

CUPRINS

10. Structura bazei de date	2
10.1. Structura fizică a bazei de date	2
10.1.1. Fișiere de date	2
10.1.2. Fișiere de reluare	3
10.1.3. Fișiere de control	4
10.2. Structura logică a bazei de date	5
10.2.1. Blocuri de date	5
10.2.2. Extensii	7
10.2.3. Segmente	9
10.2.4. Spații tabel	11
10.2.5. Scheme de obiecte	15
10.3. Dicționarul datelor	16
Bibliografie	20

10. Structura bazei de date

- Baza de date *Oracle*:
 - este o colecție de date tratate unitar;
 - este creată pentru a asigura stocarea și extragerea informațiilor.
- Orice bază de date are o structură fizică și o structură logică. Deoarece structura fizică este separată de cea logică, stocarea fizică a datelor poate fi administrată fără să afecteze accesul la structurile logice.

10.1. Structura fizică a bazei de date

- Structura fizică a bazei de date *Oracle* presupune existența unui set de fișiere binare prin intermediul cărora se realizează stocarea fizică a datelor.
- Aceste fișiere sunt împărțite în următoarele categorii:
 - fișiere de date (*Datafiles*);
 - fișiere de reluare (*Redo Log Files*);
 - fișiere de control (*Control Files*).

10.1.1. Fișiere de date

- Sunt fișiere fizice ale sistemului de operare care stochează:
 - datele tuturor structurilor logice ale bazei (obiecte create de utilizator);
 - informația necesară funcționării sistemului.
 - Primul fișier de date creat este cel care stochează dicționarul datelor. Dimensiunea acestui fișier este de cel puțin 150 MB.
- Caracteristicile fișierelor de date:
 - dimensiunea fișierelor de date poate fi schimbată ulterior creării acestora și poate fi extinsă automat până la o valoare specificată, atunci când spațiul alocat este depășit (alocare dinamică a spațiului);
 - unul sau mai multe fișiere de date formează o unitate logică numită spațiu tabel (*tablespace*);
 - un fișier de date poate fi asociat unui singur spațiu tabel și unei singure baze de date.



- ❖ Informațiile din fișierele de date pot fi accesate în timpul operațiilor uzuale asupra bazei și stocate în memoria *cache*.
 - ❖ Dacă un utilizator interoghează date dintr-un tabel al bazei și informațiile cererii solicitate nu sunt încă în memoria *cache*, atunci acestea vor fi preluate din fișierul de date și stocate în memorie.
- Alocarea unui fișier de date se face conform sintaxei următoare:

```
DATAFILE ['nume_fișier']  
[SIZE întreg [ {K | M} ] ] [REUSE]  
[clauza_autoextend]
```

- În cazul spațiilor tabel *undo*, pentru fiecare fișier de date asociat trebuie specificată o dimensiune.
- Pentru celelalte spații tabel, clauza *SIZE* poate fi omisă dacă fișierul există deja.
- Opțiunea *REUSE* permite sistemului să reutilizeze fișierele deja existente. Dacă fișierul specificat nu există, sistemul *Oracle* va ignora această clauză și îl va crea.
- Dacă se dorește ca fișierele de date să fie auto-extensibile, atunci se precizează *clauza_autoextend*.

10.1.2. Fișiere de reluare

- Înregistrează toate modificările care au loc asupra datelor bazei (indiferent dacă au fost permanentizate sau nu) și care nu au fost scrise încă în fișierele de date.
- O bază de date *Oracle* conține două sau mai multe fișiere de reluare.
 - Specificarea acestora se face în momentul creării sau modificării bazei.
 - Fișierele de reluare sunt utilizate în manieră circulară. Cele care au fost folosite în întregime, pot fi arhivate până când sistemul le va reutiliza.
- Fișierele de reluare asigură protecția bazei de date în cazul defectărilor.
 - De exemplu, dacă operațiile bazei sunt întrerupte din cauza unei avarii de curent, datele din memorie nu pot fi scrise în fișierele de date și sunt pierdute temporar. Atunci când baza de date este repornită, informațiile vor putea fi recuperate din cele mai recente fișiere de reluare.
 - Deoarece și aceste fișiere pot fi alterate, sistemul *Oracle* permite utilizarea fișierelor de reluare multiplexate. Aceasta presupune că pentru fiecare fișier de reluare se creează un grup de copii ale sale care sunt localizate pe discuri separate.

Orice modificare făcută asupra unuia dintre membrii grupului se propagă la întregul grup. Dacă un fișier de reluare este pierdut sau afectat, poate fi utilizată o copie a acestuia de pe alt disc.

- Vizualizările *V\$LOG* și *V\$LOGFILE* din dicționarul datelor furnizează informații despre grupurile fișierelor de reluare și membrii acestora.

