Concepte, caracteristici şi alternative de securitate în sistemul *Oracle*

În cadrul acestui curs, vom prezenta câteva metode şi caracteristici ale bazei de date Oracle referitoare la cerinţele, ameninţările şi conceptele prezentate în cursul precedent.

Vor fi introduse:

- Metode de autentificare;
- Autorizarea: privilegii, roluri, profiluri şi limitări ale resurselor.

1. Metode de autentificare

Autentificarea presupune verificarea identităţii unui utilizator, dispozitiv sau entităţi, care doreşte să acceseze date, resurse sau aplicaţii. Validarea identităţii stabileşte o relaţie de încredere pentru interacţiunile viitoare. De asemenea, autentificarea permite asumarea răspunderii prin faptul că permite legarea accesului şi acţiunilor de anumite identităţi. După autentificare, procesele de autorizare pot permite sau limita nivelurile de acces şi acţiunile permise acelei entităţi.

Sistemul *Oracle* permite unei instanţe a bazei de date să utilizeze oricare dintre metode, sau chiar pe toate. Sunt necesare proceduri speciale de autentificare pentru administratorii bazei de date. De asemenea, sistemul *Oracle* criptează parolele pe perioada transmiterii acestora, pentru a asigura securitatea autentificării în reţea.

Pentru a valida identitatea utilizatorilor bazei de date şi a preveni folosirea neautorizată a numelui unui utilizator al bazei de date, autentificarea utilizatorilor se poate realiza prin orice combinaţie a metodelor descrise în continuare.

1.1. Autentificarea la nivelul sistemului de operare

Anumite sisteme de operare permit sistemului *Oracle* să utilizeze informaţia pe care o păstrează, în scopul autentificării utilizatorilor. Beneficiile acestei metode sunt următoarele:

- Odată ce sunt autentificaţi de către sistemul de operare, utilizatorii se pot conecta la sistemul *Oracle* mai uşor, fără furnizarea unui nume de utilizator şi a unei parole. De exemplu, se poate invoca *SQL*Plus* prin comanda:
  SQLPLUS /
- Având control asupra autentificării centralizate a utilizatorilor în sistemul de operare, *Oracle* nu mai trebuie să stocheze sau să gestioneze parole, dar menţine numele utilizatorilor în baza de date.
- Urmele de audit din baza de date şi cele din sistemul de operare pot folosi aceleaşi nume de utilizatori.

Atunci când un sistem de operare este utilizat pentru a autentifica utilizatori ai bazei de date, gestionarea mediilor de baze de date distribuite şi a legăturilor de baze de date necesită o atenţie specială.

1.2. Autentificarea în reţea

Acest tip de autentificare este tratat de către protocolul *SSL* sau de către servicii *third-party* (*Kerberos*, bazat pe *PKI*, *RADIUS*, bazate pe directoare).

1.2.1 Autentificarea utilizând SSL

SSL (*Secure Socket Layer*) este un protocol la nivel de aplicaţie. Acesta poate fi folosit pentru autentificarea utilizatorului la o bază de date, care este independentă de gestiunea globală a utilizatorilor în *Oracle Internet Directory*. Aceasta înseamnă că utilizatorii pot folosi *SSL* pentru a se autentifica la baza de date chiar şi fără un server de directoare.

1.2.2. Autentificarea cu ajutorul serviciilor third-party

Autentificarea într-o reţea utilizează serviciile *third-party* de autentificare în reţea. Exemplele reprezentative includ *Kerberos*, *PKI* (*Public Key Infrastructure*), *RADIUS* (*Remote Authentication Dial-In User Service*), precum şi serviciile bazate pe directoare, care vor fi descrise în continuare.

Dacă serviciile de autentificare în reţea sunt disponibile, atunci sistemul *Oracle* poate accepta autentificarea de la un astfel de serviciu în reţea. În această situaţie, apar anumite consideraţii speciale pentru *role*-urile în reţea şi legăturile de baze de date.

Observaţie: Pentru a utiliza serviciul de autentificare în reţea cu *Oracle*, este necesară versiunea *Oracle Enterprise Edition* cu opţiunea *Oracle Advanced Security*.

Autentificarea cu Kerberos. Kerberos este un sistem de autentificare care se bazează pe secrete partajate. El presupune că terţa parte este sigură şi furnizează funcţionalităţi de *single sign-on*, stocare centralizată a parolelor, autentificare a legăturilor de baze de date, precum şi o securitate mai bună a calculatorului. Toate acestea sunt realizate printr-un server de autentificare *Kerberos* sau prin *Cybersafe Active Trust*, un server de autentificare comercial bazat pe *Kerberos*.

Autentificarea bazată pe PKI. Sistemele de autentificare bazate pe *PKI* emit certificate digitale utilizatorilor clienţi, care le folosesc pentru a se autentifica direct pe serverele din sistemul *enterprise* fără a face apel direct la un server de autentificare. Sistemul *Oracle* furnizează un *PKI* pentru utilizarea cheilor şi certificatelor publice, ce constă din următoarele componente:

- autentificare şi gestiune sigură a cheilor la nivel de sesiune, utilizând *SSL*;
- funcţii *OCI* (*Oracle Call Interface*) şi *PL/SQL*, utilizate pentru semnarea datelor specificate de către utilizator folosind o cheie şi un certificat privat; verificarea semnăturii asupra datelor este realizată utilizând un certificat de încredere (*trusted*);
- certificate de încredere, utilizate pentru a identifica entităţile *third-party* care sunt de încredere ca semnatari ai certificatelor utilizatorilor atunci când o identitate este validată. Atunci când certificatul utilizatorului este validat, semnatarul este verificat folosind puncte de încredere (*trust points*) sau un lanţ de certificate de încredere ale autorităţilor stocate în sistemul de validare. Dacă există câteva niveluri de certificate de încredere în acest lanţ, atunci un certificat de încredere de pe un nivel inferior nu necesită re-verificarea certificatelor de pe nivelurile superioare.
- portofele *Oracle*, care sunt structuri de date care conţin cheia privată a unui utilizator, un certificat utilizator şi setul de puncte de încredere ale unui utilizator (autorităţi de certificate de încredere).
- autoritatea de certificare *OracleAS*, care este o componentă a infrastructurii *Oracle Identity Management*, ce oferă o soluţie integrată pentru provizionarea certificatelor X.509 pentru indivizi, aplicaţii şi servere care necesită certificate pentru operaţiile bazate pe *PKI*, cum ar fi autentificarea, *SSL*, *S/MIME* etc.
- *Oracle Wallet Manager*, care este o aplicaţie *Java standalone* utilizată pentru a gestiona şi edita informaţiile de securitate din portofelele *Oracle*. Acesta efectuează următoarele operaţii: protejează cheile utilizatorilor, gestionează certificatele X.509 pe clienţii şi serverele *Oracle*, generează o pereche de chei (publică şi privată) şi creează o cerere de certificat către autoritatea de certificare, instalează un certificat pentru entitate,

configurează certificate de încredere pentru entitate, creează portofele, deschide un portofel pentru a permite accesul la servicii bazate pe *PKI*.

- certificatele X.509 obţinute (şi semnate) de la o entitate de încredere (o autoritate de certificare), care certifică faptul că informaţia entităţii solicitante este corectă şi cheia publică asupra certificatului aparţine entităţii identificate. Certificatul este încărcat într-un portofel *Oracle* pentru a permite autentificarea viitoare.

Figura următoare ilustrează infrastructura *Oracle* cu chei publice.

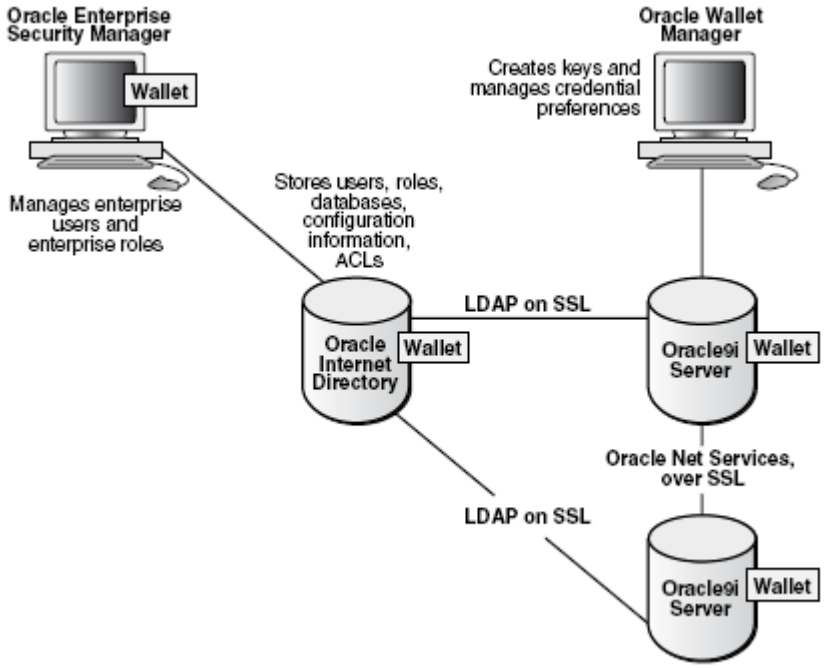

Figura 1. Infrastructura *Oracle* cu chei publice

Autentificare cu RADIUS. Sistemul *Oracle* suportă autentificarea distantă a utilizatorilor prin *RADIUS*, un protocol standard folosit pentru autentificarea, autorizarea şi asumarea responsabilităţii utilizatorului.

Serviciile bazate pe directoare. Utilizarea unui director central poate determina ca autentificarea şi administrarea să devină extrem de eficiente. Serviciile bazate pe directoare includ:

- *Oracle Internet Directory*, care utilizează *LDAP* (*Lightweight Directory Access Protocol*) şi permite ca informaţia despre utilizatori să fie stocată şi gestionată central. Deşi utilizatorii trebuie creaţi (cu parole) în fiecare bază de date care trebuie accesată, informaţia despre acel utilizator este accesibilă central în *Oracle Internet Directory*. Acesta permite gestiunea atributelor şi privilegiilor de securitate ale utilizatorilor,

inclusiv autentificarea utilizatorilor prin certificate X.509. De asemenea, permite controlul accesului la nivel de atribut. Această caracteristică permite ca privilegiile de citire, scriere sau actualizare asupra anumitor atribute să fie restricţionate doar anumitor utilizatori. Cererile şi răspunsurile la / de la directoare pot utiliza criptarea *SSL* pentru o protecţie mai bună în timpul autentificării şi al altor interacţiuni.

