

Capitolul 5

Gestiunea tabelelor

- Partea 2. Modificarea schemei -

Comanda ALTER TABLE

```
ALTER TABLE [schema.]table
[options ADD, MODIFY, etc - prezentate anul trecut la BD1]
[PCTFREE integer] [PCTUSED integer]
[INITRANS integer] [MAXTRANS integer]
[STORAGE storage_clause]
[DROP drop_clause] ...
[ALLOCATE EXTENT [( [SIZE integer [K|M] ]
                    [DATAFILE 'filename']
                    [INSTANCE integer] )]
[ PARALLEL { integer } |
  NOPARALLEL ]
[ CACHE | NOCACHE ]
[{ ENABLE | DISABLE } { enable_clause | TABLE LOCK }]
[{ ENABLE | DISABLE } ALL TRIGGERS ]]
```

ALTER TABLE - ALLOCATE

- ❑ `ALLOCATE EXTENT` alocă explicit o nouă extensie pentru acea tabelă.
- ❑ `SIZE` specifică dimensiunea extensiei în octeți (bytes). Se poate folosi K ori M pentru a specifica KB sau MB. În cazul absenței acestei clauze Oracle determină dimensiunea pe baza valorilor de `STORAGE` ale tabelului.
- ❑ `DATAFILE` specifică numele fișierului (aferent tablespace-ului în care se găsește tabelul) în care se va alocă noua extensie. În lipsă, alegerea este făcută de Oracle.

ALTER TABLE - ALLOCATE

- ❑ INSTANCE face acea extensie disponibilă pentru instanța specificată. Acest parametru este util în conjuncție cu Oracle Real Application Clusters.
- ❑ Alocarea explicită a unei extensii nu afectează dimensiunea următoarei extensii care va fi alocată pe baza parametrilor NEXT și PCTINCREASE (prezentați în capitolul anterior)

ALTER TABLE - cont

PARALLEL

NOPARALLEL

CACHE

NOCACHE

ENABLE

DISABLE

- ❑ Sunt folosite pentru a schimba setările curente.
- ❑ Aceste clauze au fost prezentate în cursul trecut.

ALTER TABLE - LOCK

- ❑ **ENABLE TABLE LOCK** - Activează posibilitatea obținerii de blocări asupra tablei de comenzile DDL.
- ❑ Aceste comenzi nu se pot executa dacă blocarea nu este permisă.
- ❑ **DISABLE TABLE LOCK** duce implicit la interzicerea operațiilor DDL asupra tablei.

ALTER TABLE - TRIGGERS

ENABLE ALL TRIGGERS

DISABLE ALL TRIGGERS

- ❑ Permit activarea / dezactivarea tuturor declanșatorilor asociați unei tabele
- ❑ Pentru activarea / dezactivarea unui singur declanșator se poate folosi comanda ALTER TRIGGER.
- ❑ Clauza DROP – specifica ștergerea unei constrângeri de integritate.

Comanda ALTER TABLE - cont

- ❑ PCTFREE,
- ❑ PCTUSED,
- ❑ INITRANS,
- ❑ MAXTRANS,
- ❑ STORAGE

Schimbă valorile și opțiunile care există în acel moment. Explicația acestor parametri s-a făcut la descrierea comenzii CREATE TABLE

Parametrii de stocare

- În cazul lui ALTER TABLE, semnificația noilor valori ale acestor parametri este următoarea:
- NEXT – În momentul alocării unei noi extensii Oracle va folosi noua valoare a lui NEXT după care creșterea se face pornind de la această valoare și cea a lui PCTINCREASE (vezi formula din capitolul precedent pentru dimensiunea extensiei **n**)

Parametrii de stocare - cont

- PCTINCREASE – schimbarea acestuia va afecta doar dimensiunea noilor extensii care se alocă.
- MINEXTENTS – poate fi schimbată cu orice valoare care e mai mica sau egala decât numărul de extensii din acel moment
- MAXEXTENTS – poate fi schimbată cu orice valoare mai mare sau egală cu numărul de extensii din acel moment