10.1.3. Fișiere de control

- Sunt fișiere binare de dimensiune redusă, necesare pentru pornirea și funcționarea bazei de date.
 - Fiecare fișier de control este asociat unei singure baze de date și conține informații despre structura fizică a acesteia (numele și data creării bazei, informații despre spațiile tabel, numele și locația fișierelor de date și a fișierelor de reluare, informații despre operațiile de *backup* sau despre *checkpoint*-urile efectuate etc.).
- Orice bază de date *Oracle* deține cel puțin un fișier de control, care este creat odată cu aceasta. Ca și în cazul fișierelor de reluare, sistemul *Oracle* permite existența fișierelor de control multiplexate. În mod implicit, sistemul realizează cel puțin o copie a fișierului de control pe durata creării bazei de date.
- La pornirea unei instanțe *Oracle*, sistemul folosește fișierul de control pentru a identifica baza și a determina dacă aceasta este în stare validă pentru utilizare. De asemenea, sunt identificate fișierele de reluare necesare execuției operațiilor bazei de date.
- Fișierele de control reflectă automat schimbările (creare, redenumire sau ștergere) care au loc la nivelul fișierelor de date sau de reluare. Informațiile din fișierele de control pot fi modificate doar de *server*-ul *Oracle*.
- Dacă un fișier de control devine inaccesibil, baza de date nu va funcționa la parametrii corespunzători. Dacă toate copiile fișierelor de control ale bazei sunt pierdute, atunci aceasta trebuie recuperată (*recovery*) înainte de a fi deschisă.
- Interogând anumite vizualizări din dicționarul datelor se pot obține următoarele informații despre fișierele de control:
 - numele și starea fișierelor asociate instanței (*V\$CONTROLFILE*);
 - starea și localizarea tuturor parametrilor (*V\$PARAMETER*);
 - înregistrările conținute (*V\$CONTROLFILE_RECORD_SECTION*).

10.2. Structura logică a bazei de date

- Utilizarea adecvată a structurilor logice determină alocarea optimă a spațiului de pe disc în vederea obținerii unei baze de date eficiente.
- Componentele structurii logice a unei baze de date *Oracle* sunt:
 - blocurile de date (*data block*);
 - extensiile (*extent*);
 - segmentele (*segment*);
 - spațiile tabel (*tablespace*);
 - obiectele schemei (*schema object*).
- Gestiunea dinamică a spațiului de pe disc pe măsura utilizării bazei de date, se realizează prin trei niveluri de granularitate:
 - segmentul;
 - extensia;
 - blocul.
- Spațiile tabel, segmentele, extensiile și blocurile permit definirea logică a organizării fizice a bazei de date și efectuarea legăturii dintre nivelul fizic și nivelul logic al acesteia.
- Obiectele schemei (tabele, vizualizări, vizualizări materializate, secvențe, unități de program, sinonime, indecși, grupări, dimensiuni, legături de baze de date) sunt structuri logice care referă în mod direct datele bazei.

10.2.1. Blocuri de date

- Sistemul administrează spațiul de stocare al fișierelor de date prin unități logice numite blocuri de date.
- Blocul reprezintă cea mai mică unitate *I/O* folosită de baza de date, corespunzătoare unui bloc fizic de octeți de pe disc.
- Dimensiunea blocului de date este definită în momentul creării bazei și poate fi modificată ulterior.
 - Aceasta trebuie să reprezinte un multiplu al dimensiunii blocurilor fizice de la nivelul sistemului de operare.
 - Modificarea dimensiunii blocurilor logice se face prin intermediul parametrului *DB_BLOCK_SIZE*.

- Din punct de vedere structural, blocul de date *Oracle* cuprinde:
 - un antet (*header*), care conține informații generale referitoare la bloc:
 - adresa;
 - tipul segmentului căruia îi aparține;
 - un catalog al tabelelor (*table directory*);
 - cuprinde informații despre tabelele care au date (înregistrări) stocate în blocul respectiv;
 - un catalog al liniilor (*row directory*);
 - conține informații despre liniile situate în bloc (de exemplu, adresa acestora);
 - odată ce s-a alocat spațiu pentru catalogul liniilor, acesta nu poate fi revendicat de celelalte porțiuni ale blocului de date; sistemul îl va utiliza doar atunci când sunt introduse linii noi în bloc;
 - un spațiu pentru date (*data space*);
 - este format din linii de date care conțin informațiile stocate în tabele sau indecși;
 - un bloc poate cuprinde una sau mai multe linii de date; dacă o linie nu încapă într-un bloc, atunci aceasta poate fi stocată în două sau mai multe blocuri înlănțuite (de exemplu, dacă tabelul conține o coloană de tip *LOB*);
 - un spațiu liber (*free space*);
 - este alocat pentru inserarea de noi linii sau actualizarea liniilor care necesită spațiu suplimentar;
 - alegerea blocului în care va fi inserată o linie nouă depinde de spațiul liber al acestuia și de valorile parametrilor *PCTFREE* și *PCTUSED*;
 - parametrul *PCTFREE* reprezintă procentul minim din blocul de date care trebuie păstrat liber pentru actualizările liniilor deja existente în bloc; valoarea implicită a acestui parametru este 10%;
 - parametrul *PCTUSED* reprezintă procentul minim al spațiului utilizat din bloc care trebuie atins pentru a permite din nou inserarea unor linii de date; valoarea sa implicită este 40%;
 - într-un bloc, se pot introduce date atâta timp cât dimensiunea spațiului liber este mai mare decât limita fixată de parametrul *PCTFREE*; sistemul *Oracle* va considera acest bloc indisponibil pentru inserarea de noi linii, până când procentajul spațiului utilizat coboară sub valoarea dată de *PCTUSED*.

