

Σημειώσεις στο μάθημα Βάσεις Δεδομένων

Hands on 02

Πίνακας περιεχομένων

1	Γενικά	4
1.1	Εισαγωγή	4
1.2	SQL	6
1.3	Το Human Resources (HR) Schema	7
2	Εντολές SQL	8
2.1	SELECT	8
2.1.1	Ανάκτηση Δεδομένων με την χρήση του <i>SELECT</i>	8
2.1.2	Χρήση αριθμητικών τελεστών στο <i>SELECT</i>	9
2.1.3	Χρήση του <i>NULL</i> στο <i>SELECT</i>	10
2.1.4	Ψευδώνυμα στηλών στο <i>SELECT</i>	10
2.1.5	Ο χαρακτήρας συνένωσης, και ο τελεστής <i>DISTINCT</i>	11
2.1.6	Ασκήσεις στο <i>SELECT</i>	12
2.2	Περιορισμός (WHERE) και ταξινόμηση εγγραφών (ORDER BY) 13	
2.2.1	Σκοπός	13
2.2.2	Περιορισμός των εγγραφών που επιλέγονται	13
2.2.3	Χρήση Συμβολοσειρών και Ημερομηνιών	14
2.2.4	Τελεστές σύγκρισης	14
2.2.5	Χρήση Τελεστών Σύγκρισης	15
2.2.6	Χρήση του <i>BETWEEN</i> για αναζητήσεις σε εύρος τιμών	15
2.2.7	Χρήση του <i>IN</i> για αναζητήσεις σε σύνολο τιμών	16
2.2.8	Χρήση του <i>LIKE</i> για σύγκριση μοτίβων (<i>pattern matching</i>)	16
2.2.9	Έλεγχος <i>NULL</i>	17
2.2.10	Σύνθεση συνθηκών με την χρήση λογικών τελεστών	17

2.2.11	<i>Ο τελεστής AND.....</i>	<i>18</i>
2.2.12	<i>Ο τελεστής OR.....</i>	<i>18</i>
2.2.13	<i>Ο τελεστής NOT.....</i>	<i>19</i>
2.2.14	<i>Σειρά εκτέλεσης πράξεων τελεστών.....</i>	<i>19</i>
2.2.15	<i>Χρήση του ORDER BY για ταξινόμηση εγγραφών.....</i>	<i>20</i>
2.2.16	<i>Ασκήσεις</i>	<i>21</i>

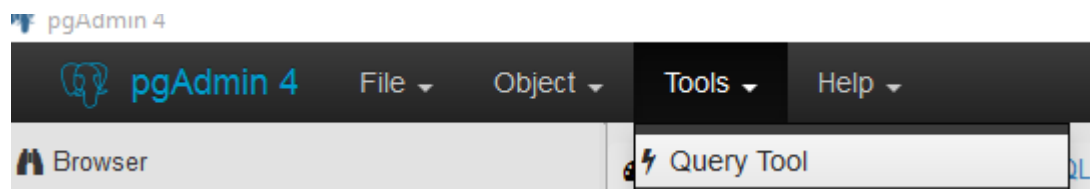
1 Γενικά

1.1 Εισαγωγή

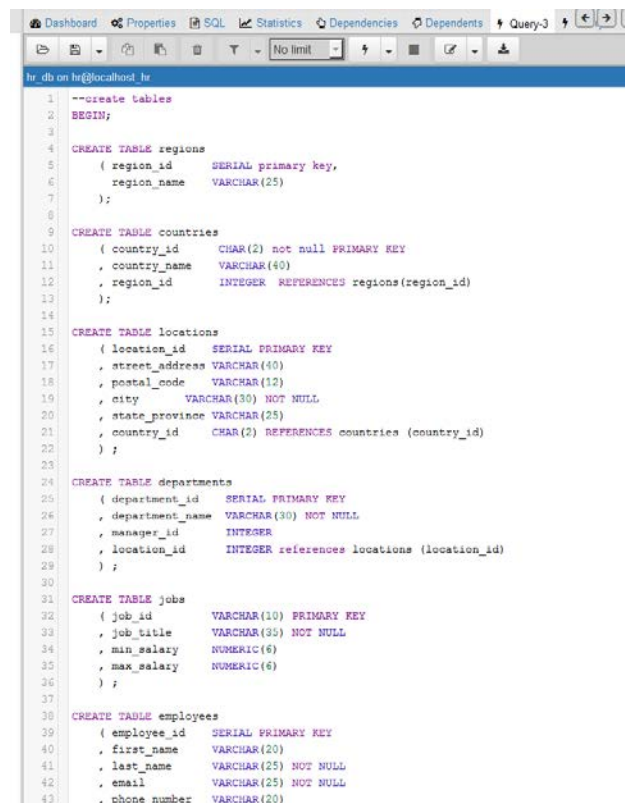
Σε αυτή την πρακτική εκπαίδευση θα μελετήσουμε πρακτικά βασικά θέματα των σχεσιακών βάσεων δεδομένων αλλά και θα δούμε διάφορες εντολές της SQL γλώσσας. Σκοπός είναι μετά την ολοκλήρωση της πρακτικής να μπορείτε να γράφετε ερωτήματα σε έναν ή περισσότερους πίνακες, να μπορείτε να τροποποιείτε δεδομένα και να δημιουργείτε αντικείμενα.

Στην περίπτωση

α. που έχετε επιλέξει το περιβάλλον του pgAdmin και αφού έχετε ολοκληρώσει τα προηγούμενα βήματα (postgresql_installation.01.oct2016, pgAdmin.01.oct2016) ,συνδέεστε στον server σας χρήστης **hr** επιλέγετε την **HRProdDB** και επιλέγετε *Tools>Query Tool* οπότε ανοίγει ένα μια καρτέλα για την εκτέλεση εντολών σε SQL.




