SCHEMA FIZICĂ

Suport Laborator Saptămâna 04

FEAA Master SIA/SDBIS

Cuprins

Schema fizică

În Oracle, spre deosebire de alte sisteme de gestiune a bazelor de date, conceptul de schemă se contopeşte cu cel al unui cont de utilizator. Nu pot fi create scheme fără contul utilizator asociat. Numele schemei va fi identic cu cel al utilizatorului. La nivel conceptual, schema nu e altceva decât un "namespace", adică un nume sub care pot fi grupate mai multe obiecte ale bazei de date: tabele, pachete PLSQL etc.

Din punctul de vedere al schemei fizice ne interesează modul în care obiectele din această schema se mapează pe structurile fizice ale bazei de date (segmente, tablespace-uri etc.), dar şi acele obiecte specifice care permit optimizarea modelului logic: indecşi, tabele de tip cluster, view-uri materializate, tabele partiţionate etc.

Crearea unei scheme noi

Vom începe prin crearea unei utilizator nou şi implicit a schemei asociate, denumită "FB". Ne dorim crearea unei aplicaţii de tip Facebook, dar, evident, la un nivel mult simplificat, asta ca să nu spunem uşor penibil. Comanda de creare a utilizatorului “FB” ar fi:

```
SYS@SQL> create user FB identified by "fb";

User created.

SYS@SQL> grant create session, create table, create sequence, create view, create procedure
to FB;

Grant succeeded.
```

Mai sus am creat utilizatorul "FB" cu parola "Fb1234" şi i-am acordat drepturi de conectare, de creare de noi tabele, secvente, view-uri si proceduri stocate. Vom discuta mai în detaliu despre utilizatori la laboratorul despre securitate.

- *Folosind SqlPlus, realizaţi o conexiune nouă cu utilizatorul "FB".*
- *Din sesiunea FB, încercaţi crearea unei table de test. Folosiţi următoarea comandă:*

```
create table test as select * from dual;
```

Ce obţineţi? Explicaţi.

Tablespace-uri și fișiere de date

Avem creată schema "FB", dar ar trebui sa specificăm şi locul unde Oracle ar trebui să aloce spaţiu pentru tabele, indecşi şi alte astfel de obiecte pe care le vom crea în această nouă schemă.

Oracle alocă spaţiu în aşa-numitele tablespace-uri. Nu vă lasaţi induşi in eroare de denumirea de "tablespace"! Nu e doar locul în care spaţiul pentru tabele poate fi alocat, ci şi pentru alte tipuri de obiecte, cum ar fi indecşii.

Un tablespace poate fi privit şi ca o colecţie de fişiere de date, împreună cu tot felul de proprietăţi prin care se stabileşte modul de alocare, tipul de tablespace etc.

Ne propunem să facem un tablespace nou nouţ, doar pentru schema FB. Pentru început, haideţi să vedem în ce locaţie sunt fişierele de date existente. Folosim un cont administrativ, SYS sau SYSTEM:

```
SYS@SQL> select file_name from dba_data_files;

FILE_NAME
--------------------------------------------
C:\APP\TALEK\ORADATA\TESTDB\SYSTEM01.DBF
C:\APP\TALEK\ORADATA\TESTDB\SYSAUX01.DBF
C:\APP\TALEK\ORADATA\TESTDB\UNDOTBS01.DBF
C:\APP\TALEK\ORADATA\TESTDB\USERS01.DBF
```

Se poate observa că, în cazul de faţă, toate fişiere de date sunt în *"C:\APP\TALEK\ORADATA\TESTDB\"*. Nu e o regulă ca toate fişierele de date să fie în acelaşi director, iar locaţia de pe calculatorul dumneavoastră s-ar putea să fie diferită în funcţie de felul în care a fost configurată baza de date. În fine, vom crea tablespace-ul *FB_TBS* (TBS vine de la tablespace) şi îi vom asocia un fişier de date în aceeaşi locaţie unde se află şi celelalte fişiere de date. Folosiţi un cont administrativ, SYS sau SYSTEM:

```
SYS@SQL> create tablespace FB_TBS datafile 'C:\APP\TALEK\ORADATA\TESTDB\FB_TBS01.DBF'
size 1M autoextend off;

Tablespace created.
```

Dacă folosiţi maşina virtuală, atunci tot ce ţine de configurarea aplicatiei FB a fost deja pregatit. Oricum, e interesant a vedea cum s-a realizat acest lucru. Deci, citiţi mai departe!

