καθώς επίσης και για να βρείτε τρόπους με τους οποίους θα ενθαρρύνετε τις πωλήσεις και θα κάνετε ευκολότερη τη ζωή των χρηστών. Και, καλά θα κάνετε να πάρετε στα σοβαρά τις απόψεις των χρηστών. Σε κάθε χρήστη ο οποίος σας κάνει την χάρη να σας γνωστοποιήσει την άποψή του, αντιστοιχούν πιθανώς αρκετοί άλλοι χρήστες με το ίδιο παράπονο, οι οποίοι απλά δεν μπαίνουν στον κόπο να σας το πούν.

-*Γραφικά*

Ένα καλοντυμένο άτομο προσελκύει πάνω του την προσοχή και απαιτεί σεβασμό· το ίδιο ισχύει για το "ντύσιμο"(βλέπε γραφικά) ενός Web site. Τα γραφικά μπορούν να έχουν διαφορετική σημασία ανάλογα με το περιεχόμενο του site, αλλά γενικά θα πρέπει να είναι εύκολα στην μεταφορά και ελκυστικά(αν και όχι κατ'ανάγκη ευχάριστα). Επίσης, τα γραφικά δεν θα πρέπει να αποδυναμώνουν το μήνυμα που προσπαθεί να μεταδώσει το site σας.

Θα πρέπει να έχετε υπόψη ότι το ταλέντο και η εμπειρία ενός επαγγελματία σχεδιαστή(*Web Designer*) είναι απαραίτητα κατά την δημιουργία πολλών web sites. Αν και οποιοσδήποτε μπορεί να δημιουργήσει μία ιστοσελίδα, οι καλά εκπαιδευμένοι, έμπειροι και ταλαντούχοι σχεδιαστές ιστοσελίδων έχουν την δυνατότητα να δημιουργούν εντυπωσιακά γραφικά και διατάξεις σελίδων(φυσικά, με την προϋπόθεση ότι έχουν εκπαιδευτεί κατάλληλα για την δημιουργία τέτοιων projects στο Web).

Σχεδίαση της Δομής ενός Web Site

Παρακάτω παρουσιάζεται *βήμα προς βήμα* η διαδικασία σχεδίασης της δομής ενός νέου Web Site. Για τα γραφικά και την διάταξη των σελίδων ίσως χρειαστεί να καταφύγετε στις υπηρεσίες ενός επαγγελματία σχεδιαστή ιστοσελίδων, κυρίως εάν δεν έχετε το ταλέντο ή τις γνώσεις για να τα κάνετε μόνοι σας.

-Βήμα πρός βήμα

1. Δημιουργείστε ένα απλό διάγραμμα ροής για τις σελίδες που θα βλέπουν οι χρήστες και τις διαχειριστικές σελίδες του site. Σ'ένα τυπικό εμπορικό site περιλαμβάνονται συνήθως σελίδες όπως οι:

   - About (λίγα λόγια για την εταιρεία),
   - Contact (πληροφορίες επαφής),
   - Order (φόρμα παραγγελίας),
   - Terms (όροι),
   - Privacy (πολιτική προστασίας της ιδιωτικότητας) και
   - Catalog (κατάλογος προϊόντων).

   Σχεδιάστε ένα πλαίσιο για την αναπαράσταση κάθε σελίδας στο διάγραμμα ροής και χρησιμοποιείστε γραμμές μεταξύ των πλαισίων για να υποδείξετε τις συνδέσεις μεταξύ των σελίδων.

2. Βεβαιωθείτε ότι περιλαμβάνετε κενά πεδία ή μηχανισμούς για όλα τα δεδομένα που θέλετε να συλλέγετε από τους χρήστες ή για τις πληροφορίες που θέλετε να εμφανίζονται. Βεβαιωθείτε ότι οι πελάτες σας συμφωνούν με την προτεινόμενη διάταξη των σελίδων, καθώς και ότι τα δεδομένα που τους παρέχουν είναι επαρκή για την κάλυψη των επιχειρησιακών τους αναγκών. Βεβαιωθείτε επίσης ότι τα δεδομένα είναι επαρκή για την υποστήριξη των υπολογισμών και της επεξεργασίας που απαιτούν τα scripts ή τα συστατικά που θα χρησιμοποιήσετε.

3. Εάν το μέγεθος ή η φύση του έργου το απαιτεί, διεξάγετε ελέγχους ευχρηστίας χρησιμοποιώντας περισσότερες από μία διατάξεις. Βεβαιωθείτε ότι οι χρήστες μπορούν να βρούν αυτό που αναζητούν και έχουν την δυνατότητα να κινούνται διαισθητικά σε όλη την έκταση του site. Γι'αυτή την εργασία ο έλεγχος ευχρηστίας μπορεί να είναι απλός, θα μπορούσατε να παρουσιάσετε απλώς τις ιδέες σας σε ένα άτομο το οποίο δεν έχει καμία επαφή με το έργο και δεν γνωρίζει τίποτα για την διαδικασία δημιουργίας ενός Web site. Ζητήστε του να σας επισημάνει οποιαδήποτε πράγματα δεν κατανοεί κατά την διάρκεια που προσπαθεί να κινηθεί στο site.

4. Σε μία λίστα, καταγράψτε όλες τις σελίδες που πρέπει να δημιουργηθούν καθώς και τις γλώσσες που θα χρησιμοποιηθούν. Για την παρασκηνιακή επεξεργασία καταγράψτε τα scripts που θα απαιτηθούν, δίνοντας ένα περιγραφικό όνομα στο καθένα. Η λίστα σας θα γίνεται πιο περιεκτική καθώς θα κατανοείτε όλο και περισσότερο πως λειτουργούν τα ASP΄ προς το παρόν όμως η δημιουργία μιας απλής λίστας με μερικές περιγραφές είναι απόλυτα εντάξει.

5. Συγκεντρώστε σε έναν φάκελο το διάγραμμα ροής, τις διατάξεις των σελίδων, τις πληροφορίες που θα εμφανίζει ή θα συλλέγει το site, τις απαιτήσεις για το πλήθος των σελίδων και τις εκδόσεις τους σε διαφορετικές γλώσσες, καθώς και τις ανάγκες του Web site σε scripts και εξετάστε όλα αυτά τα στοιχεία σε συνεργασία με τον πελάτη. Το υλικό αυτό θα αποτελέσει τη βάση για τη λήψη των τελικών αποφάσεων και την κατασκευή του Web site.

Στο ΚΕΦΑΛΑΙΟ ΙΙ που ακολουθεί παρουσιάζεται η θεωρητική ανάλυση της γλώσσας προγραμματισμού PHP που θα χρησιμοποιηθεί για την υλοποίηση του κατασκευαστικού μέρους της ιστοσελίδας μας.

ΚΕΦΑΛΑΙΟ ΙΙ

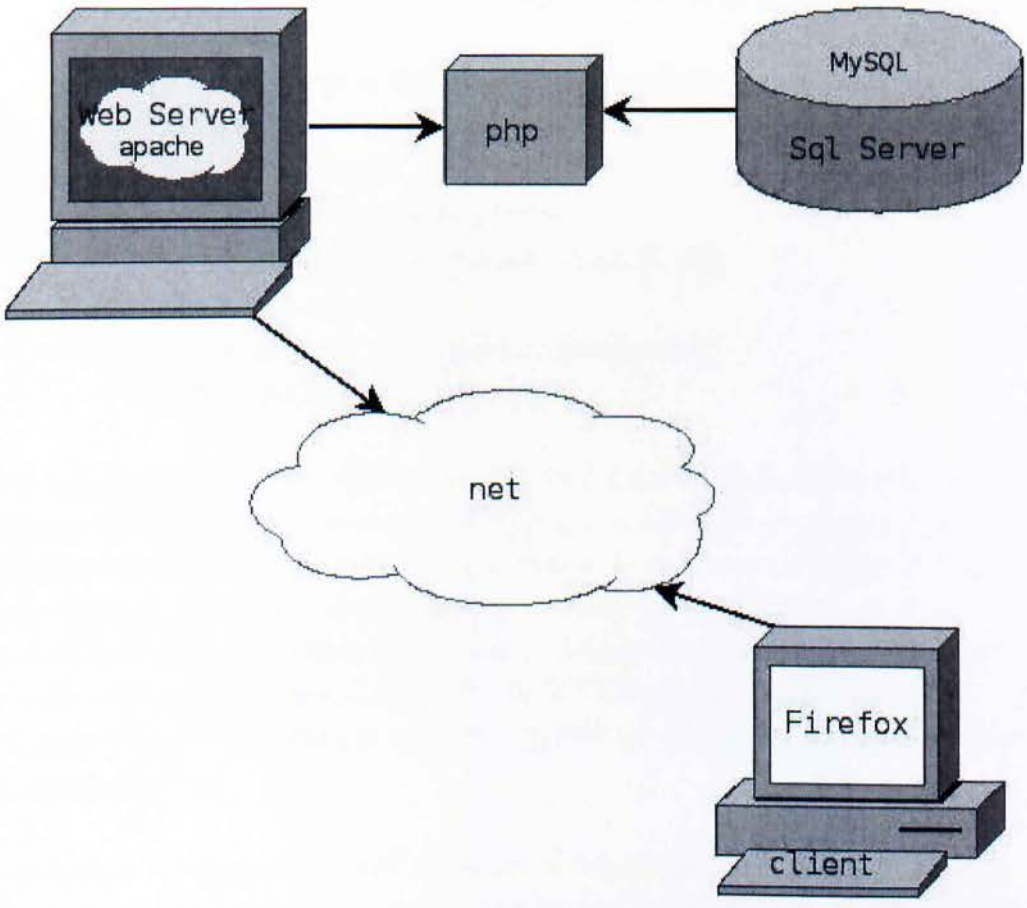
Η Γλώσσα Προγραμματισμού PHP

ΕΙΣΑΓΩΓΗ:

ΛΙΓΑ ΛΟΓΙΑ ΓΙΑ ΤΗΝ PHP

Η PHP, της οποίας τα αρχικά αντιπροσωπεύουν την λέξη "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 ενσωματοθεί στην HTML.

Είναι μια γλώσσα script από την πλευρά του διακομιστή(server), σχδιασμένη ειδικά για το Web. Μέσα σε μία HTML σελίδα μπορούμε να ενσωματώσουμε PHP κώδικα, που θα εκτελείται κάθε φορά που επισκεπτόμαστε την σελίδα. Ο PHP κώδικας μεταφράζεται στο Web server και δημιουργεί τον κώδικα HTML. Αυτή η διαδικασία φαίνεται και στο παρακάτω σχήμα, όπου ο PHP κώδικας λέει στον server να στείλει τα κατάλληλα δεδομένα στον Web Browser σε μορφή html.



Η σύνταξη της PHP παίρνει στοιχεία από τις γλώσσες προγραμματισμού C, Java και Perl και είναι εύκολη στη μάθηση. Ο κύριως στόχος της γλώσσας είναι να επιτρέπει σε web developers να γράφουν δυναμικά παραγόμενες ιστοσελίδες(WebPages) γρήγορα.

Η PHP δημιουργήθηκε το 1994 και ήταν αρχικά η δουλειά ενός ατόμου, του Rasmus Lerdorf. Στη συνέχεια, υιοθετήθηκε και από άλλα ταλαντούχα άτομα. Σήμερα, χρησιμοποιείται σχεδόν σε 20 εκατομμύρια τομείς παγκοσμίως σύμφωνα με τα στατιστικά που βλέπουμε στην σελίδα: *http://www.php.net/usage.php*
Η PHP είναι ένα προϊόν ανοικτού κώδικα, που σημαίνει ότι έχουμε πρόσβαση στον κώδικα προέλευσης και μπορούμε να τον χρησιμοποιήσουμε, να τον αλλάξουμε και να τον αναδιανείμουμε, χωρίς χρέωση.

Εκτός των άλλων, η PHP έχει και τα παρακάτω πλεονεκτήματα:

- Υψηλή απόδοση
- Διασυνδέσεις με πολλά διαφορετικά συστήματα βάσεων δεδομένων
- Ενσωματωμένες βιβλιοθήκες για πολλές συνηθισμένες Web διαδικασίες
- Χαμηλό κόστος
- Ευκολία μάθησης και χρήσης
- Δυνατή αντικειμενοστραφή υποστήριξη
- Μεταφερσιμότητα
- Διαθεσιμότητα του κώδικα προέλευσης
- Διαθεσιμότητα υποστήριξης

Αυτό που διαχωρίζει την PHP από κάτι σαν client-side JavaScript είναι ότι ο κώδικας εκτελείται στον server(εξυπηρετητή). Δηλαδή αν είχατε ένα οποιοδήποτε script στον server σας, ο client θα έπαιρνε τα αποτελέσματα της εκτέλεσης αυτού του script, χωρίς να υπάρχει κανένας τρόπος να καταλάβει τι κώδικας υπάρχει απο κάτω. Μπορείται ακόμη να ρυθμίσετε τον web server σας να χειρίζεται όλα τα HTML αρχεία σας με την PHP και τότε πραγματικά δεν υπάρχει τρόπος ο χρήστης να καταλάβει τι έχετε κάτω από το μανίκι σας!

Το καλύτερο πράγμα στην PHP είναι ότι είναι εξαιρετικά απλή για έναν νεοφερμένο αλλά προσφέρει παράλληλα πολλά προηγμένα χαρακτηριστικά για έναν επαγγελματία προγραμματιστή.

Μην τρομάζετε όταν διαβάζετε την μακροσκελή λίστα με τα χαρακτηριστικά της PHP. Μπορείτε να εξοικειωθείτε μέσα σε πολύ λίγο χρόνο και να αρχίσετε να γράφετε απλά scripts μέσα σε λίγες ώρες!

Η PHP επικεντρώνεται κυρίως στο server-side scripting, έτσι μπορείτε να ασχοληθείτε με οτιδήποτε άλλο, μπορεί ένα CGI πρόγραμμα 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 τρείς κύριοι τομείς που χρησιμοποιείται ένα PHP script.

- *Server-side scripting*. Αυτό είναι το πιο παραδοσιακό και το κύριο πεδίο για την PHP. Χρειαζόμαστε τρία πράγματα για να εκτελεστεί αυτό. Τον *PHP μεταγλωττιστή*(parser – CGI ή server module), ένα *Web Server*(εξυπηρετητή σελίδων) και ένα *Web Browser*(φυλλομετρητή). Πρέπει να τρέξουμε τον web server, με μία συνδεδεμένη εγκατάσταση της PHP. Μπορούμε να προσπελάσουμε τα αποτελέσματα του PHP προγράμματος με ένα web browser, βλέποντας την σελίδα PHP μέσα από τον server.

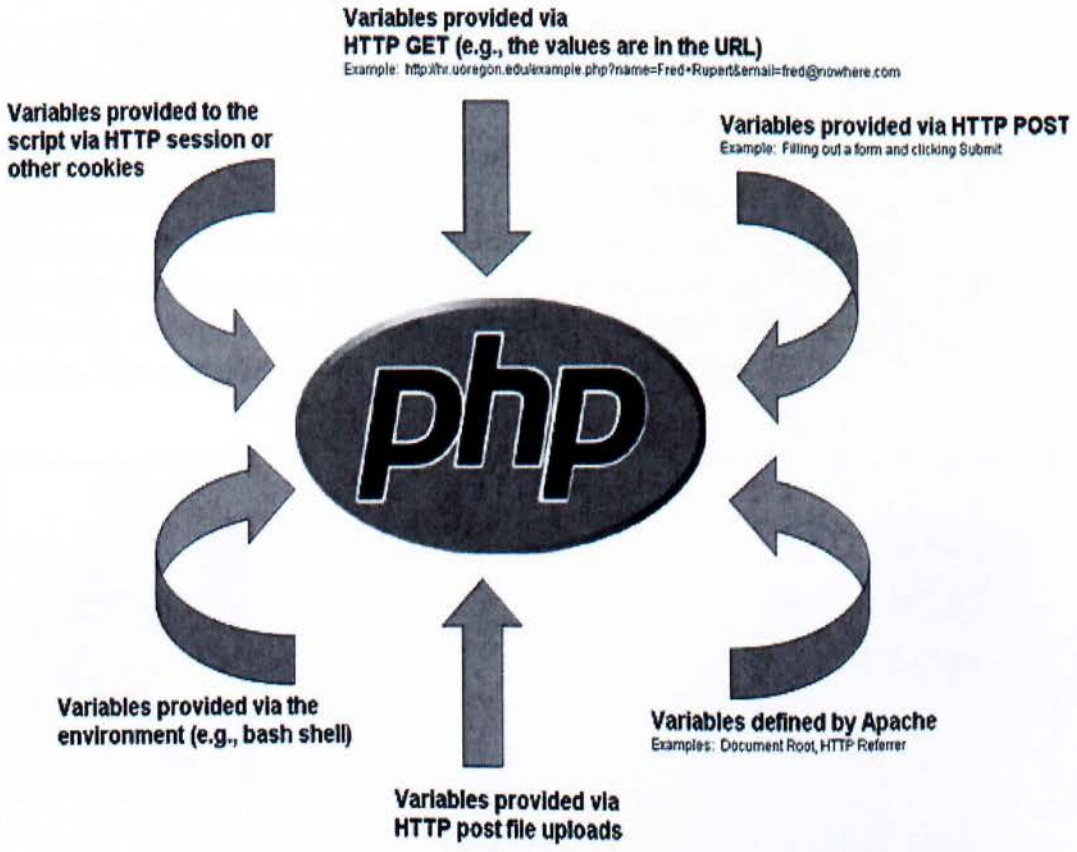
- *Command line scripting*. Μπορούμε να φτιάξουμε ένα PHP script για να το τρέχουμε χωρίς server ή browser. Χρειαζόμαστε μόνο τον PHP μεταγλωττιστή. Αυτός ο τύπος είναι ιδανικός για script που εκτελούνται συχνά με τη χρήση της cron(σε Unix ή Linux) ή με τον Task Scheduler(στα Windows). Αυτά τα scripts 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 παραθυρικές εφαρμογές, αλλά αν ξέρουμε PHP πολύ καλά μπορούμε να χρησιμοποιούμε κάποια προχωρημένα χαρακτηριστικά της PHP στις client-side εφαρμογές μας. Μπορούμε επίσης να χρησιμοποιήσουμε το εργαλείο PHP-GTK για αυτού του είδους τα προγράμματα. Υπάρχει επίσης η δυνατότητα να γράψουμε cross-platform εφαρμογές με αυτόν τον τρόπο. Το PHP-GTK είναι μια επέκταση της PHP και δεν συμπεριλαμβάνεται στην κύρια διανομή.

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(IIS), Personal Web Server, Netscape και iPlanet servers, Oreilly Website Pro server, Caudium, Xitami, OmniHTTPd και πολλούς άλλους web servers.

Έτσι με την PHP έχουμε την ελευθερία επιλογής ενός λειτουργικού συστήματος και ενός web server. Επιπλέον, μπορούμε να επιλέξουμε μεταξύ συναρτησιακού(procedural) και αντικειμενοστραφή(object oriented) προγραμματισμού ή μία ανάμειξή τους.

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.



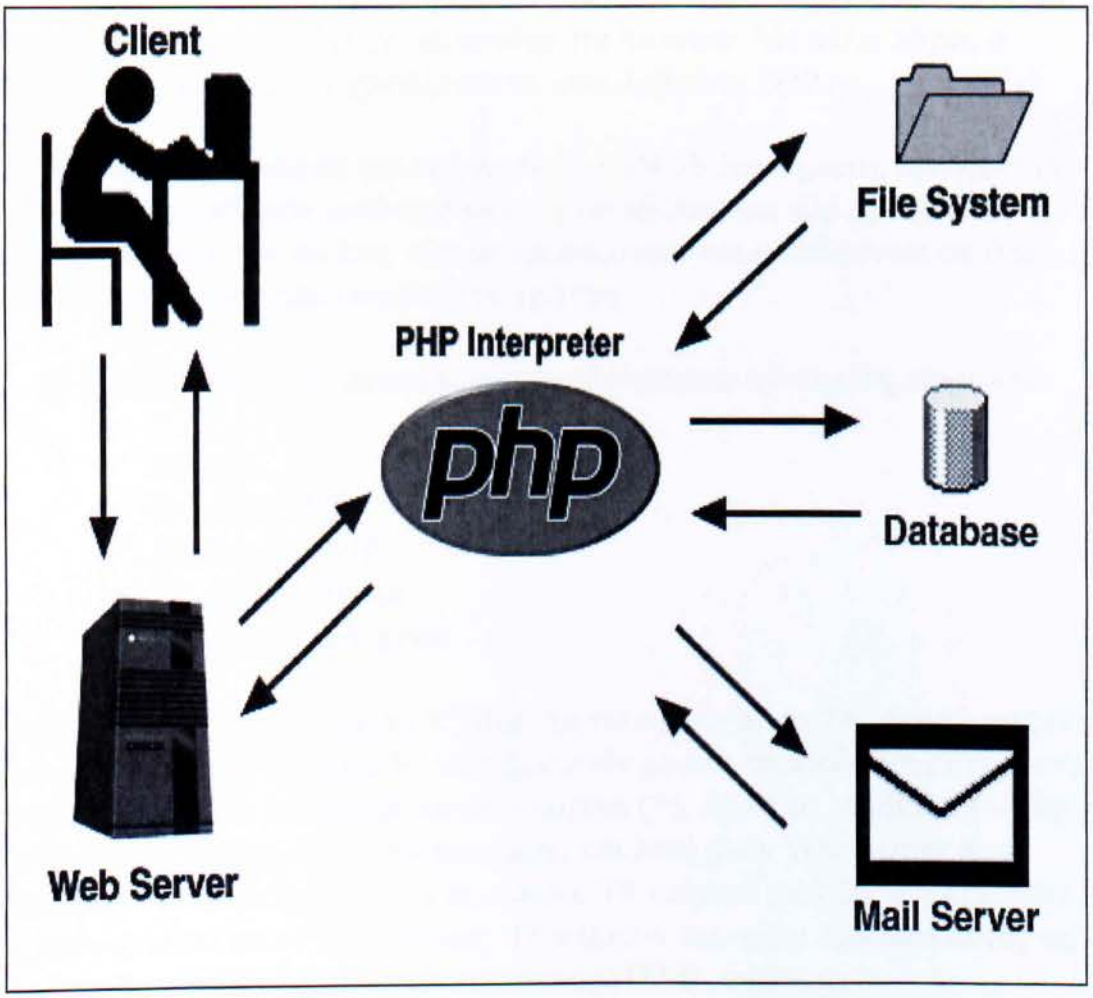
Ένα από τα πιο δυνατά και σημαντικά χαρακτηριστικά της PHP είναι η υποστήριξη που έχει για ένα μεγάλο σύνολο βάσεων δεδομένων. Η συγγραφή μιας σελίδας που υποστηρίζει βάσεις δεδομένων είναι εξαιρετικά απλή.

Μερικές από τις βάσεις δεδομένων που υποστηρίζονται είναι:

• dBase • Direct MS-SQL • Front Base • Hyper wave • IBM DB2 • InterBase • MsSQL	• MySQL • ODBC • Oracle • PostgreSQL • Solid • Sybase • Unix dbm

Έχουμε επίσης μια αφαιρετική επέκταση DBX βάσεων δεδομένων(DBX: database abstraction extension) που σας επιτρέπει διάφανα να χρησιμοποιείται οποιαδήποτε βάση δεδομένων υποστηρίζεται από αυτή την επέκταση. Επιπλέον η PHP υποστηρίζει το ODCS, το Open Database Connection Standard(Ανοιχτό Πρότυπο Σύνδεσης Βάσεων Δεδομένων) έτσι μπορείτε να συνδεθείτε 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. Μπορείτε επίσης να ανοίξετε raw network sockets και να αλληλεπιδράσετε 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. Μπορείτε επίσης να χρησιμοποιήσετε την έκδοση COBRA για να προσπελάσετε remote(απομακρυσμένα) αντικείμενα.



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

Καθώς χρησιμοποιείτε την PHP στον τομέα του e-commerce, θα βρείτε τις Cybercash payment, CyberMUT, VeriSign Payflow Pro και CCVS συναρτήσεις χρήσιμες για τα online προγράμματα πληρωμής σας.

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

Βασικοί Κανόνες Σύνταξης της PHP

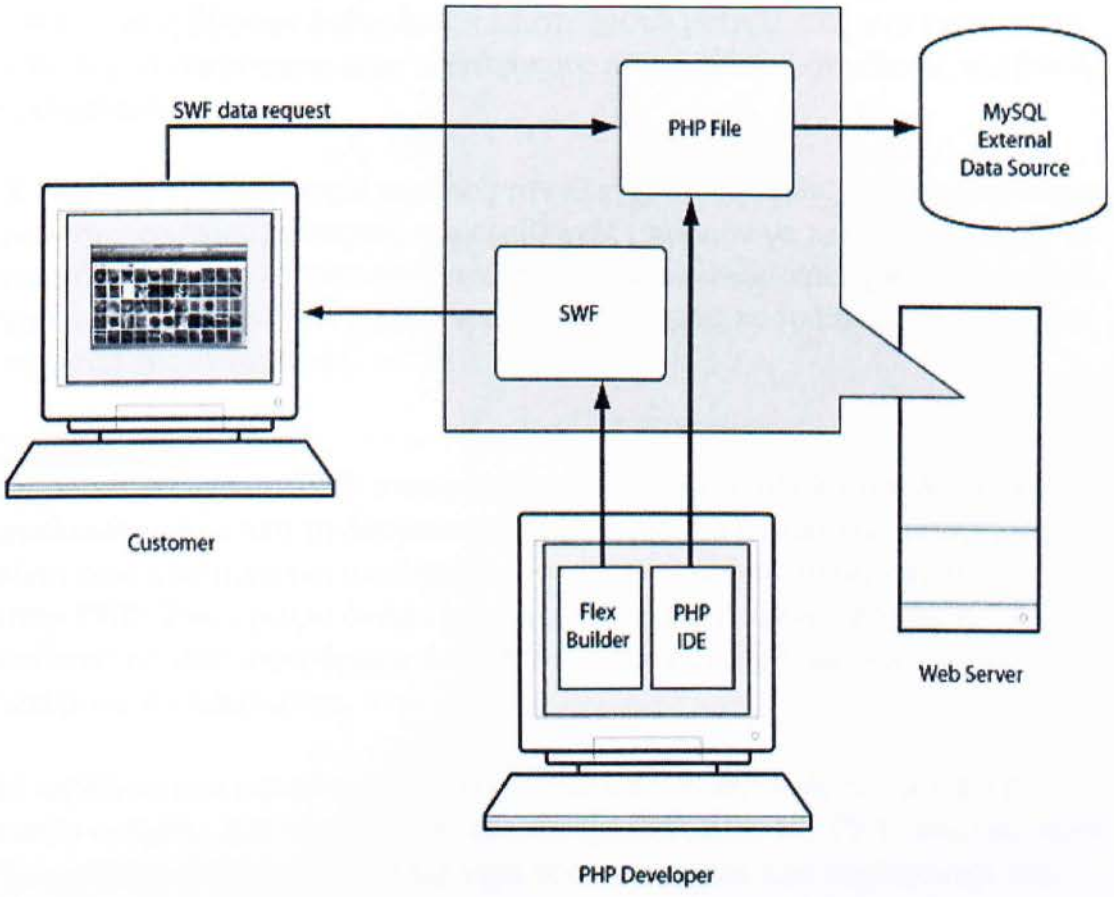
Φορτώνοντας στον browser ένα αρχείο με php κώδικα, παρατηρούμε ότι δεν φαίνεται καθόλου ο απλός PHP κώδικας, επειδή ο PHP μεταφραστής έχει τρέξει το script και το έχει αντικαταστήσει με την έξοδο του script. Αυτό σημαίνει ότι από την PHP μπορούμε να παράγουμε καθαρό κώδικα HTML, που μπορεί να προβληθεί σε οποιονδήποτε browser. Με άλλα λόγια, ο browser του χρήστη δεν χρειάζεται να καταλαβαίνει PHP.

Η PHP έχει μεταφραστεί και εκτελεστεί στο Web διακομιστή, πράγμα που είναι διαφορετικό απο αυτό που κάνει η JavaScript και άλλες τεχνολογίες απο την πλευρά του πελάτη, που μεταφράζονται και εκτελούνται σε ένα Web browser, στον υπολογιστή του χρήστη.

Ο κώδικας μετά τη μετάφραση αποτελείται απο πέντε είδη κειμένου:

- ✓ HTML
- ✓ Ετικέτες PHP
- ✓ Προτάσεις PHP
- ✓ Κενά διαστήματα
- ✓ Σχόλια αν υπάρχουν

Ο κώδικας PHP αρχίζει με το **<?php** και τελειώνει με το **?>**. Αυτό μοιάζει με τις HTML ετικέτες, επειδή όλες ξεκινούν με ένα σύμβολο μικρότερο (<) και τελειώνουν με ένα σύμβολο μεγαλύτερο (>). Αυτά τα σύμβολα (<?php και ?>) ονομάζονται PHP ετικέτες(tags) και λένε στον Web server που ξεκινά ο PHP κώδικας και που τελειώνει. Το κείμενο μεταξύ των ετικετών θα μεταφραστεί σαν PHP κώδικας. Το κείμενο που είναι έξω απο αυτές τις ετικέτες θα αντιμετωπιστεί σαν κανονικός HTML κώδικας.