- *Oracle Enterprise Security Manager*, care furnizează gestiunea centralizată a privilegiilor pentru a uşura administrarea şi a creşte nivelurile de securitate. Această componentă permite stocarea şi regăsirea *role*-urilor din *Oracle Internet Directory*.

1.3. Autentificare cu ajutorul *Oracle Database*

Oracle Database poate autentifica utilizatorii care încearcă să se conecteze la baza de date, chiar cu ajutorul informaţiei stocate în baza de date. Pentru a utiliza autentificarea de acest tip, fiecare utilizator trebuie creat cu o parolă asociată. *Oracle Database* stochează parolele utilizatorilor în dicţionarul datelor într-un format criptat pentru prevenirea modificărilor neautorizate. Parolele pot fi modificate doar de către utilizatori.

Pentru a identifica protocoalele de autentificare care sunt permise unui client sau bazei de date, un *DBA* poate seta explicit parametrul *SQLNET.ALLOWED_LOGON_VERSION* din fişierul *sqlnet.ora*. Apoi, fiecare încercare de conexiune este testată, iar în cazul în care clientul sau serverul nu îndeplinesc versiunea minimă specificată de „partener”, autentificarea eşuează cu eroarea ORA-28040. Parametrul poate lua valorile 12, 11, 10 (implicit), 9 sau 8 şi reprezintă versiuni ale serverului bazei de date.

Autentificarea cu ajutorul bazei de date include următoarele caracteristici:

- criptarea parolelor în cadrul conexiunilor în reţea cu ajutorul *AES* (*Advanced Encryption Standard*), înainte de trimiterea lor;
- blocarea conturilor după un număr de tentative de conectare consecutive eşuate; deblocarea poate avea loc automat după un anumit interval de timp sau poate necesita intervenţia administratorului bazei de date; stabilirea acestor caracteristici pentru un utilizator se poate realiza cu ajutorul comenzii *CREATE PROFILE*; administratorul bazei de date poate bloca manual conturile, astfel încât acestea să nu se deblocheze automat ci doar prin intervenţia lui;
- durata de viaţă şi expirarea parolelor – administratorul bazei de date poate specifica o durată de viaţă a parolelor, după care acestea expiră şi este necesară schimbarea lor. Poate fi stabilită o perioadă de graţie, în timpul căreia tentativele de logare primesc un

avertisment de schimbare a parolei. După această perioadă, contul va fi blocat. De asemenea, administratorul poate stabili ca starea unei parole să fie „expirată”, ceea ce conduce la necesitatea ca utilizatorul (sau administratorul) să schimbe parola înainte de conectarea la baza de date.

- istoricul parolelor constituie o opţiune de verificare a noilor parole, pentru a preveni ca acestea să nu fie reutilizate un anumit interval de timp sau de un număr specificat de ori; administratorul poate configura regulile privitoare la reutilizarea parolelor cu ajutorul instrucţiunilor *CREATE PROFILE*.
- verificarea complexităţii parolelor are în vedere ca fiecare parolă să fie suficient de complexă pentru a asigura o protecţie rezonabilă împotriva intruşilor care încearcă să intre în sistem prin „ghicirea” parolelor. Rutina de verificare a complexităţii parolelor în *Oracle* (aflată în *script*-ul *PL/SQL UTLPWDMG.sql*, care setează parametrii impliciţi ai profilului) poate determina dacă parolele îndeplinesc anumite cerinţe, precum:
    - să aibă o lungime de minimum 4 caractere
    - să nu fie aceeaşi cu *userid*-ul
    - să includă cel puţin un caracter alfabetic, unul numeric şi un semn de punctuaţie
    - să nu aparţină unei liste interne de cuvinte simple (ex: *welcome*, *account*, *database*, *user* etc.);
    - să difere de parola precedentă prin cel puţin 3 caractere.

1.4. Autentificarea şi autorizarea *multi-tier*

Într-un mediu *multi-tier*, sistemul *Oracle* controlează securitatea aplicaţiilor *middle-tier* prin limitatea privilegiilor lor, păstrând identităţile client la toate nivelurile şi auditând acţiunile întreprinse în numele clienţilor. În aplicaţiile care utilizează un *middle-tier* important, cum ar fi un monitor de procesare a tranzacţiilor, identitatea clienţilor care se conectează la acest *middle tier* trebuie păstrată. Un avantaj al utilizării unui *middle tier* este menţinerea conexiunilor (*connection pooling*), care permite multiplilor utilizatori să acceseze un server de date fără ca fiecare dintre aceştia să aibă nevoie de o conexiune separată. În astfel de medii, este necesar ca stabilirea şi întreruperea conexiunilor să fie făcute foarte repede.

Pentru astfel de medii, administratorii bazei de date *Oracle* pot utiliza *Oracle Call Interface* ca să creeze sesiuni *lightweight*, care permit autentificarea parolelor pentru fiecare utilizator. Această metodă păstrează identitatea utilizatorului real în *middle-tier* fără costul suplimentar al unei conexiuni separate la baza de date pentru fiecare utilizator.

Se pot crea sesiuni *lightweight* cu sau fără parole. Dacă un *middle-tier* se află în afara unui *firewall*, atunci securitatea este mai bună când fiecare sesiune *lightweight* are propria parolă. Pentru un server de aplicaţii interne, acest tip de sesiuni, fără parole, pot fi adecvate.

În continuare, discutăm câteva probleme de administrare şi securitate în mediile *multi-tier*.

1.4.1. Clienţi, server-e de aplicaţii şi servere de baze de date

Într-un mediu *multi-tier*, un server de aplicaţii furnizează date pentru clienţi şi serveşte ca interfaţă a lor către unul sau mai multe server-e de baze de date. Server-ul de aplicaţii poate valida informaţiile unui client, cum ar fi un browser web, iar server-ul de baze de date poate audita operaţiile efectuate de către serverul de aplicaţii. Operaţiile care pot fi auditate includ acţiuni efectuate de către server-ul de aplicaţii în numele clienţilor, cum ar fi cereri ca informaţia să fie afişată pe staţia client. O cerere de conectare la server-ul de baze de date este un exemplu de operaţie a server-ului de aplicaţii care nu are legătură cu un anumit client.

Autentificarea într-un mediu *multi-tier* se bazează pe regiunile de încredere. Autentificarea clientului aparţine domeniului server-ului de aplicaţii. Server-ul de aplicaţii este, la rândul lui, autentificat de către server-ul de baze de date. Sunt efectuate următoarele operaţii:

- clientul face dovada autenticităţii către server-ul de aplicaţii, utilizând o parolă sau un certificat X.509;
- server-ul de aplicaţii autentifică clientul şi apoi se autentifică pe el însuşi la baza de date;
- server-ul de baze de date autentifică server-ul de aplicaţii, verifică existenţa clientului şi verifică dacă server-ul de aplicaţii deţine privilegiul de a conecta acest client.

Server-ele de aplicaţii pot, de asemenea, să activeze roluri pentru un client în al cărui nume se conectează. Server-ul de aplicaţii poate obţine aceste role-uri dintr-un director, care serveşte ca depozit (*repository*) pentru autorizări. Server-ul de aplicaţii poate doar solicita ca aceste roluri să fie activate. Baza de date verifică următoarele cerinţe:

- clientul deţine aceste *role*-uri, prin verificarea depozitului intern de *role*-uri;
- server-ul de aplicaţii deţine privilegiul de a se conecta în numele utilizatorului şi, astfel, de a folosi aceste *role*-uri la fel ca utilizatorul.

Figura următoare arată un exemplu de autentificare *multi-tier*:

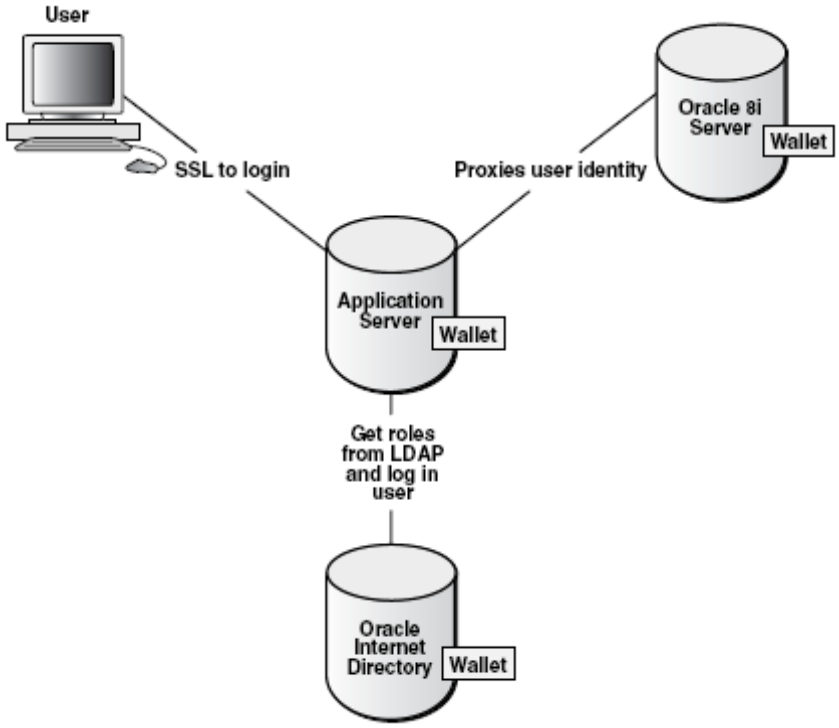

Figura 2. Autentificare *multi-tier*

1.4.2. Probleme de securitate specifice aplicaţiilor middle-tier

În aplicaţiile *middle-tier*, securitatea trebuie să aibă în vedere următoarele aspecte:

- Asumarea responsabilităţii: Server-ul de baze de date trebuie să facă distincţie între acţiunile unui client şi cele iniţiate de o aplicaţie în numele unui client. Trebuie să se poată realiza auditarea ambelor tipuri de acţiuni.
- Diferenţiere: Server-ul de baze de date trebuie să facă distincţie între un client ce accesează baza de date direct şi un server de aplicaţii care acţionează în numele său sau în cel al unui browser client.
- Cel mai mic privilegiu: Utilizatorii şi *middle-tiers* ar trebui să primească cele mai puţine privilegii necesare pentru a-şi efectua acţiunile, cu scopul de a minimiza pericolul activităţilor neautorizate.

