TP Architecture physique Oracle

1. Consultation des vues dynamiques

Les vues dynamiques (v$...) vont jouer un rôle significatif dans la compréhension des structures manipulées par un serveur de base de données. L'ordre SQL donné ci-dessous permet de lister l'ensemble des vues dynamiques.

```
SELECT table_name,
comments
FROM dictionary
WHERE table_name like 'V$%'
ORDER BY table_name;
```

Listing 1 – vues dynamiques

Vous mettrez à profit vos connaissances PL/SQL pour construire un paquetage nommé ArchiOracle, contenant les fonctions ou procédures à même de renseigner sur les structures de la base de données et de l'instance Oracle.

1.1 Consultations d'ordre général à réaliser

1. consulter la vue portant sur l'instance (v$instance) et répondre à des interrogations telles que : quel est le nom de l'hôte sur lequel tourne l'instance, et depuis quand l'instance est démarrée,
2. consulter la vue portant sur la base de données (v$database) et répondre à des interrogations telles que : quel est le nom de la base et exploite t'elle le mode archive,
3. consulter la vue (v$option) portant sur les fonctionnalités du serveur de données et répondre à des interrogations telles que : de quelles options disposons-nous ?
4. consulter la vue (v$version) portant sur la version du SGBD Oracle sous-jacent
5. consulter l'attribut server de la vue v$session pour connaître l'architecture client-serveur retenue pour servir les connexions utilisateurs (architecture dédiée ou partagée).

1.2 structures mémoire et processus de tâches de fond

1. consulter les vues portant sur la mémoire SGA (vsga, vsgainfo ou show SGA) et répondre à des interrogations telles que : quelles sont les tailles allouées à la mémoire partagée (shared pool), au tampon de données (data buffer cache), au tampon de journalisation (buffer redo) ? Exploiter aussi une commande de type show parameter shared_pool_size
2. Pour aller plus dans le détail, vous pouvez consulter la vue v$sgastat ;
3. interroger les vues vsql, vsqlarea, v$sqltext qui sont associées à la "library cache" (shared pool) et identifiez les informations contenues dans la "library cache" et donc son rôle.

4. une requête vous est donnée pour identifier les processus d'arrière-plan interagissant avec les structures mémoire et les fichiers de données

```
select p.pid, bg.name, bg.description, p.program
from v$bgprocess bg, v$process p
where bg.paddr = p.addr order by p.pid;
```

Listing 2 – processus

5. avoir la connaissance de divers paramètres portant sur l'instance : select name, value from v$parameter. Par exemple, trouver la taille du bloc de données qui va être l'unité d'échange montée en mémoire vive.

1.3 Lien entre les structures logique et physique

1. consulter les tablespaces définis (dba_tablespaces)
2. consulter l'emplacement des fichiers de données (v$datafile) ;
3. consulter l'emplacement des fichiers journaux (v$logfile) ;
4. consulter l'emplacement des fichiers de contrôle (v$controlfile) ;
5. faire le lien entre espace de table et fichier de données : select tablespace_name, file_name from dba_data_files ;

1.4 Tampon de données "data buffer cache"

Un exemple d'exploitation de la vue v$bh (jointure avec dba_objects) est donné.

```
select file#, block#, class#, dirty, objd, object_name, owner
from v$bh, dba_objects where dirty = 'Y' and objd = object_id;
```

Listing 3 – exemples

Vous expliquerez le résultat obtenu.
Comment lister vos blocs de données qui se situent en mémoire cache ?

1.5 Mémoire "LibraryCache"

Tous les ordres SQL (mais pas que) transitent par la mémoire "LibraryCache". Vous exploiterez les vues afférentes à cette zone mémoire : v$sqlarea, v$sql et v$sql_plan pour tracer et évaluer les dernières requêtes traitées par le système. Différents exemples de requête (allant dans ce sens) sont donnés

```
select to_char(logon_time, 'DD/MM/YYYY HH24:MI:SS') , username, program, sql_text
from v$session , v$sqlarea
where v$session.sql_address = v$sqlarea.address
order by username, program;

select sql_id, disk_reads from v$sql where rownum <20 order by disk_reads desc;

select r.sql_id, disk_reads, elapsed_time, username from v$sql r, v$session s where
    s.sql_id = r.sql_id and type='USER';

select parsing_schema_name, substr(sql_text,1,20) from v$sqlarea where
    parsing_schema_name ='IMOUGENOT';
```

```
select sql_FullText,(cpu_time/100000) "Cpu Time (s)"
, (elapsed_time/1000000) "Elapsed time (s)"
, fetches,buffer_gets,disk_reads,executions
FROM v$sqlarea
WHERE
Parsing_Schema_Name ='IMOUGENOT'
AND rownum <50
order by 3 desc;
```

Listing 4 – exemples

Vous construirez différentes fonctions permettant de disposer d'informations sur les activités d'un usager en particulier et sur les requêtes les plus coûteuses. Vous pouvez aussi tester tout l'intérêt des variables liées.