Τύποι Δεδομένων της PHP

Η PHP υποστηρίζει τους παρακάτω τύπους δεδομένων:

- **Integer** (ακέραιος) – χρησιμοποιείται για ακέραιους αριθμούς.
- **Float** (που ονομάζεται επίσης και Double – διπλής ακρίβειας) – χρησιμοποιείται για πραγματικούς αριθμούς.
- **String** (συμβολοσειρά) – χρησιμοποιείται για συμβολοσειρές χαρακτήρων.
- **Boolean** (λογικές τιμές) – χρησιμοποιείται για τιμές true ή false.
- **Array** (πίνακας) – χρησιμοποιείται για αποθήκευση πολλαπλών στοιχείων του ιδίου τύπου.
- **Objects** (αντικείμενα) – χρησιμοποιείται για αποθήκευση στιγμιότυπων κλάσεων.

Είναι επίσης διαθέσιμοι δύο ειδικοί τύποι: ο τύπος **NULL** και ο τύπος **resource** (πόρος). Οι μεταβλητές στις οποίες δεν έχει δοθεί μια τιμή ή στις οποίες έχει ακυρωθεί η τιμή ή τους έχει δοθεί η συγκεκριμένη τιμή NULL,

είναι τύπου NULL. Ορισμένες ενσωματωμένες συναρτήσεις(όπως οι συναρτήσεις βάσεων δεδομένων) επιστρέφουν μεταβλητές που έχουν τύπο resource. Αντιπροσωπεύουν εξωτερικούς πόρους(όπως συνδέσεις της βάσης δεδομένων).

Η PHP δεν είναι αυστηρή ως προς τον έλεγχο των τύπων. Στις περισσότερες γλώσσες προγραμματισμού, οι μεταβλητές μπορούν να περιέχουν ένα μόνο τύπο δεδομένων και αυτός ο τύπος πρέπει να δηλωθεί πρίν χρησιμοποιηθεί η μεταβλητή. Στην PHP, ο τύπος μίας μεταβλητής προσδιορίζεται απο την τιμή που της εκχωρείται.

Μεταβλητές

Οι μεταβλητές στην PHP αναπαρίστανται απο το σύμβολο του δολαρίου ακολουθούμενο από το όνομα της μεταβλητής. Το όνομα της μεταβλητής είναι case-sensitive και ακολουθεί τους ίδιους κανόνες όπως και οι ετικέτες στην PHP. Ένα έγκυρο όνομα μεταβλητής αρχίζει με ένα γράμμα ή underscore, ακολουθούμενο από οποιονδήποτε αριθμό από γράμματα, αριθμούς ή underscores.

Η εμβέλεια των μεταβλητών καθορίζεται απο το περιεχόμενο μέσα στο οποίο ορίζεται. Για την πλειοψηφία των μεταβλητών της PHP υπάρχει μόνο ενός είδους εμβέλεια. Αυτή περιέχει τόσο τα αρχεία που περιέχονται όσο και αυτά που απαιτούνται.

Η PHP παρέχει ένα μεγάλο αριθμό από προκαθορισμένες μεταβλητές σε οποιοδήποτε script τρέχει. Αρκετές απο αυτές τις μεταβλητές πάντως, δεν μπορούν να τεκμηριωθούν εντελώς αφού εξαρτώνται απο τον server στον οποίο τρέχουν, την έκδοση και το setup του server, καθώς και απο άλλους παράγοντες.

Εκτός απο τους συνηθισμένους τύπους μεταβλητών, η PHP παρέχει και έναν άλλο τύπο, την μεταβλητή variable. Οι μεταβλητές variable μας επιτρέπουν να αλλάζουμε το όνομα μίας μεταβλητής δυναμικά. Γενικότερα, η PHP επιτρέπει πολλή ελευθερία σε αυτό το θέμα. Όλες οι γλώσσες μας επιτρέπουν να αλλάξουμε την τιμή μίας μεταβλητής, αλλά δεν είναι πολλές αυτές που θα μας επιτρέψουν να αλλάξουμε τον τύπο της και ακόμα λιγότερες, αυτές που θα μας επιτρέψουν να αλλάξουμε το όνομά της.

Πιο αναλυτικά, οι έξι βασικοί τύποι της PHP είναι οι εξής:

- Οι ενσωματωμένες υπερ-καθολικές μεταβλητές είναι ορατές παντού μέσα σε ένα script.

- Οι σταθερές μόλις δηλωθούν, είναι πάντα ορατές καθολικά, δηλαδή μπορούν να χρησιμοποιηθούν μέσα και έξω από συναρτήσεις.

- Οι καθολικές μεταβλητές που δηλώνονται σε ένα script, είναι ορατές σε όλο αυτό το script, αλλά όχι μέσα σε συναρτήσεις.

- Οι μεταβλητές που χρησιμοποιούνται μέσα σε συναρτήσεις που έχουν δηλωθεί ως καθολικές, αναφέρονται στην καθολική μεταβλητή με το ίδιο όνομα.

- Οι μεταβλητές που δημιουργούνται μέσα σε συναρτήσεις και δηλώνονται ως στατικές είναι αόρατες έξω από την συνάρτηση αλλά διατηρούν την τιμή τους μεταξύ μίας εκτέλεσης της συνάρτησης και της επόμενης.

- Οι μεταβλητές που χρησιμοποιούνται μέσα σε συναρτήσεις είναι τοπικές στη συνάρτηση και παύουν να υπάρχουν όταν τερματίζει η συνάρτηση.

Στις νεότερες εκδόσεις της PHP, οι πίνακες $_GET και $_POST και μερικές άλλες ειδικές μεταβλητές έχουν τους δικούς τους κανόνες πεδίου δράσης. Είναι γνωστές ως *υπερ-καθολικές (superglobals)* και φαίνονται παντού, μέσα και έξω απο τις συναρτήσεις.

Ο πλήρης κατάλογος των υπερ-καθολικών μεταβλητών είναι ως εξής:

- **$_GLOBALS:** Ένας πίνακας απο όλες τις καθολικές μεταβλητές(όπως και η λέξη κλειδί global, αυτό μας επιτρέπει να έχουμε πρόσβαση στις καθολικές μεταβλητές μέσα σε μία συνάρτηση – π.χ. $GLOBALS ['myvariable']).
- **$_SERVER:** Ένας πίνακας μεταβλητών περιβάλλοντος διακομιστή.
- **$_GET:** Ένας πίνακας μεταβλητών που περνά στο script μέσω της μεθόδου GET.
- **$_POST:** Ένας πίνακας μεταβλητών που περνά στο script μέσω της μεθόδου POST.
- **$_COOKIE:** Ένας πίνακας απο μεταβλητές cookies.
- **$_FILES:** Ένας πίνακας μεταβλητών που σχετιζονται με αποστολές αρχείων.
- **$_REQUEST:** Ένας πίνακας όλης της εισόδου του χρήστη, συμπεριλαμβανομένων των περιεχομένων εισόδου που περιλαμβάνουν τα $_GET, $_POST και $_COOKIE.
- **$_SESSION:** Ένας πίνακας μεταβλητών συνόδου.

Σταθερές

Μπορύμε να ορίσουμε μια σταθερά χρησιμοποιώντας την **define()** συνάρτηση. Μόλις μια σταθερά οριστεί, δεν μπορεί ποτέ να αλλάξει ή να μην είναι ορισμένη.

Μόνο scalar(βαθμωτά) δεδομένα (**Boolean, Integer, Float** και **String**) μπορούν να συμπεριλαμβάνονται στις σταθερές.

Μπορούμε να πάρουμε την τιμή μιας σταθεράς απλά καλώντας την με το όνομά της. Σε αντίθεση με τις μεταβλητές, δεν θα πρέπει να βάζουμε μπροστά απο μία σταθερά το $. Μπορούμε επίσης να χρησιμοποιήσουμε τη συνάρτηση **constant()**, για να διαβάσουμε την τιμή μιας σταθεράς, αν θέλουμε να πάρουμε το όνομα μιας σταθεράς δυναμικά. Για να πάρουμε μια λίστα όλων των σταθερών που έχουν οριστεί χρησιμοποιούμε την **get defined constants()**.

Οι διαφορές μεταξύ σταθερών και μεταβλητών είναι οι εξής:

- Οι σταθερές δεν έχουν το σύμβολο του δολαρίου($) μπροστά τους.
- Οι σταθερές μπορούν να οριστούν μόνο χρησιμοποιώντας τη συνάρτηση define() και όχι με απλή ανάθεση.
- Οι σταθερές μπορούν να οριστούν και να προσπελαστούν από οπουδήποτε χωρίς να λάβουμε υπόψη τους κανόνες εμβέλειας μεταβλητών.
- Οι σταθερές δεν μπορούν να οριστούν ξανά ή να μην είναι καθορισμένες απο την στιγμή που έχουν οριστεί.
- Οι σταθερές μπορούν να υπολογίσουν μόνο βαθμωτές τιμές.

Τελεστές

Οι τελεστές είναι σύμβολα που μπορούμε να χρησιμοποιήσουμε για να χειριστούμε τιμές και μεταβλητές εκτελώντας μία λειτουργία πάνω τους. Γενικά, οι τελεστές μπορούν να πάρουν 1,2 ή 3 ορίσματα.

-Αριθμητικοί Τελεστές

Οι αριθμητικοί τελεστές είναι κανονικοί μαθηματικοί τελεστές.

Τελεστής	Όνομα	Παράδειγμα	Αποτέλεσμα
+	Πρόσθεση	$a+$b	Πρόσθεση του $a και του $b
-	Αφαίρεση	$a-$b	Αφαίρεση του $a και του $b
*	Πολ/σμός	$a*$b	Γινόμενο του $a και του $b
/	Διαίρεση	$a/$b	Διαίρεση του $a και του $b

-*Λογικοί Τελεστές*

Οι λογικοί τελεστές συνδυάζουν τα αποτελέσματα λογικών συνθηκών. Για παράδειγμα, μπορεί να ενδιαφερόμαστε σε μία περίπτωση αν η τιμή μίας μεταβλητής, της $a είναι μεταξύ 0 και 100. Θα πρέπει να ελέγξουμε τις συνθήκες $a >= 0 και $a<= 100, χρησιμοποιώντας τον τελεστή AND, ως εξής: {$a >= 0 && $a<= 100}.

Η PHP υποστηρίζει τους λογικούς τελεστές AND, OR, XOR και NOT. Το σύνολο των λογικών τελεστών και η χρήση τους συνοψίζεται στον παρακάτω πίνακα:

Τελεστής	**Όνομα**	**Παράδειγμα**	**Αποτέλεσμα**
!	NOT	!$b	Επιστρέφει true αν το $b είναι false και το αντίστροφο
&&	AND	$a&&$b	Επιστρέφει true αν τα $a και $b είναι και τα δύο true, διαφορετικά επιστρέφει false
II	OR	aIIb	Επιστρέφει true αν το $a ή το $b ή και τα δύο είναι true,διαφορετικά επιστρέφει false
And	AND	$a and $b	Το ίδιο με το &&, αλλά με χαμηλότερη προτεραιότητα
Or	OR	$a or $b	Το ίδιο με το II, αλλά με χαμηλότερη προτεραιότητα

Οι τελεστές And και Or έχουν μικρότερη προτεραιότητα απο τους τελεστές && και II.

-*Ο Τελεστής Ίσον*

Ο τελεστής σύγκρισης(==, δύο ίσον), μας επιτρέπει να ελέγχουμε αν δύο τιμές είναι ίσες. Για παράδειγμα, θα μπορύσαμε να χρησιμοποιήσουμε την παράσταση

$a == $b

Για να ελέγξουμε αν οι τιμές που είναι αποθηκευμένες στο $a και στο $b είναι ίσες. Το αποτέλεσμα που επιστρέφεται απο αυτήν την παράσταση θα είναι αληθές αν οι τιμές είναι ίσες, αλλιώς θα είναι ψευδές.

-*Τελεστές για Strings*

Υπάρχουν δύο τελεστές για strings(αλφαριθμητικών). Ο πρώτος είναι ο τελεστής σύνδεσης('.') ο οποίος επιστρέφει την σύνδεση των αριστερών και των δεξιών παραμέτρων. Ο δεύτερος είναι ο τελεστής ανάθεσης σύνδεσης('.='), ο οποίος προσθέτει την παράμετρο της δεξιάς πλευράς στην παράμετρο της αριστερής πλευράς.

-*Τελεστές Αύξησης-Μείωσης*

Η PHP υποστηρίζει τελεστές προ- και μετά- αύξησης όπως η C και η Java.

Παράδειγμα	**Όνομα**	**Αποτέλεσμα**
++$a	Προ-αύξηση	Αυξάνει το $a κατά ένα και επιστρέφει το $a.
$a++	Μετά-αύξηση	Επιστρέφει το $a, και μετά αυξάνει το $a κατά ένα.
--$a	Προ-μείωση	Μειώνει το $a κατά ένα και μετά επιστρέφει το $a.
$a--	Μετά-μείωση	Επιστρέφει το $a και μετά μειώνει το $a κατά ένα.

-*Ο Τελεστής Μη Εμφάνισης Λαθών*

Ο τελεστής μη εμφάνισης λαθών (@) μπορεί να χρησιμοποιηθεί εμπρός από οποιαδήποτε παράσταση, δηλαδή από οτιδήποτε δημιουργεί ή έχει μία τιμή. Για παράδειγμα

$a = 8(57/0);

Χωρίς τον τελεστή @, αυτή η γραμμή θα δημιουργήσει μία προειδοποίηση για διαίρεση με το 0. Αν χρησιμοποιηθεί ο τελεστής @, το λάθος δεν εμφανίζεται.

Αν δεν εμφανίζουμε τις προειδοποιήσεις με αυτόν τον τρόπο, θα πρέπει να γράψουμε κάποιον κώδικα χειρισμού των σφαλμάτων για να ελέγχουμε πότε υπάρχει μία προειδοποίηση. Αν έχουμε διαμορφώσει την PHP με την λειτουργία track_errors ενεργοποιημένη, το μήνυμα λάθους θα αποθηκευτεί στην καθολική μεταβλητή $php_errormsg.

-*Ο Τελεστής Εκτέλεσης*

Ο τελεστής εκτέλεσης είναι στην πραγματικότητα ένα ζευγάρι από τελεστές – ένα ζευγάρι απο ανάποδες αποστρόφους(``). Αυτή δεν είναι μια απλή απόστροφος και συνήθως βρίσκεται στο ίδιο πλήκτρο με την περισπωμένη(~) στο πληκτρολόγιό μας.

Η PHP θα προσπαθήσει να εκτελέσει οτιδήποτε περιέχεται μεταξύ αυτών των συμβόλων, σαν μία εντολή της γραμμής εντολών του διακομιστή. Η τιμή της παράστασης είναι η έξοδος της εντολής.

Για παράδειγμα, σε λειτουργικά συστήματα όπως το UNIX, μπορούμε να χρησιμοποιήσουμε τα εξής:

```
$out = `ls – la`;
echo `<pre>`.$out.`</pre>`;
```

ή ισοδύναμα, σε ένα Windows διακομιστή μπορούμε να χρησιμοποιήσουμε τα εξής:

```
$out = `dir c`;
echo `<pre>`.$out.`</pre>`;
```

Και οι δύο αυτές εκδοχές θα πάρουν μία λίστα του καταλόγου και θα την αποθηκεύσουν στο $out. Μπορεί μετά να εμφανιστεί στον browser ή να την χρησιμοποιήσουμε με άλλο τρόπο.

Δομές Ελέγχου

Οι δομές ελέγχου είναι οι δομές μέσα σε μία γλώσσα που μας επιτρέπουν να ελέγχουμε την ροή της εκτέλεσης ενός προγράμματος ή script. Μπορούμε να τις ομαδοποιήσουμε σε δομές υπό όρους(ή διακλάδωσης) και σε δομές επανάληψης(ή βρόγχους). Παρακάτω εξετάζουμε αναλυτικά το συγκεκριμένο χειρισμό κάθε μίας από αυτές τις δομές στην PHP.

-Προτάσεις if

Η δομή if είναι ένα απο τα πιο σημαντικά χαρακτηριστικά σε πολλές γλώσσες, συμπεριλαμβανομένης και της PHP. Επιτρέπει την υπο συνθήκη εκτέλεση κομματιών κώδικα. Η PHP έχει μια δομή if παρόμοια με αυτή της C:

```
If(expr)
statement;
```

Η expr υπολογίζεται στην Boolean τιμή της. Έτσι αν η expr είναι TRUE, η PHP θα εκτελέσει τη *δήλωση* ενώ αν είναι FALSE θα την αγνοήσει. Στο ακόλουθο παράδειγμα, αν το $a είναι μεγαλύτερο απο το $b, εμφανίζει ότι το a είναι μεγαλύτερο απο το b:

```
<?php
If ($a > $b)
print "a is bigger than b";
?>
```

Συχνά θα θέλουμε να εκτελείτε περισσότερες απο μία δηλώσεις σε μια υποθετική συνθήκη. Φυσικά, δεν χρειάζεται να εμπερικλείουμε κάθε δήλωση μέσα σε μια if δομή. Αντιθέτως, μπορούμε να συμπεριλάβουμε αρκετές δηλώσεις σε ένα statement group. Για παράδειγμα, αυτός ο κώδικας θα εμφάνιζε "a is bigger than b" αν το $a είναι μεγαλύτερο απο το $b και τότε θα ανέθεται την τιμή του $a στην μεταβλητή $b:

```
<?php
if ($a > $b) {
print "a is bigger than b";
```

```
$b = $a;
}
?>
```

Οι δηλώσεις με if μπορούν να εμφωλευτούν απεριόριστα μέσα σε άλλες δηλώσεις if(nested if), κάτι το οποίο μας δίνει μεγάλη ευελιξία για την υπό συνθήκη εκτέλεση πολλών μερών του προγραμματός μας.

-Προτάσεις else

Μία πρόταση else μας επιτρέπει να ορίζουμε μία εναλλακτική ενέργεια όταν η συνθήκη της πρότασης if είναι false. Για παράδειγμα, ο ακόλουθος κώδικας θα εμφάνιζε "a is bigger than b" αν το $a είναι μεγαλύτερο απο το $b, και "a is NOT bigger than b" στην αντίθετη περίπτωση:

```
<?php
if ($a > $b) {
print "a is bigger than b";
}
else {
print "a is NOT bigger than b";
}
?>
```

Η συνθήκη else εκτελείται μόνο αν η τιμή της έκφρασης if πάρει την τιμή FALSE.

-Προτάσεις elseif

Όταν υπάρχουν περισσότερες απο δύο επιλογές, μπορούμε να δημιουργήσουμε μία σειρά απο πολλές επιλογές χρησιμοποιώντας την πρόταση elseif, η οποία είναι ένας συνδυασμός της else και της if.Όταν έχουμε μια σειρά απο συνθήκες, το πρόγραμμα μπορεί να ελέγξει κάθε μία, μέχρι να βρει αυτή που είναι αληθής. Για παράδειγμα, το ακόλουθο κομμα΄τι κώδικα θα εμφανίσει "a is bigger than b" ή "a is equal to b" ή "a is smaller than b":

```
<?php
if ($a > $b) {
print "a is bigger than b";
}
elseif ($a == $b) {
print "a is equal to b";
}
else {
print "a is smaller than b";
}
?>
```

Μπορούν να υπάρχουν πολλά elseif μέσα στην ίδια έκφραση if. Η πρώτη elseif έκφραση(αν υπάρχει) που θα πάρει την τιμή TRUE θα είναι και αυτή

που θα εκτελεστεί. Στην PHP, μπορούμε επίσης να γράψουμε else if(σε δύο λέξεις δηλαδή) και το τελικό αποτέλεσμα είναι θα είναι ίδιο. Η έκφραση elseif εκτελείται μόνο αν η προηγούμενη έκφραση if και οποιεσδήποτε προηγούμενες εκφράσεις elseif έχουν πάρει την τιμή FALSE και η τρέχουσα έκφραση elseif πάρει την τιμή TRUE.

-Βρόγχοι while

Το απλούστερο είδος βρόγχου στην PHP είναι ο βρόγχος while. Όπως και μια πρόταση if, έτσι και αυτός βασίζεται σε μία συνθήκη. Η διαφορά μεταξύ ενός βρόγχου while και μίας πρότασης if είναι ότι η πρόταση if εκτελεί το επόμενο τμήμα κώδικα μία φορά, εφόσον η συνθήκη είναι αληθής. Οβρόγχος while εκτελεί το τμήμα επαναλαμβανόμενα, εφόσον η συνθήκη είναι αληθής.

Γενικά χρησιμοποιούμε ένα βρόγχο while όταν δεν ξέρουμε πόσες επαναλήψεις θα απαιτηθούν για να γίνει η συνθήκη αληθής. Αν απαιτείται σταθερός αριθμός επαναλήψεων, μπορούμε να χρησιμοποιήσουμε ένα βρόγχο for.
Η βασική δομή ενός βρόγχου while είναι:

While(condition) expression;

Ο παρακάτω βρόγχος while θα εμφανίσει τους αριθμούς 1 έως 5.

```
$num = 1;
While ($num <= 5)
{
echo $num."<br/>";
$num++;
}
```

Στην αρχή κάθε επανάληψης, ελέγχεται η συνθήκη. Αν η συνθήκη είναι ψευδής, το μπλόκ δεν θα εκτελεστεί και ο βρόγχος θα τερματιστεί. Θα εκτελεστεί η επόμενη πρόταση μετά το βρόγχο. Αν η έκφραση while πάρει την τιμή FALSE απο την αρχή, η εμφωλευμένη εντολή-έκφραση δε θα εκτελεστεί ούτε μία φορά. Όπως και στη δήλωση if, μπορούμε να βάλουμε πολλές συνθήκες μέσα στο ίδιο while loop εσωκλείοντας τες μέσα σε {}, ή χρησιμοποιώντας την εναλλακτική σύνταξη:

While(expr):statement…endwhile;

-Βρόγχοι do...while

Η γενική δομή ενός βρόγχου do…while είναι:

```
do
expression;
while( condition );
```

Ένας βρόγχος do…while διαφέρει από ένα βρόγχο while επειδή η συνθήκη ελέγχεται στο τέλος. Αυτό σημαίνει ότι, σε ένα βρόγχο do…while, η πρόταση ή το μπλόκ μέσα στον βρόγχο εκτελείται τουλάχιστον μία φορά.

Ακόμα και σε αυτό το παράδειγμα, στο οποίο η συνθήκη είναι ψευδής στην αρχή και δεν θα γίνει ποτέ αληθής, ο βρόγχος θα εκτελεστεί μία φορά πριν ελεγθεί η συνθήκη και θα τερματιστεί.

```
$num = 100;
do
{
echo $num.'<br/>';
}
while ($num < 1);
```

-Βρόγχος for

Ο τρόπος που χρησιμοποιήσαμε τους βρόγχους while προηγουμένως είναι πολύ συνηθισμένος. Μπορούμε να γράψουμε αυτό το στύλ βρόγχου με πιο συμπαγή μορφή, χρησιμοποιώντας ένα βρόγχο for. Η βασική δομή ενός βρόγχου for είναι:

```
for( expression1; condition; expression2) expression3;
```

Η πρώτη έκφραση(**expression1**) εκτελείται χωρίς να λάβουμε υπόψη κάποια συνθήκη στην αρχή του loop. Εδώ συνήθως ορίζουμε την αρχική τιμή ενός μετρητή.
Το **condition** ελέγχεται πριν από κάθε επανάληψη. Αν πάρει την τιμή TRUE, το loop συνεχίζει και οι εμφωλευμένες εντολές εκτελούνται. Εάν επιστρέψει FALSE, η επανάληψη σταματά. Εδώ συνήθως ελέγχουμε τον μετρητή ως προς μία οριακή τιμή.
Το **expression2** εκτελείται στο τέλος κάθε επανάληψης. Εδώ συνήθως αλλάζουμε την τιμή του μετρητή.
Το **expression3** εκτελείται μία φορά ανά επανάληψη. Αυτή η παράσταση είναι συνήθως ένα τμήμα κώδικα και θα περιέχει το μεγαλύτερο μέρος του κώδικα του βρόγχου.

Στο ακόλουθο παράδειγμα εμφανίζονται οι αριθμοί απο το 1 ως το 10:

```
<?php
for ($i = 1; $i<=10; $i++) {
print $i;
}
?>
```

-Βρόγχος foreach

Στις νεότερες εκδόσεις της PHP συμπεριλαμβάνεται η δομή foreach για την προσπέλαση των πινάκων(arrays). Η foreach χρησιμοποιείται μόνο με πίνακες και θα εμφανιστεί λάθος αν προσπαθήσουμε να την χρησιμοποιήσουμε σε μια μεταβλητή διαφορετικού τύπου ή σε μια μεταβλητή που δεν έχει αρχικοποιηθεί.

Υπάρχουν δυο τρόποι σύνταξης και ο δεύτερος είναι μια ελάχιστη αλλά πολύ σημαντική βελτίωση του πρώτου:

foreach(array_expression as $value) statement
foreach(array_expression as $key => $value) statement

Με τον πρώτο τρόπο σύνταξης κάνουμε προσπέλαση στον πίνακα(array) σύμφωνα με την array_expression.

Με τον δεύτερο τρόπο σύνταξης κάνουμε ακριβώς το ίδιο, με μόνη διαφορά ότι το τρέχον στοιχείο του κλειδιού θα ανατίθεται στη μεταβλητή $key σε κάθε loop.

-Προτάσεις switch

Η πρόταση switch δουλεύει με παρόμοιο τρόπο με την πρόταση if, αλλά επιτρέπει στην συνθήκη να πάρει περισσότερες απο δυο τιμές. Σε μια πρόταση if, η συνθήκη μπορεί να είναι TRUE ή FALSE. Σε μια πρόταση switch, η συνθήκη μπορεί να πάρει οποιονδήποτε αριθμό διαφορετικών τιμών, εφόσον καταλήγει σε έναν απλό τύπο(ακέραιο, συμβολοσειρά ή πραγματικό αριθμό). Πρέπει να γράψουμε μια πρόταση case για κάθε τιμή την οποία θέλουμε να χρησιμοποιήσουμε και προαιρετικά, μια προκαθορισμένη case για να χειριζόμαστε οτιδήποτε για το οποίο δεν παρέχεται μια συγκεκριμένη πρόταση case.

Τα ακόλουθα δυο παραδείγματα είναι δυο διαφορετικοί τρόποι για να γράψουμε το ίδιο πράγμα, ο ένας χρησιμοποιεί μια σειρά απο δηλώσεις if και ο άλλος χρησιμοποιεί τη δήλωση switch:

```
<?php
if ($i ==0) {
print "i equals 0";
} elseif ($i == 1) {
print "i equals 1";
} elseif ($i == 2) {
print "i equals 2";
}
?>

<?php
switch ($i) {
case 0:
print "i equals 0";
break;
case 1:
print "i equals 1";
break;
```

```
case 2:
print "i equals 2";
break;
}
?>
```

Η δήλωση switch εκτελείται προκειμένου να αποφευχθούν λάθη. Η switch εκτελείται γραμμή-γραμμή (για την ακρίβεια, δήλωση-δήλωση). Αρχικά δεν εκτελείται κάποιος κώδικας. Μόνο όταν μια δήλωση case βρεθεί με τιμή που ταιριάζει την τιμή της switch έκφρασης, η PHP αρχίζει να εκτελεί τις δηλώσεις. Η PHP συνεχίζει να εκτελεί τις δηλώσεις μέχρι το τέλος του switch block, ή την πρώτη φορά που θα συναντήσει μια break δήλωση. Αν δεν γράψουμε μια break δήλωση στο τέλος της λίστας δηλώσεων της case, η PHP θα συνεχίσει να εκτελεί τις δηλώσεις της επόμενης case. Για παράδειγμα:

```
<?php
switch ($i) {
case 0:
print "i equals 0";
case 1:
print "i equals 1";
case 2:
print "i equals 2";
}
?>
```

Εδώ αν η $i ισούται με 0, η PHP θα εκτελέσει όλες τις εντολές print(!!!) ενώ αν ισούται με 1 θα εκτελέσει τις τελευταιες δυο εντολές print, ενώ αν ισούατι με 2 θα εκτελεστεί η τελευταία εντολή print. Γι' αυτό τον λόγο είναι σημαντικό να μην ξεχνάμε τις εντολές break.

Σε μια switch εντολή, η συνθήκη υπολογίζεται μόνο μια φορά και το αποτέλεσμα συγκρίνεται με κάθε case εντολή. Αντίθετα σε μια elseif εντολή, η συνθήκη υπολογίζεται ξανά.

Η λίστα των δηλώσεων για μια case μπορεί επίσης να είναι άδεια, η οποία απλά περνά τον έλεγχο στη λίστα των δηλώσεων για την επομένη case.

Μια ιδιαίτερη case είναι η default case. Αυτή η case ταιριάζει σε οτιδήποτε δεν ταιριάζουν οι άλλες cases και θα πρέπει να είναι η τελευταία case εντολή. Για παράδειγμα:

```
<?php
switch ($i) {
case 0:
print "i equals 0";
break;
```

```
case 1:
print "I equals 1";
break;
case 2:
print "i equals 2";
break;
default:
print "i is not equal to 0, 1 or 2";
}
?>
```

-Σπάζοντας μια Δομή Ελέγχου ή ένα script με τις εντολές break, continue ή exit

Εάν θέλουμε να σταματήσουμε την εκτέλεση ενός τμήματος κώδικα, υπάρχουν τρείς προσεγγίσεις απο τις οποίες μπορούμε να επιλέξουμε, ανάλογα με το αποτέλεσμα που προσπαθούμε να επιτύχουμε.

