

**Ανοικτά Ακαδημαϊκά Μαθήματα**

**Τεχνολογικό Εκπαιδευτικό Ίδρυμα Αθήνας**

**Βάσεις Δεδομένων ΙΙ (Ε)**

**Ενότητα 13:** Επισκόπηση της χρήσης του jdbc API

Χ. Σκουρλάς, Α. Τσολακίδης, Δ. Βάσσης

Τμήμα Μηχανικών Πληροφορικής ΤΕ

|  |  |
| --- | --- |
| Το περιεχόμενο του μαθήματος διατίθεται με άδεια Creative Commons εκτός και αν αναφέρεται διαφορετικά | Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους. |

Περιεχόμενα

[1. Επισκόπηση της χρήσης jdbc API 3](#_Toc401565077)

[2. Ακολουθούν κάποια απλά αλλά λειτουργικά παραδείγματα χρήσης του API: 4](#_Toc401565078)

[3. Σύνδεση σε βάση δεδομένων στην περίπτωση τoυ προϊόντος MySQL 5](#_Toc401565079)

[4. Παραδείγματα διαχείρισης δηλώσεων SQL - Insert-delete-update-select statements 7](#_Toc401565080)

[5. Παράδειγμα χρήσης embedded select statement. 9](#_Toc401565081)

[6. Case Study «Διαχείριση βάσης προσωπικού» 12](#_Toc401565082)

[7. Υλοποίηση εφαρμογής 18](#_Toc401565083)

Στόχος του εργαστηρίου είναι η επισκόπηση της χρήσης του jdbc API για τη δημιουργία εφαρμογών – jsp pages και τη διαχείριση βάσεων δεδομένων. Μετά την επεξεργασία του εργαστηρίου ο ενδιαφερόμενος θα έχει κατανοήσει τα θέματα δημιουργίας μιας εφαρμογής jsp pages.

 **Λέξεις κλειδιά**: jdbc API, netbeans, jsp pages

# Επισκόπηση της χρήσης jdbc API

Στο σχήμα (Martii Leiho) προσφέρεται μία οπτική υπενθύμιση για μία σειρά από σημαντικές κλάσεις, αντικείμενα, μεθόδους (class, objects, methods) του jdbc API και με τα βέλη περιγράφεται η επικοινωνία / σύνδεσή / διάταξή τους.



**Η βασική αρχή χρήσης του jdbc API είναι απλή:**

Ας εστιάσουμε στα βέλη. Για να ορίσω ένα αντικείμενο μιάς κλάσης στην αιχμή ενός βέλους χρησιμοποιώ την κατάλληλη μέθοδο και αντικείμενο της «προηγούμενης» κλάσης που βρίσκεται στην αρχή του ιδίου βέλους.

 Statement myStatement = myconnection.createStatement();

Δηλαδή για να ορίσουμε το αντικείμενο myStatement «εφαρμόζουμε» τη μέθοδο createStatement στο αντικείμενο myconnection.

Τα αντικείμενα της βάσης δεδομένων ορίζονται σαν συμβολοσειρές (string). Ακολουθούν παραδείγματα:

 String testDatabase = “jdbc:mysql://localhost:3306/mydb?user=admin&password=1234";

**Προσοχή! Η παρακάτω εντολή είναι λανθασμένη**

 String sqlString = "INSERT INTO USERS VALUES ('user1', 'password1')";

Η σωστή διαμορφώνεται ως εξής: **… VALUES('"+user1+"','"+password1+"')**

Άρα, **String sql="insert into users values ('"+user1+"','"+password1+"')**"**;**

# Ακολουθούν κάποια απλά αλλά λειτουργικά παραδείγματα χρήσης του API:

Η DriverManagerClass παρέχει τη μέθοδο getConnection. Μπορούμε να ορίσουμε ένα αντικείμενο, για παράδειγμα το αντικείμενο myConnection τύπου Connection ως εξής:

 Connection myConnection = DriverManager.getConnection(testDatabase);

Αυτό προϋποθέτει ότι έχουμε ορίσει ένα αντικείμενο testDatabase τύπου string που «ορίζει» τον driver που χρησιμοποιούμε, το URL, τη βάση μας, username και password. Άρα θα έχουμε:

 String testDatabase = “jdbc:mysql://localhost:3306/mydb?user=admin&password=1234";

 Connection myConnection = DriverManager.getConnection(testDatabase);

**Σημείωση:** Το αντικείμενο myConnection είναι τύπου Connection και μπορούμε να χρησιμοποιήσουμε τη μέθοδο CreateStatement της κλάσης Connection για να ορίσουμε ένα αντικείμενο, για παράδειγμα το αντικείμενο myStatement τύπου Statement. Μέσω του αντικειμένου myStatement θα γράφουμε και θα εκτελούμε δηλώσεις SQL (INSERT, UPDATE, DELETE, SELECT).

 Statement myStatement = myconnection.createStatement();

Το αντικείμενο myStatement είναι τύπου Statement και μπορούμε να το «εκτελέσουμε» χρησιμοποιώντας μία από τις δύο σχετικές μεθόδους της κλάσης δηλαδή τη μέθοδο executeUpdate ή τη μέθοδο executeQuery. Ποια είναι κατάλληλη μέθοδος εξαρτάται από τη δήλωση SQL. Στην περίπτωση δήλωσης INSERT έχουμε:

 myStatement.executeUpdate(sqlString);

Η συγγραφή της SQL δήλωσης γίνεται σε ένα αντικείμενο τύπου string, για παράδειγμα στο αντικείμενο με όνομα sqlString.