Reţineţi! Un fişier de date, la un moment dat, nu poate fi asociat decât unui singur tablespace.

Interogaţi dicţionarul bazei de date şi afişaţi toate tablespace-urile disponibile.

Mai departe vom seta tablespace-ul FB_TBS ca tablespace implicit pentru utilizatorul FB.

```
SYS@SQL> alter user fb default tablespace fb_tbs quota unlimited on fb_tbs;

User altered.
```

În acest moment, tabelele şi alte obiecte care alocă spaţiu din schema FB vor fi create, implicit, în tablespace-ul FB_TBS. Remarcaţi că nu este suficient să setăm doar tablespace-ul implicit, ci trebuie să dăm şi drepturi utilizatorului FB să aloce spaţiu în respectivul tablespace, prin intermediul clauzei "QUOTA".

Este posibil, la un moment dat, să avem obiecte din alte scheme în tablespace-ul FB_TBS?

Segmente, extent-uri și blocuri de date

Reprezentarea fizică a unui obiect din baza de date care alocă spaţiu este segmentul. Cel mai comun tip de segment este cel asociat unei tabele. Putem trage o ochiadă în dicţionarul bazei de date pentru a vedea ce alte tipuri de segmente există deja create:

```
SYS@SQL> select distinct segment_type from dba_segments;

SEGMENT_TYPE
------------------
LOBINDEX
INDEX PARTITION
TABLE SUBPARTITION
ROLLBACK
TABLE PARTITION
NESTED TABLE
LOB PARTITION
SYSTEM STATISTICS
LOBSEGMENT
INDEX
TABLE
TYPE2 UNDO
CLUSTER
```

Mergem mai departe cu aplicaţia noastră FB şi vom crea câteva tabele în schema FB. Va trebui, deci, să vă conectaţi cu utilizatorul FB. Mai ştiţi parola? E "fb".