1.4.3. Probleme de identitate într-un mediu multi-tier

Autentificarea *multi-tier* păstrează identitatea clientului prin toate nivelurile (*tiers*) conexiunii, cu scopul de a menţine înregistrări de audit relevante. Dacă identitatea clientului se pierde, se pierde şi asumarea responsabilităţii sale. Devine imposibilă distingerea operaţiilor efectuate de către server-ul de aplicaţii în numele clientului de cele efectuate în nume propriu.

1.4.4. Restricţionarea privilegiilor într-un mediu multi-tier

Privilegiile într-un mediu *multi-tier* trebuie să fie limitate doar la cele necesare pentru efectuarea operaţiei solicitate.

Privilegii client – trebuie să fie cât mai limitate posibil. Operaţiile sunt efectuate în numele clientului de către server-ul de aplicaţii.

Privilegii ale server-ului de aplicaţii – trebuie să fie, de asemenea, limitate, astfel încât server-ul de aplicaţii să nu poată efectua operaţii nedorite sau care nu sunt necesare pe perioada efectuării unei operaţii client.

1.5 Autentificarea administratorilor bazei de date

Administratorii bazei de date efectuează anumite operaţii speciale (cum ar fi pornirea sau oprirea bazei de date), care nu ar trebui efectuate de către utilizatorii normali. Sistemul *Oracle* furnizează nume de utilizatori pentru administratorii bazei de date, pentru care se poate alege fie autentificare la nivelul sistemului de operare, fie prin fişiere de parole.

Figura următoare ilustrează opţiunile pe care le avem pentru autentificarea administratorilor bazei de date. Diferitele opţiuni se aplică la administrarea locală a bazei de date (pe maşina pe care aceasta rezidă) şi la administrarea mai multor baze de date, aflate pe maşini diferite, de pe un client distant.

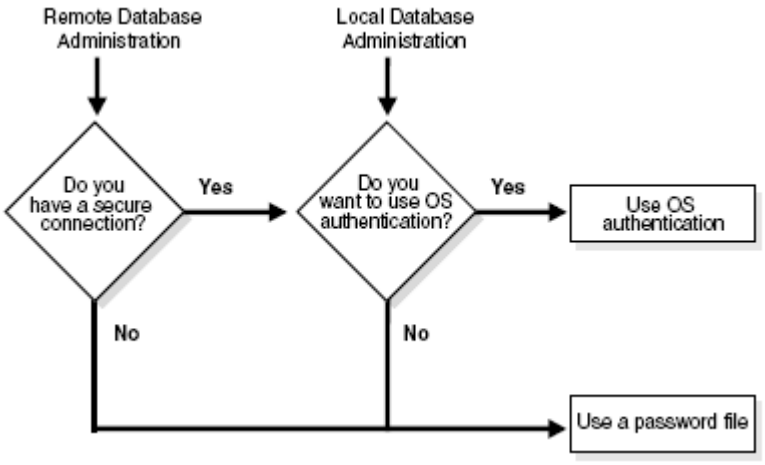

Figura 3. Autentificarea administratorilor bazei de date

Autentificarea la nivelul sistemului de operare pentru un administrator al bazei de date presupune crearea unui grup în sistemul de operare, atribuirea de privilegii *DBA* acelui grup, iar apoi adăugarea în grup a numelor de utilizatori care ar trebui să deţină acele privilegii.

Observaţie: În UNIX, grupul se numeşte *dba group*.

Baza de date utilizează fişiere de parole pentru a urmări utilizatorii bazei de date cărora li s-au acordat privilegii *SYSDBA* şi *SYSOPER*. Aceste privilegii permit următoarele operaţii şi funcţionalităţi:

- *SYSOPER* permite administratorilor bazei de date să efectueze operaţiile *STARTUP*, *SHUTDOWN*, *ALTER DATABASE OPEN*/*MOUNT*, *ALTER DATABASE BACKUP*, *ARCHIVE LOG* şi *RECOVER*. De asemenea, *SYSOPER* include privilegiul *RESTRICTED SESSION*.
- *SYSDBA* deţine toate privilegile sistem cu *ADMIN OPTION*, inclusiv privilegiul sistem *SYSOPER*, şi permite operaţia *CREATE DATABASE* şi recuperarea *time-based*.

2. Autorizare: privilegii, *role*-uri, profiluri şi limitări ale resurselor

În principal, autorizarea include două procese:

- Permisiunea doar ca anumiţi utilizatori să acceseze, proceseze sau să modifice date;
- Aplicarea unor limitări variate asupra accesului utilizatorilor sau acţiunilor. Limitările asupra utilizatorilor se pot referi la obiecte (scheme, tabele, linii), sau la resurse (timp CPU, de conectare, *idle*).

Această secţiune introduce conceptele de bază şi mecanismele pentru impunerea sau ştergerea unor astfel de limitări asupra utilizatorilor, individual sau în grupuri.

2.1. Privilegii

Un privilegiu este un drept de a executa un tip particular de instrucţiune *SQL* sau de a accesa obiectele altui utilizator. Anumite exemple de privilegii includ dreptul de a se conecta la baza de date (crea o sesiune), crea un tabel, selecta linii din tabelul altui utilizator, executa o procedură stocată a altui utilizator.

Privilegiile sunt acordate utilizatorilor pentru ca aceştia să poată realiza task-uri necesare job-urilor lor. Acordarea excesivă de privilegii care nu sunt necesare poate compromite securitatea. Un utilizator poate primi un privilegiu în două moduri:

- explicit (de exemplu, îi acordăm utilizatorului *SCOTT* privilegiul de a insera linii în tabelul *employees*);

- prin acordarea de privilegii unui *role* (un grup de privilegii, având un nume), şi apoi acordarea *role*-ului unuia sau mai multor utilizatori. De exemplu, acordăm privilegiile de a selecta, insera, actualiza şi şterge linii din tabelul *employees* unui *role* numit *clerk*, care poate fi acordat utilizatorilor *scott* şi *brian*.

Deoarece *role*-urile permit o gestiune mai uşoară şi mai bună a privilegiilor, se recomandă acordarea de privilegii *role*-urilor şi nu utilizatorilor.

Există 6 categorii principale de privilegii, unele având subcategorii:

- privilegii sistem
- privilegii asupra obiectelor schemei
- privilegii tabel
- privilegii vizualizare
- privilegii procedură
- privilegii tip

2.1.1. Privilegii sistem

Un privilegiu sistem reprezintă dreptul de a efectua o acţiune particulară asupra oricăror obiecte ale schemei, de un anumit tip. De exemplu, privilegiile de a crea spaţii tabel şi de a şterge linii din oricare tabel din baza de date sunt privilegii sistem. Există peste 100 de privilegii sistem distincte.

Acordarea şi revocarea privilegiilor sistem – privilegiile sistem pot fi acordate sau revocate utilizatorilor şi *role*-urilor. Dacă sunt acordate privilegii sistem *role*-urilor, atunci aceste *role*-uri pot fi folosite pentru gestionarea privilegiilor sistem. De exemplu, *role*-urile permit ca privilegiile să fie disponibile selectiv.

Observaţie: În general, privilegiile sistem sunt acordate doar personalului administrativ şi dezvoltatorilor de aplicaţii, nu şi utilizatorilor finali.

Acordarea sau revocarea privilegiilor sistem se poate realiza cu ajutorul:

- *Oracle Enterprise Manager 10g Database Control*
- Instrucţiunilor *GRANT* şi *REVOKE*.

Privilegiile sistem pot fi acordate sau revocate de doar două categorii de utilizatori:

- cei cărora le-a fost acordat un anumit privilegiu sistem cu clauza *WITH ADMIN OPTION*
- care deţin privilegiul sistem *GRANT ANY PRIVILEGE*.

2.1.2. Privilegii asupra obiectelor schemei

Un privilegiu asupra unui obiect al schemei este permisiunea de a efectua o anumită acţiune asupra acelui obiect.

Există diferite privilegii obiect care pot fi aplicate diferitelor tipuri de obiecte ale schemei. Un exemplu de privilegiu obiect este cel de a şterge linii din tabelul *departments*.

Anumite obiecte ale schemei, cum ar fi clusterele, indecşii, *trigger*-ii şi legăturile de baze de date nu au privilegii obiect asociate. Utilizarea lor este controlată cu ajutorul privilegiilor sistem. De exemplu, pentru a modifica un cluster, un utilizator trebuie să fie posesorul acestuia sau să deţină privilegiul sistem ALTER ANY CLUSTER.

Acordarea şi revocarea privilegiilor asupra obiectelor schemei. Aceste privilegii pot fi acordate sau revocate utilizatorilor şi *role*-urilor. Privilegiile obiect pot fi acordate sau revocate asemănător privilegiilor sistem (în cele două moduri).

Cine poate acorda privilegii asupra obiectelor schemei? Un utilizator deţine automat toate privilegiile obiect asupra obiectelor schemei sale. Un utilizator poate acorda orice privilegiu obiect asupra oricărui obiect al schemei pe care o deţine, oricărui alt utilizator sau *role*. Un utilizator având privilegiul *GRANT ANY OBJECT PRIVILEGE* poate acorda sau revoca orice privilegiu obiect specificat unui alt utilizator, cu sau fără clauza *GRANT OPTION* în instrucţiunea *GRANT*. Altfel, cel care primeşte privilegiul îl poate utiliza, dar nu îl poate acorda mai departe.