 String sqlString = "INSERT INTO USERS VALUES **('"+user1+"','"+password1+"')** ~~('user1', 'password1')~~";

Έτσι για να εκτελέσουμε μία SQL δήλωση:

 **String sql="insert into users values ('"+user1+"','"+password1+"')**"**;**

 myStatement.executeUpdate(sqlString);

Ειδικά στην περίπτωση δήλωσης SELECT μπορούμε να ορίσουμε ένα αντικείμενο, για παράδειγμα το αντικείμενο με όνομα rs, τύπου ResultSet. Στο αντικείμενο «αποθηκεύονται» τα αποτελέσματα της δήλωσης SELECT. Προηγουμένως, θα χρειασθεί να ορίσουμε το αντικείμενο sqlString που είναι μια συμβολοσειρά που «περιέχει» τη δήλωση SELECT:

String sqlString = "SELECT \* FROM MYTABLE WHERE PARAM = '"+paramValue+"'";

ResultSet rs = myStatement.executeQuery(sqlString);

Η κλάση ResultSet παρέχει μεθόδους για να διαχειριστούμε τις γραμμές που «αποθηκεύονται» στο rs:

 next(), previous(), first(), last()

Επίσης, παρέχει μία σειρά από μεθόδους κάθε μία από τις οποίες μπορεί να χρησιμοποιηθεί για να πάρουμε πληροφορία από μια στήλη:

 getString, getInt, getFloat getDate.

# Σύνδεση σε βάση δεδομένων στην περίπτωση τoυ προϊόντος MySQL

Για να συνδεθούμε στη βάση δεδομένων στην περίπτωση τoυ προϊόντος MySQL και να γράψουμε δηλώσεις SQL (SQL statements) απαιτούνται οι εξής ενέργειες:

1. **Δήλωση του JDBC driver**

libraries>δεξί κλικ> Add Jar/Folder και επιλέγουμε το αρχείο mysql-connector-java- 5.1.13 -bin.jar ή mysql.jar

1. **Εισαγωγή του package java.sql στο πρόγραμμά μας**

<%@page import="java.sql.\*" %> (χρησιμοποιούμε .\*, ;έτσι ώστε να μπορούμε να χρησιμοποιήσουμε όλες τις κλάσεις του ***package java.sql***)

1. ***«*Φόρτωση» του JDBC driver στο πρόγραμμά μας**

Class.forName(“com.mysql.jdbc.Driver”);

// Φορτώνουμε την κλάση Driver από το package com.mysql.jdbc.

1. **Σύνδεση στη βάση δεδομένων**

String testDatabase = “jdbc:mysql://localhost:3306/mydb?user=admin&password=1234";

// Ορίζουμε για τη βάση δεδομένων μας (‘εστω ότι την ονομάζουμε mydb) ένα αντικείμενο testDatabase τύπου συμβολοσειράς (string) της μορφής: “jdbc:mysql://ServerName:ServerPort/DatabaseName?user=myUsername&password=myPassword";

Connection myConnection = DriverManager.getConnection(testDatabase);

// Δημιουργούμε ένα αντικείμενο τύπου *Connection.* Tο αντικείμενό μας, myConnection, είναι μια ανοιχτή σύνδεση με τη βάση δεδομένων

1. **Δημιουργία αντικειμένου για να γράψουμε SQL statements**

Statement myStatement = myconnection.createStatement();

// Δημιουργούμε το αντικείμενο myStatement τύπου Statement. Μέσω του αντικειμένου myStatement θα γράφουμε δηλώσεις SQL.

1. **Εγγραφή SQL statements**

 Λανθασμένη εντολή

~~String sqlString = "INSERT INTO USERS VALUES ('user1', 'password1')";~~

Σωστή εντολή

**String sql="insert into users values ('"+user1+"','"+password1+"')**"**;**

// Ορίζουμε ένα αντικείμενο sqlString τύπου string για να γράψουμε τη δήλωση SQL (που θέλουμε να εκτελεστεί στη συνέχεια).

1. **Εκτέλεση SQL statements**

myStatement.executeUpdate(sqlString);

// Εκτελούμε τη δήλωση SQL που γράψαμε και πιο συγκεκριμένα χρησιμοποιούμε το αντικείμενο myStatement που δημιουργήσαμε. Επειδή έχουμε στην περίπτωση αυτή εντολή INSERT χρησιμοποιήσαμε τη μέθοδο executeUpdate. Το ίδιο θα κάναμε και σε δήλωση UPDATE ή DELETE.

Αν η sqlString αφορούσε μια SQL statement τύπου select, η μέθοδός μας θα ήταν:η executeQuery:

 myStatement.executeQuery(sqlString);

1. **Κλείσιμο του αντικειμένου «αναγραφής» SQL statements**

myStatement.close();

// Αφού εκτελέσουμε την SQL statement πρέπει να κλείσουμε το αντικείμενο που χρησιμοποιείται για την εγγραφή της SQL statements.

1. **Κλείσιμο της σύνδεσης**

myConnection.close();

// Κλείνουμε τη σύνδεση με τη βάση δεδομένων.

# Παραδείγματα διαχείρισης δηλώσεων SQL - Insert-delete-update-select statements

Για να γράψουμε μια εντολή SQL, χρησιμοποιούμε τη μέθοδο

myStatement.methodName(sqlString);

όπου myStatement είναι ένα αντικείμενο τύπου Statement και methodName είναι μια από τις μεθόδους που παρέχει η κλάση Statement.

