ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

# Ανάπτυξη Γραφικών Διεπαφών Χρήστη για την διασύνδεση Βάσεων Δεδομένων Ανοικτού Κώδικα

Του φοιτητή

Κωνσταντίνου Γαλιάτσου

Αρ. Μητρώου: 06/3173

Επιβλέπων καθηγητής

Παναγιώτης Σφέτσος

Θεσσαλονίκη 2012



`

**ΠΕΡΙΛΗΨΗ**

Για την δημιουργία μιας εφαρμογής για την διαχείριση βάσεων δεδομένων απαιτείται αρχικά ένα σύστημα βάσεων δεδομένων, το οποίο έχει όλες τις απαραίτητες λειτουργίες για την λειτουργία μιας βάσης δεδομένων. Τα συστήματα διαχείρισης βάσεων δεδομένων MySQL και PostgreSQL έχουν αποδείξει ότι έχουν δυνατότητες που τα καθιστούν κατάλληλα ακόμα και για τα πιο απαιτητικά περιβάλλοντα. Επίσης απαιτείται η έκφραση των ερωτημάτων SQL, τα οποία θα κάνουν αλλαγές στην βάση δεδομένων και θα παρουσιάζουν τα δεδομένα στον χρήστη της εφαρμογής.

Για την διασύνδεση της εφαρμογής με την βάση δεδομένων υπάρχει πληθώρα τρόπων, από βιβλιοθήκες για τις πιο γνωστές γλώσσες προγραμματισμού, όπως η Java, ως αυτοματοποιημένοι τρόποι διασύνδεσης από εργαλεία ανάπτυξης εφαρμογών, όπως το Netbeans.

Η εφαρμογή χρησιμοποιεί τις βιβλιοθήκες Swing της Java για την δημιουργία γραφικών διεπαφών χρήστη και υλοποιεί την αρχιτεκτονική πελάτη-εξυπηρετητή.



`

## ABSTRACT

For the creation of a database application, a database management system is required, which has all the necessary functions for the function of a database. The database management systems, MySQL and PostgreSQL, have proven that they have all the capabilities that make them suitable even for the most demanding enviromnents. Also, the expression of SQL queries is required which make changes to the database or present data to the user of the application.

To connect the application to the database there are plenty of ways, from libraries for the most common programming languages, to automated ways by IDEs, such as Netbeans.

The application uses the Java Swing library to create graphical user interfaces and implements the client server architecture.



`

**ΠΕΡΙΕΧΟΜΕΝΑ**





`

**Ευρετήριο σχημάτων**





`

**Ευρετήριο πινάκων**





`

## ΕΙΣΑΓΩΓΗ

Σκοπός αυτής της πτυχιακής εργασίας είναι να αναπτυχθεί μία ολοκληρωμένη εφαρμογή σε Java με γραφικό περιβάλλον, η οποία εκτελεί βασικές ενέργειες σε μια βάση δεδομένων ανοικτού κώδικα, όπως προβολή δεδομένων, δημιουργία, επεξεργασία και διαγραφή στοιχείων, καθώς και να μελετήσει τα στοιχεία που απαρτίζουν αυτήν την εφαρμογή, τα οποία εξετάζονται σε ξεχωριστά κεφάλαια.

Αυτά είναι:

1. Η ανάλυση και η παρουσίαση των συστημάτων διαχείρισης βάσεων δεδομένων, MySQL και PostgreSQL, όπου παρουσιάζονται τα χαρακτηριστικά και οι δυνατότητες του κάθε συστήματος.
2. Τρόποι διασύνδεσης με την βάση δεδομένων ανοικτού κώδικα, όπου παρουσιάζονται οι βιβλιοθήκες και οι τρόποι που μπορεί να συνδεθεί το πρόγραμμα σε μια βάση δεδομένων, όπως το JDBC για Java, αλλά και τις επιλογές που δίνουν τα περιβάλλοντα ανάπτυξης Netbeans και Eclipse.
3. Στο τρίτο κεφάλαιο παρουσιάζεται η SQL, η γλώσσα αιτημάτων προς μια βάση δεδομένων. Αυτά τα ερωτήματα θα χρησιμοποιηθούν από την εφαρμογή για την διαχείριση των δεδομένων της βάσης δεδομένων.
4. Στη συνέχεια παρουσιάζονται τα στοιχεία γραφικών της Java, και συγκεκριμένα η Swing που είναι η βιβλιοθήκη γραφικών που ταιριάζει στα πρότυπα της Java.
5. Στο πέμπτο κεφάλαιο αναπτύσονται πρότυπα γραφικών διεπαφών χρήστη, χρησιμοποιώντας τα στοιχεία που εξετάστηκαν στο προηγούμενο κεφάλαιο.
6. Στο έκτο κεφάλαιο παρουσιάζεται η αρχιτεκτονική πελάτη-εξυπηρετητή και δημιουργείται μια βασική εφαρμογή-παράδειγμα.
7. Τέλος αναπτύσεται η εφαρμογή διαχείρισης της βάσης δεδομένων και παρουσιάζεται η λειτουργία της.



`

# 1. Ανάλυση και παρουσίαση των Βάσεων Δεδομένων Ανοικτού Κώδικα MySQL και PostgreSQL

## ΕΙΣΑΓΩΓΗ

Σε αυτό το κεφάλαιο θα παρουσιαστούν τα συστήματα διαχείρισης βάσεων δεδομένων ανοικτού κώδικα MySQL και PostgreSQL, οι τρόποι σύνδεσής τους με προγράμματα Java

## 1.1 MySQL

Η MySQL είναι ένα ανοικτού κώδικα σύστημα διαχείρισης βάσεων δεδομένων SQL, το οποίο αναπτυσεται, διανέμεται και υποστηρίζεται από την Oracle Corporation.

Τα χαρακτηριστικά της είναι:

Εσωτερικά και μεταφερσιμότητα:

- Οι βάσεις δεδομένων είναι σχεσιακού τύπου.
- Γραμμένο σε γλώσσα προγραμματισμού C και C++.
- Ελεγμένο με ένα ευρύ φάσμα από μεταγλωττιστές.
- Λειτουργεί σε πολλές διαφορετικές πλατφόρμες.
- Χρησιμοποιεί σχεδιασμό διακομιστή πολλών επιπέδων με ανεξάρτητες μονάδες.
- Προσφέρει μηχανές αποθήκευσης με και χωρίς συναλλαγές (σύνολο ενεργειών σε μια βάση δεδομένων)
- Σχεδιασμένο να είναι πλήρως πολυνηματικό χρησιμοποιώντας νήματα του πυρήνα, για να χρησιμοποιεί πολλαπλούς επεξεργαστές αν είναι διαθέσιμοι.
- Χρησιμοποιεί πολύ γρήγορους πίνακες B-δέντρων (στην μηχανή αποθήκευσης MyISAM) με συμπίεση δείκτη.
- Σχεδιασμένο να υπάρχει ευκολία στην προσθήκη επιπλέον μηχανών αποθήκευσης (storage engines).
- Χρησιμοποιεί ένα πολύ γρήγορο σύστημα κατανομής μνήμης (memory allocation) βασισμένο σε νήματα.
- Υλοποιεί πίνακες κατακερματισμού (hash tables) στην μνήμη, που χρησιμοποιούνται ως προσωρινοί πίνακες.
- Υλοποιεί συναρτήσεις SQL χρησιμοποιώντας μια υψηλά βελτιστοποιημένη βιβλιοθήκη κλάσης που πρέπει να είναι όσο το δυνατόν γρηγορότερη. Συνήθως δεν υπάρχει κατανομή μνήμης μετά την αρχικοποίηση ερωτημάτων.
- Προσφέρει τον διακομιστή ως ένα ξεχωριστό πρόγραμμα για χρήση σε ένα δικτυωμένο περιβάλλον πελάτη/διακομιστή και ως μια βιβλιοθήκη που μπορεί να ενσωματωθεί(συνδεθεί) σε αυτόνομες εφαρμογές. Τετοιες εφαρμογές μπορούν να χρησιμοποιηθούν απομονωμένα ή σε περιβάλλοντα όπου δεν υπάρχει δίκτυο.



`

Τύποι δεδομένων:

Η MySQL υποστηρίζει πολλούς τύπους δεδομένων: προσημασμένοι και μη(unsigned and unsigned) Integers των 1,2,3,4,8 bytes, FLOAT, DOUBLE, CHAR, VARCHAR, BINARY, VARBINARY, TEXT, BLOB, DATE, TIME, DATETIME, TIMESTAMP, YEAR, SET, ENUM και χωρικοί (spatial) τύποι δεδομένων OpenGIS καθώς και σταθερούς και μεταβλητούς τύπους String.

Προτάσεις και συναρτήσεις:

- Πλήρη υποστήριξη operators και συναρτήσεων στους όρους SELECT και WHERE των SQL ερωτημάτων
- Πλήρη υποστήριξη των όρων GROUP BY και ORDER BY. Υποστήριξη συναθροιστικών συναρτήσεων(group functions) (COUNT(), AVG(), STD(), SUM(), MAX(), MIN(), and GROUP_CONCAT()).
- Υποστήριξη LEFT OUTER JOIN και RIGHT OUTER JOIN με κανονική σύνταξη SQL και σύνταξη ODBC.
- Υποστήριξη ψευδωνυμάτων για πίνακες και στήλες όπως απαιτείται από την SQL.
- Υποστήριξη για DELETE, INSERT, REPLACE, and UPDATE για επιστροφή του αριθμού των γραμμών που επηρρεάστηκαν, ή αλλιώς, για επιστροφή του αριθμού των γραμμών που ταιριάζουν θέτοντας μια σημαία στην σύνδεση με τον διακομιστή.
- Υποστήριξη προτάσεων SHOW είδικά για την MySQL που ανακτά πληροφορίες σχετικές με βάσεις δεδομένων, μηχανές αποθήκευσης, πίνακες και δείκτες. Η έκδοση 5.0 της MySQL προσθέτει υποστήριξη για την βάση δεδομένων INFORMATION_SCHEMA, υλοποιημένη σύμφωνα με την πρότυπη SQL.
- Μια προταση EXPLAIN για να δείχνει πως ο βελτιστοποιητής (optimizer) επιλύει ένα ερώτημα.
- Ανεξαρτησία των ονομάτων συναρτήσεων από αυτά των πινάκων ή στηλών. Ο μόνος περιιορισμός για την κλήση συνάρτησης είναι να μην υπάρχουν κενά μεταξύ του ονόματος και της παρένθεσης "( " που ακολουθεί.
- Είναι δυνατή η αναφορά πινάκων από διαφορετικές βάσεις δεδομένων στην ίδια πρόταση.



`

Ασφάλεια:

- Ένα σύστημα δικαιωμάτων και κωδικών είναι ευέλικτο και ασφαλές και ενεργοποιεί επαλήθευση βασισμένη σε host.
- Ασφάλεια κωδικών κρυπτογραφώντας όλη την μεταφορά κωδικών στην σύνδεση με τον εξυπηρετητή.

Επεκτασιμότητα και όρια:

- Υποστήριξη μεγάλων βάσεων δεδομένων. Υπάρχουν περιπτώσεις χρηστών που χρησιμοποιούν MySQL Server με 200,000 πίνακες και περίπου 5,000,000,000 γραμμές.
- Υποστήριξη έως 64 δείκτες(indexes) ανά πίνακα.

Συνδεσιμότητα:

Προγράμματα πελάτη μπορούν να συνδεθούν στην MySQL χρησιμοποιώντας διάφορα προτόκολα:

- Πελάτες μπορούν να συνδεθούν χρησιμοποιώντας TCP/IP sockets σε κάθε πλατφόρμα.
- Στα συστήματα Windows, στην οικογένεια NT (NT, 2000, XP, 2003, Vista), πελάτες μπορούν να συνδεθούν χρησιμοποιώντας ονομασμένη διασωλήνωση (named pipe) αν ο διακομιστης έχει εκκινηθεί με την κατάλληλη επιλογή (διακόπτης –enable-named-pipe). Από την MySQL 4.1 και πάνω, οι Windows servers υποστηριζουν επίσης και συνδέσεις με κοινόχρηστη μνήμη, αν έχει εκκινηθεί με την επιλογή –shared-memory. Οι πελάτες μπορούν να συνδεθούν χρησιμοποιώντας την επιλογή –protocol-memory.
- Στα συστήματα Unix, οι πελάτες μπορούν να συνδεθούν χρησιμοποιώντας τα Unix domain socket files.
- Προγράμματα πελάτη της MySQL μπορούν να γραφτούν σε πολλές γλώσσες. Μια βιβλιοθήκη πελάτη που είναι γραμμένη στην C είναι διαθέσιμη για πελάτες γραμμένους σε C++ ή σε κάθε άλλη γλώσσα που προσφέρει σύνδεση με την C .
- Εϊναι διαθέσιμα API για C, C++, Eiffel, Java, Perl, PHP, Python, Ruby, και Tcl, δίνοντας την δυνατότητα πελάτες MySQL να γραφτούν σε πολλές γλώσσες.
- Η διασύνδεση Connector/ODBC προσφέρει υποστήριξη MySQL για προγράμματα πελάτη που χρησιμοποιούν συνδέσεις ODBC (Open Database Connectivity). Για παράδειγμα μπορεί να συνδεθει η MS Access στον διακομιστή MySQL. Οι πελάτες μπορούν να τρέξουν σε Windows ή Unix.
- Η διασύνδεση Connector/JDBC προσφέρει υποστήριξη MySQL, για προγράμματα πελάτη γραμμένα σε Java, που κάνουν χρηση των συνδέσεων JDBC. Οι πελάτες μπορούν να τρέξουν σε Windows ή Unix.
- MySQL Connector/Net δινει την δυνατότητα σε προγραμματιστές να δημιουργήσουν εύκολα εφαρμογές σε .NET που απαιτούν ασφαλή, υψηλής απόδοσης σύνδεση δεδομένων με την MySQL. Εφαρμόζει τις απαιτούμενες



`

διασυνδέσεις ADO.NET και ενσωματώνεται στα εργαλεία ADO.NET. Οι προγραμματιστές μπορουν να δημιουργήσουν εφαρμογές χρησιμοποιώντας την γλώσσα .NET της επιλογής τους. Το MySQL Connector/NET είναι ένας πλήρως διαχειρίσιμος οδηγός γραμμένος σε 100% καθαρή C#.

Τοπικότητα:

- Ο διακομιστής προσφέρει μηνύματα σφαλμάτων στους πελάτες σε πολλές γλώσσες.
- Πλήρης υποστήριξη για πολλά διαφορετικά σύνολα χαρακτήρων. Υποστήριξη για Unicode υπάρχει από την έκδοση 4.1 της MySQL.
- Όλα τα δεδομένα αποθηκεύονται σε επιλεγμένο σύνολο χαρακτηρων.
- Η ταξινόμηση και οι συγκρίσεις γίνονται σύμφωνα με το επιλεγμένο σύνολο και σύνθεση χαρακτήρων. Υπάρχει η δυνατότητα να αλλάξει αυτό με την εκκίνηση του διακομιστή MySQL. Ένα παράδειγμα προχωρημένης ταξινόμησης μπορεί κανείς να κοιτάξει τον κωδικα για την ταξινόμηση της Τσέχικης γλώσσας. Ο MySQL Server υποστηρίζει πολλά διαφορετικά σύνολα χαρακτήρων που μπορούν να καθοριστούν με την μεταγλώττιση και την εκκίνηση.
- Από την έκδοση 4.1 της MySQL η ζώνη ώρας του διακομιστή μπορεί να αλλάξει δυναμικά και κάθε ξεχωριστός πελάτης μπορέι να καθορίσει την δική του ζώνη ώρας.

Πελάτες και εργαλεία:

- Η MySQL περιλαμβάνει αρκετά εργαλεία και πελάτες. Αυτά περιλαμβάνουν και προγράμματα γραμμής εντολών όπως το mysqldump, και γραφικά προγράμματα όπως το MySQL Administrator.
- MySQL Server έχει ενσωματωμένες προτάσεις SQL για έλεγχο, βελτιστοποίηση και επιδιόρθωση πινάκων. Αυτές οι προτάσεις είναι διαθέσιμες από την γραμμή εντολών μέσω του προγράμματος mysqlcheck. Περιλαμβάνεται επίσης το myisamchk, ένα πολύ γρήγορο εργαλείο γραμμής εντολών για εκτέλεση αυτών των ενεργειών σε πίνακες MyISAM.
- Τα προγράμματα MySQL μπορούν να κληθούν με τον διακόπτη –help ή -? για λήψη online βοήθειας.



`

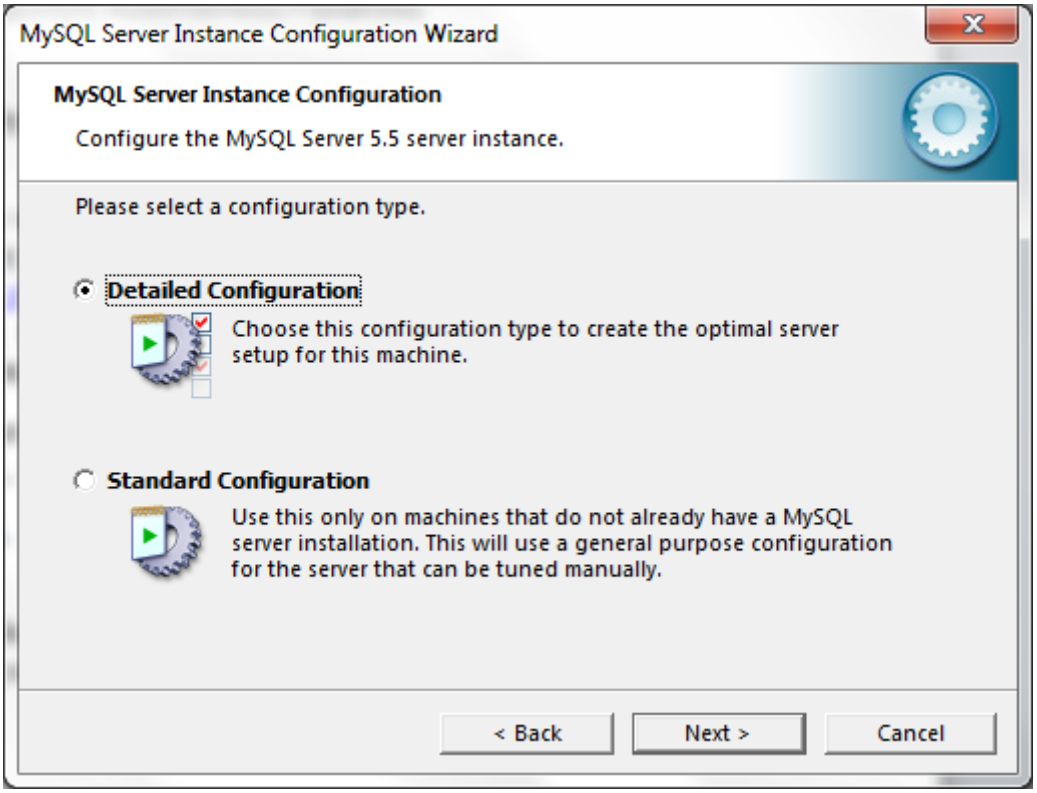
### 1.1.2 Εγκατάσταση MySQL

Η εγκατάσταση του διακομιστή MySQL γίνεται σε περιβάλλον Microsoft Windows 7. Επιλέγεται η έκδοση MySQL Community Server x86 64bit που διατίθεται στην ιστοσελίδα http://www.mysql.com/downloads/mysql/. Στην συνέχεια εκτελείται το αρχείο εγκατάστασης και επιλέγεται ο τύπος εγκατάστασης “Typical”, που ενδείκνυται από την εφαρμογή.

Αμέσως μετά την εγκατάστασης ανοίγει αυτόματα μια εφαρμογή για την ρύθμιση του διακομιστή MySQL, το “MySQL Server Instance Configuration Wizard”

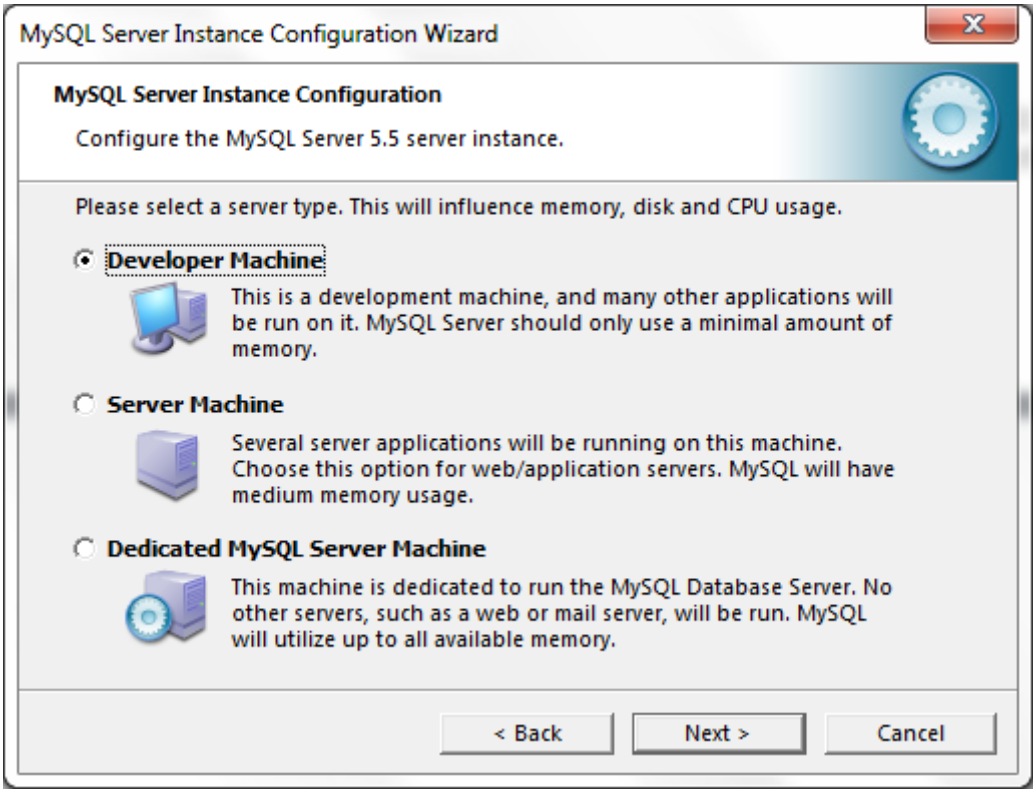
### 1.1.3 Ρύθμιση MySQL

Στην εφαρμογή αυτή δίνονται αρχικά δύο επιλογές για την ρύθμιση του διακομιστή: η λεπτομερής ρύθμιση και η κανονική ρύθμιση




Εικόνα 1. Επιλογή τύπου ρύθμισης διακομιστή MySQL

Σε αυτό το σημείο επιλέγεται η λεπτομερής ρύθμιση (Detailed Configuration). Στο επόμενο παράθυρο ζητείται ο τύπος του υπολογστή που εγκαταστάθηκε η MySQL ώστε να ρυθμιστεί το πόσους πόρους του υπολογιστή θα καταλαμβάνει ο διακομιστής. Δίνονται οι επιλογές μηχάνημα προγραμματιστή, μηχάνημα διακομιστή και διακομιστής αφιερωμένος για MySQL. Στην συγκεκριμένη περίπτωση επιλέγεται το μηχάνημα προγραμματιστή.

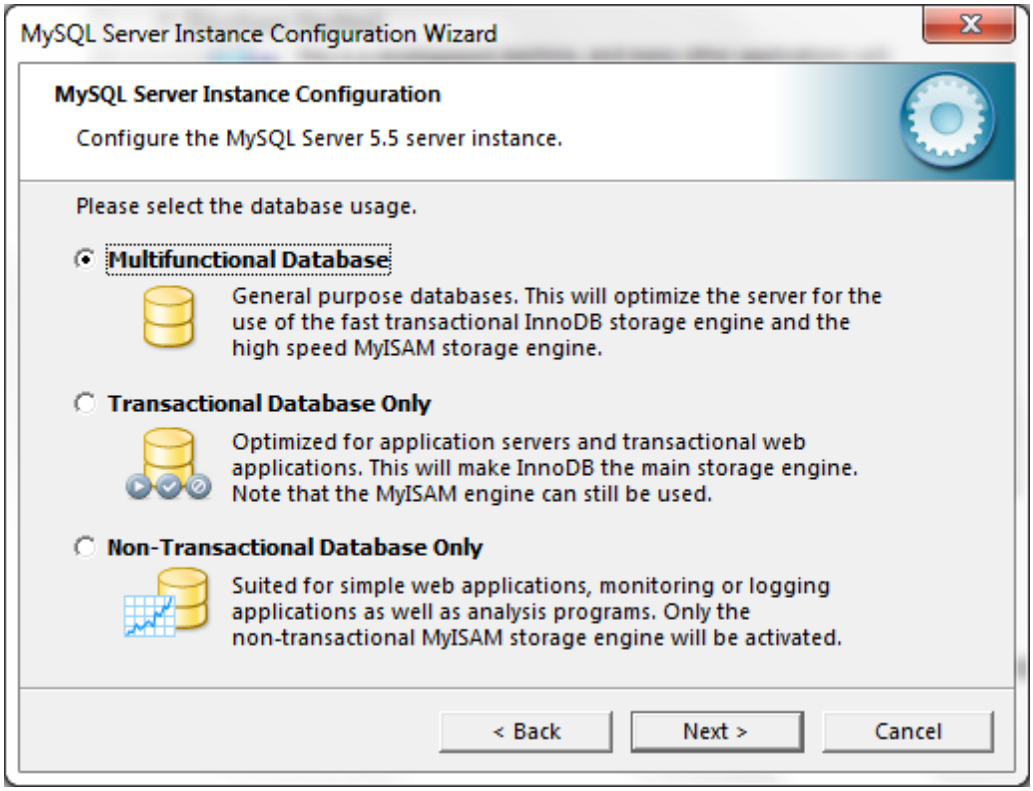



Εικόνα 2. Επιλογή τύπου υπολογιστή



`

Στην συνέχεια η εφαρμογή ζητάει από τον διαχειριστή το είδος της χρήσης της βάσης δεδομένων. Οι επιλογές που δίνονται είναι: Πολυλειτουργική βάση δεδομένων, βάση δεδομένων με συναλλαγές και βάση δεδομένων χωρίς συναλλαγές. Στην συγκεκριμένη περίπτωση επιλέγεται η πολυλειτουργική βάση δεδομένων για γενική χρήση η οποία χρησιμοποιεί την μηχανή βάσεων δεδομένων συναλλαγών InnoDB και την μηχανή αποθήκευσης MyIsam.

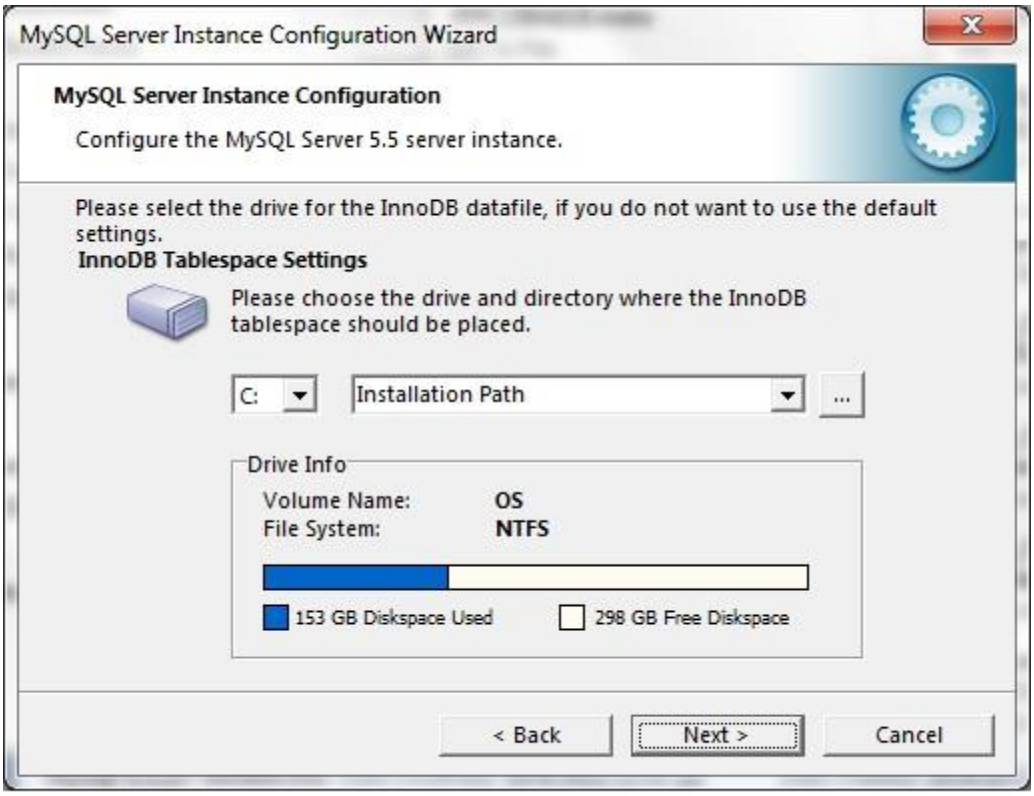



Εικόνα 3. Επιλογή είδους χρήσης βάσης δεδομένων



`

Μετά την επιλογή του είδους χρήσης ζητείται από τον διαχειριστή την τοποθεσία στην οποία θα αποθηκευτούν τα αρχεία της μηχανής της βάσης δεδομένων InnoDB. Δίνονται οι προκαθορισμένες διαδρομές: διαδρομή εγκατάστασης, \MySQL Datafiles\, \MySQL InnoDB Datafiles\ και \Data\. Οι τρεις τελευταίες επιλογές είναι φάκελοι στο πρώτο επίπεδο φακέλων του σκλήρού δίσκου.

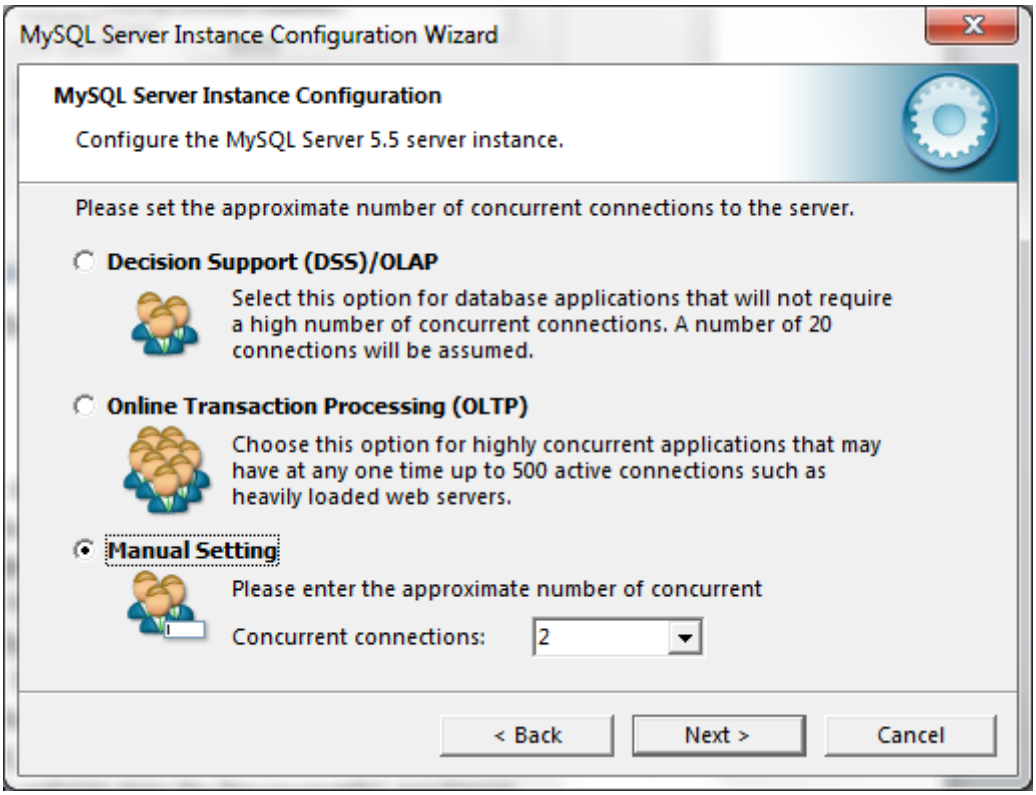



Εικόνα 4. Επιλογή διαδρομή εγκατάστασης της μηχανής βάσης δεδομένων InnoDB



`

Στην συνέχεια ζητείται από τον διαχειριστή να εισάγει τον αριθμό των ταυτόχρονων συνδέσεων προς την βάση δεδομένων. Οι επιλογές που υπάρχουν αντιστοιχούν σε είδη εφαρμογών βάσεων δεδομένων και είναι οι εξής: Λήψης αποφάσεων με είκοσι ταυτόχρονες συνδέσεις, απευθείας επεξεργασία συναλλαγών, για πεντακόσιες ταυτόχρονες, και τέλος χειροκίνητη εισαγωγή, όπου ο διαχειριστής εισάγει τον ακριβή αριθμό των ταυτόχρονων συνδέσεων. Θα επιλεχθεί η χειροκίνητη εισαγωγή, με δύο ταυτόχρονες συνδέσεις. Η πρώτη θα είναι για τον διαχειριστή (root), και η δεύτερη για λογαριασμό απλού χρήστη που θα δημιουργηθεί αργότερα.



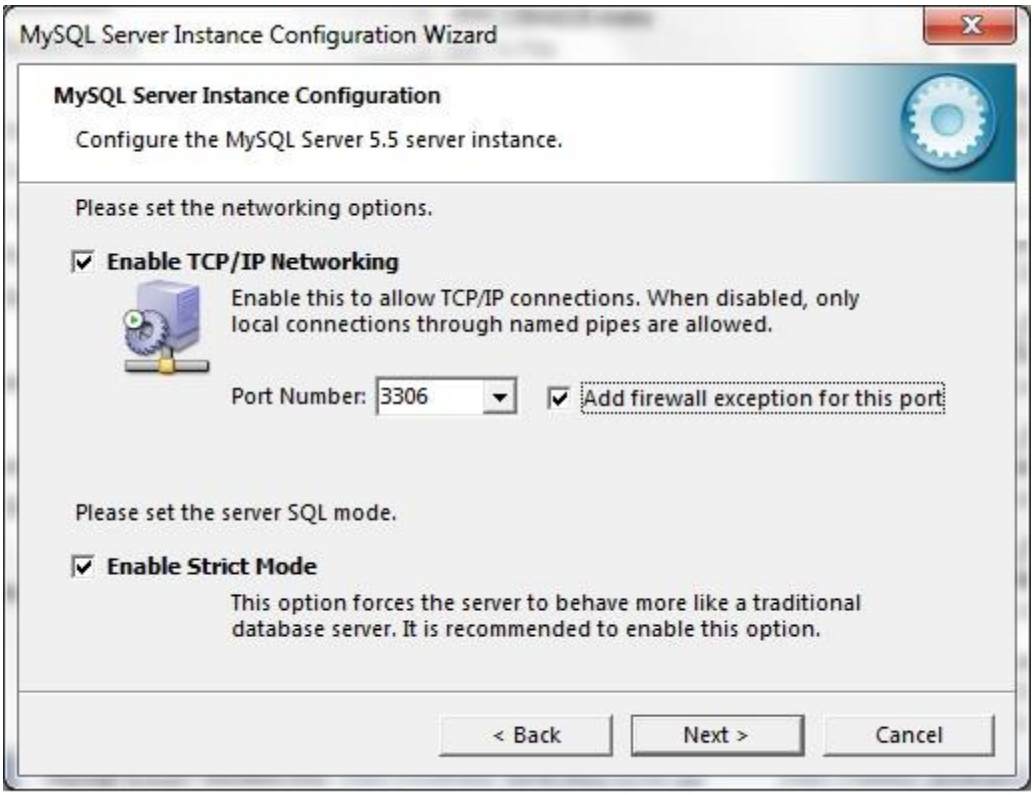

Εικόνα 5. Εισαγωγή αριθμού ταυτόχρονων συνδέσεων



`

Στο επόμενο παράθυρο βρίσκονται οι επιλογές σχετικές με το δίκτυο. Συγκεκριμένα ζητείται αν θα χρησιμοποιούνται συνδέσεις TCP/IP ή οι συνδέσεις γίνονται μόνο από τον ίδιο υπολογιστή μέσω διασωληνώσεων. Επιλέγεται η σύνδεση μέσω δικτύου TCP/IP και η θύρα την οποία θα «ακούει» ο διακομιστής η 3006, η οποία είναι η προεπιλεγμένη για την MySQL. Επιλέχθηκε επίσης η εφαρμογή να προσθέσει εξαίρεση στο Firewall του συστήματος για τον διακομιστή.