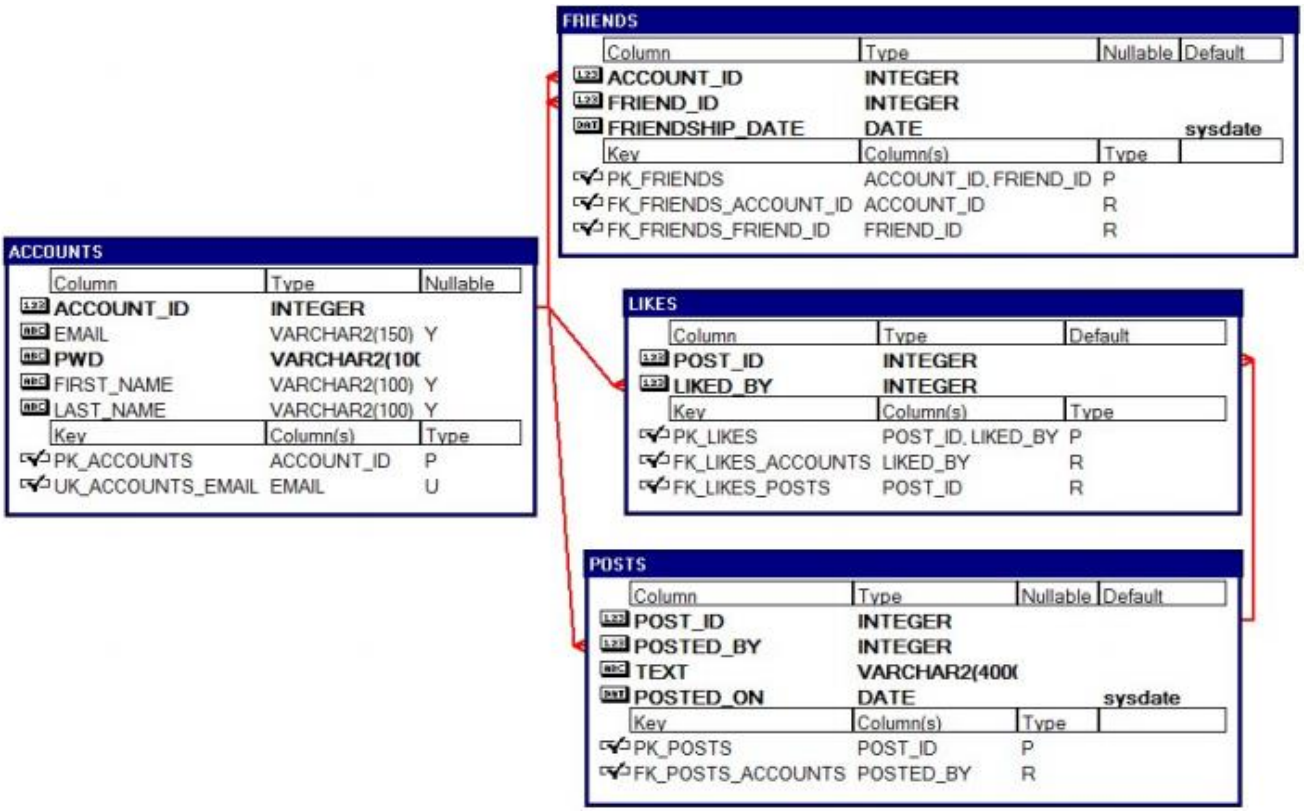
```
-- stores all FB users
create table accounts (
  account_id integer,
  email varchar2(150),
  pwd varchar2(100) not null,
  first_name varchar2(100),
  last_name varchar2(100),
  constraint pk_accounts
     primary key (account_id),
  constraint uk_accounts_email
     unique (email)
);

-- stores all friend relationships
create table friends (
  account_id integer,
  friend_id integer,
  friendship_date date default sysdate not null,
  constraint pk_friends
     primary key (account_id, friend_id),
  constraint fk_friends_account_id foreign key (account_id)
     references accounts(account_id),
  constraint fk_friends_friend_id foreign key (friend_id)
     references accounts(account_id)
);

-- all posts
create table posts (
  post_id integer,
  posted_by integer not null,
  text char(2000) not null,
  posted_on date default sysdate not null,
  constraint pk_posts
     primary key (post_id),
  constraint fk_posts_accounts foreign key (posted_by)
     references accounts(account_id)
);

-- all likes
create table likes (
  post_id integer,
  liked_by integer,
  constraint pk_likes
     primary key (post_id, liked_by),
  constraint fk_likes_posts foreign key (post_id)
     references posts(post_id),
  constraint fk_likes_accounts foreign key (liked_by)
     references accounts(account_id)
);
```

Se vede de la o poştă că avem de-a face o aplicaţie simplistă, fără prea multe finețuri. Avem conturi utilizator, cine cu cine-i prieten, ce a postat fiecare utilizator şi cine a dat "like" pentru mesajele postate. Mai jos şi diagrama entitate-relaţie, ca să fie şi mai clar:



În sfârşit vom şi popula tabelele cu câteva înregistrări de test. Mai întâi, utilizatorii:

```
insert into accounts (account_id, email, pwd, first_name, last_name)
  values (1, 'angelas@yahoo.com', 'UnAlbastruInfinit!', 'Angela', 'Similea');
insert into accounts (account_id, email, pwd, first_name, last_name)
  values (2, 'mt@gmail.com', 'SteauaFaraNume', 'Marius', 'Teicu');
insert into accounts (account_id, email, pwd, first_name, last_name)
  values (3, 'catalin_cris@hotmail.com', 'VorbesteMarea', 'Catalin', 'Crisan');
insert into accounts (account_id, email, pwd, first_name, last_name)
  values (4, 'aurelian_tem@yahoo.com', 'AhAhCeBunSunt!', 'Aurelian', 'Temisan');
insert into accounts (account_id, email, pwd, first_name, last_name)
  values (5, 'stela_popescu@gmail.com', 'Catena', 'Stela', 'Popescu');
insert into accounts (account_id, email, pwd, first_name, last_name)
  values (6, 'arsinel@gmail.com', 'FarmaciaInimii', 'Alexandru', 'Arsinel');
insert into accounts (account_id, email, pwd, first_name, last_name)
  values (7, 'loredana@yahoo.com', 'BunasearaIubite!', 'Loredana', 'Groza');
insert into accounts (account_id, email, pwd, first_name, last_name)
  values (8, 'sbanica@yahoo.com', 'ZanaZanelor', 'Stefan', 'Banica Jr.');
insert into accounts (account_id, email, pwd, first_name, last_name)
  values (9, 'moga@yahoo.com', 'SusPeToc', 'Marius', 'Moga');
insert into accounts (account_id, email, pwd, first_name, last_name)
  values (10, 'andra@gmail.com', 'Mamaruta', 'Andra', 'Maruta');
```

Apoi, prieteniile:

```
insert into friends (account_id, friend_id) values (1, 2);
insert into friends (account_id, friend_id) values (1, 3);
insert into friends (account_id, friend_id) values (1, 4);
```

```
insert into friends (account_id, friend_id) values (1, 5);
insert into friends (account_id, friend_id) values (2, 1);
insert into friends (account_id, friend_id) values (2, 6);
insert into friends (account_id, friend_id) values (3, 1);
insert into friends (account_id, friend_id) values (3, 8);
insert into friends (account_id, friend_id) values (4, 1);
insert into friends (account_id, friend_id) values (4, 5);
insert into friends (account_id, friend_id) values (4, 10);
insert into friends (account_id, friend_id) values (5, 1);
insert into friends (account_id, friend_id) values (5, 6);
insert into friends (account_id, friend_id) values (5, 10);
insert into friends (account_id, friend_id) values (6, 5);
insert into friends (account_id, friend_id) values (6, 1);
insert into friends (account_id, friend_id) values (6, 2);
insert into friends (account_id, friend_id) values (7, 8);
insert into friends (account_id, friend_id) values (7, 9);
insert into friends (account_id, friend_id) values (7, 10);
insert into friends (account_id, friend_id) values (8, 3);
insert into friends (account_id, friend_id) values (8, 7);
insert into friends (account_id, friend_id) values (9, 7);
insert into friends (account_id, friend_id) values (9, 10);
insert into friends (account_id, friend_id) values (10, 4);
insert into friends (account_id, friend_id) values (10, 5);
insert into friends (account_id, friend_id) values (10, 7);
insert into friends (account_id, friend_id) values (10, 9);
```

Urmează mesajele postate de utilizatori:

```
insert into posts (post_id, posted_by, text)
  values (1, 1, 'Un slagar de neuitat: Tu, iubirea mea (cu Marius Teicu)');
insert into posts (post_id, posted_by, text)
  values (2, 1, 'Sa mori de dragoste ranita (Festivalul Mamaia 1996)');
insert into posts (post_id, posted_by, text)
  values (3, 2, 'Mi-am cumparat ochelari noi');
insert into posts (post_id, posted_by, text)
  values (4, 2, 'Ma gasiti online pe: www.mariusteicu.ro');
insert into posts (post_id, posted_by, text)
  values (5, 2, 'Da, am compus muzica pentru cel mai boring film: Racheta alba.');
insert into posts (post_id, posted_by, text)
  values (6, 3, 'Slabeste cu dieta Catalin Crisan!');
insert into posts (post_id, posted_by, text)
  values (7, 4, 'Vai, ce buna-i fata asta (muzica si text Dragos Docan)');
insert into posts (post_id, posted_by, text)
  values (8, 5, 'Am jucat in Nea Marin miliardar... Ehe, ce vremuri!');
insert into posts (post_id, posted_by, text)
  values (9, 5, 'Am fost la farmacie. Da, draga! La Catena!');
insert into posts (post_id, posted_by, text)
  values (10, 6, 'Mi-e foame.');
insert into posts (post_id, posted_by, text)
  values (11, 7, 'Maine ma gasiti la Vocea Romaniei.');
insert into posts (post_id, posted_by, text)
  values (12, 7, 'Am inceput sa lucrez la noul album impreuna cu Fuego!');
insert into posts (post_id, posted_by, text)
  values (13, 8, 'Mi-e dor de Zana Zanelor');
insert into posts (post_id, posted_by, text)
  values (14, 8, 'Mi-e dor de Lavinia');
```

```
insert into posts (post_id, posted_by, text)
  values (15, 8, 'Mi-e dor de Oana Sarbu');
insert into posts (post_id, posted_by, text)
  values (16, 9, 'Mi-am cumparat un papion nou');
insert into posts (post_id, posted_by, text)
  values (17, 10, 'Intre noi e furtuna iar.');
insert into posts (post_id, posted_by, text)
  values (18, 10, 'Ne mintim ca la METEO.');
```