Αν θέλουμε να σταματήσουμε την εκτέλεση ενός βρόγχου, μπορούμε να χρησιμοποιήσουμε την πρόταση **break**, όπως συζητήθηκε νωρίτερα στην στην ενότητα για την πρόταση switch. Αν χρησιμοποιήσουμε την πρόταση break σε ένα βρόγχο, η εκτέλεση του script θα συνεχίσει στην επόμενη γραμμή του script, μετά το βρόγχο. Η break δέχεται ένα προαιρετικό αριθμό παραμέτρων ο οποίος της λέει πόσες εμφωλευμένες δομές πρέπει να διακοπούν-σπάσουν. Για παράδειγμα:

```
<?php
$i=0;
while (++$i) {
switch ($i) {
case 5:
echo "At 5<br>\n";
break 1; /* Exit only the switch. */
case 10:
echo "At 10; quitting<br>\n";
break 2; /* Exit the switch and the while. */
default:
break;
}
}
?>
```

Αν θέλουμε να μεταπηδήσουμε στην επόμενη επανάληψη του βρόγχου, μπορούμε αντίθετα να χρησιμοποιήσουμε την πρόταση **continue**. Αντίστοιχα με την break και η continue δέχεται έναν προαιρετικό αριθμό παραμέτρων ο οποίος της λέει πόσα επίπεδα περικλειόμενων loops πρέπει να προσπεράσει μέχρι το τέλος.

Αν θέλουμε να σταματήσουμε την εκτέλεση ολόκληρου του PHP script, μπορούμε να χρησιμοποιήσουμε την **exit**. Αυτή η προσέγγιση είναι γενικά χρήσιμη όταν εκτελούμε κάποιο έλεγχο λαθών.

-Δήλωση Return
Αν κληθεί μέσα απο μια συνάρτηση, η δήλωση return() αμέσως τερματίζει την εκτέλεση της τρέχουσας συνάρτησης και επιστρέφει την παράμετρο της, ως τιμή της συνάρτησης που κλήθηκε.

Αν κληθεί απο περιοχή με global εμβέλεια, τότε η εκτέλεση του τρέχοντος script του αρχείου τερματίζει. Αν το τρέχον script του αρχείου ήταν include()ed ή require()ed, τότε ο έλεγχος περνάει πάλι στο αρχείο που εκτελεί την κλήση. Επίσης, αν το τρέχον script του αρχείου ήταν include()ed(συμπεριλαμβανόταν), τότε η τιμή που δίνεται στη return() θα επιστρέφεται ως η τιμή της κλήσης της include().

-Δηλώσεις Require() & Include()
Η PHP παρέχει δυο πολύ απλές, αλλά πολύ χρήσιμες προτάσεις, για να μπορούμε να ξαναχρησιμοποιούμε οποιοδήποτε είδος κώδικα. Χρησιμοποιώντας μια πρόταση require() ή include(), μπορούμε να φορτώσουμε ένα αρχείο στο PHP script μας. Το αρχείο μπορεί να περιέχει οτιδήποτε μπορούμε να πληκτρολογήσουμε σε ένα script, συμπεριλαμβανομένων PHP προτάσεων, κειμένου, HTML ετικετών, PHP συναρτήσεων ή PHP κλάσεων.

Η require() και η include() είναι όμοιες σε κάθε περίπτωση εκτός απο το πως χειρίζονται την failure(αποτυχία). Η include() παράγει ένα warning ενώ η require() καταλήγει σε ένα Fatal Error. Με άλλα λόγια, χρησιμοποιούμε την require() αν θέλουμε ένα χαμένο αρχείο να σταματήσει την επεξεργασία της σελίδας. Η include() δεν συμπεριφέρεται με αυτόν τον τρόπο, το script θα συνεχίσει να εκτελείται.

-Δηλώσεις Require-once() & Include-once()
Οι δηλώσεις require_once() και include_once() περιλαμβάνουν και υπολογίζουν το συγκεκριμένο αρχείο κατα την διάρκεια της εκτέλεσης του script. Αυτή η συμπεριφορά είναι όμοια με τη δήλωση require() και include() αντίστοιχα, με τη μόνη διαφορά ότι αν ο κώδικας απο ένα αρχείο έχει ήδη συμπεριληφθεί, τότε δε θα ξανασυμπεριληφθεί.

Η require_once() καθώς και η include_once() θα πρέπει να χρησιμοποιούνται σε περιπτώσεις που υπάρχει η πιθανότητα να ξανασυμπεριλάβουμε και να ξαναϋπολογίσουμε το ίδιο αρχείο περισσότερες απο μια φορές κατά τη διάρκεια της εκτέλεσης ενός συγκεκριμένου script και θέλουμε να είμαστε σίγουροι ότι συμπεριλαμβάνεται ακριβώς μια φορά

προκειμένου να αποφύγουμε προβλήματα με επανακαθορισμό συναρτήσεων, αναθέσεις τιμών σε μεταβλητές, κ.α.

Συναρτήσεις

Στις περισσότερες γλώσσες προγραμματισμού υπάρχουν οι συναρτήσεις. Χρησιμοποιούνται για να περιέχουν κώδικα που εκτελεί μια συγκεκριμένη εργασία. Αυτό κάνει τον κώδικα πιο εύκολο στην ανάγνωση και μας επιτρέπει να τον χρησιμοποιούμε κάθε φορά που θέλουμε να ξαναεκτελέσουμε την συγκεκριμένη εργασία.

Μια συνάρτηση είναι μια ανεξάρτητη λειτουργική μονάδα κώδικα που περιγράφει το περιβάλλον που κάνει την κλήση, εκτελεί κάποια εργασία και προαιρετικά επιστρέφει ένα αποτέλεσμα.

Το παρακάτω παράδειγμα είναι η πιο απλή κλήση μιας συνάρτησης.

Function_name();

Η παραπάνω γραμμή καλεί μια συνάρτηση που ονομάζεται Function_name() και δεν απαιτεί παραμέτρους. Αυτή η γραμμή κώδικα αγνοεί οποιαδήποτε τιμή μπορεί να επιστραφεί απο αυτή την συνάρτηση.

-Συναρτήσεις οριζόμενες απο το χρήστη

Μια συνάρτηση μπορεί να οριστεί χρησιμοποιώντας την ακόλουθη σύνταξη:

```
<?php
function foo ($arg_1, $arg_2, …, $arg_n)
{
echo "Example function.\n";
return $retval;
}
?>
```

Οποιοσδήποτε έγκυρος PHP κώδικας μπορεί να εμφανιστεί σε μια συναρτήση.

Στην PHP 3, οι συναρτήσεις πρέπει να ορίζονται πρίν γίνει κάποια αναφορά σ'αυτές. Τέτοια απαίτηση δεν υπάρχει στην PHP 4 και στις νεότερες εκδόσεις.

Η PHP δεν υποστηρίζει υπερφόρτωση συναρτήσεων(function overloading), ούτε είναι δυνατόν να υπάρχουν μη-καθορισμένες ή ξανά-καθορισμένες συναρτήσεις που έχουν δηλωθεί προηγουμένως.

Τα ονόματα των συναρτήσεων είναι case-insensitive, αλλά γενικά είναι καλή τακτική να καλούμε τις συναρτήσεις όπως εμφανίζονται στη δήλωσή τους.

Οι παλιότερες εκδόσεις της PHP δεν υποστηρίζουν μεταβλητές αριθμών ή παραμέτρους στις συναρτήσεις, παρόλο που οι προκαθορισμένες παράμετροι υποστηρίζονται. Αντίθετα η PHP 4 υποστηρίζει και τα δύο.

-*Παράμετροι Συναρτήσεων*

Η πληροφορία μπορεί να περάσει στις συναρτήσεις διαμέσου της λίστας παραμέτρων, η οποία είναι μια λίστα μεταβλητών ή και σταθερών που διαχωρίζονται με κόμματα.

Η PHP υποστηρίζει το πέρασμα των παραμέτρων με τιμή(το προκαθορισμένο), πέρασμα με αναφορά και προκαθορισμένες τιμές παραμέτρων. Οι λίστες παραμέτρων μεταβλητού μήκους υποστηρίζονται μόνο απο τις νεότερες εκδόσεις της PHP. Ένα παρόμοιο αποτέλεσμα μπορεί να επιτευχθεί στις νεότερες εκδόσεις της PHP περνώντας ένα πίνακα παραμέτρων στη συνάρτηση:

```
<?php
function takes_array($input)
{
echo "$input[0] + $input[1] = ",
$input[0]+$input[1];
}
?>
```

-*Προκαθορισμένες τιμές συναρτήσεων*

Μια συνάρτηση μπορεί να ορίσει προκαθορισμένες τιμές για βαθμωτές παραμέτρους ως ακολούθως:

```
<?php
function makecoffee ($type = "cappuccino")
{
return "Making a cup of $type.\n";
}
echo makecoffee ();
echo makecoffee ("espresso");
?>
```

Το αποτέλεσμα του παραπάνω κώδικα είναι:

Making a cup of cappuccino.

Making a cup of espresso.

Η προκαθορισμένη τιμή πρέπει να είναι μια σταθερή έκφραση, όχι(για παράδειγμα) μια μεταβλητή ή ένα μέλος μιας κλάσης.

Όταν χρησιμοποιούμε προκαθορισμένες παραμέτρους θα πρέπει να είναι στη δεξιά μεριά απο οποιεσδήποτε μη καθορισμένες παραμέτρους. Σε αντίθετη περίπτωση τα πράγματα δε θα δουλέψουν όπως περιμένουμε και θα βγεί μήνυμα λάθους.

-*Λίστες μεταβλητού μήκους παραμέτρων*

Οι νεότερες εκδόσεις της PHP υποστηρίζουν λίστες παραμέτρων μεταβλητού μήκους στις συναρτήσεις που καθορίζονται απο τον χρήστη. Δεν απαιτείται κάποια ιδιαίτερη σύνταξη και οι λίστες παραμέτρων μπορούν

με σαφήνεια να παραχθούν με τους ορισμούς των συναρτήσεων και να συμπεριφέρονται κανονικά.

-*Επιστροφή Τιμών*

Οι τιμές επιστρέφονται χρησιμοποιώντας την προαιρετική δήλωση επιστροφής(return). Οποιοσδήποτε τύπος μπορεί να επιστραφεί, συμπεριλαμβανομένου λίστες και αντικείμενα. Αυτό προκαλεί τη συνάρτηση να σταματήσει την εκτέλεση της αμέσως και να περάσει ο έλεγχος πίσω στη γραμμή απο την οποία την καλέσαμε.

```
<?php
function square ($num)
{
return $num * $num;
}
echo square (4); //outputs '16'.
?>
```

-*Μεταβλητές συναρτήσεις*

Η PHP υποστηρίζει την έννοια των μεταβλητών συναρτήσεων. Αυτό σημαίνει πως αν ένα όνομα μεταβλητής έχει παρενθέσεις, η PHP θα ψάξει για μια συνάρτηση με το ίδιο όνομα που της δίνει η μεταβλητή και θα προσπαθήσει να το εκτελέσει. Ανάμεσα στα άλλα πράγματα, αυτό μπορεί να χρησιμοποιηθεί για την υλοποίηση callbacks, πίνακες συναρτήσεων κ.α.

Οι μεταβλητές συναρτήσεις δε θα δουλέψουν με δομές γλώσσας όπως οι echo(), print(), unset(), isset(), empty(), include(), require() και άλλες παρόμοιες. Χρειάζεται να χρησιμοποιήσουμε τη δική μας wrapper συνάρτηση για να χρησιμοποιήσουμε οποιαδήποτε απο τις δομές ως μεταβλητή συνάρτηση.

Στο ΚΕΦΑΛΑΙΟ ΙΙΙ που ακολουθεί παρουσιάζεται η θεωρητική ανάλυση των βάσεων δεδομένων καθώς και της MySQL, αντικειμένων που θα χρησιμοποιηθούν για την υλοποίηση του κατασκευαστικού μέρους της ιστοσελίδας μας.

ΚΕΦΑΛΑΙΟ ΙΙΙ

MySQL και Βάσεις Δεδομένων

ΕΙΣΑΓΩΓΗ:

ΣΧΕΔΙΑΖΟΝΤΑΣ ΤΗΝ ΔΟΜΗ ΜΙΑΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ

Ο καλύτερος τρόπος σχεδιασμού μιας βάσης δεδομένων είναι ο εξής: Σχεδιάζουμε σ'ένα χαρτί τους πίνακες της βάσης με τα πεδία που περιέχει ο καθένας και χαράσσουμε βέλη που θα δείχνουν ποια πεδία συνδέουν τους πίνακες μεταξύ τους και πως. Υπάρχουν βέβαια τυπικές μέθοδοι σχεδιασμού τέτοιων "λογικών διαγραμμάτων", αλλά οπωσδήποτε η τοποθέτηση σε πίνακες και η αλληλοσυσχέτιση των δεδομένων είναι τελικά θέμα εμπειρίας. Μερικές χρήσιμες γενικές οδηγίες είναι οι ακόλουθες:

1) Αναλύουμε τα δεδομένα σε πίνακες, έτσι ώστε σε κάθε πίνακα να καταγράφεται ένα μόνο είδος δεδομένων(π.χ. στοιχεία για άτομα, για προϊόντα, για τιμολόγια κ.τ.λ.).
2) Δεν καταγράφουμε πληροφορίες για το ίδιο είδος δεδομένων σε διαφορετικούς πίνακες, αλλά στον ίδιο πάντα πίνακα. Ένα συνηθισμένο λάθος είναι να δημιουργούμε πίνακες για καθεμία επιχείρηση με τις οποίες συναλλασσόμαστε ξεχωριστά και κάθε πίνακας να περιέχει τα ίδια πεδία. Τα ονόματα των εταιριών, οι λογαριασμοί ή οτιδήποτε άλλο πρέπει να καταχωρούνται στον ίδιο πίνακα.
3) Προσπαθούμε ν'αποφύγουμε, κατά το δυνατόν, την καταχώρηση των ίδιων πληρφοριών σε διαφορετικές θέσεις της βάσης δεδομένων. Όλες οι ομοειδής πληροφορίες πρέπει να καταγράφονται στον ίδιο πίνακα, στον οποίο θα αναφέρονται οι υπόλοιποι όταν χρειαστεί.
4) Ποτέ δεν καταχωρούμε διαφορετικές χρονικές στιγμές των ίδιων δεδομένων σε μια εγγραφή, εκτός εάν είναι σαφώς προκαθορισμένο το τι θα συμβεί και σε ποιές στιγμές.Όταν λοιπόν ένα δεδομένο διαφοροποιείται χρονικά με μη προβλέψιμο τρόπο, θα πρέπει να το καταχωρούμε σ'έναν πίνακα με εγγραφές που προορίζονται ειδικά για τέτοιες περιπτώσεις.
5) Καθορίζουμε σαφώς τις σχέσεις μεταξύ των πινάκων και σιγουρευόμαστε ότι το πρόγραμμα πραγματοποιεί αξιόπιστα το συσχετισμό των δεδομένων. Ίσως χρειάζεται να προσθέσουμε έναν τεχνητό κωδικό αριθμό ή μπορεί ήδη να υπάρχει ένα πεδίο που καθορίζει ποιές εγγραφές πηγαίνουν μαζί.

Παρά τον ευέλικτο χαρακτήρα αυτού του τρόπου οργάνωσης βάσεων δεδομένων, η λειτουργικότητά τους εξερτάται και απο την πληρότητα και αξιοπιστία των ίδιων των δεδομένων. Τυχόν συνδέσεις μεταξύ των πινάκων δεν πρόκειται να έχουν τα επιθυμητά αποτελέσματα, αν οι συσχετιζόμενοι αριθμοί-κλειδιά εισάγονται σε διαφορετικές θέσεις με διαφορετικούς τρόπους.

Ένα τέτοιο σύστημα επίσης, είναι τελείως ακατάλληλο σε περιπτώσεις ελλιπών δεδομένων ή λανθασμένων στοιχείων, τα οποία διορθώνονται ή συμπληρώνονται με την πάροδο του χρόνου. Οι περιπτώσεις αυτές δεν είναι

λίγες. Παρά το ότι μια βάση δεδομένων μπορεί τελικά να εξυπηρετεί τις τρέχουσες ανάγκες, η απόδοσή της θα είναι πολύ χαμηλή και η χρησιμοποίησή της άβολη. Περιπτώσεις αυτού του είδους απαιτούν εξειδικευμένο πρόγραμμα επεξεργασίας βάσεων δεδομένων ή ειδικά προγράμματα γραμμένα σε μια συμβατική γλώσσα προγραμματισμού. Εμείς για την εφαρμογή μας θα χρησιμοποιήσουμε τη γλώσσα δομημένων ερωτήσεων(Structured Query Language), την MySQL, η οποία δημιουργεί σχεσιακές βάσεις δεδομένων.

Κανονικοποίηση μιας Βάσης Δεδομένων

Κανονικοποίηση είναι η διαδικασία μείωσης του πλεονασμού των δεδομένων μέσα σε μια βάση δεδομένων. Κανονικοποίηση είναι η τεχνική που χρησιμοποιείται όταν σχεδιάζετε και ανασχεδιάζετε μια βάση δεδομένων. Είναι η διαδικασία ή σύνολο οδηγιών που χρησιμοποιούνται για βέλτιστη σχεδίαση μιας βάσης δεδομένων για μείωση των πλεοναζόντων δεδομένων. Οι οδηγίες της κανονικοποίησης, που καλούνται *κανονικές μορφές*, θα αναφερθούν παρακάτω:

Η Μη Κανονικοποιημένη Βάση Δεδομένων

Μια βάση δεδομένων που δεν έχει κανονικοποιηθεί μπορεί να περιλαμβάνει δεδομένα, τα οποία περιέχονται σε ένα ή περισσότερους πίνακες, χωρίς κανένα ιδιαίτερο λόγο. Αυτό μπορεί να είναι κακό για λόγους ασφαλείας, για λόγους χρησιμοποίησης χώρου δίσκου, για την ταχύτητα των ερωτημάτων, για την απόδοση των ενημερώσεων της βάσης δεδομένων και ίσως το σημαντικότερο, για την ακεραιότητα των δεδομένων. Μια βάση δεδομένων πρίν απο την κανονικοποίηση είναι μια βάση δεδομένων που δεν έχει διαιρεθεί λογικά σε μικρότερους, ευκολότερα διαχειριζόμενους πίνακες. Η Εικόνα 1 δείχνει την βάση δεδομένων που χρησιμοποιείται για παράδειγμα πριν απο την κανονικοποίησή της.

Λογική Σχεδίαση Βάσης Δεδομένων

Η βάση δεδομένων πρέπει να σχεδιάζεται έχοντας στο μυαλό μας τον τελικό χρήστη. Η λογική σχεδίαση βάσης δεδομένων, που επίσης αναφέρεται σαν λογικό μοντέλο, είναι η διαδικασία διάταξης δεδομένων σε λογικά οργανωμένες ομάδες αντικειμένων, οι οποίες μπορούν να συντηρηθούν εύκολα. Η λογική σχεδίαση μιας βάσης δεδομένων πρέπει να μειώσει την επανάληψη των δεδομένων ή να την εξαλείψει τελείως. Οι συμβάσεις ονοματοδοσίας που χρησιμοποιούνται σε μια βάση δεδομένων πρέπει επίσης να είναι πρότυπες και λογικές.

Ποιές είναι οι ανάγκες του Τελικού Χρήστη

Αυτές πρέπει να είναι ένα απο τα βασικότερα μελετήματα, όταν σχεδιάζεται μια βάση δεδομένων. Ο τελικός χρήστης είναι το άτομο που τελικά χρησιμοποιεί την βάση δεδομένων. Πρέπει να υπάρχει ευκολία χρήσης μέσω του *μετωπικού εργαλείου* του χρήστη(ενός προγράμματος client που

επιτρέεπι σε ένα χρήστη να προσπελάζει μια βάση δεδομένων), αλλά αυτό μαζί με την βέλτιστη απόδοση, δεν μπορεί να επιτευχθεί αν δεν ληφθούν υπόψη οι ανάγκες του χρήστη.

COMPANY_DATABASE	
emp_id	cust_id
last_name	cust_name
first_name	cust_address
middle_name	cust_city
address	cust_state
city	cust_zip
state	cust_phone
zip	cust_fax
phone	ord_num
pager	qty
position	ord_date
date_hire	prod_id
pay_rate	prod_desc
bonus	cost
date_last_raise	

Εικόνα 1

Ορισμένα θέματα σχεδίασης που σχετίζονται με τον χρήστη είναι τα παρακάτω:

- Ποιά δεδομένα πρέπει να αποθηκεύονται στην βάση δεδομένων;
- Πως θα προσπελάζει ο χρήστης την βάση δεδομένων;
- Ποιά προνόμια έχει ο χρήστης;
- Πως πρέπει να ομαδοποιούνται τα δεδομένα μέσα στην βάση δεδομένων;
- Ποιά δεδομένα προσπελάζονται συνηθέστερα;
- Πως σχετίζονται όλα τα δεδομένα μέσα στην βάση δεδομένων;
- Ποιά μέτρα πρέπει να λαμβάνονται για να σιγουρεύετε ότι τα δεδομένα είναι ακριβή;

Πλεονασμός Δεδομένων

Τα δεδομένα δεν πρέπει να είναι πλεονάζοντα, πράγμα που σημαίνει ότι η επανάληψη των δεδομένων πρέπει να μειωθεί στο ελάχιστο, για αρκετούς λόγους. *Για παράδειγμα*, δεν είναι απαραίτητο να αποθηκεύεται η διεύθυνση κατοικίας ενός χρήστη σε περισσότερους απο ένα πίνακες. Με τα επαναλαμβανόμενα δεδομένα χρησιμοποιείται χώρος χωρίς να υφίσταται πραγματικός λόγος. Υπάρχει επίσης ο κίνδυνος της σύγχυσης, *για παράδειγμα*, μια διεύθυνση για ένα υπάλληλο σε ένα πίνακα δεν ταιριάζει με

την διεύθυνση για τον ίδιο υπάλληλο σε έναν άλλο πίνακα. Έτσι ο πλεονασμός των δεδομένων θα μπορούσε να αποβεί καταστροφικός.

Οι Κανονικές Μορφές

Κανονική μορφή είναι ένας τρόπος μέτρησης των επιπέδων ή του βάθους στο οποίο έχει κανονικοποιηθεί μια βάση δεδομένων. Το επίπεδο κανονικοποίησης μιας βάσης δεδομένων καθορίζεται απο την κανονική μορφή.

Παρακάτω ακολουθούν τρείς απο τις πιο συνηθισμένες κανονικές μορφές στην διαδικασία κανονικοποίησης:

- Η πρώτη κανονική μορφή,
- Η δεύτερη κανονική μορφή,
- Η τρίτη κανονική μορφή.

Απο τις τρείς κανονικές μορφές, κάθε επόμενη κανονική μορφή εξαρτάται απο τα βήματα κανονικοποίησης που έχουν ληφθεί στην προηγούμενη κανονική μορφή. *Για παράδειγμα*, για να κανονικοποιήσουμε μια βάση δεδομένων χρησιμοποιώντας την δεύτερη κανονική μορφή, η βάση δεδομένων **πρέπει πρώτα** να βρίσκεται στην πρώτη κανονική μορφή.

Η Πρώτη Κανονική Μορφή

Ο αντικειμενικός σκοπός της πρώτης κανονικής μορφής είναι να διαιρέσουμε τα δεδομένα βάσης σε λογικές μονάδες(πίνακες). Όταν σχεδιάζεται κάθε πίνακας, ένα πρωτεύον κλειδί εκχωρείται στους περισσότερους ή σε όλους τους πίνακες. Η επόμενη εικόνα δείχνει πως φαίνεται η μη κανονικοποιημένη βάση δεδομένων της προηγούμενης εικόνας, που έχει αναπτυχθεί εκ νέου με χρήση της πρώτης κανονικής μορφής.

COMPANY_DATABASE

EMPLOYEE_TBL			*CUSTOMER_TBL*
emp_id	emp_id	cust_id	cust_id
last_name	last_name	cust_name	cust_name
first_name	first_name	cust_address	cust_address
middle_name	middle_name	cust_city	cust_city
address	address	cust_state	cust_state
city	city	cust_zip	cust_zip
state	state	cust_phone	cust_phone
zip	zip	cust_fax	cust_fax
phone	phone	cust_num	ord_name
pager	pager	qty	qty
position	position	ord_date	ord_date
position_desc	position_desc		
date_hire	date_hire		
	pay_rate		

pay_rate	bonus	prod_id	*PRODUCTS_TBL*
bonus	date_last_raise	prod_desc	prod_id
date last raise		cost	prod_desc
			cost

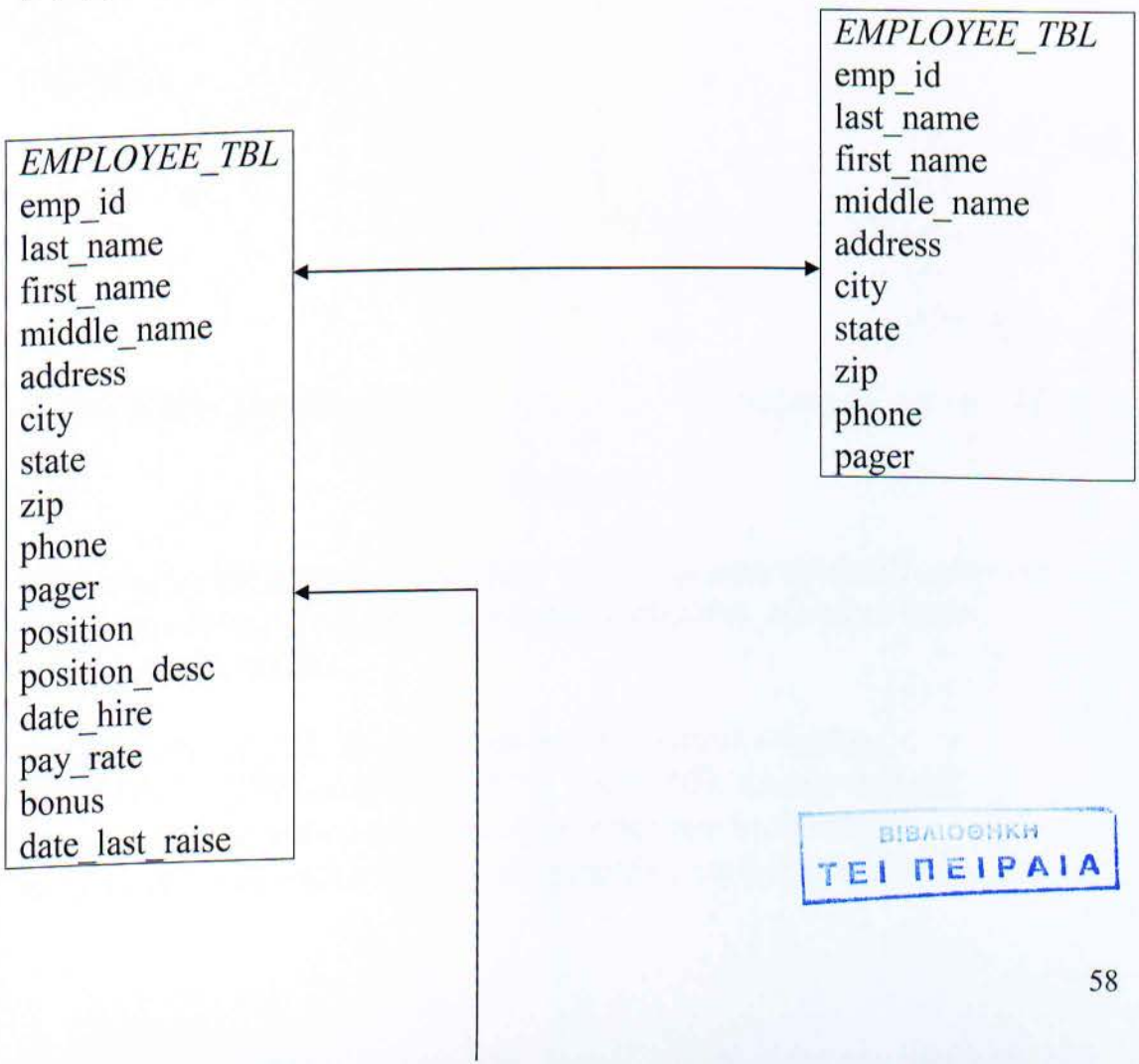
Εικόνα 2

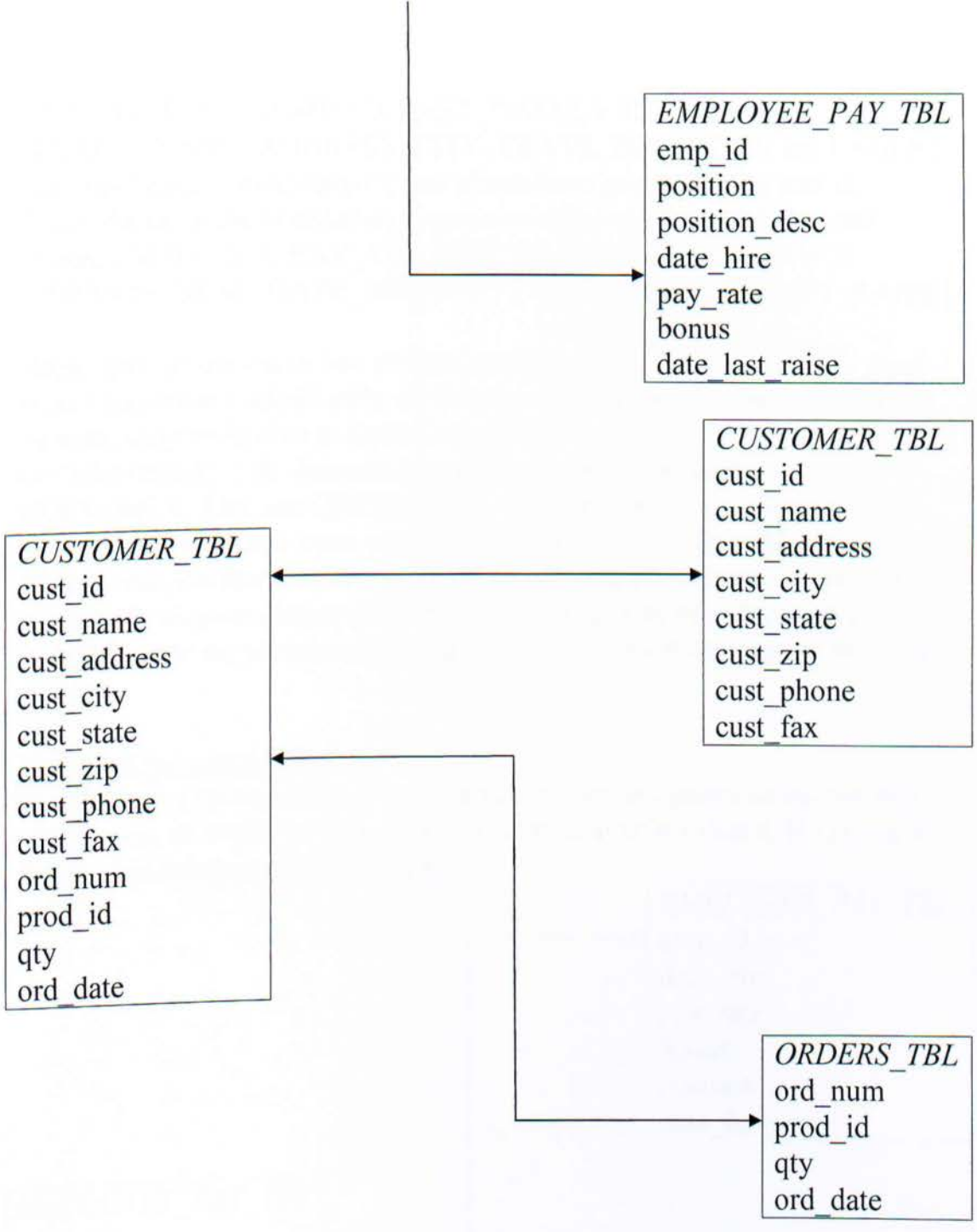
Μπορούμε να δούμε ότι για να επιτευχθεί η πρώτη κανονική μορφή, τα δεδομένα έχουν διαιρεθεί σε λογικές μονάδες σχετικών πληροφοριών, όπου η καθεμία έχει ένα πρωτεύον κλειδί και είναι σίγουρο ότι δεν υπάρχουν επαναλαμβανόμενες ομάδες σε κανένα απο τους πίνακες.