Επίσης στο ίδιο παράθυρο βρίσκεται επιλογή για τον τύπο της SQLτου διακομιστή. Επιλέγεται να είναι αυστηρό (strict), ώστε ο διακομιστής να συμπεριφέρεται σαν παραδοσιακός διακομιστής βάσεων δεδομένων.

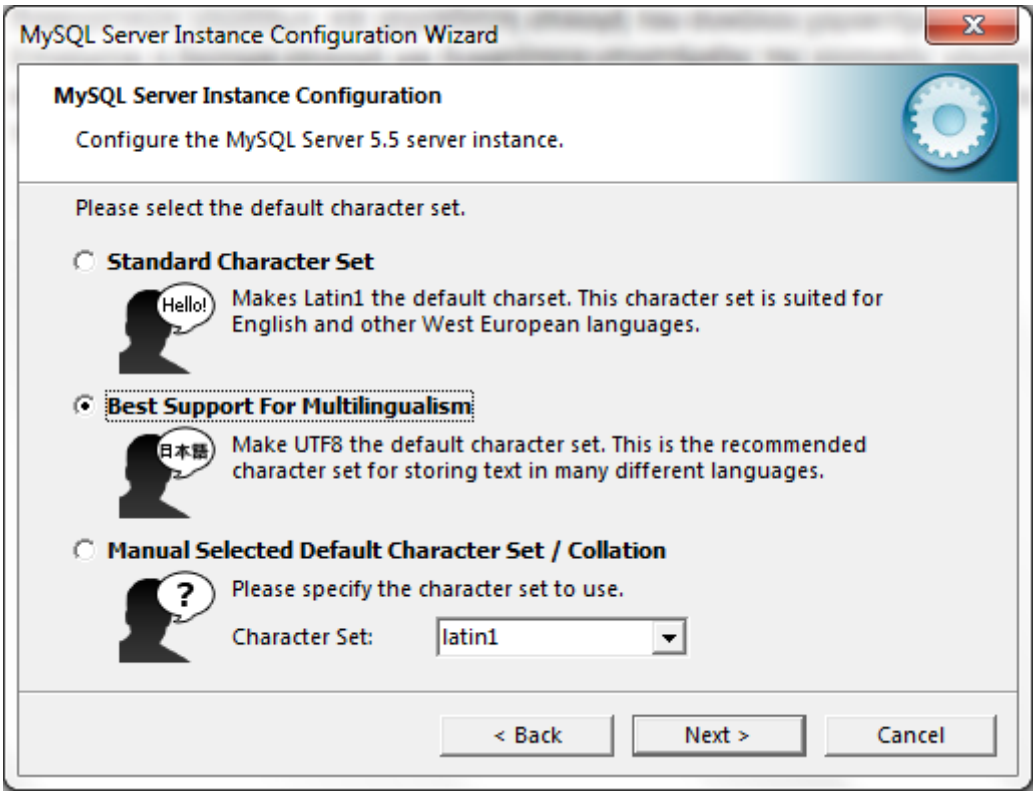



Εικόνα 6. Επιλογές δικτύου και τύπου SQL



`

Οι επόμενες επιλογές που συναντάει ο διαχειριστής είναι για το σύνολο χαρακτήρων που θα υποστηρίζει ο διακομιστής MySQL. Οι διαθέσιμες επιλογές είναι: Κανονικό σύνολο χαρακτήρων το οποίο είναι το latin1 που υποστηρίζει τα αγγλικά και άλλες δυτικοευρωπαϊκές γλώσσες, Καλύτερη υποστήριξη για πολυγλωσσία με το σύνολο χαρακτήρων να είναι το UTF8 για υποστήριξη πολλών διαφορετικών γλώσσων, και χειροκίνητη επιλογή του συνόλου χαρακτήρων. Επιλέγεται η δεύτερη επιλογή για δυνατότητα υποστήριξης της ελληνικής γλώσσας καθώς επίσης και επειδή γλώσσες προγραμματισμού όπως η Java υποστηρίζουν το σύνολο UTF8.

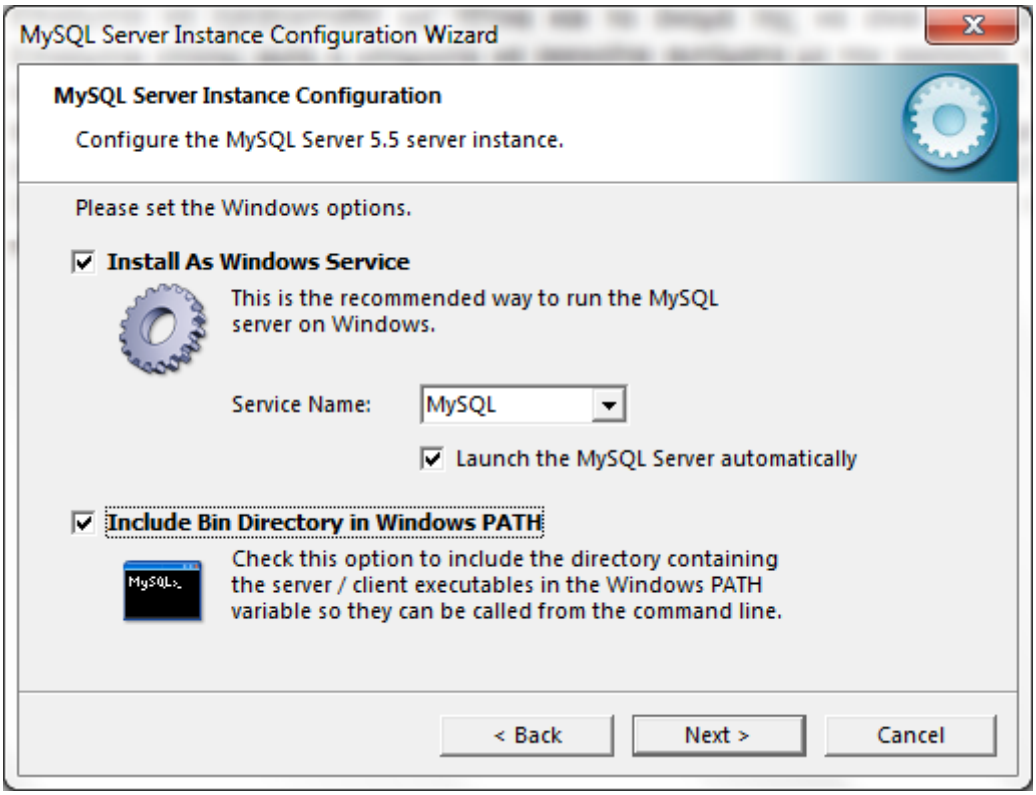



Εικόνα 7. Επιλογή συνόλου χαρακτήρων υποστήριξης του διακομιστή.



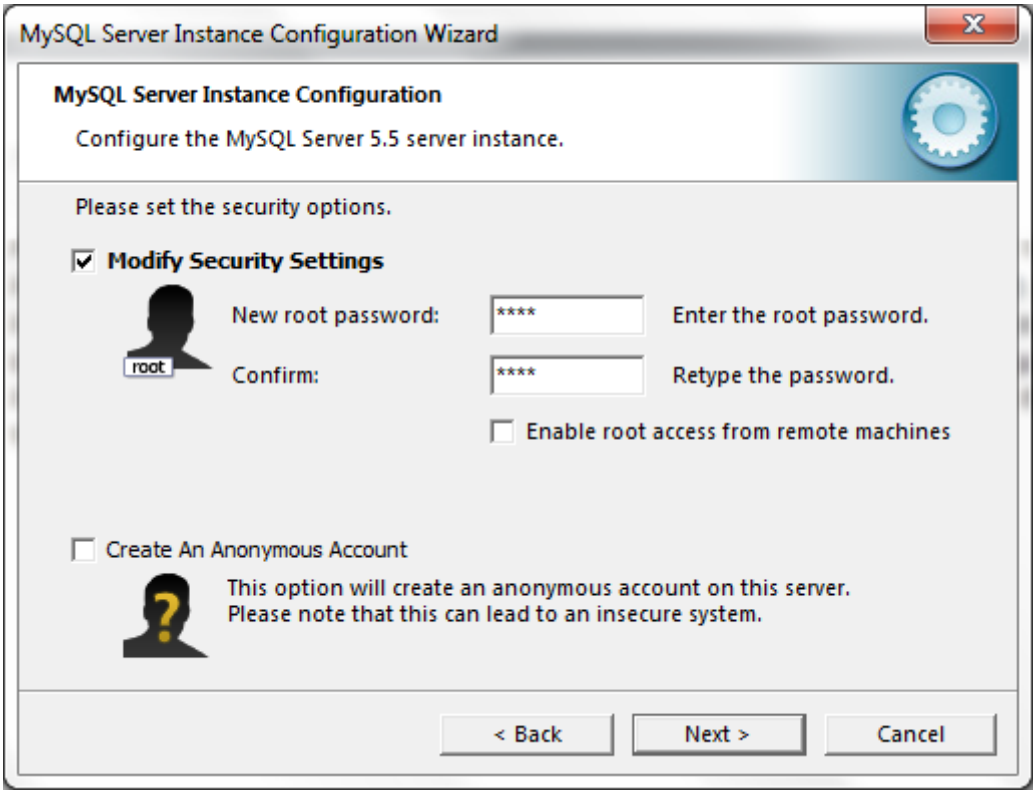
`

Στην συνέχεια υπάρχουν επιλογές συγκεκριμένες για το περιβάλλον Windows. Η πρώτη είναι αν θα εγκατασταθεί ο διακομιστής ως υπηρεσία των Windows. Επιλέγεται να εγκατασταθεί ως τέτοια και το όνομά της να είναι “MySQL”. Επιλέγεται επίσης αυτή η υπηρεσία να εκκινείται αυτόματα με την εκκίνηση του υπολογιστή. Η δεύτερη επιλογή είναι αν θα συμπεριληφθεί ο φάκελος “bin” (ο φάκελος που περιέχει την εφαρμογή διακομιστή, την εφαρμογή πελάτη γραμμής εντολών καθώς και διάφορα εργαλεία) στην μεταβλητή συστήματος PATH. Επιλέγεται να συμπεριληφθεί για την εύκολη πρόσβαση στα εκτελέσιμα αρχεία της MySQL.




Εικόνα 8. Επιλογές για το περιβάλλον Windows

Ακολουθούν οι ρυθμίσεις λογαριασμού όπου ζητείται η εισαγωγή κωδικού για τον λογαριασμό διαχειριστή (root). Δεν επιλέγεται το να επιτρέπεται η πρόσβαση με λογαριασμό διαχειριστή από απομακρυσμένους υπολογιστές για λόγους ασφαλείας. Στην επιλογή αν θα επιτρέπεται να δημιουργηθεί ανώνυμος λογαριασμός αφήνεται κενό για λόγους ασφαλείας όπως και σημειώνεται στο παράθυρο.

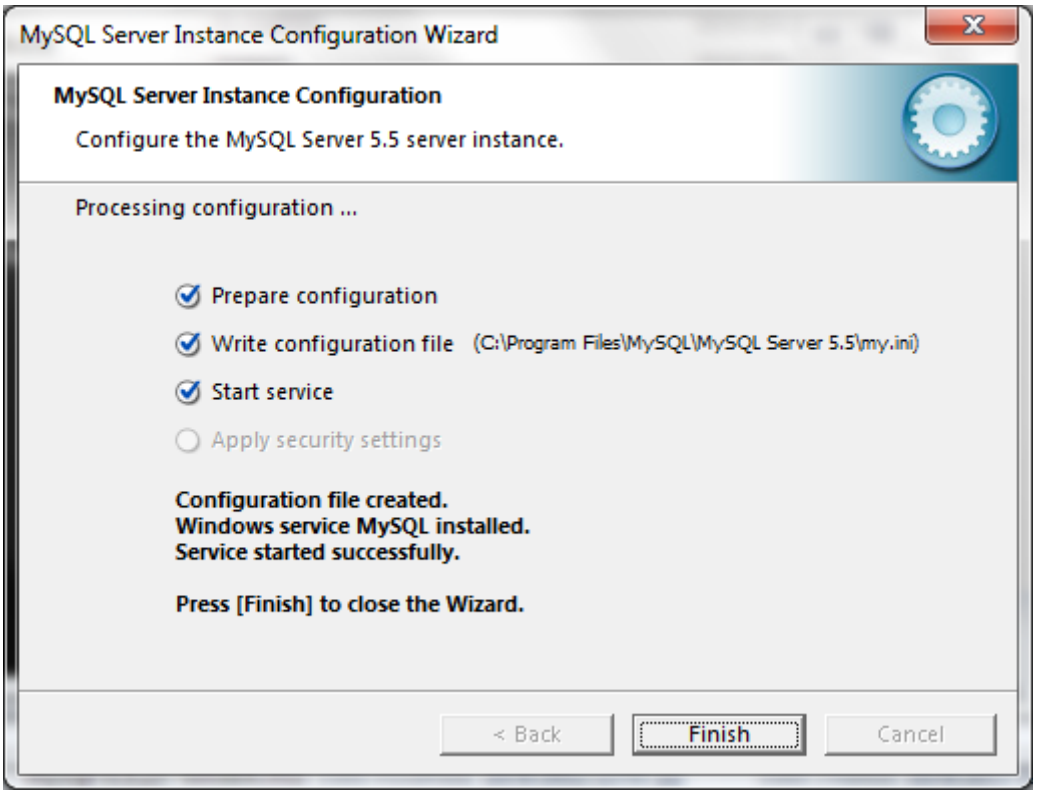



Εικόνα 9. Ρυθμίσεις λογαριασμού διαχειριστή

Αργότερα, κατά την εφαρμογή των ρυθμίσεων δεν μπορούσε να ολοκληρωθεί η διαδικασία και η εφαρμογή αυτή ήταν σε κατάσταση μη απόκρισης. Με επανάληψη της διαδικασίας και με αποεπιλογή των ρυθμίσεων ασφαλείας η διαδικασία ολοκληρώθηκε.



`




Εικόνα 10. Ολοκλήρωση των ρυθμίσεων

### 1.1.4 Διαχείριση λογαριασμών

Η αρχική ρύθμιση του διακομιστή MySQL με το εργαλείο MySQL Server Instance Configuration Wizard όπως έγινε, δηλαδή με μη ρύθμιση των επιλογών ασφαλείας άφησε τον λογαριασμό διαχειριστή χωρίς κωδικό, πράγμα εντελώς μη ασφαλές για τον διακομιστή.

Για την δημιουργία κωδικού διαχειριστή έγινε αρχικά σύνδεση στον διακομιστή χρησιμοποιώντας τον πελάτη γραμμής εντολών mysql.exe, που έρχεται με το πακέτο εγκατάστασης MySQL, πληκτρολογώντας την εξής εντολή:

“C:\Program Files\MySQL\MySQL Server 5.5\bin\mysql.exe” –u root

Στην συνέχεια για να δοθεί κωδικός πληκτρολογήθηκε η εξής εντολή SQL:

mysql> SET PASSWORD = PASSWORD('root');

όπου το “mysql” είναι η προτροπή του πελάτη mysql.exe. Με αυτήν την εντολή δημιουργήθηκε ο κωδικός “root” για τον διαχειριστή. Στο εξής η σύνδεση του διαχειριστή γίνεται με την εντολή στην γραμμή εντολών:

“C:\Program Files\MySQL\MySQL Server 5.5\bin\mysql.exe” –u root -proot

Στην συνέχεια δημιουργήθηκε ένας λογαριασμός απλού χρήστη.

mysql> CREATE USER 'kostas'@'localhost' IDENTIFIED BY 'password';

Με αυτήν την εντολή SQL δημιουργήθηκε ο λογαριασμός με όνομα kostas, ο οποίος μπορεί να συνδεθεί μόνο από τον υπολογιστή στον οποίο βρίσκεται ο διακομιστής, με κωδικό “password”.

Ο λογαριασμός αυτός όμως δεν έχει κανένα δικαίωμα για δημιουργία, διαγραφή και επεξεργασία πινάκων.

mysql> GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,DROP ON *.* TO 'kostas'@'localhost';

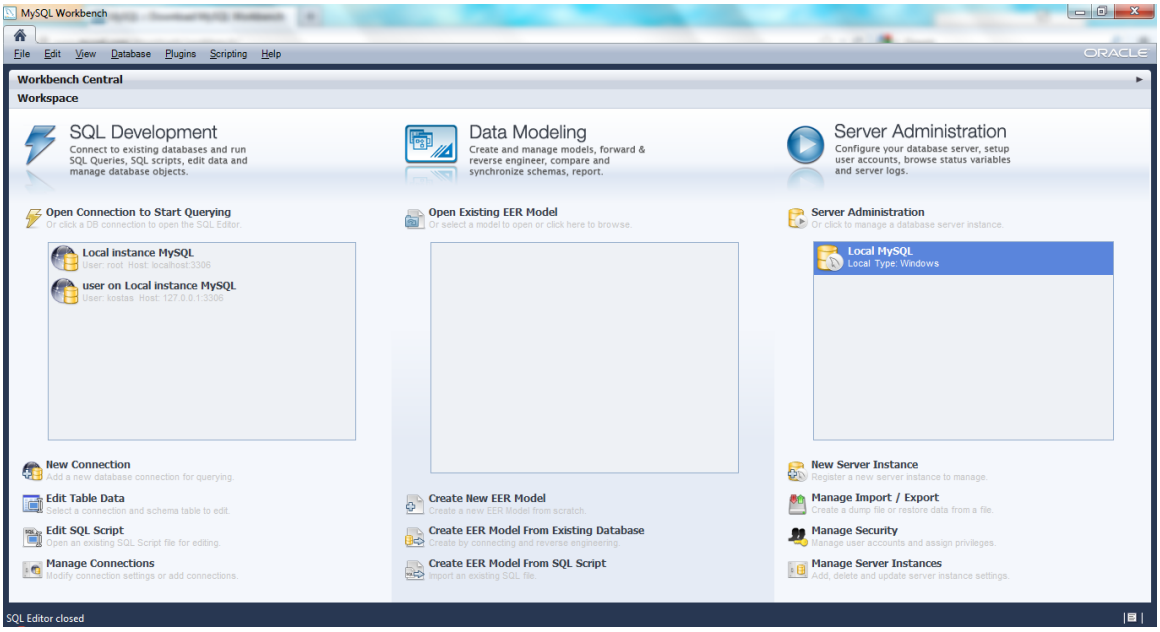
Με αυτήν την εντολή δόθηκαν δικαιώματα για προβολή, επεξεργασία, δημιουργία και διαγραφή πινάκων σε όλες τις βάσεις δεδομένων που περιέχει ο διακομιστής MySQL, στον λογαριασμό kostas.



`

### 1.1.5 MySQL Workbench

Το MySQL WorkBench είναι μια εφαρμογή που διατίθεται από την Oracle η οποία έχει την δυνατότητα για μοντελοποίηση δεδομένων, σχεδίαση βάσεων δεδομένων, ανάπτυξη SQL, όπου μπορεί κανείς να «τρέξει» ερωτήματα SQL για προβολή δεδομένων, και δημιουργία πινάκων, επεξεργασία τους και διαγραφή. Επίσης υπάρχει η δυνατότητα για διαχείριση του διακομιστή με δημιουργία αντιγράφων ασφαλείας, διαχείριση ασφαλείας.




Εικόνα 11. MySQL WorkBench

## 1.2 PostgreSQL

Η PostgreSQL είναι ένα ανοικτού κώδικα αντικειμενοστραφές και σχεσιακό σύστημα διαχείρισης βάσεων δεδομένων. Λειτουργεί σε όλα τα κύρια λειτουργικά συστήματα όπως Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64), και Windows. Είναι πλήρως συμβατό με το ACID (Atomicity, Consistency, Isolation, Durability), έχει πλήρη υποστήριξη για ξένα κλειδιά, joins, views, triggers και αποθηκευμένες διαδικασίες (σε πολλές γλώσσες). Περιλαμβάνει τους περισσότερους τύπους δεδομένων όπως ορίζονται από το πρότυπο SQL:2008, όπως INTEGER, NUMERIC, BOOLEAN, CHAR, VARCHAR, DATE, INTERVAL, και TIMESTAMP. Υποστηρίζει επίσης την αποθήκευση μεγάλων δυαδικών αρχείων(binary large objects), όπως φωτογραφία, ήχο και βίντεο. Έχει έμφυτες διασυνδέσεις, μεταξύ άλλων, για C/C++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC.

Η PostgreSQL περιέχει χαρακτηριστικά όπως Multi-Version Concurrency Control (MVCC), αποκατάσταση σε προηγούμενο χρόνο (point in time recovery), tablespaces, ασύγχρονη αναπαραγωγή (asynchronous replication),ενσωματωμένες συναλλαγές (nested transactions (savepoints)), online/hot backups, ένα εξελιγμένο σχεδιαστή/βελτιστοποιητή ερωτημάτων,και από τα πριν εγγραφή καταγραφών για ανοχή λαθών. Υποστηρίζει διεθνή σύνολα χαρακτήρων, κωδικοποιήσεις χαρακτήρων πολλών byte, Unicode, και αντιλαμβάνεται τοπικούς χαρακτήρες για ταξινόμηση, διάκριση πεζών-κεφαλαίων, και διάταξη. Είναι πλήρως επεκτάσιμο και στον απόλυτο αριθμό των δεδομένων που μπορεί να διαχειριστεί και στον αριθμό των ταυτόχρονα συνδεδεμένων χρηστών που μπορεί να δεχτεί. Υπάρχουν ενεργά συστήματα PostgreSQL σε περιβάλλοντα παραγωγής που μπορούν να διαχειριστούν πάνω από 4 terabyte δεδομένων. Ακολουθεί πίνακας με τις δυνατότητες της PostgreSQL.

Πίνακας 1: Όρια της PostgreSQL

| | |
|---|---|
| Μέγιστο μέγεθος βάσης δεδομένων | Απεριόριστο |
| Μέγιστο μέγεθος πίνακα | 32 TB |
| Μέγιστο μέγεθος γραμμής | 1.6 TB |
| Μέγιστο μέγεθος πεδίου | 1 GB |
| Μέγιστος αριθμός γραμμών ανά πίνακα | Απεριόριστο |
| Μέγιστος αριθμός στηλών ανά πίνακα | 250 - 1600 ανάλογα με τους τύπους των στηλών |
| Μέγιστος αριθμός ευρετηρίων ανά πίνακα | Απεριόριστο |



`

Η PostgreSQL ακολουθεί αυστηρά τις προδιαγραφές του ANSI-SQL:2008. Έχει πλήρη υποστήριξη για υποερωτήματα (περιλαμβανόμενα τα subselect στο πεδίο FROM), επίπεδα δεσμευμένα για ανάγνωση και απομονωμένα επίπεδα σειριακής συναλλαγής. Καθώς η PostgreSQL έχει πλήρως σχεσιακό κατάλογο συστήματος το οποίο υποστηρίζει πολλαπλά σχήματα σε κάθε βάση δεδομένων, ο κατάλογός του είναι επίσης προσβάσιμος μέσω του Information Schema όπως καθορίζεται από το προτυπο SQL.

Τα χαρακτηριστικά της ακεραιότητας δεδομένων περιλαμβάνει κύρια κλειδιά, ξένα κλειδιά με περιορισμένα και επικαλυπτόμενα (restricting and cascading) updates/deletes, περιορισμοί με ελέγχους (check constraints), μοναδικοί περιορισμοί (unique constraints), και μη μηδενικοί περιορισμοί (not null constraints).

Έχει επίσης πλήθος από επεκτάσεις και προχωρημένα χαρακτηριστικά. Ανάμεσα στις ευκολίες είναι η αυτόματη συμπλήρωση στηλών μέσω ακολουθιών και LIMIT/OFFSET επιτρέποντας την επιστροφή μέρους από σύνολα αποτελεσμάτων(result sets). Η PostgreSQL υποστηρίζει συμπυκνωμένα, μερικά, και λειτουργικά ευρετήρια που μπορούν να χρησιμοποιήσουν οποιαδήποτε από τις B-tree, R-tree, hash, ή GIST μεθόδους αποθήκευσης.

Το GiST (Generalized Search Tree) είναι ένα προχωρημένο σύστημα ευρετηρίασης που ενώνει μια μεγάλη γκάμα από διαφορετικούς αλγόριθμους ταξινόμησης και αναζήτησης, περιλαμβάνοντας τα B-Tree, B+-Tree, R-Tree, δέντρα μερικού αθροίσματος (partial sum trees), B+-Tree κατάταξης, και πολλά άλλα. Προσφέρει επίσης μια διασύνδεση που επιτρέπει και την δημιουργία ορισμένων από τον χρήστη τύπων δεδομένων καθώς και επεκτάσιμες μεθόδους για εύρεση αυτών. Με αυτόν τον τρόπο το GiST προσφέρει την ευελιξία να καθορίσει κανείς τι αποθηκεύει, πως το αποθηκεύει και την ικανότητα να ορίσει νέους τρόπους αναζήτησής του, τρόπους που ξεπερνούν αυτούς που προσφέρουν τα συνηθισμένα B-Tree, R-Tree, και οι άλλοι γενικευμένοι αλγόριθμοι αναζήτησης.

Το GiST λειτουργεί ως θεμέλιο για πολλά κοινά προγράμματα που χρησιμοποιεί η PostgreSQL όπως το OpenFTS και το PostGIS. Το OpenFTS(Open Source Full Text Search Engine) προσφέρει καταλογοποίηση δεδομένων σε απευθείας σύνδεση και κατάταξη σχετικότητας για αναζήτηση σε βάσεις δεδομένων. Το PostGIS είναι ένα πρόγραμμα που προσθέτει υποστήριξη για γεωγραφικά αντικείμενα στην PostgreSQL, επιτρέποντάς την να χρησιμοποιηθεί ως χωρική βάση δεδομένων για γεωγραφικά πληροφοριακά συστήματα (Geographic Information Systems – GIS).

Άλλα προχωρημένα χαρακτηριστικά είναι η κληρονομικότητα πινάκων, το σύστημα κανόνων, και γεγονότα βάσης δεδομένων. Η κληρονομικότητα πινάκων προσθέτει αντικειμενοστρέφεια στην δημιουργία πινάκων, επιτρέποντας τους προγραμματιστές βάσεων δεδομένων να παράγουν νέους πίνακες από άλλους,



`

χειρίζοντας τους ως κλάσεις βάσης. Ακόμα περισσότερο η PostgreSQL υποστηρίζει και μονή και πολλαπλή κληρονομικότητα με αυτόν τον τρόπο.

Το σύστημα κανόνων, ή αλλιώς το σύστημα επανεγγραφής ερωτημάτων (query rewrite system), επιτρέπει τον σχεδιαστή της βάσης δεδομένων να δημιουργεί κανόνες οι οποίοι αναγνωρίζουν συγκεκριμένες λειτουργίες για έναν δοσμένο πίνακα ή view και δυναμικά τις μετατρέπουν σε αναπληρωματικες ενέργειες όταν γίνουν αντικείμενο επεξεργασίας.

Το σύστημα γεγονότων είναι ένα σύστημα επικοινωνίας μεταξύ των διεργασιών, μέσα στο οποίο μηνύματα και γεγονότα μπορούν να μεταφερθούν μεταξύ του πελάτη και του διακομιστή χρησιμοποιώντας τις εντολές LISTEN και NOTIFY, επιτρέποντας και την απλή επικοινωνία χρήστη προς χρήστη (Peer to Peer) και προηγμένο συντονισμό σε γεγονότα της βάσης δεδομένων. Από την στιγμή που κοινοποιήσεις μπορούν να εκδοθούν από triggers και αποθηκευμένες διαδικασίες, οι πελάτες PostgreSQL μπορούν να παρακολουθούν τα γεγονότα της βάσης δεδομένων όπως UPDATE, INSERT, ή DELETE την στιγμή που συμβαίνουν στον πίνακα.

Η PostgreSQL μπορεί να τρέξει αποθηκευμένες διαδικασίες σε πολλές γλώσσες προγραμματισμού όπως Java, Perl, python, Ruby, Tcl, C/C++, και την δική της γλώσσα PL/pgSQL, που είναι παρόμοια με την PL/SQL της Oracle. Συμπεριλαμβανόμενες στην βιβλιοθήκη προτυπων συναρτήσεων είναι εκατοντάδες ενσωματωμένες συναρτήσεις που κυμαίνονται από απλές μαθηματικές και String λειτουργίες ως κρυπτογραφία και συμβατότητα με την Oracle. Οι trigger και οι αποθηκευμένες διαδικασίες μπορούν να γραφτούν σε C και να φορτωθούν στην βάση δεδομένων ως μια βιβλιοθήκη, επιτρέποντας μεγάλη ευελιξία και επέκταση των δυνατοτήτων της. Παρόμοια η PostgreSQL περιλαμβάνει ένα framework που επιτρέπει στους προγραμματιστές να καθορίσουν και να δημιουργήσουν τους δικούς τους τύπους δεδομένων παράλληλα με τις υποστηρικτικές συναρτήσεις και λειτουργίες που καθορίζουν την συμεριφορά τους. Ως αποτέλεσμα, ένα σύνολο από προηγμένους τύπους δεδομένων έχουν δημιουργηθεί που περιέχουν από γεωμετρικά και χωρικά αρχικά ως διευθύνσεις δικτύου , ακόμα και τύπους δεδομένων ISBN/ISSN (International Standard Book Number/International Standard Serial Number), όλα από τα οποία μπορούν να προστεθούν στο σύστημα.

Όπως υπάρχουν πολλές διαδικαστικές γλώσσες που υποστηρίζονται από την PostgreSQL , έτσι υπάρχουν πολλές διασυνδέσεις βιβλιοθηκών, επιτρέποντας διάφορες γλώσσες να μεταγλωττίζονται και να διερμηνεύονται σε διασύνδεση με την PostgreSQL. Υπάρχουν διασυνδέσεις για Java (JDBC), ODBC, Perl, Python, Ruby, C, C++, PHP, Lisp, Scheme και Qt ανάμεσα σε άλλες.

Ο πηγαίος κώδικας της PostgreSQL είναι διαθέσιμος με μια ελεύθερη ανοικτού κώδικα άδεια, την PostgreSQL License. Αυτή η άδεια δίνει την ελευθερία για



`

χρήση, επεξεργασία και διανομή της PostgreSQL σε οποιαδήποτε μορφή, ανοικτού ή κλειστού κώδικα.

Τεχνικά χαρακτηριστικά της PostgreSQL:

- Πλήρως συμβατή με το ACID
- Συμβατότητα με το πρότυπο ANSI SQL
- Ακεραιότητα αναφορών
- Αναπαραγωγή, επιτρέπει την αντιγραφή της αρχικής βάσης σε πολλαπλές δευτερεύοντες βάσεις.
- Έμφυτες διασυνδέσεις για ODBC, JDBC, .Net, C, C++, PHP, Perl, TCL, ECPG, Python, και Ruby
- Κανόνες
- Views
- Triggers
- Unicode
- Ακολουθίες
- Κληρονομικότητα
- Εξωτερικές ενώσεις (outer joins)
- Sub-select
- Ανοικτό API
- Αποθηκευμένες διαδικασίες
- Έμφυτη υποστήριξη SSL
- Αναμονή (για τις εμπορικές λύσεις)
- Καλύτερο κλείδωμα από το επίπεδο γραμμής
- Λειτουργικοί και μερικοί κατάλογοι
- Έμφυτη πιστοποίηση Kerberos
- Υποστήριξη για ερωτήματα UNION, UNION ALL και EXCEPT
- Φορτώσιμες επεκτάσεις για SHA1, MD5, XML και άλλες λειτουργίες
- Εργαλεία για παραγωγή φορητού κώδικα SQL και μοίρασμα με άλλα συστήματα συμβατά με SQL
- Σύστημα επέκτασης τύπων δεδομένων που προσφέρει ορισμένους από τον χρήστη τύπους δεδομένων και γρήγορη ανάπτυξη νέων τύπων δεδομένων.
- Συναρτήσεις για συμβατότητα με άλλες βάσεις δεδομένων για ευκολία μετάβασης από άλλα συστήματα διαχείρισης βάσεων δεδομένων (DBMS) λιγότερο συμβατά με την SQL.



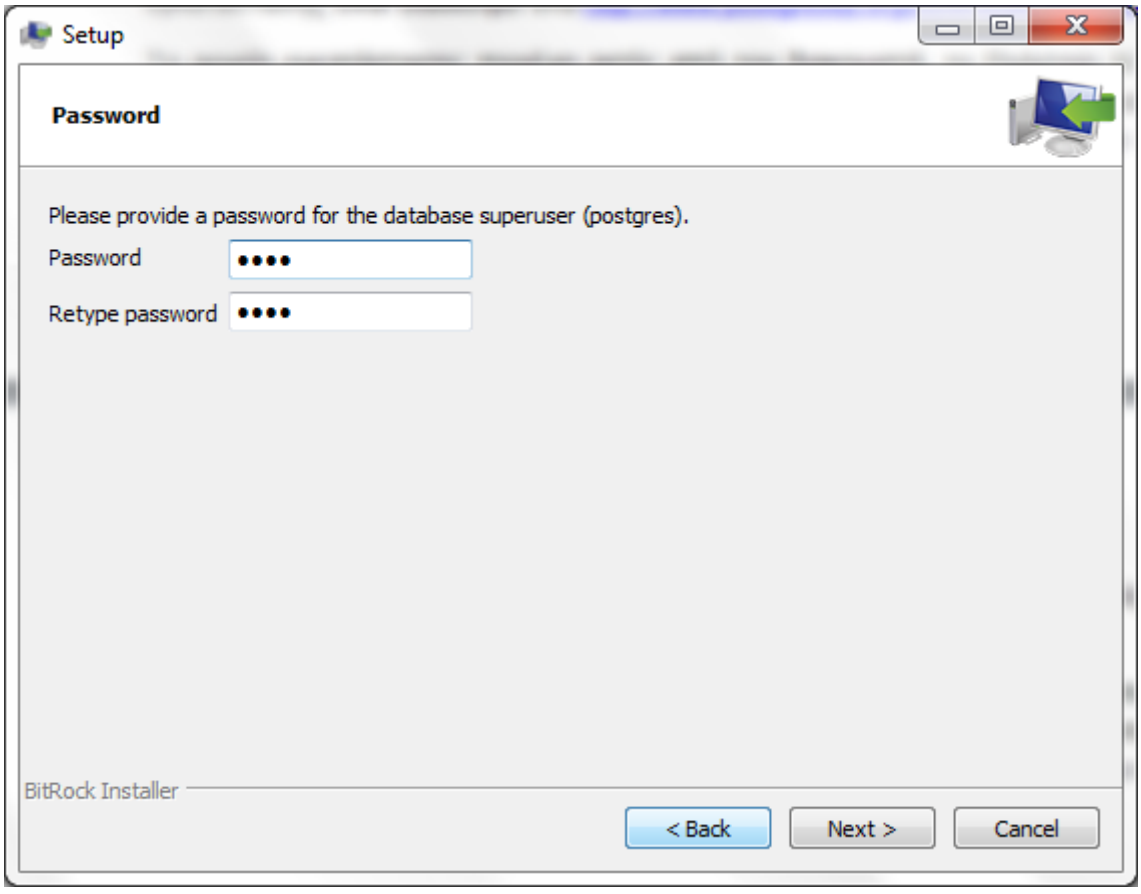
`

### 1.2.1 Εγκατάσταση PostgreSQL

Η PostgreSQL διατίθεται για λειτουργικό Microsoft Windows και το αρχείο εγκατάστασης είναι διαθέσιμο στο http://www.postgresql.org/download/windows/.