Στη συνέχεια ανοίγετε το αρχείο HR_pgsql.sql (με notepad) από το οποίο επιλέγετε ΟΛΟ το κείμενο και κάνετε αντιγραφή και επικόλληση μέσα στην καρτέλα SQL του pgAdmin που ανοίξατε προηγουμένως.



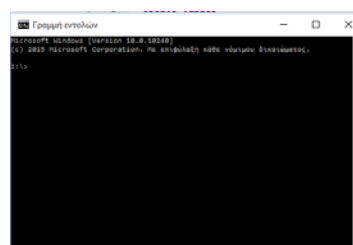
```

hr_db on hr@localhost:hr
1  --create tables
2  BEGIN;
3
4  CREATE TABLE regions
5  (
6    region_id    SERIAL primary key,
7    region_name  VARCHAR(25)
8  );
9
10 CREATE TABLE countries
11 (
12   country_id    CHAR(2) not null PRIMARY KEY
13   , country_name VARCHAR(40)
14   , region_id    INTEGER REFERENCES regions(region_id)
15 );
16
17 CREATE TABLE locations
18 (
19   location_id    SERIAL PRIMARY KEY
20   , street_address VARCHAR(40)
21   , postal_code   VARCHAR(12)
22   , city          VARCHAR(30) NOT NULL
23   , state_province VARCHAR(25)
24   , country_id    CHAR(2) REFERENCES countries (country_id)
25 );
26
27 CREATE TABLE departments
28 (
29   department_id    SERIAL PRIMARY KEY
30   , department_name VARCHAR(30) NOT NULL
31   , manager_id      INTEGER
32   , location_id      INTEGER references locations (location_id)
33 );
34
35 CREATE TABLE jobs
36 (
37   job_id          VARCHAR(10) PRIMARY KEY
38   , job_title      VARCHAR(35) NOT NULL
39   , min_salary     NUMERIC(6)
40   , max_salary     NUMERIC(6)
41 );
42
43 CREATE TABLE employees
44 (
45   employee_id    SERIAL PRIMARY KEY
46   , first_name    VARCHAR(20)
47   , last_name     VARCHAR(25) NOT NULL
48   , email         VARCHAR(25) NOT NULL
49   , phone_number  VARCHAR(20)

```

Πατάτε το εικονίδιο  και περιμένετε για να ολοκληρωθεί η εκτέλεση των εντολών. Μετά από αυτό είστε έτοιμοι να ξεκινήσετε.

β. που έχετε επιλέξει το περιβάλλον του SQL Developer και αφού έχετε ολοκληρώσει τα προηγούμενα βήματα (oracleexpress_installation.01.oct2016, sqldeveloper.01.oct2016) , ανοίγετε μια γραμμή εντολών των windows στην οποία δίνετε:



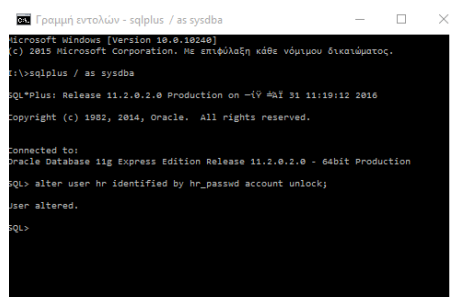
```

Γραμμή εντολών
Microsoft Windows [version 10.0.10240]
(c) 2015 Microsoft Corporation. Με επιφύλαξη κάθε νόμιμου δικαιώματος.
C:\>

```

sqlplus / as sysdba

sql>alter user hr identified by hr_passwd account unlock;



```

Γραμμή εντολών - sqlplus / as sysdba
Microsoft Windows [version 10.0.10240]
(c) 2015 Microsoft Corporation. Με επιφύλαξη κάθε νόμιμου δικαιώματος.
C:\>sqlplus / as sysdba

SQL*Plus: Release 11.2.0.2.0 Production on -[V ΘΑΙ 31 11:19:12 2016
Copyright (c) 1982, 2014, Oracle. All rights reserved.

Connected to:
Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production
SQL> alter user hr identified by hr_passwd account unlock;

User altered.
SQL>

```

Μετά από αυτό είστε έτοιμοι να ξεκινήσετε αφού δημιουργήσετε μια νέα σύνδεση αυτή τη φορά για τον χρήστη *hr* (ΟΧΙ για τον hr2).

1.2 SQL

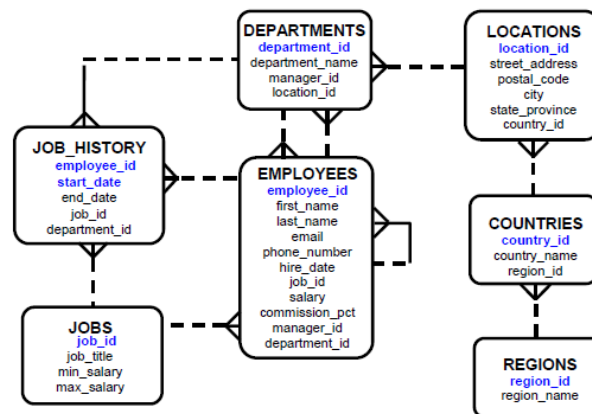
Όπως έχουμε ήδη αναφέρει με την χρήση της sql (structure query language) σε μια σχεσιακή βάση δεδομένων, αποκρύπτονται η διαδικαστικές λεπτομέρειες που απαιτούνται για την διαχείριση των δεδομένων. Η sql αποτελεί την επιλεγμένη από την ANSI (American National Standard Institute) και το ISO (International Standards Organization) γλώσσα για σχεσιακές βάσεις δεδομένων και προσφέρει την δυνατότητα για:

- α. εκτέλεση ερωτημάτων
- β. εισαγωγή, ενημέρωση και διαγραφή γραμμών από πίνακα
- γ. δημιουργία, αντικατάσταση, μεταβολή και διαγραφή αντικειμένων
- δ. έλεγχο πρόσβασης στην βάση δεδομένων και σε αντικείμενα
- ε. διασφάλιση της ακεραιότητας και των περιορισμών

