Alocarea și revocarea drepturilor. Gestiunea tranzacțiilor

- Cum putem controla accesul utilizatorilor la baza de date
- Care sunt categoriile de drepturi ce se pot atribui utilizatorilor
- Cum acordăm drepturi utilizatorilor la anumite obiecte ale bazei de date
- Cum revocăm anumite drepturi ale utilizatorilor
- Ce sunt rolurile
- Cum se creează și cum se șterge un rol
- Cum se alocă drepturi rolurilor
- Cum se acordă un rol unui utilizator
- Ce este o tranzacție
- Când începe o tranzacție
- Când se termină o tranzacție
- Cum devin permanente modificările făcute într-o tranzacție asupra bazei de date
- Cum se anulează modificările făcute într-o tranzacție asupra bazei de date

Drepturi şi roluri

- V-aţi întrebat vreodată ce ar însemna ca elevii dintr-o şcoală să aibă acces liber la catalog şi să poată face orice modificare doresc în catalog? Dar dacă orice utilizator conectat la internet ar avea acces nerestricţionat la baza de date a CIA, NASA, a unei bănci şi aşa mai departe?
- Evident, în viaţa reală accesul în anumite locuri este restricţionat. Dacă faci parte dintr-un anumit grup restrâns de persoane, ca de exemplu angajaţii băncii, poţi avea acces în anumite zone restricţionate sau la anumite resurse la care alte persoane nu au acces.
- Ca şi în lumea reală şi în cazul bazelor de date trebuie să putem defini o serie de drepturi pentru utilizatorii bazei de date, sau să restricţionăm accesul acestora la anumite obiecte ale bazei de date.
- Controlul securităţii în Oracle se asigură prin specificarea: utilizatorilor bazei de date, schemelor, privilegiilor (drepturilor) şi rolurilor.

Utilizatorii bazei de date şi schemele

- Fiecare bază de date are o listă de nume de utilizatori. Pentru a accesa baza de date un utilizator trebuie să folosească o aplicaţie şi să se conecteze cu un nume potrivit. Fiecărui nume de utilizator îi este asociată o parolă. Orice utilizator are un domeniu de securitate care determină privilegiile şi rolurile, cota de spaţiu pe disc alocat şi limitele de resurse ce le poate utiliza (timp CPU etc).

Privilegiile

- Privilegiul este dreptul unui utilizator de a executa anumite instrucţiuni SQL. Privilegiile pot fi:
  - **privilegii de sistem** - permit utilizatorilor să execute o gamă largă de instrucţiuni SQL, ce pot modifica datele sau structura bazei de date. Aceste privilegii se atribuie de obicei numai administratorilor bazei de date.
  - **privilegii de obiecte** - permit utilizatorilor să execute anumite instrucţiuni SQL numai în cadrul schemei sale, şi nu asupra întregii baze de date.
- Acordarea privilegiilor reprezintă modalitatea prin care acestea pot fi atribuite utilizatorilor. Există două căi de acordare ***explicit*** (privilegiile se atribuie în mod direct utilizatorilor) şi ***implicit*** (prin atribuirea acestora unor roluri, care la rândul lor sunt acordate utilizatorilor).

Rolurile

- Rolurile sunt grupe de privilegii, care se atribuie utilizatorilor sau altor roluri. Rolurile permit:
- Reducerea activităţilor de atribuire a privilegiilor. Administratorul bazei de date în loc să atribuie fiecare privilegiu tuturor utilizatorilor va atribui aceste privilegii unui rol, care apoi va fi disponibil utilizatorilor;
- Manipularea dinamică a privilegiilor. Dacă se modifică un privilegiu de grup, acesta se va modifica în rolul grupului. Automat modificarea privilegiului se propagă la toţi utilizatorii din grup;
- Selectarea disponibilităţilor privilegiilor. Privilegiile pot fi grupate pe mai multe roluri, care la rândul lor pot fi activate sau dezactivate în mod selectiv;
- Proiectarea unor aplicaţii inteligente. Se pot activa sau dezactiva anumite roluri funcţie de utilizatorii care încearcă să utilizeze aplicaţia.
- Un rol poate fi creat cu parolă pentru a preveni accesul neautorizat la o aplicaţie. Această tehnică permite utilizarea parolei la momentul pornirii aplicaţiei, apoi utilizatorii pot folosi aplicaţia fără să mai cunoască parola.