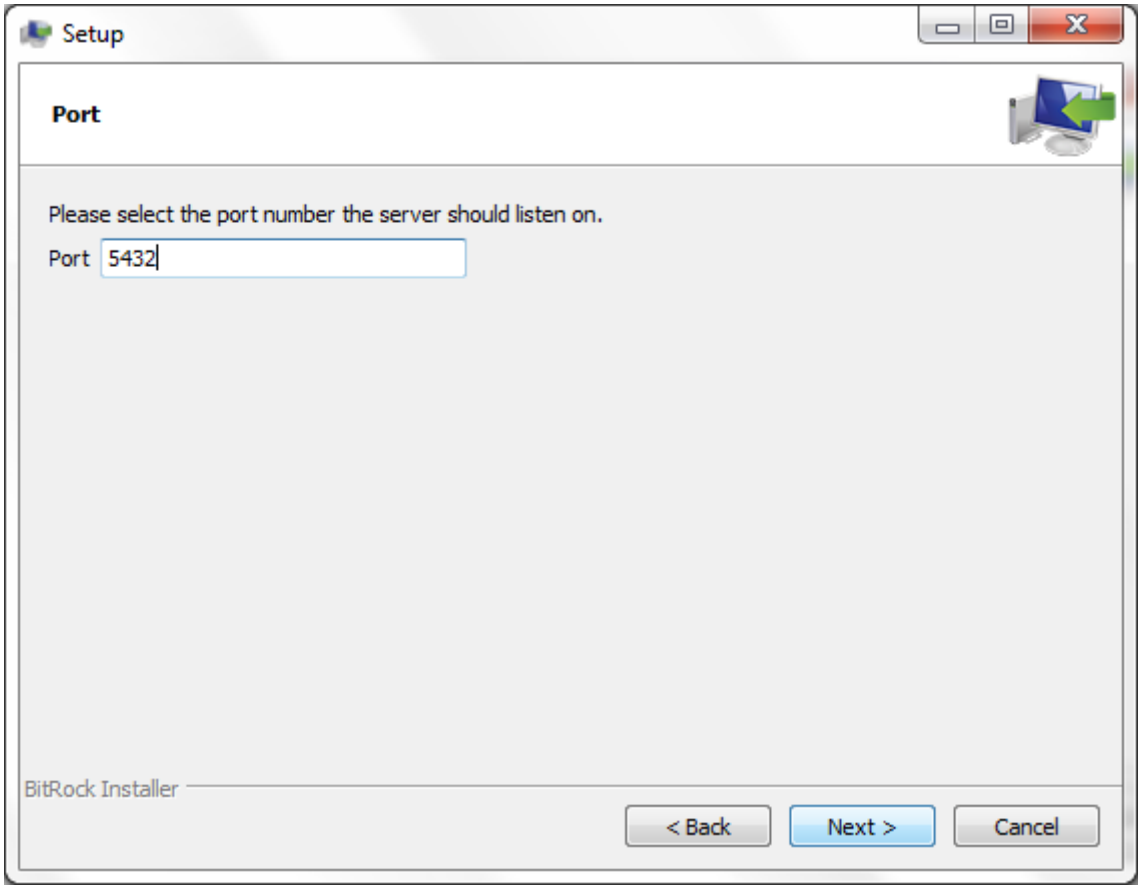
Το αρχείο εγκατάστασης περιέχει εκτός από τον διακομιστή, το PgAdmin III που είναι ένα γραφικό περιβάλλον-πελάτης για τον διακομιστή PostgreSQL για την διαχείριση και ανάπτυξη βάσεων δεδομένων και το StackBuilder που είναι ένας διαχειριστής πακέτων που χρησιμεύει στο να εγκαθιστά επιπλέον οδηγούς και εφαρμογές για την PostgreSQL.

Κατά την διαδικασία εγκατάστασης ζητούνται στοιχεία από τον διαχειριστή όπως ο φάκελος εγκατάστασης του διακομιστή, ο φάκελος στον οποίο θα βρίσκονται τα δεδομένα και ο κωδικός για τον λογαριασμό διαχειριστή,το όνομα του οποίου στην PostgreSQL από προεπιλογή είναι postgres.




Εικόνα 12. Ρύθμιση κωδικού διαχειριστή postgres

Στην συνέχεια ζητείται η θύρα δικτύου στην οποία θα «ακούει» ο διακομιστής. Αφήνεται στην προεπιλογή «5432».




Εικόνα 13. Θύρα για λήψη συνδέσεων του διακομιστή PostgreSQL

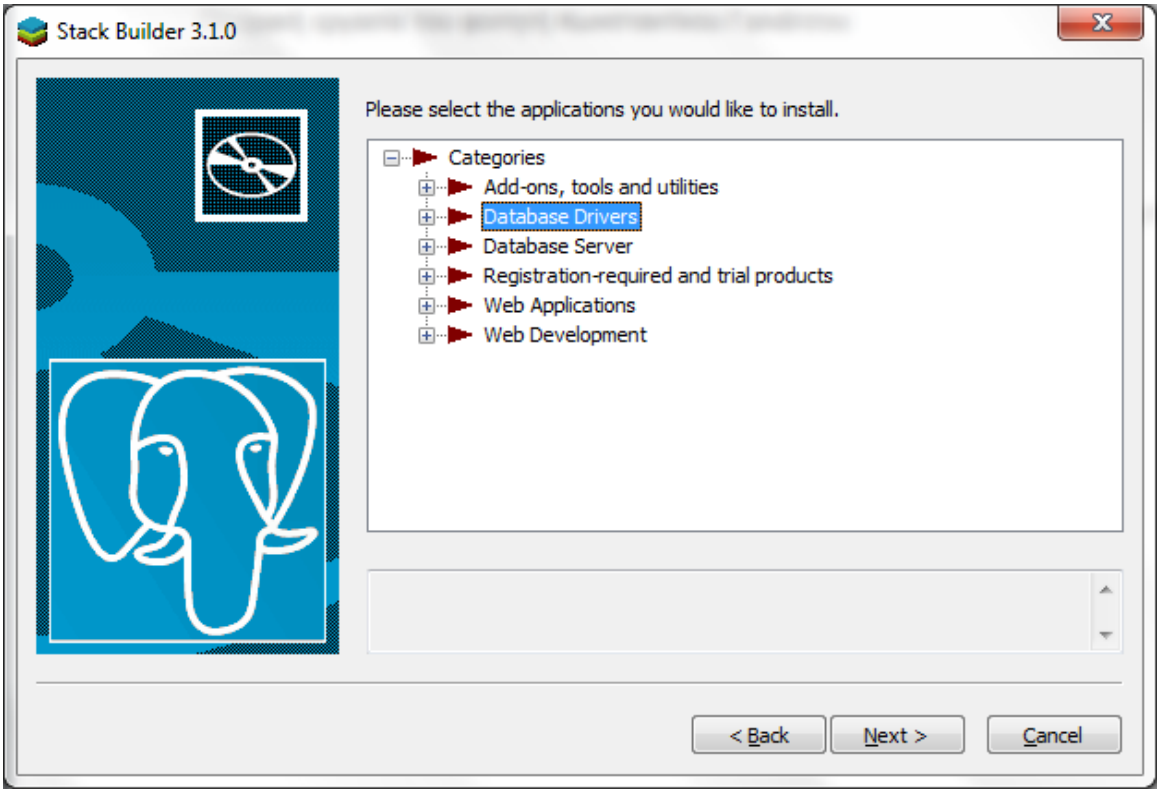
Επόμενη ρύθμιση είναι οι ρυθμίσεις τοπικότητας όπου αφήνεται η προεπιλογή και στην συνέχεια πραγματοποιείται η εγκατάσταση του διακομιστή και των εργαλείων που τον συνοδεύουν.



`

### 1.2.2 StackBuilder

Ο StackBuilder όπως ειπώθηκε προηγουμένως είναι μια εφαρμογή για εγκατάσταση οδηγών και εφαρμογών για τον διακομιστή PostgreSQL.




Εικόνα 14. StackBuilder

Από την κατηγορία Database Drivers εγκαθιστούνται οι οδηγοί για JDBC και ODBC, που θα χρειαστούν σε επόμενα κεφάλαια.

### 1.2.3 PgAdmin III

Το PgAdmin III είναι ένα μια εφαρμογή γραφικού περιβάλλοντος – πελάτης για την PostgreSQL, που έχει όλες τις δυνατότητες για την ανάπτυξη και διαχείριση βάσεων δεδομένων, όπως διαχείριση πινάκων και δεδομένων, διαχείριση εναυσμάτων, δημιουργία και αποκατάσταση αντιγράφων ασφαλείας και άλλα.

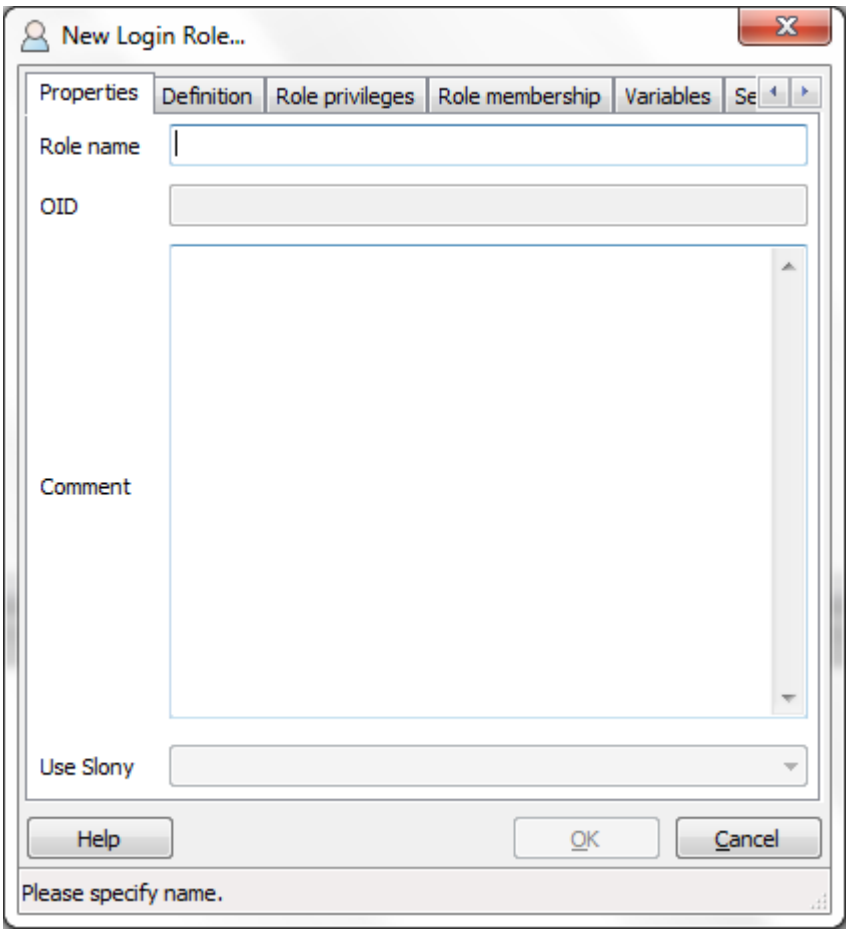


`

### 1.2.4 Διαχείριση λογαριασμών

Θα δημιουργηθεί ένας νεός λογαριασμός χρησιμοποιώντας τις δυνατότητες του PdAdmin III.

Από την γραμμή μενού επιλέγεται το Edit μετά New Object και New Login Role. Στην συνέχεια ανοίγει ένα νέο παράθυρο για την συμπλήρωση των στοιχείων του νέου λογαριασμού, το οποίο φαίνεται στην Εικόνα 16.




Εικόνα 15. Παράθυρο δημιουργίας λογαριασμού για τον διακομιστή PostgreSQL

Στην καρτέλα Properties συμπληρώνεται το όνομα του λογαριασμού, όπως ζητείται από το Role name. Στην καρτέλα Definition ζητείται ο κωδικός του χρήστη και στην καρτέλα Role privileges συμπληρώνονται τα δικαιώματα του λογαριασμού. Επιλέγεται να μπορεί να δημιουργεί αντικείμενα βάσεων δεδομένων.

Ο κώδικας SQL που αντιστοιχεί στην δημιουργία αυτού του λογαριασμού είναι:

```
CREATE ROLE kostas LOGIN
  CREATEDB
   VALID UNTIL 'infinity';
```

όπως αυτόματα συμπληρώνεται στην τελευταία καρτέλα “SQL” του παραθύρου δημιουργίας λογαριασμών.

## ΕΠΙΛΟΓΟΣ

Η MySQL και η PostgreSQL έχουν πλήθος χαρακτηριστικών, όπως υποστήριξη για πολλά λειτουργικά συστήματα, υποστήριξη πολλών επεξεργαστών και γρήγορη αναζήτηση που τα καθιστά κατάλληλα για απαιτητικά περιβάλλοντα.

Παράλληλα υπάρχει η δυνατότητα να δημιουργηθεί ένα πρόγραμμα πελάτης και να συνδεθεί με το σύστημα διαχείρισης βάσεων δεδομένων χρησιμοποιώντας ποικίλους τρόπους διασύνδεσης με τις βάσεις δεδομένων, οι οποίοι θα εξεταστούν στο επόμενο κεφάλαιο.



`

## 2. Τρόποι διασύνδεσης με τις Βάσεις Δεδομένων Ανοικτού Κώδικα

### ΕΙΣΑΓΩΓΗ

Σε αυτό το κεφάλαιο εξετάζονται οι διάφοροι τρόποι διασύνδεσης γλωσσών προγραμματισμού, frameworks, και περιβάλλοντα ανάπτυξης προγραμμάτων (IDEs) με τις βάσεις δεδομένων που εξετάζονται, καθώς και δυνατότητες αυτών των διασυνδέσεων.

### 2.1 MySQL Connectors

Οι MySQL Connectors προσφέρουν διασύνδεση του διακομιστή MySQL με προγράμματα πελάτη. Το API προσφέρει χαμηλού επιπέδου πρόσβαση στο προτόκολο και στους πόρους MySQL. Και οι Connectors και το API δίνουν την δυνατότητα σύνδεσης και εκτέλεσης προτάσεων MySQL από άλλη γλώσσα ή περιβάλλον, περιλαμβανόμενα την Java (JDBC), ODBC, Perl, Python, PHP, Ruby, και native C και ενσωματωμένα MySQL instances.

Μια σειρά από Connectors έχουν αναπτυχθεί από την MySQL:

- Ο Connectors/ODBC προσφέρει έναν οδηγό που υποστηρίζει την σύνδεση σε έναν διακομιστή MySQL χρησιμοποιώντας το Open Database Connectivity (ODBC) API. Η υποστήριξη για διασύνδεση ODBC είναι διαθέσιμη από τις πλατφόρμες Windows, Unix και Mac OS X.
- Ο Connector/Net δίνει την δυνατότητα για την δημιουργία εφαρμογών .NET που χρησιμοποιούν δεδομένα αποθηκευμένα σε μια βάση δεδομένων MySQL. Ο Connector/Net υλοποιεί μια πλήρως λειτουργική διασύνδεση ADO.NET και προσφέρει υποστήριξη για χρήση με εργαλεία ADO.NET aware. Εφαρμογές που θέλουν να χρησιμοποιήσουν το Connector/Net μπορούν να γραφτούν σε οποιαδήποτε από τις υποστηριζόμενες .NET γλώσσες.
- Το MySQL Visual Studio Plugin λειτουργεί με τον Connector/Net και το Visual Studio 2005. Το επιπρόσθετο είναι ένα MySQL DDEX Provider, που σημαίνει ότι μπορεί να χρησιμοποιήσει κανείς το σχήμα και τα εργαλεία για διαχείριση δεδομένων μέσα στο Visual Studio για την δημιουργία και επεξεργασία αντικειμένων μέσα σε μια βάση δεδομένων MySQL.
- Ο Connector/J προσφέρει υποστήριξη για τον οδηγό που συνδέει μια εφαρμογή Java στην MySQL χρησιμοποιώντας το προτυπο Java Database Connectivity (JDBC) API.



`

- Ο Connector/MXJ είναι ένα εργαλείο που επιτρέπει την ανάπτυξη και την διαχείριση ενός MySQL server και βάση δεδομένων δια μέσου μιας εφαρμογής Java.
- Ο Connector/C++ είναι ένα εργαλείο που επιτρέπει την ανάπτυξη και την διαχείριση ενός MySQL server και βάση δεδομένων δια μέσου μιας εφαρμογής C++.
- Ο Connector/C είναι μια αυτόνομη αντικατάσταση για την βιβλιοθήκη πελάτη της MySQL (libmysqld).
- Ο Connector/OpenOffice.org είναι ένα εργαλείο που επιτρέπει την εφαρμογή OpenOffice.org να συνδεθεί στον διακομιστή MySQL.

## 2.2 JDBC

Το JDBC API είναι ένα πρότυπο για μεταξύ της γλώσσας προγραμματισμού Java και ένα μεγάλο πεδίο από βάσεις δεδομένων, συνδεσιμότητα ανεξαρτήτως βάσης δεδομένων. Το JDBC API προσφέρει ένα call-level API για πρόσβαση σε βάσεις δεδομένων βασισμένες στο πρότυπο SQL. Η τεχνολογία JDBC επιτρέπει την χρήση της γλώσσας προγραμματισμού Java για την εκμετάλευση των δυνατοτήτων «Γράψε μια φορά, τρέξε παντού» (“Write once, run anywhere”) για εφαρμογές που απαιτούν πρόσβαση σε επιχειρησιακά δεδομένα.

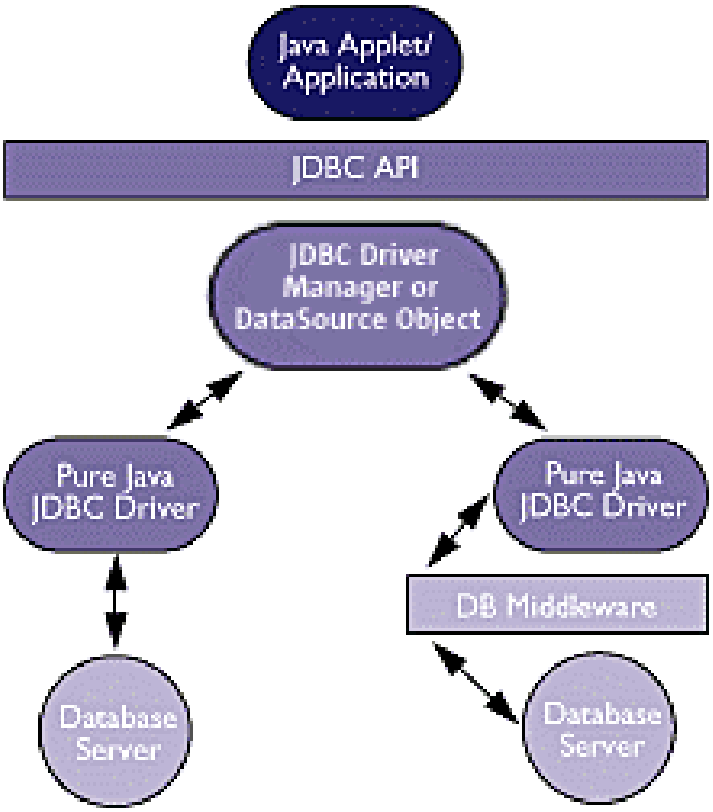
Το JDBC API δίνει την δυνατότητα για τρία πράγματα:

- Εγκαθίδρυση μιας σύνδεσης με μια βάση δεδομένων ή πρόσβαση σε οποιαδήποτε πινακοειδή πηγή δεδομένων.
- Αποστολή προτάσεων SQL.
- Επεξεργασία των αποτελεσμάτων.

Το JDBC API περιέχει δύο κύριες ομάδες από διασυνδέσεις: Η πρώτη είναι το JDBC API για προγραμματιστές εφαρμογών, και το δεύτερο είναι το low-level API για οδηγούς JDBC για προγραμματιστές οδηγών. Οι εφαρμογές και οι μικροεφαρμογές μπορούν να έχουν πρόσβαση σε βάσεις δεδομένων μέσω του JDBC API χρησιμοποιώντας οδηγούς βασισμένους στην τεχνολογία JDBC που είναι γραμμένη εξ’ολοκλήρου σε Java. Η πρόσβαση γίνεται όπως δείχνει η Εικόνα 2.



`




Εικόνα 16: Αρχιτεκτονική του JDBC

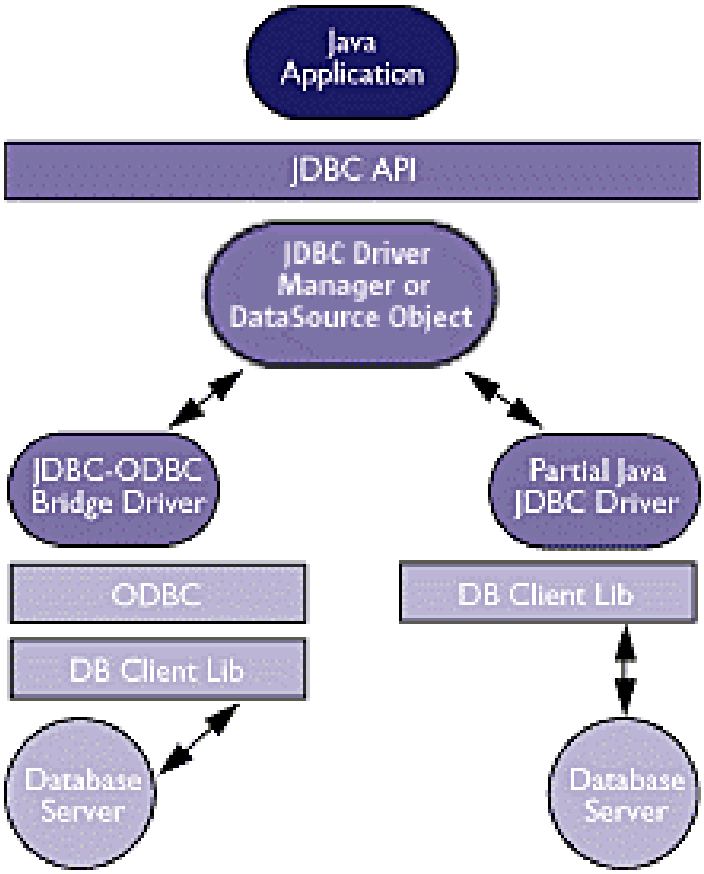
Οι οδηγοί JDBC χωρίζονται σε 4 κατηγορίες:

- JDBC Type 4: Οδηγός απευθείας σύνδεσης σε βάση δεδομένων, γραμμένος εξ'ολοκλήρου σε Java. Αυτός ο τύπος οδηγού μετατρέπει τις κλήσεις JDBC σε δικτυακό προτόκολο άμεσα χρησιμοποιημένο από τα DBMS, επιτρέποντας μια απευθείας κλήση από το μηχάνημα-πελάτης στον διακομιστή του συστήματος διαχείρισης βάσεων δεδομένων και προσφέρει μια πρακτική λύση για πρόσβαση σε δίκτυα intranet. Αυτή η μορφή παρουσιάζεται στην αριστερή πλευρά της Εικόνας 2.
- JDBC Type 3: Οδηγός γραμμένος εξ'ολοκλήρου σε Java για middleware βάσης δεδομένων. Αυτή η μορφή οδηγού μεταφράζει κλήσεις JDBC στο προτόκολο του middleware, το οποίο στη συνέχεια μεταφράζεται στο προτόκολο του DBMS από τον διακομιστή middleware. Το middleware προσφέρει συνδεσιμότητα σε πολλές διαφορετικές βάσεις δεδομένων. Αυτή η μορφή παρουσιάζεται στην δεξιά πλευρά της Εικόνας 2.
- JDBC Type 2: A native API partly Java technology-enabled driver. Αυτή η μορφή οδηγού μετατρέπει κλήσεις JDBC σε κλήσεις για το API των Oracle, Sybase, Informix, DB2, ή άλλα DBMS. Ας σημειωθεί ότι αυτός ο τύπος οδηγού απαιτεί δυαδικό κώδικα να φορτωθεί σε κάθε μηχάνημα πελάτη. Αυτή η μορφή παριστάνεται στην δεξιά πλευρά της Εικόνας 3.



`

- JDBC Type 1: Γέφυρα JDBC-ODBC συν ένας οδηγός ODBC. Αυτή η μορφή προσφέρει πρόσβαση με JDBC μέσω οδηγών ODBC . Σε κάθε μηχάνημα που χρησιμοποιεί Γέφυρα JDBC-ODBC πρέπει να φορτωθεί δυαδικός κώδικας ODBC – και σε πολλές περιπτώσεις, κώδικας πελάτη βάσης δεδομένων. Αυτή η μορφή παριστάνεται στην αριστερή πλευρά της Εικόνας 3.




Εικόνα 17: Αρχιτεκτονική του JDBC με χρήση ODBC και βιβλιοθήκες πελάτη βάσεων δεδομένων.

### 2.2.1 Υλοποίηση

Ο οδηγός MySQL Connector/J είναι διαθέσιμος στο http://dev.mysql.com/downloads/connector/j/3.1.html.

Η πρόσβαση ενός προγράμματος Java στον οδηγό JDBC μπορεί να γίνει μέσω της εφαρμογής Netbeans. Συγκεκριμένα στον κατάλογο των project πηγαίνουμε στο πρόγραμμα που αναπτύσσουμε, και με δεξί κλίκ στο αντικείμενο επιλέγουμε Προσθήκη JAR/φακέλου και επιλέγουμε το αρχείο .jar του οδηγού JDBC και πλέον ο οδηγός είναι διαθέσιμος για χρήση στο πρόγραμμά μας.



`

Με την εφαρμογή Eclipse το ίδιο γίνεται επιλέγοντας το “Migrate Jar File” που βρίσκεται στο Refactor στην γραμμή μενού και επιλέγοντας το αρχείο .jar του οδηγού.

Για την ανάπτυξη προγράμματος Java που χρησιμοποιεί JDBC για σύνδεση με βάση δεδομένων απαιτούνται τα εξής βήματα:

- Δήλωση του οδηγού JDBC
- Δημιουργία της σύνδεσης
- Εκτέλεση της πρότασης SQL
- Λήψη του αποτελέσματος (αν υπάρχει)
- Κλείσιμο της σύνδεσης

Αρχικά δημιουργούμε όλα τα αντικείμενα που θα χρειαστούν για την εκτέλεση των παραπάνω ενεργειών.

```
private static Connection dbConnection = null;
private static Statement statement = null;
private static ResultSet rs = null;
```

Το πρώτο βήμα γίνεται με την μέθοδο Class.forName(). Το όρισμα που δέχεται είναι το όνομα του οδηγού JDBC. Στην περίπτωση της MySQL είναι com.mysql.jdbc.Driver και της PostgreSQL org.postgresql.Driver.

```
Class.forName("com.mysql.jdbc.Driver");
```

Για την δημιουργία της σύνδεσης χρησιμοποιείται η μέθοδος DriverManager.getConnection() με ορίσματα το url της σύνδεσης, το όνομα χρήστη και τον κωδικό του. Η μορφή του url είναι jdbc:σύστημα βάσης δεδομένων://διακομιστής:πορτα/βάση δεδομένων, για παράδειγμα jdbc:mysql://localhost:3306/test.

```
dbConnection  = DriverManager.getConnection("jdbc:mysql://localhost:3306/test",
"username", "password");
```

Για την εκτέλεση της πρότασης SQL αρχικά απαιτείται ένα αντικείμενο τύπου Statement με το οποίο θα δημιουργήσουμε μια πρόταση και θα την εκτελέσουμε στον διακομιστή.

```
statement = dbConnection.createStatement();
rs = statement.executeQuery("SELECT * FROM athletes");
```

Αν ο τύπος της πρότασης SQL είναι τέτοιος που επιστρέφει αποτελέσματα τότε χρειάζεται και ένα αντικείμενο ResultSet το οποίο περιέχει τα επιστρεφόμενα αποτελέσματα.



`

```
while(rs.next()) {
        System.out.println(rs.getString(1) + " " + rs.getString(2));
}
```

Ένα ολοκληρωμένο παράδειγμα βρίσκεται στο Παράρτημα 1.

### 2.3 ODBC

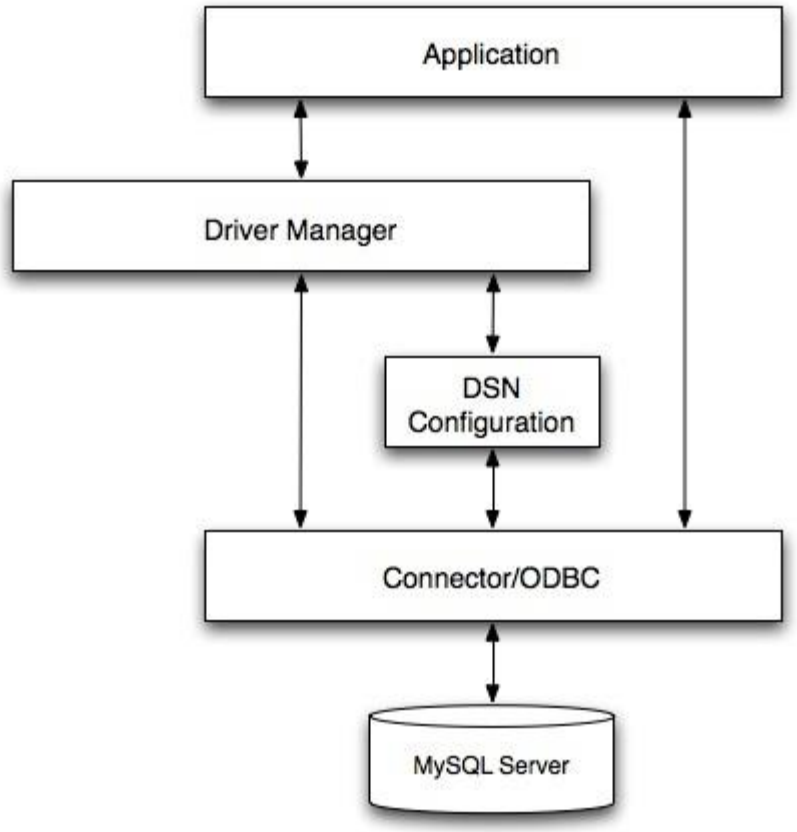
#### 2.3.1 MySQL Connector/ODBC

Το ODBC (Open Database Connectivity) προσφέρει έναν τρόπο για να έχουν πρόσβαση τα προγράμματα client σε ένα ευρύ πεδίο από βάσεις δεδομένων ή πηγές δεδομένων. Το ODBC είναι ένα τυποποιημένο API που ενεργοποιεί συνδέσεις σε servers βάσεων δεδομένων SQL. Αναπτύχθηκε σύμφωνα με τις προδιαγραφές του SQL Access Group και καθορίζει ένα σύνολο από κλήσεις συναρτήσεων, κωδικών σφαλμάτων και τύπων δεδομένων που μπορούν να χρησιμοποιηθούν για την ανάπτυξη εφαρμογών ανεξαρτήτως βάσης δεδομένων. Το ODBC συνήθως χρησιμοποιείται όταν απαιτείται ανεξαρτησία από βάση δεδομένων ή ταυτόχρονη σύνδεση σε πολλές πηγές δεδομένων. Το MySQL Connector/ODBC είναι το όνομα της οικογένειας των οδηγών MySQL ODBC, που δίνει πρόσβαση σε μια βάση δεδομένων MySQL χρησιμοποιώντας το πρότυπο Open Database Connectivity (ODBC) API. Το MySQL Connector/ODBC προσφέρει σύνδεση μέσω του οδηγού και native διασυνδέσεις στην βάση δεδομένων MySQL, με πλήρη υποστήριξη της λειτουργίας του συστήματος, συμπεριλαμβανομένων και των αποθηκευμένων διαδικασιών, συναλλαγών και με το Connector/ODBC 5.1 πλήρη συμβατότητα με Unicode.

Η αρχιτεκτονική του Connector/ODBC είναι βασισμένη σε 5 συστατικά, όπως φαίνεται στο παρακατω σχήμα:



`




Εικόνα 18: Η αρχιτεκτονική του MySQL Connector/ODBC

#### 2.3.1.1 Εφαρμογή

Η εφαρμογή χρησιμοποιεί το API του ODBC για να έχει πρόσβαση στα δεδομένα του διακομιστή MySQL. Το API του ODBC με την σειρά του επικοινωνεί με τον Διαχειριστή Οδηγού (Driver Manager), η εφαρμογή επικοινωνεί με τον Διαχειριστή Οδηγού χρησιμοποιώντας πρότυπες κλήσεις ODBC. Την εφαρμογή δεν την ενδιαφέρει που και πως είναι αποθηκευμένα τα δεδομένα, ούτε πως είναι ρυθμισμένο το σύστημα να έχει πρόσβαση στα δεδομένα. Το μόνο που χρειάζεται να γνωρίζει είναι το Data Source Name (DSN).

Ένας αριθμός από ενέργειες είναι κοινές σε όλες τις εφαρμογές ανεξάρτητα του τρόπου που χρησιμοποιούν το ODBC. Αυτές οι ενέργειες είναι:

- Επιλογή του διακομιστή MySQL και σύνδεση σε αυτόν
- Υποβολή προτάσεων για εκτέλεση



`

- Ανάκτηση αποτελεσμάτων (αν υπάρχουν)
- Επεξεργασία σφαλμάτων
- Εκτέλεση ή επαναφορά της συναλλαγής που περιέχει την πρόταση SQL
- Αποσύνδεση από τον διακομιστή MySQL

Επειδή η περισσότερη δουλειά της πρόσβασης στα δεδομένα γίνεται με SQL, οι κύριες εργασίες για τις εφαρμογές που χρησιμοποιούν ODBC είναι να υποβάλλουν προτάσεις SQL και να ανακτούν τα όποια αποτελέσματα παράγουν αυτές οι προτάσεις.

#### 2.3.1.2 Διαχειριστής Οδηγού (Driver Manager)

Ο διαχειριστής οδηγού είναι μια βιβλιοθήκη που διαχειρίζεται την επικοινωνία μεταξύ της εφαρμογής και του οδηγού ή οδηγών. Εκτελεί τις εξής ενέργειες:

- Επιλύει τα Data Source Name (DSN). Το DSN είναι ένα String διαμόρφωσης που αναγνωρίζει ένα δοσμένο οδηγό βάσης δεδομένων, μια βάση δεδομένων, τον host της βάσης και προαιρετικά τις πληροφορίες πιστοποίησης που επιτρέπει σε μια εφαρμογή ODBC να συνδεθεί σε μια βάση δεδομένων χρησιμοποιώντας μια προτυποποιημένη αναφορά. Επειδή οι πληροφορίες για την σύνδεση σε μια βάση δεδομένων αναγνωρίζεται από το DSN, κάθε εφαρμογή συμβατή με το ODBC μπορεί να συνδεθεί στην πηγή δεδομένων χρησιμοποιώντας την ίδια αναφορά DSN. Αυτό μειώνει την ανάγκη να ρυθμίζεται χωριστά κάθε εφαρμογή που χρειάζεται πρόσβαση σε μια δοσμένη βάση δεδομένων. Αντ' αυτού ρυθμίζεται η εφαρμογή να χρησιμοποιήσει ένα προρυθμισμένο DSN.
- Φορτώνει και αποφορτώνει τον οδηγό που απαιτείται για να έχει πρόσβαση σε μια συγκεκριμένη βάση δεδομένων, όπως ορίζεται από το DSN. Για παράδειγμα αν έχει ρυθμιστεί ένα DSN να συνδέεται σε μια βάση δεδομένων MySQL τότε ο διαχειριστής οδηγού φορτώνει τον οδηγό Connector/ODBC για να επιτρέψει το ODBC API να επικοινωνεί με τον MySQL host.
- Επεξεργάζεται κλήσεις συναρτήσεων του ODBC ή τις μεταφέρει στον οδηγό για επεξεργασία.

#### 2.3.1.3 Οδηγός Connector/ODBC

Ο οδηγός Connector/ODBC είναι μια βιβλιοθήκη που εφαρμόζει τις κλήσεις που υποστηρίζεται από το ODBC API. Επεξεργάζεται τις κλήσεις συναρτήσεων ODBC, υποβάλει αιτήματα SQL στον διακομιστή MySQL και επιστρέφει τα αποτελέσματα πίσω στην εφαρμογή. Αν είναι απαραίτητο, ο οδηγός τροποποιεί κάποιο αίτημα της εφαρμογή ώστε αυτό να ακολουθεί τη σύνταξη που υποστηρίζει η MySQL.



`

#### 2.3.1.4 Ρύθμιση DSN

Τα αρχεία ρύθμισης του ODBC αποθηκεύουν τις πληροφορίες για τον οδηγό και την βάση δεδομένων που απαιτούνται για την σύνδεση στον διακομιστή. Χρησιμοποιούνται από τον διαχειριστή οδηγού για να καθορίσει το ποιός οδηγός θα φορτωθεί σύμφωνα με τον ορισμό του DSN. Ο οδηγός το χρησιμοποιεί αυτό για να διαβάσει τις παραμέτρους της σύνδεσης σύμφωνα με το DSN που προσδιορίστηκε.