Μπορούμε να κατηγοριοποιήσουμε τις εντολές τις SQL ανάλογα ως εξής:

Εντολή	Περιγραφή
<i>SELECT</i> <i>INSERT</i> <i>UPDATE</i> <i>DELETE</i> <i>MERGE</i>	Ανακτά, εισάγει, τροποποιεί και διαγράφει δεδομένα. Αναφέρεται ως <i>data manipulation language</i> (DML)
<i>CREATE</i> <i>ALTER</i> <i>DROP</i> <i>RENAME</i> <i>TRUNCATE</i> <i>COMMENT</i>	Ορίζει, μεταβάλλει και διαγράφει δομές δεδομένων. Αναφέρεται ως <i>data definition language</i> (DDL)
<i>GRANT</i> <i>REVOKE</i>	Παραχωρεί ή αφαιρεί δικαιώματα πρόσβασης στην βάση δεδομένων και σε δομές της. Αναφέρεται ως <i>data control language</i> (DCL)
<i>COMMIT</i> <i>ROLLBACK</i> <i>SAVEPOINT</i>	Δίνει την δυνατότητα ελέγχου των αλλαγών που γίνονται από τις DML εντολές. Οι αλλαγές μπορούν να ομαδοποιηθούν σε λογικές συναλλαγές (logical transactions). Αναφέρεται ως <i>transaction control language</i> (TCL)

1.3 **To** Human Resources (HR) Schema

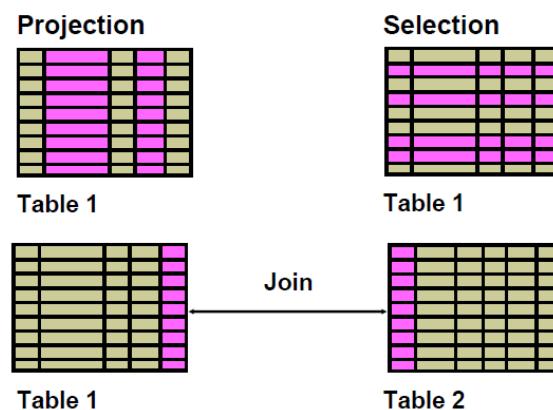


2 Εντολές SQL

2.1 SELECT

2.1.1 Ανάκτηση Δεδομένων με την χρήση του *SELECT*

Για την ανάκτηση δεδομένων από την βάση δεδομένων χρησιμοποιείτε την εντολή *SELECT* της sql η οποία σας δίνει την δυνατότητα να περιορίσετε και τα πεδία τα οποία εμφανίζονται. Πιο συγκεκριμένα οι δυνατότητες που σας δίνονται με την χρήση του *SELECT* είναι



Προβολή (projection): με την οποία επιλέγετε να προβάλλονται μια ή περισσότερες στήλες (πεδία) της επιλογής σας.

Επιλογή (selection): με την οποία επιλέγετε της γραμμές του πίνακα που επιστρέφονται με το ερώτημα σας. Εδώ μπορούν να εφαρμοστούν διάφορα κριτήρια για τον περιορισμό των γραμμών.

Σύνδεση (joining): με την οποία ανακτούνται δεδομένα που βρίσκονται σε διαφορετικούς συσχετισμένους πίνακες. Μια βασική σύνταξη του *SELECT* δείχνεται στην παρακάτω εικόνα.

```
SELECT *|{[DISTINCT] column|expression [alias],...}
FROM    table;
```

Μπορούμε να επιλέξουμε **όλες τις στήλες** από έναν πίνακα με την χρήση του συμβόλου ***. Ένας άλλος τρόπος είναι να γράψουμε τα ονόματα όλων των στηλών του πίνακα αμέσως μετά το *SELECT*.

SELECT department_id, department_name, manager_id, location_id FROM departments;


```
SELECT *  
FROM departments;
```

	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	10	Administration	200	1700
2	20	Marketing	201	1800
3	30	Shipping	124	1500
4	40	IT	103	1400
5	50	Sales	149	2500
6	60	Executive	100	1700
7	70	Accounting	205	1700
8	80	Contracting	(null)	1700

Μπορούμε να επιλέξουμε **ορισμένες στήλες** από έναν πίνακα με την χρήση του συμβόλου γράφοντας τα ονόματα όλων των στηλών χωρισμένα με κόμμα του πίνακα που ενδιαφερόμαστε αμέσως μετά το *SELECT*.

SELECT location_id, department_id FROM departments;

Στη σύνταξη εντολών SQL, όπως είναι το *SELECT*, ισχύει ότι οι εντολές δεν είναι case sensitive, ότι μπορούμε να τις γράψουμε σε μια ή περισσότερες γραμμές, ότι οι λέξεις κλειδιά δεν μπορούν να διαχωριστούν μεταξύ γραμμών και ότι τερματίζονται με την χρήση του αγγλικού ερωτηματικού (;).

2.1.2 Χρήση αριθμητικών τελεστών στο *SELECT*

Κατά την σύνταξη του *SELECT* μπορούμε να πραγματοποιήσουμε και αριθμητικές πράξεις σε αριθμητικά δεδομένα ή σε ημερομηνίες με την χρήση τελεστών όπως + (πρόσθεση), - (αφαίρεση), * (πολλαπλασιασμός), / (διαίρεση). Η χρήση των τελεστών μπορεί να γίνει σε οποιοδήποτε σημείο εκτός από το *FROM*.