- Pentru acordarea unui drept unui anumit utilizator **vasile** se va folosi comanda **GRANT**. De exemplu, pentru a se conecta la baza de date, un utilizator trebuie să aibă permisiunea de a crea o sesiune. Acest drept se alocă de către un utilizator privilegiat (utilizatorul system de exemplu) prin comanda
- **GRANT CREATE SESSION TO vasile**
- Acum utilizatorul **vasile** se poate conecta la baza de date.
- Revocarea unui drept unui anumit utilizator se face folosind comanda **REVOKE** ca în exemplul următor:
- **REVOKE CREATE SESSION FROM vasile**

Drepturile de sistem

- Un drept de system permite unui utilizator să efectueze anumite operaţii asupra bazei de date precumexecutarea comenzilor DDL. Cele mai uzuale drepturi system sunt prezentate în tabelul următor.

Drept	Permite…
CREATE SESSION	conectarea la baza de date
CREATE SEQUENCE	crearea secvenţelor
CREATE SYNONYM	crearea sinonimelor
CREATE TABLE	crearea tabelelor
CREATE ANY TABLE	crearea unor tabele în orice schemă, nu doar în propria schemă
DROP TABLE	ştergerea tabelelor
DROP ANY TABLE	ştergerea unor tabele din orice schemă nu doar din schema proprie
CREATE PROCEDURE	crearea de proceduri memorate
EXECUTE ANY PROCEDURE	executarea unei proceduri în orice schemă
CREATE USER	crearea de utilizatori
DROP USER	ştergerea utilizatorilor
CREATE VIEW	crearea vederilor

Privilegii sistem

Acordarea drepturilor de sistem

- După cum am precizat acordarea drepturilor se face folosind comanda **GRANT**. În exemplul următor se acordă câteva drepturi sistem utilizatorului **ion**:

GRANT CREATE SESSION, CREATE USER, CREATE TABLE TO ion;

- Se poate de asemenea folosi opţiunea **WITH ADMIN OPTION** care permite unui utilizator să aloce şi el drepturile primite cu această opţiune, mai departe, altor utilizatori:

GRANT EXECUTE ANY PROCEDURE TO ion WITH ADMIN OPTION;

- Dreptul acordat utilizatorului **ion**, de a executa orice procedură poate fi acordată de acesta mai departe utilizatorului **george**. Pentru aceasta **ion** se va conecta la baza de date folosind comanda

CONNECT ion/test

- unde **ion** este username-ul iar **test** este parola şi apoi va acorda dreptul lui **george**:

GRANT EXECUTE ANY PROCEDURE TO george;

- Un drept se poate aloca tuturor utilizatorilor bazei de date folosin opţiunea **PUBLIC** ca în următorul exemplu:

CONNECT system/manager

GRANT EXECUTE ANY PROCEDURE TO PUBLIC;

- În acest moment orice utilizator al bazei de date are dreptul de a executa o procedură în orice schemă.

Drepturile la nivel de obiect

- Un drept la nivel de obiect permite unui utilizator să execute anumite acţiuni asupra obiectelor bazei de date, ca de exemplu executarea anumitor comenzi **DML** pe tabelele bazei de date. De exemplu **GRANT INSERT ON adm.elevi** permite unui utilizator să insereze linii noi în tabela **elevi** din schema **adm**. Cele mai des întâlnite drepturi la nivel de obiect sunt prezentate în tabelul următor:

Drept	Permite …
SELECT	Interogarea tabelei
INSERT	Inserarea de noi linii în tabelă
UPDATE	Modificarea valorilor din tabelă
DELETE	Ştergerea datelor din tabelă
EXECUTE	Executarea unor proceduri memorate

Privilegii la nivel de obiect

Acordarea drepturilor la nivel de obiect

- Comanda **GRANT** se foloseşte pentru a acorda drepturi. Exemplul următor acordă utilizatorului **ion** dreptul de **SELECT**, **INSERT**, şi **UPDATE** pe tabela elevi şi dreptul de **SELECT** asupra tabelei **angajaţi**:

GRANT SELECT, INSERT, UPDATE ON adm.elevi TO ion;

GRANT SELECT ON profesori.angajati TO ion;

- Următoarea comandă permite utilizatorului **ion** să modifice doar valorile din coloanele **prenume** şi **adresa**, din tabela **elevi**, utilizatorului **ion**:

GRANT UPDATE (prenume,adresa) ON adm.elevi TO ion;