### 2.3.2 Υλοποίηση

Για την υλοποίηση εφαρμογής σε Java η οποία συνδέεται μέσω οδηγού ODBC είναι απαραίτητα τα εξής βήματα:

1. Εγκατάσταση του οδηγού ODBC
2. Ρύθμιση του οδηγού
3. Υλοποίηση της εφαρμογής

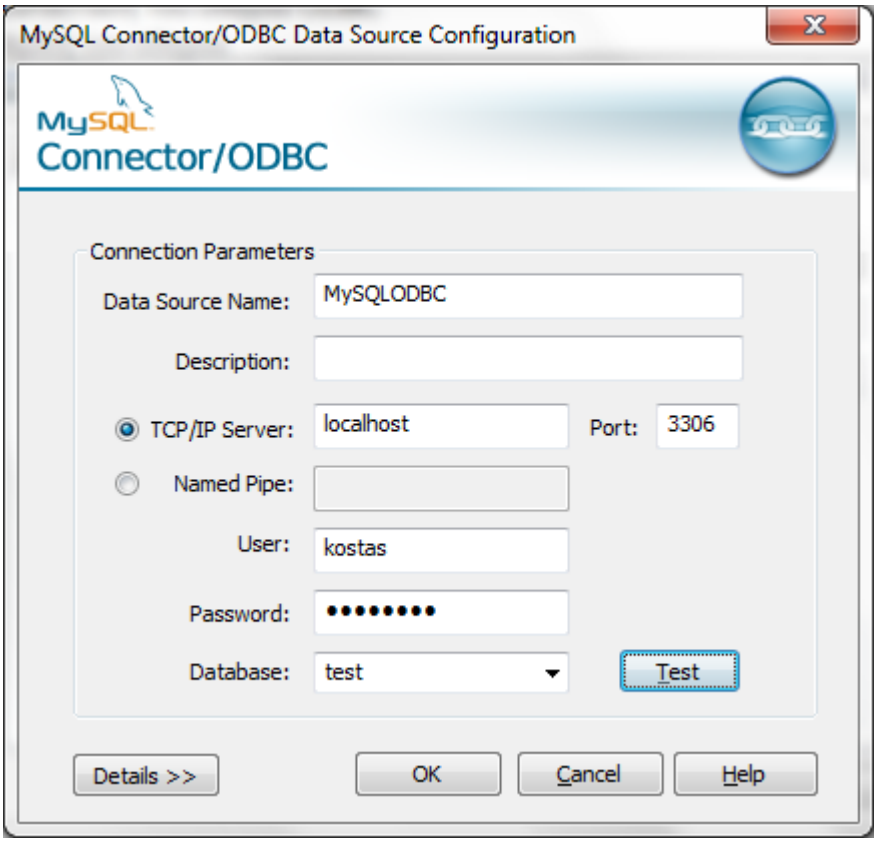
Για το πρώτο βήμα το αρχείο εγκατάστασης του οδηγού ODBC για την βάση δεδομένων MySQL βρίσκεται στο

http://dev.mysql.com/downloads/connector/odbc/.

Στην συνέχεια ρυθμίζουμε το DSN για την σύνδεση στην βάση δεδομένων. Η εφαρμογή για την ρύθμιση είναι «Πηγές λογισμικού (ODBC)». Στην δεύτερη καρτέλα της εφαρμογής μπορεί κανείς να ρυθμίσει μια σύνδεση ODBC για οποιονδήποτε χρήστη στο σύστημα. Επιλέγουμε «Προσθήκη» και μετά το «MySQL ODBC <έκδοση> Driver» από την λίστα των οδηγών ODBC που είναι διαθέσιμοι στο σύστημα. Αμέσως μετά ανοίγει εφαρμογή για την ρύθμιση του DSN του οδηγού. Εκεί εισάγουμε το όνομα για την συγκεκριμένη σύνδεση, το οποίο χρησιμοποιείται από την εφαρμογή. Εδώ εισήχθη το όνομα «MySQLODBC». Απαιτείται επίσης η διεύθυνση IP του υπολογιστή που βρίσκεται ο διακομιστής MySQL, και η θύρα στην οποία αυτός «ακούει». Ζητούνται και τα στοιχεία του λογαριασμού με τον οποίο θα γίνεται η εκτέλεση SQL κώδικα, καθώς και η βάση δεδομένων που θα χρησιμοποιηθεί.



`




Εικόνα 19. Ρύθμιση DSN για το MySQL Connector/ODBC

Για την υλοποίηση εφαρμογής σε Java η οποία θα συνδέεται στην βάση δεδομένων SQL μέσω του οδηγού ODBC είναι απαραίτητη η χρήση JDBC – ODBC γέφυρας (JDBC-ODBC bridge) καθώς δεν είναι δυνατή η απευθείας σύνδεση εφαρμογών Java με οδηγούς ODBC.

Ο οδηγός-γέφυρα είναι ο “sun.jdbc.odbc.JdbcOdbcDriver”

Στην εφαρμογή αρχικά δημιουργούνται όλα τα απαραίτητα αντικείμενα:

```
Connection con = null;
Statement statement = null;
ResultSet rs = null;
String Driver = "sun.jdbc.odbc.JdbcOdbcDriver";
```

Και φορτώνεται ο οδηγός στην εφαρμογή με την μέθοδο Class.forName()

```
Class.forName(Driver);
```

Στην συνέχεια δημιουργείται η σύνδεση χρησιμοποιώντας για όνομα της σύνδεσης το “jdbc:odbc” ακολοθούμενο από το όνομα της σύνδεσης ODBC προς την βάση δεδομένων MySQL που είχε δημιουργηθεί προηγουμένως “MySQLODBC”.



`

```
con = DriverManager.getConnection("jdbc:odbc:MySQLODBC");
```

Στην συνέχεια γράφουμε τις απαραίτητες προτάσεις για την εκτέλεση κώδικα SQL

```
statement = con.createStatement();
query = "SELECT * FROM athletes";
rs = statement.executeQuery(query);
```

όπου στέλνεται το ερώτημα SQL μέσω του αντικειμένου statement στην βάση δεδομένων και τα αποτελέσματα λαμβάνονται από το αντικείμενο ResultSet (rs), με το οποίο θα παρουσιαστούν.

```
while(rs.next()) {
        System.out.print(rs.getString(1) + " ");
        System.out.println(rs.getString(2));
}
```

Ολόκληρη η εφαρμογή βρίσκεται στο Παράρτημα 2.



`

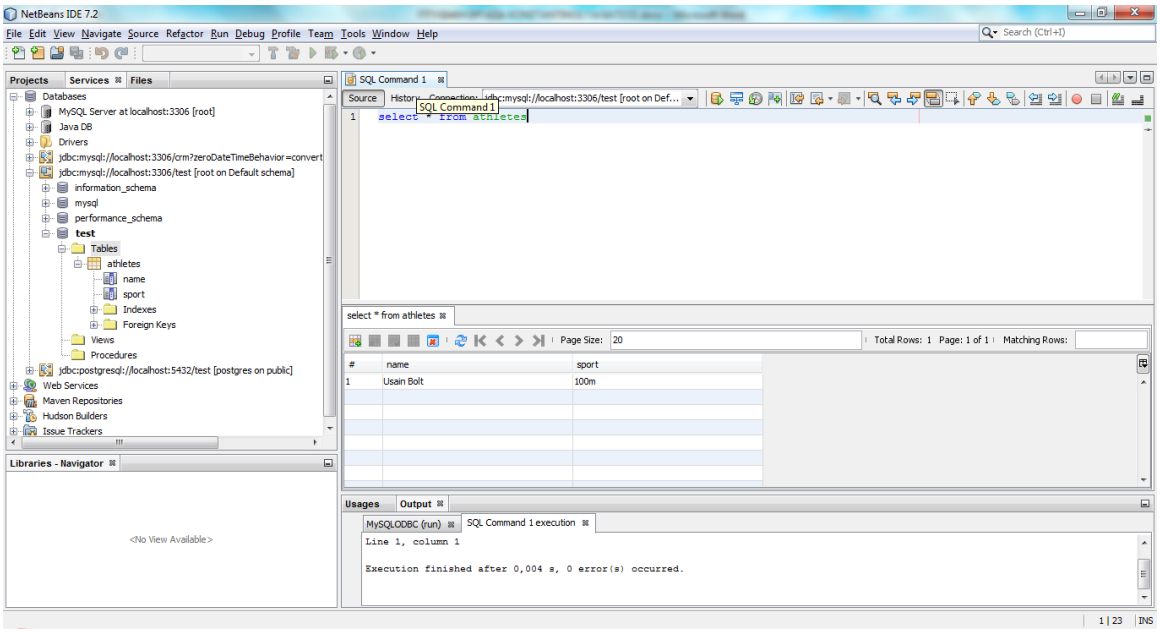
## 2.4 Netbeans IDE

Το Netbeans έρχεται με ενσωματωμένη υποστήριξη για τα συστήματα διαχείρισης βάσεων δεδομένων MySQL και PostgreSQL, μεταξύ άλλων.

Η χρήση των βάσεων δεδομένων γίνεται από την αριστερή εργαλειοθήκη στον κόμβο Databases. Δημιουργώντας μια νέα σύνδεση παρατηρούμε ότι το Netbeans διαθέτει οδηγούς JDBC για MySQL και PostgreSQL. Συνεχίζουμε εισάγοντας τα στοιχεία της σύνδεσης.

Οι διαθέσιμες λειτουργίες που διαθέτει το Netbeans είναι δημιουργία βάσης δεδομένων, δημιουργία πινάκων, εκτέλεση προτάσεων SQL και SQL scripts.

Επίσης για τα συστατικά γραφικών Swing της Java, το Netbeans δίνει την δυνατότητα για αυτά να εμφανίζουν τα περιεχόμενα ενός πίνακα, ή του αποτελέσματος ενός ερωτήματος SQL.




Εικόνα 20. Προβολή δεδομένων πίνακα μέσω Netbeans

## 2.5 Eclipse IDE

Το Eclipse δεν έχει ενσωματωμένη λειτουργία σύνδεσης και διαχείρισης βάσεων δεδομένων. Αυτό το προσφέρει το πρόσθετο Data Tools Platform (DTP).

Το DTP περιέχει ένα συνεκτικό σύνολο από εργαλεία για την δημιουργία, την διαχείριση και την χρήση δεδομένων.

Το DTP είναι προορισμένο για χρήση σε ανάπτυξη και διαχειριστικές εργασίες με κέντρο τα δεδομένα. Η γενική χρήση του DTP εφαρμόζεται σε προβλήματα με θέμα τα δεδομένα και δεν απευθύνεται συγκεκριμένα σε ήδη υπάρχοντες εργασίες στο Eclipse. Το DTP αποτελείται από διάφορα έργα, όπως το Model Base, Enablement, Connectivity και SQL Development Tools. Αυτά τα έργα προσφέρουν τα κύρια μοντέλα, βασικά πλαίσια του DTP και εργαλεία για συγκεκριμένες πηγές δεδομένων, λειτουργικότητα για διαχείριση των συνδέσεων, και εργαλεία ερωτημάτων απαραίτητα για την ανάπτυξη, πρόσβαση και χρήση δεδομένων.

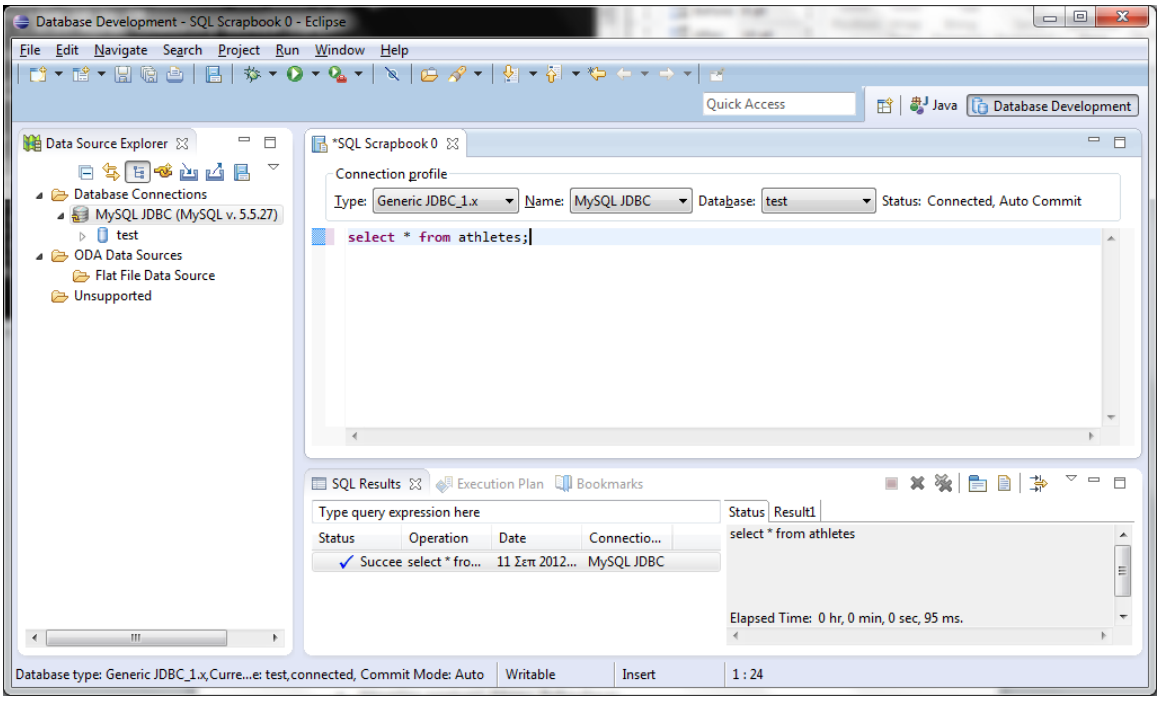
Το Data Tools Platform μπορεί να εγκατασταθεί στο Eclipse μέσω του “Install new software” που βρίσκεται στην γραμμή μενού στο Help.

Μετά την εγκατάσταση, η διαχείριση των βάσεων δεδομένων είναι διαθέσιμη στην πάνω δεξιά γωνία, στο κουμπί με όνομα “Open Perspective”, στην λίστα που ανοίγει επιλέγουμε το “Database Development”. Τότε στα αριστερά του eclipse υπάρχει ένα πλαίσιο με όνομα “Data Source Explorer”. Με δεξί κλικ στο αντικείμενο “Database Connections” επιλέγουμε New για να δημιουργήσουμε μια νέα σύνδεση σε βάση δεδομένων. Με επιλογή του κουμπιού δίπλα από την λίστα των οδηγών ανοίγει νεό παράθυρο, στο οποίο κατευθυνόμαστε στην καρτέλα “Jar List” και επιλέγουμε “Add JAR/ZIP”, όπου εισάγουμε την τοποθεσία του οδηγού JDBC και στην συνέχεια στην καρτέλα “Properties” εισάγουμε τα στοιχεία που απαιτούνται, όπως το url της σύνδεσης, όπου στην συγκεκριμένη περίπτωση για την MySQL είναι jdbc:mysql://localhost:3306/, η βάση δεδομένων που θα χρησιμοποιηθεί, για παράδειγμα test, το όνομα της κλάσης του οδηγού, για παράδειγμα com.mysql.jdbc.Driver και το όνομα του λογαριασμού που θα συνδεθεί στον διακομιστή της βάσης δεδομένων.

Μετά την ολοκλήρωση της σύνδεσης μπορεί κανείς να εκτελέσει κώδικα SQL κάνοντας δεξί κλίκ στην σύνδεση που δημιουργήθηκε στο Data Source Explorer και επιλογή του “Open SQL ScrapBook.



`




Εικόνα 21. Εκτέλεση SQL κώδικα στο eclipse

### 2.5.1 Model Base

Το DTP Model Base προσφέρει τα θεμέλια για την ανάπτυξη βάσεων δεδομένων. Χρησιμοποιεί πρακτικές όπως ανάπτυξη model-driven με UML, και το Eclipse Modeling Framework. Το Model Base αποτελείται από τα εξής στοιχεία:

- Μοντέλο ορισμού βάσης δεδομένων
- Μοντέλο επεξεργασίας
- Μοντέλο SQL
- Μοντέλο ερωτημάτων SQL
- Μοντέλο ερωτημάτων SQL XML

### 2.5.2 Connectivity

Το DTP Connectivity προσφέρει πόρους για την σύνδεση σε πηγές δεδομένων.

Η λειτουργία για διαχείριση συνδέσεων που προσφέρεται από το έργο Connectivity περιέχει τα εξής στοιχεία για τον ορισμό, την σύνδεση και την εργασία με πηγές δεδομένων.

- Driver Manager Network (DMF): προσφέρει μια σελίδα προτιμήσεων στο Eclipse δίνοντας την δυνατότητα στους χρήστες να δημιουργήσουν ορισμούς οδηγών βασισμένοι σε ήδη υπαρχοντα πρότυπα.

- Connection Manager Framework (CMF): Το CMF είναι η βάση για συγκεκριμένους τύπους συνδέσεων που λέγονται Connection Profiles (CP). Τα CP προστίθενται στο CMF μέσω σημείων επεκτάσεων. Η σύνδεση στις πηγές δεδομένων γίνεται την δημιουργία και την ρύθμιση των CP για κάθε τύπο πηγής δεδομένων. Με την δημιουργία ένος CP καθορίζονται παράμετροι, όπως το URL της σύνδεσης, το όνομα χρήστη και τον κωδικό.
- Υποστήριξη συνδέσεων JDBC: Το DTP περιλαμβάνει ένα πρότυπο ορισμού οδηγού JDBC και CP για την ενεργοποίηση της συνδεσιμότητας της βάσης βάσης δεδομένων.
- Περιηγητής πηγών δεδομένων: Ο περιηγητής πηγών δεδομένων είναι μια προβολή των οντοτήτων CP. Οι δυνατότητες του CP προβάλονται και τα περιεχόμενα των πηγών δεδομένων παρουσιάζονται μέσω αυτής της προβολής.

### 2.5.3 Εργαλεία Ανάπτυξης SQL

Τα εργαλεία ανάπτυξης SQL του DTP προσφέρουν πλαίσια και εργαλεία για υποστήριξη SQL. Το έργο αποτελείται από τα εξής στοιχεία:

- Routines Editor Framework: Το Routines Editor Framework είναι ένα επεκτάσιμο πλαίσιο για επεξεργασία ρουτινών βάσων δεδομένων και προτάσεων SQL.
- Routines Debugger Framework: Το Routines Debugger Framework προσφέρει μια επεκτάσιμη υποστήριξη για το debugging διαφόρων βάσεων δεδομένων με συνεπή τρόπο με την υπάρχουσα υποδομή debug του Eclipse.
- SQL Query Parser: Το SQL Query Parser προοσφέρει ένα επεκτάσιμο πλαίσιο που επιτρέπει συστατικά και εργαλεία SQL για διάφορες κύριες διαλέκτους SQL.
- SQL Query Builder: Το SQL Query Builder επιτρέπει την δημιουργία, επεξεργασία και εκτέλεση προτάσεων SQL χρησιμοποιώντας το γραφικό περιβάλλον του SQL Query Builder που προσφέρει πρόσβαση στο σχήμα και στα αντικείμενα της βάσης δεδομένων χωρίς την πληκτρολόγηση κώδικα SQL.
- SQL Execution Plan Framework: Το SQL Execution Plan Framework προσφέρει τα μέσα για την σύλληψη και παρουσίαση πλάνων εκτέλεσης με ένα γενικό τρόπο, επιτρέποντας προσαρμοσμένη υποστήριξη για συγκεκριμένες μηχανές εκτέλεσης SQL.



`

## ΕΠΙΛΟΓΟΣ

Το πρόγραμμά μας μπορεί να συνδεθεί με την βάση δεδομένων ανοικτού κώδικα με ODBC, JDBC, βιβλιοθήκες για .NET και C/C++. Το πρόγραμμα αρχικά θα φορτώσει τις απαραίτητες βιβλιοθήκες, θα δημιουργήσει τα απαραίτητα αντικείμενα για πρόσβαση στην βάση και χειρισμού του αποτελέσματος εκτέλεσης κώδικα SQL, θα εκτελέσει τον κώδικα SQL και θα παρουσιάσει τα αποτελέσματα αν υπάρχουν.

Στο επόμενο κεφάλαιο θα δούμε τα βασικά του κώδικα SQL και της σύνταξής του.



`

## 3. Παρουσίαση της SQL

### ΕΙΣΑΓΩΓΗ

Αίτημα προς την σχεσιακή βάση δεδομένων (ή αίτημα για συντομία) συνιστά, μεταξύ άλλων, μια ερώτηση επί των δεδομένων. Η απάντηση στη συγκεκριμένη ερώτηση προκύπτει σε μορφή (νέας) σχέσης η οποία περιέχει το αποτέλεσμα της επεξεργασίας. Για παράδειγμα θα μπορούσαμε να θέλουμε να βρούμε όλους τους φοιτητές που είναι νεώτεροι των 18 ετών, ή όλους τους φοιτητές που έχουν γραφτεί στο μάθημα Reggae203. Η ειδική γλώσσα αιτημάτων επιτρέπει την διατύπωση ανάλογων αιτημάτων προς την βάση.

Η SQL αποτελεί σήμερα την πλέον δημοφιλή εμπορική γλώσσα αιτημάτων για το σχεσιακό DBMS.

### 3.1 Ιστορία της SQL

Η γλώσσα διατύπωσης δομημένων ερωτημάτων (Structured Query Language – SQL) αποτελεί σήμερα την πλέον δημοφιλή και πλέον διαδεδομένη εμπορική έκδοση γλώσσας για την διαχείριση της σχεσιακής βάσης δεδομένων. Σε μια πρώτη, αρχική μορφή, η SQL αναπτύχθηκε από την IBM, πιο συγκεκριμένα: με τα ερευνητικά προγράμματα της εταιρίας που έφερναν τις κωδικές ονομασίες SEQUEL-XRM και SYSTEM-R, την περίοδο 1974-77. Σχεδόν αμέσως άρχισαν να εμφανίζονται στην αγορά προϊόντα λογισμικού DBMS με SQL και από άλλους κατασκευαστές. Όλα αυτά οδήγησαν στο να αποτελεί σήμερα η SQL πρότυπο για τις σχεσιακές βάσεις δεδομένων. Καθώς οι ανάγκες για την επεξεργασία της πληροφορίας στις βάσεις δεδομένων εξελίσσονται, η SQL προσαρμόζεται και βελτιώνεται ώστε να τις εξυπηρετεί.

### 3.2 Χαρακτηριστικά της SQL

Η SQL περιλαμβάνει έναν αριθμό από επιμέρους χαρακτηριστικά όπως:

- Η γλώσσα ορισμού δεδομένων (Data Definition Language - DDL): Ένα υποσύνολο εντολών SQL οι οποίες διεκπαιρώνουν την δημιουργία, την διαγραφή, και την μετατροπή των δηλώσεων πινάκων και των όψεων. Στους πίνακες, δηλώνονται και περιορισμοί ακεραιότητας, την στιγμή δημιουργίας ενός πίνακα ή αργότερα, με μετατροπή της δήλωσής του. Η DDL παρέχει επίσης την δυνατότητα να καθορίζονται δικαιώματα ή προνόμια για την πρόσβαση σε πίνακες και όψεις.,

Παρά το γεγονός ότι το πρότυπο δεν αναφέρεται σε ευρετήρια, οι εμπορικές του εκδόσεις παρέχουν εντολές για την δημιουργία ή την ακύρωση ευρετηρίων.

- Η γλώσσα διαχείρισης την δεδομένων (Data Manipulation Language - DML): Το συγκεκριμένο υποσύνολο της SQL δίνει την δυνατότητα στον χρήστη να εισάγει, να διαγράφει και να ενημερώνει πλειάδες πινάκων.
- Ενσωματωμένη και δυναμική σύνταξη της SQL: Η συγκεκριμένη δυνατότητα της SQL επιτρέπει την ενσωμάτωση του κώδικα της γλώσσας μέσα σε κώδικα της λεγόμενης γλώσσας υποδοχής (πχ, Cobol). Η δυναμική υφή της SQL καθιστά δυνατή την σύνταξη κώδικα ενός αιτήματος (όπως και την εκτέλεσή του) την στιγμή που διεκπαιρώνεται η επεξεργασία.
- Εναύσματα: Το νέο πρότυπο SQL:1999 περιλαμβάνει την υποστήριξη εναυσμάτων, Έναυσμα ορίζεται ότι αποτελούν ενέργειες οι οποίες δηλώνονται ότι θα εκτελούνται (αυτόματα) από το DBMS κάθε φορά που συμβαίνει οι μεταβολές στην βάση δεδομένων να πληρούν κάποιες προϋποθέσεις.
- Ασφάλεια: Η SQL διαθέτει μηχανισμούς ελέγχου, όσον αφορά στην πρόσβαση των χρηστών σε αντικείμενα της βάσης δεδομένων, όπως οι πίνακες και οι όψεις.
- Διαχείριση των συναλλαγών: Ο χρήστης έχει στην διάθεσή του εντολές με τις οποίες μπορεί να επεμβαίνει και να ελέγχει συγκεκριμένες στιγμές στην λειτουργικότητα και στον τρόπο διεκπαιρέωσης των συναλλάγών από το σύστημα.
- Λειτουργικότητα Πελάτη-Εξυπηρετητή και πρόσβαση σε απομακρυσμένες βάσεις δεδομένων: Οι εντολές αυτές ελέγχουν τον τρόπο με τον οποίο μια εφαρμογή πελάτης μπορεί να συνδέεται σε έναν εξυπηρετητή βάσης δεδομένων SQL, με τον οποίο να συλλειτουργεί και με τον τρόπο αυτό να αποκτά πρόβαση στα δεδομένα μέσα από ένα δίκτυο.

### 3.3 Μορφή αιτήματος SQL

Η βασική δομή του αιτήματος SQL έχει ως εξης:

SELECT [DISTINCT] λίστα πιλογής

FROM λίστα πινάκων

WHERE συνθήκη

Το κάθε ένα αίτημα περιέχει στην σύνταξή του τη συνιστώσα SELECT όπου δηλώνονται οι στήλες του αποτελέσματος, όπως περιλαμβάνει και τη συνιστώσα FROM η οποία δηλώνει ένα καρτεσιανό γινόμενο πινάκων. Με την (προαιρετική) χρήση της συνιστώσας WHERE ορίζονται συνθήκες επιλογής πλειάδων από το καρτεσιανό γινόμενο των πινάκων της FROM. Στο παράδειγμα που ακολουθεί παρουσιάζεται ένα απλό αίτημα αυτής της μορφής.

Να βρεθούν τα ονόματα και οι ηλικίες όλων των ναυτικών:



`

SELECT DISTINCT S.sname, S.age

FROM Sailors S

Στο αποτέλεσμα της επεξεργασίας προκύπτει ένα σύνολο πλειάδων, κάθε μία από τις οποίες είναι ζευγάρι τιμών τύπου <sname, age>. Στην περίπτωση όπου δύο ή περισσότεροι ναυτικοί έχουν το ίδιο όνομα και την ίδια ηλικία, η απάντηση εμφανίζει μόνο ένα ζευγάρι με τις αντίστοιχες τιμές των στηλών sname, age.

Αν παραλείψουμε τον όρο DISTINCT, θα μας προκύψει ένα ξεχωριστό αντίγραφο του ζεύγους τιμών <s, a> για την κάθε μια περίπτωση ναυτικού ο οποίος συμβαίνει να έχει όνομα s και ηλικία a. Κατα συνέπεια το αποτέλεσμα θα προκύψει να έχει την μορφή συνόλου μη διακριτών πλειάδων/τιμών.

Το επόμενο αίτημα ισοδυναμεί με την επίδραση του τελεστή της επιλογής στη σχεσιακή άλγεβρα.

Να βρεθούν όλοι οι ναυτικοί οι οποίοι έχουν τιμή διατίμησης (rating) μεγαλύτερη του 7.

SELECT S.sid, S.name, S.rating, S.age

FROM Sailors AS S

WHERE S.rating > 7

Όπως γίνεται φανερό στα δυο προηγούμενα παραδείγματα η συνιστώσα SELECT υλοποιεί στην πράξη την επεξεργασία της προβολής, ενώ η επεξεργασία την υλοποιεί η συνιστώσα WHERE στην σύνταξη του αιτήματος.

Πιο αναλυτικά η σύνταξη της εντολής SQL έχει ως εξής:

- Η λίστα πινάκων στην συνιστώσα FROM είναι μια λίστα από ονόματα πινάκων. Κάθε ένα από τα τελευταία μπορεί να ακολουθείται από το όνομα μιας μεταβλητής διαστήματος. Η χρήση μεταβλητών διαστήματος είναι ιδιαίτερα χρήσιμη όταν μερικά ονόματα πινάκων αναφέρονται περισσότερες από μια φορές στο ίδιο γινόμενο πινάκων.
- Η λίστα επιλογής στην συνιστώσα SELECT αποτελείται από ονόματα στηλών, ή παραστάσεις οι οποίες περιέχουν ονόματα στηλών. Οι στήλες πρέπει να ανήκουν στη δομή του γινομένου πινάκων της συνιστώσας FROM. Όταν γίνεται χρήση μεταβλητών διαστήματος, του ονόματος της κάθε στήλης προηγείται το όνομα της αντίστοιχης μεταβλητής διαστήματος.
- Η καταλληλότητα στην συνιστώσα WHERE διατυπώνεται με την χρήση των λογικών συνδετικών AND, OR και NOT. Αποτελεί μια σύνταξη Boole με συνδιασμούς συνθηκών τύπου παράσταση op παράσταση, όπου το op συμβολίζει όποιο μέλος από τις συνθήκες τελεστών σύγκρισης: {<, <=, =, <>, >=, >}. Παράσταση μπορεί να αποτελέσει το όνομα της στήλης, μια σταθερά, μια αριθμητική παράσταση, ή μια συμβολοσειρά.



`

- Η χρήση του όρου DISTINCT είναι προαιρετική. Δηλώνει τον περιορισμό οτι ο πίνακας του αποτελέσματος δεν πρέπει να καταχωρεί διπλοεγγραφές, δηλαδή δεν πρέπει να υπάρχουν σε αυτόν δυο αντίγραφα της ίδιας γραμμής/πλειάδας. Όταν παραλείπεται το DISTINCT, προαποφασισμένη λειτουργία αποτελεί η δυνατότητα ύπαρξης διπλοεγγραφών στο αποτέλεσμα.

  Το πρότυπο SQL υποστηρίζει μια πολύ πιο εμπλουτισμένη διατύπωση της λίστας επιλογής από την απλή αναφορά ονομάτων στηλών. Το κάθε ένα από τα μέλη της λίστας επιλογής μπορεί να έχει την μορφή παράσταση AS όνομα στήλης, όπου η παράσταση μπορεί να είναι η όποια αριθμητική παράσταση ή συμβολοσειρά η οποία χρησιμοποιεί ονόματα στηλών (πιθανά με προθέματα ονόματα μεταβλητών διαστημάτων) ή/και σταθερές. Αυτή η μορφή χρησιμοποιείται για να δώσει ένα ψευδώνυμο σε ένα ή περισσότερα μέλη της λίστας επιλογής. Μπορεί επίσης να συμπεριλαμβάνει συναθροίσεις όπως η sum (υλοποιεί την άθροιση) και η count (υλοποιεί την απαρίθμηση).

### 3.4 Τρόπος υπολογισμού του αιτήματος SQL

Οι παραπάνω κανόνες παρά το γεγονός ότι περιγράφουν (με απλό, όχι τυπικό τρόπο) την βασικής έκδοσης σύνταξης ένος αιτήματος SQL, δεν επεξηγούν την σημασία της σύνταξης ενός αιτήματος. Στο αποτέλεσμα της επεξεργασίας προκύπτει μια σχέση (ένα σύνολο μη διακριτών τιμών/πλειάδων στην SQL). Για να γίνει κατανοητό το πως προκύπτει το συγκεκριμένο περιεχόμενο χρησιμοποιείται η παρακάτω ιδεατή στατηγική του υπολογισμού του αιτήματος:

1. Υπολογίζεται το καρτεσιανό γινόμενο των πινάκων οι οποίοι αναφέρονται στη λίστα πινάκων.
2. Εξαιρούνται όλες οι πλειάδες του καρτεσιανού γινομένου οι οποίες δεν ικανοποιούν την συνθήκη καταλληλότητας.
3. Εξαιρούνται όλες οι στήλες που δεν αναφέρονται στην λίστα επιλογής.
4. Εφόσον υπάρχεί όρος DISTINCT, επιλέγεται μόνο μια πλειάδα (εκπρόσωπος) από μια ομάδα ταυτόσημων πλειάδων (διπλοεγγραφών).



`

### 3.5 Τελεστές συνάθροισης

Εκτός από την απλή ανάκτηση δεδομένων, συχνά ζητούμενο αποτελεί η διεκπαιρέωση κάποιων υπολογισμών αριθμητικής επεξεργασίας ή άθροισης (επιμέρους) ποσοτήτων. Η SQL υποστηρίζει την σύνταξη αριθμητικών παραστάσεων. Η SQL υποστηρίζει πέντε τελεστές συνάθροισης, κάθε ένας από τους οποίους μπορεί να ενεργήσει σε μια (οποιαδήποτε) στήλη ενός πίνακα, πχ την στήλη Α:

1. COUNT ([DISTINCT] A): Το πλήθος των (διακριτών) τιμών της στήλης Α.
2. SUM ([DISTINCT] A): Το άθροισμα όλων των (διακριτών) τιμών της στήλης Α.
3. AVG ([DISTINCT] A): Ο μέσος όρος όλων των (διακριτών) τιμών της στήλης Α.
4. MAX(A): Η μέγιστη τιμή η οποία καταχωρείται στην στήλη Α.
5. MIN(A): Η ελάχιστη τιμή η οποία καταχωρείται στην στήλη Α.

### 3.6 Οι τελεστές UNION, INTERSECT, EXCEPT

Η σύνταξη του βασικού αιτήματος SQL που εξετάστηκε, ενισχύεται με τρείς επιπλέον δυνατότητες επεξεργασίας συνόλων. Από την στιγμή που το αποτέλεσμα της επεξεργασίας προκύπτει σε μορφή συνόλου πλειάδων, όχι απαραίτητα διακριτών, είναι λογικό να υποστηρίζονται τελεστές όπως η ένωση, η τομή και η διαφορά δυο συνόλων. Η SQL υποστηρίζει τους τρεις αυτούς τύπους επεξεργασίας με τις ονομασίες UNION, INTERSECT και EXCEPT αντίστοιχα. Μάλιστα υποστηρίζει και επιπλέον τελεστές συνόλων όπως ο IN (ο οποίος ελέγχει την ύπαρξη ενός μέλους σε ένα δοσμένο σύνολο) και τον EXISTS (για τον έλεγχο του κενού συνόλου). Χρήση του NOT πριν την αναφορά των IN και EXISTS δηλώνει μετάπτωση της λογικής τιμής του αποτελέσματος στον αντίθετο.

#### 3.6.1 Οι συνιστώσες GROUP BY και HAVING

Μέχρι τώρα έχουν αναφερθεί περιπτώσεις όπου η επεξεργασία της συνάθροισης επενεργεί στο σύνολο των επιλεγμένων γραμμών ενός πίνακα. Συμβαίνει συχνά η συνάθροιση να χρειάζεται να επενερργεί σε ομάδες πλειάδων ενός πίνακα. Το πλήθος των ομάδων ενός πίνακα δεν είναι γνωστό από τα πριν, εξαρτάται από το στιγμιότυπο του πίνακα και την πληροφορία την οποία αυτό καταχωρεί. Για παράδειγμα έστω το εξής αίτημα:

Να βρεθεί η ηλικία του νεώτερου ναυτικού για κάθε μαι τιμή διατίμησης.



`

Αν τύχαινε να γνωρίζουμε ότι η βάση καταχωρεί τιμές διατίμησης από το 1 ως το 10 θα μπορούσαμε να συντάξουμε 10 αιτήματα του τύπου:

SELECT MIN(S.age)

FROM SAILORS S

WHERE S.rating = i

Όπου το i παίρνει τιμές από 1,2,3,..,10. Φυσικά δεν είναι τόσο εύκολο να συντάξει κανείς δέκα αιτήματα αυτού του τύπου. Ακόμα πιο σημαντικό, μπορεί να μην είναι γνωστες από τα πριν οι τιμές διατίμησης (rating) τις οποίες καταχωρεί η βάση.

Για την αντιμετώπιση αυτού του τύπου αιτημάτων χρειάζεται να κάνουμε χρήση μιας σημαντικής επέκτασης στη βασική σύνταξη αιτήματος SQL: της συνιστώσας GROUP BY. Μάίστα η συγκεκριμένη επέκταση περιλαμβάνει και μια προαιρετική συνιστώσα η οποία ονομάζεται HAVING που χρησιμεύει για να διατυπώνονται συνθήκες καταλληλότητας σε επίπεδο ομάδων. Ο γενικός τύπος σύνταξης αιτήματος SQL ο οποίος περιλαμβάνει τις παραπάνω επεκτάσεις έχει ως εξής:

SELECT [DISTINCT] λίστα επιλογής

FROM λίστα πινάκων

WHERE καταλληλότητα

GROUP BY ομαδοποίηση

HAVING καταλληλότητα ομάδας

Με την χρήση της συνιστώσας GROUP BY το προηγούμενο αίτημα γίνεται:

SELECT S.rating, MIN(S.age)

FROM Sailors S

GROUP BY S.rating

Στο σημείο αυτό να εξετάσουμε μερικά αξιοπρόσεκτα σημεία που αφορούν στις νέες συνιστώσες σύνταξης της εντολής SELECT:

- Τη λίστα επιλογής της SELECT συνιστούν: (1) μια σειρά από ονόματα στηλών και (2) μια σειρά από όρους οι οποίοι έχουν να κάνουν με αποτελέσματα διαδικασιών συνάθροισης (aggop), όπου ο κάθε όρος έχει την μορφή aggop (όνομα στήλης) AS νέο όνομα. Η (προαιρετική) δήλωση νέου ονόματος (AS νέο όνομα) ονοματίζει την στήλη στον πίνακα ο οποίος προκύπτει στο αποτέλεσμα της επεξεργασίας του αιτήματος. Στην θέση του aggop μπορεί να χρησιμοποιηθεί ένας οποιοσδήποτε τελεστής συνάθροισης.