Şi, în sfârşit, "like"-urile:

```
insert into likes (post_id, liked_by) values (1, 3);
insert into likes (post_id, liked_by) values (1, 5);
insert into likes (post_id, liked_by) values (3, 6);
insert into likes (post_id, liked_by) values (6, 8);
insert into likes (post_id, liked_by) values (10, 2);

commit;
```

Schema e pregatită, deşi, deocamdată, nu am acordat o atenţie deosebită schemei fizice, ci mai degrabă celei logice.

Intuiţi ce tipuri de segmente au fost create în schema FB după rularea scripturilor de creare/populare?

Pentru a verifica segmentele create în schema FB, dar şi dimensiunea lor, putem folosi următoarea interogare:

```
SQL> column segment_name format a17
SQL> select segment_name, segment_type, bytes/1024 size_kb from user_segments;

SEGMENT_NAME      SEGMENT_TYPE          SIZE_KB
----------------- ------------------ ----------
UK_ACCOUNTS_EMAIL INDEX                      64
POSTS             TABLE                     128
PK_POSTS          INDEX                      64
PK_LIKES          INDEX                      64
PK_FRIENDS        INDEX                      64
PK_ACCOUNTS       INDEX                      64
LIKES             TABLE                      64
FRIENDS           TABLE                      64
ACCOUNTS          TABLE                      64
```

Tabela POSTS e cea mai mare, având 128KB. Atât are alocat segmentul din spatele tabelei POSTS. La rândul său, acest segment este împărţit în unităţi de alocare mai mici, denumite extent-uri (în română, extensii). Pentru a vedea extent-urile segmentului POSTS, putem folosi următoarea interogare:

```
SQL> select extent_id, bytes, blocks from user_extents where segment_name = 'POSTS';

 EXTENT_ID      BYTES     BLOCKS
---------- ---------- ----------
         0      65536          8
         1      65536          8
```

Se poate observa că fiecare din extent-urile de mai sus conţine câte opt blocuri Oracle. Blocul Oracle este structura fizică cea mai mică în care sunt stocate înregistrările din tabele, dar şi metadate specifice blocului: tranzacţii care afectează înregistrările din bloc, tipul blocului, cărui segment aparţine etc.

Dimensiunea blocului de date Oracle se decide la crearea unei noi baze de date. Implicit este 8KB.

Cum puteţi afla dimensiunea blocului bazei de date Oracle cu care lucraţi acum?

Fiecare înregistrare stocată în blocul Oracle poate fi localizată prin intermediul unei adrese precise, denumită ROWID. Spre exemplu, pentru a putea a afla ROWID-ul postării cu ID-ul 10, putem executa următoarea interogare:

```
SQL> column text format a20 trunc
SQL> select rowid, text from posts where post_id = 10;

ROWID              TEXT
------------------ --------------------
AAAWddAAHAAAAAzAAA Mi-e foame.
```

Având acest ROWID putem şti exact unde se află înregistrarea corespunzătoare. Dat fiind că acest ROWID depinde de locaţia fizică a înregistrării, nu vă aşteptaţi să obţineţi aceeaşi valoare a ROWID-ului ca cel de mai sus.

