

Σημειώσεις στο μάθημα Βάσεις Δεδομένων II

Hands on 2.2

Πίνακας περιεχομένων

1	Γενικά	4
1.1	Εισαγωγή	4
2	Το Human Resources (HR) Schema	6
3	Δημιουργία Όψεων	7
3.1	Τι είναι μια Όψη (view);	7
3.2	Δημιουργία/Τροποποίηση Όψεων (views)	7
3.3	Ανάκτηση Δεδομένων από μια Όψη	9
3.4	DML Εντολές σε μια Όψη	9
3.5	Διαγραφή Όψης	10
3.6	Inline Όψεις	10
3.7	Top-n Ανάλυση	11
3.8	Ασκήσεις	12
4	Άλλα Αντικείμενα των ΒΔ	13
4.1	Αντικείμενα ΒΔ	13
4.2	Ακολουθίες (sequences)	13
4.3	Ευρετήρια (indexes)	15
4.4	Συνώνυμα (synonyms)	16
4.5	Ασκήσεις	17
5	Έλεγχος Πρόσβασης	17
5.1	Έλεγχος Πρόσβασης Χρηστών	17
5.2	Δικαιώματα	17
5.3	Δημιουργία Χρήστη	18
5.4	Συστημικά Δικαιώματα	18
5.5	Ρόλοι	19
5.6	Προνόμια σε Αντικείμενα	19
5.7	Ασκήσεις	20

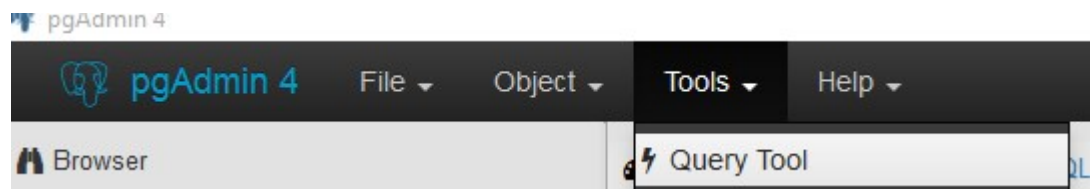
1 Γενικά

1.1 Εισαγωγή

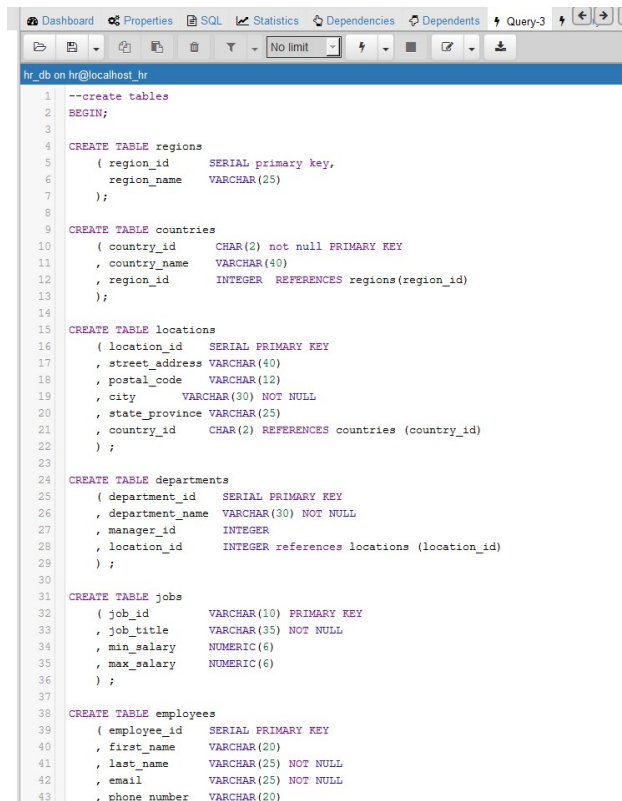
Σε αυτή την πρακτική εκπαίδευση θα συνεχίζουμε να εξετάζουμε πρακτικά θέματα των σχεσιακών βάσεων δεδομένων και διάφορες εντολές της SQL. Σκοπός είναι μετά την ολοκλήρωση της πρακτικής να έχετε μια πληρέστερη εικόνα ενός RDBMS καθώς και των δυνατοτήτων της SQL μέσα σε αυτό.

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 σαν χρήστης **hr** επιλέγετε την **HRProdDB** και επιλέγετε **Tools>QueryTool** οπότε ανοίγει ένα 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.



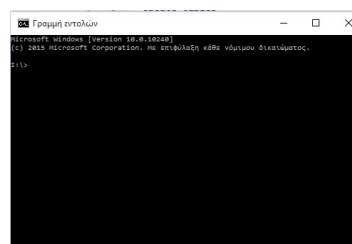
```

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)
50 );

```

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:



sqlplus / as sysdba

sql>alter user hr identified by hr_passwd account unlock;

```
Γρομμή 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} #41 31 11:19:12 2016
Copyright (c) 1982, 2014, Oracle. All rights reserved.

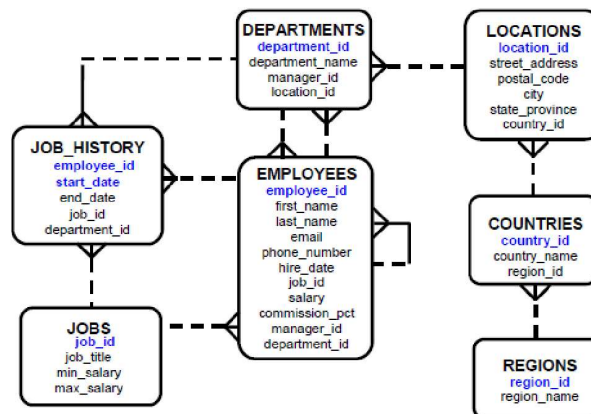
Connected to:
Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production
SQL> alter user hr identified by hr_passwd account unlock;

User altered.

SQL>
```

Μετά από αυτό είστε έτοιμοι να ξεκινήσετε αφού δημιουργήσετε μια νέα σύνδεση για τον χρήστη hr.