- Folosind opţiunea **WITH GRANT OPTION** veţi permite utilizatorului să acorde mai departe dreptul primit şi altor utilizatori:

GRANT SELECT ON adm.elevi TO ion WITH GRANT OPTION;

- Dreptul de a interoga tabela adm.elevi poate fi acum acordat de către **ion** oricărui alt utilizator:

CONNECT ion/test

GRANT SELECT ON adm.elevi TO george;

- **Revocarea drepturilor la nivel de obiect** se va face folosind comanda **REVOKE**. Următoarea comandă revocă dreptul de inserare de noi linii la tabela elevi utilizatorului **ion**:

REVOKE INSERT ON elevi FROM ion;

- Comanda va fi rulată din contul **adm**.
- ***Observaţie!*** Dacă am acordat un drept unui utilizator **A** folosind opţiunea **WITH GRANT OPTION**, iar acest utilizatorul **A** a acordat şi el la rândul lui dreptul altor utilizatori **B**, **C** şi **D**, atunci când vom revoca dreptul utilizatorului **A**, va fi revocat automat acel drept şi tuturor utilizatorilor cărora utilizatorul A le-a acordat acel drept, respectiv utilizatorilor **B**, **C** şi **D**.

Gestiunea rolurilor

- După cum am precizat la începutul capitolului, putem crea un rol, prin intermediul căruia vom putea acorda drepturi unui grup de utilizatori având rolul respectiv, lucru mult mai uşor decât acordarea drepturilor fiecărui utilizator separat.
- De exemplu, în loc să acordăm drepturi de **select**, **insert** şi **update** mai multor utilizatori:

GRANT SELECT, INSERT, UPDATE ON adm.elevi TO ion;

GRANT SELECT, INSERT, UPDATE ON adm.elevi TO vasile;

GRANT SELECT, INSERT, UPDATE ON adm.elevi TO gheorghe;

GRANT SELECT, INSERT, UPDATE ON adm.elevi TO maria;

GRANT SELECT, INSERT, UPDATE ON adm.elevi TO alin;

- E mai comod să creăm un rol, să acordăm drepturi pentru acest rol şi apoi să acordăm rolul respectiv celor cinci utilizatori. Vom scrie aşadar:

  CREATE ROLE profi;

  GRANT SELECT, INSERT, UPDATE ON adm.elevi TO profi;

  GRANT profi TO ion, vasile, gheorghe, maria, alin;

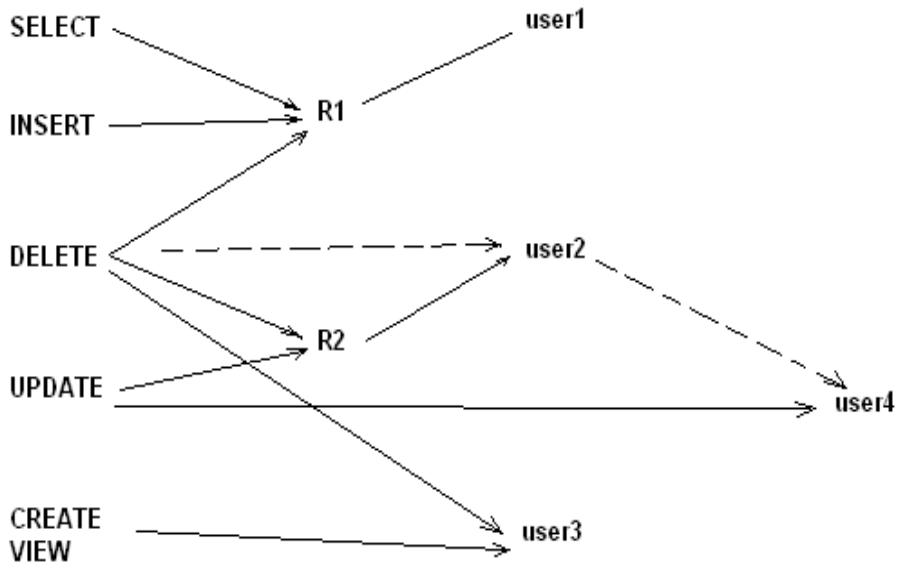
- În orice moment putem şterge un rol folosind comanda **DROP ROLE**. Aceasta va duce la revocarea tuturor drepturilor acordate utilizatorilor prin intermediul acestui rol.