- spațiul liber poate fi administrat automat de către sistem sau manual de către administrator;
 - pentru anumite tipuri de segmente (de date sau index), sistemul menține una sau mai multe liste de blocuri de date care au dimensiunea spațiului liber mai mare decât valoarea parametrului *PCTFREE*; acestea se numesc liste libere (*free list*); blocurile de date care compun listele libere sunt folosite pentru comenzile *INSERT*.

10.2.2. Extensii

- Extensia este o unitate logică de alocare a spațiului bazei de date, compusă dintr-o mulțime contiguă de blocuri de date (din același fișier de date).
- Una sau mai multe extensii formează un segment.
 - Inițial, segmentul are o singură extensie (*initial extent*);
 - De exemplu, la crearea unui tabel, sistemul alocă acestuia un segment cu o extensie inițială care conține un anumit număr de blocuri de date.
 - Blocurile ce corespund extensiei inițiale sunt rezervate pentru liniile tabelului.
 - Dacă spațiul de stocare alocat segmentului este complet folosit, sistemul *Oracle* alocă pentru acesta o nouă extensie (*next extent*).
- O extensie este alocată atunci când este creat sau extins un segment și este dezalocată când segmentul este șters sau trunchiat.
- Atunci când extensiile sunt eliberate, sistemul modifică fișierele de date (dacă administrarea spațiilor tabel se face local) sau reactualizează dicționarul datelor (dacă administrarea spațiilor tabel se face prin dicționar). În acest fel, dimensiunea spațiului de stocare disponibil este cunoscută în orice moment. Eliberarea unei extensii implică ștergerea datelor existente în blocurile de date alocate acesteia. Blocurile respective vor fi reutilizate pentru extensiile nou create.
- Parametrii care permit definirea dimensiunilor și a limitelor extensiilor în cadrul segmentelor sunt definiți cu ajutorul clauzei *STORAGE*.
 - Aceasta poate fi specificată în momentul creării sau modificării obiectelor bazei de date (tabele, vizualizări materializate, *cluster*-e, indecși, partiții).
 - Clauza *STORAGE* are următoarea sintaxă:

```
STORAGE ( {INITIAL întreg [ {K | M} ]  
          | NEXT întreg [ {K | M} ]  
          | MINEXTENTS întreg
```

```
| MAXEXTENTS { întreg | UNLIMITED}  
| PCTINCREASE întreg  
| FREELISTS întreg  
| FREELIST GROUPS întreg  
| OPTIMAL [ { întreg [ {K | M} ] | NULL} ]  
| BUFFER_POOL {KEEP | RECYCLE | DEFAULT});
```

- *INITIAL* specifică dimensiunea în octeți a extensiei inițiale. Sistemul alocă spațiu pentru această extensie atunci când creează un segment.
- *NEXT* precizează dimensiunea în octeți a următoarei extensii care va fi alocată segmentului. Valoarea maximă a acesteia depinde de sistemul de operare.
- *MINEXTENTS* reprezintă numărul minim de extensii care trebuie alocate atunci când se creează un segment, iar *MAXEXTENTS* numărul total de extensii. Opțiunea *UNLIMITED* determină alocarea automată a extensiilor.
- *PCTINCREASE* specifică procentul de creștere a dimensiunii unei extensii (începând cu a treia extensie), față de dimensiunea extensiei anterioare. Valoarea minimă pentru *PCTINCREASE* este 0, ceea ce presupune că, exceptând extensia inițială, toate extensiile alocate au aceeași dimensiune.
- *FREELISTS* specifică numărul de componente ale fiecărui grup de liste libere pentru un tabel, o partiție, o grupare sau un index (valoarea minimă este 1, iar valoarea maximă depinde de dimensiunea blocurilor de date).
- *FREELIST GROUPS* precizează numărul de grupuri de liste libere pentru obiectele create.
- *OPTIMAL* precizează dimensiunea optimă în octeți pentru segmentele de revenire, opțiunea fiind relevantă doar pentru aceste tipuri de segmente. Sistemul menține dimensiunea specificată prin dezalocarea dinamică a extensiilor care nu mai sunt folosite. Opțiunea *NULL* presupune că nu se specifică o dimensiune optimă pentru segmentele de revenire. În acest caz, sistemul nu va dezaloca niciodată extensiile alocate acestor tipuri de segmente.
- Clauza *BUFFER_POOL* permite specificarea unei zone de memorie *cache* (*buffer pool*) implicite pentru un obiect al schemei. Toate blocurile de date alocate obiectului vor fi stocate în această zonă. Opțiunea *KEEP* permite sistemului să mențină obiectul în memorie, evitând astfel operațiile *I/O*. Opțiunea *RECYCLE* determină depunerea blocurilor de date alocate segmentului într-o zonă de memorie *RECYCLE*. Opțiunea *DEFAULT* indică o zonă *buffer pool* implicită.