Παράδειγμα χρήσης αριθμητικών τελεστών είναι το παρακάτω όπου στις τιμές του πεδίου *salary* προσθέτουμε 300 (έστω ότι θέλουμε να δώσουμε αύξηση 300 ευρώ σε ΟΛΟΥΣ!!!).

```
SELECT last_name, salary, salary + 300  
FROM employees;
```

	LAST_NAME	SALARY	SALARY+300
1	King	24000	24300
2	Kochhar	17000	17300
3	De Haan	17000	17300
4	Hunold	9000	9300
5	Ernst	6000	6300
6	Lorentz	4200	4500
7	Mourgos	5800	6100
8	Rajs	3500	3800
9	Davies	3100	3400
10	Matos	2600	2900

...

Στην περίπτωση όπου μια αριθμητική παράσταση περιέχει περισσότερους του ενός τελεστές τότε ο πολλαπλασιασμός και η διαίρεση εκτελούνται πρώτοι. Εάν οι τελεστές έχουν την ίδια προτεραιότητα τότε η εκτέλεση γίνεται από αριστερά προς τα δεξιά. Τέλος με την χρήση παρενθέσεων μπορούμε να καθορίσουμε ποιες αριθμητικές παραστάσεις θα εκτελεστούν πρώτες. Για παράδειγμα το αποτέλεσμα των δύο παρακάτω εντολών sql διαφέρει (γιατί;).

```
SELECT last_name, salary, 12*(salary+100) FROM employees;
```

```
SELECT last_name, salary, 12*salary+100 FROM employees;
```

2.1.3 Χρήση του NULL στο SELECT

Το *NULL* δηλώνει ότι τιμή μιας στήλης σε μια γραμμή δεν μας είναι διαθέσιμη ή δεν την γνωρίζουμε ή δεν μπορεί να εφαρμοστεί. Διαφέρει από το τιμή μηδέν ή το κενό διάστημα.

SELECT last_name, job_id, salary, commission_pct FROM employees;				
	LAST_NAME	JOB_ID	SALARY	COMMISSION_PCT
1	King	AD_PRES	24000	(null)
2	Kochhar	AD_VP	17000	(null)
...				
12	Zlotkey	SA_MAN	10500	0.2
13	Abel	SA_REP	11000	0.3
14	Taylor	SA_REP	8600	0.2
...				
19	Higgins	AC_MGR	12000	(null)
20	Gietz	AC_ACCOUNT	8300	(null)

Η χρήση του null σε μια αριθμητική έκφραση μας επιστρέφει πάλι null π.χ.

SELECT last_name, 12*salary*commission_pct FROM employees;		
	LAST_NAME	12*SALARY*COMMISSION_PCT
1	King	(null)
2	Kochhar	(null)
...		
12	Zlotkey	25200
13	Abel	39600
14	Taylor	20640
...		
19	Higgins	(null)
20	Gietz	(null)

2.1.4 Ψευδώνυμα στηλών στο SELECT

Με τη χρήση των ψευδώνυμων σε στήλες μπορούμε να μετονομάσουμε μια στήλη (κατά την εκτέλεση του *SELECT*). Η λειτουργία αυτή είναι χρήσιμη κατά την εκτέλεση υπολογισμών. Πραγματοποιείτε είτε γράφοντας το ψευδώνυμο αμέσως μετά το όνομα της στήλης που θέλουμε να μετονομάσουμε ή γράφοντας την δεσμευμένη λέξη *AS* και μετά το ψευδώνυμο. Στην περίπτωση που αυτό περιέχει κενά ή ειδικούς χαρακτήρες είναι χρήση διπλών εισαγωγικών (π.χ. "SALARY INCR") οπότε και είναι case sensitive.

```
SELECT last_name AS name, commission_pct comm
FROM employees;
```

	NAME	COMM
1	King	(null)
2	Kochhar	(null)
3	De Haan	(null)

...

```
SELECT last_name "Name", salary*12 "Annual Salary"
FROM employees;
```

	Name	Annual Salary
1	King	288000
2	Kochhar	204000
3	De Haan	204000

...

2.1.5 Ο χαρακτήρας συνένωσης, και ο τελεστής *DISTINCT*

Ο χαρακτήρας συνένωσης δηλώνετε με το σύμβολο // (δύο κάθετες μπάρες) και χρησιμοποιείται για να συνενώσει τιμές στηλών ή χαρακτήρες σε άλλες στήλες (κατά την εκτέλεση του *SELECT*). Στην περίπτωση που γίνει συνένωση στήλης με *NULL* τότε το αποτέλεσμα είναι η τιμή της στήλης.

```
SELECT last_name || ' is a ' || job_id
AS "Employee Details"
FROM employees;
```

	Employee Details
1	Abel is a SA_REP
2	Davies is a ST_CLERK
3	De Haan is a AD_VP
4	Ernst is a IT_PROG
5	Fay is a MK_REP

...

18	Vargas is a ST_CLERK
19	Whalen is a AD_ASST
20	Zlotkey is a SA_MAN

Κατά την εκτέλεση του ερωτήματος η προκαθορισμένη λειτουργία είναι η εμφάνιση όλων των εγγραφών συμπεριλαμβανομένων και των διπλότυπων. Η χρήση του τελεστή *DISTINCT* μας δίνει την δυνατότητα να απαλείψουμε τις διπλότυπες εγγραφές. Τοποθετούμε τον τελεστή αυτόν αμέσως μετά την δεσμευμένη λέξη *SELECT*, πχ. *SELECT DISTINCT ...*

```
SELECT department_id  
FROM employees;
```

1

	DEPARTMENT_ID
1	90
2	90
3	90
4	60
5	60

...

```
SELECT DISTINCT department_id  
FROM employees;
```

2

	DEPARTMENT_ID
1	(null)
2	90
3	20
4	110

Στην περίπτωση που μετά το *DISTINCT* επιλέξουμε περισσότερες από μια στήλες τότε ο τελεστής λειτουργεί στο συνδυασμό των τιμών αυτών των στηλών.

```
SELECT DISTINCT department_id, job_id
FROM employees;
```