Θα μελετήσουμε διάφορες μορφές της μεθόδου methodName.

**Στη συνέχεια, διορθώστε το διαγραμμένο μέρος των δηλώσεων.**

 **Εισαγωγή δεδομένων (insert statement)**

Χρησιμοποιούμε τη μέθοδο executeUpdate μέσω της εντολής:

myStatement.executeUpdate(sqlString);

όπου sqlString είναι μια συμβολοσειρά της μορφής:

String sqlString = "INSERT INTO MYTABLE VALUES ~~('value1', 'value2'~~)";

**Διαγραφή δεδομένων (delete statement)**

Χρησιμοποιούμε τη μέθοδο executeUpdate μέσω της εντολής:

myStatement.executeUpdate(sqlString);

όπου το sqlString είναι της μορφής:

String sqlString = "DELETE FROM MYTABLE WHERE PARAM = ~~'paramValue~~'";

**Ενημέρωση δεδομένων (update statement)**

Χρησιμοποιούμε τη μέθοδο executeUpdate μέσω της εντολής:

myStatement.executeUpdate(sqlString);

όπου το sqlString είναι της μορφής:

String sqlString = "UPDATE MYTABLE SET PARAM1=~~'value1'~~ WHERE PARAM2='~~value2~~'";

**Επιλογή δεδομένων (select statement)**

Για επιλογή δεδομένων χρησιμοποιούμε τη μέθοδο executeQuery μέσω της

εντολής:

ResultSet rs = myStatement.executeQuery(sqlString);

όπου sqlString είναι μια συμβολοσειρά της μορφής:

String sqlString = "SELECT \* FROM MYTABLE WHERE PARAM = '~~paramValue~~'";

Η μέθοδος *executeQuery* μας δίνει τη δυνατότητα να ανακτήσουμε γραμμές δεδομένων από τη βάση δεδομένα. Τα δεδομένα αυτά πρέπει να τα αποθηκεύσουμε προσωρινά σε κάποια δομή ώστε να τα χρησιμοποιήσουμε.

Η Java παρέχει μια εύχρηστη δομή μέσω της κλάσης *ResultSet*. Πιο συγκεκριμένα, η μέθοδος *ExecuteQuery* μας επιστρέφει ένα αντικείμενο τύπου *ResultSet*. Ας ονομάσουμε το συγκεκριμένο αντικείμενο *rs*. Μέσα στο αντικείμενο αυτό αποθηκεύονται προσωρινά όλες οι γραμμές δεδομένων που επιστρέφει η δήλωση SELECT που εκτελέσαμε. Μέσω του αντικειμένου αυτού μπορούμε να επεξεργαστούμε τις γραμμές στο πρόγραμμά μας

Το αντικείμενο *rs* μας παρέχει και έναν δείκτη ο οποίος αρχικά δείχνει πρίν από την πρώτη γραμμή αποτελεσμάτων που είναι αποθηκευμένα στο αντικείμενο *rs*.

**Το rs (ως αντικείμενο της κλάσης ResultSet) μας παρέχει μεθόδους για να επεξεργαζόμαστε τις γραμμές που ανακτήσαμε από τη βάση δεδομένων. Οι πιο σημαντικές μέθοδοι αναφέρονται παρακάτω:**

Για να χρησιμοποιήσουμε την πληροφορία που είναι καταχωρημένη στο αντικείμενο rs, θα πρέπει:

* Να μετακινήσουμε το δείκτη στη γραμμή που θέλουμε να επεξεργασθούμε ή να τον μετακινούμε διαδοχικά αν θέλουμε να επεξεργασθούμε όλες τις αποθηκευμένες γραμμές. Αυτό γίνεται με χρήση των μεθόδων: **next(), previous(), first(), last()**
* Να χρησιμοποιήσουμε την κατάλληλη μέθοδο για να πάρουμε πληροφορία από μια στήλη. Αυτό γίνεται με τις συναρτήσεις **getString, getInt, getFloat getDate** κ.τ.λ.

**Πρακτικός κανόνας για τη συγγραφή δήλωσης SQL στα προγράμματά μας:**

Δοκιμάζουμε τη δήλωση με συγκεκριμένες τιμές στις μεταβλητές.

Τοποθετούμε την εντολή σε " " και αντικαθιστούμε κάθε συγκεκριμένη τιμή, για παράδειγμα την τιμή 'ANALYST'με το όνομα της μεταβλητής γραμμένο με "+ +", ως εξής: '"+job+"'

Δείτε και το παράδειγμα:

String sql="insert into emp(ename,job,sal,deptno) values ('"+name+"','"+job+"','"+sal+"',"+deptno+")";

# Παράδειγμα χρήσης embedded select statement.

Ακολουθεί παράδειγμα χρήσης της executeQuery που βοηθά στην κατανόηση του τρόπου που χρησιμοποιούμε τις διάφορες μεθόδους της κλάσης ResultSet.