Putem decodifica un ROWID folosind pachetul *DBMS_ROWID*. Spre exemplu, ROWID-ul de mai sus poate fi decodificat astfel:

```
SQL> set verify off
SQL> select DBMS_ROWID.ROWID_OBJECT(&&rowid) "object",
DBMS_ROWID.ROWID_RELATIVE_FNO(&&rowid) "file", DBMS_ROWID.ROWID_BLOCK_NUMBER(&&rowid)
 "block", DBMS_ROWID.ROWID_ROW_NUMBER(&&rowid) "row" from dual;

Enter value for rowid: 'AAAWddAAHAAAAAzAAA'

    object       file      block        row
---------- ---------- ---------- ----------
     91997          7         51          0
```

Deci, înregistrarea cu acel ROWID e parte din obiectul cu ID-ul 91997. La nivel fizic, înregistrarea e undeva prin fişierul de date cu ID-ul 7, în blocul 51, şi s-ar părea că e prima

înregistrare din blocul 51. Putem să ne prindem repede cine-i obiectul cu ID-ul 91997 şi care e fişierul 7:

```
SQL> column owner format a10
SQL> column object_name format a10
SQL> select owner, object_name from dba_objects where object_id = 91997;

OWNER      OBJECT_NAM
---------- ----------
FB         POSTS

SQL> column file_name format a50
SQL> select file_name from dba_data_files where file_id = 7;

FILE_NAME
--------------------------------------------------
C:\APP\TALEK\ORADATA\TESTDB\FB_TBS01.DBF
```

Alocarea spațiului

Ori de câte ori adăugăm înregistrări sau creăm noi segmente, Oracle trebuie să aloce spaţiu pentru noile date. Alocarea spaţiului se face pe mai multe paliere: la nivel de tablespace, la nivel de segment şi chiar la nivelul blocului Oracle.

Un block Oracle, spre exemplu, stochează înregistrările de jos în sus, spre header-ul blocului care conţine tot felul de metadate. Din punctul de vedere al alocării ne interesează proprietăţile *PCTFREE* şi *PCTUSED*. *PCTFREE* e procentul minim de spaţiu liber în bloc de la care se consideră că blocul nu mai poate "primi" noi înregistrări. *PCTUSED* este procentul de utlizare al blocului sub care se consideră că blocul poate fi folosit pentru înregistrări noi. Ca exemplu, dacă presupunem că *PCTFREE* este 20, iar *PCTUSED* este 40, atunci Oracle va insera în blocul oracle până când spaţiul liber în bloc ajunge să fie 20%. Din acest moment, în acel bloc nu se mai pot adăuga noi înregistrări. Oricum, e posibl ca înregistrări din acel bloc să fie şi sterse, iar spaţiul liber în bloc să devină mai mare de 20%. Chiar şi aşa, Oracle nu va considera acel bloc un candidat pentru noi "INSERT"-uri decât dacă gradul de utlizare al blocului va fi de sub 40%. Aceste proprietăţi, deşi se aplică blocului Oracle, sunt setate la nivelul segmentului.

Pentru a ţine evidenţa blocurilor libere la nivelul segmentului, se pot folosi două tipuri de management al spaţiului: MANUAL şi AUTOMAT. Evidenţa manuală a spaţiului implică crearea unor aşa-numite FREELIST-uri, stocate la nivelul header-ului segmentului, în care sunt consemnate blocurile libere (vezi logica PCTFREE, PCTUSED). Varianta automată, în locul FREELIST-urilor, foloseşte nişte mapări pe biţi (bitmaps) pentru a ţine evidenţa alocării spaţiului. Tipul de management al spaţiului, AUTOMAT sau MANUAL, se stabileşte la crearea tablespace-ului, aşa cum vom vedea imediat.

Determinaţi tipul de managment al spaţiului pentru segmentele din schema FB.