Αντί ενός μεγάλου πίνακα, υπάρχουν τώρα μικρότεροι και ευκολότερα διαχειριζόμενοι πίνακες: EMPLOYEE_TBL, CUSTOMER_TBL και PRODUCT_TBL. Τα πρωτεύοντα κλειδιά είναι συνήθως οι πρώτες στήλες που αναφέρονται μέσα σε ένα πίνακα, που στην περίπτωση μας είναι: EMP_ID, CUST_ID και PROD_ID.

Η Δεύτερη Κανονική Μορφή

Ο σκοπός της δεύτερης κανονική μορφής είναι να πάρει τα δεδομένα που εξαρτώνται μόνο μερικώς απο το πρωτεύον κλειδί και να εισάγει αυτά τα δεδομένα σε ένα άλλο πίνακα. Η Εικόνα 3 δείχνει την δεύτερη κανονική μορφή:





Πρώτη Κανονική Μορφή

Δεύτερη Κανονική Μορφή

Εικόνα 3

Σύμφωνα με την εικόνα, η δεύτερη κανονική μορφή παράγεται απο την πρώτη κανονική μορφή, διαιρώντας περαιτέρω δυο πίνακες σε πιο συγκεκριμένες ομάδες.

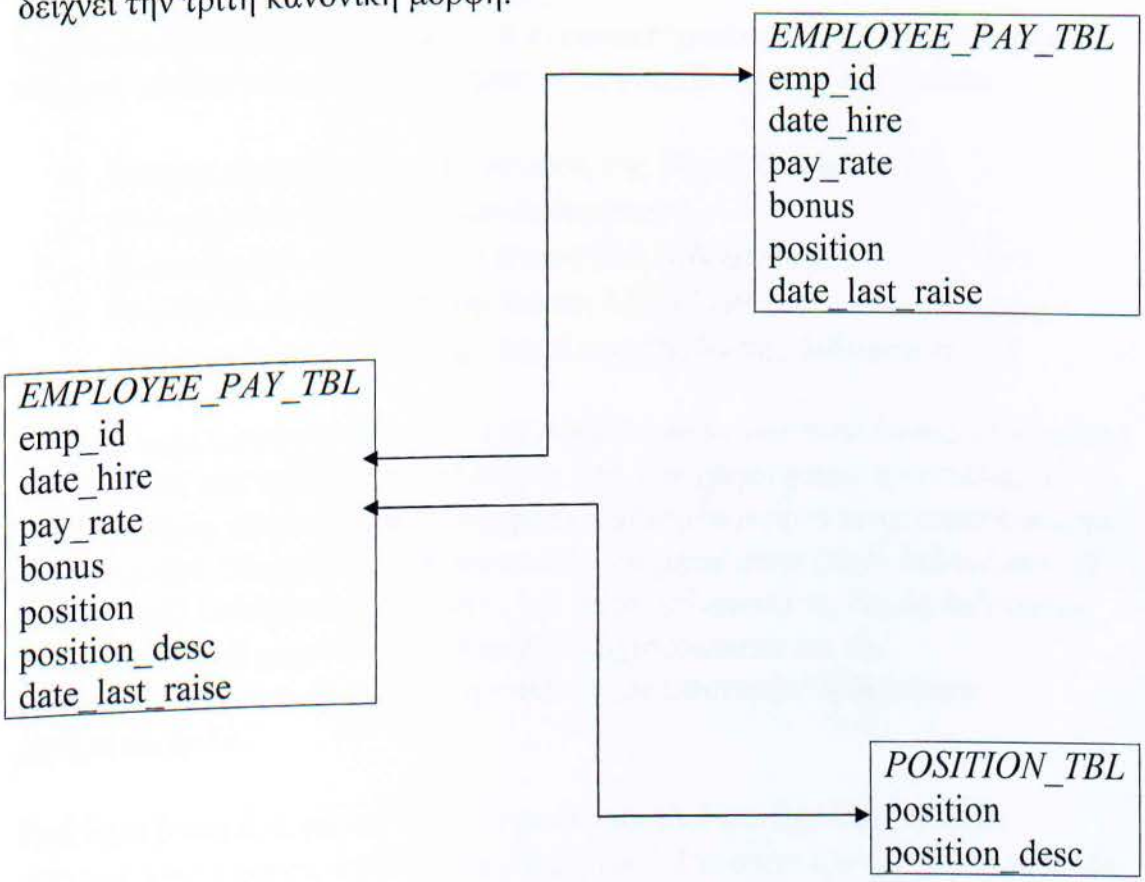
Ο EMPLOYEE_TBL διαιρείται σε δυο πίνακες με ονόματα EMPLOYEE_TBL και EMPLOYEE_PAY_TBL. Οι προσωπικές πληροφορίες για κάθε υπάλληλο εξαρτώνται απο το πρωτεύον κλειδί(EMP_ID), οπότε αυτές οι πληροφορίες παρέμειναν στον

EMPLOYEE_TBL (EMP_ID, LAST_NAME, FIRST_NAME, MIDDLE_NAME, ADDRESS, CITY, STATE, ZIP, PHONE και PAGER). Από την άλλη, οι πληροφορίες που εξαρτώνται μόνο γενικώς απο το EMP_ID(απο κάθε υπάλληλο) χρησιμοποιούνται για να γεμίσουν τον πίνακα EMPLOYEE_PAY_TBL (EMP_ID, POSITION, POSITION_DESC, DATE_HIRE, PAY_RATE και DATE_LAST_RAISE).

Παρατηρούμε ότι και οι δυο πίνακες περιέχουν την στήλη EMP_ID. Αυτό είναι το πρωτεύον κλειδί κάθε πίνακα και χρησιμοποιείται για να ταιριάξει τα αντίστοιχα δεδομένα ανάμεσα στους δυο πίνακες.
Ο CUSTOMER_TBL διαιρείται σε δυο πίνακες με ονόματα CUSTOMER_TBL και ORDER_TBL. Αυτό που γίνεται εδώ είναι παρόμοιο με αυτό που έγινε στον EMPLOYEE_TBL. Οι στήλες που εξαρτώνται μερικώς απο το πρωτεύον κλειδί κατευθύνθηκαν σε ένα άλλο πίνακα. Οι πληροφορίες παραγγελίας για ένα πελάτη δεν εξαρτώνται απευθείας απο τις γενικές πληροφορίες πελάτη, που βρίσκονται στον αρχικό πίνακα.

Η Τρίτη Κανονική Μορφή

Ο σκοπός της τρίτης κανονικής μορφής είναι να αφαιρέσει δεδομένα απο τον πίνακα, τα οποία δεν εξαρτώνται απο το πρωτεύον κλειδί. Η Εικόνα 4 δείχνει την τρίτη κανονική μορφή.



Εικόνα 4

Ένας άλλος πίνακας δημιουργήθηκε για να εμφανίσει την χρήση της τρίτης κανονικής μορφής. Ο EMPLOYEE_PAY_TBL διαιρείται σε δυο πίνακες, ένα πίνακα που περιέχει τις πληροφορίες πληρωμής του υπαλλήλου και ένα άλλο που περιέχει τις περιγραφές θέσεων, οι οποίες δεν χρειάζεται να βρίσκονται στον EMPLOYEE_PAY_TBL. Η στήλη POSITION_DESC είναι τελείως ανεξάρτητη απο το πρωτεύον κλειδί EMP_ID.

Συμβάσεις Ονοματοδοσίας

Οι συμβάσεις ονοματοδοσίας είναι ένα απο τα βασικότερα θέματα όταν κανονικοποιείται μια βάση δεδομένων. Τα ονόματα είναι το μέσο με το οποίο αναφερόμαστε σε αντικείμενα μέσα στην βάση δεδομένων. Θέλουμε να δώσουμε στους πίνακές μας ονόματα, τα οποία περιγράφουν τον τύπο των πληροφοριών που περιέχουν, έτσι ώστε τα δεδομένα που ψάχνουμε να μπορούν να βρεθούν εύκολα. Τα περιγραφικά ονόματα πινάκων είναι ιδιαίτερα σημαντικά για χρήστες που κάνουν ερωτήματα σε βάση δεδομένων και οι οποίοι δεν είχαν καμία συμμετοχή στην σχεδίαση της βάσης δεδομένων. Πρέπει να καθοριστεί μια σύμβαση ονοματοδοσίας για όλη την επιχείρηση, που να καθοδηγεί για την ονοματοδοσία όχι μόνο πινάκων μέσα στην βάση δεδομένων, αλλά επίσης χρηστών, ονομάτων αρχείων και άλλων σχετικών αντικειμένων.

Πλεονεκτήματα της Κανονικοποίησης

Η κανονικοποίηση παρέχει αρκετά πλεονεκτήματα σε μια βάση δεδομένων. Μερικά απο τα κύρια πλεονεκτήματα περιλαμβάνουν τα παρακάτω:

- Μεγαλύτερη συνολική οργάνωση της βάσης δεδομένων
- Μείωση των πλεοναζόντων δεδομένων
- Συνέπεια δεδομένων μέσα στην βάση δεδομένων
- Πιο ευέλικτη σχεδίαση της βάσης δεδομένων
- Καλύτερος χειρισμός της ασφάλειας της βάσης δεδομένων

Η οργάνωση επιτυγχάνεται απο την διαδικασία κανονικοποίησης, κάνοντας την δουλειά του καθενός ευκολότερη, απο τον χρήστη που προσπελάζει πίνακες μέχρι τον διαχειριστή της βάσης δεδομένων που είναι υπεύθυνος για την συνολική διαχείριση κάθε αντικειμένου μέσα στην βάση δεδομένων. Ο πλεονασμός δεδομένων μειώνεται και αυτό απλοποιεί τις δομές δεδομένων και εξοικονομεί χώρο δίσκου. Επειδή ελαχιστοποιούνται τα επαναλαμβανόμενα δεδομένα, η πιθανότητα ασυνεπών δεδομένων μειώνεται πολύ.

Επειδή η βάση δεδομένων έχει κανονικοποιηθεί και έχει διαιρεθεί σε μικρότερους πίνακες, έχετε μεγαλύτερη ευελιξία στην τροποποίηση των ήδη υπάρχων δομών. Είναι πολύ ευκολότερο να τροποποιήσουμε ένα μικρό τμήμα πίνακα με λίγα δεδομένα, παρά να τροποποιήσουμε ένα μεγάλο πίνακα που περιέχει όλα τα κρίσιμα δεδομένα της βάσης δεδομένων μας.

Τέλος, παρέχεται ασφάλεια κατά την έννοια ότι ο διαχειριστής της βάσης δεδομένων μπορεί να εκχωρήσει προσπέλαση σε περιορισμένους πίνακες σε ορισμένους χρήστες. Η ασφάλεια ελέγχεται ευκολότερα όταν έχει γίνει κανονικοποίηση.

Η ακεραιότητα δεδομένων είναι η επιβεβαίωση ότι υπάρχουν συνεπή και ακριβή δεδομένα μέσα σε μια βάση δεδομένων.

Αναφορική Ακεραιότητα
Σημαίνει απλώς ότι οι τιμές σε μια στήλη μέσα σε ένα πίνακα εξαρτώνται απο τις τιμές μιας στήλης σε ένα άλλο πίνακα. *Για παράδειγμα*, για να έχει ένας πελάτης μια εγγραφή του πίνακα ORDER_TBL, πρέπει πρώτα να υπάρχει μια εγγραφή για αυτόν τον πελάτη στον πίνακα CUSTOMER_TBL. Οι περιορισμοί ακεραιότητας μπορούν επίσης να ελέγχουν τιμές, θέτοντας μια περιοχή τιμών για μια στήλη. Ο περιορισμός ακεραιότητας πρέπει να δημιουργηθεί κατά την διάρκεια της δημιουργίας του πίνακα. Η αναφορική ακεραιότητα ελέγχεται συνήθως μέσω της χρήσης του πρωτεύοντος και του εξαρτημένου κλειδιού.

Σε ένα πίνακα ένα εξαρτημένο κλειδί, συνήθως ένα μόνο πεδίο, αναφέρεται απευθείας σε ένα πρωτεύον κλειδί μέσα σε ένα άλλο πίνακα για να επιβάλλει αναφορική ακεραιότητα. Στην προηγούμενη παράγραφο, το CUST_ID στον ORDER_TBL είναι ένα εξαρτημένο κλειδί, που αναφέρεται στο CUST_ID στον CUSTOMER_TBL.

Μειονεκτήματα της Κανονικοποίησης
Αν και οι περισσότερες επιτυχημένες βάσεις δεδομένων είναι κανονικοποιημένες σε κάποιο βαθμό, υπάρχει ένα σημαντικό μειονέκτημα στις κανονικοποιημένες βάσεις δεδομένων:

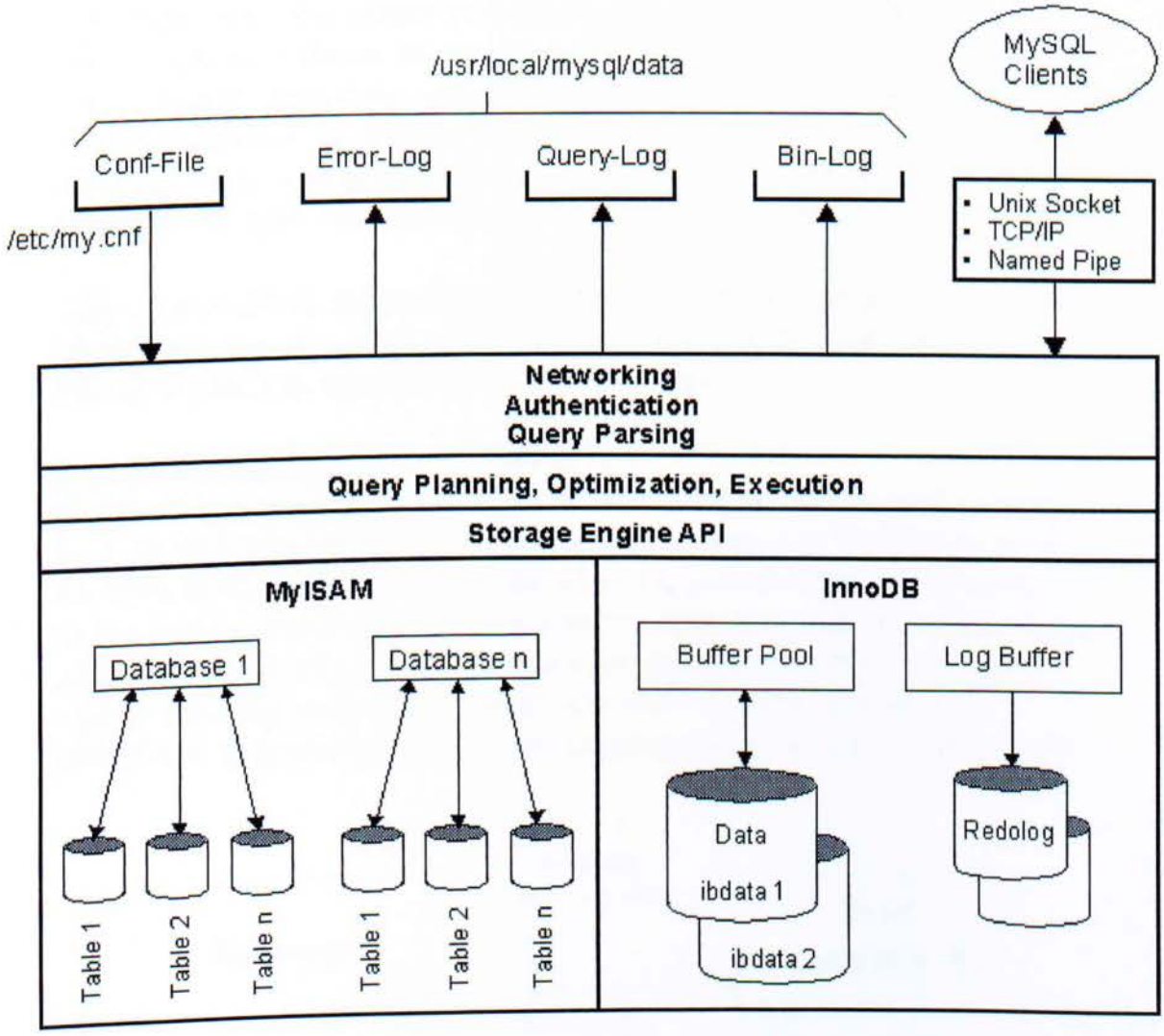
Μειωμένη απόδοση βάσης δεδομένων

Η αποδοχή της μειωμένης απόδοσης απαιτεί την γνώση ότι όταν στέλνεται ένα ερώτημα ή μια αίτηση συναλλαγής στην βάση δεδομένων, υπάρχουν διάφοροι παράγοντες που εμπλέκονται, όπως η χρησιμοποίηση ΚΜΕ, η χρησιμοποίηση μνήμης και η είσοδος/έξοδος(Ε/Ε). Για να το πούμε πιο απλά, μια κανονικοποιημένη βάση δεδομένων απαιτεί περισσότερη ΚΜΕ, μνήμη και Ε/Ε για να επεξεργαστεί συναλλαγές και ερωτήματα βάσης δεδομένων απο ότι μια μη κανονικοποιημένη βάση δεδομένων.

Μια κανονικοποιημένη βάση δεδομένων πρέπει να εντοπίσει τους ζητούμενους πίνακες και μετά να συνενώσει τα δεδομένα απο τους πίνακες είτε για να πάρει τις ζητούμενες πληροφορίες είτε για να επεξεργαστεί τα επιθυμητά δεδομένα.

Αποκανονικοποίηση μιας Βάσης Δεδομένων

Αποκανονικοποίηση είναι η διαδικασία του να παίρνεται μια κανονικοποιημένη βάση δεδομένων και να τροποποιείτε τις δομές πινάκων, ώστε να επιτρέπεται ελεγχόμενος πλεονασμός για να έχουμε αυξημένη απόδοση της βάσης δεδομένων. *Η προσπάθεια αύξησης της απόδοσης είναι ο μόνος λόγος για να αποκανονικοποιήσουμε μια βάση δεδομένων*. Μια αποκανονικοποιημένη βάση δεδομένων δεν είναι το ίδιο πράγμα με μια βάση δεδομένων που δεν έχει κανονικοποιηθεί.



- Table Locking
- Disk Storage
- **No Transactions**
- B-Tree Indices
- R-Tree Indices
- Full-Text Searches

- Row Locking
- Disk Storage
- **Transactions**
- B-Tree Indices
- Foreign Key Constraints
- Crash Recovery

Η αποκανονικοποίηση μιας βάσης δεδομένων είναι η διαδικασία του να μειώνεται κατά λίγο η κανονικοποίηση μιας βάσης δεδομένων. Η κανονικοποίηση μπορεί να μειώσει την απόδοση με τις συχνά επαναλαμβανόμενες συνενώσεις πινάκων.

Η αποκανονικοποίηση μπορεί να περιλαμβάνει την συνένωση χωριστών πινάκων ή την δημιουργία επαναλαμβανόμενων δεδομένων μέσα στους πίνακες για να μειώσει των αριθμό των πινάκων που πρέπει να συνενωθούν για επαναφορά των ζητούμενων δεδομένων, πράγμα το οποίο έχει σαν αποτέλεσμα λιγότερη Ε/Ε και μείωση χρόνου χρησιμοποίησης της ΚΜΕ.

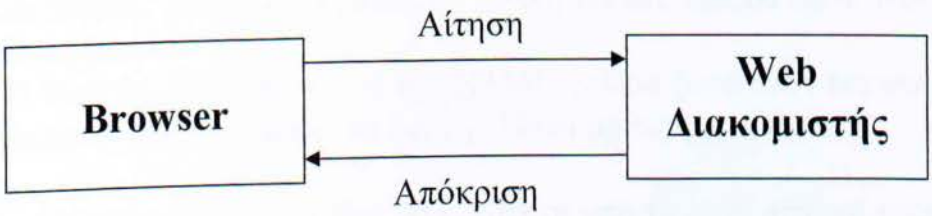
Υπάρχει όμως κόστος στην αποκανονικοποίηση.

Ο πλεονασμός δεδομένων αυξάνεται σε μια αποκανονικοποιημένη βάση δεδομένων, κάτι που μπορεί να αυξήσει την απόδοση, αλλά απαιτεί μεγαλύτερη προσπάθεια για να παρακολουθούνται τα σχετικά δεδομένα. Η κωδικοποίηση εφαρμογών έχει περισσότερες επιπλοκές, επειδή τα δεδομένα είναι διασπαρμένα σε διάφορους πίνακες και μπορεί να είναι δύσκολο να τα βρούμε. Επίσης η αναφορική ακεραιότητα είναι πιο δύσκολη, με τα σχετικά δεδομένα να έχουν διαιρεθεί ανάμεσα σε αρκετούς πίνακες.

Υπάρχει ένας καλός μέσος δρόμος ανάμεσα στην κανονικοποίηση και στην αποκανονικοποίηση, αλλά απαιτεί καλή γνώση των πραγματικών δεδομένων και των απαιτήσεων που επιθυμούμε .

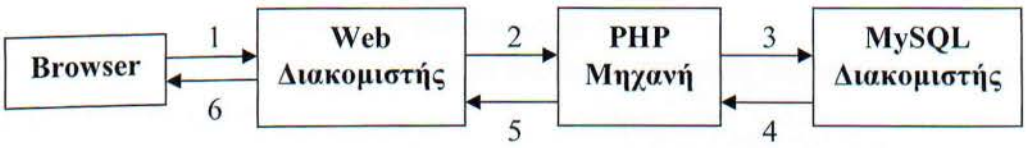
Αρχιτεκτονική Web Βάσεων Δεδομένων

Η βασική λειτουργία ενός Web διακομιστή φαίνεται στο παρακάτω σχήμα. Αυτό το σύστημα αποτελείται απο δύο αντικείμενα: ένα Web browser και ένα Web διακομιστή. Απαιτείται μεταξύ τους μια σύνδεση επικοινωνίας. Ο browser κάνει μια αίτηση στον διακομιστή και ο διακομιστής στέλνει πίσω μία απόκριση. Αυτή η αρχιτεκτονική εξυπηρετεί όταν ο διακομιστής παρέχει στατικές ιστοσελίδες ενώ σε μια Web τοποθεσία με βάσεις δεδομένων, η αρχιτεκτονική που την υποστηρίζει είναι λίγο πιο περίπλοκη.



Σχήμα 1: Η σχέση πελάτη/διακομιστή μεταξύ ενός Web browser και ενός Web διακομιστή απαιτεί επικοινωνία.

Οι Web εφαρμογές με βάσεις δεδομένων που θα δημιουργήσουμε, ακολουθούν την γενική δομή που φαίνεται στο επόμενο σχήμα.



Σχήμα 2: Η βασική Web αρχιτεκτονική βάσεων δεδομένων αποτελείται απο τον Web browser, τον Web διακομιστή, την μηχανή script και τον διακομιστή βάσεων δεδομένων.

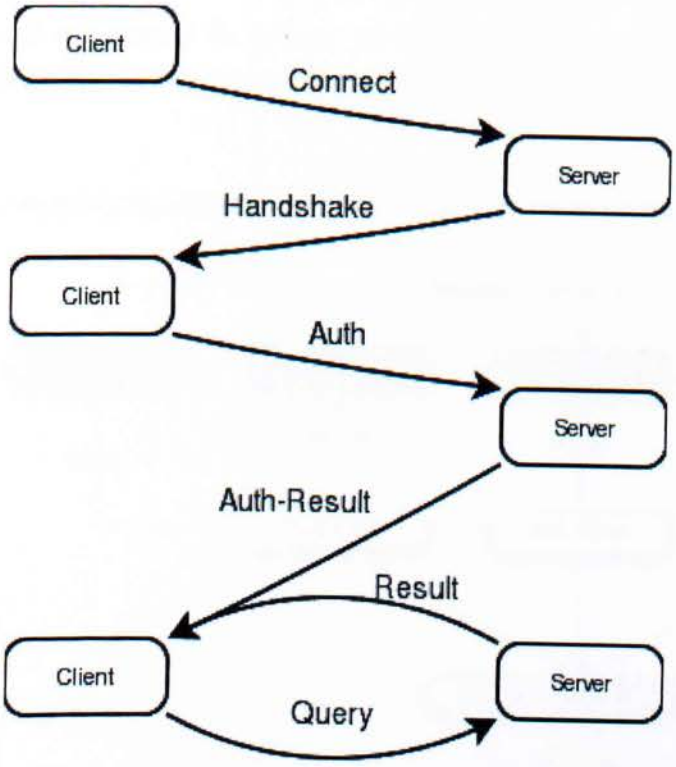
Μία τυπική Web συναλλαγή βάσεων δεδομένων αποτελείται απο τις παρακάτω φάσεις, που είναι αριθμημένες στο παραπάνω σχήμα. Πιο συγκεκριμένα:

1. Ο Web browser ενός χρήστη κάνει μια HTTP αίτηση για μία συγκεκριμένη Web σελίδα. Για παράδειγμα, θέλουμε να δούμε όλες τις ξένες ταινίες. Η σελίδα με τα αντίστοιχα αποτελέσματα ονομάζεται(ας πούμε foreignmovies.php).
2. Ο Web διακομιστής λαμβάνει την αίτηση για την σελίδα foreignmovies.php, ανακαλεί το αρχείο και το περνά στην μηχανή PHP για επεξεργασία.
3. Η PHP μηχανή αρχίζει την ανάλυση του script. Μέσα στο script, υπάρχει μια εντολή που κάνει την σύνδεση με την βάση δεδομένων και εκτελεί ένα ερώτημα(εκτελεί την αναζήτηση των ξένων ταινιών). Η PHP ανοίγει μία σύνδεση με τον MySQL διακομιστή και στέλνει το κατάλληλο ερώτημα.
4. Ο MySQL διακομιστής λαμβάνει το ερώτημα της βάσης δεδομένων και το επεξεργάζεται και στέλνει τα αποτελέσματα, δηλαδή μία λίστα απο ταινίες, ξανά στην PHP μηχανή.
5. Η PHP μηχανή σταματά την εκτέλεση του script, που συνήθως περιλαμβάνει την μορφοποίηση των αποτελεσμάτων του ερωτήματος σε HTML. Επιστρέφει μετά την τελική HTML σελίδα στον Web διακομιστή.
6. Ο Web διακομιστής περνά την HTML σελίδα ξανά στον browser, όπου ο χρήστης μπορεί να δεί την λίστα με τις ξένες ταινίες.