2 To Human Resources (HR) Schema



3 Δημιουργία Όψεων

3.1 Τι είναι μια Όψη (view);

Με την χρήση των views (όψεων) μπορούμε να προβάλουμε ένα λογικό υποσύνολο των δεδομένων ενός πίνακα ή ένα συνδυασμό δεδομένων από πίνακες. Μια view είναι ένας λογικός πίνακας βασισμένος σε έναν πίνακα ή σε μια άλλη view. Η view δεν περιέχει δικά της δεδομένα αλλά λειτουργεί σαν ένα “παράθυρο” μέσα από το οποίο μπορούμε να δούμε ή να μεταβάλουμε δεδομένα πινάκων. Ο ορισμός της view αποθηκεύεται σαν ένα ερώτημα *SELECT* μέσα στα μετα-δεδομένα του RDBMS

EMPLOYEES Table

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY
100	Steven	King	SKING	515.123.4567	17-JUN-87	AD_PRES	24000
101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-89	AD_VP	17000
102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-93	AD_VP	17000
103	Alexander	Hunold	AHUNOLD	590.423.4567	03-JAN-90	IT_PROG	9000
104	Bruce	Skott	BSKOTT	590.423.4568	04-JAN-90	IT_PROG	6000
105	David	Turner	DTURNER	590.423.4569	07-DEC-91	IT_PROG	4200
106	Ellen	Abel	EABEL	590.423.4567	10-AUG-93	IT_PROG	5600
107	Greg	King	GKING	590.423.4567	17-JUN-93	IT_PROG	3500
108	John	Smith	JSMITH	590.423.4567	10-SEP-95	IT_PROG	3100
109	Joshua	Clark	JCLARK	590.423.4567	03-OCT-96	IT_PROG	2500
110	Kevin	Conner	KCONNER	590.423.4567	09-SEP-96	IT_PROG	2500
111	Shelley	Stevens	SSTEVENS	590.423.4567	05-MAY-97	IT_PROG	19000
112	Timothy	Gietz	TGIETZ	590.423.4567	15-SEP-97	IT_PROG	11000
113	Walter	Taylor	WTAYLOR	590.423.4567	01-JAN-98	IT_PROG	8500
114	Wendell	Olson	WOLSON	590.423.4567	07-FEB-98	IT_PROG	13000
115	Xavier	Patel	XPADEL	590.423.4567	05-APR-98	IT_PROG	9500
116	Yusuf	Kolungra	YKOLUNGRA	590.423.4567	08-MAY-98	IT_PROG	8000
117	Zoe	Grant	ZGRANT	590.423.4567	01-JUN-98	IT_PROG	7000
118	Carol	De Haan	CDEHAAN	590.423.4567	03-JUN-98	IT_PROG	6000
119	Cheryl	Winters	CWINTERS	590.423.4567	07-JUN-98	IT_PROG	5500
120	Christina	Mavris	CMAVRIS	590.423.4567	08-JUN-98	IT_PROG	4900
121	Cristina	Mavris	CMAVRIS	590.423.4567	08-JUN-98	IT_PROG	4900
122	Dana	Fey	DFEY	590.423.4567	17-AUG-97	IT_PROG	6000
123	Doris	Green	DGREEN	590.423.4567	13-JUN-97	IT_PROG	5500
124	Douglas	Grant	DGRANT	590.423.4567	13-JUN-97	IT_PROG	5500
125	Jay	Fay	JFAY	590.423.4567	17-AUG-97	IT_PROG	6000
126	Jenna	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
127	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
128	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
129	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
130	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
131	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
132	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
133	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
134	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
135	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
136	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
137	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
138	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
139	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
140	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
141	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
142	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
143	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
144	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
145	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
146	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
147	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
148	John	Wambauer	JWAMBAUER	590.423.4567	17-AUG-97	IT_PROG	6000
149	Zlotkey					SA_MAN	10500
174	Abel					SA_REP	11000
175	Taylor					SA_REP	8500
202	Pat	Fay	PFAY	603.123.6666	17-AUG-97	MK_REP	6000
205	Shelley	Higgins	SHIGGINS	515.123.8080	07-JUN-94	AC_MGR	12000
206	William	Gietz	WGIEZT	515.123.8181	07-JUN-94	AC_ACCOUNT	8300

Με την χρήση των views μπορούμε:

- να περιορίσουμε την πρόσβαση σε ευαίσθητα δεδομένα, επιλέγοντας αυτά να μην προβάλλονται
 - να περιορίσουμε την πολυπλοκότητα σύνθετων ερωτημάτων στον χρήστη δίνοντάς του την δυνατότητα να παίρνει τα δεδομένα που θέλει χρησιμοποιώντας τις views που χρειάζεται
 - να ομαδοποιήσουμε τις απαιτήσεις χρηστών-εφαρμογών και να τις εξυπηρετήσουμε με την χρήση συγκεκριμένων views
- Τις views μπορούμε να τις κατηγοριοποιήσουμε σε απλές και σε σύνθετες. Οι απλές λαμβάνουν τα δεδομένα από ένα πίνακα, δεν περιέχουν συναρτήσεις ή ομαδοποιήσεις στον ορισμό τους και υποστηρίζουν DML ενέργειες. Σε αντίθεση οι σύνθετες βασίζονται σε περισσότερους από ένα πίνακες, περιέχουν συναρτήσεις ή ομαδοποιήσεις δεδομένων στον ορισμό τους και συνήθως δεν υποστηρίζουν DML ενέργειες.

3.2 Δημιουργία/Τροποποίηση Όψεων (views)

Η σύνταξη για την δημιουργία μιας view φαίνεται στην παρακάτω εικόνα, όπου το:

```
CREATE [ OR REPLACE ] VIEW
... [ owner. ]view-name [ ( column-name [ , ... ] ) ]
... AS select-without-order-by
... [ WITH CHECK OPTION ]
```

[OR REPLACE] δηλώνει ότι η view θα ξαναδημιουργηθεί εάν υπάρχει ήδη

VIEW δηλώνει το όνομα της view

column-name δηλώνει τα ονόματα των στηλών όπως θα εμφανίζονται στην view

AS select-without-order-by δηλώνει το ερώτημα που καθορίζει την view

[WITH CHECK OPTION] δηλώνει ότι μόνο εγγραφές που ορίζονται μέσα από την view μπορούν να εισαχθούν ή να τροποποιηθούν

Για παράδειγμα στην εικόνα που φαίνεται δημιουργούμε την view EMPVU80 η οποία περιέχει τους υπαλλήλους μόνο του τμήματος 80

```
CREATE VIEW empvu80
AS SELECT employee_id, last_name, salary
FROM employees
WHERE department_id = 80;
View created.
```