- Să dăm un exemplu mai complex de acordare a drepturilor şi privilegiilor. Să presupunem că rulăm pe rând următoarele comenzi:

```
CONNECT hr/test;
CREATE ROLE r1;
CREATE ROLE r2;
GRANT SELECT, INSERT, DELETE ON hr.elevi TO r1
WITH GRANT OPTION;
GRANT DELETE, UPDATE ON hr.elevi TO r2
WITH GRANT OPTION;
GRANT r1 TO user1
GRANT r2 TO user2
GRANT CREATE VIEW TO user3 WITH GRANT OPTION
GRANT DELETE ON hr.elevi TO user3
GRANT UPDATE ON hr.elevi TO user4

CONNECT user2/pas2
GRANT DELETE ON hr.elevi TO user4
GRANT UPDATE ON hr.elevi TO user4
```

- În acest moment utilizatorii au următoarele drepturi :

UTILIZATOR	DREPT
user1	SELECT, INSERT, DELETE ON hr.elevi
user2	DELETE, UPDATE ON hr.elevi
user3	DELETE ON hr.elevi CREATE VIEW
user4	DELETE, UPDATE ON hr.elevi



Schema de acordare a drepturilor

- Dacă acum ştergem rolul **r2**:

DROP ROLE r2

- utilizatorul **user2** va pierde dreptul de **DELETE** şi **UPDATE** asupra tabelei **hr.elevi**, şi prin intermediul său va pierde dreptul de **DELETE** şi utilizatorul **user4**, care a primit acest drept de la **user2**. Deşi user4 a primit de la user2 şi dreptul de **UPDATE**, el nu va pierde acest drept deoarece a primit acest drept şi direct de la utilizatorul **SYSTEM**. Aşadar după ştergerea rolului **r2**, drepturile utilizatorilor sunt următoarele:

UTILIZATOR	DREPT
user1	SELECT, INSERT, DELETE ON hr.elevi
user2	-
user3	DELETE ON hr.elevi CREATE VIEW
user4	UPDATE ON hr.elevi

Gestiunea tranzacţiilor

- O tranzacţie este un grup de comenzi SQL care sunt văzute ca o singură unitate. Imaginaţi-vă o tranzacţie ca un grup de comenzi SQL care nu pot fi separate, şi al căror efect este în întregime salvat în baza de date, fie este în întregime anulat. Să ne gândim de exemplu la efectuarea unui transfer bancar dintr-un cont în alt cont.
- O comandă **UPDATE** va efectua operaţia de scădere a sumei de bani tranzacţionată dintr-un cont, iar o altă comandă **UPDATE** va adăuga suma respectivă la cel de al doilea cont. Dacă ambele operaţii decurg normal fără probleme, atunci ele vor deveni **ambele** permanente.
- Dacă una dintre aceste două comenzi eşuează (de exemplu nu poate fi contactată banca în care se depun banii) atunci ambele comenzi vor fi anulate. E normal să renunţăm la scăderea sumei de bani dintr-un cont, dacă aceştia nu pot fi depuşi în celălalt cont, în caz contrar ar duce la pierderea banilor respectivi.

- În general o tranzacţie poate fi formată din mai multe comenzi **INSERT**, **UPDATE**, şi **DELETE**.
- Pentru a face permanentă o tranzacţie folosiţi comanda **COMMIT**. Dacă doriţi să renunţaţi la modificările efectuate în cadrul unei tranzacţii trebuie să rulaţi o comandă **ROLLBACK**.
- Comanda **ROLLBACK** fără nici un parametru, încheie tranzacţia curentă şi renunţă la toate modificările făcute în cadrul acestei tranzacţii. Aveţi însă posibilitatea definirii în cadrul unei tranzacţii a unui aşa numit punct de întoarcere, sau punct de salvare. Odată definit un astfel de punct de salvare, veţi putea renunţa doar la o parte din modificările făcute în cadrul tranzacţiei curente.
- Definirea unui punct de revenire se face cu comanda **SAVEPOINT** având sintaxa:

  SAVEPOINT *nume_punct_de_revenire*
- Revenirea la un punct de revenire se face cu comanda ROLLBACK astfel:

  ROLLBACK TO *nume_punct_de_revenire*
- Definirea punctelor de revenire este utilă în cazul unor tranzacţii mari, când în cazul în care faceţi o greşeală nu trebuie să renunţaţi la toate operaţiile din cadrul tranzacţiei ci doar la o parte dintre acestea.