	DEPARTMENT_ID	JOB_ID
1	110	AC_ACCOUNT
2	90	AD_VP
3	50	ST_CLERK
4	80	SA_REP
5	50	ST_MAN

2.1.6 Ασκήσεις στο *SELECT*

1. Τα ακόλουθα *SELECT* εκτελούνται χωρίς σφάλματα;

```
SELECT last_name, job_id, salary AS Sal FROM employees; (ΣΩΣΤΟ/ΛΑΘΟΣ)
```

```
SELECT * FROM job_grades; (ΣΩΣΤΟ/ΛΑΘΟΣ)
```

2. Η παρακάτω εντολή περιέχει τέσσερα σφάλματα μπορείτε να τα εντοπίσετε;

```
SELECT employee_id, last_name sal x 12 ANNUAL SALARY FROM employees;
```

3. Μέσα από το pgAdmin ή τον SQL Developer δείτε την δομή και τα δεδομένα των πινάκων *DEPARTMENTS*, *EMPLOYEES*. Το τμήμα HR θέλει τα επώνυμα, τον κωδικό θέσης, την ημερομηνία πρόσληψης και τον αριθμό του κάθε εργαζομένου με το πεδίο (στήλη *HIRE_DATE* να εμφανίζεται με το ψευδώνυμο *STARTDATE*). Σώστε το ερώτημα αυτό με το όνομα lab_01_05.sql
4. Το τμήμα HR θέλει μοναδικά τους κωδικούς θέσεων από τον πίνακα *EMPLOYEES*.

5. Το τμήμα HR θέλει καλύτερες περιγραφές στο ερώτημα lab_01_05.sql για αυτό το λόγο δώστε κατάλληλα ονόματα στις στήλες *Emp#*, *Employee*, *Job* και *Hire Date* αντίστοιχα.
6. Το τμήμα HR ζητά επίσης μια αναφορά όπου θα φαίνονται όλοι οι υπάλληλοι και οι κωδικοί θέσεων. Εμφανίστε την αναφορά αυτή χρησιμοποιώντας το επίθετο σε συνένωση με τον κωδικό θέσης και ονομάστε τη στήλη *Employee and Title*
7. Εμφανίστε όλα τα δεδομένα του πίνακα *EMPLOYEES* χωρίζοντας τις τιμές όλων των πεδίων με κόμμα και ονομάστε την στήλη αυτή *THE_OUTPUT*.

2.2 Περιορισμός (WHERE) και ταξινόμηση εγγραφών (ORDER BY)

2.2.1 Σκοπός

Σκοπός της ενότητας αυτής είναι να μάθουμε πως μπορούμε να περιορίσουμε και να ταξινομήσουμε τις εγγραφές ενός ερωτήματος. Πιο συγκεκριμένα για τον περιορισμό των αποτελεσμάτων θα δούμε την χρήση του *WHERE* σε συνδυασμό με διάφορους τελεστές σύγκρισης όπως είναι τα *=*, *<=*, *BETWEEN*, *IN*, *LIKE*, αλλά και τελεστών για *NULL* και για λογικούς ελέγχους με *AND*, *OR* και *NOT*.

2.2.2 Περιορισμός των εγγραφών που επιλέγονται

Για να περιορίσουμε τις εγγραφές που επιλέγονται χρησιμοποιούμε το *WHERE* με το οποίο ορίζεται η συνθήκη που πρέπει να επαληθεύουν οι εγγραφές για να εμφανίζονται στο αποτέλεσμα. Ακολουθεί πάντα μετά από το *FROM* στην σύνταξη του ερωτήματος.

```
SELECT *|{[DISTINCT] column|expression [alias],...}
FROM table
[WHERE condition(s)];
```

Το *WHERE* μπορούμε να το χρησιμοποιήσουμε για να συγκρίνουμε τιμές στηλών (για τα ονόματα των οποίων ΔΕΝ μπορούμε να χρησιμοποιήσουμε ψευδώνυμα), αριθμητικές εκφράσεις ή συναρτήσεις και αποτελείται από τρία μέρη: το όνομα της στήλης που θέλουμε να συγκρίνουμε, τον τελεστή σύγκρισης και το όνομα της στήλης, σταθερά ή σύνολο τιμών με τον οποίο γίνεται η σύγκριση.

Για παράδειγμα παρακάτω επιλέγουμε το *employee ID*, *last name*, *job ID* και το *department number* όλων των υπαλλήλων που ανήκουν στο department 90.

```
SELECT employee_id, last_name, job_id, department_id
FROM employees
WHERE department_id = 90;
```

	EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
1	100	King	AD_PRES	90
2	101	Kochhar	AD_VP	90
3	102	De Haan	AD_VP	90

2.2.3 Χρήση Συμβολοσειρών και Ημερομηνιών

Οι συμβολοσειρές και οι ημερομηνίες στο *WHERE* πρέπει να εμπεριέχονται σε μονά εισαγωγικά (' '). Η αναζήτηση στις συμβολοσειρές είναι *case sensitive* πράγμα που σημαίνει ότι παίζει ρόλο η χρήση πεζών ή κεφαλαίων. Το παρακάτω ερώτημα δεν επιστρέφει αποτελέσματα καθώς οι τιμές του *last_name* συντάσσονται με συνδυασμό κεφαλαίων και μικρών π.χ. 'Whalen'

```
SELECT last_name, department_id FROM employees WHERE last_name = 'WHALEN';
```

Στις ημερομηνίες οι αναζητήσεις πρέπει επίσης να γίνονται με την χρήση μονών εισαγωγικών π.χ.

```
SELECT last_name FROM employees WHERE hire_date = '17-FEB-96';
```

2.2.4 Τελεστές σύγκρισης

Operator	Meaning
=	Equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
<>	Not equal to
BETWEEN ...AND...	Between two values (inclusive)
IN (set)	Match any of a list of values
LIKE	Match a character pattern
IS NULL	Is a null value