Exemplu: Presupunem că utilizatorul *SCOTT* deţine tabelul *t2*.

```
SQL>GRANT GRANT ANY OBJECT PRIVILEGE TO u1;
SQL> CONNECT u1/u1
Connected.
SQL> GRANT SELECT ON scott.t2 TO u2;
SQL> SELECT GRANTEE, OWNER, GRANTOR, PRIVILEGE, GRANTABLE FROM
DBA_TAB_PRIVS
WHERE TABLE_NAME = 'employees';
GRANTEE OWNER   GRANTOR PRIVILEGE GRA
---------------------------------------
U2      SCOTT   SCOTT   SELECT    NO
```

Utilizarea privilegiilor cu sinonime. Un obiect al schemei şi sinonimele sale sunt echivalente în privinţa privilegiilor (acestea se aplică oricum ar fi referit obiectul – prin nume sau prin sinonime). Privilegiile pot fi acordate fie utilizând numele obiectului, fie sinonimul său (efectul este acelaşi). Dacă un sinonim este şters, atunci toate acordările de privilegii rămân valabile pentru obiectul schemei, chiar dacă acestea au fost acordate prin intermediul sinonimului.

Privilegii tabel

Privilegiile asupra tabelelor permit securizarea acestora la nivelul operaţiilor *LMD* şi *LDD*.

Operaţii LMD. Pot fi acordate privilegii pentru utilizarea operaţiilor *INSERT*, *UPDATE*, *DELETE* şi *SELECT* asupra unui tabel sau vizualizări. Aceste privilegii trebuie acordate doar utilizatorilor şi *role*-urilor care necesită interogarea sau prelucrarea datelor din tabele.

Privilegiile *INSERT* şi *UPDATE* pot fi restricţionate doar asupra unor coloane ale tabelului. În cadrul unei operaţii *INSERT* selective, un utilizator va putea insera o linie specificând valori doar pentru coloanele selectate. Toate celelalte coloane primesc valoarea *NULL* sau pe cea stabilită implicit. În cadrul unei operaţii *UPDATE* selective, un utilizator va putea actualiza numai valorile anumitor coloane ale unei linii. Privilegiile *INSERT* şi *UPDATE* selective sunt utilizate pentru a restricţiona accesul la datele sensibile.

De exemplu, dacă nu dorim ca operatorii de date să modifice coloana *salary* a tabelului *employees*, atunci trebuie acordate privilegiile selective *INSERT* sau *UPDATE*, care exclud coloana *salary*. O soluţie alternativă constă în utilizarea unei vizualizări care exclude această coloană.

Operaţii LDD. Privilegiile *ALTER*, *INDEX* şi *REFERENCES* permit efectuarea de operaţii *LDD* asupra unui tabel. Deoarece aceste privilegii permit şi altor utilizatori să modifice sau să creeze dependenţe asupra unui tabel, privilegiile trebuie acordate în mod conservator.

Un utilizator care încearcă să efectueze o operaţie *LDD* asupra unui tabel poate avea nevoie de privilegii sistem sau obiect suplimentare. De exemplu, pentru a crea un *trigger* pe un tabel, este necesar atât privilegiul obiect *ALTER TABLE* pentru tabel cât şi privilegiul sistem *CREATE TRIGGER*.

La fel ca în cazul privilegiilor *INSERT* şi *UPDATE*, privilegiul *REFERENCES* poate fi acordat asupra anumitor coloane ale tabelului. Privilegiul *REFERENCES* permite celui căruia îi este acordat să utilizeze tabelul asupra căruia s-a făcut acordarea pentru o cheie părinte corespunzătoare oricărei chei externe pe care doreşte să o creeze în propriile sale tabele. Această acţiune este controlată cu ajutorul unui privilegiu special, deoarece prezenţa cheilor externe restricţionează prelucrarea datelor şi modificările care pot fi realizate asupra cheii părinte. Un privilegiu *REFERENCES* specific unei coloane restricţionează destinatarul său să utilizeze coloanele menţionate (care trebuie să includă cel puţin o cheie primară sau unică a tabelului părinte).

Privilegii vizualizare

O vizualizare este o prezentare a datelor selectate din unul sau mai multe tabele sau vizualizări. O vizualizare arată structura şi datele tabelelor de bază şi poate fi gândită ca rezultatul unei cereri stocate. Vizualizarea nu conţine date, dar le derivă pe acestea din tabelele sau vizualizările pe care se bazează. O vizualizare poate fi interogată, iar datele pe care le prezintă pot fi modificate, şterse sau pot fi inserate linii noi. Atunci când sunt posibile, aceste operaţii modifică direct tabelele pe care se bazează vizualizarea şi sunt supuse constrângerilor de integritate şi *trigger*-ilor asupra tabelelor.

Privilegiile obiect *LMD* pentru tabele pot fi aplicate similar în cazul vizualizărilor. Privilegiile obiect pentru o vizualizare permit diferite operaţii *LMD.*

Privilegii necesare pentru crearea vizualizărilor. Pentru crearea unei vizualizări, trebuie îndeplinite următoarele cerinţe:

- trebuie acordat unul dintre următoarele privilegii sistem, fie explicit, fie prin intermediul unui role: *CREATE VIEW* (pentru crearea unei vizualizări în schema curentă), *CREATE ANY VIEW* (pentru crearea unei vizualizări în orice schemă);
- trebuie acordat explicit unul dintre următoarele privilegii:
    - privilegiile obiect *SELECT, INSERT, UPDATE, DELETE* asupra tuturor obiectelor de bază ale vizualizării;
    - privilegiile sistem *SELECT ANY TABLE, INSERT ANY TABLE, UPDATE ANY TABLE* sau *DELETE ANY TABLE*.
- suplimentar, pentru a acorda altor utilizatori acces la vizualizare, trebuie primite privilegii obiect asupra obiectelor de bază cu clauza *GRANT OPTION* sau privilegii sistem corespunzătoare cu clauza *ADMIN OPTION*. În caz contrar, cei cărora le acordăm privilegii nu pot accesa vizualizarea.

Creşterea gradului de securitate a tabelelor cu ajutorul vizualizărilor. Pentru a utiliza o vizualizare, sunt necesare privilegii adecvate doar pentru vizualizarea însăşi. Nu sunt necesare privilegii asupra obiectelor de bază ale vizualizării.

Vizualizările adaugă două niveluri suplimentare de securitate asupra tabelelor, securitatea la nivel de coloană şi cea bazată pe valoare:

- O vizualizare poate furniza acces la coloanele selectate din tabelele de bază. De exemplu, putem defini o vizualizare asupra tabelului *employees* pentru a afişa doar coloanele *employee_id*, *last_name* şi *manager_id*:

```
CREATE VIEW employees_manager AS
```

```
SELECT last_name, employee_id, manager_id
FROM employees;
```

- O vizualizare poate asigura securitatea bazată pe valoare a informaţiei din tabel.

Exemplu:

```
CREATE VIEW lowsal AS
      SELECT * FROM employees
      WHERE salary < 1000;
```

Această vizualizare permite accesul la toate liniile tabelului *employees* pentru care valoarea coloanei *salary* este mai mică decât 1000.

```
CREATE VIEW own_salary AS
      SELECT last_name, salary
      FROM employees
      WHERE last_name = USER;
```

În această vizualizare sunt vizibile doar liniile care corespund utilizatorului curent, fiind combinată atât securitatea la nivel de coloană cât şi cea la nivel de tabel.

Privilegii procedură

EXECUTE este singurul privilegiu obiect pentru proceduri, incluzând aici atât procedurile şi funcţiile *standalone* cât şi pachetele. Acest privilegiu trebuie acordat doar utilizatorilor care au nevoie de a executa o procedură sau de a compila o procedură care apelează o anumită procedură.

Execuţia procedurilor şi domenii de securitate. Un utilizator cu privilegiul obiect *EXECUTE* asupra unei anumite proceduri poate executa procedura sau compila o unitate de program care referă acea procedură. Nu se face nicio verificare de privilegii la *run-time*, atunci când procedura este apelată. Un utilizator cu privilegiul sistem *EXECUTE ANY PROCEDURE* poate executa orice procedură din baza de date. Privilegiul de a executa proceduri poate fi acordat utilizatorilor şi prin intermediul *role*-urilor.

Posesorul unei proceduri trebuie să deţină toate privilegiile obiect necesare asupra obiectelor referite. Dacă posesorul acordă unui alt utilizator dreptul de a utiliza acea procedură, atunci privilegiile obiect ale posesorului asupra obiectelor referite de către procedură se aplică la exercitarea procedurii de către acel utilizator. Acestea se mai numesc „drepturile creatorului”.

Utilizatorul unei proceduri, care nu este posesorul acesteia, se numeşte „apelant”. Sunt necesare privilegii suplimentare asupra obiectelor referite pentru procedurile apelantului, dar nu şi pentru cele ale creatorului.

Drepturile creatorului. Un utilizator al unei proceduri cu drepturile creatorului necesită doar privilegiul de a executa procedura şi niciun privilegiu asupra obiectelor accesate de către procedură, deoarece o procedură cu drepturile creatorului operează sub domeniul de securitate al utilizatorului care posedă procedura, indiferent cine o execută. Posesorul procedurii trebuie să aibă toate privilegiile obiect necesare asupra obiectelor referite. Mai puţine privilegii trebuie acordate utilizatorilor unei proceduri cu drepturi de creator, aceasta conducând la un control mai strâns al accesului la baza de date.

Procedurile cu drepturi ale creatorului pot fi utilizate pentru a controla accesul la obiecte private ale bazei de date şi a adăuga un nivel de securitate a bazei de date. Prin scrierea unei astfel de proceduri şi acordarea doar a privilegiului *EXECUTE* unui utilizator, acesta poate fi forţat să acceseze obiectele referite doar prin intermediul procedurii.