- O tranzacţie fiind un grup de comenzi SQL tratate ca un întreg, trebuie să stabilim unde începe o tranzacţie şi unde se termină aceasta.
- O tranzacţie începe la întâlnirea unuia dintre următoarele evenimente:
- În momentul conectării la baza de date şi la începerea rulării primei comenzi **DML** (**INSERT**, **UPDATE**, **DELETE**).
- La terminarea unei tranzacţii anterioare şi rularea următoarei comenzi **DML**.

- O tranzacţie se termină când apare unul dintre următoarele evenimente:
- La executarea unei comenzi **COMMIT** sau **ROLLBACK** (fără nici un parametru, întrucât **ROLLBACK TO ...** nu termină tranzacţia ci doar revine la un punct precizat din cadrul tranzacţiei curente)
- La executarea unei comenzi **DDL** (**CREATE**, **ALTER**, **DROP**, **RENAME**, **TRUNCATE**), caz în care este executată automat comanda **COMMIT**.
- La executarea unei comenzi **DCL** (**GRANT** sau **REVOKE**) caz în care este executată automat comanda **COMMIT**.
- Vă deconectaţi de la baza de date. Dacă ieşiţi normal din **SQL*Plus** cu comanda **Exit**, sau daţi **Logout** din Oracle Database Express Edition atunci are loc un **COMMIT** automat. Dacă ieşirea se face anormal, de exemplu în cazul unei pene de curent, atunci se execută în mod automat o comandă **ROLLBACK**.
- Executaţi o comandă **DML** care eşuează, caz în care are loc un **ROLLBACK** automat pentru acea singură comandă.

- Pentru a experimenta folosirea tranzacţiilor vom crea următoarea tabelă:

  create table savepoint_test (n number)
- Inserăm acum câteva linii în această tabelă:

  insert into savepoint_test values (1);

  insert into savepoint_test values (2);

  insert into savepoint_test values (3);
- Definim acum un punct de salvare:

  savepoint sp1;
- Și mai inserăm câteva linii în tabelă:

  insert into savepoint_test values (10);

  insert into savepoint_test values (20);

  insert into savepoint_test values (30);

- Definim un nou punct de salvare:

savepoint sp2;

- şi inserăm în final încă trei linii:

  insert into savepoint_test values (100);

  insert into savepoint_test values (200);

  insert into savepoint_test values (300);

- Verificăm acum dacă datele au fost inserate în tabelă:

select * from savepoint_test;

- şi vedem că toate datele au fost inserate:

```
SQL> select * from savepoint_test;

         N
----------
         1
         2
         3
        10
        20
        30
       100
       200
       300

9 rows selected.
```

- Revenim acum la punctul de revenire sp2

  ROLLBACK TO sp2
- şi verificăm conţinutul tabelei:

  select * from savepoint_test;
- Observaţi că ultimele linii inserate după definirea punctului de salvare sp2 au fost şterse din tabelă

```
SQL> rollback to sp2;

Rollback complete.

SQL> select * from savepoint_test;

         N
----------
         1
         2
         3
        10
        20
        30

6 rows selected.

SQL>
```

- Inserăm alte trei linii:

insert into savepoint_test values (111);

insert into savepoint_test values (222);

insert into savepoint_test values (333);

- testăm conţinutul tabelei:

