SGBD
TPn° 5, Enseignants : Nakechbandi M
Thème : La Structure physique de la Base de données.

Consignes : TP à me rendre à la fin de la séance : vous m'envoyer votre compte-rendu nakech@free.fr
N'oubliez pas de préciser votre nom. L'objet du message doit être composé comme suit : **da2i_tp4_initials**.

Utiliser la command spool pour conservez une trace écrite de toutes les commandes que vous effectuez, avec les résultats.

Pour mener à bien ce tp on travail sur **2 sessions** (une session d'un utilisateur **dba**, et un autre d'un utilisateur **non dba**)

1. Paramètres d'initialisation . (certaine questions sont traitées en tp2)

a) Testez les deux techniques d'observation des paramètres d'initialisation.(consulter la table **v$parameter)**

b) Affichez le nom de la base, le nom de l'instance et la taille des blocs de données. (consulter la table v$instance)

c) Affichez les nom et taille des fichiers de données(dba_data_files)

d) Trouver le nom de l'utilisateur actuel (consulter la table v$session)

e) Dans quel fichier se trouve le tablespace SYSTEM ?

2. Dictionnaire des données

a) Créez les vues du dictionnaire des données : localiser, consultez puis exécutez le script *catalog.sql* qui se trouve dans `$ORACLE_HOME/rdbms/admin/`. Ne faites pas de copie de ce script, exécutez-le depuis son emplacement (d'autres scripts y sont inclus localement). Vérifiez que l'installation s'est correctement déroulée. (**indication** : l'utilisateur sys est le propriétaire du dictionnaire de données)

b) donner toutes les informations sur les tables sur lesquelles vous avez des droits ;

c) Idem que la question précédente, donner toutes les informations sur les indexe sur lesquelles vous avez des droits ;

d) Idem que la question précédente, donner toutes les informations sur les `triggers` sur lesquelles vous avez des droits ;

e) donner la description des attributs de la table TRANSACTION du tp1 ;

f) donner la liste des contraintes (avec leur statut) créées au cours des TPs précédents ;

g) donner les informations sur les contraintes de type clé primaire que vous aviez créées au cours des TP précédents.

3. Gestion des fichiers de contrôle

a) Quel est le nom des fichiers de contrôle ? Où sont-ils localisés ? (**v$controlfile, …)**

b) Ajoutez un fichier de contrôle à votre base.
indication : `alter system set controlFiles='/.../oradata/…' …)`

4. Gestion des fichiers redo-log

a) Quels sont les fichiers redo-log de votre base ? (**v$logfile**)

b) Combien y a-t-il de groupes et de membres ? (SELECT group#, sequence#, bytes, members, status FROM v$log;)

c) Ajouter un fichier de redo-log au groupe 1.(alter database add logfile member '/…/oradata/…/f_log.rdo' to group 1;) (l'opération peut êytre répétée pour tous les groupes)
verifier(`SELECT * FROM v$logfile;`)

d) Supprimer du membre ajouté au groupe 1:

```
alter database drop logfile member '/…/oradata/…/f_log.rdo' ;
```

5. Gestion des tablespaces et des fichiers de données (à faire dans la session non dba)

a) Créez des tablespaces permanents avec les noms, tailles et caractéristiques suivants :
- DATA01, 10M (`CREATE TABLESPACE DATA01 DATAFILE '/.../oradata/.../ DATA01.dbf' size 10M;`)
- DATA02, 100M
- TEMP, 2M : temporaire (CREATE TEMPORARY TABLESPACE temp…)
- RONLY, 1M : en lecture seule (`ALTER TABLESPACE RONLY READ ONLY;`)

b) Vérifiez que tous les tablespaces ont été correctement créés. (consulter le dictionnaire, voir cours)
exemple :

```
select tablespace_name,status from dba_tablespaces
                                    where tablespace_name = 'RONLY';
```

c) - Consulter les volumes des tablespaces (`select tablespace_name,bytes from dba_free_space;`)

- Allouez 500K de plus au tablespace DATA02 et vérifiez.

- Trouver le nombre de blocks utilisé par DATA01 (

```
select  BLOCKS from dba_data_files
                                    where TABLESPACE_NAME='DATA01';)
```

d) Déplacez le tablespace DATA01. (`alter tablespace DATA01 OFFLINE …`)

f) Créer une table t dans le tablespace RONLY (`CREATE TABLE t(ID int) TABLESPACE RONLY;`)
Votre conclusion

g) Supprimez le tablespace RONLY et vérifiez.