La execuţie, privilegiile posesorului unei proceduri stocate cu drepturi ale creatorului sunt verificate întotdeauna înainte de execuţia procedurii. Dacă un privilegiu necesar asupra unui obiect referit a fost revocat posesorului unei astfel de proceduri, atunci procedura nu poate fi executată de către posesor sau de vreun alt utilizator.

Observaţie: Execuţia *trigger*-ilor se desfăşoară în acelaşi mod ca procedurile cu drepturi ale creatorului. Utilizatorul execută o instrucţiune *SQL*, pentru care deţine privilegii. Ca rezultat al acestei instrucţiuni, este declanşat un *trigger* ale cărui instrucţiuni sunt executate temporar sub domeniul de securitate al utilizatorului care deţine *trigger*-ul.

Drepturile apelantului. O procedură cu drepturi de apelant se execută cu toate privilegiile acestuia. *Role*-urile sunt activate, mai puţin în situaţia în care procedura cu drepturi de apelant a fost invocată direct sau indirect de către o procedură cu drepturi de creator. Un utilizator al unei proceduri cu drepturi ale apelantului necesită privilegii (direct sau printr-un *role*) asupra obiectelor pe care le accesează procedura prin intermediul unor referinţe externe care sunt rezolvate în schema apelantului.

Apelantul necesită privilegii la *run-time* pentru a accesa referinţele din program încapsulate în instrucţiuni *LMD* sau instrucţiuni *SQL* dinamice, deoarece acestea sunt recompilate la *run-time*.

Pentru toate celelalte referinţe externe, cum ar fi apeluri directe de funcţii *PL/SQL*, privilegiile posesorului sunt verificate la momentul compilării şi nu are loc nicio verificare la *run-time*. Prin urmare, utilizatorul unei proceduri cu drepturi de apelant nu necesită privilegii asupra referinţelor externe din afara *LMD* sau instrucţiunilor *SQL* dinamice. Alternativ,

dezvoltatorul unei proceduri cu drepturi de apelant necesită doar să acorde privilegii procedurii înseşi, nu asupra tuturor obiectelor referite direct de către procedura cu drepturi de apelant.

Se poate crea un pachet software care constă din mai multe unităţi de program, unele dintre acestea cu drepturi ale creatorului iar altele cu drepturi ale apelantului, prin care se restricţionează punctele de intrare în program. Un utilizator care are privilegiul de a executa o procedură poate executa şi unităţi de program interne în mod indirect, dar nu poate apela direct aceste programe interne.

Privilegii sistem necesare pentru a crea sau a modifica o procedură. Pentru a crea o procedură, utilizatorul trebuie să deţină privilegiul sistem *CREATE PROCEDURE* sau *CREATE ANY PROCEDURE*. Pentru a modifica o procedură (a o recompila) un utilizator trebuie fie să posede procedura fie privilegiul sistem *ALTER ANY PROCEDURE*.

Utilizatorul care posedă procedura trebuie să deţină privilegii asupra obiectelor referite în corpul procedurii. Pentru a crea o procedură, privilegiile necesare asupra obiectelor referite trebuie să fi fost acordate explicit, nu prin intermediul *role*-urilor. Aceasta include şi privilegiul *EXECUTE* asupra procedurilor apelate în cadrul celei create.

Observaţie: Acelaşi lucru este valabil pentru *trigger*-i.

Pachete şi obiecte ale pachetelor

Un utilizator ce deţine privilegiul obiect *EXECUTE* asupra unui pachet poate executa orice procedură sau funcţie publică din pachet şi poate accesa sau modifica valoarea oricărei variabile din pachet. Construcţiilor din cadrul unui pachet nu le pot fi acordate privilegii *EXECUTE* individuale. Prin urmare, este utilă considerarea a două alternative pentru stabilirea securităţii atunci când sunt dezvoltate proceduri, funcţii şi pachete în cadrul unei aplicaţii de baze de date. Aceste alternative sunt descrise în cadrul exemplelor care urmează.

Exemplu: 4 proceduri create în corpurile a două pachete.

```
CREATE PACKAGE BODY hire_fire AS
   PROCEDURE hire(...) IS
   BEGIN
      INSERT INTO employees . . .
   END hire;
   PROCEDURE fire(...) IS
   BEGIN
      DELETE FROM employees . . .
   END fire;
END hire_fire;
/
```

```
CREATE PACKAGE BODY raise_bonus AS
   PROCEDURE give_raise(...) IS
   BEGIN
     UPDATE employees SET salary = . . .
   END give_raise;
   PROCEDURE give_bonus(...) IS
   BEGIN
     UPDATE employees SET bonus = . . .
   END give_bonus;
END raise_bonus;
/
```

Accesul la execuţia procedurilor este dat prin acordarea privilegiului *EXECUTE* asupra pachetului, utilizând următoarele instrucţiuni:

```
GRANT EXECUTE ON hire_fire TO big_bosses;
GRANT EXECUTE ON raise_bonus TO little_bosses;
```

Exemplu: 4 definiţii de proceduri în cadrul unui singur pachet. Două proceduri *standalone* suplimentare şi un pachet sunt create special pentru a furniza acces la procedurile definite în pachetul principal.

```
CREATE PACKAGE BODY employee_changes AS
   PROCEDURE change_salary(...) IS BEGIN ... END;
   PROCEDURE change_bonus(...) IS BEGIN ... END;
   PROCEDURE insert_employee(...) IS BEGIN ... END;
   PROCEDURE delete_employee(...) IS BEGIN ... END;
END employee_changes;

CREATE PROCEDURE hire
   BEGIN
     employee_changes.insert_employee(...)
   END hire;

CREATE PROCEDURE fire
   BEGIN
     employee_changes.delete_employee(...)
   END fire;

PACKAGE raise_bonus IS
   PROCEDURE give_raise(...) AS
   BEGIN
     employee_changes.change_salary(...)
   END give_raise;

   PROCEDURE give_bonus(...)
   BEGIN
     employee_changes.change_bonus(...)
   END give_bonus;
```

Utilizând această metodă, procedurile care lucrează efectiv (cele din pachetul *employee_changes*) sunt definite în cadrul unui singur pachet şi pot partaja variabile declarate global, cursoare etc. Prin declararea procedurilor top-level *hire* şi *fire* şi a unui pachet

suplimentar, *raise_bonus*, se pot acorda privilegii *EXECUTE* selective asupra procedurilor din pachetul principal:

```
GRANT EXECUTE ON hire, fire TO big_bosses;
GRANT EXECUTE ON raise_bonus TO little_bosses;
```

Privilegii tip

În continuare, ne vom referi la utilizarea privilegiilor pentru tipuri, metode şi obiecte.

Privilegii sistem pentru tipurile utilizator (tipuri obiect, vectori şi tablouri imbricate). Aceste privilegii definite de sistemul *Oracle* sunt următoarele:

- CREATE TYPE – permite crearea de tipuri utilizator în propria schemă;
- CREATE ANY TYPE – permite crearea de tipuri utilizator în orice schemă;
- ALTER ANY TYPE – permite modificarea oricărui tip utilizator, din orice schemă;
- DROP ANY TYPE – permite suprimarea unui tip utilizator din orice schemă;
- EXECUTE ANY TYPE – permite utilizarea şi referirea unui tip utilizator în cadrul oricărei scheme.

Privilegii obiect. Singurul privilegiu obiect care se aplică tipurilor utilizator este EXECUTE. Dacă acest privilegiu există asupra unui tip definit de utilizator, atunci utilizatorul poate folosi tipul pentru a defini un tabel sau o coloană dintr-un tabel relaţional sau a declara o variabilă sau un parametru de acel tip.
Privilegiul EXECUTE permite utilizatorilor să invoce metodele din cadrul tipului, inclusiv constructorul tipului. Acest lucru este similar privilegiului EXECUTE asupra unei proceduri PL/SQL stocate.

Privilegii necesare pentru a crea tipuri şi tabele ce utilizează acele tipuri. Pentru a crea un tip, trebuie îndeplinite următoarele condiţii:

- trebuie deţinut privilegiul sistem CREATE TYPE pentru a crea un tip în schema proprie sau CREATE ANY TYPE pentru a crea tipuri în schemele altor utilizatori. Aceste privilegii pot fi acordate explicit sau prin intermediul unui *role*.
- posesorului tipului trebuie să i se fi acordat explicit (nu prin intermediul *role*-urilor) privilegiul obiect EXECUTE pentru a accesa tipurile referite în definiţia respectivului tip, sau să i se fi acordat privilegiul sistem EXECUTE ANY TYPE.
- dacă posesorul tipului doreşte să acorde acces altor utilizatori la tip, atunci el trebuie să fi primit privilegiile EXECUTE asupra tipurilor referite cu GRANT OPTION sau privilegiul sistem EXECUTE ANY TYPE cu ADMIN OPTION. În caz contrar,

posesorul tipului nu are privilegii suficiente pentru a acorda acces altor utilizatori asupra tipului.

Ultimele 2 condiţii dintre cele enumerate anterior sunt necesare şi la crearea unui tabel ce utilizează tipuri, pe lângă cerinţele specifice creării tabelului.

Exemplu de privilegii pentru crearea tipurilor şi a tabelelor ce utilizează tipuri. Presupunem că există 3 utilizatori (*user1, user2, user3*) care deţin role-urile CONNECT şi RESOURCE.

