UBD – handouts :

* De fiecare data cand cauti ceva necunoscut, cauti in dictionary :

select table\_name from dictionary where table\_name like 'USER%';

* Obiectele create de un user:

Desc user\_objects;

* Toate constrangerile userului current:

DESC USER\_CONSTRAINTS;

FISIERE :

* Fisiere de control: (v$controlfile)
* Vizualizare fisiere de control :

SHOW PARAMETER CONTROL\_FILES;

* Fisiere de date: (v$datafile)
* Fisiere temporare: (v$tempfile)
* Fisiere de log (v$logfile)
* Informatii despre marimea inregistrarii, numarul total de inregistrari alocate si cele folosite referitoare la parametrii de control, :

SQL> desc v$controlfile\_record\_section

* adaugarea unui nou fisier de **log**

alter database add logfile member 'e:\temp\log2.rdo' to group 1;

BAZA DE DATE:

* baza de date: v$database
* starea instantei si a grupurilor : v$thread
* grupuri si membri : v$log
* Instanta : v$instance
* Parametri: (v$parameter)

TABLESPACE-URI Si FISIERELE LOR:

* **Informatii** despre tablespace-uri (la nivel de baza de date) : desc DBA\_TABLESPACES
* **Informatii** despre tablespace (stare): select \* from v$tablespace;
* **Crearea** unui **tablespace** permanent ‘UBD’ cu un fisier de date UBD1 cu dimensiunea de 1 M, cu sau fara extensie :

CREATE TABLESPACE ubd DATAFILE 'E:/userdata01.dbf’ SIZE 1M EXTENT MANAGEMENT LOCAL UNIFORM SIZE 128K;

CREATE TABLESPACE ubd DATAFILE 'E:/Student/ubd1.dbf' SIZE 1M;

* **Extinderea** spatiului alocat unui **tablespace**

SQL> ALTER DATABASE DATAFILE 'E:/Student/ubd1.dbf' AUTOEXTEND ON NEXT 2M;

SQL> ALTER DATABASE DATAFILE 'E:/Student/ubd1.dbf' RESIZE 2M;

* **Stergerea** din dictionar a unui **tablespace**

SQL> DROP TABLESPACE userdata INCLUDING CONTENTS AND DATAFILES;

FISIERELE TABLESPACE\_URILOR:

* **Adaugarea** unui nou **fisier** de date la un **tablespace**

SQL> ALTER TABLESPACE bd\_data ADD DATAFILE 'E:/Student/ubd1.dbf' SIZE 1M;

* **Informatii** despre **fisierele** de date atasate unui tablespace: desc dba\_data\_files;
* Informatii despre starea fisierelor de **date**: Desc v$datafile
* Informatii despre fisierele de **date** **temporare** : Desc DBA\_TEMP\_FILES
* Informatii despre fisierele **temporare** la nivel de user: desc v$tempfile.
* parametrii bazei de date: desc DATABASE\_PROPERTIES sau v$parameter.

SEGMENTE

* segmente de tip **tabela**

SQL> select owner,segment\_name,segment\_type, tablespace\_name, blocks, extents

from dba\_segments where owner='SCOTT' and segment\_type='TABLE'

* Informatii despre dimensiunea **extensiilor** alocate unui **segment**

SQL> desc dba\_extents

* Informatii despre tabela emp privita ca un segment

SQL> select owner, segment\_name, segment\_type, tablespace\_name, bytes, from dba\_extents

where owner='SCOTT' and segment\_name='EMP';

* Informatii despre **extensiile libere** dintr-un tablespace

SQL> desc dba\_free\_space

* **spatii contigue** dintr-un tablespace

SQL> desc dba\_free\_space\_coalesced

SQL> ALTER TABLESPACE BD COALESCE ; (condensarea blocurilor)