`

Η κάθε μια στήλη στην παραπάνω σειρά (1) μπορεί να αναφέρεται επίσης στη λίστα ομαδοποίησης. Αυτό διότι η κάθε μια γραμμή στο αποτέεσμα του αιτήματος αναφέρεται σε μια ομάδα, δηλαδή ένα σύνολο από πλειάδες οι οποίες έχουν ταυτόσημες τιμές στις στήλες ως προς τις οποίες γίνεται η ομαδοποίηση. Όταν συμβεί μπορεί να αναφέρεται μια στήλη στην σειρά (1) και όχι στην λίστα ομαδοποίησης, δεν είναι φανερό ποιά ακριβώς θα είναι η τιμή της συγκεκριμένης στήλης στη γραμμή του αποτελέσματος.

- Οι παραστάσεις οι οποίες αναφέρονται στην καταλληλότητα ομάδας της σύνταξης HAVING οφείλουν να υπολογίζουν μια και μόνη τιμή για την κάθε μια ομάδα. Εμπειρικά η σύνταξη HAVING υπολογίζει ένα φίλτρο το οποίο με την σειρά του καθορίζει το κατά πόσον η υπό εξέταση ομάδα γραμμών επιτρέπεται να παράγει μια γραμμή στον πίνακα αποτέλεσμα. Κατα συνέπεια, μια στήλη η οποία αναφέρεται στην καταλληλότητα της ομάδας οφείλει επίσης να εμφανίζεται στην λίστα ομαδοποίησης ή να αποτελεί ονοματισμό του αποτελέσματος ενός τελεστή συνάθροισης.
- Στην περίπτωση όπου παραλείπεται η συνιστώσα σύνταξης GROUP BY, το σύνολο των περιεχομένων του πίνακα θεωρείται ότι αποτελεί μια ομάδα.

### 3.7 Εμφωλευμένα αιτήματα

Η δυνατότητα σύνταξης αιτημάτων εμφώλευσης αποτελεί ένα από τα πλεόν δυναμικά χαρακτηριστικά της SQL. Εμφωλευμένο αίτημα ονομάζεται ένα αίτημα η σύνταξη του οποίου ενσωματώνει την σύνταξη ένος άλλου αιτήματος. Το δεύτερο (ενσωματωμένο, εμφωλευμένο) αίτημα ονομάζεται υποαίτημα. Όταν συντάσσεται ένα αίτημα συχνά χρειάζεται να χρησιμοποιηθεί ένας πίνακας-αποτέλεσμα. Ο τελευταίος υπολογίζεται με βάση ένα άλλο αίτημα το οποίο απότελεί υποαίτημα του κυρίως αιτήματος. Συνήθως το υποαίτημα προσάπτεται στην συνιστώσα WHERE του αιτήματος.

### 3.8 Το σχεσιακό μοντέλο

Η κύρια δομή καταχώρισης και αναπαράστασης των δεδομένων στο σχεσιακό μοντέλο είναι η σχέση. Τη σχέση συναποτελούν το σχήμα της σχέσης και το στιγμιότυπο της σχέσης. Το στιγμιότυπο της σχέσης είναι ένας πίνακας, και το σχήμα της σχέσης είναι η περιγραφή των επικεφαλίδων στις στήλες του πίνακα. Το σχήμα καθορίζει το όνομα της σχέσης, το όνομα του κάθε ενός πεδίου (ή στήλης, ή γνωρίσματος), και το πεδίο ορισμού των τιμών για το κάθε ένα πεδίο. Στο σχεσιακό σχήμα, το πεδίο ορισμού προσδιορίζεται με το όνομα πεδίου



`

ορισμού και αντιστοιχίζεται σε ένα σύνολο επιτρεπτών τιμών. Για παράδειγμα, μια βάση δεδομένων η οποία αποθηκεύει πληροφορίες για φοιτητές.

Students(sid:String, name:String, login:String, age:integer, gpa:real)

Η δήλωση αυτή σημαίνει ότι το πεδίο με όνομα sid ορίζεται σε πεδίο ορισμού το οποίο ονομάζεται String.

Το πεδίο ορισμού για το κάθε ένα πεδίο ή στήλη στο στιγμιότυπο μιας σχέσης προσδιορίζεται από το σχήμα της σχέσης. Με αυτήν την έννοια, το σχήμα δυμπεριλαμβάνει στον ορισμό του έναν αριθμό περιορισμών πεδίου ορισμού οι οποίοι είναι σημαντικό να ισχύουν για το κάθε στιγμιότυπο της σχέσης: οι τιμές τις οποίες παίρνει μια στήλη πρέπει να προέρχονται από το σύνολο τιμών του αντίστοιχου πεδίου ορισμού το οποίο δηλώνεται ότι σχετίζεται με την συγκεκριμένη στήλη.

Οι περιορισμοί πεδίου ορισμού είναι τόσο σημαντικοί για το σχεσιακό μοντέλο ώστε στα επόμενα θεωρούνται μόνο στιγμιότυπα σχέσεων τα οποία τους ικανοποιούν. Οπότε στιγμιότυπο σχέσης σημαίνει στιγμιότυπο που ικανοποιεί το σύνολο των περιορισμών του πεδίου ορισμού που ορίζονται στο αντίστοιχο σχήμα σχέσης.

Σχεσιακή βάση δεδομένων ονομάζεται μια συλλογή σχέσεων που έχουν διακριτά ονόματα. Σχήμα σχεσιακής βάσης δεδομένων ονομάζεται η συλλογή των σχημάτων των σχέσεων στην βάση δεδομένων. Στιγμιότυπο της σχεσιακής βάσης αποτελεί μια συλλογή στιγμιότυπων σχέσεων, ένα για το σχήμα κάθε μία σχέσης στο σχήμα της βάσης δεδομένων.

### 3.9 Δημιουργία και τροποποίηση πινάκων

Το πρότυπο SQL χρησιμοποιεί την λέξη πίνακας για να δηλώσει μια σχέση. Το υποσύνολο των εντολών οι οποίες υποστηρίζουν την δημιουργία, την τροποποίηση και την διαγραφή πινάκων ονομάζεται Γλώσσα Ορισμού Δεδομένων (Data Definition Language – DDL)

Η εντολή CREATE TABLE χρησιμοποιείται για την δημιουργία ενός νέου πίνακα. Για την δημιουργία της σχέσης Students μπορεί να γίνει χρήση της εξής εντολής:

```
CREATE TABLE STUDENTS(sid CHAR(20),
                      name CHAR(30),
                      login CHAR(20),
                      age INTEGER,
                      gpa REAL);
```



`

Η εισαγωγή πλειάδων γίνεται με την εντολή INSERT. Η εισαγωγή μιας νέας πλειάδας τιμών στον πίνακα γίνεται ως εξής:

```
INSERT
INTO Students (sid, name, login, age, gpa)
VALUES (53688, 'Smith', 'smith@ee', 18, 3.2);
```

Προεραιτηκά μπορούμε να παραλείψουμε την αναφορά σε ονόματα στηλών στην σύνταξη INTO, υπονοώντας αναφορά στη θέση των πεδίων του πίνακα σε πλήρη αντιστοίχηση με τις τιμές οι οποίες εισάγονται. Παρά την ευκολία γραφής του δεν, το τελευταίο δεν συνιστάται και είναι ένδιξη καλού προγραμματισμού η αναφορά σε κάθε μία στήλη με το όνομά της.

Η διαγραφή των πλειάδων γίνεται με την εντολή DELETE. Για παράδειγμα μπορούμε να διαγράψουμε το σύνολο των πλειάδων οι οποίες καταχωρούν την τιμή ‘Smith’ στην στήλη name του πίνακα Students με την εξής εντολή:

```
DELETE
FROM Students S
WHERE S.name = 'Smith'
```

Η μεταβολή του περιεχομένου στις στήλες εγγραφών οι οποίες ήδη υπάρχουν επιτυγχάνεται με χρήση της εντολής UDPATE. Για παράδειγμα, μία και μόνη εντολή μπορεί να αυξήσει κατά ένα έτος την ηλικία (age) και να ελατώσει κατά ένα το μέσο βαθμό (gpa) του φοιτητή του οποίου ο κωδικός ιδούται με 53688:

```
UPDATE Students S
SET S.age = S.age + 1, s.gpa = S.gpa + 1
WHERE S.sid = 53688
```

Τα παραπάνω παραδείγματα επισημαίνουν δύο σημαντικά σημεία, όσον αφορά στην σύνταξη των εντολών SQL. Η συνιστώσα WHERE είναι αυτή η οποία επενεργεί πρώτη και και καθορίζει το υποσύνολο των πλειάδων οι οποίες πρόκειται να επηρεαστούν. Στην συνέχεια ενεργοποιείται η συνιστώσα SET και καθορίζει το είδος της ενημέρωσης.

### 3.10 Περιορισμοί ακεραιότητας σε πίνακες

Μια βάση δεδομένων είναι χρήσιμη μόνο όσο καταχωρεί χρήσιμη και σωστή πληροφορία. Λογική απαίτηση αποτελεί το DBMS να αποτρέπει την καταχώρηση λανθασμένης πληροφορίας. Περιορισμός ακεραιτότητας (Integrity Constraint – IC) ονομάζεται η συνθήκη η οποία δηλώνεται να ισχύει επάνω σε ένα σχήμα της βάσης δεδομένων και η οποία επιβάλλει περιορισμούς επί των δεδομένων τα



`

οποία επιτρέπεται να καταχωρεί ένα (κάθε) στιγμιότυπο της βάσης. Ένα στιγμιότυπο της βάσης ονομάζεται έγκυρο όταν τα δεδομένα τα οποία καταχωρεί ικανοποιούν το σύνολο των περιορισμών ακεραιότητας οι οποίοι έχουν δηλωθεί να ισχύουν πάνω στο συγκεκριμένο σχήμα της βάσης δεδομένων. Το DBMS λειτουργεί με τέτοιο τρόπο ώστε να επιβάλει διαρκώς την ισχύ των περιορισμών ακεραιότητας, επιτρέποντας να καταχωρούνται μόνο τα έγκυρα στιγμιότυπα της βάσης.

Υπάρχουν πολλοί τύποι περιορισμών ακεραιότητας που μπορούν να δηλωθούν στα πλαίσια του σχεσιακού μοντέλου. Ήδη έχει εξεταστεί ο πρώτος τύπος σε σχέση με το σχήμα μιας σχέσης: τους περιορισμούς πεδίου ορισμού. Στην γενική περίπτωση μπορούν να δηλωθούν και να ισχύουν και άλλοι τύποι περιορισμών. Θα εξεταστούν τύποι περιορισμών ακεραιότητας διαφορετικοί των περιορισμών πεδίου ορισμού του οποίους ο διαχειριστής της βάσης δεδομένων και οι χρήστες έχουν την δυνατότητα να δηλώνουν ώστε να ισχύουν στα πλαίσια ενός σχεσιακού μοντέλου.

### 3.10.1 Περιορισμοι κλειδιού

Θεωρώντας την σχέση Students και τον περιορισμό ότι δεν επιτρέπεται σε δύο φοιτητές να έχουν το ίδιο sid: το συγκεκριμένο αποτελεί περίπτωση περιορισμού κλειδιού. Περιορισμός κλειδιού ονομάζεται η δήλωση ότι υπάρχει κάποιο ελαχιστοποιημένο υποσύνολο του συνόλου των πεδίων μίας σχέσης του οποίου η (συνδυασμένη) τιμή προσδιορίζει μονοσήμαντα την κάθε πλειάδα. Ένα σύνολο πεδίων το οποίο προσδιορίζει την κάθε πλειάδα σύμφωνα με τον περιορισμό κλειδιού λέγεται ότι αποτελεί υποψήφιο κλειδί για την σχέση. Για συντομία το τελευταίο αναφέρεται και ως κλειδί.

Προσέχοντας τον παραπάνω ορισμό του υποψηφίου κλειδιού διαπιστώνεται ότι:

1. Δύο διακριτές πλειάδες οι οποίες ανήκουν στο ίδιο έγκυρο στιγμιότυπο βάσης δεδομένων δεν μπορούν να έχουν ταυτόσημες τιμές σε όλα τα πεδία τα οποία συναποτελούν το υποψήφιο κλειδί.
2. Κανένα υποσύνολο του συνόλου των πεδίων του κλειδιού δεν παρέχει την δυνατότητα μονοσήμαντου προσδιορισμού των πλειάδων στην σχέση.

Στην SQL ένα υποσύνολο του συνόλου των στηλών ενός πίνακα μπορούν να δηλωθούν ότι αποτελούν υποψήφιο κλειδί με χρήση της συνιστώσας UNIQUE στην σύνταξη της εντολής CREATE TABLE. Το πολύ ένα από αυτά τα υποψήφια κλειδιά μπορεί να δηλωθεί ότι αποτελεί το κύριο κλειδί, με χρήση της συνιστώσας PRIMARY KEY. Η SQL δεν απαιτεί να δηλώνεται το κύριο κλειδί πρώτα ως υποψήφιο κλειδί στην συνιστώσα UNIQUE και επίσης δεν υποχρεώνει να



`

χρησιμοποιείται οποιαδήποτε από τις δύο συνιστώσες (UNIQUE και PRIMARY KEY) ως μέρος της σύνταξης της εντολής CREATE TABLE.

Το παράδειγμα της δήλωσης της σχέσης Students επαναδιατυπώνεται τώρα με σύνταξη η οποία ορίζει περιορισμούς κλειδιού:

```
CREATE TABLE STUDENTS(sid CHAR(20),
                      name CHAR(30),
                      login CHAR(20),
                      age INTEGER,
                      gpa REAL,
                      UNIQUE (name, age),
                      CONSTRAINT StudentsKey PRIMARY KEY (sid));
```

Ο παραπάνω ορισμός δηλώνει για κύριο κλειδί το πεδίο sid και το συνδυασμό name και age ως υποψήφιο κλειδί. Στην συγκεκριμένη περίπτωση σύνταξης της CREATE TABLE σημειώνεται ο τρόπος με τον οποίο μπορούμε να ονοματίσουμε έναν περιορισμό: προσθέτοντας την συνιστώσα CONSTRAINT όνομα_του_περιορισμού. Στην περίπτωση που συμβαίνει να παραβιάζεται ο περιορισμός, το μήνυμα λάθους που μας κοινοποιεί αυτόματα το DBMS αναφέρεται στον περιορισμό με το όνομα που του δώσαμε, διευκολύνοντάς μας στον εντοπισμό της πηγής του προβλήματος.

### 3.10.2 Περιορισμοί ξένου κλειδιού

Στο σχεσιακό DBMS συμβαίνει συχνά μα συνδυάζεται η πληροφορία η οποία καταχωρείται σε μια σχέση με πληροφορία η οποία καταχωρείται σε διαφορετική σχέση. Κάθε φορά που συμβαίνει Κάθε φορά που συμβαίνει να ενημερώνεται η η πληροφορία σε μια σχέση χρειάζεται να γίνεται και έλεγχος της πληροφορίας και στην δεύτερη σχέση (συνεζευγμένη) σχέση και πιθανά η ενημέρωση να επεκτείνεται και σε εκείνη. Αυτά όλα γίνονται σε τρόπο ώστε το σύνολο της πληροφορίας που καταχωρεί η βάση δεδομένων να παραμένει σε συνεπή και απόλυτα συγχρονισμένη κατάσταση. Το DBMS εφαρμόζει αυτού του είδους τους αυτοματισμούς οι οποίοι κατωχυρώνουν την ακεραιότητα των δεδομένων, μέσω των δηλωμένων περιορισμών ακεραιότητας. Η πλέον συχνά εμφανιζόμενη περίπτωση περιορισμού ακεραιότητας ανάμεσα σε δύο σχέσεις είναι ο περιορισμός ξένου κλειδιού.

Έστω ότι παράλληλα με την Students, υπάρχει και η σχέση:

Enrolled(sid:String, cid:String, grade:String)

Για να διασφαλιστεί η συμμετοχή μόνο κανονικά εγγεγραμμένων φοιτητών σε κάθε μάθημα, κάθε τιμή που καταχωρείται στο πεδίο sid σε στιγμιότυπο της σχέσης Enrolled πρέπει να είναι ήδη καταχωρημένη στο πεδίο sid μιας πλειάδας στην



`

σχέση Students. Το πεδίο sid της Enrolled λέγεται ότι συνιστά ξένο κλειδί το οποίο αναφέρεται στην σχέση Students. Το ξένο κλειδί στην σχέση που περιέχει την αναφορά χρειάζεται να αντιστοιχίζεται σε ίση τιμή του κύριου κλειδιού στην αναφερόμενη σχέση. Πιο συγκεκριμένα, το ξένο και το κύριο κλειδί πρέπει να απαρτίζονται από τον ίδιο αριθμό στηλών (μάλιστα συμβατών μία προς μία όσον αφορά τον τύπο δεδομένων που καταχωρούν) χωρίς να είναι απαραίτητο οι στήλες αυτές να ονόματίζονται παρόμοια στην δομή των δύο σχέσεων.

Η δημιουργία της σχέσης Enrolled (sid:String, cid:String, grade:String) γίνεται με την εξής εντολή:

```
CREATE TABLE Enrolled (
   sid CHAR(20),
   cid CHAR(20),
   grade CHAR(10),
   PRIMARY KEY (sid, cid),
   FOREIGN KEY (sid) REFERENCES Students)
```

Στην συγκεκριμένη περίπτωση ο περιορισμός του ξένου κλειδιού επιβάλλει σε κάθε μία τιμή του sid της Enrolled να έχει ίση τιμή καταχωρημένη στην Students: ισοδύναμα, το πεδίο sid της Enrolled είναι ξένο κλειδί το οποίο αναφέρεται στην Students. Με την ευκαιρία στην σχέση Enrolled υπάρχει και ο περιορισμός του κύριου κλειδιού: η βάση καταχωρεί για τον/την κάθε φοιτητή/τρια έναν ακριβώς βαθμό για το κάθε μάθημα στο οποίο εγγράφεται/συμμετέχει.

Να σημειωθεί πως στην MySQL οι λέξεις FOREIGN KEY και REFERENCES αναλύονται (parsed) αλλά αγνοούνται από τις μηχανές αποθήκευσης εκτός της InnoDB.

### 3.10.3 Γενικού τύπου περιορισμοί

Οι περιορισμοί επί του πεδίου ορισμού, του κύριου και του ξένου κλειδιού έχουν ουσιαστικό ρόλο και σημασία για το σχεσιακό μοντέλο δεδομένων. Για τον λόγο αυτό, τα διάφορα εμπορικά DBMS δίνουν ιδιαίτερη έμφαση στην υποστήριξή τους. Υπάρχουν όμως και περιπτώσεις όπου προκύπτει η ανάγκη να δηλώνονται περιορισμοί γενικότερου τύπου.

Για παράδειγμα μπορεί να χρειάζεται οι ηλικίες των φοιτητών να βρίσκονται σε μια προκαθορισμένη ακτίνα επιτρεπόμενων τιμών. Δηλώνοντας το σχετικό περιορισμό, το DBMS θα απορρίψει όλες εκείνες τις υποψήφιες (νέες) εγγραφές οι οποίες παραβιάζουν τον συγκεκριμένο κανόνα. Στην πράξη το DBMS αποσοβεί μια κατηγορία λαθών σε επίπεδο εισαγωγής νέων δεδομένων.



`

Τα σύγχρονα συστήματα βάσεων δεδομένων υποστηρίζουν αυτούς τους γενικού τύπου περιορισμούς μέσω των περιορισμών σε επίπεδο πίνακα και των διασφαλίσεων. Οι περιορισμοί σε επίπεδο πίνακα αναφέρονται σε ένα μόνο πίνακα και ενεργοποιούνται κάθε φορά που ενημερώνεται αυτός ο πίνακας. Αντίθετα οι διασφαλίσεις αναφέρονται σε έναν αριθμό από πίνακες και ενεργοποιούνται κάθε φορά που ενημερώνεται οποιοσδήποτε από από αυτούς.

Μπορούμε να ορίσουμε σύνθετους περιορισμούς σε έναν μόνο πίνακα χρησιμοποιώντας τους περιορισμούς πίνακα οι οποίοι έχουν την μορφή CHECK παράσταση-συνθήκης. Για παράδειγμα για να διασφαλιστεί ότι η ηλικία του φοιτητή θα είναι πάνω από ακέραιος αριθμός πάνω από 16 θα χρησιμοποιήσουμε την σύνταξη:

```
CREATE TABLE STUDENTS(sid CHAR(20),
                      name CHAR(30),
                      login CHAR(20),
                      age INTEGER,
                      gpa REAL,
                      UNIQUE (name, age),
                      CONSTRAINT StudentsKey PRIMARY KEY (sid),
                      CHECK (age > 16));
```

Να σημειωθεί πως στην MySQL οι περιορισμοί CHECK αναλύονται (parsed) αλλά αγνοούνται από όλες τις μηχανές αποθήκευσης.

### 3.11 Εναύσματα

Μια διαδικασία την οποία δηλώνει ο διαχειριστής της βάσης δεδομένων και η οποία ενεργοποιείται αυτόματα όποτε συμβαίνουν μεταβολές ορισμένου τύπου στα δεδομένα λέγεται έναυσμα.

Μια βάση δεδομένων η δήλωση της οποίας σύνδυάζεται με δηλώσεις εναυσμάτων ονομάζεται ενεργή βάση δεδομένων.

Στην περιγραφή ενός εναύσματος υπάρχουν τρία διακριτά μέρη:

1. Γεγονός: Μια μεταβολή στην βάση δεδομένων η οποία ενεργοποιεί το έναυσμα.
2. Συνθήκη: Ένα αίτημα ή ένας έλεγχος που ενεργοποιείται με την ενεργοποίηση του εναύσματος.
3. Ενέργεια: Μια διαδικασία η οποία εκτελείται όταν ισχύει η συνθήκη και ενεργοποιείται το έναυσμα.

Το έναυσμα μπορούν να ενεργοποιήσουν ενέργειες όπως η εισαγωγή, η διαγραφή και η ενημέρωση δεδομένων, ανεξέρτητα από τον χρήστη ή την εφαρμογή που πραγματοποιεί την συγκεκριμένη ενέργεια.



`

Το ρεπερτόριο των ενεργειών ενός εναύσματος περιλαμβάνει τον έλεγχο του αποτελέσμετος σε σχέση με την λογική συνθήκη, η αναφορά στις παλιές και νέες τιμές των πλειάδων που τροποποιούνται διαμέσου της πρότασης του εναύσματος.

Σημαντικό χαρακτηριστικό αποτελεί η χρονική στιγμή κατά την οποία εφυπνίζεται η ενέργεια ενός εναύσματος, σε σχέση με την πρόταση που την ενεργοποιεί. Σε σχέση με το τι ακριβώς θέλουμε να κάνει το έναυσμα, μπορεί να επιθυμούμε η ενέργειά του να αφυπνίζεται πριν συμβούν οι μεταβολές στο περιεχόμενο του πίνακα, ή εκ των υστέρων.

Η σύνταξη ενός εναύσματος στην MySQL είναι:

**CREATE TRIGGER** όνομα_εναύσματος χρόνος_εναύσματος
γεγονός_εναύσματος
**ON** όνομα_πίνακα **FOR EACH ROW** ενέργεια_εναύσματος

Ο χρόνος εναύσματος είναι η χρονική στιγμή που θα ενεργοποιηθεί το έναυσμα. Μπορεί να είναι BEFORE ή AFTER που δείχνει ότι το έναυσμα ενεργοποιείται πριν ή μετά συμβούν μεταβολές.

Το γεγονός εναύσματος δείχνει το είδος της πρότασης που ενεργοποιεί το έναυσμα. Μπορεί να είναι ένα από τα εξής:

- INSERT: Το έναυσμα ενεργοποιείται όποτε μία νέα γραμμή εισάγεται στον πίνακα.
- UPDATE: Το έναυσμα ενεργοποιείται όποτε μια γραμμή του πίνακα μεταβάλεται.
- DELETE: Το έναυσμα ενεργοποιείται όποτε μια γραμμή διαγράφεται από τον πίνακα.

Το γεγονός εναύσματος δεν αναπαριστά μια ακριβή πρόταση SQL αλλά τον τύπο των λειτουργιών στον πίνακα.

Το όνομα πίνακα είναι το όνομα του πίνακα στον οποίο θα δηλωθεί το έναυσμα και ότι αλλαγές πραγματοποιούνται στον πίνακα που ικανοποιείται η συνθήκη για το γεγονός εναύσματος.

Η ενέργεια εναύσματος είναι η πρόταση που θα εκτελεστεί όταν θα ενεργοποιηθεί το έναυσμα. Για εκτέλεση πολλών προτάσεων εσωκλείονται στην πρόταση BEGIN … END.

Για την αναφορά σε στήλες πίνακα χρησιμοποιούνται τα ψευδώνυμα OLD και NEW, το πρώτο για την υπάρχουσας γραμμή πριν αυτή μεταβληθεί ή διαγραφτεί και το δεύτερο για την νέα γραμμή που εισάγεται ή μια υπάρχουσα γραμμή αφού μεταβληθεί.

Η σύνταξη της δημιουργίας εναύσματος στην PostgreSQL είναι όμοια με την διαφορά ότι την θέση της ενέργειας εναύσματος παίρνει η κλήση μιας συνάρτησης ορισμένης από τον χρήστη.



`

### 3.12 Συναρτήσεις

Η σύνταξη δημιουργίας συνάρτησης στην MySQL είναι:

**CREATE [AGGREGATE] FUNCTION** όνομα_συνάρτησης (παράμετροι)
**RETURNS** τύπος επιστροφής
σώμα συνάρτησης
**RETURN** αντικείμενο επιστροφής

Όπου το όνομα της συνάρτησης είναι αυτό με το οποίο θα την καλούμε, οι παράμετροι μπορεί να είναι ένας ή περισσότεροι και δηλώνονται όνομα_μεταβλητής τύπος_μεταβλητής. Ο τύπος επιστροφής αναπαριστά τον τύπο του αντικειμένου που επιστρέφεται και παίρνει μια τιμή όπως STRING, INTEGER, REAL, DECIMAL και άλλα. Το σώμα της συνάρτησης περιέχει τον κώδικα της συνάρτησης και το αντικείμενο επιστροφής είναι η μεταβλητή η οποία θα επιστραφεί στο σημείο που κλήθηκε η συνάρτηση.

Στην PostgreSQL η σύνταξη την δημιουργίας συνάρτησης είναι:

**CREATE FUNCTION** όνομα_συνάρτησης **(**παράμετροι**)**
**RETURNS** τύπος_επιστροφής
**AS**
$$
σώμα_συνάρτησης
$$
**LANGUAGE SQL;**

Η διαφορά με την MySQL είναι ότι το σώμα της συνάρτησης ξεκινάει από την λέξη AS ακολουθούμενη από τα σύμβολα «$$» και τελειώνει με αυτά. Να σημειωθεί πως στην θέση των συμβόλων «$$» μπορεί να χρησιμοποιηθεί οποιοδήποτε το οποίο δεν έχει κάποια σημασία για την PostgreSQL. Η συνάρτηση τελειώνει με την δήλωση της γλώσσας που χρησιμοποιήθηκε για την δημιουργία της συνάρτησης, δηλαδή την SQL.



`

## ΕΠΙΛΟΓΟΣ

Είδαμε τα βασικά στοιχεία της σύνταξης SQL, η οποία είναι:

SELECT [λίστα επιλογής]

FROM [πίνακες]

WHERE [συνθήκες]

Στην πιο απλή της μορφή, για την παρουσίαση στοιχείων ενός ή περισσότερων πινάκων. Πιο εμπλουτισμένα αιτήματα μπορεί να έχουν ψευδώνυμα, να καλούν συναρτήσεις μέτρησης και άθροισης, όπως η count() και η sum(), καθώς και να ταξινομούν το αποτέλεσμα με βάση κάποια στήλη ενός πίνακα, έχοντας ή όχι κάποια συνθήκη. Επίσης υπάρχει η δυνατότητα για δημιουργία, επεξεργασία και διαγραφή πινάκων. Οι πίνακες μπορούν να περιέχουν περιορισμούς δίνοντας την δυνατότητα τα δεδομένα που εμπεριέχονται σε αυτά να είναι σωστά.



`

# 4. Παρουσίαση των βιβλιοθηκών γραφικών της Java

## ΕΙΣΑΓΩΓΗ

Σε αυτό το κεφάλαιο θα παρουσιαστεί η βιβλιοθήκη γραφικών της Java: Swing.

Θα παρουσιαστεί η δομή της βιβλιοθήκης Swing και τα μέρη-αντικείμενα που την απαρτίζουν.

## 4.1 Επισκόπηση της αρχιτεκτονικής Swing

### 4.1.1 Στόχοι σχεδίασης

Ο γενικός σκοπός του έργου Swing είναι να αναπτύξει ένα σύνολο από επεκτάσιμα συστατικά γραφικών διεπαφών χρήστη (Graphican User Interface – GUI) ώστε να δώσει την δυνατότητα στους προγραμματιστές να αναπτύσουν πιο γρήγορα ισχυρά front end για εφαρμογές.

Γι’ αυτό το σκοπό τέθηκε ένα σύνολο από στόχους σχεδίασης που οδήγησε στην τελική αρχιτεκτονική. Αυτοί οι κανόνες ορίζουν ότι η Swing θα:

1. Είναι πλήρως υλοποιημένη σε Java για συνοχή σε πολλαπλές πλατφόρμες και ευκολότερη συντήρηση.
2. Προσφέρει ένα API ικανό να υποστήρίζει πολλά look-and-feel ώστε οι προγραμματιστές και χρήστες να μην περιορίζονται σε ένα look-and-feel.
3. Προσφέρει την δύναμη του προγραμματισμού βασισμένο στο μοντέλο (model-driven) χωρίς να το απαιτεί στο API υψηλότερου επιπέδου.
4. Τηρεί τους σχεδιαστικούς κανόνες του JavaBeans ώστε να εξασφαλίζει ότι τα στοιχεία θα συμπεριφέρονται σωστά σε IDE και εργαλεία προγραμματισμού.
5. Προσφάρει συμβατότητα με τα API του AWT όπου υπάρχει επικάλυψη για την χρήση της βάσης γνώσεων του AWT και την ευκολία φορητότητας.



`

### 4.1.2 Model-View-Controller

Η αρχιτεκτονική της Swing βασίζεται στο σχέδιο Model-View-Controller (MVC). Η αρχιτεκτονική MVC σημαίνει ότι μια εφαρμογή με γραφικό περιβάλλον χωρίζεται σε τρία ξεχωριστά μέρη:

- Το Model που αντιπροσωπεύει τα δεδομένα της εφαρμογής.
- Το View που είναι η οπτική παρουσίαση αυτών των δεδομένων.
- Το Controller που λαμβάνει την είσοδο από τον χρήστη στο View και τα μεταφράζει σε αλλαγές στο Model.

## 4.2 Ιεραρχία κλάσεων

### 4.2.1 Object

Η κλάση Object (java.lang.Object) είναι η ρίζα της ιεραρχίας των κλάσεων. Κάθε κλάση έχει την Object ως υπερκλάση. Όλα τα αντικείμενα, όπως και οι πίνακες, εφαρμόζουν τις μεθόδους αυτής της κλάσης.

### 4.2.2 Component

Επόμενο στην ιεραρχία κλάσεων των συστατικών γραφικών της Java είναι η Component , η οποία βρίσκεται στο πακέτο AWT (Abstract Window Toolkit), η οποία κληρονομεί την Object.

Ένα Component είναι ένα αντικείμενο που έχει γραφική αναπαράσταση που μπορεί να εμφανιστεί στην οθόνη και μπορεί να αλληλεπιδρά με τον χρήστη. Παραδείγματα από Components είναι τα buttons, checkboxes, scrollbars μιας τυπικής διεπιφάνειας χρήστη.

Η κλάση Component είναι μια αφηρημένη υπερκλάση από τα συστατικά του Abstract Window Toolkit. Ο δομητής της κλάσης είναι ο Component() που δημιουργεί ένα νέο αντικείμενο Component. Η κλάση Component μπορεί επίσης να επεκταθεί για την δημιουργία ενός ελαφριού συστατικού. Ένα ελαφρύ συστατικό είναι ένα συστατικό που δεν σχετίζεται με κανένα παράθυρο. Ένα ελαφρύ συστατικό πρέπει να φιλοξενηθεί από ένα γηγενές (native) container, κάπου υψηλότερα στην ιεραρχία των συστατικών.



`

Οι μέθοδοι της κλάσης αυτής συμπεριλαμβάνουν τις:

- void add(PopupMenu popup) η οποία προσθέτει ένα popup μενού στο component
- void addComponentListener(ComponentListener l) η οποία προσθέτει έναν ακροατή συστατικού ώστε να λαμβάνει γεγονότα συστατικών από αυτό το συστατικό
- void addFocusListener(FocusListener l) που προσθέτει το συγκεκριμένο ακροατή ώστε να λαμβάνει γεγονότα εστίασης από αυτό το συστατικό όταν αυτό λαμβάνει εστιάση εισόδου
- void addKeyListener(KeyListener l) η οποία προσθέτει τον συγκεκριμένο ακροατή ώστε να λαμβάνει γεγονότα πληκτρολογίου από αυτό το συστατικό

### 4.2.3 Container

Η κλάση Container επεκτείνει την λειτουργία της Component. Ένα αντικείμενο Container της Abstract Window Toolkit (AWT) είναι ένα συστατικό που μπορεί να περιέχει άλλα συστατικά AWT.

Τα στοιχεία που προστίθενται στο Container παρακολουθούνται σε μια λίστα. Η ταξινόμηση της λίστας καθορίζει το στοίβαγμα από μπρος προς τα πίσω μέσα στο Container. Αν δεν έχει καθοριστεί δείκτης στην προσθήκη συστατικού στο container τότε προστίθεται στο τέλος της λίστας.

Ο δομητής της κλάσης Container είναι ο Container(). Άλλες μέθοδοι της κλάσης αυτής είναι:

- Component add(Component comp), η οποία προσθέτει το συγκεκριμένο συστατικό στο τέλος αυτού του container
- Component add(Component comp, int index), η οποία προσθέτει το δοσμένο συστατικό στην συγκεκριμένη θέση
- Component getComponent(int n) που επιστρέφει το n-οστό συστατικό που εμπεριέχεται στο container
- Component getComponentAt(int x, int y), επιστρέφει το συστατικό που βρίσκεται στην θέση x, y στο container
- Int getComponentCount(), που επιστρέφει τον αριθμό των συστατικών που εμπεριέχονται στο container



`

### 4.2.4 JComponent

Η JComponent ανήκει στο πακέτο Swing και είναι η βασική αφηρημένη κλάση για όλα τα συστατικά της Swing εκτός από τα ανωτάτου επιπέδου container. Για να χρησιμοποιήσει κανείς ένα συστατικό που κληρονομεί την JComponent πρέπει να τοποθετήσει το συστατικό σε μια περιορισμένη ιεραρχία της οποίας η ρίζα είναι ένα ανωτάτου επιπέδου container της Swing. Τα ανωτάτου επιπέδου container της Swing – όπως JFrame, JDialog και JApplet – είναι εξειδικευμένα συστατικά που προδίδουν έναν χώρο για άλλα συστατικά της Swing να σχεδιαστούν.