h) Tester la procédure suivante qui liste des infos sur l'espace réservé pour chaque objet d'un utilisateur (voir IV.1.5).

```
rem espace reserve pour chaque objet de l'utilisateur nakech
set pagesize 200
col segment_name format a12
col owner format a12
col tablespace format a10
col segment_type format a12
break on owner skip 2 on report
compute sum of taille_en_K on owner report
select owner, segment_name, segment_type,
    tablespace_name tablespace,
    extents, round(bytes/1024) taille_en_K
from sys.dba_segments
where owner = 'NAKECH'  ;
```

```
/
clear computes
clear breaks
set pagesize 24
```

i) Tester la procédure suivante qui donne l'espace total réservé par utilisateur.

```
rem espace total reserve par utilisateur
col espace_alloue format a14
select owner, round(sum(bytes)/1024) || 'K' espace_alloue
from sys.dba_segments
group by owner
/
```

j) Tester la procédure suivante qui donne l'espace libre par tablespace.

```
rem espace libre par tablespace
col tablespace_name format a15
col espace_libre format a15
select tablespace_name,
      round(sum(bytes)/1024) || 'K' "espace libre"
from sys.dba_free_space
group by tablespace_name
/
```

6. Gestion des tables (à faire dans la session non dba**)**

a) Créez les tables suivantes (en tant que user SYSTEM) dans le tablespace DATA01 :

```
customers cust_code varchar2(3)
name varchar2(50)
region varchar2(5)
orders ord_id number(3)
ord_date date
cust_code varchar2(3)
date_of_dely date
```

(Dans la table *orders*, des lignes peuvent être insérées sans *date_of_dely* puis mises à jour plus tard.)

b) Exécutez le script *ins_cord.sql*.

c) Quel sont les fichiers et les blocs correspondant aux ordres du vendeur ayant le *cust_code*=A04 ? (La requête aboutit *a priori* à une erreur… pourquoi ?)

d) Réduisez le PCTFREE de *customers* à 5. La table comporte-t-elle des *migrated rows* ?

e) Combien d'extents sont utilisés par la table *orders* ?

f) Ajoutez un extent à la main, avec la taille par défaut, et vérifiez qu'un extent a bien été ajouté.

g) Supprimez la table *big_emp*. Créez une autre table (dans le tablespace DATA01), *orders2*, copie de la table *orders*, mais avec MINEXTENTS=8. Vérifiez que la table a bien été créée avec le nombre spécifié d'extents.

h) Tronquez la table *orders* sans libérer l'espace. Contrôlez le nombre d'extents pour vérifier qu 'aucun n'a été libéré.

i) Réduisez le MINEXTENTS de *orders2* à 4. Cette opération est-elle permise ? Y a-t-il réduction du nombre d'extents ?

j) Tronquez la table *orders2* pour libérer de l'espace. Combien reste-t-il d'extents ?

k) Exécutez le script *ins_ord2.sql*. Libérez l'espace non-utilisé de *orders2* et vérifiez le nombre d'extents. De l'espace a-t-il été libéré ? pourquoi ?

l) Combien de blocs sont utilisés par les données dans la table *orders2* ? Faites le nécessaire pour que les blocs inutilisés soient libérés. Vérifiez que l'espace a bien été libéré.

m) Ajoutez la table suivante :

```
product
prod_code number(6)
description varchar2(30)
price number(8,2)
```

dans le tablespace *data02* en utilisant un taille d'extents uniforme de 10K. Consultez la taille des extents pour cette table. Qu'observez-vous ?

7. Gestion des clusters (à faire dans la session non dba**)**

```
CREATE CLUSTER sc_person (person_id NUMBER(10))SIZE 512;

desc user_clusters
set linesize 121
SELECT cluster_name, tablespace_name, hashkeys, degree, single_table
FROM user_clusters;

CREATE INDEX idx_personnel ON CLUSTER sc_person;
SELECT index_name, index_type, tablespace_name  FROM user_indexes;

CREATE TABLE person ( person_id   NUMBER(10), first_name  VARCHAR2(25),
last_name   VARCHAR2(25) NOT NULL,active_flag VARCHAR2(1) NOT NULL)
CLUSTER sc_person (person_id);

CREATE TABLE invoice (person_id NUMBER(10), inv_id    NUMBER(10),
inv_item  VARCHAR2(25) NOT NULL,inv_date  DATE NOT NULL)
CLUSTER sc_person (person_id);

SELECT table_name, cluster_name, tablespace_name FROM user_tables;

INSERT INTO person VALUES (1, 'Dan', 'Morgan', 'A');
INSERT INTO person VALUES (9, 'Jack', 'Cline', 'A');
INSERT INTO invoice VALUES (1, 1, 'Clock', SYSDATE);
INSERT INTO invoice VALUES (9, 1, 'Clock', SYSDATE);
COMMIT;

SELECT rowid, first_name, last_name
FROM person;
SELECT rowid, inv_id, inv_item
FROM invoice;
```