Parametri utilizare bloc

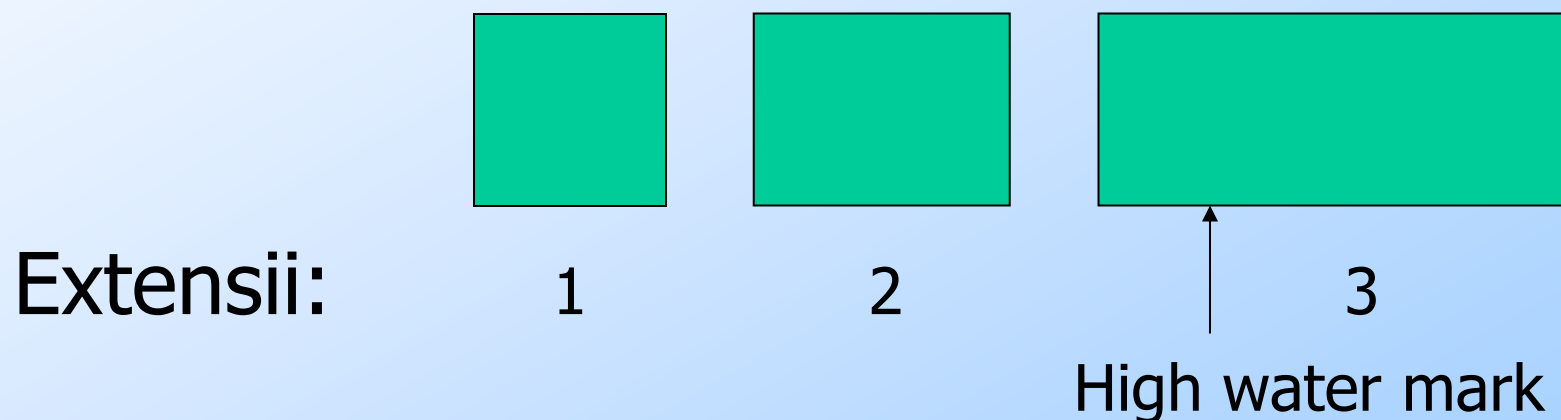
- ❑ PCTFREE – schimbarea sa afectează următoarele operații de inserare. Blocurile 'umplute' după vechea valoare nu sunt afectate decât în momentul în care ele ajung în lista de blocuri cu spațiu liber – FREELIST - deci lista de blocuri în care se pot face operații de inserare.
- ❑ Un bloc ajunge în lista de blocuri cu spațiu liber doar dacă se efectuează ștergeri din el pana sub PCTUSED

Parametri utilizare bloc - cont

- ❑ PCTUSED – orice modificare a acestui parametru afectează toate blocurile din tabelă. Dacă o linie e actualizată sau ștearsă, blocul care o conține va fi testat dacă poate fi pus în lista de blocuri cu spațiu liber.
- ❑ INITTRANS – schimbarea acestui parametru va afecta doar blocurile noi
- ❑ MAXTRANS – vezi observația din cursul anterior. La schimbare va lua automat valoarea 255.

High water mark

- Pentru orice segment (inclusiv deci pentru segmentele conținând tabele) exista un marcaj al ultimului bloc care a fost vreodată utilizat.
- Acest marcaj se numește 'high water mark' (HWM)



De ce se numeste asa?



High water mark

Surse: http://www.tripper.ro/view.php?ce=gal&gal_id=119&id=&limba_id=1 și <http://www.plaiurimioritice.ro/tag/lac-de-acumulare/>

High water mark - cont

- ❑ Pe măsură ce datele sunt inserate în tabelă, marcajul HWM este mutat spre blocuri superioare
- ❑ Acest marcaj NU este resetat în momentul în care sunt șterse linii din tabela (resetarea se face doar în cazul în care se executa TRUNCATE)
- ❑ Când Oracle face o parcurgere completă a unei tabele (full table scan) atunci sunt citite toate blocurile pana la HWM, deci inclusiv blocuri golite ca urmare a ștergerilor de înregistrări (linii)!

High water mark - cont

- Daca se dorește însă 'defragmentarea' tabelii se poate executa ALTER TABLE MOVE care mută o tabelă dintr-un tablespace în altul (cele doua nu sunt neapărat distincte):

ALTER TABLE emp MOVE tablespace2

- În acest caz se păstrează definițiile tuturor constrângerilor de integritate și ale indecșilor, dar aceștia din urma trebuie refăcuți (indecșii sunt bazați pe ROWID iar în procesul de compactare acesta se schimbă).