Η διαδικασία είναι βασικά η ίδια, ανεξάρτητα απο το ποιά μηχανή script ή ποιόν διακομιστή βάσης δεδομένων χρησιμοποιούμε. Συνήθως το πρόγραμμα του Web διακομιστή, η PHP μηχανή και ο διακομιστής βάσης δεδομένων βρίσκονται στον ίδιο υπολογιστή.
Ωστόσο, είναι πολύ συνηθισμένο να βρίσκεται ο διακομιστής βάσης δεδομένων σε διαφορετικό υπολογιστή. Αυτό γίνεται για λόγους ασφαλείας, για μεγαλύτερη χωρητικότητα ή για κατανομή του φόρτου.Από την πλευρά της ανάπτυξης η δουλειά θα είναι σχεδόν ίδια, με την διαφορά ότι τώρα θα έχουμε καλύτερη απόδοση.

Διαμόρφωση Web Βάσης Δεδομένων

Αφού δημιουργήσουμε μια βάση δεδομένων, θα πρέπει να διαμορφώσουμε τους χρήστες μας καθώς και τα δικαιώματά τους.
Ένα σύστημα MySQL μπορεί να έχει πολλούς χρήστες όπως φαίνεται και στο παρακάτω σχήμα.



Για λόγους ασφαλείας, ο χρήστης admin θα πρέπει γενικά να χρησιμοποιείται για διαχειριστικούς σκοπούς μόνο. Για κάθε χρήστη που θα χρησιμοποιήσει το σύστημα, θα πρέπει να διαμορφώσουμε ένα λογαριασμό και ένα κωδικό πρόσβασης. Είναι καλύτερα να έχουμε διαφορετικούς κωδικούς πρόσβασης για το σύστημά μας και για την MySQL, ειδικά σε σχέση με τον κωδικό πρόσβασης του administrator.

Δεν είναι απαραίτητο να διαμορφώσουμε κωδικούς πρόσβασης για χρήστες, αλλά καλό θα ήταν να διαμορφώσουμε κωδικούς πρόσβασης για όλους τους χρήστες που δημιουργούμε. Για τη διαμόρφωση μίας Web βάσης δεδομένων, είναι καλύτερα να διαμορφώσουμε τουλάχιστον ένα χρήστη ανά Web εφαρμογή για να μπορούμε να ορίζουμε διαφορετικά δικαιώματα.

Εισαγωγή στο Σύστημα Δικαιωμάτων της MySQL

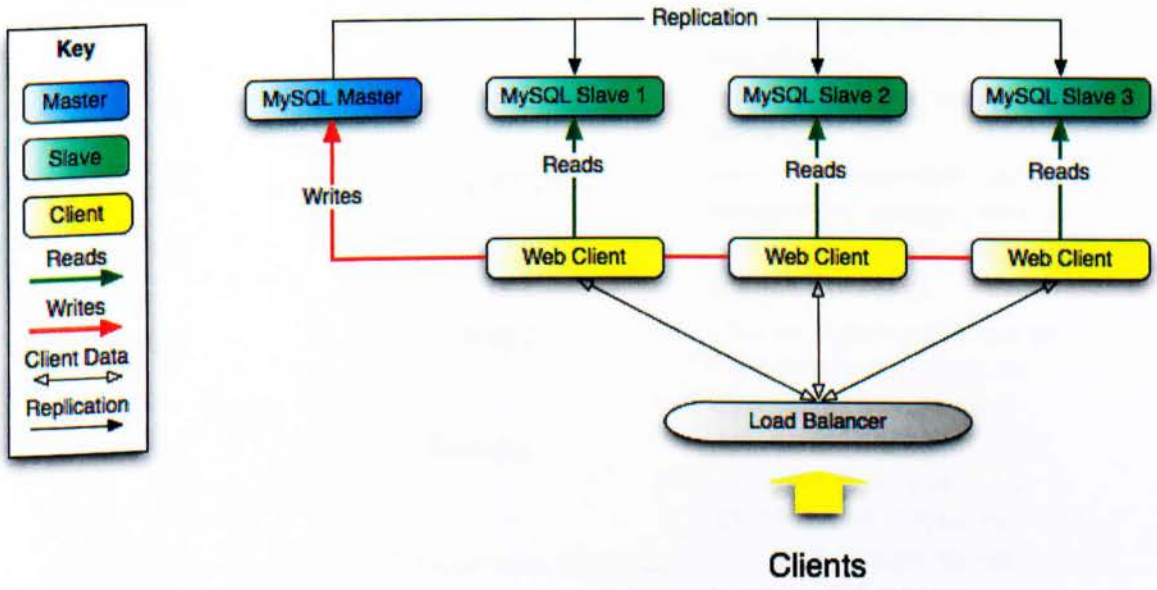
Μία από τις καλύτερες λειτουργίες της MySQL είναι ότι υποστηρίζει ένα προχωρημένο σύστημα δικαιωμάτων. Ένα δικαίωμα είναι η δυνατότητα να εκτελείτε μία συγκεκριμένη ενέργεια σε ένα συγκεκριμένο αντικείμενο και το δικαίωμα σχετίζεται με ένα συγκεκριμένο χρήστη. Η ιδέα είναι παρόμοια με τα δικαιώματα αρχείων. Όταν δημιουργούμε ένα χρήστη στην MySQL, του εκχωρούμε ένα σύνολο δικαιωμάτων για να καθορίσουμε τι μπορεί και τι δεν μπορεί να κάνει μέσα στο σύστημα.

Αρχή των Ελαχίστων Δικαιωμάτων

Η αρχή των ελαχίστων δικαιωμάτων μπορεί να χρησιμοποιηθεί για να βελτιώσει την ασφάλεια οποιουδήποτε μηχανογραφικού συστήματος. Είναι μια απλή αλλά ταυτόχρονα σημαντική αρχή, η οποία όμως παραβλέπεται. Η αρχή είναι η εξής:

Ένας χρήστης(ή διαδικασία) θα πρέπει να έχει το ελάχιστο επίπεδο δικαιωμάτων που απαιτείται για να μπορεί να εκτελεί την εργασία που του έχει ανατεθεί.

Αυτή η αρχή εφαρμόζεται στην MySQL, όπως και οπουδήποτε αλλού.



Για παράδειγμα, για να "τρέξει" ένας χρήστης ερωτήματα(Query) από το Web, δεν χρειάζεται όλα τα δικαιώματα τα οποία μπορεί να έχει ο admin. Θα πρέπει συνεπώς να δημιουργήσουμε άλλον ένα χρήστη, ο οποίος θα έχει μόνο τα απαραίτητα δικαιώματα για πρόσβαση στην βάση δεδομένων που μόλις δημιουργήσαμε.

Τύποι και Επίπεδα Δικαιωμάτων

Υπάρχουν τρείς βασικοί τύποι δικαιωμάτων στην MySQL:

- Δικαιώματα για κανονικούς χρήστες,
- Δικαιώματα για τους διαχειριστές και
- Μερικά Ειδικά Δικαιώματα.

Οποιοσδήποτε χρήστης μπορεί να πάρει οποιαδήποτε απο αυτά τα δικαιώματα, αλλά είναι λογικό να περιοριστούν τα δικαιώματα του διαχειριστή σε διαχειριστές **μόνο**, σύμφωνα με την αρχή των ελαχίστων δικαιωμάτων.

Θα πρέπει να εκχωρούμε δικαιώματα σε χρήστες μόνο για τις βάσεις δεδομένων και τους πίνακες που πρέπει να χρησιμοποιήσουν.

Δεν πρέπει να εκχωρούμε σε κανέναν πρόσβαση στην βάση δεδομένων MySQL, εκτός απο τον διαχειριστή. Εκεί αποθηκεύονται όλοι οι χρήστες, κωδικοί πρόσβασης κτλ.

Τα δικαιώματα για τους κανονικούς χρήστες σχετίζονται άμεσα με συγκεκριμένους τύπους SQL εντολών και αν επιτρέπεται ένας χρήστης να τρέξει τις εντολές. Τα δικαιώματα αυτά φαίνονται στον παρακάτω πίνακα. Τα στοιχεία κάτω απο τη στήλη Εφαρμόζονται Σε, είναι τα αντικείμενα στα οποία μπορούν να δοθούν δικαιώματα αυτού του τύπου.

Δικαίωμα	**Εφαμόζονται Σε...**	**Περιγραφή**
SELECT	Πίνακες, στήλες	Επιτρέπει στους χρήστες να επιλέγουν γραμμές(εγγραφές) από πίνακες.
INSERT	Πίνακες, στήλες	->>- να εισάγουν νέες γραμμές σε πίνακες.
UPDATE	Πίνακες, στήλες	->>- να τροποποιούν τιμές σε υπάρχουσες γραμμές πινάκων.
DELETE	Πίνακες	->>- να διαγράφουν υπάρχουσες γραμμές πινάκων.
INDEX	Πίνακες	->>- να δημιουργούν και να διαγράφουν ευρετήρια σε συγκεκριμένους πίνακες.
ALTER	Πίνακες	->>- να αλλάζουν την δομή υπαρχόντων πινάκων, π.χ. να προσθέτονται στήλες σε πίνακες.
CREATE	Βάσεις δεδομένων, πίνακες	->>- να δημιουργούν νέες βάσεις δεδομένων ή πίνακες.
DROP	Βάσεις δεδομένων, πίνακες	->>- να διαγράφουν βάσεις δεδομένων ή πίνακες.

Πίνακας 1: Δικαιώματα για Χρήστες.

Τα περισσότερα δικαιώματα των κανονικών χρηστών είναι σχετικά ακίνδυνα σε σχέση με την ασφάλεια του συστήματος. Το δικαίωμα *ALTER* μπορεί να χρησιμοποιηθεί για να παρακαμφτεί το σύστημα δικαιωμάτων μετονομάζοντας πίνακες, αλλά γενικώς χρειάζεται στους χρήστες. Η ασφάλεια είναι πάντα μια ανταλλαγή μεταξύ χρησιμότητας και ασφάλειας, γι'αυτό και συνήθως παρέχεται στους χρήστες το δικαίωμα *ALTER*.

Εκτός απο τα δικαιώματα που αναφέρονται στον παραπάνω πίνακα, τα δικαιώματα *REFERENCES* και *EXECUTE* δεν χρησιμοποιούνται αυτή τη στιγμή και το δικαίωμα *GRANT* παρέχεται με το *WITH GRANT OPTION* αντί με την λίστα privileges.

Κανόνες & Λειτουργίες που διέπουν την MySQL

- *Διαμόρφωση ενός χρήστη για το Web*

Θα χρειαστεί να διαμορφώσουμε ένα χρήστη για τα PHP script μας, για να μπορούμε να συνδεόμαστε με την MySQL. Μπορούμε να εφαρμόσουμε την αρχή των ελαχίστων δικαιωμάτων. Ένα ερώτημα που εύλογα προκύπτει είναι τι θα πρέπει να κάνουν τα scripts;

Στις περισσότερες των περιπτώσεων, χρειάζεται μόνο να "τρέχουμε" ερωτήματα(Query) με τα οποία θα επιλέγουμε, θα εισάγουμε, θα διαγράφουμε και θα ενημερώνουμε βάσεις δεδομένων(*SELECT, INSERT, DELETE* και *UPDATE* αντίστοιχα).

- *Χρήση της σωστής Βάσης Δεδομένων*

Αφού διαμορφώσουμε τον χρήστη που θα χρησιμοποιούμε για τα script μας, θα πρέπει να συνδεθούμε σε ένα MySQL λογαριασμό επιπέδου χρήστη. Το πρώτο πράγμα που θα χρειαστεί να κάνουμε όταν συνδεθούμε είναι να καθορίσουμε ποιά βάση δεδομένων θέλουμε να χρησιμοποιήσουμε. Ένας τρόπος είναι να πληκτρολογήσουμε:

mysql> use ***dbname;***

όπου ***dbname*** είναι το όνομα της βάσης δεδομένων.

π.χ. *mysql> use movies;*

Όταν πληκτρολογήσουμε αυτήν την εντολή, η MySQL θα βγάλει το μήνυμα *Database changed.*
Αν δεν επιλέξουμε μια βάση δεδομένων πρίν αρχίσουμε να δουλεύουμε, η MySQL θα μας δώσει το παρακάτω μήνυμα λάθους:

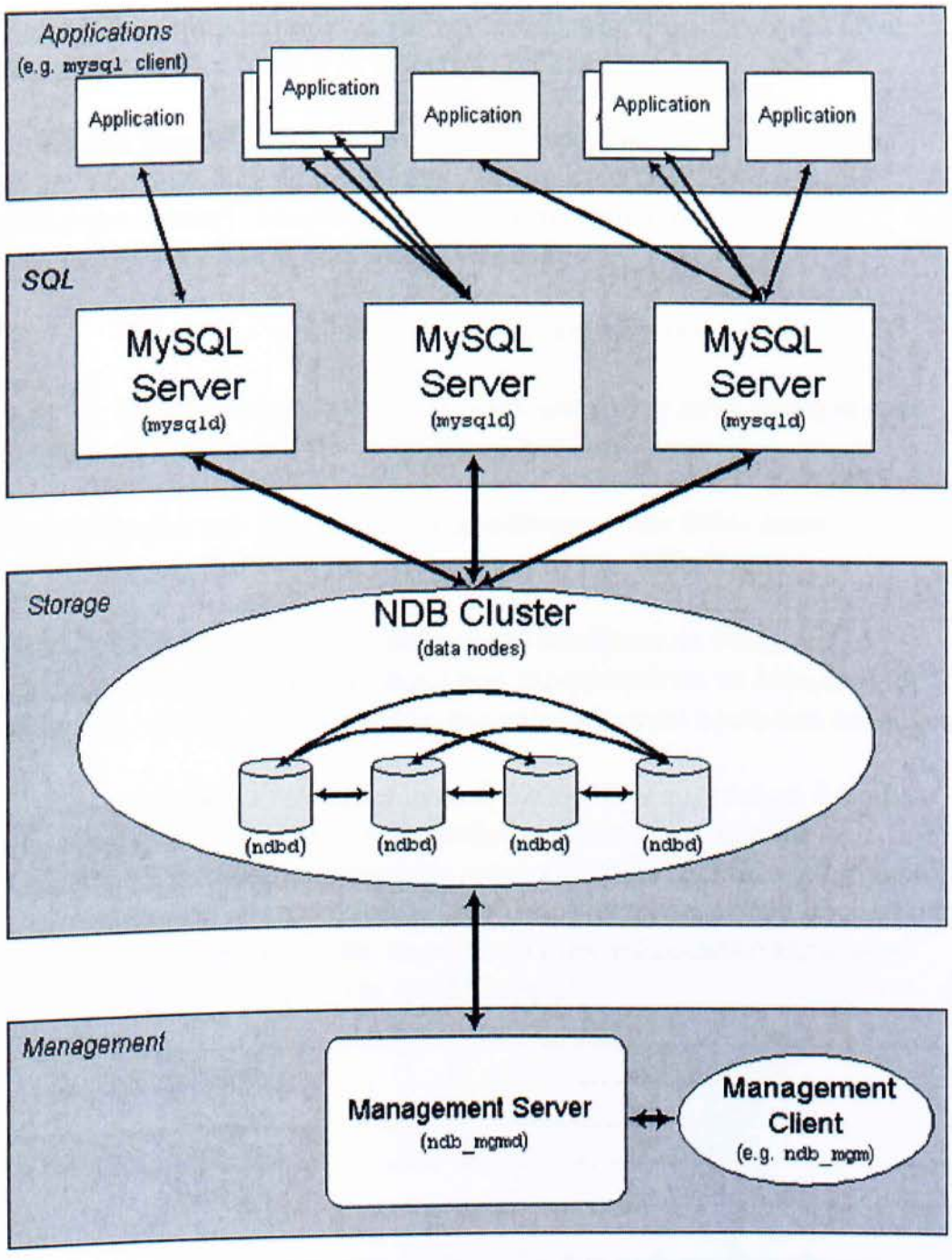
ERROR 1046 (3D000): No Database Selected.

- *Δημιουργία Πινάκων Βάσεων Δεδομένων*

Το επόμενο βήμα για την διαμόρφωση της βάσης δεδομένων είναι να δημιουργήσουμε τους πίνακες που θα περιέχει. Αυτό γίνεται χρησιμοποιώντας την MySQL εντολή *CREATE TABLE*. Η γενική μορφή της εντολής που αναφέραμε πιο πρίν είναι η εξής:

CREATE TABLE tablename(columns)

Θα πρέπει να αντικαταστήσουμε το *tablename* με το όνομα του πίνακα που θα δημιουργήσουμε και το *columns* με μια λίστα, χωρισμένη με κόμματα, των στηλών του πίνακά μας. Κάθε στήλη θα έχει ένα όνομα, ακολουθούμενη από ένα τύπο δεδομένων.



Το Σχεσιακό Μοντέλο Βάσεων Δεδομένων

Τα βασικά στοιχεία μίας σχεσιακής βάσης δεδομένων είναι:

- Πίνακες
- Κλειδιά
- Ευρετήρια

Πίνακες είναι οι μονάδες αποθήκευσης δεδομένων σε μία σχεσιακή βάση δεδομένων. Οι περισσότεροι πίνακες έχουν κλειδιά, που συσχετίζουν τους πίνακες· τον ένα με τον άλλο και επιτρέπουν βέλτιστη αποθήκευση δεδομένων με ελάχιστη ταυτόχρονα επανάληψή τους.

Τα *ευρετήρια* χρησιμοποιούνται για την αναζήτηση δεδομένων και είναι παρόμοια με τα ευρετήρια στο τέλος των βιβλίων.

Το Σχεσιακό Μοντέλο Δεδομένων, όπως αναφερθήκαμε και παραπάνω, είναι ένα σύστημα διαχείρισης βάσεων δεδομένων(DBMS)(*database management system*) και αποτελείται απο ένα σύνολο δεδομένων και προγράμματα πρόσβασης στα δεδομένα αυτά.

Το σύνολο των δεδομένων καλείται *βάση δεδομένων*.

Στόχος του DBMS είναι η εύκολη και γρήγορη χρήση και ανάκτηση των δεδομένων. Η διαχείρηση των δεδομένων περιλαμβάνει:

- τον ορισμό των δομών για την αποθήκευση των δεδομένων.
- τον ορισμό μεθόδων για την διαχείρηση των δεδομένων.

Ο ορισμός της δομής της βάσης δεδομένων βασίζεται σε ένα μοντέλο δεδομένων το οποίο ορίζει τον τρόπο που περιγράφονται τα δεδομένα, οι σχέσεις τους, η σημασία τους καθώς και οι περιορισμοί πάνω στα δεδομένα αυτά.
Το Σχεσιακό Μοντέλο(*relational model*) Δεδομένων παριστάνει δεδομένα και τις σχέσεις τους ως ένα σύνολο πινάκων. Κάθε πίνακας(*table*) αποτελείται από στήλες(*columns*) με μοναδικά ονόματα. Μία γραμμή(*row*) του πίνακα παριστάνει μία σχέση(*relationship*) ανάμεσα σε ένα σύνολο από τιμές. Ο πίνακας που ακολουθεί παριστάνει έναν τηλεφωνικό κατάλογο. Αποτελείται από δύο στήλες και τρείς γραμμές.

Όνομα	*Τηλέφωνο*
Γιώργος	12345
Χρήστος	54321
Άρης	32154

Η MySQL αποτελεί σήμερα την πιο διαδεδομένη γλώσσα διαχείρισης σχεσιακών βάσεων δεδομένων. Η MySQL μας παρέχει δυνατότητες για:

- τον ορισμό, τη διαγραφή και την μεταβολή πινάκων και κλειδιών,
- την σύνταξη ερωτήσεων(*queries*),
- την εισαγωγή, διαγραφή και μεταβολή στοιχείων,
- τον ορισμό όψεων(*views*) πάνω στα δεδομένα,
- τον ορισμό δικαιωμάτων πρόσβασης,
- τον έλεγχο της ακεραιότητας των στοιχείων καί
- τον έλεγχο συναλλαγών(*transaction*).

Στην περιγραφή της σύνταξης της MySQL θα χρησιμοποιήσουμε τα παρακάτω σύμβολα:

[έκφραση]
η έκφραση εμφανίζεται προαιρετικά.

έκφραση 1 | έκφραση 2
μπορεί να γραφεί η έκφραση 1 ή η έκφραση 2.

έκφραση...
η έκφραση μπορεί να επαναληφθεί.

Τύποι Δεδομένων

Τα δεδομένα κάθε στήλης ενός πίνακα πρέπει να έχουν ένα συγκεκριμένο τύπο. Οι βασικοί τύποι που υποστηρίζονται από την MySQL είναι οι παρακάτω:

BIT
Ναί ή Όχι

CURRENCY
Τιμή που παριστάνει με ακρίβεια αριθμούς από -922.337.203.685.477,5808 έως 922.337.203.685.477,5807

DATETIME
Χρόνος

SINGLE
Αριθμός κινητής υποδιαστολής μονής ακρίβειας

DOUBLE
Αριθμός κινητής υποδιαστολής διπλής ακρίβειας

SHORT
Ακέραιος 2 byte (-32768 έως 32767)

LONG
Ακέραιος 4 byte (-2.147.483.648 έως 2.147.483.647)

TEXT
Κείμενο μέχρι 255 χαρακτήρες

LONGTEXT
Κείμενο μέχρι 1.2GB

Δημιουργία Πινάκων

Νέοι πίνακες δημιουργούνται με την εντολή *CREATE TABLE* όπως έχουμε αναφέρει και σε προηγούμενη ενότητα. Η ανωτέρω εντολή συντάσσεται ως εξής:

CREATE TABLE όνομα_πίνακα (πεδίο1 τύπος [(μέγεθος)] [, πεδίο2 τύπος [(μέγεθος)] [, ...])

Για παράδειγμα η εντολή:

CREATE TABLE Check (Name TEXT (20) , AccountNum SHORT)

δημιουργεί τον πίνακα με όνομα Check και με δύο στήλες: την Name και την AccountNum.

Δείκτες

Η πρόσβαση στα στοιχεία ενός πίνακα γίνεται ταχύτερα όταν αυτά οργανωθούν με τη βοήθεια δεικτών. Ένας δείκτης ορίζεται για μια συγκεκριμένη στήλη ή στήλες και επιτρέπει την γρήγορη πρόσβαση σε γραμμές με βάση τιμές της συγκεκριμένης στήλης. Ουσιαστικά όταν ορίζουμε έναν δείκτη το DBMS υλοποιεί μία δομή δεδομένων(π.χ. ταξινομημένο ή κατακερματισμένο πίνακα ή δέντρο) για γρήγορη πρόσβαση στα αντίστοιχα δεδομένα. Δείκτες δημιουργούνται με την εντολή *CREATE INDEX*. Η σύνταξή της είναι η παρακάτω:

CREATE [UNIQUE] INDEX όνομα_δείκτη ON όνομα_πίνακα (πεδίο [ASC | DESC] [, πεδίο [ASC | DESC] , ...])

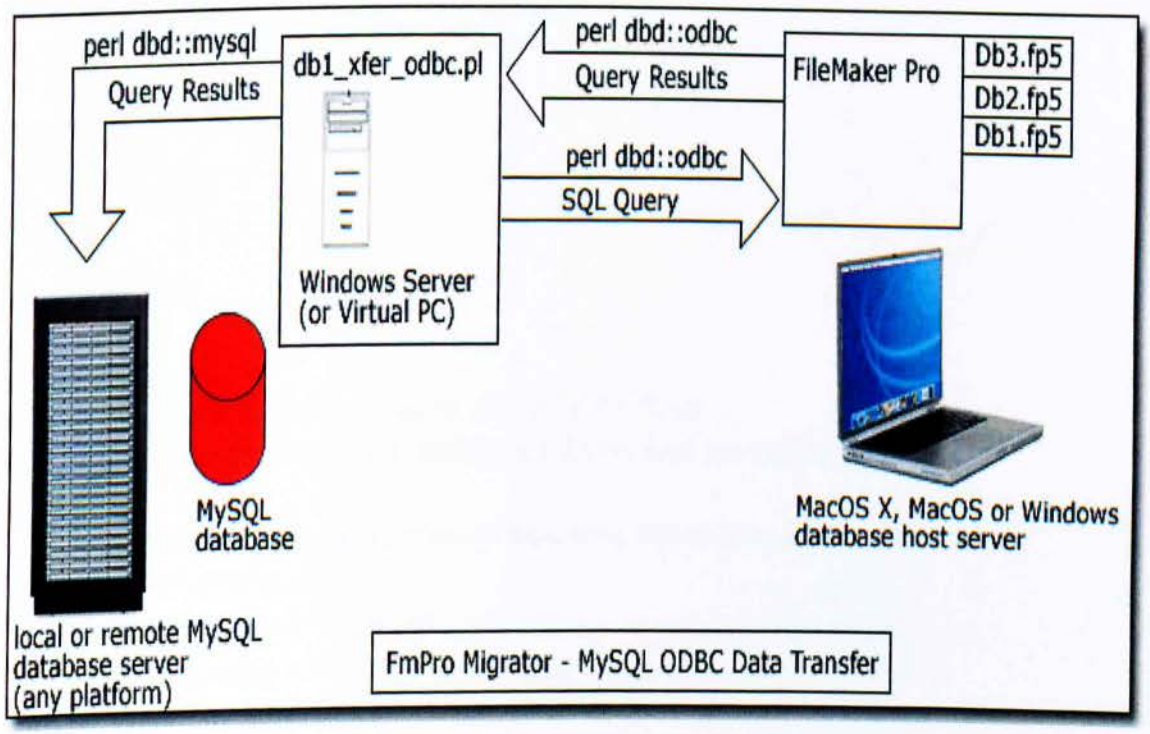
Η λέξη UNIQUE ορίζει πως δεν θα επιτρέπονται διπλές εμφανίσεις μιας τιμής για το συγκεκριμένο δείκτη. Οι λέξεις ASC και DESC ορίζουν αύξουσα ή φθίνουσα ταξινόμηση του πίνακα με βάση το δείκτη.

Παράδειγμα:

CREATE INDEX NameIdx ON Check (Name) δημιουργεί τον δείκτη NameIdx για την στήλη Name στον πίνακα Check.

Ένας δείκτης μπορεί να διαγραφεί με την σύνταξη:

DROP INDEX όνομα_δείκτη ON όνομα_πίνακα.



Προσθήκη Στοιχείων

Η προσθήκη δεδομένων σε ένα πίνακα γίνεται με την εντολή *INSERT INTO* σύμφωνα με την σύνταξη:

INSERT INTO όνομα_πίνακα [(πεδίο1 [, πεδίο2 [, ...]])] VALUES (τιμή1 [, τιμή2 [, ...])

Παράδειγμα:

INSERT INTO Check (Name, AccountNum) VALUES ("Ioannidis", 1234)

Επιλογή Στοιχείων

Η επιλογή δεδομένων από έναν ή περισσότερους πίνακες γίνεται με την εντολή *SELECT* σύμφωνα με την παρακάτω βασική σύνταξη:

SELECT πεδία
FROM πίνακες
[WHERE κριτήρια]

Απλό παράδειγμα:

SELECT Name, AccountNum FROM Check WHERE Name LIKE "Ioan"*

Τα πεδία μπορούν να είναι ονόματα πεδίων ή συγκεντρωτικές συναρτήσεις της MySQL πάνω σε πεδία. Τέτοιες συναρτήσεις είναι οι παρακάτω:

Avg
: Μέσος όρος

Count
: Μέτρηση

Min
: Ελάχιστο

Max
: Μέγιστο

Sum
: Σύνολο

Παράδειγμα:

SELECT Sum (AccountBalance) FROM Check

Ο αστερίσκος ως ορισμός πεδίου επιλέγει όλα τα πεδία.

Τα κριτήρια αναζήτησης είναι εκφράσεις πάνω στα πεδία. Ορισμένοι βασικοί τελεστές είναι οι:

- αριθμητικοί +-*/mod
- σύγκρισης < <= > >> = <> like
- λογικοί and or not

Παράδειγμα:

*SELECT * FROM Check WHERE Balance > 10000 AND Name LIKE "Ioann*"*

Βασικό στοιχείο του σχεσιακού μοντέλου είναι η αποθήκευση δεδομένων σε πίνακες που σχετίζονται μεταξύ τους. Έτσι για παράδειγμα ένας πίνακας μπορεί να κρατάει τα στοιχεία των πελατών και ένας άλλος τους λογαριασμούς. Με τον τρόπο αυτό τα στοιχεία ενός πελάτη που έχει πολλούς λογαριασμούς σε μια π.χ. τράπεζα φυλάσσονται μόνο μία φορά. Ανάκτηση από δύο τέτοιους πίνακες γίνεται με τον προσδιορισμό τους στο *SELECT* μαζί με τη χρήση της κατάλληλης έκφρασης που θα ενώσει τους πίνακες. Ακολουθεί παράδειγμα ώστε να γίνει πιο κατανοητό:

SELECT Check, Balance FROM Check, Accounts WHERE Check.AccountId = Accounts.AccountId

Αλλαγή Στοιχείων
Αλλαγή σε στοιχεία γίνεται με την εντολή *UPDATE* σύμφωνα με την ακόλουθη σύνταξη:
UPDATE όνομα_πίνακα
SET πεδίο = νέα_τιμή
WHERE κριτήρια;

Παράδειγμα:
UPDATE Accounts
SET Balance=10000
WHERE AccountId=12233

Διαγραφή Στοιχείων
Διαγραφή γραμμών από έναν πίνακα γίνεται με την εντολή *DELETE* σύμφωνα με τη σύνταξη:
DELETE
FROM όνομα_πίνακα
WHERE κριτήρια

Παράδειγμα:
DELETE
FROM Accounts
WHERE
AcountId=12332

Κανόνες Σύνταξης της MySQL

Οι Πίνακες Βάσεων Δεδομένων
Όπως αναφερθήκαμε και παραπάνω οι βάσεις δεδομένων (*databases*) περιέχουν αντικείμενα (*objects*) που ονομάζονται Πίνακες(*Tables*). Οι Εγγραφές (*Records*) των δεδομένων αποθηκεύονται σ'αυτούς τους πίνακες.