Utilizatorul *user1* efectuează următoarele instrucţiuni LDD în schema sa:

```
CREATE TYPE type1 AS OBJECT (attr1 NUMBER);
CREATE TYPE type2 AS OBJECT (attr2 NUMBER);

GRANT EXECUTE ON type1 TO user2;
GRANT EXECUTE ON type2 TO user2 WITH GRANT OPTION;
```

Utilizatorul *user2* efectuează următoarele comenzi LDD în schema sa:

```
CREATE TABLE tab1 OF user1.type1;
CREATE TYPE type3 AS OBJECT (
   attr3 user1.type2);
CREATE TABLE tab2 (col1 user1.type2);
```

Următoarele instrucţiuni sunt executate cu succes deoarece *user2* are privilegiul EXECUTE asupra lui *user1.type2* cu GRANT OPTION:

```
GRANT EXECUTE ON type3 TO user3;
GRANT SELECT ON tab2 TO user3;
```

Următoarea instrucţiune va eşua deoarece *user2* nu deţine privilegiul EXECUTE asupra lui *user1.type1* cu GRANT OPTION:

```
GRANT SELECT ON tab1 TO user3;
```

Utilizatorul *user3* poate efectua cu succes următoarele instrucţiuni:

```
CREATE TYPE type4 AS OBJECT (attr4 user2.type3);
CREATE TABLE tab3 OF type4;
```

Privilegii asupra accesului la tipuri şi la obiecte. Privilegiile existente la nivel de coloană şi tabel pentru instrucţiunile LMD se aplică atât coloanelor obiect cât şi coloanelor linie. Sistemul *Oracle* defineşte următoarele privilegii pentru tabelele obiect:

- SELECT permite accesarea unui obiect din tabel şi a atributelor sale;
- UPDATE permite modificarea atributelor obiectelor care constituie linii în tabel;
- INSERT permite crearea de noi obiecte în tabel;
- DELETE permite ştergerea liniilor.

Privilegii tabel şi coloană similare se aplică asupra coloanelor obiect. Regăsirea instanţelor nu furnizează informaţii asupra tipului. Clienţii trebuie să acceseze informaţiile asupra tipurilor utilizator pentru a interpreta „imaginile” instanţelor de tipuri. Atunci când un client solicită astfel de informaţii, baza de date *Oracle* verifică existenţa privilegiului EXECUTE asupra tipului.

Considerăm următoarea schemă:

```
CREATE TYPE emp_type (
     eno NUMBER,
     ename CHAR(31),
     eaddr addr_t);
CREATE TABLE emp OF emp_t;
```

Considerăm următoarele două cereri:

```
SELECT VALUE(emp) FROM emp;
SELECT eno, ename FROM emp;
```

Pentru fiecare cerere, *Oracle Database* verifică existenţa privilegiului SELECT al utilizatorului asupra tabelului *emp*. În cazul primei cereri, utilizatorul trebuie să obţină informaţia asupra tipului *emp_type* pentru a putea interpreta datele. Atunci când cererea accesează tipul *emp_type*, sistemul *Oracle* verifică privilegiul EXECUTE al utilizatorului.

Execuţia celei de-a doua cereri nu implică tipuri utilizator, deci nu este necesară verificarea privilegiilor.

Utilizând schema din exemplul din secţiunea precedentă, *user3* poate efectua următoarele cereri:

```
SELECT tab1.col1.attr2 FROM user2.tab1 tab1;
SELECT attr4.attr3.attr2 FROM tab3;
```

Se observă că în ambele cereri *user3* nu are privilegii explicite asupra tipurilor de bază, dar instrucţiunile sunt executate cu succes deoarece posesorii tipului şi tabelului deţin privilegiile necesare cu GRANT OPTION.

Oracle Database verifică privilegiile asupra următoarelor evenimente şi returnează o eroare în cazul în care clientul nu deţine privilegiul pentru acţiune:

- păstrarea unui obiect în *cache* folosind valoarea sa REF determină baza de date *Oracle* să verifice privilegiul SELECT asupra tabelului obiect care îl conţine;
- modificarea unui obiect existent sau ştergerea unui obiect din *cache* determină baza de date *Oracle* să verifice privilegiul UPDATE asupra tabelului obiect destinaţie;
- crearea unui nou obiect determină baza de date să verifice privilegiul INSERT asupra tabelului obiect destinaţie;

- ştergerea unui obiect determină verificarea privilegiului DELETE;
- păstrarea unui obiect de tip utilizator determină verificarea privilegiului EXECUTE asupra obiectului.

Modificarea atributelor unui obiect într-o aplicaţie client scrisă într-un limbaj de generaţia a treia determină ca baza de date *Oracle* să actualizeze întregul obiect. Prin urmare, utilizatorul necesită privilegiul UPDATE asupra tabelului obiect. Deţinerea privilegiului UPDATE doar asupra anumitor coloane ale tabelului obiect nu este suficientă, chiar dacă aplicaţia modifică doar atributele corespunzătoare acelor coloane. Prin urmare, baza de date *Oracle* nu suportă privilegii la nivel de coloană pentru tabelele obiect.

Dependenţe între tipuri. Asemănător cazului obiectelor stocate (proceduri, tabele etc.), referirea unui tip de către alte obiecte se numeşte dependenţă. Există câteva probleme referitoare la tipurile de care depind tabelele. Deoarece un tabel conţine date care se bazează pe definiţia tipului pentru a putea fi accesate, orice modificare asupra tipului determină ca toate datele stocate să devină inaccesibile. Modificările care pot cauza acest lucru apar atunci când privilegiile necesare solicitate de către tip sunt revocate sau tipul sau tipurile dependente sunt şterse. Dacă apare una dintre aceste acţiuni, atunci tabelul devine invalid şi nu poate fi accesat.

Un tabel care este invalid din cauza unor privilegii care lipsesc poate deveni valid şi accesibil în mod automat dacă privilegiile necesare sunt acordate din nou. Un tabel care este invalid din cauză că un tip dependent a fost şters nu mai poate fi accesat din nou, iar singura acţiune permisă este suprimarea tabelului.

Din cauza efectelor severe pe care le pot cauza retragerea unui privilegiu asupra unui tip sau suprimarea unui tip, instrucţiunile SQL REVOKE şi DROP TYPE implementează implicit o semantică restrânsă. Aceasta înseamnă că, dacă tipul utilizator din aceste instrucţiuni are tabele sau tipuri dependente, atunci este primită o eroare iar instrucţiunea eşuează. Dacă este utilizată clauza FORCE, atunci aceste instrucţiuni sunt executate întotdeauna cu succes. Dacă există tabele dependente, atunci acestea sunt invalidate.

3. *Role*-uri (introducere)

Gestiunea şi controlul privilegiilor devin mai simple prin utilizarea *role*-urilor, care sunt grupuri, având un nume, de privilegii înrudite care vor fi acordate ca grup utilizatorilor şi altor *role*-uri. În cadrul unei baze de date, fiecare nume de *role* trebuie să fie unic, diferit de numele utilizatorilor sau al altor *role*-uri. Spre deosebire de obiectele schemei, *role*-urile nu sunt conţinute în vreo schemă. Prin urmare, un utilizator care a creat un *role* poate fi suprimat fără niciun efect asupra *role*-ului.

Role-urile sunt utilizate pentru a uşura administrarea unui sistem *end-user* şi a privilegiilor asupra obiectelor schemei şi sunt cel mai adesea menţinute în *Oracle Internet Directory*. *Role*-urile nu trebuie să fie folosite de către dezvoltatorii de aplicaţii, deoarece privilegiile pentru accesarea obiectelor schemei în cadrul construcţiilor programatice stocate trebuie să fie acordate în mod direct.

3.1. Proprietăţile *role*-urilor

- *Administrarea redusă a privilegiilor*: în loc de a acorda acelaşi set de privilegii în mod explicit mai multor utilizatori, acestea pot fi acordate unui *role,* iar apoi doar acest *role* trebuie să fie acordat fiecărui membru al grupului de utilizatori.
- *Gestiunea dinamică a privilegiilor*: Dacă privilegiile unui grup trebuie să se schimbe, atunci doar privilegiile din *role* trebuie modificate. Domeniile de securitate ale tuturor utilizatorilor cărora le-a fost acordat *role*-ul reflectă automat modificările acestuia.
- *Disponibilitatea selectivă a privilegiilor*: *Role*-urile acordate unui utilizator pot fi activate sau dezactivate selectiv. Acest lucru permite controlul asupra privilegiilor unui utilizator în orice situaţie.
- *Gradul de conştientizare al aplicaţiei*: Dicţionarul datelor înregistrează *role*-urile existente, deci pot fi proiectate aplicaţii care interoghează dicţionarul şi activează (sau dezactivează) automat *role*-urile selectiv atunci când un utilizator încearcă să execute aplicaţia cu ajutorul unui nume de utilizator dat.
- *Securitate specifică aplicaţiei*: Utilizarea *role*-urilor poate fi protejată cu ajutorul unei parole. Aplicaţiile pot fi create astfel încât să activeze un *role* doar dacă este furnizată parola corectă.

Administratorii de baze de date creează adesea *role*-uri pentru o aplicaţie de baze de date. DBA-ul acordă unui *role* toate privilegiile necesare pentru rularea aplicaţiei. Apoi, el acordă acest *role* (*secure application role*) altor *role*-uri sau utilizatori. O aplicaţie poate avea mai multe *role*-uri diferite, fiecăruia fiindu-i acordat un set diferit de privilegii care permit un acces mai mult sau mai puţin restricţionat în timpul utilizării aplicaţiei.

DBA-ul poate crea un *role* cu parolă pentru a preveni utilizarea neautorizată a privilegiilor acordate *role*-ului. În mod obişnuit, o aplicaţie este proiectată astfel încât, atunci când demarează, ea activează propriul *role*. Ca rezultat, un utilizator al aplicaţiei nu trebuie să cunoască parola pentru un *role* al aplicaţiei.

3.2. Utilizări comune ale role-urilor

În general, un *role* este creat cu unul din următoarele scopuri:

- pentru a gestiona privilegiile pentru o aplicaţie de baze de date (*role*-uri aplicaţie);
- pentru a gestiona privilegiile pentru un grup de utilizatori (*role*-uri utilizator).