Exemplu

- ❑ Creare tabela și umplere cu date:

```
create table table_size_test (  
  a char(100), b number )  
storage (initial 65K next 65K pctincrease 0)  
tablespace ts_01;
```

```
begin  
for i în 1 .. 10000 loop -- PL/SQL block  
  insert into table_size_test values  
    (dbms_random.string('X', 100), i);  
end loop;  
end;
```

```
/
```

```
commit;
```

Exemplu - cont

□ Crearea unui index:

```
create index ix_table_size_test on  
table_size_test(a) storage (initial 65K  
next 65K pctincrease 0)  
tablespace ts_02;
```

Exemplu - cont

□ Vizualizare spațiu utilizat:

```
select substr(segment_name,1,20) segment,  
       bytes / 1024 "Size [KB]"  
from user_segments  
where segment_name în ('TABLE_SIZE_TEST',  
    'IX_TABLE_SIZE_TEST');
```

SEGMENT	Size [KB]
-----	-----
TABLE_SIZE_TEST	1280
IX_TABLE_SIZE_TEST	1280

Exemplu - cont

❑ Ștergere din tabela

```
delete from table_size_test where  
    mod(b,2)=0;  
commit;
```

❑ Vizualizare spațiu (rezultat)

SEGMENT	Size [KB]
-----	-----
TABLE_SIZE_TEST	1280
IX_TABLE_SIZE_TEST	1280

Exemplu - cont

- ❑ Alter table move

```
alter table table_size_test move;
```

- ❑ Vizualizare spațiu (rezultat)

SEGMENT	Size [KB]

TABLE_SIZE_TEST	640
IX_TABLE_SIZE_TEST	1280

Exemplu - cont

- ❑ Indexul însă a devenit UNUSABLE:

```
select status from user_indexes  
where index_name = 'IX_TABLE_SIZE_TEST';  
  
STATUS
```

UNUSABLE

- ❑ Îl refacem:

```
alter index ix_table_size_test rebuild;
```

Exemplu - cont

□ Date despre indexul refăcut (acum a devenit valid):

```
select status, bytes/1024
from user_indexes
join user_segments on index_name =
    segment_name
where index_name = 'IX_TABLE_SIZE_TEST';
```

STATUS	BYTES/1024
--------	------------

-----	-----
-------	-------

VALID	704
-------	-----

High water mark - cont

- ❑ Începând cu Oracle 10 se mai poate face ajustarea HWM în cazul segmentelor care utilizează ASSM – Automatic Segment Space Management astfel:
 - Permite schimbarea ROWID-ului liniilor:
`ALTER TABLE emp ENABLE ROW MOVEMENT;`
 - Dăm comanda de shrink:
`ALTER TABLE emp SHRINK SPACE;`
- ❑ Efectul comenzii de shrink este: mută liniile compactându-le și muta HWM. Pentru asta e nevoie de o blocare a tablei dar pentru o perioada scurta de timp.

High water mark - cont

□ Variante ale comenzii:

1. Mutare linii și HWM într-o tabelă:

```
ALTER TABLE emp SHRINK SPACE;
```

2. Mutare linii și HWM într-o tabela plus compactare obiecte dependente (ex. indecși):

```
ALTER TABLE emp SHRINK SPACE CASCADE;
```

3. Muta doar liniile fără sa mute HWM:

```
ALTER TABLE emp SHRINK SPACE COMPACT;
```

High water mark - cont

Restricții pentru SHRINK:

- ☐ Doar în tablespace-uri cu ASSM
- ☐ Nu se pot compacta (lista e mai lungă):
 - Segmente UNDO
 - Segmente temporare
 - Tabele de tip cluster
 - Tabele cu o coloană de tip LONG
 - Indecși de tip LOB
- ☐ Se poate utiliza pachetul de sistem DBMS_SPACE pentru a vedea informații despre spațiul utilizat.