Έστω ότι έχουμε τον παραπάνω πίνακα myTable:

myTable,

|  |  |
| --- | --- |
| ID | *Int* |
| Name | *Text* |

select \* from myTable



Δημιουργείστε ένα αρχείο test.jsp και γράψτε τον παρακάτω κώδικα

<%--

 Document : test

 Created on : April 31, 2014, 8:24:04 AM

 Author :

--%>

// Import the package java.sql – all the classes

<%@page import="java.sql.\*" %>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<%

int j=0;

// Define parameters for retrieving database results

int m[] = new int[5000];

String s[] = new String[5000];

// Load the driver

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

// Define the connection parameters

String myDatabase = "jdbc:mysql://localhost:3306/personnel?user=root";

// Connect to the database

Connection myConnection = DriverManager.getConnection(myDatabase);

// Create object to execute statements

Statement myStatement = myConnection.createStatement();

// Select everything from the database table

String sqlString = "SELECT \* FROM myTable ";

ResultSet rs=myStatement.executeQuery(sqlString);

// Store the results in vectors in order to pass them to the JSP

while(rs.next()){

m[j]=rs.getInt("id");

s[j]=rs.getString("name");

j++;

}

// Close the connection to the database

myStatement.close();

myConnection.close();

// Print

out.println("j="+j+"<br>");

for(int i=0; i<j; i++)

{

out.println(m[i]+"");

out.println(s[i]+"<br>");

}

%>

**Στη συνέχεια τρέξτε το πρόγραμμα.**

j=3
1 1
1 ann
1 tom

**Σύνοψη – Επισκόπηση**

Αρχικά βλέπουμε τον πίνακα myTable. Τα βήματα είναι τα παρακάτω:

|  |  |
| --- | --- |
|  ID | *Int* |
| Name | *Text* |

myTable



1. Oρίζουμε δυο παραμέτρους.
2. Στις παραμέτρους αυτές καταχωρούμε τις τιμές των δυο στηλών του πίνακα mytable.
3. Φορτώνουμε το driver και προετοιμάζουμε τη συμβολοσειρά για σύνδεση.
4. Συνδεόμαστε, στην περίπτωσή μας, με username root και κενό password, στη βάση δεδομένων mydb.
5. Συνδεόμαστε στη βάση δεδομένων, personnel, δημιουργούμε το αντικείμενο τύπου statement και προετοιμάζουμε την SQL string.
6. Εκτελούμε το SQL Statement.
7. Μέχρι στιγμής έχουμε καταχωρήσει όλον τον πίνακα αποτελεσμάτων στο αντικείμενο rs. Μένει να «διαβάσουμε» τα στοιχεία του rs. Αυτό γίνεται με έναν βρόχο while.
8. Η εντολή while(rs.next()) έχει διπλό ρόλο:Εκτελεί την εντολή rs.next(). Επιπλέον, ελέγχει αν rs.next()=true. Αυτό σημαίνει ότι, αν rs.next()=true, τότε ο βρόχος θα εκτελεστεί. Αν rs.next()=false τότε ο βρόχος δε θα εκτελεστεί και το πρόγραμμα θα προχωρήσει παρακάτω. Η μέθοδος next() της resultSet επιστρέφει true αν υπάρχει επόμενη γραμμή για να διαβάσει. Αν δεν υπάρχει επόμενη γραμμή, δηλαδή έχουμε φτάσει στο τέλος των αποτελεσμάτων, τότε επιστρέφει false. Με τον τρόπο αυτό, αν υπάρχει επόμενη γραμμή, ο βρόχος εκτελείται, και σταματά να εκτελείται όταν οι γραμμές «τελειώσουν». Ο βρόχος κάνει το εξής:
	1. Διαβάζει, τις στήλες της τρέχουσας γραμμής και τις καταχωρεί στις αντίστοιχες μεταβλητές του προγράμματος (m[j], (m[j],), μέσω της μεθόδου getString() ή getInt().
	2. Τυπώνει στην οθόνη τις παραπάνω μεταβλητές.

# Case Study «Διαχείριση βάσης προσωπικού»

Σας αναθέτουν να σχεδιάσετε και μα υλοποιήσετε εφαρμογή η οποία αφορά τη διαχείριση του προσωπικού εταιρείας. Στην εφαρμογή θα έχουν πρόσβαση μόνο όσοι χρήστες είναι εγγεγραμμένοι και διαθέτουν username –password. Αρχικά θα εμφανίζεται στο χρήστη μια login φόρμα όπου θα δίνεται η δυνατότητα στον χρήστη να συνδεθεί. Αφού συνδεθεί ο χρήστης θα έχει τις παρακάτω δυνατότητες:

1. Προβολή όλων των τμημάτων της εταιρείας μαζί με τα στοιχεία εργαζομένων που ανήκουν σε αυτά.
2. Εισαγωγή ενός νέου εργαζόμενου.
3. Διαγραφή ενός εργαζόμενου
4. Ενημέρωση των στοιχείων των εργαζομένων.

Οι πίνακες της εφαρμογής θα είναι οι παρακάτω:



DROP DATABASE personnel\_db;

CREATE DATABASE personnel;

USE personnel;

CREATE TABLE Dept(DEPTNO INT(2) NOT NULL,

 DNAME VARCHAR(14), LOC VARCHAR(14),

 NO\_OF\_EMPLOYEES INT(3),

 PRIMARY KEY(DEPTNO));

CREATE TABLE Emp(EMPNO INT(4) NOT NULL AUTO\_INCREMENT,

 ENAME VARCHAR(10), JOB VARCHAR(9),

 SAL FLOAT(7,2),

 DEPTNO INT(2), PRIMARY KEY(EMPNO),

 FOREIGN KEY(DEPTNO) REFERENCES Dept(DEPTNO));

INSERT INTO Dept(DEPTNO, DNAME, LOC)

 VALUES (10, 'ACCOUNTING', 'NEW YORK');

INSERT INTO Dept(DEPTNO, DNAME, LOC)

 VALUES (20, 'RESEARCH', 'DALLAS');

INSERT INTO Dept(DEPTNO, DNAME, LOC)

 VALUES (30, 'SALES', 'CHICAGO');