10.2.3. Segmente

- Segmentul este un set de extensii care conține toate datele unei structuri logice dintr-un spațiu tabel.
 - Un segment conține cel puțin o extensie.
 - Fiecărui obiect fizic stocat îi corespunde un segment.
- Deoarece nu toate obiectele unei baze de date sunt stocate fizic, există obiecte cărora nu le corespunde niciun segment (de exemplu, vizualizările).
- Segmentul folosește blocuri de date care se găsesc în același spațiu tabel.
- Sistemul permite administrarea automată a spațiului liber din interiorul segmentelor bazei de date. În acest fel, spațiul liber, respectiv cel utilizat al segmentului este monitorizat prin intermediul unor obiecte numite *bitmap*-uri.
- Baza de date *Oracle* conține diferite tipuri de segmente:
 - segmente de date (*data segment*);
 - segmente index (*index segment*);
 - segmente temporare (*temporary segment*);
 - segmente de revenire (*undo segment*).

Segmentele de date

- Sunt definite atunci când este folosită comanda de creare a unui tabel sau a unei grupări.
- Un singur segment de date este folosit pentru stocarea tuturor datelor dintr-un tabel nepartiționat care nu face parte din nici o grupare, dintr-o submulțime a unui tabel partiționat sau dintr-o grupare de tabele.
- Una sau mai multe coloane ale unui tabel pot stoca obiecte de tip *LOB* (de exemplu, documente text, imagini, videoclipuri). În acest caz, *server*-ul *Oracle* stochează datele în segmente separate, numite segmente *LOB*.
- coloană a unui tabel poate fi definită de tip tablou imbricat. Valorile unei coloane de tip tablou imbricat sunt stocate în segmente diferite.
- Parametrii de stocare pentru un tabel sau o grupare, specificați prin clauza *STORAGE*, determină modul de alocare a extensiilor segmentului. Aceștia sunt importanți, deoarece pot afecta eficiența stocării și a accesului la date.
- Pentru vizualizările materializate sistemul creează segmentele de date în aceeași manieră ca și în cazul tabelelor sau grupărilor.

Segmentele index

- Sunt folosite pentru a stoca datele unui index. Fiecare index nepartiționat este conținut într-un singur segment. În cazul indecșilor partiționați, fiecărei submulțimi i se asociază câte un segment index.
- Sistemul creează un segment index de fiecare dată când este folosită comanda *CREATE INDEX*. În această comandă se pot specifica parametrii de stocare pentru extensiile asociate segmentului index și spațiul tabel în care este creat acesta. Spațiul tabel respectiv poate fi diferit de cel care conține segmentul de date al tabelului asupra căruia a fost creat indexul.

Segmentele temporare

- Sunt utilizate de sistem pentru analiza și execuția comenzilor *SQL* care necesită un spațiu temporar de stocare.
- Sistemul alocă în mod automat segmente temporare atunci când este necesar și le suprimă după execuția comenzii *SQL*.
- Segmentele temporare sunt alocate în majoritatea cazurilor de sortare (atunci când operația respectivă nu se poate face în memorie sau dacă folosirea indecșilor nu presupune o soluție mai eficientă).
- Comenzile *CREATE INDEX* și *SELECT* (cu opțiunile *ORDER BY*, *DISTINCT*, *GROUP BY* și operațiile *UNION*, *INTERSECT* sau *MINUS*) pot determina alocarea unui segment temporar. O singură comandă *SQL* poate necesita mai multe segmente temporare (de exemplu, o interogare ce conține simultan clauzele *DISTINCT*, *GROUP BY* și *ORDER BY*).

Segmentele de revenire

- Înregistrează valorile anterioare ale datelor care au fost modificate de către tranzații (permanent sau nu), permițând astfel revenirea la forma inițială a acestora.
- O bază de date conține unul sau mai multe segmente de revenire. Ele sunt folosite pentru a oferi consistență la citire, a anula acțiunea tranzațiilor sau a efectua operațiile de recuperare a bazei de date.
- Segmentele de revenire nu pot fi accesate de către utilizatorii sau administratorii bazei de date. Ele pot fi scrise și citite doar de către sistem. Atunci când o tranzație devine activă, sunt modificate blocurile de date din segmentul de revenire asociat. Sistemul înregistrează în fișierele de reluare toate schimbările care au loc asupra blocurilor de date și informațiile pentru revenire. Astfel, în cazul eventualelor defecțiuni, *Oracle*

poate restaura automat baza de date la starea anterioară, folosind segmentele de revenire corespunzătoare tranzacțiilor care erau active în momentul defecțiunii.

- În versiunile anterioare, administrarea segmentelor de revenire se realiza manual. Începând cu versiunea *Oracle9i* se permite administrarea acestora și în mod automat (*Automatic Undo Management*). Server-ul *Oracle* gestionează crearea, alocarea și optimizarea segmentelor de revenire, menținând valorile vechi ale datelor în spații tabel de revenire.