Η κλάση της JComponent προσφέρει:

- Την βασική κλάση και για τα πρότυπα αλλά και συστατικά κατασκευασμένα από τον προγραμματιστή που χρησιμοποιούν την αρχιτεκτονική Swing.
- Ένα pluggable look and feel που μπορεί να καθοριστεί από τον προγραμματιστή ή (προαιρετικά) να επιλεχθεί από τον χρήστη την ώρα που τρέχει το πρόγραμμα. Το look and feel για κάθε συστατικό προσφέρεται από ένα UI delegate – ένα αντικείμενο που κληροδοτεί την ComponentUI ( javax.swing.plaf.ComponentUI).
- Ολοκληρωμένος χειρισμός πληκτρολόγησης.
- Υποστήριξη για tooltips – μικρές περιγραφές που αναδύονται όταν ο κέρσορας βρίσκεται πάνω από ένα συστατικό.
- Υποστήριξη για προσβασιμότητα. Η JComponent περιέχει όλες τις μεθόδους της διεπαφής Accessible, αλλά στην ουσία δεν υλοποιεί την διεπαφή. Αυτή είναι ευθύνη των κάθε κλάσεων που επεκτείνουν την Jcomponent.
- Υποστήριξη ιδιοτήτων συγκεκριμένων για κάθε συστατικό. Με την μέθοδο putClientProperty(java.lang.Object, java.lang.Object) και getClientProperty(java.lang.Object), μπορεί κανείς να συσχετίσει ζεύγη ονομάτων-αντικειμένων με κάθε αντικείμενο που κληροδοτεί την JComponent.
- Μια υποδομή για σχεδίαση που περιλαμβάνει διπλό buffering και υποστήριξη για όρια.

Η Jcomponent και οι υποκλάσεις της τεκμηριώνουν προκαθορισμένες τιμές για συγκεκριμένες ιδιότητες. Για παράδειγμα, η JTable τεκμηριώνει το προκαθορισμένο ύψος των γραμμών να είναι 16. Κάθε υποκλάση της JComponent που έχει ένα ComponentUI θα δημιουργήσει το ComponentUI ως μέρος του δομητή της. Για να προσφέρει ένα ιδιαίτερο look and feel κάθε ComponentUI μπορεί να καθορίσει ιδιότητες πίσω στο JComponent που το δημιούργησε.

Ο δομητής της JComponent είναι ο JComponent(). Άλλες μέθοδοι για την κλάση αυτή είναι:



`

- void addAncestorListener(AncestorListener listener), η οποία καταγράφει έναν ακροατή έτσι ώστε να λαμβάνει AncestorEvents όταν αυτή ή κάποιος από τους απογόνους της μετακινείται ή γίνεται ορατός ή αόρατος.
- void setBackground(Color bg) η οποία καθορίζει το χρώμα του παρασκηνίου για το συγκεκριμένο συστατικό. Το χρώμα παρασκηνίου χρησιμοποιείται μόνο όταν το αντικείμενο είναι αδιαφανές, και μόνο από υποκλάσεις των υλοποιήσεων των JComponent ή ComponentUI. Άμεσες υποκλάσεις του Jcomponent πρέπει να υπερφορτώσουν την paintComponent για να χρησιμοποιήσουν αυτήν την ιδιότητα.
- void setEnabled(boolean enabled) η οποία καθορίζει ένα αυτό το συστατικό θα είναι ενεργοποιημένο ή όχι. Ένα συστατικό που είναι ενεργό μπορεί να αντιδρά σε είσοδο από τον χρήστη, ενώ ένα που έιναι ανενεργό δεν μπορεί να αντιδράσει σε είσοδο από τον χρήστη. Μερικά συστατικά μπορεί να αλλάξουν την γραφική τους αναπαράσταση όταν απενεργοποιηθούν ούτων ώστε να πληροφορήσουν τον χρήστη ότι δεν μπορούν να λάβουν είσοδο.

### 4.3 Συστατικά γραφικών της Java

Σε αυτήν την ενότητα θα παρουσιαστούν οι πιο κοινές κλάσεις συστατικών της Java Swing. Θα αναφερθούν οι δομητές και κάποιες μέθοδοι για καθεμία από τις κλάσεις αυτές και θα παρουσιαστεί σαν παράδειγμα ο τρόπος δημιουργίας του κάθε συστατικού με κώδικα. Για λόγους μη επανάληψης τα κοινά κομμάτια κώδικα δεν θα επαναλαμβάνονται, όπως η μέθοδος main, τα imports και άλλα, παρά μόνο ο κώδικας δημιουργίας του συστατικού.

Τα imports για μια εφαρμογή που χρησιμοποιεί της βιβλιοθήκη γραφικών της Java, Swing είναι:

```
import javax.swing.*;
```

#### 4.3.1 JFrame

Ένα Frame είναι ένα παράθυρο πρώτου επιπέδου με ένα τίτλο και όρια. Το μέγεθος του frame περιλαμβάνει όποια περιοχή έχει οριστεί για τα όρια. Ένα πλαίσιο (frame), υλοποιημένο ως ένα αντικείμενο της κλάσης JFrame, είναι ένα παράθυρο το οποίο έχει διακόσμηση όπως όρια, έναν τίτλο και υποστηρίζει συστατικά πλήκτρων που κλείνουν ή μικραίνουν το παράθυρο. Εφαρμογές με γραφικό περιβάλλον χρήστη συνήθως περιλαμβάνουν τουλάχιστον ένα frame.

Οι δομητές αυτής της κλάσης είναι:

- JFrame() ο οποίος δημιουργεί ένα JFrame το οποίο αρχικά είναι διαφανές

- JFrame(GraphicsConfiguration gc) ο οποίος δημιουργεί ένα JFrame στο καθορισμένο GraphicsConfiguration μιας συσκευής οθόνης και με κενό τίτλο
- JFrame(String title) ο οποίος δημιουργεί ένα JFrame, αρχικά διαφανές με το καθορισμένο τίτλο
- JFrame(String title, GraphicsConfiguration gc) ο οποίος δημιουργεί ένα JFrame με τον καθορισμένο τίτλο και το καθορισμένο GraphicsConfiguration μίας συσκευής οθόνης

Άλλες μέθοδοι αυτής της κλάσης είναι:

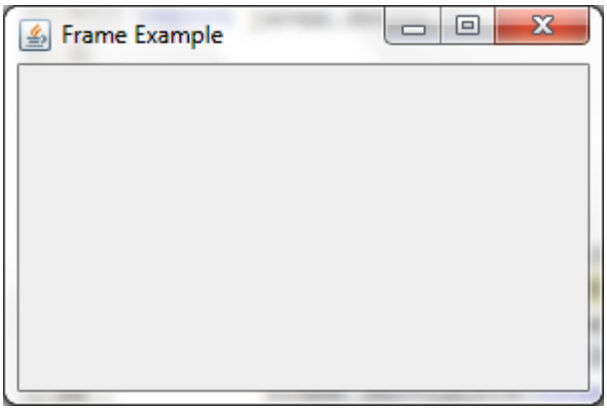
- setIconImage(Image image) η οποία καθορίζει την εικόνα που θα παρουσιαστεί ως η εικόνα για αυτό το παράθυρο
- void setDefaultCloseOperation(int operation) η οποία καθορίζει την λειτουργία που θα γίνει από προεπιλογή όταν ο χρήστης εκκινεί το κλείσιμο αυτού του frame. Οι διάφορες επιλογές είναι DO_NOTHING_ON_CLOSE, η οποία κάνει το frame να μην κάνει τίποτα, και απαιτεί από το πρόγραμμα να χειριστεί την διαδικασία κλεισίματος της εφαρμογής, HIDE_ON_CLOSE το οποίο αυτόματα κρύβει το frame, DISPOSE_ON_CLOSE το οποίο αυτόματα κρύβει και διαθέτει το frame και EXIT_ON_CLOSE που κλείνει την εφαρμογή χρησιμοποιώντας την μέθοδο System exit.
- Component add(Component comp) η οποία κληρονομείται από την κλάση Container και προσθέτει το καθορισμένο συστατικό στο τέλος αυτού του πλαισίου
- Container getContentPane() η οποία επιστρέφει το αντικείμενο contentPane για αυτό το frame
- void pack() η οποία κληρονομείται από την κλάση java.awt.Window και αναγκάζει το παράθυρο να έχει μέγεθος τέτοιο ώστε να χωράει τα συστατικά που περιέχει

Για παράδειγμα ο κώδικας δημιουργίας ενός JFrame με τίτλο “Frame Example”, το οποίο περιέχει μια επιγραφή “Hello World” είναι

```
JFrame frame = new JFrame("Παράδειγμα Frame");
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.setSize(640, 480);
frame.setVisible(true);
```



`




Εικόνα 22. Παράδειγμα δημιουργίας ένος JFrame

### 4.3.2 JLabel

Το αντικείμενο JLabel αναπαριστά μια περιοχή εμφάνισης μιας μικρής συμβολοσειράς κειμένου ή μιας εικόνας ή και τα δύο. Ένα JLabel δεν αντιδρά σε γεγονότα εισόδου. Μπορεί κανείς να προσδιορίσει που στοιχίζονται τα περιεχόμενα του JLabel μέσα στην περιοχή εμφάνισής του, ορίζοντας την οριζόντια και κάθετη στοιχιση. Από προεπιλογή τα JLabel είναι οριζόντια στοιχισμένα στο κέντρο της περιοχής εμφάνισης. Τα JLabel μόνο που περιέχουν μόνο κείμενο στοιχίζονται με βάση την κύρια γωνία, από προεπιλογή, ενώ αυτά που περιέχουν μόνο εικόνα στοιχίζονται οριζόντια στο κέντρο, από προεπιλογή.

Μπορεί κανείς να ορίσει την θέση του κειμένου σε σχέση με την εικόνα. Από προεπιλογή, το κείμενο βρίκεται στο τελευταίο άκρο της εικόνας, με το κείμενο και την εικόνα να έιναι στοιχισμένα κάθετα.

Οι δομητές της κλάσης αυτής είναι:

- JLabel(), ο οποίος δημιουργεί ένα αντικείμενο JLabel χωρίς εικόνα και με άδειο String για τον τίτλο του.
- JLabel(Icon image) ο οποίος δημιουργεί ένα αντικείμενο JLabel με την συγκεκριμένη εικόνα.
- JLabel(Icon image, int horizontalAlignment), ο οποίος δημιουργεί ένα αντικείμενο JLabel με την συγκεκριμένη εικόνα και με οριζόντια ευθυγράμμιση.
- JLabel(String text), ο οποίος δημιουργεί ένα JLabel με το συγκεκριμένο κείμενο.
- JLabel(String text, Icon icon, int horizontalAlignment), ο οποίος δημιουργεί ένα JLabel με το συγκεκριμένο κείμενο, εικόνα και με οριζόντια ευθυγράμμιση/
- JLabel(String text, int horizontalAlignment), που δημιουργεί ένα JLabel με το συγκεκριμένο κείμενο, και με οριζόντια ευθυγράμμιση.

Άλλες μέθοδοι για την κλάση αυτή είναι:



`

- Icon getIcon() η οποία επιστρέφει την γραφική εικόνα (ανάγλυφο, εικονίδιο) που εμφανίζει το JLabel.
- String getText(), η οποία επιστρέφει το κείμενο που παρουσιάζει το JLabel
- public void setIcon(Icon icon) η οποία καθορίζει το εικονίδιο που αυτό θα παρουσιάσει. Αν η τιμή του εικονιδίου είναι null τότε τίποτα δεν θα παρουσιαστεί.
- public void setText(String text) η οποία καθορίζει μια γραμμή κειμένου που θα παρουσιάσει αυτό το συστατικό. Αν η τιμή του συστατικού είναι Null τότε τίποτα δεν θα παρουσιαστεί.

Για παράδειγμα ο κώδικας δημιουργίας ενός JLabel το οποίο βρίσκεται σε ένα JFrame και περιέχει το κείμενο “This is a JLabel” με αριστερή στοίχιση είναι:

```
JFrame frame = new JFrame("Παράδειγμα JLabel");
JLabel label = new JLabel("Αυτό είναι ένα JLabel", JLabel.LEFT);
frame.add(label);
frame.pack();
frame.setVisible(true);
```

### 4.3.3 JButton

Το JButton είναι μια υλοποίηση ενός πλήκτρου πίεσης.

Τα JButtons μπορούν να ρυθμιστούν και σε ένα βαθμό να ελεγχθούν, με Actions. Χρησιμοποιώντας ένα Action με ένα JButton έχει πολλά πλεονεκτήματα πέρα από το να ρυθμίζεται άμεσα το πλήκτρο.

Οι δομητές αυτής της κλάσης είναι:

- JButton() που δημιουργεί ένα πλήκτρο χωρίς κείμενο ή εικονίδιο
- JButton(Action a) ο οποίος δημιουργεί ένα πλήκτρο του οποίου οι ιδιότητες λαμβάνονται από το Action που παρέχεται
- JButton(Icon icon) ο οποίος δημιουργεί ένα πλήκτρο με εικονίδιο
- JButton(String text) ο οποίος δημιουργεί ένα πλήκτρο με κείμενο
- JButton(String text, Icon icon) ο οποίος δημιουργεί ένα πλήκτρο με το αρχικό κείμενο και εικονίδιο



`

Άλλες μέθοδοι για την κλάση JButton είναι:

- boolean isDefaultButton() η οποία επιστρέφει την τιμή της ιδιότητας defaultButton, η οποία αν είναι true σημαίνει ότι το συγκεκριμένο πλήκτρο είναι το τρέχον προεπιλεγμενο για το container του. Τα περισσότερα look and feel καθιστούν το προεπιλεγμένο πλήκτρο διαφορετικά, και μπορεί να προσφέρουν δεσμεύσεις για την πρόσβαση στο προεπιλεγμένο πλήκτρο
- void setEnabled(boolean b) η οποία ενεργοποιεί ή απενεργοποιεί το κουμπί ανάλογα την τιμή που καθορίζεται. Αυτή η μέθοδος κληρονομείται από την κλάση AbstractButton
- void setText(String text) η οποία κληρονομείται από την κλαση AbstractButton και καθορίζει το κείμενο πάνω στο πλήκτρο
- boolean isSelected() η οποία επίσης κληρονομείται από την AbstractButton και επιστρέφει την κατάσταση του πλήκτρου. True αν είναι επιλεγμένο ή false αν δεν είναι
- void addActionListener(ActionListener l) η οποία κληρονομείται από την AbstractButton και προσθέτει έναν ActionListener. Με την προσθήκη ακροατή γεγονότων, όταν ο χρήστης πατάει το πλήκτρο εκτελείται ο κώδικας που έχει καθοριστεί για το γεγονός.

Στο παράδειγμα θα δημιουργηθεί ένα πλήκτρο που βρίσκεται σε ένα JFrame με αρχικό κείμενο «Μη πατημένο» το οποίο θα αλλάζει σε «Πατημένο» μόλις ο χρήστης πατήσει το πλήκτρο. Η διαδικασία για την δημιουργία αυτού του παραδείγματος απαιτεί αρχικά την εισαγωγή των βιβλιοθηκών που έχουν τις απαραίτητες κλάσεις για την δημιουργία και τον χειρισμό των γεγονότων

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
```

Στην συνέχεια, στην δήλωση της κλάσης υλοποιούμε την διεπαφή ActionListener, τον ακροατή γεγονότων

```
public class JButtonExample implements ActionListener {
```

και δηλώνουμε το JButton:

```
JButton button;
```

Έπειτα, στον δομητή της κλάσης δημιουργούμε και εισάγουμε ιδιότητες στα συστατικά JFrame και JButton:

```
public  JButtonExample() {
JFrame frame = new JFrame("Παράδειγμα JFrame");
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

button = new JButton("Μη πατημένο");
button.setSize(80, 20);
button.addActionListener(this);
```



`

```
frame.add(button);
frame.pack();
frame.setVisible(true);
}
```

Στην συνέχεια υπερφορτώνουμε την μέθοδο actionPerformed της διεπαφής ActionListener την οποία υλοποιεί αυτή η κλάση. Αυτή η μέθοδος είναι αυτή που εκτελείται όταν στο συστατικό στο οποίο προστέθηκε το ActionListener συμβεί ένα γεγονός, στην συγκεκριμένη περίπτωση πατηθεί το πλήκτρο.

```
public void actionPerformed(ActionEvent e) {
button.setText("Πατημένο");
}
```

Τέλος στην main μέθοδο δημιουργούμε ένα αντικείμενο της ίδιας κλάσης και το παράθυρο δημιουργείται.

```
public static void main(String[] args) {
JButtonExample jbe = new JButtonExample();
}
```

### 4.3.4 JToggleButton

Το JToggleButton είναι μια υλοποίηση ενός πλήκτρου δύο καταστάσεων.Οι κλάσεις JRadioButton και το JCheckBox είναι υποκλάσεις αυτής της κλάσης.

Δομητές αυτής της κλάσης είναι:

- JButton() που δημιουργεί ένα πλήκτρο χωρίς κείμενο ή εικονίδιο
- JButton(Action a) που δημιουργεί ένα πλήκτρο του οποίοι οι ιδιότητες λαμβάνονται από το Action που δίνεται.
- JButton(Icon icon), που δημιουργεί ένα πλήκτρο με εικονίδιο
- JButton(String text) που δημιουργεί ένα πλήκτρο με κείμενο
- JButton(String text, Icon icon) που δημιουργεί ένα πλήκτρο με αρχικό κείμενο και εικονίδιο

Άλλες μεθοδοι για την κλάση αυτή είναι:

- boolean isDefaultButton() η οποία λαμβάνει την τιμή της ιδιότητας defaultButton, η οποία αν είναι αληθής σημαίνει ότι αυτό το πλήκτρο είναι το τρέχον προεπιλεγμένο για το JrootPane του. Τα περισσότερα look and feel καθοστούν το προεπιλεγμένο πλήκτρο διαφορετικά και μπορεί να προσφέρουν δεσμεύσεις για την πρόσβαση στο προεπιλεγμένο πλήκτρο



`

- protected String paramString() η οποία επιστρέφει ένα String αναπαράστασης του πλήκτρου.

  Για παράδειγμα θα υλοποιηθεί ένα πλήκτρο το οποίο θα παριστάνει έναν διακόπτη. Αρχικά θα αναγράφει το κείμενο «Κλειστό» και με διαδοχικά πατήματα του πλήκτρου το κείμενο θα εναλλάσεται σε «Ανοικτό» και «Κλειστό»

  Για την υλοποίηση του παραδείγματος απαιτείται η υλοποίηση της διεπαφής ActionListener όπως και στο JButton. Ομοίως όπως στην τελευταία κλάση εισάγονται οι κατάλληλες βιβλιοθήκες

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
```

  Στην συνέχεια δηλώνεται η κλάση να υλοποιεί την διεπαφή ActionListener

```
public class JToggleButtonExample implements ActionListener {
```

  και δημιουργούνται το πλήκτρο και οι συμβολοσειρές που θα χρησιμοποιηθούν για το κείμενο πάνω στο πλήκτρο

```
JToggleButton button;
final String open = "Ανοικτό", closed = "Κλειστό";
```

  Έπειτα δημιουργείται ο δομητής της κλάσης ο οποίος περιέχει την δημιουργία του JFrame, JToggleButton και την προσθήκη του δεύτερου στον πρώτο. Επίσης προστίθεται ένας ActionListener στο πλήκτρο.

```
public JToggleButtonExample() {
JFrame frame = new JFrame("Παράδειγμα JToggleButton");
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

button = new JToggleButton(open);
button.addActionListener(this);

frame.add(button);
frame.pack();
frame.setVisible(true);
}
```

  Στην συνέχεια υλοποιείται η μέθοδος που ορίζεται από την διεπαφή ActionListener

```
public void actionPerformed(ActionEvent e) {
if(button.isSelected()) {
button.setText(closed);
}
else {
button.setText(open);
}
}
```

Τέλος δημιουργείται το αντικείμενο της ίδιας κλάσης από την main μέθοδο όπου δημιουργούνται όλα τα συστατικά και σχεδιάζονται στην οθόνη

```
new JToggleButtonExample();
```

### 4.3.5 JRadioButton

Το JRadioButton είναι μια υλοποίηση ενός πλήκτρου radio, ένα αντικείμενο που, όπως και το JToggleButton μπορεί να επιλεχθεί ή να αποεπιλεχθεί και δείχνει την κατάστασή του στον χρήστη. Σε αντίθεση όμως με το JToggleButton, χρησιμοποιείται με το αντικείμενο ButtonGroup για να δημιουργηθεί μια ομάδα από πλήκτρα από τα οποία μόνο ένα μπορεί να επιλεχθεί κάθε φορά.

Οι δομητές αυτής της κλάσης είναι:

- JRadioButton() ο οποίος δημιουργεί ένα αρχικά αποεπιλεγμένο πλήκτρο radio χωρίς καθορισμένο κείμενο
- JRadioButton(Action a) ο οποίος δημιουργεί ένα RadioButton του οποίου οι ιδιότητες λαμβάνονται από το Action που δίνεται
- JRadioButton(Icon icon) που δημιουργεί ένα αρχικά αποεπιλεγμένο RadioButton με την δοσμένη εικόνα αλλά χωρίς κείμενο
- JRadioButton(Icon icon, boolean selected) που δημιουργεί ένα RadioButton με την συγκεκριμένη εικόνα και κατάσταση επιλογής, αλλά χωρίς κείμενο
- JRadioButton(String text) που δημιουργεί ένα αποεπιλεγμένο RadioButton με το δοσμένο κείμενο
- JRadioButton(String text, boolean selected) ο οποίος δημιουργεί ένα RadioButton με το δοσμένο κείμενο και κατάσταση επιλογής
- JRadioButton(String text, Icon icon) που δημιουργεί ένα RadioButton με το δοσμένο κείμενο και εικόνα, το οποίο είναι αρχικά αποεπιλεγμένο
- JRadioButton(String text, Icon icon, boolean selected), ο οποίος δημιουργεί ένα RadioButton με το δοσμένο κείμενο, εικόνα και κατάσταση επιλογής

Το παράδειγμα υλοποίησης εφαρμογής με JRadioButton θα δειχθεί στην ενότητα ButtonGroup.

### 4.3.6 ButtonGroup

Αυτή η κλάση χρησιμοποιείται για να δημιουργήσει ένα πεδίο πολλαπλού αποκλεισμού για ένα σύνολο από πλήκτρα. Δημιουργώντας ένα σύνολο απλο πλήκτρα με το ίδιο αντικείμενο ButtonGroup σημαίνει ότι θέτοντας ένα από αυτά τα



`

πλήκτρα σε κατάσταση “on”, όλα τα υπόλοιπα πλήκτρα στο ίδιο ButtonGroup τίθενται σε κατάσταση “off”.

Ένα ButtonGroup μπορεί να χρησιμοποιηθεί με οποιοδήποτε σύνολο από αντικείμενα τα οποία κληρονομούν την AbstractButton. Τυπικά μια ομάδα από πλήκτρα περιέχει αντικείμενα από JRadioButton, JRadioButtonMenuItem ή JToggleButton. Δεν έχει νόημα να τοποθετηθεί ένα αντικείμενο JButton ή JMenuItem σε ένα ButtonGroup επειδή τα JButton και JMenuItem δεν υλοποιούν την επιλεγμένη κατάσταση.

Αρχικά όλα τα πλήκτρα στο ButtonGroup είναι αποεπιλεγμένα.

Ο δομητής αυτής της κλάσης είναι ο ButtonGroup() που δημιουργεί ένα νέο Buttongroup.

Άλλες μέθοδοι για την κλάση αυτή είναι:

- void add(AbstractButton b) η οποία προσθέτει το πλήκτρο στην ομάδα
- void clearSelection() η οποία καθαρίζει την επιλογή ώστε κανένα από τα πλήκτρα στο ButtonGroup να μην είναι επιλεγμένο
- int getButtonCount() η οποία επιστρέφει τον αριθμό των πλήκτρων στην ομάδα
- Enumeration<AbstractButton> getElements() η οποία επιστρέφει όλα τα πλήκτρα που συμμετέχουν στην ομάδα
- ButtonModel getSelection() η οποία επιστρέφει το μοντέλο του επιλεγμένου πλήκτρου
- boolean isSelected(ButtonModel m) η οποία επιστρέφει true εάν ένα ButtonModel είναι επιλεγμένο
- void remove(AbstractButton b) η οποία αφαιρεί το πλήκτρο από το ButtonGroup
- void setSelected(ButtonModel m, boolean b) η οποία καθορίζει την τιμή επιλογής για το δοσμένο ButtonModel. Μόνο ένα πλήκτρο μπορεί να είναι επιλεγμένο κάθε στιγμή.

Το παράδειγμα υλοποίησης μιας εφαρμογής με JRadioButton και ButtonGroup θα περιέχει τρία JRadioButton σε ένα ButtonGroup που θα βρίσκονται σε ένα JFrame.

Αρχικά εισάγουμε το συστατικό GridBagLayout ώστε να μπορέσουμε στο παράδειγμα να οργανώσουμε την θέση των πλήκτρων.

```
import java.awt.GridBagLayout;
```

Στην συνέχεια δημιουργούμε το JFrame και τα τρία JRadioButtons, θέτουμε το GridBagLayout στο frame και τοποθετούμε τα πλήκτρα στο JFrame

```
JFrame frame = new JFrame("Παράδειγμα JRadioButton");
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.setLayout(new GridBagLayout());

JRadioButton button1 = new JRadioButton("Πρώτο");
```



`

```
JRadioButton button2 = new JRadioButton("Δεύτερο");
JRadioButton button3 = new JRadioButton("Τρίτο");

frame.add(button1);
frame.add(button2);
frame.add(button3);
```

Στην συνέχεια δημιουργούμε το GroupButton στο οποίο τοποθετούμε τα τρία JRadioButton

```
ButtonGroup group = new ButtonGroup();
group.add(button1);
group.add(button2);
group.add(button3);
```

Και τέλος σχεδιάζεται το JFrame στην οθόνη

```
frame.pack();
frame.setVisible(true);
```

### 4.3.7 JCheckBox

Το JCheckBox είναι η υλοποίηση ενός πλαισίου επιλογής, ένα αντικείμενο που μπορεί να επιλεχθεί ή να αποεπιλεχθεί και το οποίο εμφανίζει την κατάστασή του στον χρήστη. Συμβατικά οποιοσδήποτε αριθμός JCheckbox μέσα σε μια ομάδα μπορεί να επιλεχθεί.

Τα πλήκτρα μπορούν να ρυθμιστούν και σε ένα βαθμό να ελεγχθούν μέσω Actions. Χρησιμοποιώντας ένα Action με ένα πλήκτρο έχει πολλά πλεονεκτήματα πέρα από το να ρυθμίζεται άμεσα το πλήκτρο.

Οι δομητές της κλάσης αυτής είναι:

- JCheckBox() οποίος δημιουργεί ένα αρχικά αποεπιλεγμένο πλήκτρο πλαισίου επιλογής χωρίς κείμενο και εικονίδιο
- JCheckBox(Action a) ο οποίος δημιουργεί ένα JCheckBox του οποίου οι ιδιότητες λαμβάνονται από το δοσμένο Action
- JCheckBox(Icon icon) ο οποίος δημιουργεί ένα αρχικά αποεπιλεγμένο JCheckBox με εικονίδιο
- JCheckBox(Icon icon, boolean selected) ο οποίος δημιουργεί ένα JcheckBox με εικονίδιο και ιδιότητα του αν θα είναι επιλεγμένο ή όχι
- JCheckBox(String text) ο οποίος δημιουργεί ένα αρχικά αποεπιλεγμένο JCheckBox με κείμενο
- JCheckBox(String text, boolean selected) που δημιουργεί ένα JCheckBox με κείμενο και την ιδιότητα του αν θα είναι επιλεγμένο ή όχι



`

- JCheckBox(String text, Icon icon) που δημιουργεί ένα αρχικά αποεπιλεγμένο πλήκτρο πλαισίου επιλογής με το δοσμένο κείμενο και εικόνα
- JCheckBox(String text, Icon icon, boolean selected) που δημιουργεί ένα JCheckBox με κείμενο και εικόνα και με την ιδιότητα του αν θα είναι επιλεγμένο ή όχι

Στο παράδειγμα που ακολουθεί θα δημιουργηθεί ένα JFrame που θα περιέχει 3 JCheckBox τα οποία είναι τοποθετημένα σε ένα GridBagLayout που προστέθηκε στο JFrame. Το GridBagLayout χωρίζει το container σε μεταβλητού μεγέθους και αριθμού ορθογώνια στο καθένα από τα οποία μπορεί να τοποθετηθεί από ένα συστατικό.

Για την δυνατότητα χρήσης του GridBagLayout εισήχθη η κατάλληλη βιβλιοθήκη:

```
import java.awt.GridBagLayout;
```

Στην συνέχεια δημιουργήθηκε το JFrame και τα τρία JCheckBox με κείμενο «Πρώτο», «Δεύτερο» και «Τρίτο»

```
JFrame frame = new JFrame("Παράδειγμα JFrame");
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

JCheckBox j1 = new JCheckBox("Πρώτο");
JCheckBox j2 = new JCheckBox("Δεύτερο");
JCheckBox j3 = new JCheckBox("Τρίτο");
```

Έπειτα καθορίζεται ο τρόπος σχεδιασμού των συστατικών στο JFrame και προστίθενται σε αυτό τα JCheckBox και τέλος παρουσιάζεται το παράθυρο με τα συστατικά του.

```
frame.setLayout(new GridBagLayout());
frame.add(j1);
frame.add(j2);
frame.add(j3);

frame.pack();
frame.setVisible(true);
```



`

### 4.3.8 JComboBox

Το JComboBox είναι ένα συστατικό που συνδυάζει ένα πλήκτρο ή ένα επεξεργάσιμο πεδίο και μία drop-down λίστα . Ο χρήστης μπορεί να επιλέξει μία τιμή από την λίστα, η οποία εμφανίζεται κατ' απαίτηση του χρήστη. Αν το JComboBox είναι επεξεργάσιμο τότε το αντικείμενο περιλαμβάνει ένα επεξεργάσιμο πεδίο στο οποίο ο χρήστης μπορεί να πληκτρολογήσει μια τιμή.

Οι δομητές είναι:

- JComboBox() που δημιουργεί ένα JComboBox με το προεπιλεγμένο μοντέλο δεδομένων
- JComboBox(ComboBoxModel aModel), δημιουργεί να JComboBox που λαμβάνει τα αντικείμενά του από το υπάρχον ComboBoxModel
- JComboBox(Object[] items) ο οποίος δημιουργεί ένα JComboBox που περιέχει τα αντικείμενα στο δοσμένο πίνακα
- JComboBox(Vector<?> items) που δημιουργεί ένα JComboBox που περιέχει τα αντικείμενα στο δοσμένο Vector

Άλλες μέθοδοι για την κλαση αυτή είναι:

- void addItem(Object anObject) η οποία προσθέτει το δοσμένο αντικείμενο στην λίστα αντικειμένων. Αυτή η μέθοδος λειτουργεί μόνο όταν το JComboBox χρησιμοποιεί ένα μεταβλητό μοντέλο δεδομένων
- void addItemListener(ItemListener aListener) η οποία προσθέτει ένα ακροατή αντικειμένων. Ο δοσμένος ακροατής θα λαμβάνει ένα ή δύο ItemEvent όταν το επιλεγμένο αντικείμενο αλλάζει
- Object getItemAt(int index) η οποία επιστρέφει το αντικείμενο της λίστας στην συγκεκριμένη θέση. Αν το index είναι εκτός ορίων τότε η μέθοδος θα επιστρέψει null
- int getItemCount() η οποία επιστρέφει τον αριθμό των αντικειμένων στην λίστα
- ComboBoxModel getModel() η οποία επιστρέφει το μοντέλο δεδομένων που χρησιμοποιείται από το JComboBox.
- Object getSelectedItem() η οποία επιστρέφει το τρέχον επιλεγμένο αντικείμενο. Αν το JComboBox είναι επεξεργάσιμο τότε αυτή η τιμή μπορεί να μήν έχει προστεθεί στο JComboBox με μεθόδους όπως addItem ή τους δομητές
- Object[] getSelectedObjects() η οποία επιστρέφει έναν πίνακα που περιέχει τα επιλεγμένα αντικείμενα
- void insertItemAt(Object anObject, int index) η οποία εισάγει ένα αντικείμενο στην λίστα αντικειμένων στην συγκεκριμένη θέση. Αυτή η μέθοδος λειτουργεί μόνο όταν το JComboBox χρησιμοποιεί ένα μεταβλητό μοντέλο δεδομένων
- boolean isEditable() η οποία επιστρέφει true αν το JComboBox είναι επεξεργάσιμο. Από προεπιλογή δεν είναι επεξεργάσιμο



`

- void removeAllItems() η οποία αφαιρεί όλα τα αντικείμενα από την λίστα
- void removeItem(Object anObject) η οποία αφαιρεί το δοσμένο αντικείμενο από την λίστα. Αυτή η μέθοδος λειτουργεί μόνο όταν το JComboBox χρησιμοποιεί ένα μεταβλητό μοντέλο δεδομένων
- void removeItemAt(int anIndex) η οποία αφαιρεί το αντικείμενο στην δοσμένη θέση. Αυτή η μέθοδος λειτουργεί μόνο όταν το JComboBox χρησιμοποιεί ένα μεταβλητό μοντέλο δεδομένων
- void setModel(ComboBoxModel aModel) η οποία θέτει το μοντέλο δεδομένων που το JComboBox χρησιμοποιεί για να λάβει την λίστα των αντικειμένων

Στο παράδειγμα επίδειξης της λειτουργίας του JComboBox θα δημιουργηθεί μια εφαρμογή που περιέχει ένα JComboBox με λίστα τριών αντικειμένων.

Αρχικά δημιουργούνται τα JFrame και το κενό JComboBox

```
JFrame frame = new JFrame("Παράδειγμα JComboBox");
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

JComboBox box = new JComboBox();
```

Στην συνέχεια προστίθενται αντικείμενα στο JComboBox

```
box.addItem("Πρώτο");
box.addItem("Δεύτερο");
box.addItem("Τρίτο");
```

Τέλος εισάγεται το JComboBox στο JFrame και το τελευταίο σχεδιάζεται στην οθόνη.

```
frame.add(box);
frame.pack();
frame.setVisible(true);
```

### 4.3.9 JTextArea

Το JTextArea είναι μια περιοχή πολλών γραμμών που εμφανίζει απλό κείμενο. Προορίζεται να είναι ένα ελαφρύ συστατικό που προσφέρει πηγαία συμβατότητα με την κλάση java.awt.TextArea.

Η java.awt.TextArea χειρίζεται εσωτερικά την κύλιση. Το JTextArea είναι διαφορετικό και δεν χειρίζεται την κύλιση, αλλά υλοποιεί την διασύνδεση Scrollable. Αυτό του επιτρέπει να τοποθετηθεί μέσα σε ένα JScrollPane αν η συμπεριφορά κύλισης είναι επιθυμητή, ή να χρησιμοποιηθεί απευθείας αν όχι.

Το java.awt.TextArea έχει την ικανότητα να τυλίγει τις γραμμές. Αυτό ελέγχεται από την πολιτική οριζόντιας κύλισης. Από την στιγμή που η κύλιση δεν γίνεται



`

άμεσα από την JTextArea, η προς τα πίσω συμβατότητα πρέπει να παρέχεται με άλλο τρόπο. Η JTextArea έχει δεσμευμένη ιδιότητα να τύλιγμα γραμμών που ελέγχει εάν θα τυλίξει τις γραμμές ή όχι. Από προεπιλογή η ιδιότητα αυτή είναι ορισμένη σε false (να μην τυλίγονται οι γραμμές).

Οι δομητές της κλάσης αυτής είναι:

- JTextArea() ο οποίος δημιουργεί ένα νέο κενό JTextArea
- JTextArea(Document doc) ο οποίος δημιουργεί ένα νέο JTextArea με το καθορισμένο μοντέλο εγγράφου, και προεπιλογές για όλες τις υπόλοιπες παραμέτρους (null, 0, 0)
- JTextArea(Document doc, String text, int rows, int columns) ο οποίος δημιουργεί ένα νέο JTextArea με το καθορισμένο αριθμό γραμμών και στηλών, και το καθορισμένο μοντέλο
- JTextArea(int rows, int columns) ο οποίος δημιουργεί ένα νέο κενό JTextArea με το καθορισμένο αριθμό γραμμών και στηλών
- JTextArea(String text) ο οποίος δημιουργεί ένα νέο JTextArea με το καθορισμένο κείμενο να εμφανίζεται.
- JTextArea(String text, int rows, int columns) ο οποίος δημιουργεί ένα νέο JTextArea με το δοσμένο κείμενο και με τον καθορισμένο αριθμό γραμμών και στηλών

Άλλες μέθοδοι για αυτήν την κλάση είναι:

- void append(String str) η οποία επισυνάπτει το δοσμένο κείμενο στο τέλος του εγγράφου. Δεν κάνει τίποτα αν το μοντέλο είναι null, ή η συμβολοσειρά είναι null ή κενή
- Document createDefaultModel() η οποία δημιουργεί την προεπιλεγμένη υλοποίηση για το μοντέλο που θα χρησιμοποιηθεί για την κατασκευή αν δεν δίνεται ένα ρητά. Ένα νέο αντικείμενο της PlainDocument επιστρέφεται
- void insert(String str, int pos) η οποία εισάγει το καθορισμένο κείμενο στην καθορισμένη θέση.
- void setColumns(int columns) η οποία ορίζει τον αριθμό των στηλών για αυτό το JTextArea.
- void setRows(int rows) η οποία ορίζει τον αριθμό των γραμμών για το συγκεκριμένο JTextArea

Θα υλοποιηθεί ένα παράδειγμα εφαρμογής που χρησιμοποιεί το JTextArea. Το JTextArea θα έχει μέγεθος δύο γραμμών και 10 στηλών και θα περιέχει κείμενο δύο γραμμών.