INSERT INTO Dept(DEPTNO, DNAME, LOC)

 VALUES (40, 'OPERATIONS', 'BOSTON');

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'SMITH', 'CLERK', 800, 20);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'ALLEN', 'SALESMAN', 1600, 30);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'WARD', 'SALESMAN', 1250, 30);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'JONES', 'MANAGER', 2975, 20);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'MARTIN', 'SALESMAN', 1250, 30);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'BLAKE', 'MANAGER', 2850, 30);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'CLARK', 'MANAGER', 2450, 10);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'SCOTT', 'ANALYST', 3000, 20);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'KING', 'PRESIDENT', 5000, 10);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'TURNER', 'SALESMAN', 1500, 30);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'ADAMS', 'CLERK', 1100, 20);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'JAMES', 'CLERK', 950, 30);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'FORD', 'ANALYST', 3000, 20);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'MILLER', 'CLERK', 1300, 10);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'BATES', 'ANALYST', 1300, NULL);

UPDATE dept

SET no\_of\_employees =

 (SELECT COUNT(\*)

 FROM emp

 WHERE emp.deptno = dept.deptno);

Select \* from dept;

Select \* from emp;



CREATE TABLE user(

 uname text,

 upass text,

 Uid int(11),

 Uphone varchar(45), Ucity varchar(45));

INSERT INTO `user` VALUES ('admin','1234',1,NULL,NULL);

Select \* from user;



**Χρήση Cursor**

DELIMITER $$

DROP PROCEDURE IF EXISTS CursorProc$$

CREATE PROCEDURE CursorProc()

BEGIN

DECLARE no\_more\_depts, available\_employees INT DEFAULT 0;

DECLARE dept\_code VARCHAR(255);

DECLARE cur\_dept CURSOR FOR

 SELECT deptno FROM dept;

 DECLARE CONTINUE HANDLER FOR NOT FOUND

 SET no\_more\_depts = 1;

 /\* for loggging information \*/

CREATE TABLE infologs (

Id int(11) NOT NULL AUTO\_INCREMENT,

Msg varchar(255) NOT NULL, PRIMARY KEY (Id));

OPEN cur\_dept;

FETCH cur\_dept INTO dept\_code;

 REPEAT

 SELECT no\_of\_employees INTO available\_employees

 FROM dept

 WHERE deptno = dept\_code;

 IF available\_employees < 5 THEN

 INSERT INTO infologs(msg)VALUES (dept\_code);

 END IF;

 FETCH cur\_dept INTO dept\_code;

 UNTIL no\_more\_depts = 1

 END REPEAT;

CLOSE cur\_dept;

 SELECT \* FROM infologs;

 DROP TABLE infologs;

END$$

DELIMITER ;

CALL CURSORPROC();



# Υλοποίηση εφαρμογής

Τα στοιχεία της σύνδεσης είναι:

 Ip:localhost

 Port:3306

 Database: personnel

 User=root

 Password=

*Υπενθυμίζουμε ότι ο χρήστης (βλέπε και πίνακα user) έχει:*

 User=root

 Password=1234

**Στάδια:**

1. **Διαχείριση Σύνδεσης**

Υλοποιείται με χρήση δύο (2) σελίδων: index.jsp, check.jsp

Index.jsp



Αν δοθούν λανθασμένα στοιχεία τότε



<%--

 Document : index

 Created on : April 30, 2014, 2:04:01 PM

 Author :

--%>

**<%@page contentType="text/html" pageEncoding="UTF-8"%>**

**<!DOCTYPE html>**

**<html>**

 **<head>**

 **<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">**

 **<title>JSP Page</title>**

 **</head>**

 **<body>**

 **WELCOME<p></p>**

 **LOGIN<p></p>**

 **<p></p>**

 **<form name="formName" method="post" action="check.jsp" >**

 **Name: <input type="text" name="y"><p></p>**

 **Password:<input type="text" name="k"><P></P>**

 **<input type="submit" value="LOGIN"><p></p>**

 **</form>**

 **</body>**

**</html>**

**<%--**

 **Document : check**

 **Created on : April 30, 2014, 2:12:02 PM**

 **Author :**

**--%>**

**<%@page import="java.sql.\*" %>**

**<%@page contentType="text/html" pageEncoding="UTF-8"%>**

**<!DOCTYPE html>**

**<html>**

 **<head>**

 **<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">**

 **<title>JSP Page</title>**

 **</head>**

 **<body>**

 **<%**

 **Boolean found;**

 **String x =request.getParameter("y");**

 **String p = request.getParameter("k");**

 **String URL;**

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

 **String DB = "jdbc:mysql://localhost:3306/personnel?user=root";**

 **Connection myConnection = DriverManager.getConnection(DB);**

 **Statement SMT = myConnection.createStatement();**

 **String sql="SELECT \* FROM user WHERE Uname='"+x+"'AND Upass='"+p+"' ";**

 **ResultSet rs=SMT.executeQuery(sql);**

 **found= rs.first();**

 **if (found){**

 **URL = "home.jsp?p1="+x+"";**

 **response.sendRedirect(URL);**

 **}**

 **else**

 **{**

 **out.println("INCORRECT TRY AGAIN");%><P></P>**

 **<a href="index.jsp">Try Again</a><%**

 **}**

 **SMT.close();**

 **myConnection.close();**

 **%>**

 **</body>**

**</html>**

**Σημαντική σημείωση**

Έστω ότι τα στοιχεία της σύνδεσης είναι:

 Ip:localhost

 Port:3306

 Database: personnel

 User=root

 Password=1234

Τότε αντί της εντολής

 String DB = "jdbc:mysql://localhost:3306/personnel?user=root";

Χρησιμοποιούμε την εντολή:

 String DB = "jdbc:mysql://localhost:3306/personnel\_db?user=root&password=1234";

1. **Προβολή όλων των εργαζομένων και των τμημάτων στα οποία ανήκουν.**

Υλοποιείται με χρήση της σελίδας (home.jsp)



κ.λπ.<%--

 Document : home

 Created on : April 30, 2014, 2:26:58 PM

 Author :

--%>

<%@page import="java.sql.\*" %>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>

 <%

 Integer empno[]=new Integer[100];

 String name[]=new String[100];

 String job[]=new String[100];

int i;

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

 String DB = "jdbc:mysql://localhost:3306/personnel?user=root";

 Connection myConnection = DriverManager.getConnection(DB);

 Statement SMT = myConnection.createStatement();

 Statement SMT1 = myConnection.createStatement();

 %>

<html>

 <head>

 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

 <title>JSP Page</title>

 </head>

 <body>

 <%

 String sql="SELECT \* FROM dept ";

 ResultSet rs=SMT.executeQuery(sql);

 String dname;

 int count=0;

 while (rs.next())

 {

 i=rs.getInt("deptno");

 String sql1="SELECT \* FROM emp WHERE deptno='"+i+"' ";

 ResultSet rs1=SMT1.executeQuery(sql1);

 dname=rs.getString("dname");

 %>

 <br> <INPUT TYPE="BUTTON" VALUE="ΠΡΟΣΘΗΚΗ ΕΝΟΣ ΝΕΟΥ ΕΡΓΑΖΟΜΕΝΟΥ" style="font-size:7px"

 ONCLICK="window.location.href='insert.jsp?deptno=<%= i%>'">

 <%

 out.println(" <br>Oι εργαζόμενοι στο Τμήμα:"+dname+" είναι οι:");

 while(rs1.next())

 {

 name[count]=rs1.getString("ename");

 job[count]=rs1.getString("job");

 empno[count]=rs1.getInt("empno");

 %> <br>

 <%= name[count]%> (<%= job[count]%>)

 <INPUT TYPE="BUTTON" VALUE="ΑΛΛΑΓΗ" style="font-size:7px"

 ONCLICK="window.location.href='update.jsp?empno=<%= empno[count]%>'">

 <INPUT TYPE="BUTTON" VALUE="ΔΙΑΓΡΑΦΗ " style="font-size:7px"

 ONCLICK="window.location.href='delete.jsp?empno=<%= empno[count]%>'">

 <br>

 <% String x="o";

 }}%>

 <%

 SMT.close();

 SMT1.close();

 myConnection.close();

 %>

 </body>

</html>

1. **Εισαγωγή Νέου εργαζόμενου (insert.jsp, ins\_submit.jsp)**



Για παράδειγμα



1. **Ενημέρωση στοιχείων εργαζομένων (update.jsp, up\_submit.jsp)**

Αν συνδεθούμε εκ νέου θα δούμε



κ.λπ.

Επιλέγουμε ΑΛΛΑΓΗ για τον υπάλληλο ANDREW.



Αλλάζουμε τα στοιχεία



Επιλέγουμε update οπότε ενημερώνεται η γραμμή και επιστρέφουμε στην αρχική φόρμα:



1. **Διαγραφή ενός εργαζομένου (delete.jsp)**

Αν συνδεθούμε εκ νέου θα δούμε



κ.λπ.

Αν επιλέξουμε ΔΙΑΓΡΑΦΗ για τον υπάλληλο ANDREW διαγράφεται ο υπάλληλος.

**Σημείωση:** Θα έπρεπε να βλέπαμε όλα τα στοιχεία του πριν τον διαγράψουμε. Επιπλέον, θα έπρεπε να υπάρχει και επιβεβαίωση πριν από τη διαγραφή.

**Γενικά θα μπορούσαμε να βελτιώσουμε σε πολλά σημεία την εφαρμογή με σχετικά απλό τρόπο!** <%--

 Document : insert

 Created on : April 30, 2014, 2:32:55 PM

 Author :

--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>

 <html>

 <head>

 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

 <title>JSP Page</title>

 </head>

 <body>

 <form name="insert" method="get" action="ins\_submit.jsp">

 Name<input type="text" name="ename"><br>

 Job<input type="text" name="job"><br>

 Salary<input type="text" name="sal"><br>

 <input type="hidden" name="deptno" value="<%= request.getParameter("deptno") %>">

 <input type="submit" value="insert">

 </form>

 </body>

</html>

<%--

 Document : ins\_submit

 Created on : April 30, 2014, 2:34:18 PM

 Author :

--%>

<%@page import="java.sql.\*"%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html>

 <head>

 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

 <title>JSP Page</title>

 </head>

 <body>

 <%

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

 String DB = "jdbc:mysql://localhost:3306/personnel?user=root";

 Connection myConnection = DriverManager.getConnection(DB);

 Statement SMT = myConnection.createStatement();

 String name=request.getParameter("ename");

 String job=request.getParameter("job");

 String sal=request.getParameter("sal");

 int deptno=Integer.parseInt(request.getParameter("deptno"));

 String sql="insert into emp(ename,job,sal,deptno) values ('"+name+"','"+job+"','"+sal+"',"+deptno+")";

 SMT.executeUpdate(sql);

 String URL = "index.jsp";

 response.sendRedirect(URL);

 %>

 <%

 SMT.close();

 myConnection.close();