Οι τελεστές σύγκρισης χρησιμοποιούνται στις συνθήκες για την σύγκριση μια έκφρασης με μια τιμή ή μια άλλη έκφραση. Η χρήση τους γίνεται στο *WHERE*

```
... WHERE expr operator value
```

π.χ.

```
... WHERE hire_date = '17-FEB-96';
```

```
... WHERE salary > 6000;
```

```
... WHERE hire_date < '17-FEB-96';
```

2.2.5 Χρήση Τελεστών Σύγκρισης

Στο παρακάτω παράδειγμα επιλέγουμε το *last_name* και το *salary* από τον πίνακα *EMPLOYEES* για εκείνους του υπαλλήλους των οποίων ο μισθός είναι μικρότερος ή ίσος με 3000. Παρατηρήστε ότι ορίζεται μια τιμή στο τμήμα του *WHERE* η οποία συγκρίνεται με τις τιμές που περιέχει η στήλη *SALARY* του πίνακα.

```
SELECT last_name, salary
FROM employees
WHERE salary <= 3000 ;
```

	LAST_NAME	SALARY
1	Matos	2600
2	Vargas	2500

2.2.6 Χρήση του BETWEEN για αναζητήσεις σε εύρος τιμών

Με την χρήση του *BETWEEN* μπορεί να γίνει αναζήτηση σε εύρος τιμών για το οποίο καθορίζουμε πρώτο το κάτω και δεύτερο το πάνω όριο. Μπορεί να γίνει ορισμός του εύρους σε αριθμητικές τιμές, σε κείμενο και σε ημερομηνίες π.χ.

```
SELECT last_name, salary
FROM employees
WHERE salary BETWEEN 2500 AND 3500 ;
```

Lower limit Upper limit

	LAST_NAME	SALARY
1	Rajs	3500
2	Davies	3100
3	Matos	2600
4	Vargas	2500

```
SELECT last_name
FROM employees
WHERE last_name BETWEEN 'King' AND 'Smith';
```

	LAST_NAME
1	King
2	Kochhar
3	Lorentz
4	Matos
5	Mourgos
6	Rajs

2.2.7 Χρήση του *IN* για αναζητήσεις σε σύνολο τιμών

```
SELECT employee_id, last_name, salary, manager_id
FROM employees
WHERE manager_id IN (100, 101, 201) ;
```

	EMPLOYEE_ID	LAST_NAME	SALARY	MANAGER_ID
1	101	Kochhar	17000	100
2	102	De Haan	17000	100
3	124	Mourgos	5800	100
4	149	Zlotkey	10500	100
5	201	Hartstein	13000	100
6	200	Whalen	4400	101
7	205	Higgins	12000	101
8	202	Fay	6000	201

Με την χρήση του *IN* μπορεί να γίνει έλεγχος ύπαρξης μιας τιμής σε σύνολο τιμών. Μπορεί να χρησιμοποιηθεί για οποιοδήποτε τύπο δεδομένων και στην περίπτωση των συμβολοσειρών και των ημερομηνιών πρέπει αυτά να περικλείονται σε μονά εισαγωγικά π.χ.

```
SELECT employee_id, manager_id, department_id
FROM employees
WHERE last_name IN ('Hartstein', 'Vargas');
```

2.2.8 Χρήση του *LIKE* για σύγκριση μοτίβων (pattern matching)

```
SELECT first_name
FROM employees
WHERE first_name LIKE 'S%';
```

Με την χρήση του *LIKE* μπορεί να γίνει σύγκριση με χαρακτήρες μπαλαντέρ (wildcard) σε τιμές οι οποίες μπορεί να είναι κείμενο/αριθμοί ή να συνδυασμός τους με την χρήση του

% το οποίο δηλώνει ύπαρξη μηδέν ή περισσότερων χαρακτήρων

_ το οποίο δηλώνει ενός χαρακτήρα (ακριβώς)

Χρήση του *LIKE* γίνεται σε περιπτώσεις που δεν είναι γνωστή επακριβώς η τιμή την οποία ψάχνουμε, για παράδειγμα στην παραπάνω εικόνα γίνεται αναζήτηση για εκείνους του υπαλλήλους που το όνομα τους ξεκινά με το γράμμα S και μπορεί να περιέχει, στην συνέχεια, οποιοσδήποτε χαρακτήρες και οποιοδήποτε αριθμό χαρακτήρων (χρήση του %).

Υπάρχει η δυνατότητα να γίνει χρήση συνδυασμού των δύο χαρακτήρων μπαλαντέρ (% _) κατά την αναζήτηση π.χ.


```
SELECT last_name
FROM employees
WHERE last_name LIKE '_o%';
```

LAST_NAME
1 Kochhar
2 Lorentz
3 Mourgos

2.2.9 Έλεγχος NULL

```
SELECT last_name, manager_id
FROM employees
WHERE manager_id IS NULL;
```

LAST_NAME	MANAGER_ID
1 King	(null)

Επειδή η τιμή *null* σημαίνει την απουσία τιμής δεν μπορεί να γίνει η χρήση του τελεστή της ισότητας = και για αυτό τον λόγο χρειαζόμαστε ειδικό τελεστή. Έτσι για να ελέγξουμε την (μη)ύπαρξη του *null* χρησιμοποιούμε τους τελεστές *IS NOT NULL* και *IS NULL*.

Στο πιο πάνω παράδειγμα αναζητούνται οι υπάλληλοι οι οποίοι δεν έχουν μάνατζερ.