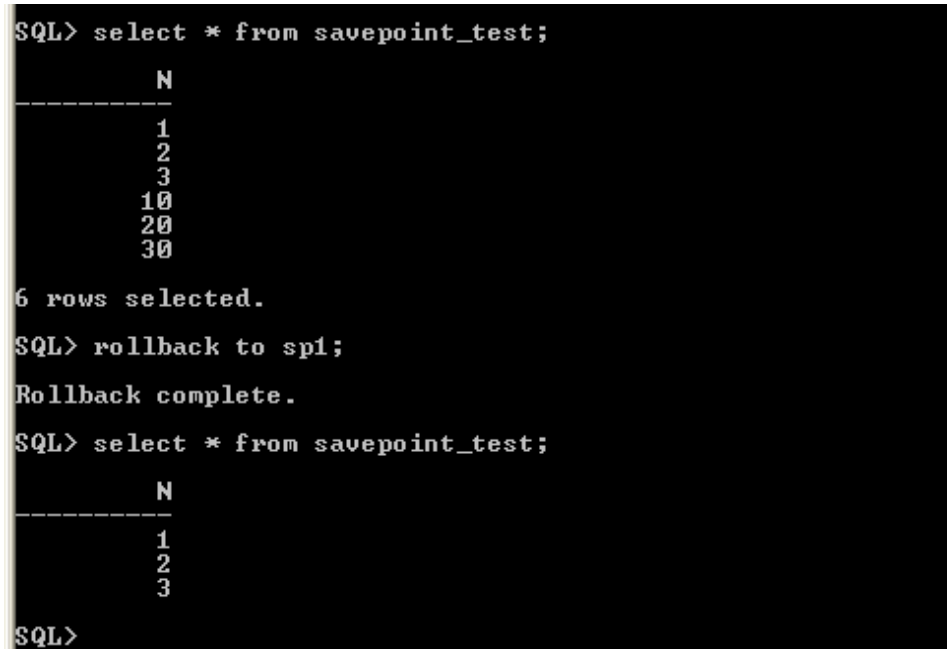
select * from savepoint_test;

```
SQL> insert into savepoint_test values(333);

1 row created.

SQL> select * from savepoint_test;

         N
----------
         1
         2
         3
        10
        20
        30
       111
       222
       333

9 rows selected.
```

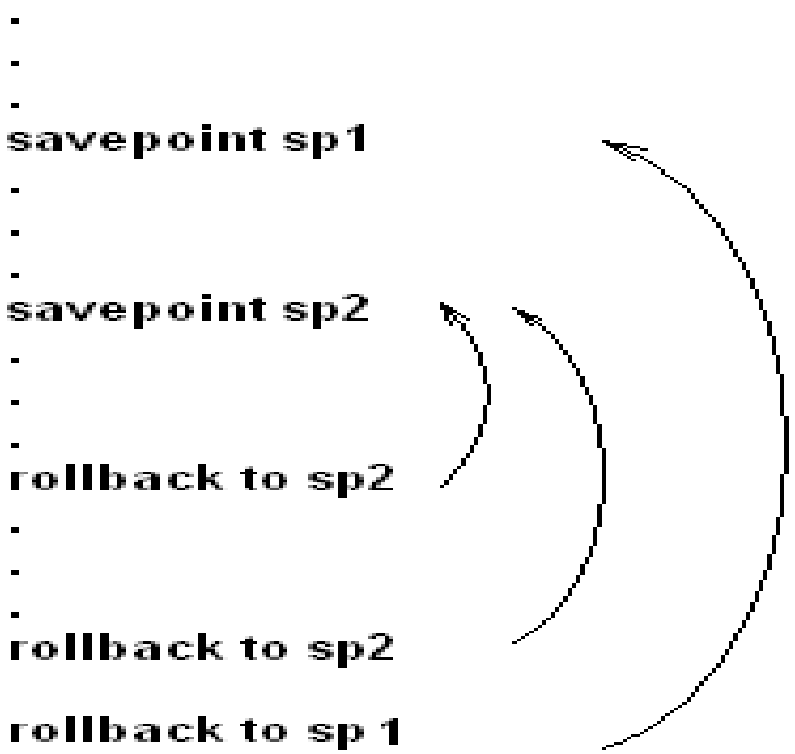
- Revenim la punctul de salvare **sp2**:
  ROLLBACK TO sp2
- şi verificăm conţinutul tabelei:
  select * from savepoint_test;
- Evident ultimele trei linii nu se mai găsesc în tabelă conţinutul tabelei.
- Dacă revenim acum la punctul de salvare sp1, în tabelă nu mai rămân decât trei linii

ROLLBACK TO sp1

select * from savepoint_test;



- Schematic tranzacţia anterioară arată astfel:



- În această lecție am învățat:
- Cum putem controla accesul utilizatorilor la baza de date
- Care sunt categoriile de drepturi ce se pot atribui utilizatorilor
- Cum acordăm drepturi utilizatorilor la anumite obiecte ale bazei de date
- Cum revocăm anumite drepturi ale utilizatorilor
- Ce sunt rolurile
- Cum se creează și cum se șterge un rol
- Cum se alocă drepturi rolurilor
- Cum se acordă un rol unui utilizator
- Ce este o tranzacție
- Când începe o tranzacție
- Când se termină o tranzacție
- Cum devin permanente modificările făcute într-o tranzacție asupra bazei de date
- Cum se anulează modificările făcute într-o tranzacție asupra bazei de date