Segmente de undo

* Informatii din dictionar privind segmentele de undo

SQL> desc dba\_rollback\_segs

* Segmentele de undo folosite de instanta curenta

SQL> desc v$rollname

* Statistici despre **segmentele** de **undo**

SQL> desc v$rollstat

* Informatii despre **useri** si **sesiuni**

SQL> desc v$session

* Informatii despre **tranzactii**

SQL> desc v$transaction

* Creare **tablespace** de undo:

create undo tablepace BD\_UNDO datafile ’e:\student\undo\_db01.dbf’ size 2M;

* Creare segment de undo, in interiorul unui tablespace de undo:

SQL> create rollback segment UBD\_UNDO tablespace BD\_UNDO

storage (initial 100k next 100k optimal 4M minextents 20 maxextents 100);

* Informatii despre **blocurile** de undo folosite de tranzactia **curenta**

SQL> SELECT s.username, t.xidusn, t.ubafil, t.ubablk, t.used\_ublk FROM v$session s,

v$transaction t WHERE s.saddr = t.ses\_addr;

* Statistici despre **dimensiunea spatiului** de undo

SQL> desc v$undostat

* **Stergerea** din dictionar a unui segment de undo

SQL> drop rollback segment ubd;

* Informatii despre **segmentele temporare de sortare**

SQL> desc v$sort\_segment

* Informatii despre **sesiuni** si tablespace-ul in care se afla segmentele temporare de **sortare** folosite in sesiunea curenta

SQL> desc v$sort\_usage

* **Setarea** zonei de **memorie** utilizata pentru **sortare** in sesiunea curenta la 10K.

SQL> alter system set sort\_area\_size=10240 deferred;

Exercitii:

* **Inserare** in emp, apoi lista cu **adresa** **tranzactiei**, data initierii starea + numele segmentului de undo pe care il foloseste.

SQL> insert into emp values (997, 'TEST-EX', 'TRANZACT', 1111, sysdate, 100, 0, 10);

SQL> select addr, start\_time, status, xidusn from v$transaction;

* Lista cu nume + stare tablespace alocat pentru segmental temporar de sortare din sesiunea curenta si numar maxim de blocuri de sortare allocate fiecarui segment temporar.

SQL>select tablespace\_name from v$sort\_segment;

SQL>select a.tablespace, a.segtype, c.status,

b.max\_sort\_size, b.max\_blocks, b.max\_used\_size

from v$sort\_usage a, v$sort\_segment b, dba\_tablespaces c

where a.tablespace = b.tablespace\_name

and a.tablespace = c.tablespace\_name;

Tabele

* Informatii despre tabelele din baza de date

SQL> desc dba\_tables;

* Toate **tabelele** userului **current** :

SQL> Select table\_name from user\_tables;

* Vizualizarea ID-rilor pentru fiecare linie din tabela.

SQL> CREATE TABLE emp\_test as select \* from scott.emp;

SQL> SELECT rowid, empno, ename FROM emp\_test;

* **Alocarea** unei **extensii** la o tabela

SQL> ALTER TABLE scott.emp\_test ALLOCATE EXTENT(SIZE 500K DATAFILE ’e:/DISK3/DATA01.DBF’);

* **Stergerea** unei **coloane** dintr-o tabela

SQL> ALTER TABLE scott.emp\_test DROP COLUMN comm CASCADE CONSTRAINTS CHECKPOINT 1000;

* **Redenumirea** unei **coloane** dintr-o tablela

SQL> ALTER TABLE scott.emp\_test RENAME COLUMN sal TO salary;

* **Dezactivarea** unei **coloane** dintr-o tabela

ALTER TABLE scott.emp\_test SET UNUSED COLUMN comm CASCADE CONSTRAINTS;

* **Stergerea** coloanelor **dezactivate** dintr-o tabela

ALTER TABLE scott.emp\_test DROP UNUSED COLUMNS CHECKPOINT 1000;

* **Informatii** despre coloane **dezactivate**

SQL> desc dba\_unused\_col\_tabs;