Η view αυτή θα έχει τρεις (3) στήλες: employee_id, last_name, salary. Μπορούμε να δούμε την δομή της δίνοντας την εντολή *DESCRIBE empvu80*; Στη περίπτωση όπου θέλουμε στη view που θα δημιουργήσουμε τα ονόματα των στηλών της να είναι διαφορετικά από αυτά που εμφανίζονται στο SELECT τότε μπορούμε είτε να χρησιμοποιήσουμε ψευδώνυμα μέσα στο *SELECT*, είτε να ορίσουμε τα ονόματα των στηλών της αμέσως μετά την δήλωση *CREATE VIEW* και πριν από το *SELECT*, όπως δείχνεται και στα παραδείγματα που ακολουθούν.

```
CREATE VIEW salvu50
AS SELECT employee_id ID_NUMBER, last_name NAME,
salary*12 ANN_SALARY
FROM employees
WHERE department_id = 50;
View created.
```

```
CREATE VIEW salvu50 (ID_NUMBER, NAME, ANN_SALARY)
AS SELECT employee_id, last_name, salary*12
FROM employees
WHERE department_id = 50;
View created.
```

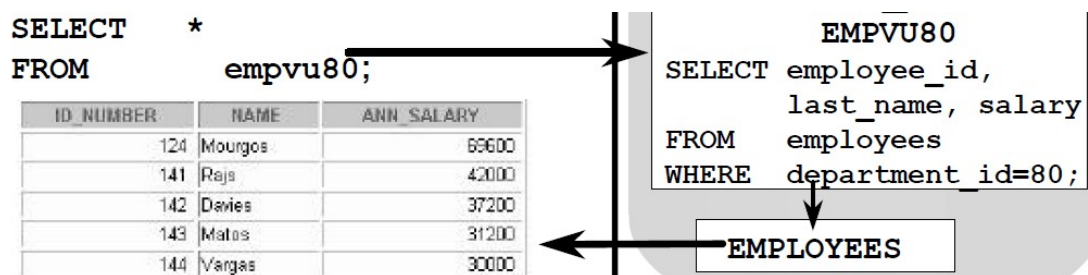
Εκτός όμως από τις απλές views όπως έχουμε πεί υπάρχουν και οι σύνθετες. Μια σύνθετη view δείχνεται παρακάτω, στην οποία υπολογίζονται ο ελάχιστος, ο μέγιστος και ο μέσος μισθός ανά τμήμα. Προσέξτε ότι στην δημιουργία της οι στήλες ορίζονται αμέσως μετά το *CREATE VIEW*.


```
CREATE VIEW dept_sum_vu
(name, minsal, maxsal, avgsal)
AS SELECT      d.department_name, MIN(e.salary),
              MAX(e.salary), AVG(e.salary)
FROM          employees e, departments d
WHERE         e.department_id = d.department_id
GROUP BY      d.department_name;
View created.
```

Για να τροποποιήσουμε μια όψη μπορούμε να χρησιμοποιήσουμε την επιλογή *OR REPLACE* κατά την οποία δημιουργείται ξανά η όψη ακόμη και αν υπάρχει μια άλλη με το ίδιο όνομα. Με τον τρόπο αυτό μια όψη μπορεί να τροποποιηθεί χωρίς να υπάρχει η ανάγκη να την σβήσουμε, να την δημιουργήσουμε και να δώσουμε ξανά τα κατάλληλα δικαιώματα στους χρήστες (θα δούμε τα δικαιώματα σε άλλη ενότητα).

3.3 Ανάκτηση Δεδομένων από μια Όψη

Μπορούμε να ανακτήσουμε δεδομένα από μια view με τον ίδιο ακριβώς τρόπο όπως και από έναν πίνακα. Μπορούμε να εμφανίσουμε το σύνολο των δεδομένων και στηλών ή να επιλέξουμε κάθε φορά τα επιθυμητά.



Στην πιο πάνω εικόνα φαίνεται η διαδικασία που εκτελείται κατά την χρήση μιας view όπου η ΒΔ μεταφράζει το ερώτημα πάνω στην όψη στο αντίστοιχο ερώτημα με το οποίο αυτή έχει οριστεί.

3.4 DML Εντολές σε μια Όψη

Μέσα σε μια όψη μπορούμε να εκτελέσουμε DML εντολές (*INSERT*, *UPDATE*, *DELETE*) όταν ικανοποιούνται συγκεκριμένα κριτήρια. Έτσι μπορούμε να διαγράψουμε εγγραφές στην περίπτωση όπου η όψη δεν περιλαμβάνει εκφράσεις *GROUP BY*, *DISTINCT* και συναρτήσεις συνάθροισης. Μπορούμε να τροποποιήσουμε εγγραφές μόνο εάν δεν περιλαμβάνει συναρτήσεις συνάθροισης, εκφράσεις *GROUP BY*, *DISTINCT* και αν καμία από τις στήλες δεν προκύπτει μέσα από κάποιο υπολογισμό (π.χ. *SALARY*12*), ενώ μπορούμε να εισάγουμε δεδομένα μόνο όταν στον ορισμό της δεν υπάρχουν συναρτήσεις συνάθροισης, εκφράσεις *GROUP BY*, *DISTINCT*, στήλες που προκύπτουν από υπολογισμό ή πεδία *NOT NULL* των πινάκων της που όμως δεν προβάλλονται στην όψη.

Στην περίπτωση που θέλουμε οι εισαγωγές ή οι ενημερώσεις εγγραφών που εκτελούμε να περιορίζονται μόνο σε δεδομένα που προκύπτουν από το πεδίο ορισμού της π.χ. στην EMPV20 μόνο για υπαλλήλους του τμήματος 20, εισάγουμε στον ορισμό της τον περιορισμό *WITH CHECK OPTION*.

```
CREATE VIEW empvu20
AS SELECT *
FROM employees
WHERE department_id = 20
WITH CHECK OPTION;
```

Έτσι μπορούμε να επεξεργαστούμε μόνο δεδομένα που προκύπτουν από το *SELECT* της όψης και οποιαδήποτε προσπάθεια σε δεδομένα εκτός αυτού του ορισμού αποτυγχάνει.

```
UPDATE empvu20
  SET    department_id = 10
  WHERE employee_id = 201;
UPDATE empvu20
  *
ERROR at line 1:
ORA-01402: view WITH CHECK OPTION where-clause violation
```

3.5 Διαγραφή Όψης

Για να διαγράψουμε μια όψη χρησιμοποιούμε την έκφραση *DROP VIEW view_name* και με αυτό τον τρόπο αφαιρούμε τον ορισμό της από την ΒΔ. Η ενέργεια αυτή δεν επηρεάζει τους πίνακες στους οποίους αναφερόταν η όψη, όμως επηρεάζει τυχόν άλλες όψεις που την περιλάμβαναν στους ορισμούς τους.

```
DROP VIEW empvu80;
View dropped.
```