2.2.10 Σύνθεση συνθηκών με την χρήση λογικών τελεστών

Operator	Meaning
AND	Returns TRUE if <i>both</i> component conditions are true
OR	Returns TRUE if <i>either</i> component condition is true
NOT	Returns TRUE if the condition is false

Με την χρήση των λογικών τελεστών μπορούμε να συνδυάσουμε επιμέρους συνθήκες ώστε το τελικό αποτέλεσμα να προκύπτει από την επαλήθευση τους ή την επαλήθευση της αντίστροφής τους. Σε κάθε περίπτωση μια εγγραφή εμφανίζεται στο αποτέλεσμα μόνο εάν η συνολική συνθήκη είναι αληθής, επιστρέφει TRUE. Ο λογικοί τελεστές οι οποίοι είναι διαθέσιμοι στην SQL είναι οι: *AND*, *OR* και *NOT* και στον παραπάνω πίνακα δείχνεται πότε επιστρέφεται αποτέλεσμα ανά τελεστή.

2.2.11 Ο τελεστής AND

```
SELECT employee_id, last_name, job_id, salary
FROM employees
WHERE salary >= 10000
AND job_id LIKE '%MAN%';
```

	EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
1	149	Zlotkey	SA_MAN	10500
2	201	Hartstein	MK_MAN	13000

Ο τελεστής *AND* απαιτεί όλες οι συνθήκες που συμμετέχουν να είναι αληθείς, δηλαδή να επιστρέφουν TRUE. Για να γίνει κατανοητό αυτό στο παράδειγμα που δείχνεται επιστρέφονται μόνο εκείνοι οι υπάλληλοι που έχουν μισθό μεγαλύτερο από 10.000 και των οποίων ο τίτλος εργασίας περιέχει την συμβολοσειρά 'MAN'. Στον παρακάτω πίνακα φαίνονται τα αποτελέσματα του συνδυασμού δύο συνθηκών με *AND*.

AND	TRUE	FALSE	NULL
TRUE	TRUE	FALSE	NULL
FALSE	FALSE	FALSE	FALSE
NULL	NULL	FALSE	NULL

2.2.12 Ο τελεστής OR

```
SELECT employee_id, last_name, job_id, salary
FROM employees
WHERE salary >= 10000
OR job_id LIKE '%MAN%';
```

	EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
1	100	King	AD_PRES	24000
2	101	Kochhar	AD_VP	17000
3	102	De Haan	AD_VP	17000
4	124	Mourgos	ST_MAN	5800
5	149	Zlotkey	SA_MAN	10500
6	174	Abel	SA_REP	11000
7	201	Hartstein	MK_MAN	13000
8	205	Higgins	AC_MGR	12000

Ο τελεστής *OR* απαιτεί κάποια από τις συνθήκες που συμμετέχουν να είναι αληθείς, δηλαδή να επιστρέφει TRUE. Για να γίνει κατανοητό αυτό στο παράδειγμα που δείχνεται επιστρέφονται εκείνοι οι υπάλληλοι που έχουν μισθό μεγαλύτερο από 10.000 ή των οποίων ο τίτλος εργασίας (*job_id*) περιέχει την συμβολοσειρά 'MAN'. Στον παρακάτω πίνακα φαίνονται τα αποτελέσματα του συνδυασμού δύο συνθηκών με *OR*.

OR	TRUE	FALSE	NULL
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	FALSE	NULL
NULL	TRUE	NULL	NULL

2.2.13 Ο τελεστής NOT

```
SELECT last_name, job_id
FROM employees
WHERE job_id
      NOT IN ('IT_PROG', 'ST_CLERK', 'SA_REP') ;
```

	LAST_NAME	JOB_ID
1	De Haan	AD_VP
2	Fay	MK_REP
3	Gietz	AC_ACCOUNT
4	Hartstein	MK_MAN
5	Higgins	AC_MGR
6	King	AD_PRES
7	Kochhar	AD_VP
8	Mourgos	ST_MAN
9	Whalen	AD_ASST
10	Zlotkey	SA_MAN

Στο πιο πάνω ερώτημα επιστρέφονται εκείνοι οι υπάλληλοι των οποίων ο τίτλος εργασίας (*job_id*) δεν περιέχει την συμβολοσειρά 'MAN'. Στον παρακάτω πίνακα φαίνονται τα αποτελέσματα όταν εφαρμόζεται το *NOT*.

NOT	TRUE	FALSE	NULL
	FALSE	TRUE	NULL

Το *NOT* μπορεί επίσης να χρησιμοποιηθεί με το *BETWEEN*, *LIKE* και *NULL*.

2.2.14 Σειρά εκτέλεσης πράξεων τελεστών

Operator	Meaning
1	Arithmetic operators
2	Concatenation operator
3	Comparison conditions
4	IS [NOT] NULL, LIKE, [NOT] IN
5	[NOT] BETWEEN
6	Not equal to
7	NOT logical condition
8	AND logical condition
9	OR logical condition

Στον πίνακα φαίνεται η σειρά εκτέλεσης των πράξεων των τελεστών στην περίπτωση που εμφανίζονται στην ίδια έκφραση. Η σειρά αυτή μπορεί να ελεγχθεί με την χρήση παρενθέσεων. Για παράδειγμα το ερώτημα 1 μεταφράζεται ως εξής "Επέλεξε μια εγγραφή στην περίπτωση που ο υπάλληλος έχει τίτλο εργασίας 'AD_PRES' και έχει μισθό μεγαλύτερο από 15.000, ή έχει τίτλο εργασίας 'SA_REP' ". Το ερώτημα 2 μεταφράζεται "Επέλεξε μια

εγγραφή στην περίπτωση που ένας υπάλληλος έχει τίτλο εργασίας 'AD_PRES' ή 'SA_REP', και έχει μισθό μεγαλύτερο από 15.000'.

```
SELECT last_name, job_id, salary
FROM employees
WHERE job_id = 'SA_REP'
OR job_id = 'AD_PRES'
AND salary > 15000;
```