* **Informatii** despre **obiectele** din baza de date

SQL> desc dba\_objects

SQL> SELECT object\_name, created FROM DBA\_OBJECTS WHERE object\_name like ’EMP%’ AND owner = ‘SCOTT’;

* lista cu **tabelele** **userului** **curent**, tablespace'ul in care e creat si owner:

SELECT OWNER, TABLESPACE\_NAME, TABLE\_NAME FROM DBA\_TABLES WHERE OWNER='SCOTT';

* **copiati** tabelul dept in dept1, **adaugati** coloana telefon in dept1, **aratati** din dictionar structura coloanei.

CREATE TABLE DEPT1 AS SELECT \* FROM DEPT;

ALTER TABLE SCOTT.DEPT1 ADD TELEFON VARCHAR2(30);

SELECT OWNER, TABLE\_NAME, COLUMN\_NAME, DATA\_TYPE FROM **DBA\_TAB\_COLUMNS** WHERE TABLE\_NAME='DEPT1';

* sa se faca o **copie** a tabelei emp. Sa se **verifice** cu cat a **scazut spatiul** liber din tablespaceul permanent dupa creare

create table emp\_ubd as (select \* from emp);

SELECT TABLESPACE\_NAME, COUNT(\*), MAX(BLOCKS), SUM(BLOCKS) FROM DBA\_FREE\_SPACE GROUP BY TABLESPACE\_NAME;

* **aratati** din dictionary proprietarul + numele tabelei partitionate de userul system.

select a.instance\_name, b.owner, b.table\_name, b.tablespace\_name,

b.max\_extents from dba\_tables b, v$instance a, database\_properties c

where b.owner='SCOTT' AND

b.tablespace\_name = c.property\_value and

c.property\_name like 'DEFAULT%PERMANENT%TABLESPACE';

* sa se **creeze** un **view** deptview pe toate coloanele din tabela dept din userul curent

apoi sa se arate din dictionar proprietarul, numele, tipul si data cand a fost creeat.

CREATE VIEW DEPT\_VIEW AS SELECT \* FROM DEPT;

SELECT OWNER, OBJECT\_TYPE, OBJECT\_NAME, CREATED FROM DBA\_OBJECTS WHERE OBJECT\_NAME LIKE 'DEPT\_VIEW';

* sa se faca o **copie** a tabelei emp, emp\_new din userul curent si

sa se **elimine** **coloana** comm, si sa se verifice in dictionar noua structura

create table emp\_new as select \* from emp;

alter table emp\_new drop column comm cascade constraints checkpoint 1000;

select column\_name, data\_type from dba\_tab\_column where table\_name = 'EMP\_NEW';

* in copia emp\_new, **dezactivati** coloana job apoi activati-o la loc.

alter table emp\_new set unused column job cascade constraints;

alter table emp\_new add job varchar2(9);

* sa se aloce tabelei emp\_new o extensie de 100k, in table space ul permanent

aferent userului curent.

select property\_value from database\_properties where property\_name like

'DEFAULT%PERMANENT%TABLESPACE'; -> USERS

alter table emp\_new allocate extent (size 100k);

INDECSI

* **Informatii** din dictionar despre **indecsi**

SQL> desc dba\_indexes

* **Informatii** din dictionar despre **coloanele** **indecsilor**

SQL> desc dba\_ind\_columns

* **Crearea** unui index de tip B-Tree

SQL> CREATE INDEX scott.emp\_name\_idx ON scott.emp(ename) PCTFREE 30 STORAGE(INITIAL 200K NEXT 200K

PCTINCREASE 0 MAXEXTENTS 50) TABLESPACE bd\_data;

* **Crearea** unui index de tip BITMAP

SQL> CREATE BITMAP INDEX scott.dept\_name\_idx ….