Αρχικά δημιουργούμε το JFrame και το JTextArea

```
JFrame frame = new JFrame("Παράδειγμα JTextArea");
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```



`

```
JTextArea area = new JTextArea(2, 10);
```

Στην συνέχεια εισάγουμε το κείμενο στο JTextArea

```
area.setText("Πρώτη γραμμή\nΔεύτερη γραμμή");
```

Τέλος προσθέτουμε το JTextArea στο JFrame και τα σχεδιάζουμε στην οθόνη

```
frame.add(area);
frame.pack();
frame.setVisible(true);
```

### 4.3.10 JTextField

Το JTextField είναι ένα ελαφρύ συστατικό που επιτρέπει την επεξεργασία μίας γραμμης κειμένου. Το JTextField προορίζεται να είναι συμβατό με την java.awt.TextField, αλλά περιέχει δυνατότητες που δεν έχει η κλάση java.awt.TextField.

Το JTextField έχει μια μέθοδο για να εγκαθιδρύσει μια συμβολοσειρά που χρησιμοποιείται ως η συμβολοσειρά εντολής για το γεγονός δράσης που τροφοδοτείται. Το java.awt.TextField χρησιμοποιούσε το κείμενο από το πεδίο ως την συμβολοσειρά εντολής για το ActionEvent. Το JTextField χρησιμοποιεί την σύνολο συμβολοσειράς εντολής μαζί με την μέθοδο SetActionCommand αν δεν είναι null, διαφορετικά θα χρησιμοποιήσει το κείμενο του πεδίου ως συμβατότητα με το java.awt.TextField.

Οι μέθοδοι setEchoChar και getEchoChar δεν προσφέρονται άμεσα για την αποφυγή νέων υλοποιήσεων ενός προστιθέμενου look-and-feel ακούσια εκθέτοντας χαρακτήρες κωδικών. Για την προσφορά υπηρεσιών κωδικών μια ξεχωριστή κλάση JPasswordField επεκτείνει την JTextField για να προσφέρει αυτήν την υπηρεσία με ένα ανεξέρτητα προστιθέμενο look-and-feel.

Το java.awt.TextField μπορεί να παρακολουθείται για αλλαγές με την πρόσθεση ένός Textlistener για TextEvent. Σε συστατικά βασισμένα στο JTextComponent, οι αλλαγές μεταδίδονται από το μοντέλο μέσω ενός DocumentEvent σε DocumentListener. Το DocumentEvent δίνει την τοποθεσία της αλλαγής και του είδους της αλλαγής που επιθυμείται. Το κομμάτι κώδικα μπορεί να είναι το εξής:

```
DocumentListener myListener = ??;
JTextField myArea = ??;
myArea.getDocument().addDocumentListener(myListener);
```

Δομητές αυτής της κλάσης είναι:



`

- JTextField() που δημιουργεί ένα νέο JtextField
- JTextField(Document doc, String text, int columns) ο οποίος δημιουργεί ένα νέο JTextField που χρησιμοποιεί το δοσμένο μοντέλο αποθήκευσης κειμένου και το δοσμένο αριθμό στηλών
- JTextField(int columns) ο οποίος δημιουργεί ένα κενό JTextField με τον δοσμένο αριθμό στηλών
- JTextField(String text) που δημιουργεί ένα νέο JTextField αρχικοποιημένο με το δοσμένο κείμενο
- JTextField(String text, int columns) ο οποίος δημιουργεί ένα νέο JTextField αρχικοποιημένο με το δοσμένο κείμενο και στήλες

Άλλες μέθοδοι για την κλάση αυτή είναι:

- protected void actionPropertyChanged(Action action, String propertyName) η οποία ανανεώνει την κατάσταση του πεδίου κειμένου σε απάντηση των αλλαγών ιδιοτήτων στην συσχετισμένη ενέργεια (action). Αυτή η μέθοδος επικαλείται από το PropertyChangeListener που επιστρέφεται από την μέθοδο createActionPropertyChangeListener.
- void addActionListener(ActionListener l) η οποία προσθέτει το δοσμένο ακροατή ενεργειών για να λάβει ενέργειες γεγονότων από αυτό το πεδίο κειμένου

Για το παράδειγμα ενός JTextField θα δημιουργηθεί ένα JTextField σε ένα JFrame το οποίο θα έχει μέγεθος δέκα στηλών και θα περιέχει κείμενο.

Αρχικά δημιουργούμε το JFrame και το JTextField

```
JFrame frame = new JFrame("Παράδειγμα JtextField");
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

JTextField field = new JTextField(10);
```

Στην συνέχεια εισάγουμε το κείμενο στο JTextField

```
field.setText("Text");
```

Τέλος προσθέτουμε το JTextField στο JFrame και τα σχεδιάζουμε στην οθόνη

```
frame.add(field);
frame.pack();
frame.setVisible(true);
```

### 4.3.11 JPanel

Το JPanel είναι ένα ελαφρύ container το οποίο μπορεί να φιλοξενήσει άλλα συστατικά της Swing.

Δομητές αυτής της κλάσης είναι:

- JPanel() που δημιουργεί ένα νέο JPanel με διπλό buffer και με διάταξη ροής (flow layout)
- JPanel(boolean isDoubleBuffered) που δημιουργεί ένα νέο JPanel με FlowLayout και την δοσμένη στρατηγική για buffer
- JPanel(LayoutManager layout) που δημιουργεί ένα νέο JPanel με buffer και με τον δοσμένο διαχειριστή διάταξης
- JPanel(LayoutManager layout, boolean isDoubleBuffered) που δημιουργεί ένα νέο JPanel με τον δοσμένο διαχειριστή διάταξης και στρατιγική για buffer

Άλλες μέθοδοι είναι:

- void updateUI() η οποία επανεφέρει την ιδιότητα UI με τιμή από το τρέχον look and feel
- PanelUI getUI() η οποία επιστρέφει το αντικείμενο look and feel που αναπαριστά αυτό το αντικείμενο
- void setUI(PanelUI ui) η οποία θέτει το αντικείμενο look and feel που αναπαριστά αυτό το συστατικό
- String getUIClassID() η οποία επιστρέφει ένα String που προσδιορίζει το όνομα από την κλάση look and feel που αναπαριστά αυτό το αντικείμενο
- String paramString() που επιστρέφει ένα String αναπαράστασης αυτού του JPanel.

Στο παράδειγμα υλοποίησης εφαρμογής που χρησιμοποιεί το JPanel θα εισάγουμε ένα JPanel στο JFrame που βρίσκεται στο πρώτο επίπεδο στο παράθυρο.Στην συνέχεια θα βάψουμε κόκκινο το JPanel και θα του προσθέσουμε ένα JLabel με κείμενο.

Αρχικά εισάγουμε την βιβλιοθήκη Color

```
import java.awt.Color;
```

Δημιουργούμε το JFrame και το JPanel

```
JFrame frame = new JFrame("Παράδειγμα JPanel");
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

JPanel panel = new JPanel();
```

Βάφουμε το JPanel και του προσθέτουμε ένα JLabel με κείμενο

```
panel.setBackground(Color.red);
panel.add(new JLabel("Κείμενο"));
frame.add(panel);
```

Τέλος προσθέτουμε το JPanel στο JFrame και τα σχεδιάζουμε στην οθόνη



`

```
frame.add(panel);
frame.pack();
frame.setVisible(true);
```

## ΕΠΙΛΟΓΟΣ

Ο σχεδιασμός της Swing βασίζεται στο μοντέλο MVC και τηρεί όλα τα πρότυπα της Java, όπως το Write once, Run anywhere.

Τα συστατικά της Swing περιλαμβάνουν container, και άλλα συστατικά, όπως κουμπιά και πλαίσια για παρουσίαση και εγγραφή κειμένου, τα οποία τοποθετούνται μέσα στα πρώτα.

Στο επόμενο κεφάλαιο θα δούμε πως μπορούμε να συνδυάσουμε τα διάφορα συστατικά της Swing για να δημιουργήσουμε μια διεπαφή χρήστη.



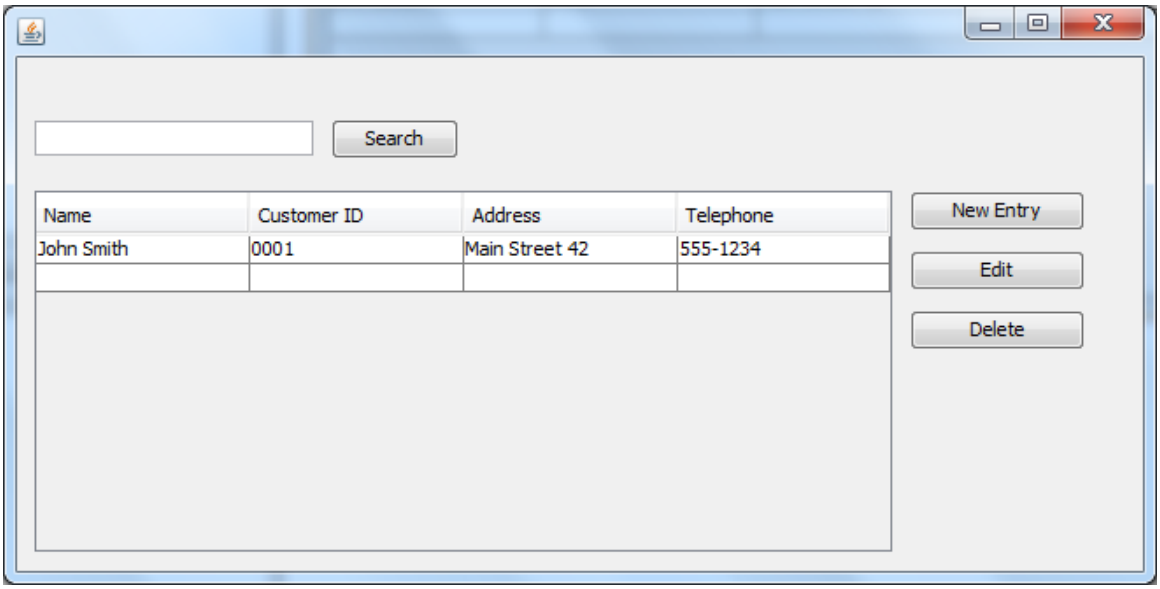
`

## 5. Ανάπτυξη Γραφικών Διεπαφών Χρήστη (templates) σε γλώσσα προγραμ. Java

### ΕΙΣΑΓΩΓΗ

Για αυτήν την ενότητα δημιουργήθηκαν φόρμες γραφικών διεπαφών χρήστη χρησιμοποιώντας την βιβλιοθήκη Swing της Java. Τα συστατικά των φορμών δεν παρέχουν κάποια λειτουργία.

### 5.1 Βασική φόρμα




Εικόνα 23: Βασική Φόρμα

Σε αυτήν φόρμα χρησιμοποιήθηκαν ένα JTextField και ένα JButton για την αναζήτηση.

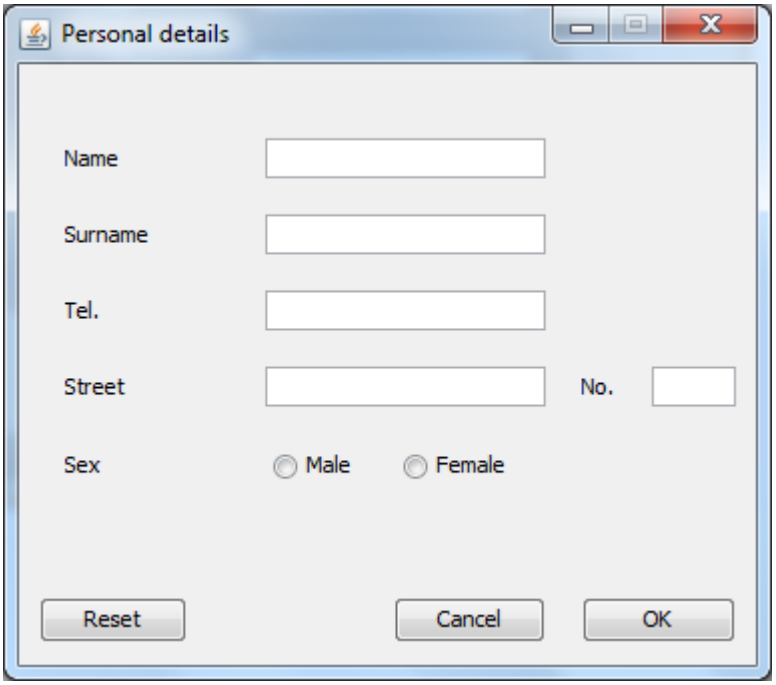
Το μεγαλύτερο μέρος της φόρμας καταλαμβάνει το JTable, ένας πίνακας δύο διαστάσεων, ο οποίος χρησιμοποιείται μόνο για να προβάλει πληροφορίες. Το JTable τοποθετήθηκε μέσα σε ένα JScrollPanel που προσθέτει μπάρες κύλισης στο αντικείμενο αν αυτό είναι μεγαλύτερο από το αντικείμενο που το περιέχει.

Τέλος στα δεξιά υπάρχουν 3 JButtons για την προσθήκη, επεξεργασία και διαγραφή εγγραφών από το JTable.



`

### 5.2 Φόρμα επεξεργασίας προσωπικών στοιχείων.




Εικόνα 24: Φόρμα επεξεργασίας προσωπικών στοιχείων

Αυτή η φόρμα αναπαριστά μια εφαρμογή εισαγωγής/επεξεργασίας προσωπικών στοιχείων.

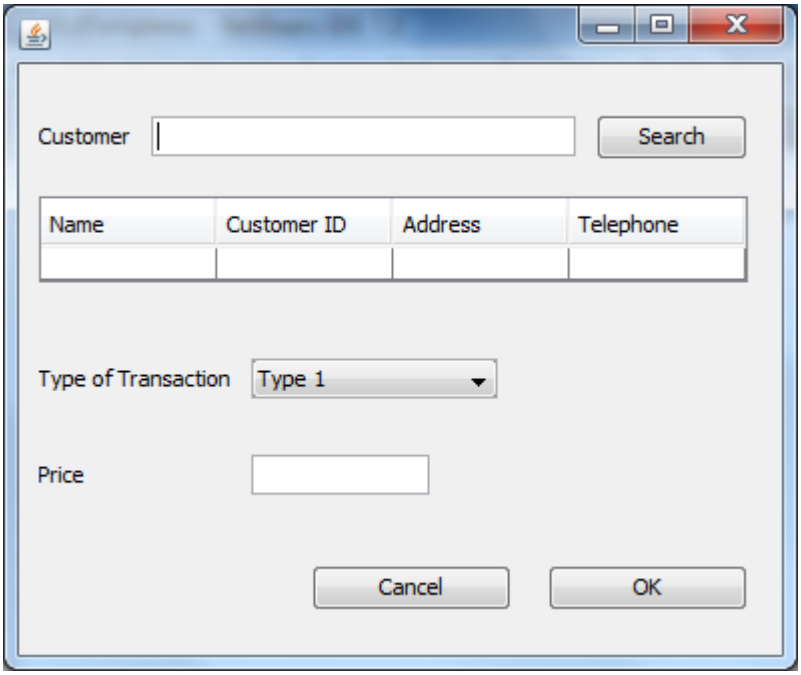
Χρησιμοποιήθηκαν JLabel για την επιγραφή των στοιχείων που ζητούνται (Name, Surname, κτλ) και JTextField για το πεδίο εισαγωγής από τον χρήστη, εκτός από το τελευταίο στοιχείο, το φύλο, όπου έχουν τοποθετηθεί δύο JRadioButton.

Επίσης υπάρχουν 3 JButton, το πρώτο (Reset) είναι για τον καθαρισμό των JTextField, το δεύτερο (Cancel) για την ακύρωση της ενέργειας, και το τρίτο (OK) για την ολοκλήρωσή της.



`

### 5.3 Φόρμα συναλλαγών




Εικόνα 25: Φόρμα συναλλαγών

Σε αυτή τη φόρμα δημιουργούνται συναλλαγές για τον πελάτη.

Ένα JLabel, ένα JtextField και ένα JButton χρησιμοποιούνται για το κομμάτι την αναζήτησης του πελάτη. Στη συνέχεια ένα JTable χρησιμοποιείται για την εμφάνιση του αποτελέσματος την αναζήτησης. Για την επιλογή του τύπου συναλλαγής υπάρχει ένα JComboBox. Για την τιμή της συναλλαγής ο χρήστης εισάγει την τιμή στο αντίστοιχο JTextField. Τέλος υπάρχουν δύο JButtons (Cancel, OK) που ακυρώνουν και ολοκληρώνουν την διαδικασία, αντίστοιχα.

## ΕΠΙΛΟΓΟΣ

Σε αυτήν την ενότητα είδαμε τον τρόπο με τον οποίο μπορούμε να συνδυάσουμε τα συστατικά γραφικών της Java ώστε να δημιουργήσουμε πρότυπα γραφικών διεπαφών χρήστη. Όταν σε αυτές τις γραφικές διεπαφές προστεθεί μια λειτουργικότητα, όπως αρχιτεκτονική πελάτη-εξυπηρετητή για την σύνδεση με βάσεις δεδομένων, τότε θα έχουμε μια ολοκληρωμένη εφαρμογή με γραφικό περιβάλον.



`

# 6. Η Αρχιτεκτονική Πελάτη-Εξυπηρετητή. Μικρή εφαρμογή υλοποίησης.

## ΕΙΣΑΓΩΓΗ

Σε αυτό το κεφάλαιο θα δούμε την αρχιτεκτονική πελάτη-εξυπηρετητή (client-server), μια αρχιτεκτονική που χρησιμοποιείται σε πολλές εφαρμογές.

Θα εξεταστούν τα βασικά ζητήματα της αρχιτεκτονικής αυτής και θα παρουσιαστεί μια απλή εφαρμογή που υλοποιεί αυτό το πρότυπο.

## 6.1 Η αρχιτεκτονική Πελάτη Εξυπηρετητή

Ο όρος «Πελάτης-Εξυπηρετητής» είναι ένας γενικός όρος για κάθε αρχιτεκτονική εφαρμογής που χωρίζει την επεξεργασία ανάμεσα σε δύο ή περισσότερες διεργασίες, σε δύο ή περισσότερους υπολογιστές. Κάθε εφαρμογή βάσης δεδομένων είναι μια εφαρμογή πελάτη-εξυπηρετητή αν διαχειρίζεται τα δεδομένα και την ανάκτησή τους στην διεργασία της βάσης δεδομένων και την διαχείριση και παρουσίαση των δεδομένων κάπου αλλού. Ο εξυπηρετητής είναι η μηχανή βάσης δεδομένων που αποθηκεύει τα δεδομένα, και ο πελάτης είναι η διεργασία που ανακτά ή δημιουργεί τα δεδομένα. Η ιδέα πίσω από αυτήν την αρχιτεκτονική είναι η εφαρμογή να προσφέρει σε πολλούς χρήστες πρόσβαση στα ίδια δεδομένα.

Η πιο απλή μορφή της αρχιτεκτονικής πελάτη-εξυπηρετητή ονομάζεται αρχιτεκτονική δύο βαθμίδων (two-tier). Στην πραγματικότητα οι περισσότερες αρχιτεκτονικές πελάτη-εξυπηρετητή είναι δύο βαθμίδων. Ο όρος «δύο βαθμίδες» περιγράφει τον τρόπο με τον οποίο η επεξεργασία της εφαρμογής μπορεί να διαιρεθεί σε πολλαπλούς σταθμούς εργασίας με ένα ομοιόμορφο επίπεδο παρουσίασης που επικοινωνεί με ένα κεντρικό επίπεδο αποθήκευσης δεδομένων.

Οι περισσότερες εφαρμογές στο διαδίκτυο, όπως το email, Telnet, FTP κ.α., είναι απλές εφαρμογές δύο βαθμίδων. Χωρίς να προσφέρουν πολλή ερμηνεία ή επεξεργασία δεδομένων, αυτές οι εφαρμογές προσφέρουν μια απλή διεπαφή για την πρόσβαση σε πληροφορίες μέσω του διαδικτύου. Όταν οι προγραμματιστές εφαρμογών γράφουν τις δικές τους εφαρμογές πελάτη-εξυπηρετητή, τείνουν να ακολουθούν την δομή δύο βαθμίδων.



`

### 6.2 Μια εφαρμογή υλοποίησης

Υλοποιήθηκε μια μικρή εφαρμογή επίδειξης της αρχιτεκτονικής πελάτη-εξυπηρετητή σε Java. Στην εφαρμογή ο πελάτης συνδέεται στον εξυπηρετητή και στην συνέχεια στέλνει ένα μήνυμα και δέχεται την απάντηση από τον εξυπηρετητή.

#### 6.2.1 Ο πελάτης

Στην εφαρμογή αυτό που κάνει ο πελάτης είναι να ανοίγει μία σύνδεση με τον εξυπηρετητή, και αν αυτή είναι επιτυχής στέλνει ένα μήνυμα και δέχεται μια απάντηση από αυτόν.

Αναλυτικά:

```
import java.net.*;

import java.io.*;
```

Αυτές είναι οι δύο βιβλιοθήκες που απαιτούνται για την εφαρμογή του πελάτη. Από την java.net χρησιμοποιείται η κλάση Socket, που αναπαριστά το ένα άκρο μιας σύνδεσης. Η java.io περιέχει κλάσεις που θα χρησιμοποιηθούν για την αποστολή και λήψη δεδομένων από sockets.

```
public class ClientSocketExample {

public static void main(String[] args) {

        Socket client = null;

        BufferedReader in = null;

        PrintWriter out = null;

        String Message = "Hello World";
```

Εδώ ξεκινά η main μέθοδος και δηλώνονται τα αντικείμενα που θα χρησιμοποιηθούν.

Το Socket για την σύνδεση με τον εξυπηρετητή, BufferedReader για την είσοδο δεδομένων από τον εξυπηρετηρή, PrintWriter για έξοδο και μια συμβολοσειρά για αποστολή στον εξυπηρετητή.



`

```
 try {
        client = new Socket("127.0.0.1", 5000);
        in = new BufferedReader (new InputStreamReader
(client.getInputStream()));
        out = new PrintWriter(client.getOutputStream(), true);
    }catch(IOException ex) {
        System.err.println(ex.getMessage());
        System.exit(-1);
    }
```

Με την δήλωση client = new Socket("127.0.0.1", 5000);, δημιουργείται το αντικείμενο Socket και γίνεται προσπάθεια για σύνδεση στον εξυπηρετητή, τα στοιχεία του οποίου δηλώνονται μέσα στις παρενθέσεις, διεύθυνση IP 127.0.0.1 (το ίδιο μηχάνημα στον οποίο βρίσκεται ο πελάτης), και θύρα που «ακούει» ο εξυπηρετητής την 5000.

Στην συνέχεια δημιουργούνται ρεύματα εισόδου και εξόδου με τα αντικείμενα BufferedReader και PrintWriter αντίστοιχα, τα οποία δεσμεύονται στον αντικείμενο Socket.

Τα παραπάνω εσωκλείονται σε μια πρόταση try catch επειδή σε περίπτωση που για παράδειγμα, ο εξυπηρετητής δεν λειτουργεί ή δεν «ακούει» την δοσμένη θύρα τότε «πετάνε» οι προτάσεις εξαίρεση IOException.

```
    out.println(Message);
    System.out.println("Client: " + Message);
```

Το μήνυμα τελικά αποστέλεται στον εξυπηρετητή με την μέθοδο println του αντικειμένου PrintWriter.

```
    String input = null;
    try {
        //Receive data from server
        while((input = in.readLine()) != null) {
        System.out.println("Server: " + input); }
```



`

```
} catch (IOException ex) {
    System.err.println(ex.getMessage());
}
```

Εδώ γίνεται λήψη όλων των μηνυμάτων που αποστέλει ο εξυπηρετητής τα οποία είναι σε μορφή συμβολοσειράς.

```
client.close();
out.close();
in.close();
```

Τέλος κλείνουν όλα τα ρεύματα που χρησιμοποιήθηκαν στην εφαρμογή.

### 6.2.2 Ο εξυπηρετητής

Ο εξυπηρετητής αρχικά ακούει κάποια θύρα του μηχανήματος για εισερχόμενες συνδέσεις. Αφού δημιουργηθεί η σύνδεση, τότε δέχεται την αίτηση από τον πελάτη και απαντά ανάλογα.

Συγκεκριμένα:

```
import java.net.*;
import java.io.*;
```

Όπως και στον πελάτη, εισάγονται οι κατάλληλες βιβλιοθήκες.

```
public class ServerSocketExample {
    static ServerSocket server = null;
    static Socket client = null;
    static PrintWriter out = null;
    static BufferedReader in = null;
    final static String inputMessage = "Hello World";
    static String outputMessage = "Hi!";
```

Δηλώνονται τα αντικείμενα τα οποία θα χρησιμοποιηθούν. Το αντικείμενο ServerSocket αναπαριστά το μέρος του εξυπηρετητή σε μια σύνδεση. Τα υπόλοιπα λειτουργούν ομοίως με τα αντίστοιχα αντικείμενα του πελάτη.



`

```
public static void main(String[] args) {
try {
    server = new ServerSocket(5000);
}catch(IOException ex) {
    System.err.println(ex.getMessage());
    System.exit(1);
}
```

Ο εξυπηρετητής «ακούει» μια συγκεκριμένη θύρα, εδώ την 5000. Η πρόταση εσωκλείεται σε try catch επειδή αν η θύρα που έχει οριστεί χρησιμοποιείται ήδη από άλλη εφαρμογή στον ίδιο υπολογιστή τότε «πετάει» εξαίρεση IOException.

```
    try {
      client = server.accept();
      }catch(IOException ex) {
      System.err.println(ex.getMessage());
      System.exit(2);
    }
```

Ο εξυπηρετητής περιμένει για εισερχόμενες συνδέσεις με το συγκεκριμένο socket. Η μέθοδος accept κλειδώνει μέχρι να δεχτεί μια αίτηση.

```
try {
    out = new PrintWriter(client.getOutputStream(), true);
    in = new BufferedReader(new InputStreamReader(client.getInputStream()));
} catch (IOException ex) {
    System.err.println(ex.getMessage());
    System.exit(3);
}
```

Δημιουργούνται τα ρεύματα εισόδου και εξόδου πάνω στο socket.



`

```
try {
    if(in.readLine().compareTo(inputMessage) == 0);
    out.println(outputMessage);
} catch (IOException ex) {
    System.err.println(ex.getMessage());
    System.exit(4);
}
```

Εδώ ο εξυπηρετητής εκτελεί μια ενέργεια πάνω στην είσοδο που δέχεται από τον πελάτη. Συγκεκριμένα, ελέγχει αν η είσοδος που δέχτηκε αντιστοιχεί στην συμβολοσειρά “Hello World” και τότε απαντά στον πελάτη με “Hi”.