10.2.4. Spații tabel

- O bază de date este compusă dintr-o mulțime de unități logice de stocare numite spații tabel. Acestea pot fi asociate unei singure baze de date și sunt formate dintr-unul sau mai multe segmente.
- Spațiile tabel sunt utilizate pentru a grupa logic o mulțime de obiecte.
 - De exemplu, pentru a simplifica unele operații de administrare, spațiile tabel pot grupa toate obiectele unei anumite aplicații.
 - Fiecare obiect al bazei are specificat un spațiu tabel în care trebuie să fie creat. Datele care alcătuiesc obiectul sunt apoi stocate în fișierele de date alocate spațiului tabel respectiv. Un fișier de date poate fi alocat unui singur spațiu tabel.
- În orice bază de date *Oracle*, primul spațiu tabel creat este *SYSTEM*, căruia îi va fi alocat automat (în timpul creării bazei de date) primul fișier de date. Pe lângă spațiul tabel *SYSTEM*, baza de date conține și alte spații tabel (*non-SYSTEM*), pentru a stoca logic obiectele sale.

Spațiul tabel *SYSTEM*

- Conține dicționarul datelor, inclusiv unitățile de program stocate.
- Conține segmentul de revenire *SYSTEM*.
- Nu trebuie să conțină date ale utilizatorilor, deși este permis acest lucru.

Spațiile tabel *non-SYSTEM*

- Permit administrarea flexibilă a bazei de date.
- Separă segmentele de revenire, segmentele temporare, segmentele de date și segmentele index.
- Separă datele dinamice de cele statice.
- Controlează spațiul alocat pentru obiectele utilizatorilor.

- Sistemul permite atribuirea implicită a spațiilor tabel.
- De asemenea, pentru fiecare utilizator se poate alocă explicit un spațiu tabel, în care vor fi stocate toate obiectele create de acesta. Folosirea mai multor spații tabel permite flexibilitate în execuția operațiilor bazei de date.
- Spațiile tabel pot fi *online* (accesibile) sau *offline* (neaccesibile).
 - În mod normal, spațiile tabel sunt *online* pentru a permite utilizatorilor accesul la informațiile stocate în ele.
 - Spațiul tabel *SYSTEM* nu poate fi niciodată *offline*.
- Comanda *CREATE TABLESPACE* permite crearea unui spațiu tabel:

```
CREATE [UNDO] TABLESPACE nume_spațiu_tabel
    [DATAFILE specificație_fișier [clauza_autoextend]
        [, specificație_fișier [clauza_autoextend]...] ]
    [MINIMUM EXTENT întreg [ {K | M} ] ]
    [BLOCKSIZE întreg [K] ]
    [ {LOGGING | NOLOGGING} ]
    [DEFAULT clauza_storage]
    [ {ONLINE | OFFLINE} ]
    [ {PERMANENT | TEMPORARY} ]
    [clauza_administrare_extensie]
    [clauza_administrare_segment];
```

- Opțiunea *UNDO* permite crearea unui spațiu tabel de revenire (*undo*). Spațiile tabel *undo* sunt rezervate sistemului, deci utilizatorii nu pot crea obiecte ale bazei de date în acestea. Dacă sunt definite mai multe spații tabel *undo*, numai unul dintre ele poate fi activ la un moment dat. Un spațiu tabel *undo* poate fi suprimat numai dacă nu este activ.
- Clauza *MINIMUM EXTENT* specifică dimensiunea minimă a extensiilor care vor fi alocate spațiului tabel. Această dimensiune se va aplica pentru toate extensiile care vor fi asociate segmentelor din spațiul tabel respectiv. Clauza nu este relevantă pentru spațiile tabel temporare administrate prin dicționarul datelor.
- Pentru specificarea dimensiunii nonstandard a blocurilor de date alocate spațiului tabel se folosește clauza *BLOCKSIZE*. Spațiile tabel temporare nu permit blocuri de date cu dimensiune nonstandard.

- Prin opțiunea *LOGGING* se specifică dacă toate schimbările ce au loc asupra tabelelor, indecșilor și partițiilor care vor avea alocat spațiul tabel definit sunt înregistrate implicit în fișierele de reluare. În modul *NOLOGGING*, datele sunt modificate cu jurnalizare minimă adică, doar pentru a marca extensiile noi cu *INVALID* și pentru a înregistra modificările de la nivelul dicționarului datelor. Clauza *DEFAULT* permite specificarea parametrilor implicați de stocare pentru toate obiectele create în spațiul tabel definit.



- ❖ Valoarea unui parametru de stocare specificat la nivel de segment are prioritate față de cea precizată pentru acesta la nivelul spațiului tabel, exceptând parametrii *MINIMUM EXTENT* și *UNIFORM SIZE*.
 - ❖ Dacă la nivel de segment parametrii de stocare nu au fost specificați explicit, atunci aceștia vor avea valorile specificate pentru spațiul tabel în care se găsesc segmentele respective.
 - ❖ Atunci când parametrii de stocare nu sunt specificați explicit pentru un spațiu tabel, *server-ul Oracle* folosește valorile sistem implicite.
 - ❖ Dacă parametrii de stocare sunt modificați, noile opțiuni se aplică doar pentru extensiile care nu au fost deja alocate.
 - ❖ Există parametri de stocare ce nu pot fi specificați la nivel de spațiu tabel, ci numai la nivel de segment.
- Dacă se folosește opțiunea *ONLINE*, spațiul tabel devine disponibil imediat după ce a fost creat pentru toți utilizatorii care au dreptul de a-l accesa. Această opțiune este implicită. *OFFLINE* presupune că spațiul tabel nu este disponibil după creare, până când opțiunea este modificată la valoarea *ONLINE* prin comanda *ALTER TABLESPACE*.
 - Clauza *PERMANENT* este specificată dacă spațiul tabel va fi folosit pentru a stoca obiecte permanente. Dacă spațiul tabel trebuie să stocheze obiecte temporare (de exemplu, segmentele folosite pentru sortări) atunci este utilizată clauza *TEMPORARY*. În acest caz nu se pot specifica clauzele *EXTENT MANAGEMENT LOCAL* sau *BLOCKSIZE*. Pentru a crea spații tabel temporare gestionate local se utilizează comanda *CREATE TEMPORARY TABLESPACE*.
 - Specificarea modului de administrare a extensiilor alocate spațiului tabel se face prin clauza *_administrare_extensie*.
 - Pentru spațiile tabel permanente, administrate local se poate specifica clauza *_administrare_segment*.