DEALLOCARE SPATIU LIBER

- ❑ Spațiul liber ocupat de un segment (cel de după HWM) poate fi dealocat folosind:

```
ALTER TABLE [schema.]tabela
```

```
DEALLOCATE UNUSED [KEEP int [K | M] ]
```

- ❑ În cazul folosirii KEEP se păstrează o parte a acestui spațiu liber (dimensiunea e data în bytes, KB sau MB).
- ❑ Spațiul astfel dealocat poate fi folosit de alte segmente.

DEALOCARE SPATIU - cont

- ❑ În cazul în care HWM este într-o extensie cu număr mai mic decât MINEXTENTS, se dealocă toate extensiile de după MINEXTENTS.
- ❑ Pentru a dealoca tot spațiul disponibil (pana la HWM), inclusiv în cazul în care HWM e sub MINEXTENTS, se folosește KEEP 0.

Trunchiere

- ❑ Comanda de trunchiere golește o tabelă și resetează HWM.
- ❑ Spațiul ocupat de tabelă este dealocat în afara cazului când se specifica explicit REUSE STORAGE
- ❑ Sintaxa comenzii este:

```
TRUNCATE TABLE [schema.]tabela  
[ { DROP | REUSE } STORAGE]
```
- ❑ Comanda TRUNCATE e o comanda DDL deci este comisa automat și nu se poate face rollback (nu poate fi anulată ca în cazul unui DELETE)

DROP STORAGE

- ❑ În cazul DROP STORAGE:
 - Sunt dealocate toate extensiile superioare lui MINEXTENTS
 - HWM e resetat
 - Valoarea lui NEXT_EXTENT (dimensiunea următoarei extensii care va fi alocată la nevoie) este resetată la valoarea extensiei cu numărul cel mai mic care a fost dealocată
- ❑ În ambele cazuri (REUSE sau DROP), trunchierea afectează toți indecșii tabelului respective.

DROP TABLE

- ❑ Ștergerea unei tabele se face cu DROP TABLE
- ❑ Sintaxa este:

**DROP [schema.] tabela
[CASCADE CONSTRAINTS]**

- ❑ Efectul este ștergerea tablei și a tuturor constrângerilor de integritate aferente (inclusiv cele referențiale – FOREIGN KEY)
- ❑ Dacă nu se specifica CASCADE tabela nu se poate șterge dacă exista constrângeri referențiale care o referă.

Validare structura

- ❑ Se face cu comanda ANALYZE TABLE.
- ❑ Aceasta colectează statistici despre tabelă și le stochează în dicționarul de date.
- ❑ Printre alte opțiuni sunt și cele de:
 - Validare a structurii unei tabele
 - Identificarea liniilor care au migrat sau sunt înlănțuite
- ❑ În cazul validării structurii, toate blocurile tabelei sunt verificate din punct de vedere al integrității.

VALIDATE STRUCTURE - cont

- Sintaxa este:

ANALYZE TABLE [schema.]tabela

VALIDATE STRUCTURE [CASCADE]

- În cazul folosirii opțiunii CASCADE, este validată și structura tuturor indecșilor asociați tablei și se face și o verificare încrucișată între conținutul de date al tablei și cel al indecșilor.

Migrare și înlănțuire

- ❑ ANALYZE TABLE poate fi folosită și pentru detectarea liniilor care au migrat sau a celor înlănțuite (din cauza lui PCTUSED sau pentru că sunt prea voluminoase).
- ❑ Pentru aceasta, întâi se calculează sau se estimează statisticile asupra tabelului respective.
- ❑ Statisticile estimate se fac pe baza unui eșantion de linii (implicit 1064 linii).

Migrare și înlănțuire - cont

- ❑ Sintaxa comenzii în acest caz este:

```
ANALYZE TABLE [schema.]tabela
```

```
{ COMPUTE STATISTICS
```

```
| ESTIMATE STATISTICS
```

```
[SAMPLE integer { ROWS | PERCENT }]
```

- ❑ COMPUTE va genera statistici pornind de la o parcurgere completă a tablei.
- ❑ La ESTIMATE se poate specifica (în linii sau în procente) dimensiunea eșantionului.