3.6 Inline Όψεις

Μια inline όψη είναι ένα υποερώτημα εκφρασμένο με ψευδώνυμο το οποίο μπορεί να χρησιμοποιηθεί μέσα σε μια έκφραση SQL. Πιο συγκεκριμένα μπορούμε να δημιουργήσουμε μια inline όψη με ένα υποερώτημα στο *FROM* τμήμα της SQL. Με τον τρόπο αυτό η όψη αυτή μπορεί να χρησιμοποιηθεί και να γίνει αναφορά σε αυτή από το κυρίως ερώτημα. Για να γίνει πιο κατανοητός ο ορισμός αυτός ας εξετάσουμε το πιο κάτω παράδειγμα.

```
SELECT a.last_name, a.salary, a.department_id, b.maxsal
FROM employees a, (SELECT department_id, max(salary) maxsal
                   FROM employees
                   GROUP BY department_id) b
WHERE a.department_id = b.department_id
AND a.salary < b.maxsal;
```

Εδώ η inline όψη *b* επιστρέφει το αναγνωριστικό των τμημάτων καθώς και το μέγιστο μισθό ανά τμήμα μέσα από τον πίνακα *EMPLOYEES*. Στη συνέχεια με την χρήση του *WHERE a.department_id = b.department_id AND a.salary < b.salary* στο κυρίως ερώτημα επιλέγουμε να προβάλουμε τα ονόματα, τους μισθούς, το τμήμα και τον μέγιστο μισθό εκείνων των υπαλλήλων των οποίων ο μισθός είναι μικρότερος από τον μέγιστο μισθό του τμήματος που ανήκουν.

3.7 Top-n Ανάλυση

```
SELECT ROWNUM as SENIOR, E.last_name, E.hire_date
FROM (SELECT last_name, hire_date FROM employees
      ORDER BY hire_date) E
WHERE rownum <= 4;
```

Με top-n ερωτήματα μπορούμε να επιλέξουμε τις n-μεγαλύτερες ή n-μικρότερες εγγραφές ενός πίνακα με βάση μια συνθήκη (στην συνέχεια αυτές να συνδυαστούν με άλλες για περαιτέρω επεξεργασία). Παραδείγματα τέτοιου είδους ερωτημάτων είναι π.χ. Η εύρεση των τριών μεγαλύτερων μισθών ή των τεσσάρων πιο πρόσφατων προσληφθέντων υπαλλήλων. Ένα τέτοιου είδους ερώτημα αποτελείται από τμήματα που δείχνονται πιο κάτω

```
SELECT [column_list], ROWNUM
FROM (SELECT [column_list]
      FROM table
      ORDER BY Top-N_column)
WHERE ROWNUM <= N;
```

- ένα υποερώτημα ή inline όψη για την δημιουργία των ταξινομημένων αποτελεσμάτων. Το τμήμα αυτό πάντα περιέχει την έκφραση *ORDER BY* έτσι ώστε να διασφαλίζεται πάντα η σωστή βαθμονόμηση των εγγραφών. Για την εύρεση των μεγαλύτερων τιμών απαιτείται η χρήση του *DESC*
- ένα εξωτερικό ερώτημα το οποίο περιορίζει τον αριθμό των εγγραφών. Για να το πετύχουμε αυτό κάνουμε χρήση της ψευδοστήλης που υπάρχει ενσωματωμένη στα περισσότερα συστήματα ΒΔ (*rownum* για την Oracle, *row_number* για SQL Server κ.ο.κ). Η ψευδοστήλη αυτή αποδίδει ένα σειριακό αριθμό ξεκινώντας από το 1 σε κάθε εγγραφή του υποερωτήματος. Χρησιμοποιώντας την ψευδοστήλη αυτή στο *WHERE* μπορούμε να περιορίσουμε τον αριθμό των εγγραφών σε n.

Με βάση την ανάλυση του ορισμού παρατηρώντας το αρχικό παράδειγμα της παραγράφου βλέπουμε ότι επιστρέφουμε το όνομα και την ημερομηνία πρόσληψης των τεσσάρων πιο πρόσφατα προσληφθέντων υπαλλήλων από τον πίνακα *EMPLOYEES*. Στο παράδειγμα μας επίσης βλέπουμε και την χρήση μιας inline όψης για την παραγωγή του top-4 αποτελέσματος μας.

3.8 Ασκήσεις

1. Δημιουργήστε μια όψη με το όνομα *EMPLOYEES_VU* η οποία θα προβάλλει το αναγνωριστικό του υπαλλήλου, το όνομα του υπαλλήλου και τον αριθμό του τμήματος από τον πίνακα *EMPLOYEES*. Αλλάξτε τον τίτλο της στήλης *last_name* σε *EMPLOYEE*.
2. Επιλέξτε τα περιεχόμενα της όψης *EMPLOYEES_VU*.
3. Από την *EMPLOYEES_VU*, προβάλετε μόνο το όνομα του υπαλλήλου και τον αριθμό του τμήματος.
4. Δημιουργήστε μια όψη με όνομα *DEPT50* η οποία θα περιλαμβάνει το αναγνωριστικό του υπαλλήλου, το επώνυμο του υπαλλήλου και τον αριθμό του τμήματος για όλους τους υπαλλήλους του τμήματος 50. Ονομάστε τα ονόματα των στηλών της όψης *EMPNO*, *EMPLOYEE*, *DEPTNO*. Φροντίστε ώστε να μην δίνεται η δυνατότητα να αλλάξουν τμήμα οι υπάλληλοι μέσω της συγκεκριμένης όψης.
5. Προβάλετε την όψη της όψης *DEPT50*.
6. Προσπαθήστε να αλλάξετε τμήμα μέσω της όψης *DEPT50*.
7. Δημιουργήστε μία όψη *SALARY_VU* η οποία θα περιέχει το επώνυμο του υπαλλήλου, το όνομα του τμήματος, το μισθό και την κατηγορία του μισθού όλων των υπαλλήλων. Χρησιμοποιήστε τους πίνακες *EMPLOYEES*, *DEPARTMENTS* και *JOB_GRADES*. Ονομάστε τις στήλες *EMPLOYEE*, *DEPARTMENT*, *SALARY* και *GRADE*.