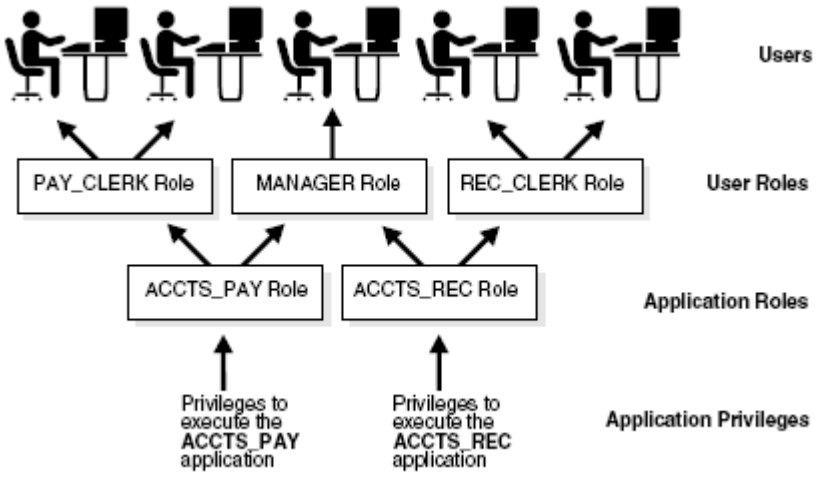

Figura 4. Utilizarea *role*-urilor

Role-uri aplicaţie. Unui astfel de *role* îi sunt acordate toate privilegiile necesare pentru a rula o aplicaţie de baze de date. Apoi, aceste *role*-uri sunt acordate altor *role*-uri sau anumitor utilizatori.

Role-uri utilizator. Un astfel de *role* se creează pentru un grup de utilizatori ai bazei de date cu cerinţe comune de privilegii.

3.3. Acordarea şi revocarea role-urilor

Privilegiile sistem sau cele asupra obiectelor schemei pot fi acordate unui *role*, iar un *role* poate fi acordat unui utilizator al bazei de date sau unui alt *role* (dar nu lui însuşi). *Role*-urile nu pot fi acordate circular, adică un *role* X nu poate fi acordat unui *role* Y dacă acestuia i-a fost acordat deja *role*-ul X.

Pentru a permite disponibilitatea selectivă a privilegiilor, *Oracle Database* permite aplicaţiilor şi utilizatorilor să activeze şi să dezactiveze *role*-uri. Domeniul de securitate al unui utilizator include privilegiile tuturor *role*-urilor activate curent pentru utilizator şi exclude privilegiile *role*-urilor dezactivate curent.

Un *role* acordat unui alt *role* se numeşte *role* acordat indirect şi poate fi activat sau dezactivat explicit pentru un utilizator. La activarea unui *role* care conţine alte *role*-uri, sunt activate implicit toate *role*-urile acordate indirect ale acestuia.

Role-urile pot fi acordate (sau revocate) utilizatorilor sau altor *role*-uri prin următoarele metode:

- *Oracle Enterprise Manager 10g Database Control*
- Instrucţiunile SQL GRANT şi REVOKE.

Privilegiile sunt acordate şi revocate de la *role*-uri utilizând aceleaşi opţiuni. *Role*-urile pot fi acordate sau revocate utilizatorilor utilizând sistemul de operare pe care rulează *Oracle*, sau prin servicii de reţea.

Cine poate acorda sau revoca role-uri? Orice utilizator care deţine privilegiul sistem GRANT ANY ROLE poate acorda sau revoca orice *role* cu excepţia unui *role* global. Acest privilegiu sistem este foarte puternic. Orice utilizator căruia i-a fost acordat un *role* cu ADMIN OPTION îl poate acorda sau revoca . Această opţiune permite puteri administrative *role*-urilor în mod selectiv.

3.4. Domenii de securitate ale *role*-urilor şi utilizatorilor

Fiecare *role* şi utilizator au propriul domeniu unic de securitate. Domeniul de securitate al unui *role* include privilegiile acordate *role*-ului plus acele privilegii ale *role*-urilor acordate. Domeniul de securitate al unui utilizator include privilegii asupra tuturor obiectelor schemei sale, privilegiile care îi sunt acodate şi privilegiile din *role*-urile care îi sunt acordate şi care sunt activate. Un *role* poate fi activat pentru un utilizator şi dezactivat pentru un altul.Acest domeniu include privilegiile şi *role*-urile acordate grupului de utilizatori PUBLIC.

3.5. Blocurile PL/*SQL* şi *role*-urile

Utilizarea *role*-urilor într-un bloc PL/SQL depinde dacă blocul este unul anonim sau neanonim. (procedură, funcţie stocată sau trigger) şi dacă se execută cu drepturile creatorului sau ale apelantului.

Blocuri neanonime cu drepturile creatorului. Toate *role*-urile sunt dezactivate în orice bloc PL/SQL neanonim care se execută cu drepturile creatorului. *Role*-urile nu sunt folosite pentru verificarea privilegiilor şi nu pot fi stabilite în cadrul unei proceduri cu drepturile creatorului.

Vizualizarea SESSION_ROLES arată toate *role*-urile care sunt activate. Dacă un bloc PL/SQL neanonim care se execută cu drepturile creatorului interoghează SESSION_ROLES, cererea nu returnează nicio linie.

Blocuri anonime cu drepturile apelantului. Blocurile PL/SQL neanonime care se execută cu drepturile apelantului şi blocurile PL/SQL anonime sunt executate pe baza privilegiilor acordate prin intermediul *role*-urilor activate. *Role*-urile curente sunt utilizate pentru verificarea privilegiilor în cadrul unui bloc PL/SQL cu drepturile apelantului şi se poate folosi SQL dinamic pentru a seta un *role* în cadrul sesiunii.

3.6. Instrucţiuni LDD şi role-uri

Un utilizator necesită unul sau mai multe privilegii pentru a executa cu succes o instrucţiune LDD, în funcţie de acea instrucţiune. De exemplu, pentru a crea un tabel, utilizatorul trebuie să deţină privilegiul CREATE TABLE sau CREATE ANY TABLE. Pentru a crea o vizualizare asupra unui tabel care aparţine altui utilizator, este necesar privilegiul CREATE VIEW sau CREATE ANY VIEW, dar şi privilegiul obiect SELECT pentru tabel sau privilegiul sistem SELECT ANY TABLE.

Oracle Database evită dependenţele asupra privilegiilor primite prin intermediul *role*-urilor prin restricţionarea utilizării privilegiilor specifice în anumite instrucţiuni LDD. Următoarele reguli evidenţiază aceste restricţii ale privilegiilor privitoare la instrucţiunile LDD:

- Toate privilegiile sistem şi cele asupra obiectelor schemei, care permit unui utilizator să realizeze o operaţie LDD, sunt utilizabile atunci când sunt primite prin intermediul unui *role*. De exemplu:
    - Privilegiile sistem: CREATE TABLE, CREATE VIEW şi CREATE PROCEDURE;
    - Privilegiile obiect: ALTER şi INDEX asupra unui tabel.

  Excepţie: Privilegiul obiect REFERENCES pentru un tabel nu poate fi utilizat pentru a defini cheia externă dacă privilegiul este primit printr-un role.
- Toate privilegiile sistem şi obiect care permit unui utilizator să efectueze o operaţie LMD care este necesară pentru a lansa o instrucţiune LDD nu sunt utilizabile atunci când sunt primite prin intermediul unui *role*. De exemplu, un utilizator care primeşte privilegiul sistem SELECT ANY TABLE sau privilegiul obiect SELECT asupra unui tabel printr-un *role* nu poate utiliza niciun privilegiu pentru a crea o vizualizare asupra unui tabel care aparţine altui utilizator.

Următorul exemplu clarifică utilizarea permisă şi restricţionată a privilegiilor primite prin intermediul *role*-urilor.

Presupunem că unui utilizator i-a fost acordat:

- un *role* care are privilegiul sistem CREATE VIEW;
- un *role* care are privilegiul obiect SELECT asupra tabelului *employees*, dar utilizatorului i-a fost acordat în mod indirect acest privilegiu;
- privilegiul obiect SELECT asupra tabelului *departments* (acordat în mod direct).

În acest caz:

- utilizatorul poate lansa comenzi SELECT atât asupra tabelului *employees* cât şi asupra tabelului *departments*;
- deşi utilizatorul deţine atât privilegiul CREATE VIEW cât şi SELECT asupra tabelului *employees* prin intermediul unui *role*, utilizatorul nu poate crea o vizualizare utilizabilă asupra tabelului *employees*, deoarece privilegiul obiect SELECT pentru tabelul *employees* a fost acordat prin intermediul unui *role*. Orice vizualizare creată va produce o eroare atunci când este accesată.

- utilizatorul poate crea o vizualizare asupra tabelului *departments*, deoarece utilizatorul deţine privilegiul CREATE VIEW printr-un *role* şi privilegiul SELECT pentru tabelul *departments* în mod direct.

3.7. *Role*-uri predefinite

Următoarele *role*-uri sunt definite automat în *Oracle Database*:

- CONNECT
- RESOURCE
- DBA
- EXP_FULL_DATABASE
- IMP_FULL_DATABASE

Aceste *role*-uri sunt furnizate pentru compatibilitate cu versiunile anterioare ale *Oracle Database* şi pot fi modificate ca orice *role* din baza de date.

Sistemul de operare şi role-urile

În anumite medii, securitatea bazei de date poate fi administrată cu ajutorul sistemului de operare. Acesta poate fi folosit pentru a gestiona acordarea şi revocarea *role*-urilor bazei de date şi pentru a gestiona autentificarea cu ajutorul parolelor. Această funcţionalitate nu este disponibilă în toate sistemele de operare.

Role-uri în medii distribuite

În cazul utilizării *role*-urilor într-un mediu de baze de date distribuite, trebuie să ne asigurăm că toate *role*-urile necesare sunt stabilite ca *role*-uri implicite pentru o sesiune distribuită (distantă). Aceste *role*-uri nu pot fi activate la conectarea la o bază de date distantă din cadrul unei sesiuni a unei baze de date locale. De exemplu, nu se poate executa o procedură distantă care încearcă să activeze un *role* la o locaţie distantă.

Role-uri pentru securitatea aplicaţiilor

Oracle Database furnizează aceste *role*-uri, care pot fi activate doar de către pachetele PL/SQL autorizate. Acest mecanism restricţionează activarea acestor *role*-uri la aplicaţia apelantă.