Οι πίνακες αναγνωρίζονται με τα ονόματά τους, όπως "Persons", "Orders", "Suppliers" κ.ά.

Οι πίνακες περιέχουν Στήλες (*Columns*) και Γραμμές(*Rows*) με δεδομένα. Οι γραμμές περιέχουν εγγραφές, όπως μία εγγραφή για κάθε άτομο. Οι στήλες περιέχουν δεδομένα, όπως π.χ. LastName, FirstName, Address και City.

Ακολουθεί ένα παράδειγμα ενός Πίνακα που ονομάζεται "Members":

LastName	**FirstName**	**Address**	**City**
Γεωργίου	Γεώργιος	Θησέως 17	Αθήνα
Ιακώβου	Ιάκωβος	Προμάχων 1-3	Αθήνα
Αντωνιάδης	Αντώνιος	Πανεπιστημίου 45	Θεσ/νίκη

Τα **LastName**, **FirstName**, **Address** και **City** είναι οι στήλες του πίνακα. Οι γραμμές περιέχουν τρείς εγγραφές για τρία άτομα.

Τα Ερωτήματα της MySQL(SQL Queries)
Με την MySQL μπορούμε να θέσουμε ένα ερώτημα (*Query*) σε μία βάση δεδομένων και να έχουμε ένα αποτέλεσμα (*Result*) σε μορφή πίνακα(tabular form).
Ένα ερώτημα σαν το εξής:
SELECT LastName FROM Members, θα δώσει ένα αποτέλεσμα σαν το εξής:

LastName
Γεωργίου
Ιακώβου
Αντωνιάδης

Πρέπει να έχουμε υπόψη μας ότι μερικά συστήματα βάσεων δεδομένων απαιτούν το σύμβολο ; (*semicolon*) στο τέλος μιας εντολής SQL. Δεν θα χρησιμοποιήσουμε εδώ το σύμβολο ; .

Χειρισμός Δεδομένων της MySQL(Data Manipulation)
Όπως αναφέρθηκε, η SQL είναι μια σύνταξη για την εκτέλεση ερωτημάτων (*queries*). Αλλά η γλώσσα της SQL περιλαμβάνει επίσης μια σύνταξη για την ενημέρωση εγγραφών, την εισαγωγή νέων εγγραφών και τη διαγραφή υπαρχόντων εγγράφων.

Αυτές οι εντολές ερωτημάτων και ενημέρωσης αποτελούν μαζί τη *Γλώσσα Χειρισμού Δεδομένων(Data Manipulation Language,**DML**)* που αποτελεί κομμάτι της SQL:

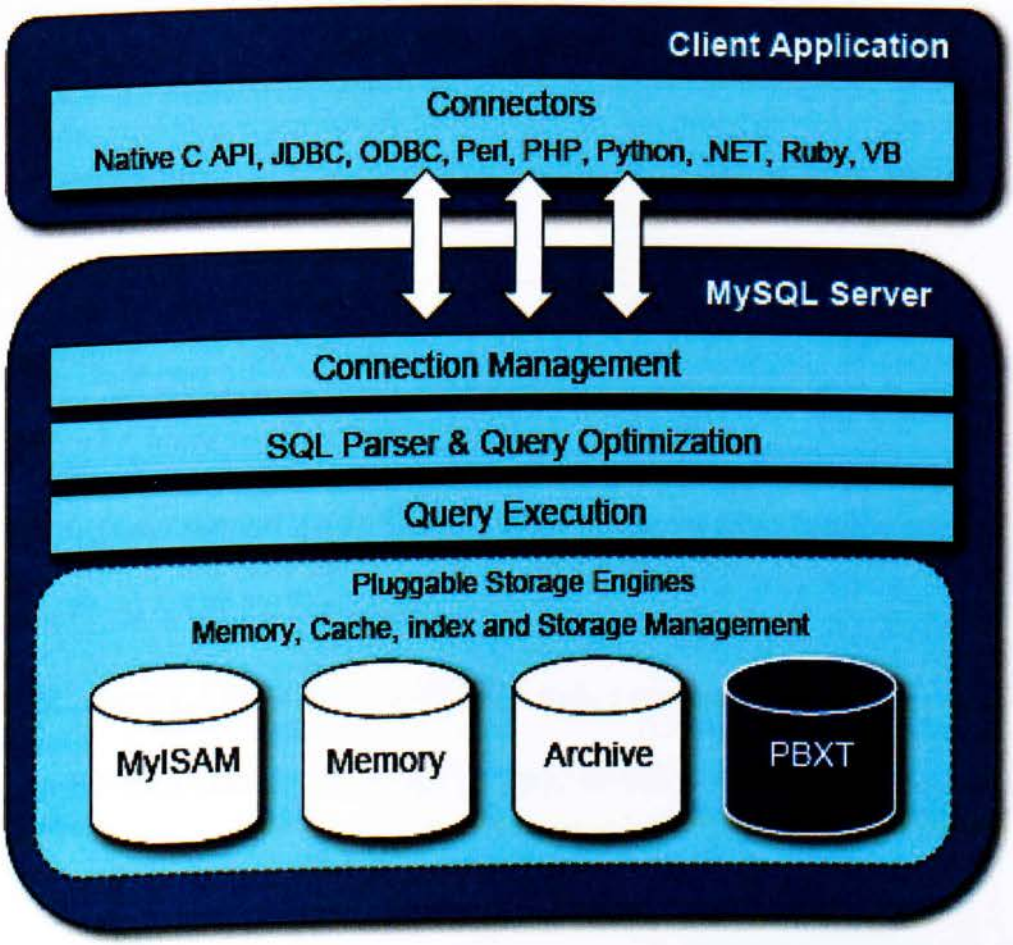
- ***SELECT*** – εξάγει δεδομένα απο μία βάση δεδομένων.
- ***UPDATE*** – ενημερώνει δεδομένα σε μια βάση δεδομένων.
- ***DELETE*** – διαγράφει δεδομένα από μια βάση δεδομένων.
- ***INSERT*** – εισάγει νέα δεδομένα σε μια βάση δεδομένων.

Ορισμός Δεδομένων της MySQL(Data Definition)

Η *Γλώσσα Ορισμού Δεδομένων(Data Definition Language,**DDL**)*, που αποτελεί, μέρος της SQL, επιτρέπει τη δημιουργία και τη διαγραφή πινάκων μιας βάσης δεδομένων. Όπως είπαμε και παραπάνω μπορούμε επίσης να ορίσουμε indexes (*keys*), να καθορίσουμε συνδέσμους (*links*) ανάμεσα στους πίνακες και να επιβάλλουμε περιορισμούς ανάμεσα στους πίνακες μιας βάσης δεδομένων.

Οι σημαντικότερες εντολές DDL στην SQL είναι οι εξής:

- ***CREATE TABLE*** – δημιουργεί έναν νέο πίνακα σε μια βάση δεδομένων.
- ***ALTER TABLE*** – τροποποιεί έναν πίνακα σε μια βάση δεδομένων.
- ***DROP TABLE*** – διαγράφει ένα πίνακα απο μια βάση δεδομένων.
- ***CREATE INDEX*** – δημιουργεί ένα index (*search key*).
- ***DROP INDEX*** – διαγράφει ένα index.



Η Εντολή Select της SQL

Η εντολή *SELECT* επιλέγει στήλες δεδομένων απο μία βάση δεδομένων. Το αποτέλεσμα αποθηκεύεται σε μορφή πίνακα και αποκαλείται *result key*. Την χρησιμοποιούμε για να εμφανίζουμε(επιλέγουμε) πληροφορίες απο έναν πίνακα ως εξής:

SELECT ονόματα_στηλών FROM όνομα_πίνακα

Παράδειγμα: Επιλογή Στηλών απο έναν Πίνακα

Για να επιλέξουμε τις στήλες "LastName" και "FirstName", χρησιμοποιούμε την εντολή *SELECT* ως εξής:

SELECT LastName, FirstName FROM Members

Το αποτέλεσμα:

LastName	**FirstName**
Γεωργίου	Γεώργιος
Ιακώβου	Ιάκωβος
Αντωνιάδης	Αντώνιος

Παράδειγμα: Επιλογή όλων των Στηλών

Για να επιλέξουμε όλες τις στήλες απο τον Πίνακα "Members", χρησιμοποιούμε το σύμβολο * αντί για όνομα στήλης, ως εξής:

*SELECT * FROM Members*

Το αποτέλεσμα:

LastName	**FirstName**	**Address**	**City**
Γεωργίου	Γεώργιος	Θησέως 17	Αθήνα
Ιακώβου	Ιάκωβος	Προμάχων 1-3	Αθήνα
Αντωνιάδης	Αντώνιος	Πανεπιστημίου 45	Θεσ/νίκη

Το Where Clause της SQL

Το *WHERE clause* χρησιμοποιείται για να καθορίσουμε ένα κριτήριο επιλογής(*selection criteria*). Για να μπορέσουμε να επιλέξουμε δεδομένα υπό συνθήκη από έναν πίνακα, πρέπει να προσθέσουμε ένα *WHERE clause* σε μια εντολή *SELECT*, ως εξής:

SELECT στήλη FROM πίνακα WHERE στήλη συνθήκη τιμή

Με το *WHERE clause*, μπορούμε να χρησιμοποιήσουμε αυτές τις συνθήκες:

Τελεστής(Operator)	Συνθήκη(Condition)
=	Ίσο
<>	Όχι ίσο
>	Μεγαλύτερο από
<	Μικρότερο από
>=	Μεγαλύτερο από ή ίσο με
<=	Μικρότερο από ή ίσο με
LIKE	Επεξηγείται παρακάτω...

Σε μερικές εκδόσεις της SQL, ο τελεστής για το όχι ίσο(<>), μπορεί να αναγράφεται και ως εξής: != .

Παράδειγμα: Επιλογή Ατόμων από μια Πόλη
Για να επιλέξουμε μόνο τα άτομα που κατοικούν στην πόλη Αθήνα, προσθέτουμε ένα *WHERE clause* στην εντολή *SELECT*, ως εξής:

*SELECT * FROM Members WHERE City='Αθήνα'*

Το αποτέλεσμα:

LastName	FirstName	Address	City
Γεωργίου	Γεώργιος	Θησέως 17	Αθήνα
Ιακώβου	Ιάκωβος	Προμάχων 1-3	Αθήνα

Έχουμε χρησιμοποιήσει μόνο εισαγωγικά(*single quotes*) στις τιμές των συνθηκών στα παραδείγματα. Η SQL χρησιμοποιεί μονά εισαγωγικά στις αλφαριθμητικές τιμές δηλαδή σε κείμενο. Τα περισσότερα συστήματα βάσεων δεδομένων αποδέχονται και τα διπλά εισαγωγικά(*double quotes*). Ωστόσο οι αριθμητικές τιμές δεν πρέπει να περικλείονται σε εισαγωγικά.

- Για τις τιμές κειμένου(αλφαριθμητικές):

Η ΣΩΣΤΗ σύνταξη είναι η εξής:
*SELECT * FROM Members WHERE FirstName= 'Γεώργιος'*

Η ΛΑΘΟΣ σύνταξη είναι η εξής:
*SELECT * FROM Members WHERE FirstName= Γεώργιος*

- Για τις αριθμητικές τιμές:

Η ΣΩΣΤΗ σύνταξη είναι η εξής:
*SELECT * FROM Members WHERE Year> 1975*

Η ΛΑΘΟΣ σύνταξη είναι η εξής:
*SELECT * FROM Members WHERE Year< '1975'*

Η Συνθήκη LIKE

Η συνθήκη *LIKE* χρησιμοποιείται για να καθορίσουμε μια αναζήτηση για ένα υπόδειγμα (*pattern*) σε μία στήλη.

Η σύνταξη είναι ως εξής:
SELECT στήλη FROM πίνακα WHERE στήλη LIKE υπόδειγμα

Μπορούμε να χρησιμποιήσουμε το σύμβολο "%" για να ορίσουμε χαρακτήρες μπαλαντέρ (*wildcards*) πρίν και μετά από το υπόδειγμα.

Παράδειγμα: Επιλογή Ατόμων με υπόδειγμα ονόματος

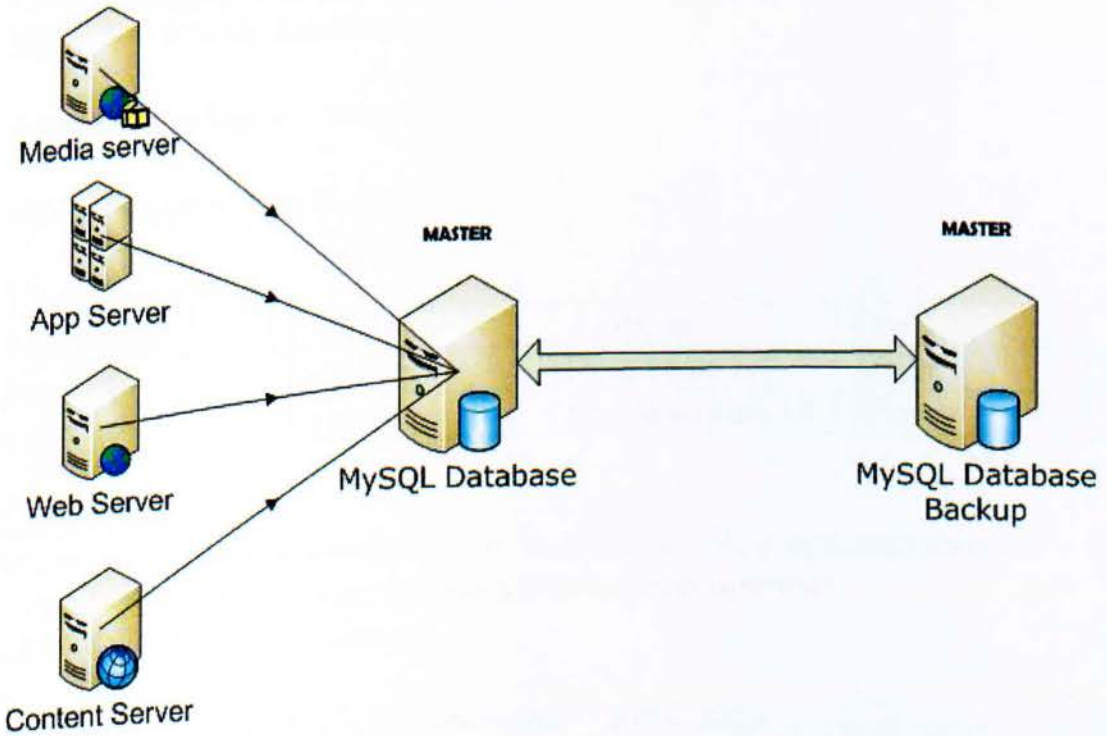
1)Η επόμενη εντολή SQL θα επιστρέψει τα άτομα που το όνομά τους αρχίζει από 'Σ'.
*SELECT * FROM Members WHERE FirstName LIKE 'Σ%'*

2)Η επόμενη εντολή SQL θα επιστρέψει τα άτομα που το όνομά τους τελειώνει σε 'ς'.
*SELECT * FROM Members WHERE FirstName LIKE '%ς'*

3)Η επόμενη εντολή SQL θα επιστρέψει τα άτομα που το όνομά τους περιέχει το 'ρι'.
*SELECT * FROM Members WHERE FirstName LIKE '%ρι%'*

Όλες οι παραπάνω εντολές θα επιστρέψουν το εξής αποτέλεσμα:

LastName	FirstName	Address	City	Year
Σωτηρίου	Σωτήριος	Επισκόπων 20	Αθήνα	1971



Οι Λογικοί Τελεστές And και Or

Τα *AND* και *OR* ενώνουν δύο ή περισσότερες συνθήκες σ'ένα *WHERE clause*. Ο τελεστής *AND* εμφανίζει μία γραμμή αν ΟΛΕΣ οι συνθήκες είναι αληθείς (*true*), ενώ ο τελεστής *OR* εμφανίζει μια γραμμή αν ΟΠΟΙΑΔΗΠΟΤΕ απο τις συνθήκες είναι αληθής (*true*).

Ο αρχικός πίνακας είναι ο εξής:

LastName	**FirstName**	**Address**	**City**
Γεωργίου	Γεώργιος	Θησέως 17	Αθήνα
Ιακώβου	Ιάκωβος	Προμάχων 1-3	Αθήνα
Ιακώβου	Αντώνιος	Καποδιστρίου 18	Αθήνα

Παράδειγμα:
Χρησιμοποιούμε το *AND* για να εμφανίσουμε αυτούς που το όνομά τους είναι "Αντώνιος" και το επώνυμό τους είναι "Ιακώβου":

```
SELECT * FROM Members

WHERE FirstName= 'Αντώνιος'

AND LastName= 'Ιακώβου'
```

Το αποτέλεσμα:

LastName	**FirstName**	**Address**	**City**
Ιακώβου	Αντώνιος	Καποδιστρίου 18	Αθήνα

Παράδειγμα:
Χρησιμοποιώντας το *OR* για να εμφανίσουμε αυτούς που το όνομά τους είναι 'Αντώνιος' ή το επώνυμό τους είναι 'Ιακώβου':

```
SELECT * FROM Members

WHERE FirstName= 'Αντώνιος'

OR LastName= 'Ιακώβου'
```

Το αποτέλεσμα:

LastName	**FirstName**	**Address**	**City**
Ιακώβου	Ιάκωβος	Προμάχων 1-3	Αθήνα
Ιακώβου	Αντώνιος	Καποδιστρίου 18	Αθήνα

Παράδειγμα:
Μπορούμε επίσης να συνδυάσουμε τα *AND* και *OR*, χρησιμοποιώντας παρενθέσεις για να σχηματίσουμε πολύπλοκες εκφράσεις:

```
SELECT * FROM Members

(FirstName= 'Ιάκωβος' OR FirstName= 'Αντώνιος')
```

AND LastName= 'Ιακώβου'

Το αποτέλεσμα:

LastName	FirstName	Address	City
Ιακώβου	Ιάκωβος	Προμάχων 1-3	Αθήνα
Ιακώβου	Αντώνιος	Καποδιστρίου 18	Αθήνα

Ο Τελεστής Between…And

Ο τελεστής αυτός επιλέγει μια περιοχή δεδομένων ανάμεσα σε δύο τιμές. Οι τιμές αυτές μπορεί να είναι αριθμοί, κείμενο ή ημερομηνίες.

SELECT όνομα_στήλης FROM όνομα_πίνακα

WHERE όνομα_στήλης

BETWEEN τιμή1 AND τιμή2

Ο αρχικός πίνακας είναι ο εξής:

LastName	FirstName	Address	City
Γεωργίου	Γεώργιος	Θησέως 17	Αθήνα
Ιακώβου	Ιάκωβος	Προμάχων 1-3	Αθήνα
Ιακώβου	Αντώνιος	Καποδιστρίου 18	Αθήνα
Μαρκόπουλος	Αντώνιος	Αριστοτέλους 20	Θεσ/νίκη

Παράδειγμα:
Για να εμφανίσουμε τα άτομα που βρίσκονται αλφαβητικά ανάμεσα στους "Ιακώβου" και "Μαρκόπουλος", αλλά και να τους περιλαμβάνουν:
*SELECT * FROM Members WHERE LastName*

BETWEEN 'Ιακώβου' AND 'Μαρκόπουλος'

Το αποτέλεσμα:

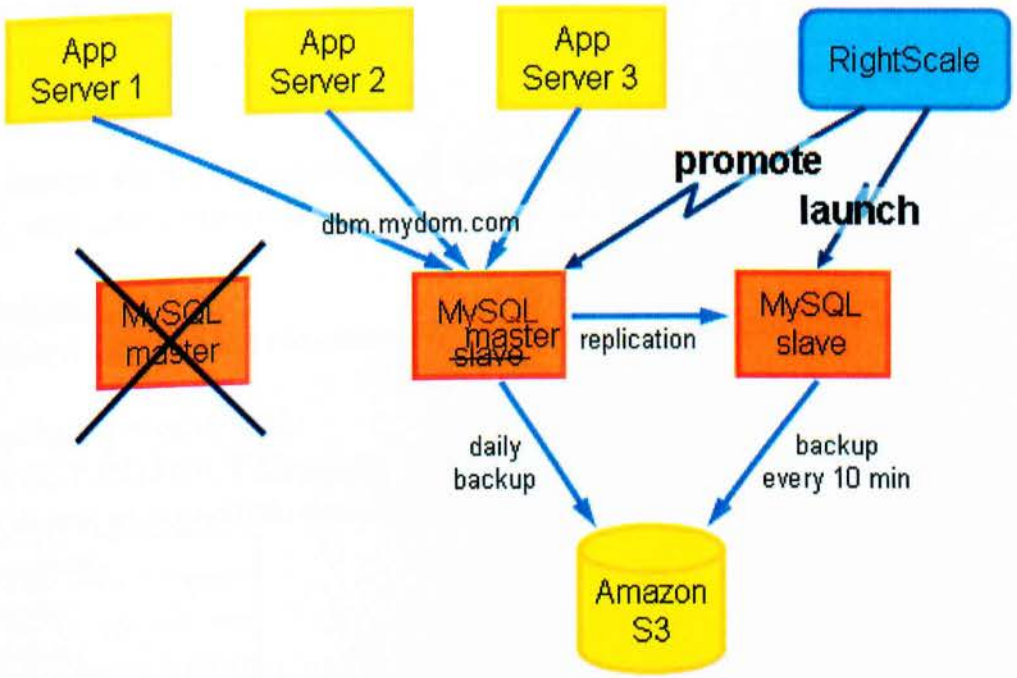
LastName	FirstName	Address	City
Ιακώβου	Ιάκωβος	Προμάχων 1-3	Αθήνα
Ιακώβου	Αντώνιος	Καποδιστρίου 18	Αθήνα
Μαρκόπουλος	Αντώνιος	Αριστοτέλους 20	Θεσ/νίκη

Παράδειγμα:
Για να εμφανίσουμε τα άτομα που βρίσκονται έξω απο την περιοχή που χρησιμοποιήσαμε στο προηγούμενο παράδειγμα, χρησιμοποιούμε τον τελεστή *NOT*:
*SELECT * FROM Members WHERE LastName*

NOT BETWEEN 'Ιακώβου' AND 'Μαρκόπουλος'

Το αποτέλεσμα:

LastName	FirstName	Address	City
Γεωργίου	Γεώργιος	Θησέως 17	Αθήνα



Η Λέξη Κλειδί-Distinct

Η λέξη κλειδί *DISTINCT* χρησιμοποιείται για να επιστρέφει μόνο διακριτές (distinct,different) τιμές. Η εντολή *SELECT* της SQL επιστρέφει στοιχεία απο τις στήλες ενός πίνακα, αλλά τι μπορούμε να κάνουμε αν θέλουμε να επιλέξουμε μόνο διακριτά στοιχεία(distinct elements);

Στην SQL, αυτό που πρέπει να κάνουμε είναι να προσθέσουμε μία λέξη κλειδί *DISTINCT* στην εντολή *SELECT*, ως εξής:

SELECT DISTINCT ονόματα_στηλών FROM όνομα_πίνακα

Παράδειγμα:
Επιλογή εταιρειών από έναν πίνακα κρατήσεων.

Ένας απλός πίνακας(***Πίνακας 1***)κρατήσεων με την ονομασία *Orders*:

Company	OrderNumber
Aegean	3412
Olympic	2312
Easyjet	4678
Olympic	6798

Η επόμενη εντολή SQL:
SELECT Company FROM Orders
θα δώσει το παρακάτω αποτέλεσμα:

Company
Aegean
Olympic
Easyjet
Olympic

Βλέπουμε ότι η εταιρεία Olympic εμφανίζεται δύο φορές στο αποτέλεσμα. Μερικές φορές αυτό δεν είναι επιθυμητό.

Παράδειγμα:
Επιλογή ξεχωριστών εταιριών απο έναν πίνακα κρατήσεων.

Η επόμενη εντολή SQL:
SELECT DISTINCT Company FROM Orders
Θα δώσει το παρακάτω αποτέλεσμα:

Company
Aegean
Olympic
Easyjet

Όπως παρατηρούμε η εταιρεία Olympic τώρα εμφανίζεται μόνο μία φορά στο αποτέλεσμα.

Η Λέξη Κλειδί Order By
Η συγκεκριμένη λέξη κλειδί χρησιμοποιείται για να ταξινομήσει το αποτέλεσμα. Το *ORDER BY clause* χρησιμοποιείται για να ταξινομήσει τις γραμμές.
Έστω ότι έχουμε τον προηγούμενο αρχικό πίνακα, τον ***Πίνακα 1***.

Company	**OrderNumber**
Aegean	3412
Olympic	2312
Easyjet	4678
Olympic	6798

Παράδειγμα:
Για να εμφανίσουμε τις εταιρείες σε αλφαβητική σειρά:
SELECT Company, OrderNumber FROM Orders

ORDER BY Company

Το αποτέλεσμα:

Company	OrderNumber
Aegean	3412
Easyjet	4678
Olympic	6978
Olympic	2312

Παράδειγμα:
Για να εμφανίσουμε τις εταιρείες σε αλφαβητική σειρά ΚΑΙ(*AND*) τις παραγγελίες σε αριθμητική σειρά:
SELECT Company, OrderNumber FROM Orders

ORDER BY Company, OrderNumber

Το αποτέλεσμα:

Company	OrderNumber
Aegean	3412
Easyjet	4678
Olympic	2312
Olympic	6978

Παράδειγμα:
Για να εμφανίσουμε τις εταιρείες σε αντίστροφη αλφαβητική σειρά(*reverse alphabetical order*):
SELECT Company, OrderNumber FROM Orders

ORDER BY Company DESC

Το αποτέλεσμα θα είναι:

Company	OrderNumber
Olympic	6978
Olympic	2312
Easyjet	4678
Aegean	3412

Η Εντολή INSERT INTO
Αυτή η εντολή εισάγει νέες γραμμές σ'έναν πίνακα. Η σύνταξή της έχει ως εξής:

INSERT INTO όνομα_πίνακα

VALUES (τιμή1, τιμή2, ...)

Μπορούμε επίσης να καθορίσουμε τις στήλες για τις οποίες θέλουμε να εισάγουμε δεδομένα:

INSERT INTO όνομα_πίνακα (στήλη1, στήλη2, ...)

VALUES (τιμή1, τιμή2, ...)