4 Άλλα Αντικείμενα των ΒΔ

4.1 Αντικείμενα ΒΔ

Εκτός από τους πίνακες και τις όψεις που έχουμε μελετήσει μέχρι τώρα μια ΒΔ μπορεί να περιλαμβάνει επιπλέον πολλά περισσότερα αντικείμενα όπως είναι οι sequences (ακολουθίες), τα indexes (ευρετήρια) και τα synonyms (συνώνυμα).

4.2 Ακολουθίες (sequences)

Μια ακολουθία είναι ένα αντικείμενο που δημιουργούν οι χρήστες και το οποίο μπορεί να διαμοιραστεί μεταξύ χρηστών για την παραγωγή μοναδικών ακεραίων. Η συνηθέστερη χρήση των ακολουθιών είναι στην χρήση πρωτευόντων κλειδίων σε πίνακες. Οι αριθμοί των ακολουθιών αποθηκεύονται και δημιουργούνται ανεξάρτητα από τους πίνακες, και για τον λόγο αυτό η ίδια ακολουθία μπορεί να χρησιμοποιηθεί σε πολλαπλούς πίνακες. Η σύνταξη για την δημιουργία μιας ακολουθίας¹ έχει ως εξής:

```
CREATE SEQUENCE sequence
    [INCREMENT BY n]
    [START WITH n]
    [{MAXVALUE n | NOMAXVALUE}]
    [{MINVALUE n | NOMINVALUE}]
    [{CYCLE | NOCYCLE}]
    [{CACHE n | NOCACHE}] ;
```

στην οποία η έκφραση

<i>Sequence</i>	δηλώνει το όνομα της ακολουθίας
<i>INCRIMENT BY n</i>	δηλώνει το βήμα της ακολουθίας μεταξύ των αριθμών που παράγονται
<i>START WITH n</i>	δηλώνει την πρώτη τιμή της ακολουθίας (από εκεί δηλαδή που ξεκινά)
<i>MAXVALUE n</i>	δηλώνει την μέγιστη τιμή της ακολουθίας
<i>NOMAXVALUE</i>	δηλώνει ότι δεν έχει οριστεί μέγιστη τιμή για την ακολουθία
<i>MINVALUE n</i>	δηλώνει την ελάχιστη τιμή της ακολουθίας
<i>NOMINVALUE</i>	δηλώνει ότι δεν έχει οριστεί ελάχιστη τιμή για την ακολουθία
<i>CYCLE NOCYCLE</i>	δηλώνει ότι η ακολουθία θα συνεχίσει να παράγει τιμές (από την αρχή) και μετά την μέγιστη ή την ελάχιστη τιμή
<i>CACHE n NOCACHE</i>	δηλώνει ότι θα δεσμευτούν ή όχι <i>n</i> τιμές στην μνήμη της ΒΔ

Στην πιο κάτω εικόνα δημιουργείται η ακολουθία *DEPT_DEPTID_SEQ* η οποία ξεκινά από τον αριθμό 120, έχει βήμα 10 και μέγιστη τιμή 9999.

¹ Εδώ η σύνταξη δημιουργίας της ακολουθίας βασίζεται στο RDBMS της Oracle. Ανάλογη είναι η σύνταξη και στα υπόλοιπα γνωστά συστήματα ΒΔ.

```
CREATE SEQUENCE dept_deptid_seq
    INCREMENT BY 10
    START WITH 120
    MAXVALUE 9999
    NOCACHE
    NOCYCLE;

Sequence created.
```

Για να μπορέσουμε να χρησιμοποιήσουμε την ακολουθία που χρησιμοποιήσαμε είναι απαραίτητο να μπορούμε να καλέσουμε την επόμενη παραγόμενη τιμή. Η σύνταξη της ενέργειας αυτής διαφέρει στα ΣΔΒΔ. Στην Oracle για παράδειγμα χρησιμοποιούμε την ψευδοστήλη *NEXTVAL* ως εξής *sequence.NEXTVAL*. Με τον τρόπο αυτό παράγεται μια νέα τιμή και η τρέχουσα τιμή είναι διαθέσιμη στην ψευδοστήλη *CURRVAL*. Με παρόμοιο τρόπο στον SQL Server² για να πάρουμε την επόμενη τιμή χρησιμοποιούμε την σύνταξη

```
CREATE SEQUENCE Test.CountBy1
    START WITH 1
    INCREMENT BY 1 ;

GO
```

```
SELECT NEXT VALUE FOR Test.CountBy1 AS FirstUse;
SELECT NEXT VALUE FOR Test.CountBy1 AS SecondUse;
```

Ένα ολοκληρωμένο παράδειγμα χρήσης ακολουθίας στην Oracle κατά την εισαγωγή μιας νέας εγγραφής δείχνεται παρακάτω.

```
INSERT INTO departments (department_id,
    department_name, location_id)
VALUES (dept_deptid_seq.NEXTVAL,
    'Support', 2500);

1 row created.
```

Εισάγεται μια νέα εγγραφή στον πίνακα *DEPARTMENTS* και στην θέση του *DEPARTMENT_ID* εισάγεται η τιμή που παράγεται από την ακολουθία *DEPT_DEPTID_SEQ*.

Για την διαγραφή μιας ακολουθίας υπάρχει η σύνταξη *DROP SEQUENCE*.

```
DROP SEQUENCE dept_deptid_seq;

Sequence dropped.
```

² Στην PostgreSQL η σύνταξη έχει ως εξής *nextval(sequence_name)*, *currval(sequence_name)* κ.ο.κ.
<https://www.postgresql.org/docs/8.1/static/functions-sequence.html>

4.3 Ευρετήρια (indexes)

Όπως έχουμε δει και στην θεωρία τα ευρετήρια μπορούν να επιταχύνουν την ανάκτηση των εγγραφών χρησιμοποιώντας δείκτες. Σκοπός της χρήσης τους είναι να μειώσουν στο δίσκο το I/O που απαιτείται για την παραγωγή του αποτελέσματος. Τα ευρετήρια συντηρούνται από το εκάστοτε ΣΔΒΔ και δεν απαιτείται παρέμβαση από τον χρήστη, ενώ είναι φυσικά και λογικά ανεξάρτητα αντικείμενα με αποτέλεσμα να μπορούν να τροποποιηθούν ή και να διαγραφούν χωρίς να επηρεάζεται ο πίνακας για τον οποίον έχουν δημιουργηθεί.