1.6 Exploiter les vues user_tables et dba_tables

Les informations concernant l'organisation logique et physique des tables de votre schéma utilisateur sont disponibles depuis les vues user_tables et dba_tables. Les attributs suivants donnent une idée du nombre de blocs alloués à chaque table :

1. NUM_ROWS : nombre de lignes
2. BLOCKS : nombre de blocs utilisés
3. EMPTY_BLOCKS : nombre de blocs alloués, mais non utilisés
4. AVG_SPACE : espace libre moyen dans les blocs utilisés
5. AVG_ROW_LEN : longueur moyenne des lignes

1.6.1 Question 1

Vérifiez que que toutes vos tables sont situées dans le même tablespace et qu'il en va de même pour les autres schémas utilisateurs.

1.6.2 Question 2

Combien de blocs de données sont utilisés pour contenir les données de chacune de vos tables ? Ces informations sont évaluées ou ré-évaluées par la commande

```
analyze table nom_table compute statistics
```

Créer la table test(num char(3), commentaire char(97)), et appliquer sur num une contrainte de domaine : nombres entiers compris entre 0 et 999

1.6.3 Question 3

Est-ce que la longueur des tuples de cette table est variable ? Relever le nombre de blocs utilisés, alloués mais inutilisés, . . .après

1. la création de la table
2. l'insertion de 50 lignes
3. l'insertion de 100 lignes supplementaires
4. l'insertion de 100 lignes supplementaires

Vous pouvez vous aider d'une procédure pour insérer des tuples dans la table. Comparer à chaque étape l'espace réellement occupé par la table et l'espace théoriquement nécessaire à sa mémorisation. Comment expliquer les différences ?

1.6.4 Question 4

Supprimer un tiers des tuples de la table (par exemple, ceux dont le num est divisible par 3). Réevaluer les statistiques sur cette table à l'issue de l'opération. Supprimer l'integralité des tuples, avec avec l'ordre

```
delete from test
```

valider (commit) et regénerez les statistiques. Que constatez-vous ? Passer par l'ordre suivant

```
truncate table test
```

et ré-évaluer les statistiques. Que constatez vous ?

2. Notion de performance d'accès

Un indicateur nommé buffer_hit_ratio permet d'évaluer la performance en matière d'accès aux données. Le calcul se fait fait de la manière suivante

```
Select 1- (phy.value / ( cons.value + db.value - phy.value))
from v$sysstat phy, v$sysstat cons, v$sysstat db
where phy.name ='physical reads' and cons.name ='consistent gets' and db.name ='db block
    gets';
```

Listing 5 – processus

Vous expliquerez en quoi il consiste.

3. Surveillance des ressources mémoires allouées

Trois vues sont consultables pour vérifier en particulier que les différents fichiers de la basse de données ne sont pas saturés.

1. dba_free_space informe sur les "extents" (blocs contigûs) qui restent libres dans les tablespaces
2. dba_tablespaces donne des informations sur les tablespaces de la base de données
3. dba_data_files donne des informations sur la taille des fichiers associés aux tablespaces

Des exemples de requêtes exploitant ces vues vous sont donnés. Vous construirez une procédure basée sur un curseur explicite qui vous permettra de connaître le degré de remplissage des différents fichiers

```
select file_name, tablespace_name, blocks, bytes, maxbytes, autoextensible
from dba_data_files;

select tablespace_name, sum(bytes), count(blocks)
from dba_free_space group by tablespace_name;


select t.tablespace_name,nvl(sum(F.bytes/1024/1024),0) "free space (en Mo)"
from dba_free_space f right join dba_tablespaces t on
```

```
 f.tablespace_name =t.tablespace_name
group by t.tablespace_name
order by nvl(sum(F.bytes/1024/1024),0);
```