Mai departe, să luăm exemplul tabelelor *ACCOUNTS* şi *POSTS*. Spuneam că alocarea la nivelul tablespace-urilor în numele segmentul Oracle se face prin extent-uri. Iată:

```
SQL> column segment_name format a10
SQL> select segment_name, extent_id, blocks from user_extents where segment_name in
('ACCOUNTS', 'POSTS');

SEGMENT_NA  EXTENT_ID     BLOCKS
---------- ---------- ----------
ACCOUNTS            0          8
POSTS               0          8
POSTS               1          8
```

Se poate observa că tabela *ACCOUNTS*, dat fiind că e o tabelă mică, are doar un singur extent. Pe de altă parte, segmentului tabelei *POSTS*, dat fiind că are mai multe înregistrări, s-ar părea că nu i-a fost suficient un singur extent, prin urmare, Oracle i-a mai adăugat unul nou pentru a face loc mai multor înregistrări.

Din punctul de vedere al alocării extent-urilor, există câteva proprietăţi interesante care pot fi setate la nivelul segmentului, prin intermediul clauzei *STORAGE*, cum ar fi: *INITIAL*, *NEXT*, *PCTINCREASE*, *MINEXTENTS*, *MAXEXTENTS*. Totuşi abordarea lor e diferenţiată în funcţie de tipul de tablespace şi de tipul de management al spaţiului la nivel de segment, de aceea nu vom insista cu aceste proprietăţi, mai ales că multe din aceste clauze nu mai sunt relvante în contextul tablespace-urilor de tip "locally managed".

Asemenător segmentului, unde are loc un management al spaţiului intra-segment, şi la nivelul tablespace-ului trebuie să existe un mecanism de control/evidenţă a spaţiului liber şi a celui ocupat. Exită două tipuri de tablespace-uri:

- de tip "*dictionary manged*", unde managementul spaţiului în tablespace se face prin tabele din dicţionar (a se evita folosirea lor);
- de tip "*locally managed*", unde managementul spaţiului în tablespace se face printr-un bitmap stocat în header-ul fişierului de date.

Oracle recomandă folosirea tablespace-urilor de tip "locally managed".

Haideţi să facem două tablespace-uri noi, unul pentru backup-uri şi unul pentru a stoca obiectele de tip index.

```
SQL> create tablespace fb_idx_tbs datafile 'C:\APP\TALEK\ORADATA\TESTDB\FB_IDX_TBS01.DBF'
size 500K autoextend off extent management local autoallocate;

Tablespace created.

SQL> create tablespace fb_bak_tbs datafile 'C:\APP\TALEK\ORADATA\TESTDB\FB_BAK_TBS01.DBF'
size 500K autoextend off extent management local uniform size 100k;

Tablespace created.
```

Care ar fi principala diferenţa dintre clauza AUTOALLOCATE şi UNIFORM?

Pentru a evita să tot dăm drepturi de alocare pe fiecare tablespace în parte, putem da dreptul de UNLIMITED TABLESPACE pentru utilizatorul FB. În medii de producţie n-o să facem asta, dar pentru testele de acum e în regula:

```
SQL> grant unlimited tablespace to fb;

Grant succeeded.
```

De ce ar fi de evitat dreptul de UNLIMITED TABLESPACE pentru mediile de producţie?

Acum, haideţi să vedem ce indecşi avem în schema FB. Vom folosi contul FB.

```
SQL> column segment_name format a20
SQL> select segment_name, tablespace_name from user_segments where segment_type =
'INDEX';

SEGMENT_NAME         TABLESPACE_NAME
-------------------- ------------------------------
PK_ACCOUNTS          FB_TBS
PK_FRIENDS           FB_TBS
PK_LIKES             FB_TBS
PK_POSTS             FB_TBS
UK_ACCOUNTS_EMAIL    FB_TBS
```

Pentru a muta index-ul *PK_POSTS* din tablespace-ul *FB_TBS* în *FB_IDX_TBS*, putem da următoarea comandă.

```
SQL> alter index pk_posts rebuild tablespace FB_IDX_TBS;

Index altered.
```

- Verificaţi că, într-adevăr, index-ul PK_POSTS a fost mutat în tablespace-ul FB_IDX_TBS.
- De ce ar fi benefic să izolăm indecşii într-un tablespace separat?
- Scrieţi o procedură sau un bloc anonim PL/SQL prin care să se mute toţi indecşii din tablespace-ul FB_TBS în FB_IDX_TBS.