Η σύνταξη για να δημιουργηθεί ένα ευρετήριο έχει ως εξής:

```
CREATE INDEX index  
ON table (column[, column]...);
```

Η πιο συγκεκριμένα για την βελτιστοποίηση του ερωτήματος στον πεδίο *LAST_NAME* του πίνακα *EMPLOYEES* φτιάχνεται ένα ευρετήριο *EMP_LAST_NAME_IDX*

```
CREATE INDEX emp_last_name_idx  
ON employees(last_name);  
Index created.
```

Η προσθήκη ενός ευρετηρίου βέβαια σε έναν πίνακα δεν σημαίνει αυτόματα και την βελτίωση του χρόνου απόκρισης του. Καθώς και άλλες εντολές εκτελούνται στις εγγραφές του πίνακα, είναι απαραίτητο να γίνονται και οι απαραίτητες ενημερώσεις σε ευρετήρια που τυχόν υπάρχουν. Όσο λοιπόν περισσότερα τα ευρετήρια σε έναν πίνακα τόσο και περισσότερες ενημερώσεις που πρέπει να γίνουν. Καταλήγουμε λοιπόν ότι δεν δημιουργούμε ευρετήρια αυθαίρετα χωρίς κανένα κριτήριο αλλά ακολουθώντας κάποιους πρακτικούς κανόνες, όπως:

- α. όταν μια στήλη περιέχει μεγάλο εύρος τιμών
- β. όταν μια στήλη περιέχει μεγάλο αριθμό *NULLs*
- γ. όταν μια ή περισσότερες στήλες χρησιμοποιούνται στην έκφραση του *WHERE* κατά την σύνδεση πινάκων
- δ. όταν οι πίνακες διαθέτουν μεγάλο αριθμό εγγραφών και περιμένουμε να ανακτούμε περίπου το 2-4% των τιμών

Με παρόμοια λογική εμπειρικοί κανόνες για την ΜΗ δημιουργία ευρετηρίων είναι:

- α. όταν ο πίνακας διαθέτει λίγες εγγραφές

β. όταν δεν χρησιμοποιούνται οι στήλες συχνά σε κάποια συνθήκη

γ. όταν ανακτούμε συνήθως μεγάλο αριθμό εγγραφών (>4%)

δ. όταν ο πίνακας ενημερώνεται πολύ συχνά ++ removing an index

Στα ευρετήρια δεν δίνεται συνήθως η δυνατότητα να μεταβάλουμε τις παραμέτρους τους. Στην περίπτωση λοιπόν που θέλουμε να κάνουμε κάτι τέτοιο πρέπει να διαγράψουμε το ευρετήριο και να το ξαναδημιουργήσουμε. Αυτό γίνεται με την χρήση της εντολής

DROP INDEX *index*;

όπου *index* είναι το όνομα του ευρετηρίου π.χ. για να διαγράψουμε το ευρετήριο *UPPER_LAST_NAME_IDX* δίνουμε:

**DROP INDEX upper_last_name_idx;
Index dropped.**

4.4 Συνώνυμα (synonyms)

Τα συνώνυμα είναι αντικείμενα των ΒΔ με τα οποία μας δίνεται η δυνατότητα να αναφερθούμε σε αρχικά αντικείμενα με ένα άλλο όνομα. Με την χρήση τους λοιπόν μπορούμε να προσθέσουμε ένα λογικό αφαιρετικό επίπεδο για την προστασία των εφαρμογών από τυχόν αλλαγές στο όνομα ή στην θέση των “πηγαίων” αντικείμενων. Επίσης μπορούμε να αναφερθούμε με πιο απλό τρόπο σε αντικείμενα που υπάρχουν σε άλλους χρήστες, χωρίς να είναι απαραίτητη η ύπαρξη προθέματος αλλά και δημιουργήσουμε μικρότερα ονόματα για την απόκρυψη άλλων μεγαλύτερων. Η σύνταξη για την δημιουργία ενός συνώνυμου είναι:

CREATE SYNONYM synonym FOR object;

όπου *synonym* το όνομα του συνώνυμου και *object* το αντικείμενο για το οποίο δημιουργείται το συνώνυμο. Στην παρακάτω εικόνα δημιουργούμε το συνώνυμο *D_SUMI* για την όψη *DEPT_SUM_VU* και στην συνέχεια το διαγράφουμε.

**CREATE SYNONYM d_sum
FROM dept_sum_vu;

DROP SYNONYM d_sum;**

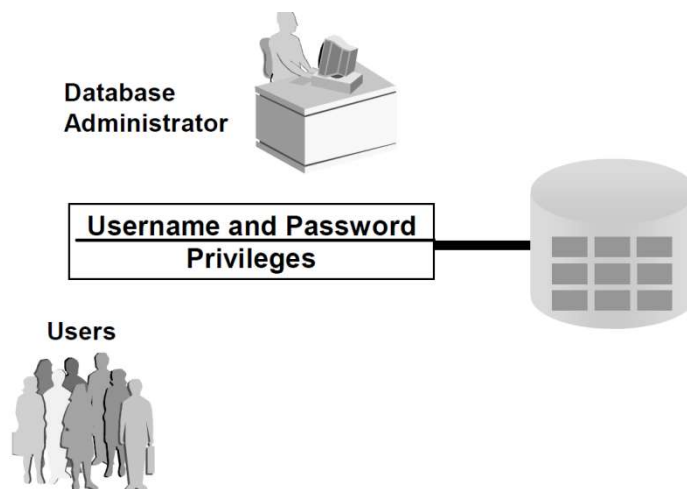
Κατά την διαγραφή του συνωνύμου δεν επηρεάζεται το “πηγαίο” αντικείμενο. Μπορούμε να δημιουργήσουμε συνώνυμα για πίνακες, όψεις, ακολουθίες, αποθηκευμένες διαδικασίες κ.α.³

4.5 Ασκήσεις

1. Δημιουργήστε την ακολουθία *DEPT_ID_SEQ* η οποία θα ξεκινά από το 200 και θα έχει μέγιστη τιμή 1000. Η ακολουθία θα αυξάνεται κατά 10 αριθμούς κάθε φορά.
2. Πραγματοποιήστε μια εισαγωγή δύο εγγραφών στον πίνακα *DEPT* παράγοντας την τιμή για την στήλη *ID* από την ακολουθία που δημιουργήσατε στην Άσκηση 1. Προσθέστε με αυτό τον τρόπο δύο νέα τμήματα το “Education” και το “Administration”. Επιβεβαιώστε το αποτέλεσμα.
3. Δημιουργήστε ένα μη-μοναδικό ευρετήριο στη στήλη *DEPT_ID* του πίνακα *EMPLOYEES*.
4. Δημιουργήστε ένα συνώνυμο για τον πίνακα *EMPLOYEES* με το όνομα *EMP_SYN*. Χρησιμοποιήστε το συνώνυμο αυτό για επιλέξετε τον υπάλληλο με *ID=1*.