Aceasta are următoarea sintaxă:

```
SEGMENT SPACE MANAGEMENT {MANUAL | AUTO}
```

- Opțiunea *MANUAL* presupune că sistemul administrează spațiul liber folosind liste libere. Dacă se specifică opțiunea *AUTO*, sistemul *Oracle* va administra spațiul liber al segmentelor spațiului tabel folosind un *bitmap*.
- Spațiile tabel pot fi modificate sau eliminate. Modificarea unui spațiu tabel se face prin comanda *ALTER TABLESPACE* care permite adăugarea unor fișiere de date, transferarea fișierelor, schimbarea parametrilor de stocare, schimbarea stării din *online* în *offline* și reciproc, recuperarea spațiului tabel respectiv etc.
- Spațiile tabel pot fi șterse chiar atunci când conțin date. Comanda prin care se șterge un spațiu tabel este *DROP TABLESPACE*. Spațiul tabel *SYSTEM* nu poate fi eliminat deoarece conține dicționarul datelor. De asemenea, nu pot fi eliminate spațiile tabel care conțin segmente active. Sistemul *Oracle9i* șterge odată cu un spațiu tabel și fișierele de date asociate acestuia.
- Informații despre spațiile tabel definite în baza de date se obțin interogând vizualizările *DBA_TABLESPACES* și *V\$TABLESPACE* din dicționarul datelor.



Relația dintre baze de date, spații tabel și fișiere de date presupune că:

- ❖ fiecare bază de date este împărțită din punct de vedere logic în unul sau mai multe spații tabel;
 - ❖ unul sau mai multe fișiere de date sunt create explicit pentru fiecare spațiu tabel, cu scopul de a stoca fizic datele din structurile sale logice;
 - ❖ suma mărimilor tuturor fișierelor de date asociate unui spațiu tabel reprezintă capacitatea totală de stocare a spațiului tabel;
 - ❖ suma capacităților de stocare a spațiilor tabel ale unei baze de date reprezintă capacitatea totală de stocare a bazei.
- Vizualizări din dicționarul datelor care oferă informațiile despre spații tabel, fișiere de date, segmente și extensii:
 - *DBA_TABLESPACES*;
 - *DBA_DATA_FILES*;
 - *DBA_SEGMENTS*;
 - *DBA_EXTENTS*;
 - *DBA_FREE_SPACE*.

10.2.5. Scheme de obiecte

- O schemă reprezintă mulțimea obiectelor bazei de date, aflate în posesia unui utilizator (fiecare utilizator deține o singură schemă). Numele schemei este același cu numele utilizatorului.
- Între spațiile tabel și schemele de obiecte nu există o corespondență biunivocă. Obiectele aceleiași scheme pot fi în spații tabel diferite, iar un spațiu tabel poate conține obiecte din mai multe scheme.
- Pentru a accesa un obiect din propria schemă, utilizatorul poate folosi doar numele acestuia. Pentru referirea unui obiect din schema altui utilizator, trebuie specificat atât numele obiectului, cât și schema din care face parte, prin folosirea notației *schema.obiect*.
- Obiecte ale schemei
 - Tabelul (*table*) este unitatea fundamentală pentru stocarea datelor unei baze *Oracle*. Sistemul permite stocarea partiționată a tabelelor. Dacă un tabel este partiționat, atunci fiecare partiție a sa corespunde unui segment.
 - Vizualizarea (*view*) este o reprezentare logică a datelor din unul sau mai multe tabele sau vizualizări. O vizualizare poate fi privită ca o „interogare stocată”. Vizualizarea nu stochează înregistrări, ea doar referă datele unor tabele de bază (*master table*) care, la rândul lor, pot fi tabele sau vizualizări.
 - O vizualizare materializată (*materialized view*) permite accesul indirect la datele unui tabel, prin stocarea rezultatelor unei interogări. Diferența față de o vizualizare obișnuită, care fiind virtuală nu stochează date, este că vizualizarea materializată necesită alocarea unui spațiu de memorie pentru stocarea liniilor rezultate în urma interogării asociate.
 - Secvența (*sequence*) este un obiect al bazei de date care permite utilizatorilor să genereze automat o listă serială de numere întregi unice. Numerele secvenței sunt independente de tabele, astfel că aceeași secvență poate fi folosită pentru mai multe tabele.
 - Unitățile de program (*program unit*) reprezintă subprograme (proceduri și funcții stocate), pachete, declanșatori și clase *Java*.
 - Sinonimele (*synonym*) sunt nume alternative pentru tabele, vizualizări, secvențe sau unități de program.