Mai departe, vom face o copie a tabelei *POSTS*, dar o vom aloca în tablespace-ul *FB_BAK_TBS*.

```
SQL> create table posts_bak
  2  tablespace fb_bak_tbs
  3  as
  4  select * from posts;

Table created.
```

Observaţi clauza "TABLESPACE" folosită la crearea tabelei *POSTS_BAK*, care indică exact tablespace-ul ţintă.

Cum ar fi să mai adăugăm înregistrări în tabela noastră de backup?

```
SQL> insert into posts_bak select t.* from posts_bak t cross join posts_bak;
insert into posts_bak select t.* from posts_bak t cross join posts_bak
            *
ERROR at line 1:
ORA-01653: unable to extend table FB.POSTS_BAK by 13 in tablespace FB_BAK_TBS
```

După cum se poate observa, Oracle protestează cu un mesaj de eroare prin care ne transmite că nu mai poate aloca spaţiu în tablespace-ul *FB_BAK_TBS*, pentru tabela *POSTS_BAK*. Ce-i de făcut? În primul rând putem să redimensionăm fişierul de date asociat tablespace-ul *FB_TBS* şi sa-l facem mai mare. Dimensiunea sa curenta e de aproximativ 500KB.

```
SQL> select file_name, bytes/1024 size_kb, autoextensible from dba_data_files where
tablespace_name = 'FB_BAK_TBS';

FILE_NAME                                          SIZE_KB AUT
-------------------------------------------------- ---------- ---
C:\APP\TALEK\ORADATA\TESTDB\FB_BAK_TBS01.DBF              504 NO
```

Vrem sa-l redimensională la 2MB.

```
SQL> alter database datafile 'C:\APP\TALEK\ORADATA\TESTDB\FB_BAK_TBS01.DBF' resize 2M;

Database altered.
```

Ce dimensiune va raporta sistemul de operare pentru fişierul FB_BAK_TBS01.DBF? Verificaţi cu "File Explorer".

Dacă încercăm din nou comanda de inserare, ar trebui să funcţioneze.

```
SQL> insert into posts_bak select t.* from posts_bak t cross join posts_bak;

324 rows created.

SQL> commit;

Commit complete.
```

Dar dacă mai încercăm încă o data aceeaşi comandă?

```
SQL> insert into posts_bak select t.* from posts_bak t cross join posts_bak;
insert into posts_bak select t.* from posts_bak t cross join posts_bak
*
ERROR at line 1:
ORA-01653: unable to extend table FB.POSTS_BAK by 13 in tablespace FB_BAK_TBS
```

Pare că ne-am întors de unde am plecat. Aparent, cei 2MB alocaţi nu sunt suficienţi. Deşi unii au impresia că DBA-ul Oracle asta face totată ziua, adică face tablespace-uri şi alocă mai mult spaţiu celor existente, de fapt să tot stai să faci asta non-stop nu e deloc amuzant. Prin urmare, putem folosi clauza AUTOEXTEND la nivelul fişierului de date.

```
SQL> alter database datafile 'C:\APP\TALEK\ORADATA\TESTDB\FB_BAK_TBS01.DBF' autoextend on
next 1M maxsize 400M;

Database altered.
```

- Încercaţi din nou INSERT-ul în tabela POSTS_BAK. Funcţionează?
- Verificaţi dimensiunea fişierului FB_BAK_TBS01.DBF în File Explorer. Cum explicaţi?
- Ce se poate face dacă nu mai există spaţiu liber pe discul pe care se află FB_BAK_TBS01.DBF? Cum mai putem adăuga spaţiu tablespace-ului FB_BAK_TBS?