1

	LAST_NAME	JOB_ID	SALARY
1	King	AD_PRES	24000
2	Abel	SA_REP	11000
3	Taylor	SA_REP	8600
4	Grant	SA_REP	7000

```
SELECT last_name, job_id, salary
FROM employees
WHERE (job_id = 'SA_REP'
OR job_id = 'AD_PRES')
AND salary > 15000;
```

2

	LAST_NAME	JOB_ID	SALARY
1	King	AD_PRES	24000

2.2.15 Χρήση του ORDER BY για ταξινόμηση εγγραφών

Syntax

```
SELECT          expr
FROM            table
[WHERE          condition(s)]
[ORDER BY {column, expr, numeric_position} [ASC|DESC]];
```

In the syntax:

```
ORDER BY       specifies the order in which the retrieved rows are displayed
ASC            orders the rows in ascending order (this is the default order)
DESC          orders the rows in descending order
```

Με την χρήση του *ORDER BY* μπορούν να ταξινομηθούν οι εγγραφές που επιστρέφονται σε ένα ερώτημα είτε με αύξουσα σειρά (*ASC*), που είναι το προκαθορισμένο, είτε με φθίνουσα (*DESC*). Το *ORDER BY* τοποθετείται πάντα τελευταίο στην σύνταξη του *SELECT* και εκτός από το όνομα της στήλης μπορεί να γίνει χρήση έκφρασης, ψευδώνυμου ή αριθμητική θέση της στήλης στο *SELECT* για να οριστεί η σειρά ταξινόμησης. Τέλος με την χρήση των *NULLS FIRST* και *NULLS LAST* μπορούμε να ορίσουμε αν στην περίπτωση που υπάρχουν *NULL* αυτά θα εμφανίζονται πρώτα ή τελευταία.

```
SELECT last_name, job_id, department_id, hire_date
FROM employees
ORDER BY hire_date DESC;
```

1

```
SELECT employee_id, last_name, salary*12 annsal
FROM employees
ORDER BY annsal;
```

2

```
SELECT last_name, job_id, department_id, hire_date
FROM employees
ORDER BY 3;
```



3

```
SELECT last_name, department_id, salary
FROM employees
ORDER BY department_id, salary DESC;
```



4

2.2.16 Ασκήσεις

1. Λόγω περικοπών, το τμήμα HR θέλει το επώνυμο και τον μισθό των εργαζομένων που κερδίζουν περισσότερα από 12.000. Σώστε το ερώτημα αυτό με το όνομα lab_02_01.sql.

	 LAST_NAME	 SALARY
1	King	24000
2	Kochhar	17000
3	De Haan	17000
4	Hartstein	13000

2. Σε ένα ερώτημα εμφανίστε το επώνυμο και τον αριθμό τμήματος του υπαλλήλου με αριθμό 176.

	 LAST_NAME	 DEPARTMENT_ID
1	Taylor	80

3. Το τμήμα HR θέλει να βρει τους υπαλλήλους με τον μεγαλύτερο και τον μικρότερο μισθό. Τροποποιήστε το ερώτημα lab_02_01.sql έτσι ώστε να προβάλετε το όνομα και τον μισθό όλων των υπαλλήλων των οποίων ο μισθός ΔΕΝ βρίσκεται μεταξύ 5.000 και 12.000. Αποθηκεύστε το ερώτημα ως lab_02_03.sql.

	 LAST_NAME	 SALARY
1	King	24000
2	Kochhar	17000
3	De Haan	17000
4	Lorentz	4200
5	Rajs	3500
6	Davies	3100
7	Matos	2600
8	Vargas	2500
9	Whalen	4400
10	Hartstein	13000

4. Δημιουργήστε ένα ερώτημα στο οποίο θα προβάλετε το επώνυμο, τον αριθμό και την ημερομηνία έναρξης όλων των υπαλλήλων με επώνυμο Matos και Taylor. Ταξινομήστε τα αποτελέσματα με αύξουσα σειρά κατά την ημερομηνία έναρξης.

	A Z LAST_NAME	A Z JOB_ID	HIRE_DATE
1	Matos	ST_CLERK	15-MAR-98
2	Taylor	SA_REP	24-MAR-98

5. Εμφανίστε το επώνυμο και το αριθμό τμήματος όλων των υπαλλήλων των τμημάτων 20 ή 50 σε αύξουσα σειρά κατά επώνυμο.

	A Z LAST_NAME	A Z DEPARTMENT_ID
1	Davies	50
2	Fay	20
3	Hartstein	20
4	Matos	50
5	Mourgos	50
6	Rajs	50
7	Vargas	50

6. Τροποποιήστε το ερώτημα lab_02_03.sql έτσι ώστε στο αποτέλεσμα σας να εμφανίζεται το επώνυμο και ο μισθός των υπαλλήλων των οποίων οι αποδοχές είναι μεταξύ 5.000 και 12.000 και οι οποίοι βρίσκονται στο τμήμα 20 ή 50. Ονομάστε τα πεδία *Employee* και *Monthly Salary* αντίστοιχα και σώστε το νέο ερώτημα με όνομα lab_02_06.sql.

	A Z Employee	A Z Monthly Salary
1	Fay	6000
2	Mourgos	5800

7. Το τμήμα HR θέλει το επώνυμο και την ημερομηνία πρόσληψης όλων των υπαλλήλων που προσλήφθηκαν το 1994.

	A Z LAST_NAME	HIRE_DATE
1	Higgins	07-JUN-94
2	Gietz	07-JUN-94

8.

9. Εμφανίστε το επώνυμο και τον τίτλο εργασίας εκείνων των υπαλλήλων που δεν έχουν μάνατζερ.

	A Z LAST_NAME	A Z JOB_ID
1	King	AD_PRES

10. Εμφανίστε τα επώνυμα των υπαλλήλων που έχουν ταυτόχρονα "α" και "ε" στο επώνυμο τους.

	A Z LAST_NAME
1	Davies
2	De Haan
3	Hartstein
4	Whalen