- Indexul (*index*) este o structură opțională, asociată tabelelor bazei de date, care conține adresa fizică a fiecărei linii dintr-un tabel.
- Grupările (*cluster*) sunt structuri opționale pentru stocarea datelor din tabele create cu scopul de a permite acces rapid la acestea. O grupare conține unul sau mai multe tabele.
 - Din punct de vedere logic, tabelele unei grupări sunt stocate unitar (în același segment) și, de obicei, sunt folosite împreună, deoarece au coloane comune. Aceste coloane sunt numite chei *cluster* și trebuie indexate, astfel încât liniile grupării să fie recuperate cu minimum de operații *I/O*.
 - Asemenea indecșilor, grupările nu afectează proiectarea aplicațiilor. Datele stocate într-o grupare pot fi accesate prin comenzi *SQL*, în același mod ca și cele dintr-un tabel obișnuit.
- Obiectul dimensiune (*dimension*) definește ierarhia („părinte“/„copil“) dintre perechi sau seturi de coloane. Fiecare valoare de pe un nivel „copil“ este asociată cu o valoare unică de pe nivelul „părinte“.
 - Obiectul dimensiune nu conține date, ci relații logice între tabele. Coloanele unui obiect dimensiune pot proveni din același tabel (denormalizat) sau de la mai multe tabele (în întregime sau parțial normalizate).
- O legătură de baze de date (*database link*) este un obiect al unei scheme dintr-o bază care permite accesarea obiectelor din altă bază. Atunci când se lucrează cu mai multe instanțe, legăturile dintre bazele de date sunt folosite pentru a transmite date între aplicații.

10.3. Dicționarul datelor

- Una dintre cele mai importante componente ale bazei de date *Oracle* este dicționarul datelor.
- Dicționarul datelor include tabele și vizualizări *read-only* ce conțin informații despre baza de date:
 - definițiile tuturor obiectelor din schemele bazei;
 - cantitatea de spațiu alocat pentru obiectele schemelor și cantitatea de spațiu utilizat de acestea la momentul curent;
 - valorile implicite ale coloanelor;
 - constrângerile de integritate;

- numele utilizatorilor *Oracle*;
- privilegiile și *role*-urile acordate fiecărui utilizator;
- informații de auditare etc.
- Dicționarul datelor:
 - este generat automat la crearea bazei de date;
 - este reactualizat de către *server*-ul *Oracle* după fiecare comandă de definire a datelor sau de control al accesului la acestea;
 - reflectă imaginea bazei de date (structura fizică și logică) la un moment dat;
 - este deținut de către utilizatorul *SYS*;
 - se află în spațiul tabel *SYSTEM*.
- Sistemul poate accesa dicționarul datelor pentru a obține informații despre utilizatori, despre obiecte sau despre structurile de stocare. Orice utilizator poate consulta dicționarul datelor pentru a afla informații despre baza de date (documentare sau administrare). Accesul la informațiile din dicționar se face utilizând instrucțiunea *SELECT* din *SQL*.
- Din punct de vedere structural, dicționarul datelor este compus din tabele de bază și vizualizări publice asupra acestora.
 - Tabelele de bază stochează informațiile asociate bazei de date, fiind primele obiecte create. În general, doar sistemul are acces direct la scrierea și citirea datelor din aceste tabele. Majoritatea datelor din tabelele de bază sunt stocate în format criptat și sunt dificil de interpretat direct de către utilizatori.
 - Dicționarul datelor conține un tabel numit *DUAL*, pe care aplicațiile sistemului sau cele create de utilizatori îl pot referi pentru a garanta un rezultat cunoscut. Acest tabel este format dintr-o singură coloană (*DUMMY*) și o singură linie care conține valoarea *X*.
 - Vizualizările decodifică informațiile stocate în tabelele de bază și le sintetizează pentru a fi disponibile utilizatorilor. Vizualizările dicționarului pot fi relative la:
 - toate obiectele din baza de date (prefix *DBA_*);
 - obiectele accesibile unui utilizator (prefix *ALL_*);
 - obiectele unui utilizator (prefix *USER_*);
 - performanțe (prefix *V\$* sau *GV\$*).
- Utilizatorii fără privilegii de administrare pot accesa doar vizualizările prefixate de *USER_* sau *ALL_*. Pentru a obține lista vizualizărilor disponibile se poate interoga vizualizarea *DICTIONARY* care are sinonimul *DICT*.