 %>

 </body>

</html><%--

 Document : update

 Created on : April 30, 2014, 2:35:38 PM

 Author :

--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<%@page import="java.sql.\*"%>

 <html>

 <head>

 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

 <title>JSP Page</title>

 </head>

 <body>

 <%

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

 String DB = "jdbc:mysql://localhost:3306/personnel?user=root";

 Connection myConnection = DriverManager.getConnection(DB);

 Statement SMT = myConnection.createStatement();

 String job="",sal="",ename="";

 int empno=Integer.parseInt(request.getParameter("empno"));

 String sql="select \* from emp where empno="+empno;

 ResultSet rs= SMT.executeQuery(sql);

 while(rs.next()){

 job=rs.getString("job");

 sal=rs.getString("sal");

 ename=rs.getString("ename");

 }

 %>

 <form name="insert" method="get" action="up\_submit.jsp">

 Name<input type="text" name="ename" value="<%= ename%>"><br>

 Job<input type="text" name="job"value="<%= job%>"><br>

 Salary<input type="text" name="sal" value="<%= sal%>"><br>

 <input type="hidden" name="empno" value="<%= empno%>">

 <input type="submit" value="update">

 </form>

 </body>

</html>

<%--

 Document : up\_submit

 Created on : April 30, 2014, 2:37:10 PM

 Author :

--%>

<%@page import="java.sql.\*"%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html>

 <head>

 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

 <title>JSP Page</title>

 </head>

 <body>

 <%

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

 String DB = "jdbc:mysql://localhost:3306/personnel?user=root";

 Connection myConnection = DriverManager.getConnection(DB);

 Statement SMT = myConnection.createStatement();

 String name=request.getParameter("ename");

 String job=request.getParameter("job");

 String sal=request.getParameter("sal");

 int empno=Integer.parseInt(request.getParameter("empno"));

 String sql="update emp set ename ='"+name+"' where empno="+empno;

 SMT.executeUpdate(sql);

 sql="update emp set job ='"+job+"' where empno="+empno;

 SMT.executeUpdate(sql);

 sql="update emp set sal ='"+sal+"' where empno="+empno;

 SMT.executeUpdate(sql);

 String URL = "index.jsp";

 response.sendRedirect(URL);

 SMT.close();

 myConnection.close();

 %>

 </body>

</html>

<%--

 Document : delete

 Created on : April 30, 2014, 2:38:19 PM

 Author :

--%>

<%@page import="java.sql.\*"%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html>

 <head>

 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

 <title>JSP Page</title>

 </head>

 <body>

 <%

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

 String DB = "jdbc:mysql://localhost:3306/personnel?user=root";

 Connection myConnection = DriverManager.getConnection(DB);

 Statement SMT = myConnection.createStatement();

 int empno=Integer.parseInt(request.getParameter("empno"));

 String sql="delete from emp where empno="+empno;

 SMT.executeUpdate(sql);

 String URL = "index.jsp";

 response.sendRedirect(URL);

 SMT.close();

 myConnection.close();

 %>

 </body>

</html>

**Δημιουργία βάσης δεδομένων**

CREATE DATABASE personnel;

USE personnel;

CREATE TABLE Dept(DEPTNO INT(2) NOT NULL,

 DNAME VARCHAR(14), LOC VARCHAR(14),

 NO\_OF\_EMPLOYEES INT(3),

 PRIMARY KEY(DEPTNO));

CREATE TABLE Emp(EMPNO INT(4) NOT NULL AUTO\_INCREMENT,

 ENAME VARCHAR(10), JOB VARCHAR(9),

 SAL FLOAT(7,2),

 DEPTNO INT(2), PRIMARY KEY(EMPNO),

 FOREIGN KEY(DEPTNO) REFERENCES Dept(DEPTNO));

INSERT INTO Dept(DEPTNO, DNAME, LOC)

 VALUES (10, 'ACCOUNTING', 'NEW YORK');

INSERT INTO Dept(DEPTNO, DNAME, LOC)

 VALUES (20, 'RESEARCH', 'DALLAS');

INSERT INTO Dept(DEPTNO, DNAME, LOC)

 VALUES (30, 'SALES', 'CHICAGO');

INSERT INTO Dept(DEPTNO, DNAME, LOC)

 VALUES (40, 'OPERATIONS', 'BOSTON');

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'SMITH', 'CLERK', 800, 20);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'ALLEN', 'SALESMAN', 1600, 30);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'WARD', 'SALESMAN', 1250, 30);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'JONES', 'MANAGER', 2975, 20);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'MARTIN', 'SALESMAN', 1250, 30);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'BLAKE', 'MANAGER', 2850, 30);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'CLARK', 'MANAGER', 2450, 10);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'SCOTT', 'ANALYST', 3000, 20);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'KING', 'PRESIDENT', 5000, 10);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'TURNER', 'SALESMAN', 1500, 30);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'ADAMS', 'CLERK', 1100, 20);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'JAMES', 'CLERK', 950, 30);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'FORD', 'ANALYST', 3000, 20);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'MILLER', 'CLERK', 1300, 10);

INSERT INTO Emp( ENAME, JOB, SAL, DEPTNO) VALUES ( 'BATES', 'ANALYST', 1300, NULL);

UPDATE dept

SET no\_of\_employees =

 (SELECT COUNT(\*)

 FROM emp

 WHERE emp.deptno = dept.deptno);

Select \* from dept;

Select \* from emp;

CREATE TABLE user(

 uname text,

 upass text,

 Uid int(11),

 Uphone varchar(45), Ucity varchar(45));