5 Έλεγχος Πρόσβασης

5.1 Έλεγχος Πρόσβασης Χρηστών



Σε ένα περιβάλλον πολλαπλών χρηστών είναι απαραίτητη η ύπαρξη ασφαλούς πρόσβασης και χρήσης της ΒΔ. Για τον λόγο αυτό όπως έχουμε δει και στην θεωρία ένα σύστημα ΒΔ πρέπει να περιέχει αξιόπιστους μηχανισμούς πιστοποίησης και εξουσιοδότησης

5.2 Δικαιώματα

³ Το πλήθος και το είδος των αντικειμένων για τα οποία μπορούμε να δημιουργήσουμε συνώνυμα εξαρτάται κάθε φορά από το ΣΔΒΔ που εργαζόμαστε.

Παρέχοντας προνόμια σε ένα χρήστη αυτός αποκτά την δυνατότητα να εκτελεί συγκεκριμένες sql εντολές. Ο διαχειριστής της βάσης δεδομένων (DBA) είναι αυτός που εκχωρεί τα προνόμια στους χρήστες ώστε να έχουν πρόσβαση στην ΒΔ και στα αντικείμενα αυτής. Από την πλευρά των χρηστών αυτοί απαιτείται να έχουν προνόμια ώστε μπορούν να συνδεθούν αλλά και εκτελούν τις ενέργειες που χρειάζεται σε αντικείμενα. Επίσης οι χρήστες μπορεί να έχουν το προνόμιο να παρέχουν προνομία σε άλλους χρήστες. Οι διαχειριστές της ΒΔ έχουν υψηλού επιπέδου συστημικά προνόμια ώστε να μπορούν να δημιουργούν/διαγράφουν χρήστες, να παίρνουν αντίγραφα ασφαλείας, να τροποποιούν πίνακες κ.α.

5.3 Δημιουργία Χρήστη

```
CREATE USER user  
IDENTIFIED BY password;
```

```
CREATE USER scott  
IDENTIFIED BY tiger;  
User created.
```

Εξ ορισμού ο χρήστης ενός συστήματος ΒΔ είναι κάποιος ο οποίος θα κάνει χρήση των υπηρεσιών που παρέχονται από το σύστημα αυτό. Η σύνταξη για την δημιουργία ενός νέου χρήστη καθώς δεν έχει οριστεί σε κάποια από τις τυποποιήσεις της SQL συντάσσεται με διαφορετικό τρόπο σε κάθε RDBMS. Για παράδειγμα στο περιβάλλον της Oracle στο οποίο και στηριζόμαστε για το μάθημα μας δεν γίνεται διάκριση μεταξύ του χρήστη και του σχήματος της ΒΔ, ενώ στην IBM DB2 όπως και στην PostgreSQL οι χρήστες ορίζονται σε επίπεδο ΛΣ. Η βασική σύνταξη για την δημιουργία ενός χρήστη στην Oracle είναι όπως δείχνεται στην πιο πάνω εικόνα στην οποία ακολουθεί παράδειγμα κατά το οποίο δημιουργείται ο χρήστης *scott* με κωδικό *tiger*.

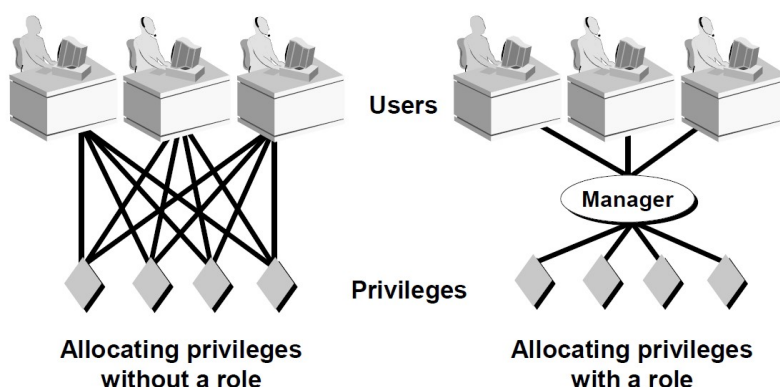
5.4 Συστημικά Δικαιώματα

```
GRANT privilege [, privilege...]  
TO user [, user| role, PUBLIC...];
```

```
GRANT create session, create table,  
create sequence, create view  
TO scott;  
Grant succeeded.
```

Μετά την δημιουργία ενός χρήστη συνήθως ο διαχειριστής εκχωρεί κάποια συστημικά προνόμια στον χρήστη ώστε αυτός να μπορεί να συνδεθεί και να εκτελέσει βασικές ενέργειες στην ΒΔ. Τα προνόμια που πρέπει να εκχωρηθούν διαφέρουν μεταξύ των ΣΔΒΔ. Στην παραπάνω εικόνα που αναφέρεται σε περιβάλλον Oracle, δίνονται στον χρήστη *scott* τα προνόμια *create session*, *create table*, *create sequence* και *create view*, έτσι ώστε να μπορεί να συνδεθεί στην ΒΔ και να μπορεί να δημιουργεί πίνακες, ακολουθίες και όψεις.

5.5 Ρόλοι



Ο ρόλος είναι μια λογική ομάδα προνομίων. Ο ρόλος μπορεί να εκχωρηθεί σε χρήστες ακριβώς όπως και τα προνόμια κάτι που κάνει ευκολότερη την εκχώρηση και την αναίρεση στην περίπτωση που θέλουμε να διαχειριστούμε πολλά προνόμια και χρήστες. Ένας χρήστης μπορεί να έχει πολλούς ρόλους, ενώ πολλοί χρήστες μπορούν να ανήκουν στον ίδιο ρόλο. Συνήθως ένας ρόλος δημιουργείται από τον διαχειριστή, με βάση μια λογική απαίτηση κάποιας εφαρμογής. Για να δημιουργηθεί ένας ρόλος η σύνταξη είναι

```
CREATE ROLE name;
```

όπου *name* το όνομα του ρόλου. Μετά την δημιουργία του μπορεί να χρησιμοποιηθεί ο όρος *GRANT* για την εκχώρηση του σε χρήστη. Για παράδειγμα, πιο κάτω δημιουργούμε τον ρόλο *manager*, στη συνέχεια εκχωρούμε τα προνόμια *create table*, *create view* στον ρόλο και τέλος δίνουμε τον ρόλο στους χρήστες *DEHAAN*, *KOCHHAR*.

```
CREATE ROLE MANAGER;
```

```
GRANT CREATE TABLE, CREATE VIEW TO MANAGER;
```

```
GRANT MANAGER TO DEHAAN, KOCHAAR;
```

Οι χρήστες αυτοί μπορούν να δημιουργούν πίνακες και όψεις.