```
try {
    server.close();
    client.close();
    in.close();
    out.close();
}catch(IOException ex) {
    System.err.println(ex.getMessage());
    System.exit(5);
}
```

Τέλος κλείνουν όλα τα ρεύματα και sockets που είχαν ανοίξει για την εφαρμογή.

## ΕΠΙΛΟΓΟΣ

Ανακεφαλαιώνοντας, σε μια εφαρμογή πελάτη εξυπηρετητή, ο εξυπηρετητής δημιουργεί τα αντικείμενα socket και αντικείμενα για ρεύματα εισόδου και εξόδου, και περιμένει μια αίτηση από τον πελάτη. Όταν δεχθεί την αίτηση τότε εκτελεί τις ενέργειες που απαιτούνται. Ο πελάτης από την πλευρά του δημιουργεί και αυτός τα αντικείμενα socket και ρευμάτων εισόδου-εξόδου. Στέλνει αίτηση για σύνδεση στον εξυπηρετητή μαζί με δεδομένα, αν είναι απαραίτητο από την εφαρμογή και



`

στην συνέχεια λαμβάνει την απάντηση από τον εξυπηρετητή μαζί με δεδομένα αν υπάρχουν.

Στο επόμενο κεφάλαιο θα δούμε πως, μαζί με γραφικές διεπαφές χρήστη μπορεί να υλοποιηθεί ένα ολοκληρωμένο πρόγραμμα που υλοποιεί την αρχιτεκτονική πελάτη-εξυπηρετητή.



`

# 7. Υλοποίηση εργαλείου χειρισμού των Βάσεων Δεδομένων με την χρήση γραφικής διεπαφής χρήστη (desktop).

## ΕΙΣΑΓΩΓΗ

Σε αυτό το κεφάλαιο υλοποιήθηκε μια εφαρμογή η οποία δημιουργεί συστατικά γραφικών διεπαφών χρήστη, τα οποία παρέχουν λειτουργίες για την σύνδεση με βάση δεδομένων. Αυτές οι λειτουργίες περιλαμβάνουν τις βασικές ενέργειες προς μια βάση δεδομένων, όπως δημιουργία και διαγραφή πίνακα, εισαγωγή, ενημέρωση και διαγραφή δεδομένων καθώς και προβολή δεδομένων.

## 7.1 Βάση Δεδομένων

Για την χρήση της εφαρμογής δημιουργήθηκε μια βάση δεδομένων η οποία αναπαριστά μία βάση δεδομένων πελατολογίου.

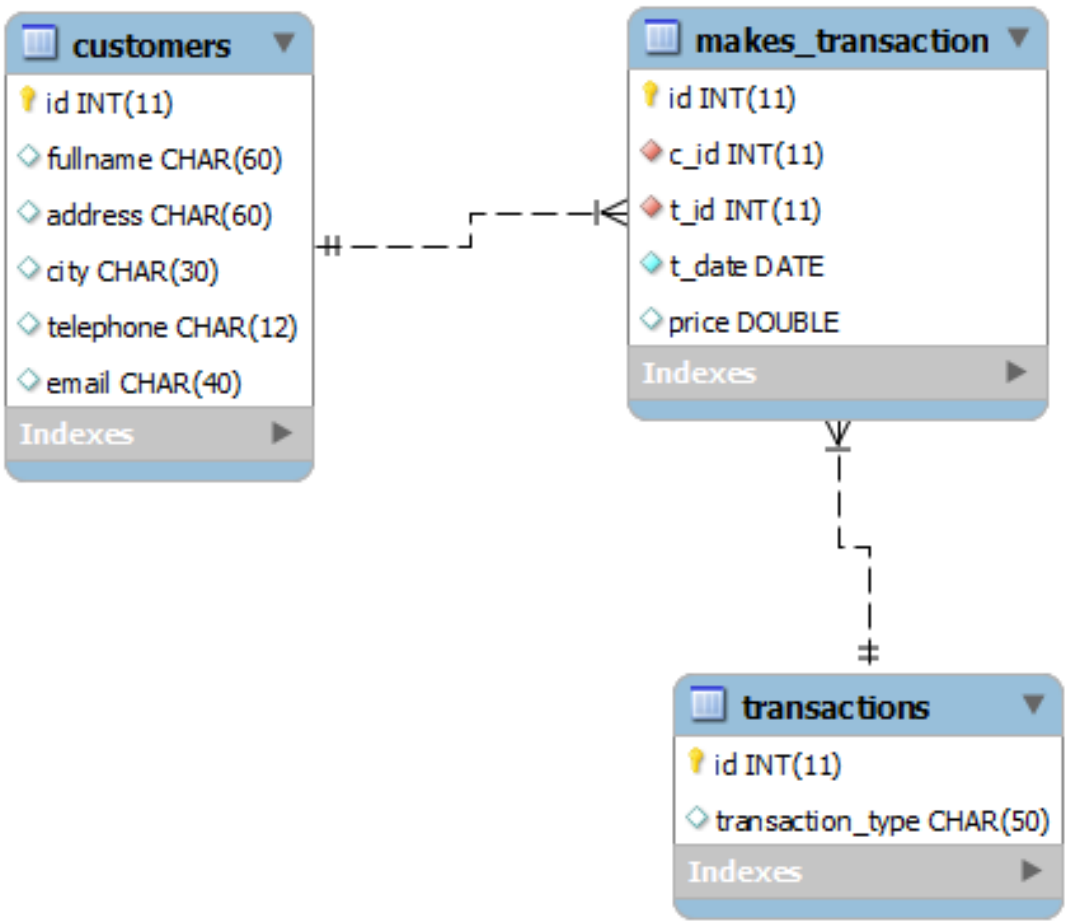
Η βάση δεδομένων υλοποιήθηκε σε MySQL και σε PostgreSQL. Η δομή της βάσης έχει ως εξής:

- Πίνακας customers: Περιέχει τα στοιχεία των πελατών. Οι στήλες του πίνακα είναι: id, fullname, address, city, telephone, email. Για τα στοιχεία των πελατών χρησιμοποιήθηκαν τυχαία δεδομένα που βρέθηκαν στο [12].
- Πίνακας transactions: Πίνακας που αναπαριστά τους τύπους των συναλλαγών που μπορεί να κάνει κάποιος πελάτης. Ο πίνακας περιέχει τις στήλες id, type_of_transaction και οι τιμές του είναι sale και purchase.
- Πίνακας makes_transaction. Αναπαριστά την διαδικασία της συναλλαγής με έναν πελάτη και χρησιμεύει και ως ιστορικό συναλλαγών. Οι στήλες του είναι οι c_id, το οποίο είναι ξένο κλειδί για το id του πίνακα customers, t_id, ξένο κλειδί για το id του transactions και t_date, για την ημερομηνία που έγινε η συναλλαγή. Τα δεδομένα του πίνακα αυτού συμπληρώνονται κατά την εκτέλεση της εφαρμογής.

Ο κώδικας SQL δημιουργίας των πινάκων βρίσκεται στο Παράρτημα 3 για την MySQL και στο Παράρτημα 4 για την PostgreSQL.



`

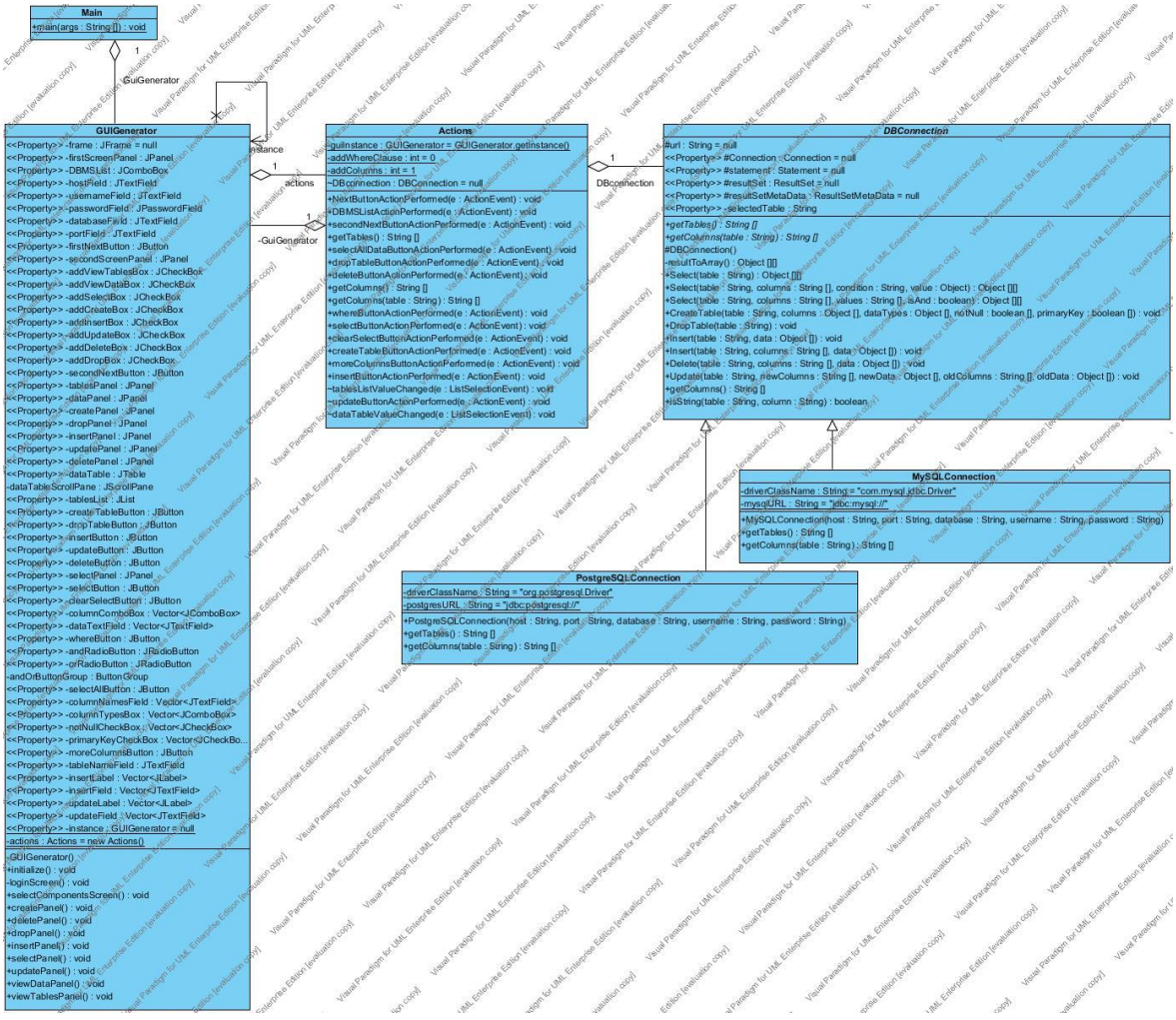



Εικόνα 26: Η δομή της βάσης δεδομένων πελατολογίου

## 7.2 Η Εφαρμογή

Στα πλαίσια αυτής της πτυχιακής εργασίας δημιουργήθηκε μία εφαρμογή η οποία εφαρμόζει όσα ειπώθηκαν στα προηγούμενα κεφάλαια. Συγκεκριμένα η εφαρμογή δίνει την δυνατότητα στον χρήστη να συνδεθεί μέσω γραφικής διεπαφής χρήστη σε μία βάση δεδομένων MySQL ή PostgreSQL. Μετά την σύνδεση με την βάση δεδομένων ο χρήστης μπορεί να διαλέξει τα συστατικά που θα τοποθετηθούν στο παράθυρο της γραφικής διεπαφής χρήστη, τα οποία δημιουργούνται δυναμικά και παρέχουν λειτουργίες για την πραγματοποίηση βασικών ενεργειών χειρισμού της βάσης δεδομένων.

Έχουν δημιουργηθεί οι κλάσεις DBConnection η οποία είναι αφηρημένη και προσφέρει την σύνδεση με την βάση δεδομένων, MySQLConnection και PostgreSQLConnection που κληρονομούν την πρώτη κλάση και εξειδικεύουν την συνδεσιμότητα για τα συγκεκριμένα περιβάλλοντα, Actions που υλοποιεί τις ενέργειες των συστατικών της γραφικής διεπαφής χρήστη, GuiGenerator που δημιουργεί τα συστατικά της γραφικής διεπαφής.




Εικόνα 27. Το διάγραμμα κλάσεων της εφαρμογής

### 7.2.1 DBConnection

Η κλάση αυτή είναι αφηρημένη ώστε να μην μπορεί να δημιουργηθεί αντικείμενο της κλάσης, αλλά μόνο αυτών που την επεκτείνουν και δεν είναι αφηρημένες. Περιέχει τα αντικείμενα Connection, ResultSet, ResultSetMetaData, Statement τα οποία είναι απαραίτητα για την σύνδεση με την βάση δεδομένων. Υλοποιεί μεθόδους των οποίων η υλοποίηση είναι ίδια για τα συστήματα MySQL και PostgreSQL οπότε δεν χρειάζεται να δημιουργηθούν ξεχωριστά. Οι μέθοδοι της κλάσης είναι:

- CreateTable, η οποία δέχεται ως ορίσματα τα ονόματα, τους τύπους των στηλών καθώς και τις στήλες οι οποίες είναι κύρια κλειδιά του πίνακα και αυτές που δεν μπορούν να πάρουν την τιμή Null. Δημιουργείται ένας πίνακας με τα συγκεκριμένα χαρακτηριστικά.
- DropTable που δέχεται σαν όρισμα το όνομα του πίνακα προς κατάργηση. Διαγράφει τον συγκεκριμένο πίνακα
- Select. Η μέθοδος αυτή έχει δύο διαφορετικές υλοποιήσεις. Η πρώτη που δέχεται για όρισμα το όνομα του πίνακα, επιστρέφει όλα τα περιεχόμενα του πίνακα αυτού. Η δεύτερη δέχεται σαν όρισμα το όνομα του πίνακα, έναν πίνακα (array) με τα ονόματα των στηλών, έναν πίνακα με τις τιμές των προηγούμενων τις οποίες θέλει ο χρήστης να αντιστοιχίσει στο αποτέλεσμα καθώς και μια λογική μεταβλητή για το λογικό «ΚΑΙ» ή το λογικό «Ή» ανάμεσα στις δοσμένες τιμές.
- Insert η οποία δέχεται σαν όρισμα το όνομα του πίνακα, το όνομα των στηλών και τα δεδομένα. Εισάγει τα δοσμένα δεδομένα στις δοσμένες στήλες του πίνακα.
- Delete που δέχεται σαν όρισμα το όνομα του πίνακα, το όνομα των στηλών και τα δεδομένα. Διαγράφει τις γραμμές του πίνακα στις οποίες αντιστοιχούν τα δοσμένα δεδομένα των στηλών.
- Update που δέχεται σαν ορίσματα το όνομα του πίνακα τα ονόματα των στηλών οι οποίες θα ανανεωθούν με νέα δεδομένα, τα νέα δεδομένα, τα ονόματα των υπάρχουσων στηλών και τα υπάρχοντα δεδομένα. Ανανεώνει τα παλιά δεδομένα με τα νέα στις δοσμένες στήλες.
- isString, δέχεται σαν όρισμα το όνομα του πίνακα και το όνομα μιας στήλης αυτού και επιστρέφει την λογική τιμή true αν ο τύπος της στήλης αυτής είναι τύπος κειμένου, πχ char, varchar κ.α. Χρησιμοποιείται από μεθόδους της ίδιας κλάσης.
- getTables η οποία είναι αφηρημένη και υλοποιείται από τις κλάσεις παιδιά
- getColumns η οποία επίσης είναι αφηρημένη και δέχεται σαν όρισμα το όνομα του πίνακα



`

### 7.2.2 MySQLConnection

Η κλάση αυτή επεκτείνει την DBConnection για το σύστημα διαχείρισης βάσεων δεδομένων MySQL. Στον δομητή της δημιουργεί την σύνδεση φορτώνοντας τον οδηγό και δημιουργεί τα αντικείμενα Connection και Statement όπως αναφέρθηκαν στο κεφάλαιο 2. Επίσης υλοποιεί της αφηρημένες μεθόδους της DBConnection getTables και getColumns. Η πρώτη επιστρέφει τα ονόματα των πινάκων της βάσης δεδομένων στην οποία είναι συνδεδεμένη η εφαρμογή και η δεύτερη μέθοδος επιστρέφει τα ονόματα των στηλών του πίνακα που δίνεται ως όρισμα. Να σημειωθεί πως η MySQL προσφέρει εντολές για την εύκολη προβολή κάποιων στοιχείων. Στην περίπτωσή μας, για την προβολή των πινάκων είναι διαθέσιμη η εντολή “SHOW TABLES;” και για την προβολή των στηλών ενός πίνακα η εντολή “SHOW COLUMNS FROM όνομα_πίνακα;”.

### 7.2.3 PostreSQLConnection

Η κλαση αυτή επεκτείνει την DBConnection και αναπαριστά την σύνδεση στο σύστημα διαχείρισης βάσεων δεδομένων PostgreSQL. Ισχύουν τα ίδια όπως στην MySQLConnection εκτός από την υλοποίηση των μεθόδων getTables και getColumns. Στην PostgreSQL τα στοιχεία των πινάκων αποθηκεύονται σε άλλους πίνακες. Οπότε για την προβολή των πινάκων εκτελείται η εντολή “SQL SELECT table_name FROM information_schema.tables WHERE table_schema = 'public';” και για την προβολή των στηλών ενός πίνακα η εντολή “SELECT column_name FROM information_schema.columns WHERE table_name= όνομα_πίνακα;”

## 7.3 GUIGenerator

Η κλάση αυτή υλοποιεί το πρότυπο σχεδίασης «Μοναδιαίο» ώστε να υπάρχει μόνο ένα στιγμιότυπο της κλάσης αυτής που θα παρέχει καθολικό σημείο πρόσβασης σε αυτό. Αυτό είναι δυνατό με το να τεθεί ο δομητής με private πρόσβαση, να περιέχει η κλάση ένα static αντικείμενο του εαυτού της στο οποίο μπορούν να έχουν πρόσβαση άλλες κλάσεις μέσω της μεθόδου getInstance η οποία επιστρέφει το αντικείμενο αυτό.

Η κλάση αυτή είναι υπεύθυνη για την δημιουργία της γραφικής διεπαφής. Συγκεκριμένα δηλώνει το σύνολο των συστατικών γραφικών διεπαφών (πλήκτρα, λίστες, πίνακες κ.α.) που μπορούν να χρειαστούν από την εφαρμογή.



`

Στην κλάση αυτή δημιουργούνται οι ακροατές γεγονότων για τα συστατικά, αλλά η υλοποίηση των γεγονότων γίνεται στην κλάση Actions την οποία θα αργότερα.

Οι μέθοδοι αυτής της κλάσης είναι:

- getInstance η οποία επιστρέφει το static αντικείμενο της κλάσης αυτής
- initialize η οποία αρχικοποιεί το JFrame που είναι το container πρώτου επιπέδου για την γραφική διεπαφή χρήστη και καλεί την επόμενη μέθοδο
- loginScreen η οποία δημιουργεί τα συστατικά που είναι απαραίτητα για την σύνδεση του χρήστη στην βάση δεδομένων
- selectComponentsScreen η οποία δημιουργεί την οθόνη για την επιλογή από τον χρήστη των συστατικών για να τοποθετηθούν στο παράθυρο. Οι επόμενες μέθοδοι είναι αυτές που δημιουργούν τα επιλεγμένα από τον χρήστη συστατικά.
- createPanel που δημιουργεί τα απαραίτητα συστατικά για την δημιουργία πίνακα στην βάση δεδομένων. Δημιουργεί ένα πεδίο κειμένου για το όνομα του πίνακα, ένα πεδίο κειμένου για το όνομα της στήλης, μία λίστα για τον τύπο της στήλης, ένα checkbox για το αν θα δέχεται ή όχι Null τιμές η στήλη και ένα checkbox για το αν θα είναι η στήλη κύριο κλειδί του πίνακα. Δημιουργεί επίσης ένα πλήκτρο το οποίο εισάγει επιπλέον τα τέσσερα τελευταία συστατικά κάθε φορά που πατιέται. Δημιουργεί επίσης ένα πλήκτρο για την ολοκλήρωση της διαδικασίας.
- deletePanel που δημιουργεί ένα JPanel που περιέχει ένα πλήκτρο για την διαγραφή της επιλεγμένης εγγραφής από το JTable που περιέχει τις εγγραφές του πίνακα
- dropPanel, περιέχει ένα πλήκτρο για την διαγραφή του επιλεγμένου πίνακα από την λίστα πινάκων
- insertPanel που δημιουργεί ένα JPanel που περιέχει πεδία κειμένου τα οποία μπορούν να αυξάνονται με το πάτημα ενός πλήκτρου στο ίδιο JPanel. Με αυτά τα συστατικά μπορεί ο χρήστης να εισάγει δεδομένα σε πίνακες
- selectPanel, δημιουργεί JPanel, πεδία κειμένου και λίστες με τα οποία μπορεί ο χρήστης να προβάλει τις εγγραφές του πίνακα της βάσης δεδομένων που αντιστοιχούν στα συμπληρωμένα στοιχεία
- updatePanel δημιουργεί Jpanel, πλήκτρα και πεδία κειμένου τα οποία μπορούν να αυξάνονται με το πάτημα του κατάλληλου πλήκτρου. Με αυτά τα συστατικά ο χρήστης μπορεί να αλλάξει τις τιμές των στοιχείων ενός πίνακα της βάσης δεδομένων
- viewDataPanel δημιουργεί ένα Jtable και ένα πλήκτρο για την προβολή εγγραφών ενός πίνακα
- viewTablesPanel δημιουργεί μια λίστα που περιέχει τους πίνακες της βάσης δεδομένων

Οι μέθοδοι που δημιουργούν τα στοιχεία για τις ενέργειες στην βάση δεδομένων συχνά απαιτούν στοιχεία που δημιουργούν άλλες μέθοδοι. Αν τα απαιτούμενα



`

στοιχεία δεν έχουν επιλεγεί για δημιουργία από τον χρήστη, τότε δημιουργούνται αυτόματα από την εφαρμογή. Για παράδειγμα η μέθοδος viewDataPanel, που απαιτεί την λίστα με τους πίνακες που δημιουργείται στην μέθοδο viewTablesPanel, αν η λίστα δεν έχει δημιουργηθεί ήδη τότε καλεί την μέθοδο viewTablesPanel.

### 7.4 Actions

Η κλάση αυτή υλοποιεί τις ενέργειες που εκτελούνται όταν ο χρήστης αλληλεπιδρά με τα συστατικά γραφικών της εφαρμογής. Η υλοποίηση των λειτουργιών των συστατικών γραφικών έχει αποσυνδεθεί από την κλάση GUIGenerator για λόγους απλότητας του κώδικα. Έτσι η εργασία της GUIGenerator είναι να δημιουργεί το περιβάλλον γραφικής διασύνδεσης χρήστη της εφαρμογής, ενώ η κλάση Actions μεταβάλλει αυτήν την γραφική διασύνδεση και εκτελεί ενέργειες προς την βάση δεδομένων. Αυτό είναι δυνατό με την χρήση ενός αντικειμένου της κλάσης GUIGenerator, το οποίο είναι καθολικά μοναδικό, και ένα αντικείμενο της κλάσης DBConnection.
Οι μέθοδοι της κλάσης είναι:

- NextButtonActionPerformed το οποίο εκτελεί τις απαραίτητες ενέργειες μετά την ολοκλήρωση της συμπλήρωσης των πεδίων για την σύνδεση της εφαρμογής με την βάση δεδομένων, οι οποίες περιλαμβάνουν την δημιουργία αντικειμένου DBConnection με μία από τις υποκλάσεις της DBConnection MySQLConnection και PostgreSQLConnection σύμφωνα με τις τιμές των πεδίων, και την κλήση της μεθόδου της GUIGenerator για την δημιουργία της επόμενης οθόνης επιλογής των συστατικών γραφικών της Java που θα τοποθετηθούν στην εφαρμογή.
- SecondNextButtonActionPerformed που ολοκληρώνει επόμενη διαδικασία η οποία είναι αυτή της επιλογής των στοιχείων γραφικών, για την εκτέλεση ενεργειών στην βάση δεδομένων, που θα τοποθετηθούν στην εφαρμογή.
- DBMSListActionPerformed είναι η μέθοδος που εκτελείται όταν επιλέγεται κάποιο στοιχείο από την λίστα των συστημάτων βάσεων δεδομένων (περιλαμβάνει MySQL και PostgreSQL) από την πρώτη οθόνη για την δημιουργία σύνδεσης με την βαση δεδομένων. Η μέθοδος αυτή συμπληρώνει αυτόματα την προεπιλεγμένη δικτυακή θύρα του κάθε συστήματος στο αντίστοιχο πεδίο.
- selectAllDataButtonActionPerformed η οποία συμπληρώνει το JTable με όλα τα στοιχεία του πίνακα ο οποίος είναι επιλεγμένος.
- dropTableButtonActionPerformed η οποία διαγράφει τον πίνακα ο οποίος είναι επιλεγμένος
- deleteButtonActionPerformed η οποία διαγράφει την εγγραφή του πίνακα η οποία παρουσιάζεται στην επιλεγμένη γραμμή του JTable της εφαρμογής.



`

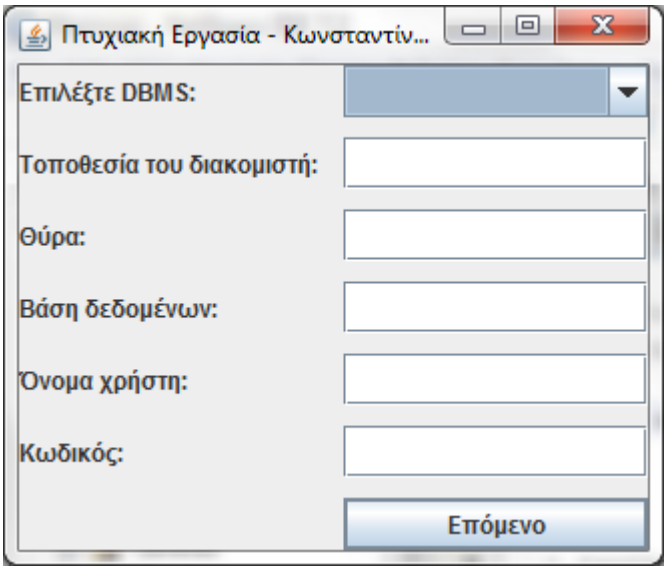
- whereButtonActionPerformed η οποία προσθέτει πεδία για την προσθήκη συνθηκών από τον χρήστη για την επιλογή στοιχείων των πινάκων που αντιστοιχούν στις συνθήκες αυτές. Οι συνθήκες αυτές τοποθετούνται στο πεδίο WHERE μιας πρότασης SQL.
- selectButtonActionPerformed η οποία προετοιμάζει την εκτέλεση ερωτήματος SELECT με συνθήκη προς την βάση δεδομένων.
- clearSelectButtonActionPerformed η οποία αφαιρεί από την εφαρμογή τα πεδία για τις συνθήκες που προστέθηκαν με την whereButtonActionPerformed
- createTableActionPerformed η οποία προετοιμάζει την δημιουργία πίνακα σύμφωνα με τις επιλογές για πεδία ορισμένων από τον χρήστη.
- moreColumnsButtonActionPerformed η οποία προσθέτει νέα πεδία για επιπλέον στήλες στην δημιουργία πίνακα.
- insertButtonActionPerformed η οποία προετοιμάζει την εισαγωγή δεδομένων στον επιλεγμένο πίνακα.
- tablesListValueChanged η οποία αφαιρεί τα παλιά και προσθέτει τα νέα πεδία για την εισαγωγή δεδομένων που αντιστοιχούν στις στήλες του πίνακα όταν ένας νέος πίνακας από την JList που περιέχει τους πίνακες της βάσης δεδομένων επιλέγεται.
- updateButtonActionPerformed η οποία προετοιμάζει την ενημέρωση δεδομένων του επιλεγμένου πίνακα.
- dataTableValueChanged η οποία αφαιρεί τα παλιά και προσθέτει να νέα πεδία που αντιστοιχούν στα στοιχεία της εγγραφής του πίνακα όταν μια εγγραφή από το JTable επιλέγεται.
- getColumns η οποία έχει δύο υλοποιήσεις. Η πρώτη χωρίς όρισμα επιστρέφει τα ονόματα των στηλών του πίνακα που είναι επιλεγμένος από το JList. Η δεύτερη με όρισμα το όνομα του πίνακα, επιστρέφει τα ονόματα των στηλών του πίνακα που ορίζεται.
- getTables η οποία επιστρέφει τα ονόματα των πινάκων που περιέχει η βάση δεδομένων.

### 7.5 Main

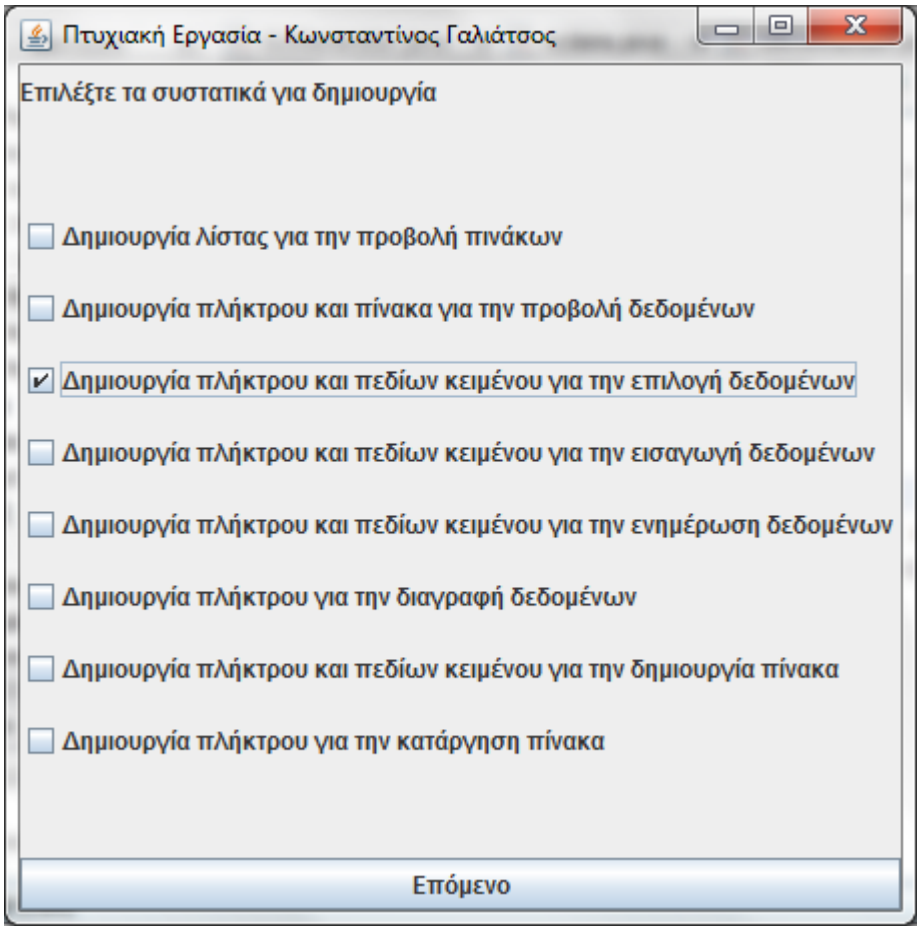
Τέλος η κλάση Main η οποία περιέχει μόνο την main μέθοδο η οποία καλεί την μέθοδο initialize της GUIGenerator η οποία εκκινεί την εφαρμογή.



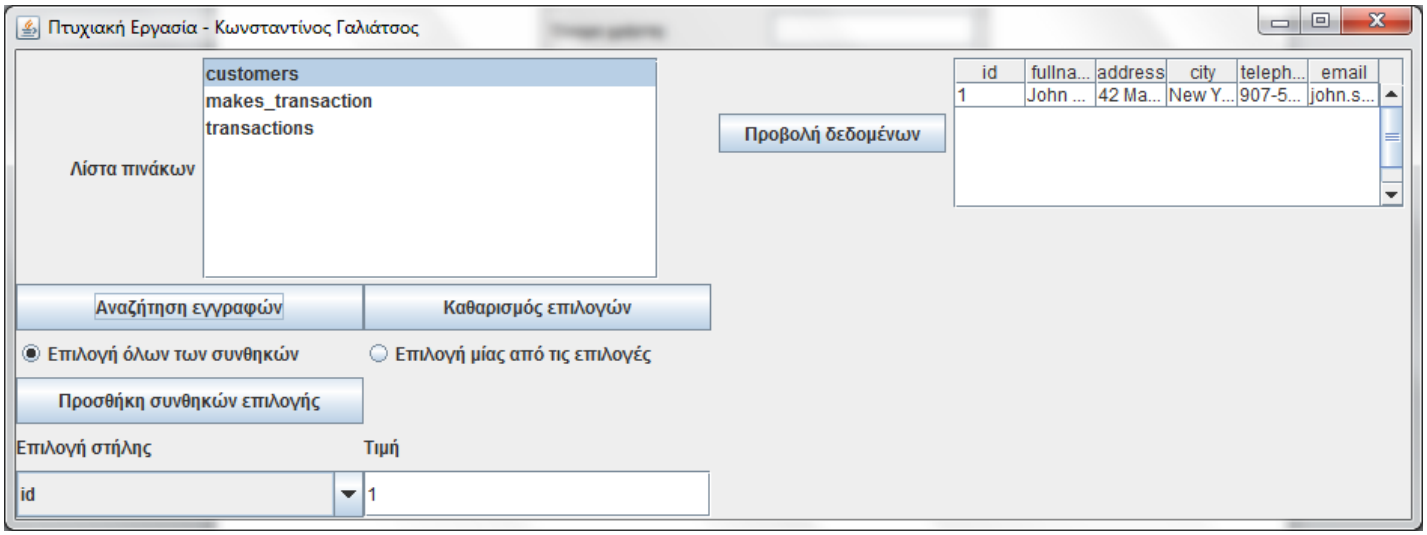
`




Εικόνα 28. Εισαγωγή στοιχείων για σύνδεση με τον διακομιστή




Εικόνα 29. Επιλογή των συστατικών γραφικών διεπαφών χρήστη




Εικόνα 30. Αναζήτηση εγγραφών του πίνακα customers με id=1

## ΕΠΙΛΟΓΟΣ

Για την δημιουργία εφαρμογής με γραφική διεπαφή χρήστη απαιτείται η δημιουργία των απαραίτητων συστατικών γραφικών, η υλοποίηση των λειτουργιών αυτών των συστατικών, όπου στην συγκεκριμένη εφαρμογή υλοποιήθηκαν ξεχωριστά από την δήλωσή τους. Για την διασύνδεση της εφαρμογής με βάση δεδομένων χρειάζονται τα απαραίτητα αντικείμενα και κατάλληλες κλήσεις μεθόδων. Στην συγκεκριμένη περίπτωση επειδή χρησιμοποιήθηκαν δύο συστήματα βάσεων δεδομένων, χρειάστηκε να επεκταθεί η κλάση που διαχειρίζεται την σύνδεση σε δύο νέες κλάσεις που υλοποιούν τα σημεία που διαφέρουν μεταξύ των δύο συστημάτων.



`

## ΣΥΜΠΕΡΑΣΜΑΤΑ

Μια εφαρμογή που συνδέεται σε μια βάση δεδομένων συναντάται αρκετά συχνά, είτε ως αυτόνομη εφαρμογή, είτε ως κομμάτι μιας μεγαλύτερης. Τέτοιες εφαρμογές ποικίλουν από εφαρμογές διαχείρισης πελατολογίου, όπως αυτή που αναπτύχθηκε, μέχρι διαδικτυακές εφαρμογές.

Η σχεδίαση και λειτουργικότητα της εφαρμογής εξερτάται από τις κάθε φορά απαιτήσεις. Στην συγκεκριμένη εφαρμογή υπήρχε για παράδειγμα η δυνατότητα να επεξεργάζεται ο χρήστης τα δεδομένα της βάσης και τα στοιχεία της προβάλονταν όπως ήταν, χωρίς δηλαδή κάποια ιδιαίτερη διαμόρφωση.

Επίσης σε τέτοιου είδους εφαρμογές ο προγραμματιστής πρέπει να μπορεί να αποτρέπει λάθη από τους χρήστες, όπως να γίνεται εισαγωγή αλφαριθμητικού εκεί που ζητείται αριθμός, και αυτό να προλαμβάνεται με μήνυμα προς τον χρήστη και να μην εκτελείται το λανθασμένο SQL query.



`

**ΒΙΒΛΙΟΓΡΑΦΙΑ**

[1] Johannes Gehrke and Raghu Ramakrishnan (2002), Συστήματα Διαχείρισης Βάσεων Δεδομένων, Εκδόσεις Τζιόλα, Τόμος Α, σσ. 99-111, 115, 187-218, 244, 247-248
[2] Αλέξανδρος Ν. Χατζηγεωργίου (2005), Αντικειμενοστρεφής Σχεδίαση UML, Αρχές, Πρότυπα και Ευρετικοί κανόνες, Εκδόσεις Κλειδάριθμος σ. 170
[3] MySQL 5.5 Documentation http://downloads.mysql.com/docs/refman-5.5-en.a4.pdf
[4] The PostgreSQL Global Development Group, PostgreSQL 9.2 Documentation http://www.postgresql.org/docs/9.2/static/index.html
[5] http://www.postgresql.org/about/
[6] http://www.oracle.com/technetwork/java/overview-141217.html
[7] http://netbeans.org/kb/docs/ide/mysql.html
[8] http://help.eclipse.org/indigo/index.jsp
[9] http://java.sun.com/products/jfc/tsc/articles/architecture/
[10] http://docs.oracle.com/javase/1.5.0/docs/guide/jdbc/bridge.html
[11] http://docs.oracle.com/javase/7/docs/api/
[12] Distributed Application Architecture: http://java.sun.com/developer/Books/jdbc/ch07.pdf
[13] http://www.briandunning.com/sample-data/
[14] Oracle Java 6 API: http://docs.oracle.com/javase/6/docs/api/
[15] How to make Frames (main Window) tutorial http://docs.oracle.com/javase/tutorial/uiswing/components/frame.html
[16] How to use labels tutorial http://docs.oracle.com/javase/tutorial/uiswing/components/label.html
[17] How to use Buttons, Check Boxes and Radio Buttons tutorial http://docs.oracle.com/javase/tutorial/uiswing/components/button.html#abstractbutton

## ΠΑΡΑΡΤΗΜΑΤΑ

### Παράρτημα 1

```
import java.sql.*;

public class Mysqlconnect {
   static String driverClassName = "com.mysql.jdbc.Driver";
   static String url = "jdbc:mysql://localhost:3306/test";
   static Connection dbConnection = null;
   static Statement statement = null;
   static ResultSet rs = null;
   public static void main(String[] args) {

   try {

      //  Δήλωση του οδηγού JDBC
      Class.forName(driverClassName);

      //  Δημιουργία της σύνδεσης
      dbConnection = DriverManager.getConnection(url, "kostas", "password");

      //  Εκτέλεση της πρότασης SQL
      statement = dbConnection.createStatement();
      String query = "SELECT * FROM athletes";
      rs = statement.executeQuery(query);

      //  Λήψη του αποτελέσματος
      while(rs.next()) {
      System.out.println(rs.getString(1) + " " + rs.getString(2));
      }

      //  Κλείσιμο της σύνδεσης
      rs.close();
      statement.close();
      dbConnection.close();
   } catch (ClassNotFoundException ex) {
      System.err.println("No such class: " + ex.getMessage());
   } catch (SQLException exc) {
      System.err.println("Couldn't connect to database: " + exc.getMessage());
   }
   }
}
```



`

**Παράρτημα 2**

```
import java.sql.*;

public class MySQLODBC {

    private static Connection con = null;
    private static Statement statement = null;
    private static ResultSet rs = null;
    private static String Driver = "sun.jdbc.odbc.JdbcOdbcDriver";
    private static String url = "jdbc:odbc:MySQLODBC";
    private static String query = null;

    public static void main(String[] args) {
    try {
        Class.forName(Driver);
    }catch(ClassNotFoundException e) {e.printStackTrace();}

    try {
        con = DriverManager.getConnection(url);
        statement = con.createStatement();
        query = "SELECT * FROM athletes";
        rs = statement.executeQuery(query);

        while(rs.next()) {
        System.out.print(rs.getString(1) + " ");
        System.out.println(rs.getString(2));
        }
    }catch(SQLException e) { e.printStackTrace();}

    }
}
```



`

**Παράρτημα 3**

```
CREATE TABLE customers (
id INT NOT NULL PRIMARY KEY,
fullname CHAR(60),
address CHAR(60),
city CHAR(30),
telephone CHAR(12),
email CHAR(40));

CREATE TABLE transactions (
id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
transaction_type CHAR(50));

CREATE TABLE makes_transaction (
id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
c_id INT NOT NULL,
t_id INT NOT NULL,
t_date DATE NOT NULL,
price DOUBLE,
FOREIGN KEY (c_id) REFERENCES customers (id) ON DELETE CASCADE,
FOREIGN KEY (t_id) REFERENCES transactions (id) ON DELETE CASCADE);
```



`

**Παράρτημα 4**

```
CREATE TABLE customers (
id INT NOT NULL PRIMARY KEY,
fullname CHAR(60),
address CHAR(60),
city CHAR(30),
telephone CHAR(12),
email CHAR(40));

CREATE TABLE transactions (
id INT NOT NULL PRIMARY KEY,
transaction_type CHAR(50));

CREATE TABLE makes_transaction (
id INT NOT NULL PRIMARY KEY,
c_id INT NOT NULL,
t_id INT NOT NULL,
t_date DATE NOT NULL,
price FLOAT,
FOREIGN KEY (c_id) REFERENCES customers (id) ON DELETE CASCADE,
FOREIGN KEY (t_id) REFERENCES transactions (id) ON DELETE CASCADE);
```



`

## ΟΔΗΓΟΣ ΧΡΗΣΗΣ ΛΟΓΙΣΜΙΚΟΥ

Με την εκκίνηση της εφαρμογής παρουσιάζεται στον χρήστη η οθόνη για την εισαγωγή των στοιχείων του διακομιστή. Στην λίστα με τα ονόματα των συστημάτων διαχείρισης βάσεων δεδομένων, ο χρήστης επιλέγει το κατάλληλο. Στην συνέχεια εισάγει την τοποθεσία του διακομιστή η οποία μπορεί να είναι είτε διεύθυνση IP είτε το όνομα του διακομιστή. Η θύρα συμπληρώνεται αυτόματα με την επιλογή του διακομιστή στην προεπιλεγμένη θύρα για κάθε σύστημα. Στην συνέχεια εισάγει ο χρήστης το όνομα της βάσης δεδομένων που θέλει να διαχειριστεί στο σύστημα διαχείρισης βάσεων δεδομένων. Τέλος εισάγει το όνομα χρήστη και τον κωδικό του λογαριασμού του στο σύστημα. Επιλέγει το πλήκτρο «Επόμενο» για την μετάβαση στην οθόνη επιλογής συστατικών που επιτρέπουν την διαχείριση της βάσης δεδομένων. Για την συνέχεια στο επόμενο στάδιο είναι απαραίτητη η συμπλήρωση όλων των στοιχείων.

Με την επιλογή του πρώτου κουτιού δημιουργείται μια λίστα με τους διαθέσιμους πίνακες της βάσης δεδομένων. Το δεύτερο κουτί δημιουργεί ένα πλήκτρο και έναν πίνακα. Με το πάτημα του πλήκτρου ο πίνακας συμπληρώνεται με όλα τα στοιχεία του επιλεγμένου πίνακα βάσης δεδομένων που είναι επιλεγμένος από την λίστα πινάκων. Το τρίτο κουτί για την αναζήτηση δεδομένων δημιουργεί τρία πλήκτρα, το ένα προσθέτει δύο radioButton για την επιλογή μιας ή όλων των συνθηκών αναζήτησης, μια λίστα επιλογής της στήλης του πίνακα και ένα πεδίο κειμένου για την προσθήκη συνθήκης αναζήτησης των δεδομένων του πίνακα. Το δεύτερο πλήκτρο αφαιρεί τις προσηγούμενες συνθήκες και συστατικά γραφικών, και το τρίτο πλήκτρο εκτελεί την αναζήτηση επί των δεδομένων. Το επόμενο κουτί για εισαγωγή δεδομένων δημιουργεί label και πεδία κειμένου, όσα οι στήλες του πίνακα για την εισαγωγή δεδομένων στον πίνακα, και ένα πλήκτρο το οποίο εκτελεί την εισαγωγή των στοιχείων. Το πέμπτο κουτί για ενημέρωση δεδομένων δημιουργεί label και πεδία κειμένου, τα οποία δημιουργούνται και συμπληρώνονται με την επιλογή μιας εγγραφής του πίνακα δεδομένων, και ένα πλήκτρο το οποίο εκτελεί την ενημέρωση των δεδομένων. Το έκτο κουτί για διαγραφή δεδομένων δημιουργεί ένα πλήκτρο το οποίο διαγράφει την επιλεγμένη εγγραφή του πίνακα από την βάση δεδομένων. Το έβδομο κουτί για δημιουργία πίνακα δημιουργεί πεδία κειμένου για το όνομα του πίνακα και των στηλών του και λίστες επιλογής του τύπου των στηλών και ένα πλήκτρο για την δημιουργία του πίνακα. Το τελευταίο κουτί δημιουργεί ένα πλήκτρο το οποίο διαγράφει τον επιλεγμένο πίνακα από την λίστα πινάκων από την βάση δεδομένων. Η εφαρμογή συνεχίζει στο επόμενο στάδιο όταν τουλάχιστον ένα κουτί είναι επιλεγμένο. Επίσης κάποιες ενέργειες απαιτούν κάποια συστατικά να υπάρχουν στην εφαρμογή. Αν αυτό συμβαίνει τότε δημιουργείται το απαιτούμενο συστατικό ακόμα και αν δεν είναι επιλεγμένο για δημιουργία από τον χρήστη.



`