INSERT INTO `user` VALUES ('admin','1234',1,NULL,NULL);

Select \* from user;

|  |
| --- |
| **Ανοικτά Ακαδημαϊκά Μαθήματα****Τεχνολογικό Εκπαιδευτικό Ίδρυμα Αθήνας** |
| **Τέλος Ενότητας** |
| **Χρηματοδότηση*** Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
* Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο ΤΕΙ Αθήνας**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
* Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

 |

**Σημειώματα**

**Σημείωμα Αναφοράς**

Copyright ΤΕΙ Αθήνας, Χ. Σκουρλάς, Α. Τσολακίδης, Δ. Βάσσης 2014. Χ. Σκουρλάς, Α. Τσολακίδης, Δ. Βάσσης. «Βάσεις Δεδομένων ΙΙ (Ε). Ενότητα 13: Επισκόπηση της χρήσης του jdbc API». Έκδοση: 1.0. Αθήνα 2014. Διαθέσιμο από τη δικτυακή διεύθυνση: [ocp.teiath.gr](https://ocp.teiath.gr/).

**Σημείωμα Αδειοδότησης**

Το παρόν υλικό διατίθεται με τους όρους της άδειας χρήσης Creative Commons Αναφορά, Μη Εμπορική Χρήση Παρόμοια Διανομή 4.0 [1] ή μεταγενέστερη, Διεθνής Έκδοση. Εξαιρούνται τα αυτοτελή έργα τρίτων π.χ. φωτογραφίες, διαγράμματα κ.λ.π., τα οποία εμπεριέχονται σε αυτό. Οι όροι χρήσης των έργων τρίτων επεξηγούνται στη διαφάνεια «Επεξήγηση όρων χρήσης έργων τρίτων».

Τα έργα για τα οποία έχει ζητηθεί άδεια αναφέρονται στο «Σημείωμα Χρήσης Έργων Τρίτων».



[1] http://creativecommons.org/licenses/by-nc-sa/4.0/

Ως **Μη Εμπορική** ορίζεται η χρήση:

* που δεν περιλαμβάνει άμεσο ή έμμεσο οικονομικό όφελος από την χρήση του έργου, για το διανομέα του έργου και αδειοδόχο
* που δεν περιλαμβάνει οικονομική συναλλαγή ως προϋπόθεση για τη χρήση ή πρόσβαση στο έργο
* που δεν προσπορίζει στο διανομέα του έργου και αδειοδόχο έμμεσο οικονομικό όφελος (π.χ. διαφημίσεις) από την προβολή του έργου σε διαδικτυακό τόπο

Ο δικαιούχος μπορεί να παρέχει στον αδειοδόχο ξεχωριστή άδεια να χρησιμοποιεί το έργο για εμπορική χρήση, εφόσον αυτό του ζητηθεί.

**Επεξήγηση όρων χρήσης έργων τρίτων**

|  |  |
| --- | --- |
| © | Δεν επιτρέπεται η επαναχρησιμοποίηση του έργου, παρά μόνο εάν ζητηθεί εκ νέου άδεια από το δημιουργό. |
| διαθέσιμο με άδεια CC-BY | Επιτρέπεται η επαναχρησιμοποίηση του έργου και η δημιουργία παραγώγων αυτού με απλή αναφορά του δημιουργού. |
| διαθέσιμο με άδεια CC-BY-SA | Επιτρέπεται η επαναχρησιμοποίηση του έργου με αναφορά του δημιουργού, και διάθεση του έργου ή του παράγωγου αυτού με την ίδια άδεια. |
| διαθέσιμο με άδεια CC-BY-ND | Επιτρέπεται η επαναχρησιμοποίηση του έργου με αναφορά του δημιουργού. Δεν επιτρέπεται η δημιουργία παραγώγων του έργου. |
| διαθέσιμο με άδεια CC-BY-NC | Επιτρέπεται η επαναχρησιμοποίηση του έργου με αναφορά του δημιουργού. Δεν επιτρέπεται η εμπορική χρήση του έργου. |
| διαθέσιμο με άδεια CC-BY-NC-SA | Επιτρέπεται η επαναχρησιμοποίηση του έργου με αναφορά του δημιουργού και διάθεση του έργου ή του παράγωγου αυτού με την ίδια άδεια. Δεν επιτρέπεται η εμπορική χρήση του έργου. |
| διαθέσιμο με άδεια CC-BY-NC-ND | Επιτρέπεται η επαναχρησιμοποίηση του έργου με αναφορά του δημιουργού. Δεν επιτρέπεται η εμπορική χρήση του έργου και η δημιουργία παραγώγων του. |
| διαθέσιμο με άδεια CC0 Public Domain | Επιτρέπεται η επαναχρησιμοποίηση του έργου, η δημιουργία παραγώγων αυτού και η εμπορική του χρήση, χωρίς αναφορά του δημιουργού. |
| διαθέσιμο ως κοινό κτήμα | Επιτρέπεται η επαναχρησιμοποίηση του έργου, η δημιουργία παραγώγων αυτού και η εμπορική του χρήση, χωρίς αναφορά του δημιουργού. |
| χωρίς σήμανση | Συνήθως δεν επιτρέπεται η επαναχρησιμοποίηση του έργου. |

**Διατήρηση Σημειωμάτων**

* Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:
* το Σημείωμα Αναφοράς
* το Σημείωμα Αδειοδότησης
* τη δήλωση Διατήρησης Σημειωμάτων
* το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει) μαζί με τους συνοδευόμενους υπερσυνδέσμους.