5.6 Προνόμια σε Αντικείμενα

Ένα προνόμιο σε αντικείμενο δίνει το δικαίωμα εκτέλεσης μιας ενέργειας σε αυτό το αντικείμενο. Κάθε αντικείμενο έχει διαφορετικό σύνολο από προνόμια που μπορούν να εκχωρηθούν. Ο ιδιοκτήτης ενός αντικειμένου έχει εξ ορισμού όλα τα προνόμια στο αντικείμενο αυτό, και μπορεί να εκχωρήσει τα προνόμια των αντικειμένων που του ανήκουν σε άλλους χρήστες ή ρόλους. Στην περίπτωση που εκχωρεί κάποιο προνόμιο με την χρήση του *GRANT OPTION* τότε με την σειρά του ο παραλήπτης μπορεί να το εκχωρήσει περαιτέρω (προνόμιο μεταφοράς προνομίων). Η σύνταξη για την εκχώρηση προνομίων δείχνεται παρακάτω.

```
GRANT object_priv [(columns)]
ON object
TO {user|role}
[WITH GRANT OPTION];
```

όπου

<i>object_priv</i> :	αναφέρεται στο προνόμιο που θα εκχωρηθεί
<i>columns</i> :	καθορίζει το πεδίο του πίνακα, όψης για το οποίο θα εκχωρηθεί το προνόμιο
<i>ON object</i> :	είναι το αντικείμενο πάνω στο οποίο εκχωρείται το προνόμιο
<i>TO</i> :	καθορίζει σε ποιόν θα εκχωρηθεί το προνόμιο
<i>WITH GRANT OPTION</i> :	καθορίζει αν ο παραλήπτης θα μπορεί να εκχωρεί το προνόμιο σε άλλους

Στην πιο κάτω εικόνα εκχωρείτε το προνόμιο του *select* πάνω στον πίνακα *employees* στους χρήστες *sue*, *rich*.

```
GRANT SELECT ON employees TO sue, rich;
```

ενώ πιο κάτω το προνόμιο του *update* στις στήλες *department_name*, *location_id* του πίνακα *departments* στο χρήστη *scott* και στον ρόλο *manager*.

```
GRANT UPDATE (department_name, location_id) ON departments TO scott, manager;
```

Μια άλλη εκχώρηση όπως έχουμε δει έχει να κάνει με το *GRANT OPTION*, όπου στο παρακάτω παράδειγμα ο χρήστης *scott* μπορεί να εκχωρήσει περαιτέρω τα προνόμια που λαμβάνει. Πρέπει βέβαια στις περιπτώσεις αυτές να έχουμε πάντα υπόψη ότι για να ισχύει ένα προνόμιο πρέπει σύμφωνα με το γράφημα προνομίων, αυτό να πηγάζει από την ρίζα, Στην περίπτωση δηλαδή που *scott* δώσει το προνόμιο στον *smith* και μετά του αφαιρεθεί το προνόμιο τότε αυτόματα αφαιρείται και από τον *smith*.

```
GRANT SELECT, UPDATE ON departments TO scott WITH GRANT OPTION;
```

Η αναίρεση προνομίων όπως έχουμε δει και στην θεωρία γίνεται με την χρήση του *REVOKE*.

```
REVOKE {privilege [,privilege]}
ON object
FROM {user [,user]}
```

για παράδειγμα

5.7 Ασκήσεις

1. Δημιουργήστε ένα επιπλέον χρήστη με το όνομα *GUEST_HR* και κωδικό *GUEST_PASSWD*. Εκχωρήστε του το κατάλληλο προνόμιο ώστε να μπορεί να συνδεθεί στην ΒΔ σας, αλλά και να μπορεί να διαβάσει από τον πίνακα *DEPARTMENTS*.
2. Επιλέξτε κάποιο συμμαθητή σας και ζητήστε του να εισάγει στον δικό του πίνακα *DEPARTMENTS* το τμήμα *HUMAN RESOURCES* με αναγνωριστικό 510, ενώ εσείς εισάγετε στον δικό σας πίνακα το τμήμα *EDUCATION* με αναγνωριστικό

500. Ζητήστε από τον συμμαθητή σας να συνδεθεί στην ΒΔ σας σαν χρήστης GUEST_HR και να δει τα δεδομένα από τον πίνακα. Το ίδιο κάνετε και εσείς στην δική του ΒΔ.
3. Δημιουργήστε έναν ακόμη χρήστη GUEST_HR2 με κωδικό GUEST_PASSWD2. Σαν χρήστης HR εκχωρήστε το προνόμιο να μπορεί να διαβάσει ο χρήστης GUEST_HR τον πίνακα *EMPLOYEES* συνοδευόμενο από το προνόμιο μεταφοράς προνομίων. Στην συνέχεια συνδεθείτε σαν GUEST_HR και εκχωρήστε το προνόμιο ώστε να μπορεί να διαβάσει ο χρήστης GUEST_HR2 τον πίνακα *EMPLOYEES*. Επιβεβαιώστε τα αποτελέσματα εκχώρησης προνομίων.
 4. Αναιρέστε το προνόμιο ανάγνωσης του *EMPLOYEES* από τον χρήστη GUEST_HR. Τι συμβαίνει με το προνόμιο του χρήστη GUEST_HR2 πάνω στον πίνακα διατηρείται ή όχι (και γιατί);
 5. Εκχωρήστε ξανά το προνόμιο ανάγνωσης στον πίνακα *EMPLOYEES* στους χρήστες GUEST_HR, GUEST_HR2 με GRANT OPTION. Στη συνέχεια συνδεθείτε διαδοχικά ως GUEST_HR και εκχωρήστε το προνόμιο ανάγνωσης στον πίνακα *EMPLOYEES* στον χρήστη GUEST_HR2 και το αντίστροφο. Τέλος συνδεθείτε ως χρήστης HR και αναιρέστε το προνόμιο ανάγνωσης που δώσατε στον χρήστη GUEST_HR. Τι συμβαίνει αυτή τη φορά με το αντίστοιχο προνόμιο του GUEST_HR2 και γιατί; Επιβεβαιώστε το αποτέλεσμα.