Ο επόμενος πίνακας "People":

LastName	FirstName	Address	City
Κωνσταντίνου	Μαρία	Καλλέργη 15	Αθήνα

και η εντολή SQL που ακολουθεί:
INSERT INTO People

VALUES ('Κιουράνης', 'Ιωάννης', 'Π.Μελά 90', 'Θεσ/νίκη')
δίνουν αυτό το αποτέλεσμα:

LastName	FirstName	Address	City
Κωνσταντίνου	Μαρία	Καλλέργη 15	Αθήνα
Κιουράνης	Ιωάννης	Π.Μελά 90	Θεσ/νίκη

Εισαγωγή Δεδομένων σε Συγκεκριμένες Στήλες
Ο επόμενος πίνακας "People":

LastName	FirstName	Address	City
Κωνσταντίνου	Μαρία	Καλλέργη 15	Αθήνα
Κιουράνης	Ιωάννης	Π.Μελά 90	Θεσ/νίκη

και η εντολή SQL που ακολουθεί:
INSERT INTO People (LastName, Address)

VALUES ('Νικολάου', 'Ταγμ. Παύλου 30')
δίνουν αυτό το αποτέλεσμα:

LastName	FirstName	Address	City
Κωνσταντίνου	Μαρία	Καλλέργη 15	Αθήνα
Κιουράνης	Ιωάννης	Π. Μελά 90	Θεσ/νίκη
Νικολάου		Ταγμ. Παύλου 30	

Η Εντολή Update
Η εντολή *UPDATE* ενημερώνει ή αλλάζει γραμμές. Η σύνταξή της είναι ως εξής:

UPDATE όνομα_πίνακα SET όνομα_στήλης=νέα_τιμή

WHERE όνομα_στήλης=τιμή

Ακολουθεί ο τροποποιημένος πίνακας "People":

LastName	FirstName	Address	City
Κωνσταντίνου	Μαρία	Καλλέργη 15	Αθήνα
Κιουράνης	Ιωάννης	Π. Μελά 90	Θεσ/νίκη
Νικολάου		Ταγμ. Παύλου 30	

Ενημέρωση μίας στήλης σε μια γραμμή
Θέλουμε να προσθέσουμε ένα όνομα στο άτομο που έχει το επώνυμο "Νικολάου":

UPDATE People SET FirstName= 'Αθήνα'

WHERE LastName= 'Νικολάου'

Ενημέρωση πολλών στηλών σε μία γραμμή
Για το ίδιο άτομο θέλουμε να αλλάξουμε τη διεύθυνση και να προσθέσουμε ένα όνομα για την πόλη:

UPDATE People

SET Address= 'Χίου 12', City= 'Αθήνα'

WHERE LastName= 'Νικολάου'

Το αποτέλεσμα θα είναι:

LastName	FirstName	Address	City
Κωνσταντίνου	Μαρία	Καλλέργη 15	Αθήνα
Κιουράνης	Ιωάννης	Π. Μελά 90	Θεσ/νίκη
Νικολάου	Αθηνά	Χίου 12	Αθήνα

Η Εντολή Delete
Η εντολή *DELETE* χρησιμοποιείται για να διαγράψουμε γραμμές απο έναν πίνακα. Η σύνταξή της είναι η ακόλουθη:

DELETE FROM όνομα_πίνακα

WHERE όνομα_στήλης= τιμή

Ο τροποποιημένος πίνακας "People" από την τελευταία φορά είναι ο ακόλουθος:

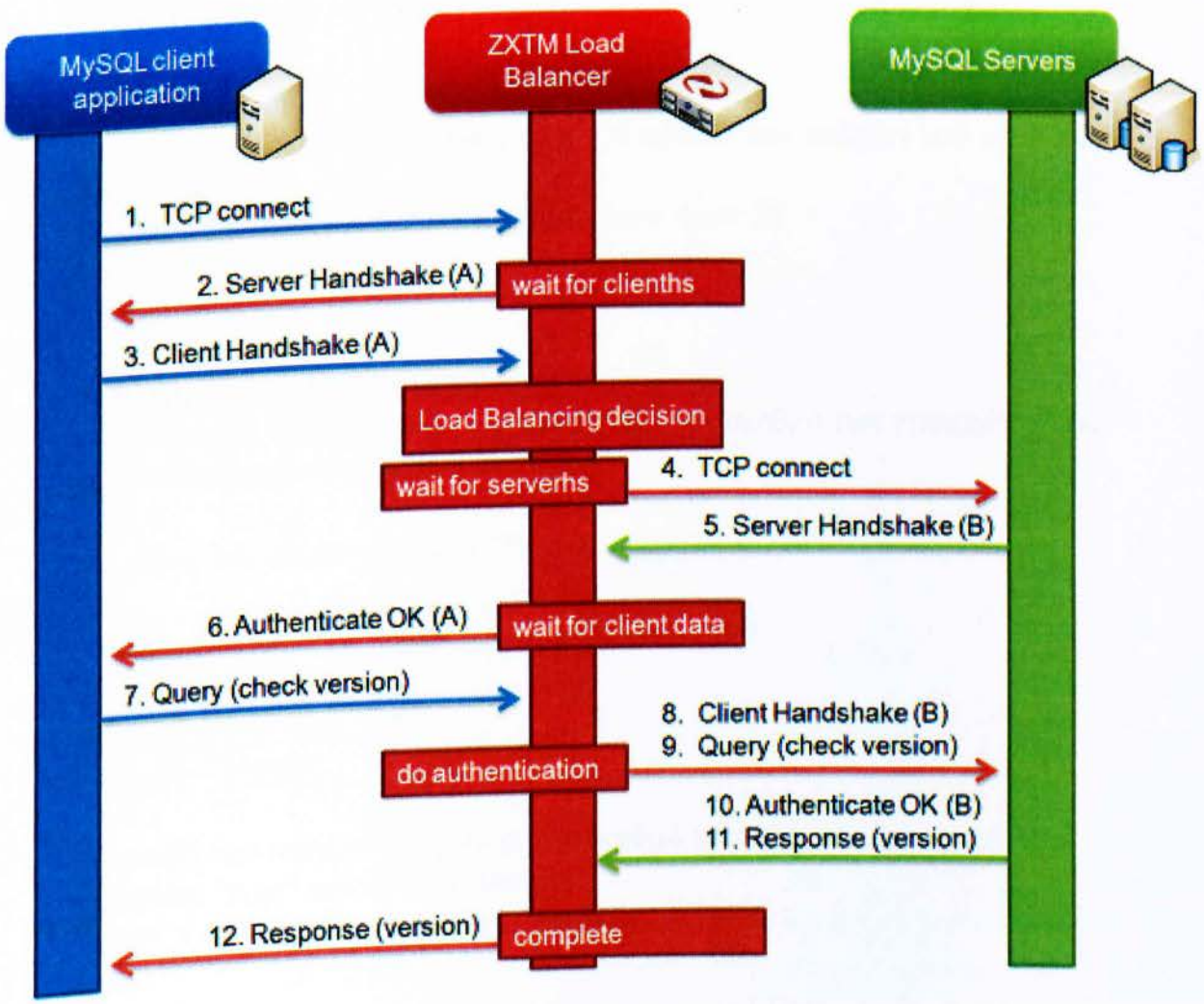
LastName	FirstName	Address	City
Κωνσταντίνου	Μαρία	Καλλέργη 15	Αθήνα
Κιουράνης	Ιωάννης	Π. Μελά 90	Θεσ/νίκη
Νικολάου	Αθηνά	Χίου 12	Αθήνα

Διαγραφή μιας γραμμής
Θα διαγράψουμε την "Νικολάου Αθηνά":

DELETE FROM People WHERE LastName= 'Νικολάου'

Το αποτέλεσμα θα είναι:

LastName	FirstName	Address	City
Κωνσταντίνου	Μαρία	Καλλέργη 15	Αθήνα
Κιουράνης	Ιωάννης	Π. Μελά 90	Θεσ/νίκη



Οι Συναρτήσεις Count της SQL
Η SQL έχει ενσωματωμένες συναρτήσεις για την μέτρηση(*counting*) των εγγράφων μίας βάσης δεδομένων.

Η σύνταξη για τις ενσωματωμένες συναρτήσεις *COUNT* είναι η εξής:

SELECT COUNT Στήλη FROM πίνακας

1) Η Συνάρτηση COUNT (*)
Η συνάρτηση *COUNT(*)* επιστρέφει τον αριθμό των επιλεγμένων γραμμών από μια επιλογή(*selection*).

Με τον ακόλουθο πίνακα "People":

Name	**Age**
Αντωνιάδης Ηλίας	34
Μάρκου Ιωάννης	45
Ιωάννου Γεώργιος	19

Η επόμενη εντολή επιστρέφει τον αριθμό των γραμμών του ανωτέρω πίνακα:
SELECT COUNT (*) FROM People

Αποτέλεσμα: *3*
, ενώ η ακόλουθη εντολή επιστρέφει τον αριθμό των ατόμων που είναι πάνω από 20 χρονών:
SELECT COUNT (*) FROM People where Age>20

Αποτέλεσμα: *2*

2) Η Συνάρτηση COUNT (column)
Η συνάρτηση *COUNT(column)* επιστρέφει τον αριθμό των γραμμών χωρίς τιμή *NULL* στην συγκεκριμένη στήλη.

Με βάση τον επόμενο πίνακα "People":

Name	**Age**
Αντωνιάδης Ηλίας	34
Μάρκου Ιωάννης	45
Ιωάννου Γεώργιος	

Η εντολή που ακολουθεί βρίσκει τον αριθμό των ατόμων που έχουν τιμή στο πεδίο "Age" του πίνακα "People":
SELECT COUNT (Age) FROM People

Αποτέλεσμα: *2*

Η συνάρτηση *COUNT(column)* είναι χρήσιμη για να βρίσκουμε τις στήλες που δεν έχουν τιμή. Το αποτέλεσμα είναι κατά ένα λιγότερο από τον αριθμό των γραμμών του αρχικού πίνακα επειδή ένα από τα άτομα δεν έχει τιμή στο πεδίο Age.

Οι Λέξεις Κλειδιά COUNT και DISTINCT

Οι εν λόγω λέξεις κλειδιά μπορούν να χρησιμοποιηθούν μαζί για να μετρήσουμε τον αριθμό των διακριτών αποτελεσμάτων.

Η σύνταξη έχει ως εξής:

SELECT DISTINCT COUNT Στήλες FROM πίνακας

Με τον πίνακα "Orders", όπως τον είχαμε δεί και στις προηγούμενες ενότητες:

Company	**OrderNumber**
Aegean	3412
Olympic	2312
Easyjet	4678
Olympic	6798

και την ακόλουθη εντολή SQL:
SELECT COUNT (Company) FROM Orders
, θα δώσει το αποτέλεσμα: ***4***
ενώ η επόμενη εντολή SQL:
SELECT DISTINCT COUNT (Company) FROM Orders
, θα δώσει το αποτέλεσμα: ***3***
Υπάρχουν όμως και άλλες λέξεις-κλειδιά που μπορούμε να χρησιμοποιήσουμε όταν προγραμματίζουμε μία βάση δεδομένων και οι οποίες μπορεί να αποδειχθούν ιδιαιτέρως χρήσιμες, όπως είναι οι ακόλουθες˙ τις οποίες και θα αναλύσουμε παρακάτω:

- **NOT NULL**
- **AUTO_INCREMENT**
- **PRIMARY KEY** και η
- **UNSIGNED**

Η **NOT NULL** σημαίνει ότι όλες οι γραμμές του πίνακα πρέπει να έχουν μία τιμή σε αυτή την ιδιότητα. Αν δεν καθοριστεί, το πεδίο μπορεί να είναι κενό(NULL). Μπορούμε να δηλώνουμε τις στήλες ενός πίνακα ως NOT NULL. Αυτή θα είναι μία μικρή βελτιστοποίηση που θα μπορούμε να κάνουμε όποτε είναι δυνατόν, η οποία θα βελτιώνει την απόδοση.

Η **AUTO_INCREMENT** είναι μία ειδική λειτουργία της MySQL, που μπορούμε να χρησιμοποιήσουμε σε ακέραιες στήλες. Αν δηλαδή, αφήσουμε αυτό το πεδίο κενό όταν εισάγουμε γραμμές στον πίνακα, η MySQL θα δημιουργήσει αυτόματα ένα μοναδικό χαρακτηριστικό. Η τιμή θα είναι κατά 1 μεγαλύτερη από την μεγαλύτερη τιμή της στήλης. Μπορούμε να έχουμε μόνο μία τέτοια στήλη σε κάθε πίνακα. Οι στήλες που είναι AUTO_INCREMENT πρέπει να έχουν δείκτη.

Η **PRIMARY KEY** μετά από ένα όνομα στήλης καθορίζει ότι αυτή η στήλη είναι το πρωτεύον κλειδί του πίνακα. Τα στοιχεία αυτής της στήλης θα πρέπει να είναι μοναδικά. Η MySQL θα βάλει αυτόματα σε ευρετήριο αυτή τη στήλη. Το αυτόματο ευρετήριο για το πρωτεύον κλειδί φροντίζει επίσης και για το ευρετήριο που απαιτείται από το AUTO_INCREMENT. Η δημιουργία ευρετηρίων μας βοηθάει στην καλύτερη απόδοση της βάσης δεδομένων μας. Τα ευρετήρια που δημιουργήθηκαν αυτόματα είναι επαρκή. Εάν "τρέχουμε" πολλά ερωτήματα σε μία στήλη που δεν είναι κλειδί, μπορούμε να προσθέσουμε ένα ευρετήριο σε αυτήν τη στήλη για να βελτιώσουμε την απόδοσή της, όπως προαναφέραμε.

Και τέλος η **UNSIGNED** μετά από έναν ακέραιο τύπο σημαίνει ότι μπορεί να έχει μόνο μία μηδενική ή θετική απροσήμαστη τιμή.

Η Λέξη Κλειδί Group By

Η λέξη κλειδί *GROUP BY* έχει προστεθεί στην SQL επειδή οι αθροιστικές συναρτήσεις(*aggregate functions*), όπως είναι η *SUM*, επιστρέφουν το σύνολο όλων των τιμών μιας στήλης κάθε φορά που καλούνται.

Χωρίς την παραπάνω λέξη κλειδί, το να βρούμε το άθροισμα για κάθε ανεξάρτητη ομάδα τιμών μιας στήλης θα ήταν αδύνατο.

Η ανωτέρω λέξη κλειδί συντάσσεται ως εξής:

SELECT στήλη, SUM στήλη FROM πίνακας GROUP BY στήλη

Παράδειγμα:
Έστω ο παρακάτω πίνακας "Orders":

Company	**Amount**
Olympic	5500
Easyjet	4500
Olympic	7100

και με την χρήση της επόμενης εντολής SQL:

SELECT Company, SUM (Amount) FROM Orders
θα δοθεί το ακόλουθο αποτέλεσμα:

Company	**SUM(Amount)**
Olympic	17100
Easyjet	17100
Olympic	17100

Επισημαίνουμε όμως ότι ο παραπάνω κώδικας δεν είναι έγκυρος επειδή η στήλη που επιστρέφει δεν αποτελεί μέρος ενός αθροίσματος(*aggregate*). Ένα *GROUP BY clause* μπορεί να το διορθώσει αυτό, ως εξής:

SELECT Company, SUM (Amount) FROM Orders

GROUP BY Company

δίνοντας το ακόλουθο αποτέλεσμα:

Company	**SUM(Amount)**
Olympic	12600
Easyjet	4500

Η Λέξη Κλειδί HAVING
Η λέξη κλειδί *HAVING* έχει προστεθεί στην SQL επειδή η λέξη κλειδί *WHERE* δεν μπορεί να χρησιμοποιηθεί σε αθροιστικές συναρτήσεις, όπως είναι η *SUM*.

Η σύνταξη της συνάρτησης *HAVING* είναι η εξής:

SELECT στήλη, SUM στήλη FROM πίνακας

GROUP BY στήλη

HAVING SUM στήλη συνθήκη

Ο παρακάτω πίνακας "Orders":

Company	**Amount**
Olympic	5500
Easyjet	4500
Olympic	7100

και αυτή η εντολή SQL:

SELECT Company, SUM (Amount) FROM Orders

GROUP BY Company HAVING SUM (Amount)>10000

θα δώσουν αυτό το αποτέλεσμα:

Company	SUM(Amount)
Olympic	12600

Τα Ψευδώνυμα(Aliases)

Στην SQL, τα ψευδώνυμα(*aliases*) χρησιμοποιούνται για ονόματα στηλών και πινάκων.

- Ψευδώνυμο Στήλης(Column Name Alias)

Η σύνταξη είναι ως εξής:

SELECT στήλη AS column_alias FROM πίνακας

- Ψευδώνυμο Πίνακα(Table Name Alias)

Η σύνταξη είναι ως εξής:

SELECT στήλη FROM πίνακας AS table_alias

Παράδειγμα:

Ο επόμενος πίνακας "People":

LastName	FirstName	Address	City
Κωνσταντίνου	Μαρία	Καλλέργη 15	Αθήνα
Κιουράνης	Ιωάννης	Π. Μελά 90	Θεσ/νίκη
Νικολάου	Αθηνά	Χίου 12	Αθήνα

και η εξής εντολή SQL:

SELECT LastName AS Family, FirstName AS Name

FROM People

δίνουν αυτό το αποτέλεσμα:

Family	Name
Κωνσταντίνου	Μαρία
Κιουράνης	Ιωάννης
Νικολάου	Αθήνα

Παράδειγμα:

Ο επόμενος πίνακας "People":

LastName	FirstName	Address	City
Κωνσταντίνου	Μαρία	Καλλέργη 15	Αθήνα
Κιουράνης	Ιωάννης	Π. Μελά 90	Θεσ/νίκη
Νικολάου	Αθήνα	Χίου 12	Αθήνα

και η ακόλουθη εντολή SQL:

SELECT LastName, FirstName

FROM People AS Employees

δίνουν αυτό το αποτέλεσμα:

LastName	FirstName
Κωνσταντίνου	Μαρία
Κιουράνης	Ιωάννης
Νικολάου	Αθήνα

Εν κατακλείδι συνοψίζοντας όλες τις παραπάνω ενότητες καταλήγουμε ότι υπάρχουν πέντε είδη αναγνωριστικών στην MySQL:

- **Βάσεις δεδομένων**
- **Πίνακες**
- **Στήλες**
- **Ευρετήρια** και
- **Ψευδώνυμα**

Οι βάσεις δεδομένων της MySQL απεικονίζονται σε καταλόγους μέσα στην δομή των αρχείων και οι πίνακες απεικονίζονται σε αρχεία. Αυτή η απεικόνιση έχει άμεση επίδραση στα ονόματα που τους δίνουμε. Επηρεάζει επίσης την ευαισθησία κεφαλαίων-πεζών αυτών των ονομάτων. Αν ο κατάλογος και τα ονόματα των αρχείων είναι ευαίσθητα σε κεφαλαία-πεζά στο λειτουργικό σας σύστημα, τα ονόματα της βάσης δεδομένων και των πινάκων θα είναι επίσης ευαίσθητα σε κεφαλαία-πεζά (για παράδειγμα, στο UNIX), διαφορετικά δεν θα είναι (για παράδειγμα, στα Windows). Τα ονόματα των στηλών και τα ψευδώνυμα δεν είναι ευαίσθητα σε κεφαλαία-πεζά, αλλά δεν μπορούμε να γράψουμε τα ίδια ονόματα με διαφορετικό τρόπο σε σχέση με τα κεφαλαία-πεζά στην ίδια SQL πρόταση.

Πρέπει να ξέρουμε ότι, η θέση του καταλόγου και των αρχείων που περιέχουν τα δεδομένα θα είναι που ορίστηκε στην διαμόρφωση. Μπορούμε να ελέγξουμε τη θέση στο σύστημα μας, χρησιμοποιώντας το βοηθητικό πρόγραμμα mysqladmin, ως εξής:

Mysqladmin variables και ψάξτε για την μεταβλητή datadir

Στον παρακάτω πίνακα συνοψίζονται τα ανωτέρω:

Τύπος	Μέγιστο Μήκος	Ευαισθησία Σε Πεζά-Κεφαλαία	Επιτρεπόμενοι Χαρακτήρες
Βάση Δεδομένων	64	Όπως το λειτουργικό σύστημα	Οτιδήποτε επιτρέπεται σε ένα όνομα καταλόγου στο λειτουργικό μας σύστημα, εκτός από τους χαρακτήρες /, \ και .
Πίνακας	64	Όπως το λειτουργικό σύστημα	Οτιδήποτε επιτρέπεται σε ένα όνομα καταλόγου στο λειτουργικό μας σύστημα, εκτός από τους χαρακτήρες / και .
Στήλη	64	Όχι	Οτιδήποτε
Ευρετήριο	64	Όχι	Οτιδήποτε
Ψευδώνυμο	255	Όχι	Οτιδήποτε

Πίνακας 2:Χαρακτηριστικά της MySQL

Οι Συναρτήσεις της SQL

Συνήθως θέλουμε να ξέρουμε πόσες γραμμές ανήκουν σε ένα συγκεκριμένο σύνολο ή τη μέση τιμή κάποιας στήλης. Η MySQL έχει ένα σύνολο από συνοπτικές ενσωματωμένες συναρτήσεις για να μπορούμε να κάνουμε μετρήσεις(*counting*) και υπολογισμούς(*calculations*) και που μας είναι χρήσιμες για να απαντηθεί αυτό το είδος ερωτήματος.

Αυτές οι συνοπτικές συναρτήσεις μπορούν να εφαρμοστούν σε ένα πίνακα συνολικά ή σε ομάδες δεδομένων του πίνακα.

Η γενική σύνταξη για τις ενσωματωμένες συναρτήσεις της SQL είναι η εξής:

SELECT function (στήλη) FROM πίνακας

Οι πιο συνηθισμένες συναρτήσεις αναφέρονται στον παρακάτω πίνακα:

Όνομα	Περιγραφή
AVG (column)	Μέσος όρος των τιμών μιάς συγκεκριμένης στήλης σε μια επιλογή. Οι τιμές NULL δεν περιλαμβάνονται στους υπολογισμούς.
Count (items)	Όπως έχουμε αναφερθεί και σε προηγούμενη ενότητα(βλ. Σελ 89-90), αν καθορίσουμε μία στήλη, θα μας δώσει τον αριθμό των μη κενών τιμών αυτής της στήλης. Αν προσθέσετε την λέξη κλειδί DISTINCT εμπρός απο το όνομα της στήλης θα πάρετε μια καταμέτρηση των διαφορετικών τιμών αυτής της στήλης. Αν καθορίσετε το COUNT(*), θα πάρετε μια καταμέτρηση των γραμμών, ανεξάρτητα απο τις NULL τιμές.
MIN (column)	Επιστρέφει την ελάχιστη τιμή μιας στήλης. Οι τιμές NULL δεν περιλαμβάνονται στον υπολογισμό.(βλ. #)
MAX (column)	Επιστρέφει την μέγιστη τιμή μιας στήλης. Οι τιμές NULL δεν περιλαμβάνονται στον υπολογισμό.(βλ. #)
STD (column)	Τυπική απόκλιση των τιμών σε μια συγκεκριμένη στήλη.
STDDEVF (column)	Ίδιο με το STD (column)
SUM (column)	Επιστρέφει το άθροισμα των τιμών μιας στήλης για μια συγκεκριμένη επιλογή. Οι τιμές NULL δεν περιλαμβάνονται στον υπολογισμό.

(#). Οι συναρτήσεις MIN & MAX μπορούν επίσης να χρησιμοποιηθούν σε στήλες που περιέχουν κείμενο(text), για να βρούμε την μεγαλύτερη ή μικρότερη τιμή σε αλφαβητική σειρά.

Πίνακας 3: Συνοπτικές συναρτήσεις της MySQL.

Ακολουθούν μερικά παραδείγματα προκειμένου να γίνουν πιο κατανοητά τα ανωτέρω.

Έστω ο αρχικός πίνακας "People":

Name	Age
Αντωνιάδης Ηλίας	34
Μάρκου Ιωάννης	45
Ιωάννου Γιώργος	19

- και η χρήση της παρακάτω εντολής επιστρέφει τον μέσο όρο ηλικίας των ατόμων που υπάρχουν στον πίνακα "People":

SELECT AVG (Age) FROM People

Αποτέλεσμα: ***32.67***

- και η χρήση της παρακάτω εντολής επιστρέφει τον μέσο όρο ηλικίας των ατόμων που είναι πάνω από 20 χρονών, του πίνακα "People":

SELECT AVG (Age) FROM People where Age>20

Αποτέλεσμα: ***39.5***

- και η χρήση της παρακάτω εντολής επιστρέφει την μεγαλύτερη ηλικία των ατόμων του πίνακα "People":

SELECT MAX (Age) FROM People

Αποτέλεσμα: ***45***

- και η χρήση της παρακάτω εντολής επιστρέφει την μικρότερη ηλικία των ατόμων του πίνακα "People":

SELECT MIN (Age) FROM People

Αποτέλεσμα: ***19***

- και η χρήση της παρακάτω εντολής επιστρέφει το άθροισμα όλων των ηλικιών του πίνακα "People":

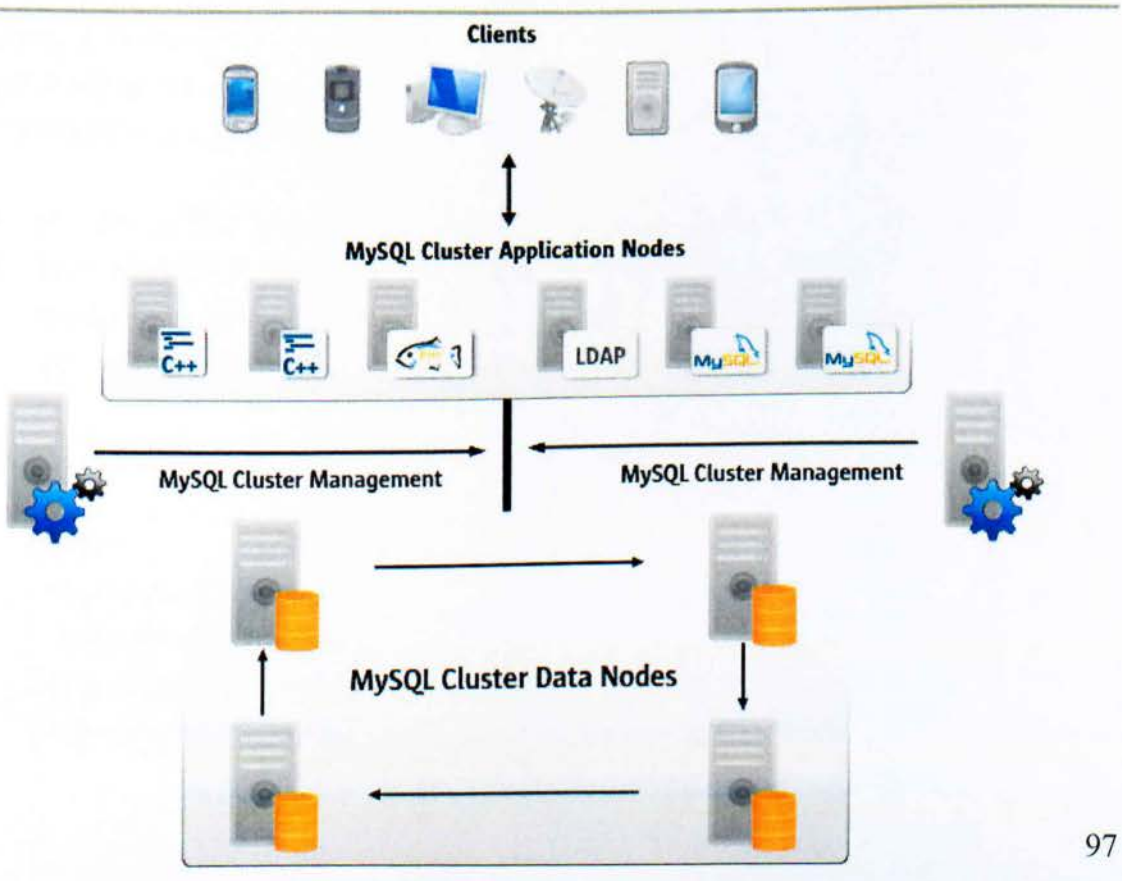
SELECT SUM (Age) FROM People

Αποτέλεσμα: ***98***

- και η χρήση της παρακάτω εντολής επιστρέφει το άθροισμα των ηλικιών των ατόμων που είναι πάνω από 20 χρονών, του πίνακα "People":

SELECT SUM (Age) FROM People where Age>20

Αποτέλεσμα: ***79***



Ένωση Πινάκων(Join)

Συνήθως για να απαντήσουμε σε μια ερώτηση από την βάση δεδομένων, θα πρέπει να χρησιμοποιήσουμε δεδομένα από περισσότερους από έναν πίνακες. Αυτά τα δεδομένα είναι σε διαφορετικούς πίνακες, επειδή σχετίζονται με διαφορετικά πραγματικά αντικείμενα.

Για να επιλέξουμε λοιπόν δεδομένα από δύο πίνακες προκειμένου να δημιουργήσουμε ένα πιο πολύπλοκο αποτέλεσμα και να βάλουμε όλες αυτές τις πληροφορίες μαζί στην MySQL, πρέπει να εκτελέσουμε μια λειτουργία που ονομάζεται **ένωση**(*join*).

Αυτό απλώς σημαίνει την ένωση δύο ή περισσοτέρων πινάκων που ορίζει τις σχέσεις μεταξύ των δεδομένων. *Αν και οι ενώσεις είναι απλές θεωρητικά, είναι ένα από τα πιο λεπτά και πολύπλοκα μέρη της MySQL.* Υπάρχουν διαφορετικοί τύποι ενώσεων στην MySQL και κάθε μία χρησιμοποιείται για διαφορετικό σκοπό. Η ένωση περισσοτέρων απο δύο πινάκων δεν είναι πιο δύσκολη από μια ένωση δύο πινάκων. Ως γενικό κανόνα, πρέπει να ενώσουμε πίνακες σε ζευγάρια, με συνθήκες ένωσης ακολουθώντας τις σχέσεις μεταξύ των δεδομένων, από πίνακα σε πίνακα.

Οι πίνακες μιας βάσης δεδομένων μπορούν να συσχετιστούν μεταξύ τους με **κλειδιά**(*keys*). Όπως έχουμε αναφέρει και παραπάνω, ένα **πρωτεύον κλειδί**(*primary key*) είναι μια στήλη με μία μοναδική τιμή στην κάθε γραμμή. Ο σκοπός είναι να ενώσει τα δεδομένα μαζί από διάφορους πίνακες, χωρίς να έχουμε επανάληψη όλων των δεδομένων σε κάθε πίνακα.

Στον πίνακα "Members" που ακολουθεί, η στήλη "ID" είναι το πρωτεύον κλειδί(*primary key*), που σημαίνει ότι δεν μπορούν να υπάρχουν δύο γραμμές που να έχουν το ίδιο ID. Το ID θα είναι αυτό που ξεχωρίζει δύο άτομα ακόμη και αν έχουν το ίδιο όνομα.
Στους πίνακες παραδειγμάτων που ακολουθούν, προσέχουμε τα εξής:

- Η στήλη "ID" είναι το πρωτεύον κλειδί του πίνακα "Members".
- Η στήλη "ID" του πίνακα "Orders" χρησιμοποιείται για να αναφερόμαστε στα άτομα του πίνακα "Members", χωρίς να χρησιμοποιούμε τα ονόματά τους.

Πίνακας "Members":

ID	**Name**
01	Νικολάου Αθηνά
02	Γεωργιάδης Γεώργιος
03	Στεφανάκος Στέφανος
04	Σταματίου Σταμάτης

Πίνακας "Orders":

ID	Product
01	Printer

Ένας άλλος κύριος τύπος ένωσης που χρησιμοποιούμε στη MySQL, είναι η **αριστερή ένωση**. Στις περιπτώσεις όπου δεν χρησιμοποιούσαμε αυτήν την ένωση εμφανίζονταν μόνο οι γραμμές για τις οποίες υπήρχε ένα ταίριασμα μεταξύ των πινάκων. Μερικές φορές όμως, θέλουμε να δούμε τις γραμμές στις οποίες δεν υπάρχει ταίριασμα όπως για παράδειγμα στην ιστοσελίδα μας, να βρούμε τους πελάτες που δεν έχουν κάνει ποτέ κράτηση εισητηρίων ή να βρούμε τις θέσεις στην αίθουσα που δεν έχουν ποτέ κρατηθεί.