Securitatea este mai puternică atunci când parolele nu sunt încapsulate în codul sursă al aplicaţiei sau stocate într-un tabel. În loc de acestea, poate fi creat un *role* pentru o aplicaţie sigură, prin specificarea pachetului PL/SQL care este autorizat să activeze *role*-ul. Identitatea

pachetului este utilizată pentru a determina dacă privilegiile sunt suficiente pentru a activa *role*-urile. Înainte de activarea *role*-ului, aplicaţia poate efectua autentificarea şi autorizarea customizate, cum ar fi verificarea dacă utilizatorul a fost conectat printr-un proxy.

Crearea role-urilor pentru securizarea aplicaţiilor

Acestea sunt create cu ajutorul instrucţiunii CREATE ROLE … IDENTIFIED USING…

Exemplu: CREATE ROLE admin_role IDENTIFIED USING hr.admin;

Această instrucţiune indică faptul că:

- *admin_role* este un *role* pentru securizarea unei aplicaţii;
- acest *role* poate fi activat numai de către modulele definite în interiorul pachetului PL/SQL *hr.admin*.

Pentru executarea acestei instrucţiuni, este necesar privilegiul sistem CREATE ROLE.

Atrunci când un astfel de *role* este atribuit unui utilizator, el devine un *role* implicit pentru acel utilizator şi este activat automat la *login*. Un utilizator cu un *role* implicit nu trebuie să fie autentificat în niciun fel pentru a utiliza *role*-ul. De exemplu, parola pentru *role* nu este cerută sau necesară.

Pentru a restricţiona un *role* doar la utilizarea specificată de clauza IDENTIFIED USING, se poate realiza una dintre următoarele acţiuni:

- imediat după acordarea unui astfel de *role* unui utilizator, se lansează o instrucţiune ALTER USER cu clauza DEFAULT ROLE ALL EXCEPT *role*. Apoi, *role* poate fi utilizat doar de către aplicaţiile care execută pachetul autorizat.
- la asignarea *role*-urilor, se utilizează GRANT ALL EXCEPT role.

Role-urile care sunt activate în interiorul unei proceduri cu drepturile apelantului au efect chiar şi după ieşirea din procedură. Prin urmare, putem avea o procedură dedicată care tratează activarea *role*-ului pentru restul sesiunii.

4. Limite ale resurselor utilizatorului

Se pot stabili limite asupra cantităţii diferitelor resurse ale sistemului disponibile fiecărui utilizator ca parte a domeniului său de securitate. Procedând astfel, se poate preveni consumul necontrolat al unor resurse importante ale sistemului, cum ar fi timpul CPU.

Caracteristica referitoare la limitarea resurselor este foarte folositoare în sistemele mari, multiuser, în care resursele sistemului sunt foarte scumpe. Consumul excesiv al acestor resurse de către unul sau mai mulţi utilizatori poate fi în detrimentul altor utilizatori ai bazei de date. În sistemele single-user sau multiuser, însă pe o scară mică, caracteristica referitoare la resurse nu este atât de importantă, deoarece consumul de resurse ale sistemului de către utilizator are o probabilitate mai mică de a produce detrimente.

Limitele resurselor pentru utilizator sunt gestionate prin *Database Resource Manager*. Se pot stabili preferinţele referitoare la gestiunea parolelor utilizând profiluri, fie stabilite individual fie folosind un profil implicit pentru mai mulţi utilizatori. Fiecare bază de date *Oracle* poate avea un număr nelimitat de profiluri. *Oracle* permite administratorului de securitate să activeze sau să dezactiveze asigurarea limitelor de resurse ale profilului, în mod universal.

Stabilirea limitelor pentru resurse determină o uşoară degradare a performanţelor atunci când utilizatorii creează sesiuni, deoarece *Oracle* încarcă toate datele referitoare la limitelele de resurse pentru fiecare utilizator la fiecare conexiune la baza de date.

4.1. Tipuri de resurse sistem şi limite ale acestora

Oracle Database poate limita utilizarea câtorva tipuri de resurse ale sistemului, inclusiv a timpului CPU şi a citirilor logice. În general, se poate controla fiecare dintre aceste resurse la nivel de sesiune, apel, sau ambele.

Nivelul sesiune. De fiecare dată când utilizatorul se conectează la baza de date, este creată o sesiune. Fiecare sesiune consumă timp CPU şi memorie pe calculatorul care rulează *Oracle Database*. Pot fi stabilite câteva limite ale resurselor la nivelul sesiunii.

Dacă un utilizator depăşeşte o limită a resurselor la nivelul sesiunii, sistemul *Oracle* termină (*rollback*) instrucţiunea curentă şi returnează un mesaj care indică faptul că a fost atinsă acea limită. În acest punct, toate instrucţiunile precedente din tranzacţia curentă sunt intacte, iar singurele operaţii pe care utilizatorul le poate efectua sunt COMMIT, ROLLBACK, sau deconectarea (caz în care tranzacţia curentă este salvată). Toate celelalte operaţii produc o

eroare. Chiar şi după salvarea sau anularea tranzacţiei utilizatorul nu mai poate lucra în sesiunea curentă.

Nivelul apel. De fiecare dată când este executată o instrucţiune SQL, sunt parcurşi câţiva paşi pentru procesarea acesteia. Pe durata procesării, sunt făcute câteva apeluri la baza de date în cadrul diferitelor faze de execuţie. Pentru a preveni aceste apeluri de la a utiliza sistemul în mod excesiv, *Oracle Database* permite stabilirea câtorva limite ale resurselor la nivelul apelului.

Dacă un utilizator depăşeşte o limită a resurselor la nivel de apel, atunci *Oracle Database* opreşte procesarea instrucţiunii, anulează efectele acesteia şi returnează o eroare. Instrucţiunile precedente din tranzacţia curentă rămân intacte, iar sesiunea utilizatorului rămâne conectată.

Timp CPU. Atunci când instrucţiunile SQL şi alte tipuri de apeluri sunt făcute către *Oracle Database*, o anumită cantitate de timp CPU este necesară pentru procesarea apelului. Apelurile medii necesită o cantitate mică de timp CPU. O instrucţiune SQL ce implică o cantitate mare de date ar putea consuma o cantitate mare de timp CPU, reducând timpul CPU disponibil pentru alte procesări.

Pentru a preveni utilizarea necontrolată a timpului CPU, se pot stabili limite fixe sau dinamice asupra timpului CPU pentru fiecare apel şi cantitatea totală de timp CPU utilizat pentru apelurile *Oracle* în timpul unei sesiuni. Limitele sunt stabilite şi măsurate în sutimi de secundă CPU utilizate de către un apel sau o sesiune.

Citiri logice. Intrările/ieşirile constituie una dintre cele mai scumpe operaţii într-un sistem de baze de date. Instrucţiunile SQL care necesită multe operaţii I/O pot monopoliza memoria şi utilizarea discului şi pot determina ca alte operaţii ale bazei de date să intre în competiţie pentru aceste resurse.

Pentru a feri sursele singulare de operaţiile I/O excesive, *Oracle Database* permite limitarea citirilor blocurilor de date logice pentru fiecare apel şi fiecare sesiune. Citirile blocurilor de date logice includ citirile blocurilor de date atât din memorie cât şi de pe disc. Limitele sunt stabilite şi măsurate în numărul de citiri de blocuri efectuate de către un apel sau în timpul unei sesiuni.

Limitarea altor resurse. *Oracle Database* asigură şi limitarea câtorva altor resurse la nivel de sesiune:

- limitarea numărului de sesiuni concurente pentru fiecare utilizator;

- limitarea timpului *idle* pentru o sesiune. Dacă timpul dintre apelurile dintr-o sesiune atinge limita timpului *idle*, atunci tranzacţia curentă este anulată, sesiunea este încheiată, iar resursele acesteia sunt returnate sistemului. Următorul apel primeşte o eroare care indică faptul că utilizatorul nu mai este conectat la instanţă. Această limită este stabilită printr-un număr de secunde.
- Limitarea timpului de conexiune pentru fiecare sesiune. Dacă durata unei sesiuni depăşeşte această limită, atunci tranzacţia curentă este anulată, sesiunea este încheiată, iar resursele acesteia sunt returnate sistemului.
- Limitarea cantităţii de spaţiu privat din SGA (System Global Area), utilizat pentru ariile SQL private, pentru o sesiune. Această limită este importantă doar în sistemele care utilizează configuraţia de server partajat. Altfel, ariile SQL private sunt localizate în PGA (Program Global Area) Această limită este stabilită ca un număr de octeţi de memorie din SGA-ul unei instanţe.

4.2. Profiluri

În general, noţiunea de profil se referă la o colecţie de atribute care se aplică unui utilizator, permiţând un singur punct de referinţă pentru fiecare mai mulţi utilizatori care partajează exact aceleaşi atribute. Profilurile utilizatorilor din *Oracle Internet Directory* conţin un domeniu larg de atribute care pot fi folosite de către director şi autentificarea fiecărui utilizator. Atributele profilurilor pot include restricţii asupra resurselor sistemului, dar pentru acest scop este preferat *Database Resource Manager*.

Determinarea valorilor pentru limitele resurselor

Înainte de crearea profilurilor şi stabilirea limitelor pentru resursele asociate lor, trebuie determinate valorile adecvate pentru fiecare limită. Aceste valori se pot baza pe tipul operaţiilor pe care un utilizator obişnuit le poate efectua. De exemplu, dacă o clasă de utilizatori nu efectuează, în mod normal, un număr mare de citiri din blocurile logice de date, atunci limitele LOGICAL_READS_PER_SESSION şi LOGICAL_READS_PER_CALL se stabilesc corespunzător.

De obicei, modalitatea cea mai bună pentru a determina valorile adecvate ale limitelor resurselor pentru un profil de utilizator dat este de a strânge informaţii istorice despre fiecare tip de utilizare de resursă. De exemplu, administratorul de baze de date sau de securitate poate utiliza clauza AUDIT SESSION pentru a aduna informaţii despre limitele CONNECT_TIME, LOGICAL_READS_PER_SESSION şi

LOGICAL_READS_PER_CALL.

Se pot aduna statistici pentru alte limite utilizând caracteristica *Monitor* a lui *Oracle Enterprise Manager* (sau SQL*Plus), mai exact monitorul *Statistics*.