* **Stergerea** unui index din dictionar

SQL> DROP INDEX emp\_name\_idx ;

* **Modificarea** **parametrilor** unui index

SQL> ALTER INDEX scott.emp\_name\_idx STORAGE(NEXT 400K MAXEXTENTS 100);

* **Alocarea** unei **extensii** pentru un index de tip B-Tree

SQL> ALTER INDEX emp\_name\_idx ALLOCATE EXTENT (SIZE 200K DATAFILE ’e:/DISK6/indx01.dbf’)

* **Eliberarea** **spatiului** **nealocat** pentru un index de tip B-Tree

SQL> ALTER INDEX emp\_name\_idx DEALLOCATE UNUSED;

* **Mutarea** unui **index** in alt **tablespace**

SQL> ALTER INDEX emp\_name\_idx REBUILD TABLESPACE SYSTEM;

* **Defragmentarea** blocurilor unui index

SQL> ALTER INDEX emp\_name\_idx COALESCE;

* **Startarea** si **stoparea** monitorizarii unui index

SQL> ALTER INDEX emp\_name\_idx MONITORING USAGE

SQL> ALTER INDEX emp\_name\_idx NOMONITORING USAGE

* **Informatii** din **dictionar** despre indecsii **monitorizati**

SQL> desc v$object\_usage

* **Startarea** analizei structurii unui index

SQL> ANALYZE INDEX emp\_name\_idx VALIDATE STRUCTURE

* Informatii din dictionar despre **starea** **indecsilor**

SQL> desc index\_stats

* **creare** index **arbore**, pe coloanele deptno si empno ale tabelei emp\_copy

create table emp\_copy as select \* from emp;

create index emp\_copy\_idx ON emp\_copy(deptno,empno); TABLESPACE test\_lab10;

SELECT index\_name, index\_type, table\_name, status from dba\_indexes WHERE index\_name='EMP\_COPY\_IDX';

* aratati din **dictionar**, numarul de **blocuri** alocate si care e procentul utilizat din **spatiul** alocat lui

ANALYZE INDEX emp\_copy\_idx VALIDATE STRUCTURE;

SELECT name, blocks, used\_space, pct\_used FROM index\_stats

* **aratati** din dictionar **numele** **instantei** **curente**, numele indexilor si tabelele pe care au fost creeati

SELECT instance\_name, index\_name, table\_name FROM dba\_ind\_columns, v$instance WHERE index\_owner = 'SYSTEM';

* monitorizati indexul creat si aratati din dictionar data si ora cand a inceput monitorizarea

alter index\_emp\_copy\_idx MONITORING USAGE;

select start\_monitoring, index\_name from v$object\_usage where table\_name = 'EMP\_COPY';

* Sa se creeze un index de tip arbore emp1\_idx pe coloanele deptno si empno a tabelei emp1, care este o copie a tabelei emp.

CREATE TABLESPACE test\_lab10 DATAFILE 'D:/testlab10.dbf' SIZE 1M EXTENT MANAGEMENT LOCAL AUTOALLOCATE;

create table emp3 as select \* from emp;

create index emp3\_idx ON emp3(deptno,empno) TABLESPACE test\_lab10;

SELECT index\_name, index\_type, table\_name, status from dba\_indexes WHERE index\_name='emp3\_idx';

* sa se faca apoi o extensie de 100 k ptr indexul creat intr-un fisier de date ubd1 cu marimea de 1M

ALTER INDEX emp3\_idx STORAGE(NEXT 100K);

* verificati in dictionar nr de blocuri alocate indexului si care e procentul utilizat din spatiul alocat

ANALYZE INDEX emp3\_idx VALIDATE STRUCTURE

SELECT name, blocks, used\_space, pct\_used, distinct\_keys,lf\_rows, del\_lf\_rows FROM index\_stats ;

* aratati din dictionar numele instantei curente numele indexilor si tablelele aferente

SELECT instance\_name, index\_owner, index\_name, table\_name FROM dba\_ind\_columns, v$instance WHERE index\_owner = 'SYSTEM';