Ένας τρόπος να απαντήσουμε αυτό το ερώτημα στην MySQL είναι να χρησιμοποιήσουμε αριστερή ένωση. Μία αριστερή ένωση θα ταιριάξει γραμμές μέσω μίας συγκεκριμένης συνθήκης ένωσης μεταξύ δύο πινάκων. Αν δεν υπάρχει γραμμή ταιριάσματος στον δεξιό πίνακα, θα προστεθεί μία γραμμή στο αποτέλεσμα που περιέχει τιμές NULL στην δεξιά στήλη.

Παράδειγμα:

```
SELECT members.member_id, member.LastName, orders.order_id

FROM members LEFT JOIN orders

ON members.member_id=ordrs.member_id;
```

Το παραπάνω SQL ερώτημα χρησιμοποιεί αριστερή ένωση για να ενώσει τους πελάτες με τις κρατήσεις εισητηρίων. Η αριστερή ένωση χρησιμοποιεί μία διαφορετική σύνταξη για την συνθήκη ένωσης. Σε αυτή την περίπτωση, η συνθήκη ένωσης χρησιμοποιεί τον ειδικό όρο ON της SQL πρότασης.

Αν θέλουμε να δούμε μόνο τους πελάτες που δεν έχουν κάνει κράτηση ποτέ, θα πρέπει να ελέγξουμε αν είναι NULL το πεδίο του πρωτεύοντος κλειδίου του δεξιού πίνακα(σε αυτήν την περίπτωση, το order_id), επειδή δεν θα είναι NULL στις υπάρχουσες γραμμές:

```
SELECT members.member_id, member.LastName

FROM members LEFT JOIN orders

USING (member_id)

WHERE orders.order_id is null;
```

Στο ανωτέρω παράδειγμα, χρησιμοποιούμε μία διαφορετική σύνταξη για την συνθήκη ένωσης. Οι αριστερές ενώσεις υποστηρίζουν είτε τη σύνταξη ON που χρησιμοποιήσαμε στο πρώτο παράδειγμα, είτε την σύνταξη USING του δεύτερου παραδείγματος. Η σύνταξη USING δεν καθορίζει τον πίνακα από τον οποίο προέρχεται η ιδιότητα ένωσης. Για αυτό τον λόγο, οι στήλες στους δύο πίνακες πρέπει να έχουν το ίδιο όνομα αν θέλουμε να χρησιμοποιήσουμε την USING.

Αποθηκευμένες Διαδικασίες και Συναρτήσεις

Οι *αποθηκευμένες διαδικασίες* είναι ομαδοποιήσεις σχετικών προτάσεων SQL, που συνήθως αναφέρονται σαν *συναρτήσεις* και *υποπρογράμματα*, που παρέχουν ευκολία και ευελιξία σε έναν προγραμματιστή. Η ευκολία και η ευελιξία προέρχονται από το γεγονός ότι μια αποθηκευμένη διαδικασία συχνά εκτελείται ευκολότερα από ένα αριθμό μεμονωμένων προτάσεων SQL. Οι αποθηκευμένες διαδικασίες μπορούν να ενταθούν μέσα σε άλλες αποθηκευμένες διαδικασίες. Αυτό σημαίνει ότι, μία αποθηκευμένη διαδικασία μπορεί να καλεί μία άλλη αποθηκευμένη διαδικασία, η οποία και αυτή με την σειρά της να καλεί μία άλλη αποθηκευμένη διαδικασία κοκ.

Οι αποθηκευμένες διαδικασίες μας επιτρέπουν να έχουμε διαδικαστικό προγραμματισμό. Οι βασικές προτάσεις SQL, DDL, DML και DQL(*CREATE TABLE, INSERT, UPDATE, SELECT* κτλ.) μας επιτρέπουν να λέμε στην βάση δεδομένων τι πρέπει να γίνει, αλλά όχι πως να το κάνει. Κωδικοποιώντας αποθηκευμένες διαδικασίες, λέμε στην μηχανή της βάσης δεδομένων πως να κάνει την επεξεργασία των δεδομένων.

Μια αποθηκευμένη διαδικασία είναι μία ομάδα μιας ή περισσοτέρων προτάσεων ή συναρτήσεων SQL που αποθηκεύονται μέσα στην βάση δεδομένων, μεταγλωττίζονται και είναι έτοιμες για εκτέλεση από ένα χρήστη της βάσης δεδομένων. Μια αποθηκευμένη συνάρτηση είναι το ίδιο με μια αποθηκευμένη διαδικασία, αλλά μία συνάρτηση χρησιμοποιείται για να επιστρέψει μια τιμή.

Οι συναρτήσεις καλούνται από διαδικασίες. Όταν μία συνάρτηση καλείται από μία διαδικασία, μπορούν να περνούν παράμετροι σε μία συνάρτηση, όπως σε μία διαδικασία, να υπολογίζεται μια τιμή και μετά η τιμή να περνά πίσω στην καλούσα διαδικασία για περαιτέρω επεξεργασία.

Όταν δημιουργείται μια αποθηκευμένη διαδικασία, τα διάφορα υποπρογράμματα και συναρτήσεις(που χρησιμοποιούν SQL), τα οποία απαρτίζουν την αποθηκευμένη διαδικασία αποθηκεύονται μέσα στην βάση δεδομένων. Αυτές οι αποθηκευμένες διαδικασίες αναλύονται εκ των προτέρων και είναι άμεσα έτοιμες για εκτέλεση, όταν καλούνται από τον χρήστη.

Η σύνταξη σε MySQL Server για δημιουργία μιας αποθηκευμένης διαδικασίας είναι:

CREATE PROCEDURE όνομα_διαδικασίας
[[({ &όνομα_παραμέτρου
DATATYPE [(μήκος) | (ακρίβεια] [, δεκαδικά])
[= DEFAULT][ΕΞΟΔΟΣ]]
[, @όνομα_παραμέτρου
DATATYPE [(μήκος) | (ακρίβεια] [, δεκαδικά])
[= DEFAULT][ΕΞΟΔΟΣ]] [ )]]
[WITH RECOMPILE]
AS προτάσεις_sql

Η σύνταξη για εκτέλεση μιας αποθηκευμένης διαδικασίας στην MySQL Server είναι η ακόλουθη:

EXECUTE [@ RETURN_STATUS =]
Όνομα_διαδικασίας
[[@όνομα_παραμέτρου=] ΤΙΜΗ |
[@όνομα_παραμέτρου=] @μεταβλητή [ΕΞΟΔΟΣ]]
[WITH RECOMPILE]

Πλεονεκτήματα Αποθηκευμένων Διαδικασιών και Συναρτήσεων
Οι αποθηκευμένες διαδικασίες παρέχουν αρκετά διακριτά αποτελέσματα επί των μεμονωμένων προτάσεων SQL που εκτελούνται στη βάση δεδομένων. Ορισμένα από τα πλεονεκτήματα περιλαμβάνουν τα παρακάτω:

- Οι προτάσεις είναι ήδη αποθηκευμένες μέσα στην βάση δεδομένων.
- Οι προτάσεις έχουν ήδη αναλυθεί και είναι σε εκτελέσιμη μορφή.
- Οι αποθηκευμένες διαδικασίες υποστηρίζουν αρθρωτό προγραμματισμό.
- Οι αποθηκευμένες διαδικασίες μπορούν να καλούν άλλες διαδικασίες και συναρτήσεις.
- Οι αποθηκευμένες διαδικασίες μπορούν να καλούνται από άλλους τύπους προγραμμάτων.
- Ο συνολικός χρόνος απόκρισης είναι τυπικά καλύτερος όταν χρησιμοποιούμε αποθηκευμένες διαδικασίες.
- Υπάρχει συνολικά ευκολία χρήσης.

Triggers
Ένα trigger είναι μία μεταγλωττισμένη διαδικασία SQL μέσα στην βάση δεδομένων, που χρησιμοποιείται για να εκτελεί ενέργειες με βάση άλλες ενέργειες, που συμβαίνουν μέσα στην βάση δεδομένων. Ένα trigger είναι

μια μορφή αποθηκευμένης διαδικασίας που εκτελείται όταν ορίζεται μία συγκεκριμένη ενέργεια(γλώσσας διαχείρισης δεδομένων) σε ένα πίνακα. Το trigger μπορεί να εκτελεστεί πρίν ή μετά μίας εντολής INSERT, DELETE ή UPDATE. Τα triggers μπορούν επίσης να χρησιμοποιηθούν για έλεγχο της ακεραιότητας δεδομένων πρίν από μία εντολή όπως οι ανωτέρω. Τα triggers μπορούν να αναιρέσουν συναλλαγές και μπορούν να τροποποιήσουν δεδομένα μέσα σε ένα πίνακα και να διαβάσουν από έναν άλλο πίνακα σε μία άλλη βάση δεδομένων.

Η Πρόταση CREATE TRIGGER

Η σύνταξη σε MySQL Server για την δημιουργία ενός trigger είναι:

CREATE TRIGGER όνομα_trigger
ON όνομα_πίνακα
FOR {INSERT | UPDATE | DELETE [, ...]}
AS προτάσεις_sql
[RETURN]

Το σώμα ενός trigger δεν μπορεί να αλλάξει. Πρέπει είτε να αντικαταστήσετε ή να αναδημιουργήσετε το trigger. Ορισμένες υλοποιήσεις επιτρέπουν την αντικατάσταση ενός trigger(με την προϋπόθεση ότι το trigger με το ίδιο όνομα υπάρχει ήδη) σαν μέρος της πρότασης *CREATE TRIGGER.*

Η Πρόταση DROP TRIGGER

Ένα trigger μπορεί να απορριφθεί χρησιμοποιώντας την πρόταση *DROP TRIGGER.* Η σύνταξη για την απόρριψη ενός trigger είναι η ακόλουθη:

DROP TRIGGER όνομα_trigger

Στο ΚΕΦΑΛΑΙΟ IV που ακολουθεί γίνεται παρουσίαση από την υλοποιημένη ιστοσελίδα και αναλύεται η δομή και η λειτουργία της καθώς επίσης και πως αλληλεπιδρά με τους χρήστες που την χρησιμοποιούν.

ΚΕΦΑΛΑΙΟ IV

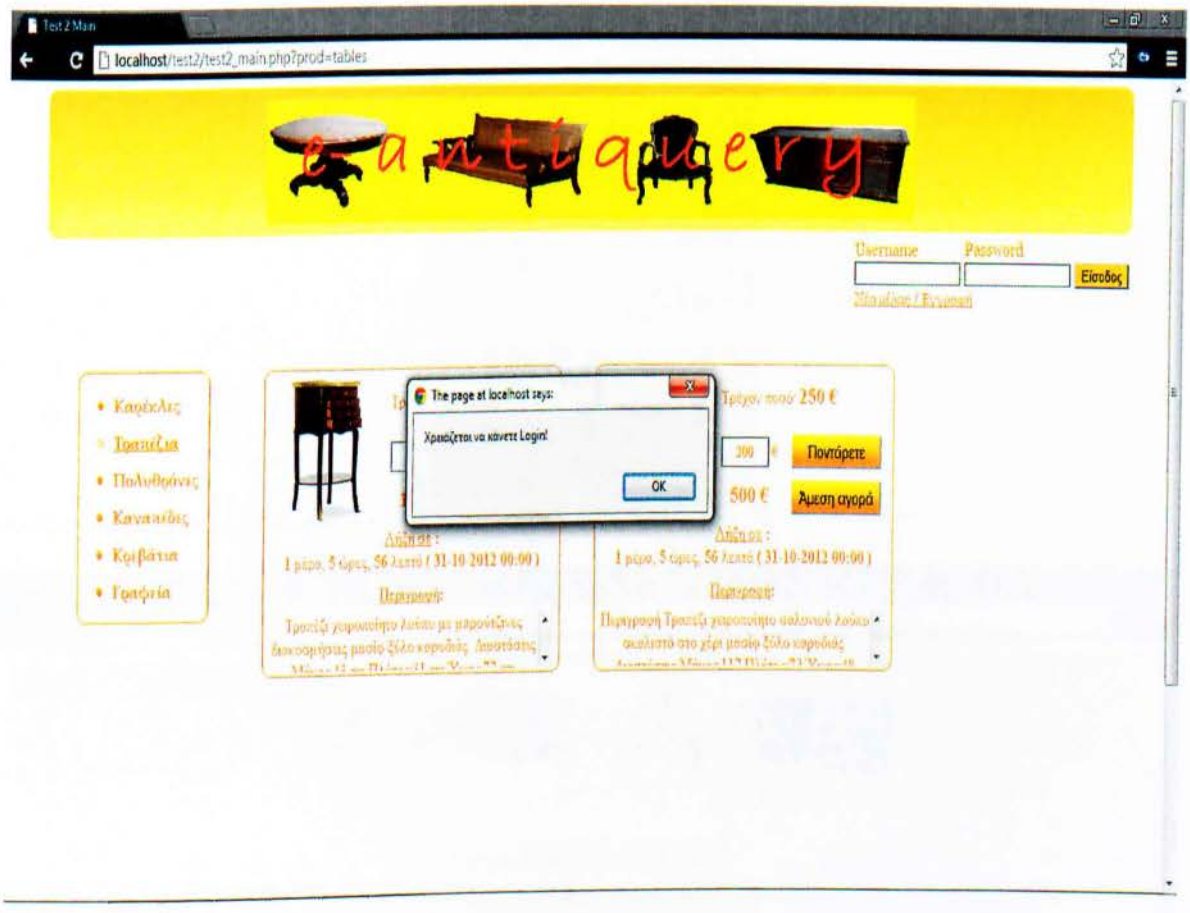
ΚΑΤΑΣΚΕΥΑΣΤΙΚΟ ΜΕΡΟΣ (WEB SITE)

Παρουσίαση του website

Στην παραπάνω εικόνα απεικονίζεται η αρχική σελίδα(main page) της ιστοσελίδας μας.Στο αριστερό μέρος παρατηρούμε μια λίστα με τις κατηγορίες των προϊόντων.Κάνοντας κλικ σε κάποια κατηγορία μπορούμε να δούμε τα προϊόντα τα οποία είναι διαθέσιμα τη δεδομένη στιγμή στη συγκεκριμένη κατηγορία.

Για κάθε προϊόν το οποίο είναι διαθέσιμο ο χρήστης έχει τη δυνατότητα να ποντάρει ένα νέο ποσό(ανώτερο του τρέχοντος ποσού) ή να προχωρήσει σε άμεση αγορα του προϊόντος στην προκαθορισμένη τιμή.

Δοκιμάζοντας να ποντάρουμε ένα νέο ποσό ή να κάνουμε άμεση αγορά σε κάποιο προϊόν θα μας εμφανιστεί μήνυμα να κάνουμε login.

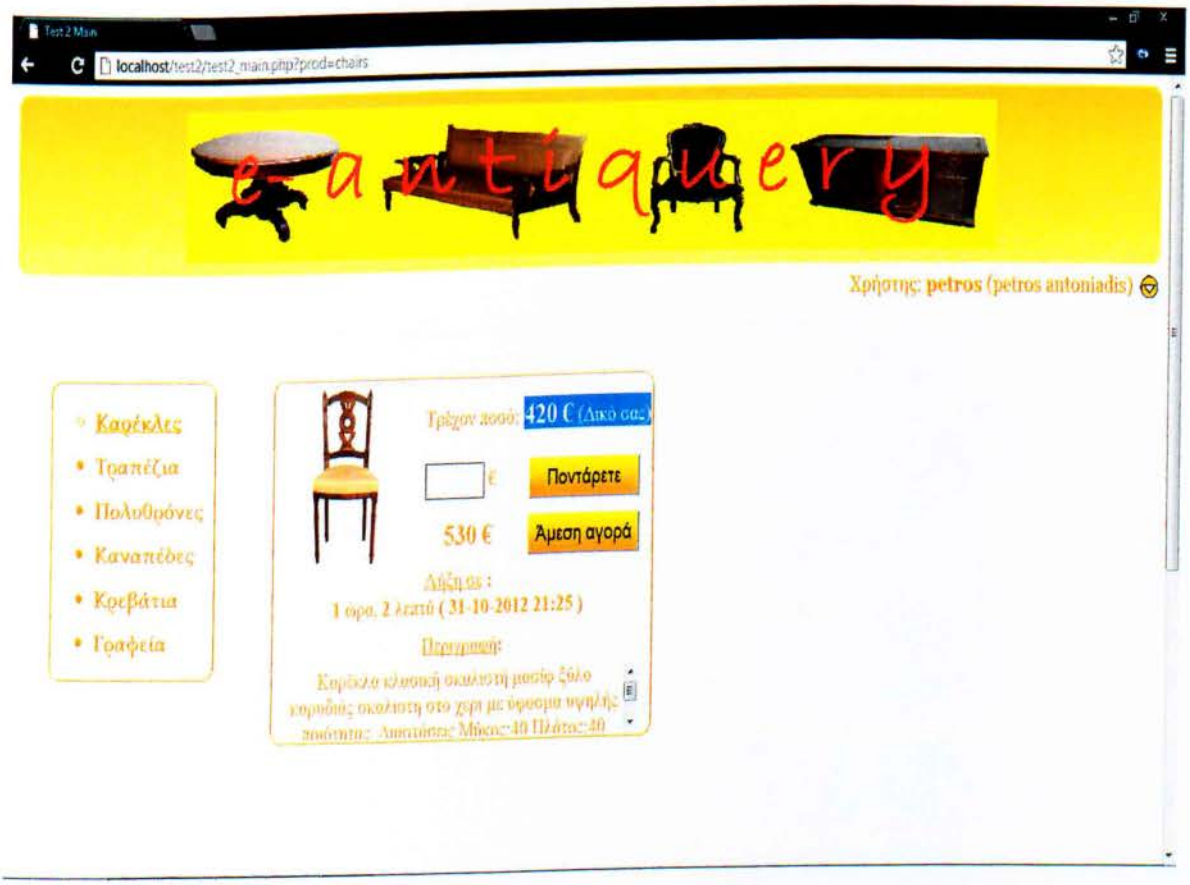


Στην περίπτωση που είμαστε ήδη μέλος,δηλαδή έχουμε πραγματοποιήσει εγγραφή σε κάποια παλιότερη επίσκεψή μας στη σελίδα,απλά συμπληρώνουμε τα πεδία username και password και πατάμε Είσοδος.Σε αντίθετη περίπτωση θα πρέπει να κάνουμε κλικ στην επιλογή Νέο μέλος/Εγγραφή και να δημιουργήσουμε έναν καινούργιο λογαριασμό(account) συμπληρώνοντας τη φόρμα που βλέπουμε παρακάτω:

Αφού συνδεθεί ο χρήστης έχει τις παρακάτω επιλογές:

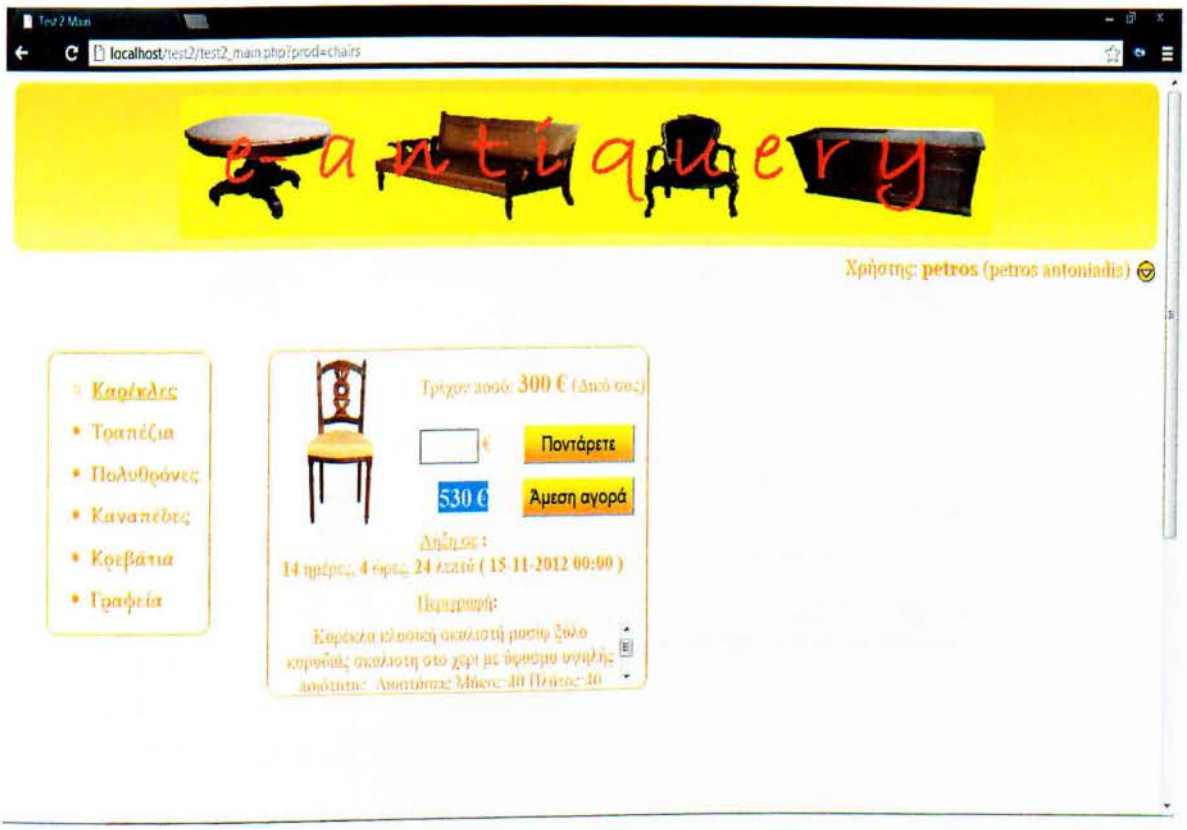
Νέο ποντάρισμα σε κάποιο προϊόν: Για να κάνει ποντάρισμα ο χρήστης,θα πρέπει η αξία του πονταρίσματός του να είναι μεγαλύτερη από το τρέχον ποσό.

Στη συνέχεια,όπως βλέπουμε και παρακάτω, το ποντάρισμα του χρήστη γίνεται το νέο τρέχον ποσό και ο πλειστηριασμός συνεχίζεται κανονικά.



Αμέσως μετά τη λήξη του πλειστηριασμού εμφανίζεται μήνυμα το οποίο ενημερώνει πως ο πλειστηριασμός έχει λήξει καθώς και το όνομα του νικητή για τη συγκεκριμένη δημοπρασία.

Άμεση αγορά προϊόντος: Σε περίπτωση που ο χρήστης συμφωνεί με την τιμή άμεσης αγοράς μπορεί να προχωρήσει στην κατοχύρωση του προϊόντος.

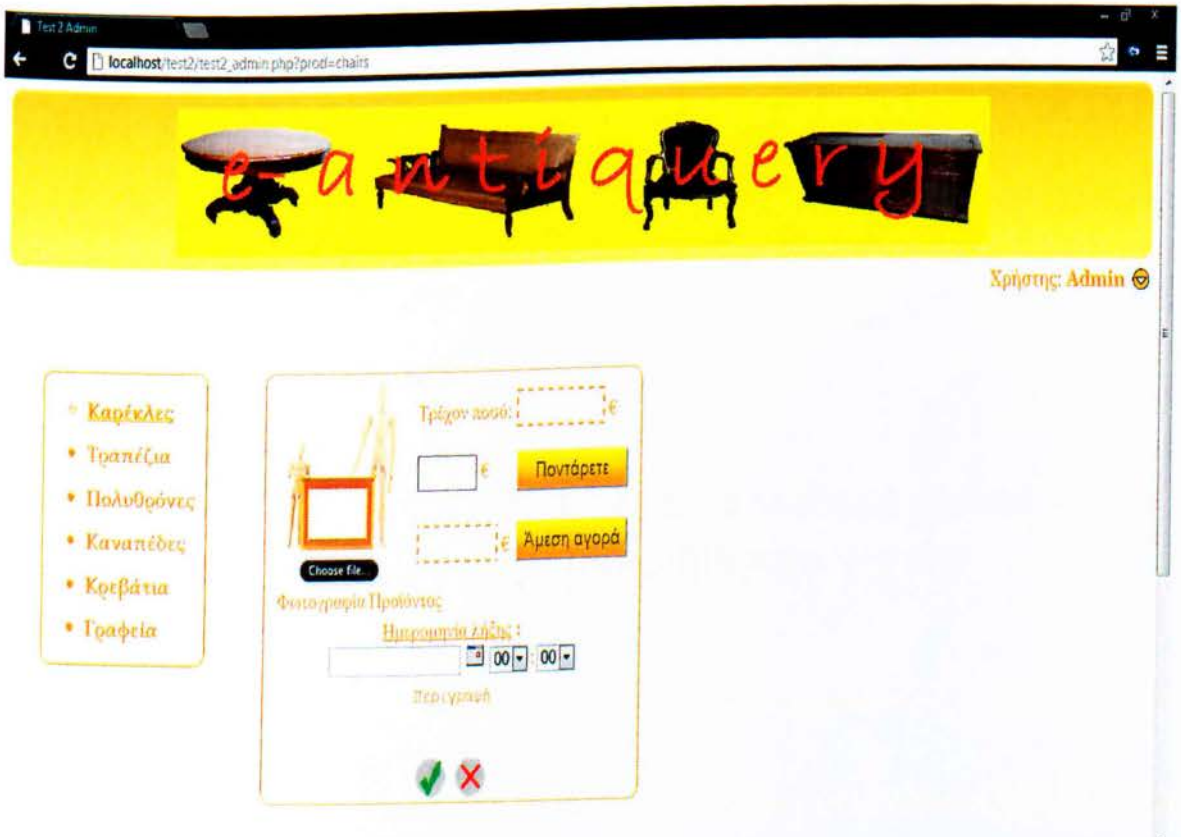


Πατώντας Άμεση Αγορά υπάρχει άμεση λήξη του συγκεκριμένου πλειστηριασμού και εμφανίζεται το παρακάτω μήνυμα:

Δυνατότητες του διαχειριστή(Admin):

Απ'τη στιγμή που θα συνδεθεί στην ιστοσελίδα,ο διαχειριστής (administrator) έχει τη δυνατότητα να αφαιρέσει κάποιο ήδη υπάρχον προϊόν.

Επιλέγοντας Προσθήκη μπορεί να προσθέσει ένα νέο προϊόν



Για κάθε νέο προϊόν θα πρέπει να συμπληρώσει τα εξής πεδία:

- Φωτογραφία προϊόντος (όπου θα ανεβάσει κάποιο αρχείο με φωτογραφία του προϊόντος)
- Τρέχον Ποσό (το οποίο θα είναι και το ποσό εκκίνησης της δημοπρασίας)
- Ποσό Άμεσης Αγοράς
- Ημερομηνία και Ώρα λήξης της δημοπρασίας
- Περιγραφή του προϊόντος

Η εργασία συνοδεύεται από CD με το website καθώς και τους κώδικες οι οποίοι χρησιμοποιήθηκαν για την υλοποίησή της…

Βιβλιογραφία

➢ Πηγές από Βιβλία-Συγγράμματα:

- **Ανάπτυξη WEB Εφαρμογών με PHP και MySQL(3η έκδοση)** – Luke Welling, Αθήνα 2005, Εκδόσεις: Μ. Γκιούρδας.
- **Μάθετε την SQL σε 24 Ώρες(3η έκδοση)** – Ryan Stephens & Ron Plew, Αθήνα 2003, Εκδόσεις: Μ. Γκιούρδας.
- **Εισαγωγή στις Βάσεις Δεδομένων** – Μιχάλης Ξένος, Δημήτρης Χριστοδουλάκης, Αθήνα 2002, Εκδόσεις: Παπασωτηρίου.
- **Μάθετε Βάσεις Δεδομένων** – Ryan Stephens & Ron Plew, Αθήνα 2003, Εκδόσεις: Μ.Γκιούρδας.
- **Εισαγωγή στις PHP 6 και MySQL 5 με εικόνες** – Κ.Ξαρχάκος, Αθήνα 2007, Εκδόσεις: Κλειδάριθμος.
- **PHP, MySQL και Apache Όλα Σε Ένα** – Julie C.Meloni, Αθήνα 2008, Εκδόσεις: Μ.Γκιούρδας.
- **PHP Solutions for Dynamic Web Design Made Easy** – David Powers.
- **PHP and MySQL for Dynamic Web Pages** – Larry Ullman.
- **SAMS Teach Yourself PHP in 10 Minutes** – Chris Newman.
- **Beginning PHP, Apache, MySQL® for Web Development** – Michael Glass, Elizabeth Nanamore, Gary Mailer, Jason Gerner.
- **PHP and MySQL Everyday Apps for Dummies** – Janet Valade.
- **PHP 5/MySQL Programming for the Absolute Beginner** – Andy Harris.
- **How to Do Everything with PHP and MySQL** – Vikram Vaswani.
- **Web Database Application with PHP and MySQL** – David Lane, Hugh E.Williams.

➢ Πηγές από Διαδίκτυο:

- http://www.php.net
- http://www.xampp.com
- http://www.wamp.com
- http://www.htmlite.com/PHPintro.php
- http://dev.mysql.com/doc/
- http://www.w3schools.com/
- http://www.plus2net.com/php_tutorial/site_map.php
- http://www.php-csl.com/snippets
- http://hudzilla.org/phpwiki/index.php?title=Main_Page
- http://www.siteowners.gr/web-hosting/
- http://www.microsoft.com