Migrare și înlănțuire - cont

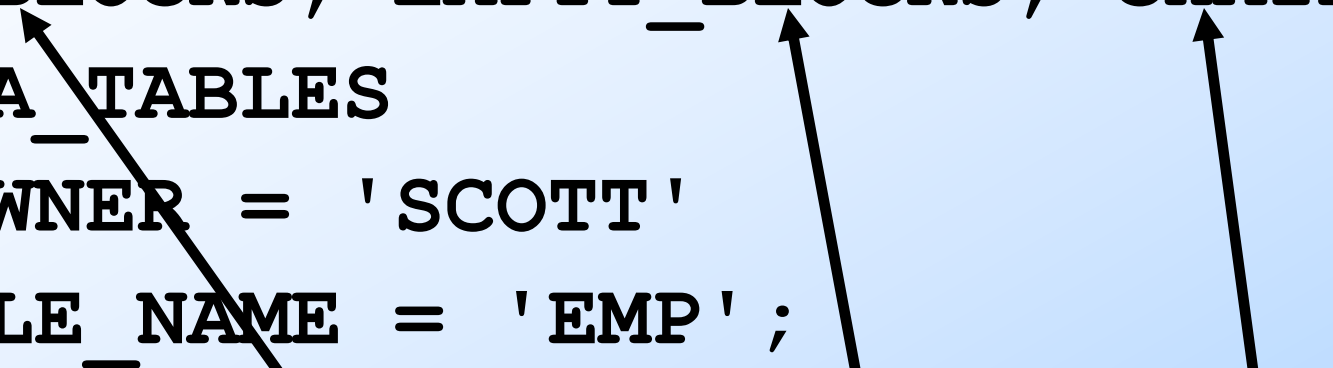
- ❑ După generarea statisticilor, în vederea de dicționar DBA_TABLES există în coloana CHAIN_CNT numărul de linii care sunt migrate sau înlănțuite.
- ❑ în cazul în care un număr mare de linii sunt în aceasta situație trebuie ca tabela sa fie reorganizată pentru a remedia această situație (de exemplu prin recrearea tablei folosind CREATE ... AS SELECT ... ORDER BY)

VEDERI

- ❑ Pe lângă DBA_TABLES se mai pot folosi și vederile DBA_OBJECTS și DBA_SEGMENTS.
- ❑ Toate cele 3 vederi pot fi unite (join) după condiția:
DBA_TABLES.OWNER = DBA_OBJECTS.OWNER= DBA_SEGMENTS.OWNER
AND
DBA_TABLES.TABLE_NAME = DBA_OBJECTS.OBJECT_NAME=
DBA_SEGMENTS.SEGMENT_NAME

Exemplu

```
SELECT BLOCKS, EMPTY_BLOCKS, CHAIN_CNT  
FROM DBA_TABLES  
WHERE OWNER = 'SCOTT'  
AND TABLE_NAME = 'EMP';
```



- Obținem în acest caz un rezultat conținând:
 - Prima coloana conține HWM
 - A doua numărul de blocuri de după HWM
 - A treia numărul de linii (înregistrări) migrate sau înlanțuite



```
SQL>
SQL>
SQL>
SQL>
SQL>
SQL> analyze table scott.emp compute statistics;
```

Table analyzed.

```
SQL> select blocks, empty_blocks, chain_cnt
2   from dba_tables
3   where owner='SCOTT'
4   and table_name='EMP';
```

BLOCKS	EMPTY_BLOCKS	CHAIN_CNT
5	3	0

```
SQL> █
```

DBA_EXTENTS

- ❑ Această vedere poate fi folosită pentru a afla numărul de extensii și alte informații despre ele.
- ❑ Printre coloanele vederii sunt: OWNER, SEGMENT_NAME, EXTENT_ID, FILE_ID, BLOCK_ID, BLOCKS
- ❑ Fiecare linie reprezintă o extensie și în BLOCKS este numărul de blocuri ale acesteia.

Lecturi obligatorii

1. Oracle Database Administrator's Guide (v19c) – Cap 20: Managing Tables.

<https://docs.oracle.com/en/database/oracle/oracle-database/19/admin/>

2. On shrinking table sizes:

<http://www.adp-gmbh.ch/blog/2005/july/20.html>

Sfârșitul capitolul 5