- Vizualizările prefixate de *DBA_* prezintă o imagine globală asupra bazei de date. Acestea pot fi interogate doar de utilizatorii care au privilegiul *SELECT ANY DICTIONARY*. Pentru a referi o astfel de vizualizare, trebuie utilizată notația cu punct și prefixat numele acesteia cu numele proprietarului său (*SYS*).
- Vizualizările prefixate de *ALL_* reflectă baza de date din perspectiva utilizatorului curent. Acestea oferă informații despre toate obiectele schemei la care are acces utilizatorul curent.
- Vizualizările prefixate de *USER_* conțin informații despre utilizatorul curent, obiectele create de acesta, privilegiile sistem pe care le deține, privilegiile pe care le-a acordat altor utilizatori etc. Exceptând coloana *OWNER*, vizualizările prefixate de *USER_* conțin aceleași coloane ca și celelalte vizualizări.
- În funcție de valorile sufixului (precizat între paranteze), aceste vizualizări furnizează informații despre:
 - utilizatori (*USERS*);
 - coloane specificate în constrângeri (*CONS_COLUMNS*);
 - tabele ale bazei (*TABLES*) și tabele externe (*EXTERNAL_TABLES*);
 - vizualizări (*VIEWS*) și vizualizări materializate (*MVIEWS*);
 - grupări (*CLUSTERS*) și indecși (*INDEXES*);
 - declanșatori (*TRIGGERS*);
 - sinonime (*SYNONIMS*) și secvențe (*SEQUENCES*);
 - obiecte (*OBJECTS*);
 - biblioteci (*LIBRARIES*);
 - politici de securitate (*POLICIES*);
 - privilegii obiect asupra tabelor (*TAB_PRIVS*);
 - tipuri obiect (*TYPES*) și colecție (*COLL_TYPES*) etc.
- Vizualizarea *DBA_OBJECTS* este utilă pentru obținerea de informații referitoare la toate categoriile de obiecte ale unei scheme. Coloanele acestei vizualizări sunt următoarele:
 - *OWNER* (proprietarul obiectului);
 - *OBJECT_NAME* (numele obiectului);
 - *SUBOBJECT_NAME* (numele subobiectelor);
 - *OBJECT_ID* (identificatorul obiectului);
 - *DATA_OBJECT_ID* (identificatorul segmentului ce conține obiectul);
 - *OBJECT_TYPE* (tipul obiectului);

- *CREATED* (data creării obiectului);
- *LAST_DDL_TIME* (data ultimei modificări structurale a obiectului);
- *TIMESTAMP* (specificarea formatului datei);
- *STATUS* (starea obiectului);
- *TEMPORARY* (precizarea că obiectul este temporar);
- *GENERATED* (specificarea modului de generare al obiectului);
- *SECONDARY* (precizarea că obiectul este secundar, creat prin metoda *ODCIIndexCreate* a lui *Oracle Data Cartridge*).

Exemplul 10.1

```
-- proprietarul, numele și tipul obiectelor din baza de date.
SELECT  OWNER, OBJECT_NAME, OBJECT_TYPE
FROM    SYS.DBA_OBJECTS;

-- proprietarul, numele și tipul tuturor obiectelor accesibile
utilizatorului curent.
SELECT  OWNER, OBJECT_NAME, OBJECT_TYPE
FROM    ALL_OBJECTS;

-- numele și tipul obiectelor create de utilizatorul curent.
SELECT  OBJECT_NAME, OBJECT_TYPE
FROM    USER_OBJECTS;
```

- Vizualizările prefixate de *V\$* oferă informații despre performanțele bazei de date.
 - Se numesc vizualizări fixe, deoarece administratorii bazei nu le pot modifica sau șterge.
 - Fiecare instanță are asociat un set propriu de vizualizări prefixate de *V\$*.
 - Fiind virtuale, acestea există în memorie doar în timp ce baza de date este pornită.
 - Prin interogarea acestor vizualizări se obțin informații despre:
 - sesiunile curente (*V\$SESSION*) și procesele active (*V\$PROCESS*);
 - obiectele care sunt blocate la un moment dat și sesiunile care le accesează (*V\$ACCESS*);
 - tranzacțiile active (*V\$TRANSACTION*);
 - procesele *background* (*V\$BGPROCESS*);
 - fișierele de reluare care trebuie arhivate (*V\$ARCHIVE*);
 - numele și numărul tuturor spațiilor tabel asociate fișierelor de control (*V\$TABLESPACE*);
 - numele și starea fișierelor de control asociate instanței bazei de date (*V\$CONTROLFILE*) etc.

Bibliografie

1. Connolly T.M., Begg C.E., *Database Systems: A Practical Approach to Design, Implementation and Management*, 5th edition, Pearson Education, 2005
2. Dollinger R., Andron L., *Baze de date și gestiunea tranzacțiilor*, Editura Albastră, Cluj-Napoca, 2004
3. Oracle and/or its affiliates, *Oracle Database Concepts*, 1993, 2017
4. Oracle and/or its affiliates, *Oracle Database Performance Tuning Guide*, 2013, 2017
5. Oracle and/or its affiliates, *Oracle Database SQL Language Reference*, 1996, 2017
6. Oracle and/or its affiliates, *Oracle Database PL/SQL Language Reference*, 1996, 2017
7. Oracle and/or its affiliates, *Oracle Database Administrator's Guide*, 2001, 2010
8. Oracle and/or its affiliates, *Pro*C/C++ Programmer's Guide*, 1996, 2014
9. Oracle University, *Oracle Database 11g: PL/SQL Fundamentals, Student Guide*, 2009
10. Popescu I., Alecu A., Velcescu L., Florea (Mihai) G., *Programare avansată în Oracle9i*, Ed. Tehnică, 2004