TP2 Architecture Oracle (3h)

1. Préalable

L'optimisation est entendue comme étant la capacité à adapter l'architecture d'un système aux différents besoins des applications clientes (in fine, l'objectif est de satisfaire les usagers). Il va donc s'agir d'un ensemble d'ajustements au niveau de l'instance et de la base de données qui va permettre d'atteindre des objectifs de **performance** (c'est à dire augmenter le débit transactionnel, et réduire le temps des réponses).

2. Serveur Oracle

Un serveur Oracle est, en grande majorité, constitué d'une instance (structures cache + processus) et d'une base de données (ensemble de fichiers).
Quelques questions d'ordre général, et qui concerne le serveur, vous sont posées.

1. consulter la vue (v$version) portant sur la version du SGBD Oracle sous-jacent

```
-- Oracle Database 12c Enterprise Edition Release 12.2.0.1.0 - 64bit Production
select banner from v$version ;
```

2. 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).

```
-- dediee (dedicated) : 1 processus serveur pour 1 processus client
select sid, server, username, machine from v$session where type='USER';
```

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 ?

```
-- cartouches OLAP, Spatial, et index bitmap par exemple
 select parameter from v$option where value = 'TRUE';
```

3. L'instance

Une vue dynamique donne également quelques renseignements à propos de l'instance :

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,

```
select instance_number, instance_name, host_name, startup_time from v$instance;
INSTANCE_NUMBER INSTANCE_NAME  HOST_NAME   STARTUP_TI
--------------- ---------------- ---------------------------------------------------------------- ----------
      1 CMASTER   prodpeda-oracle.umontpellier.fr   02/10/2021
```

3.1 Structures mémoire

3.1.1 Taille des sous-structures de la SGA

Vous consulterez les vues portant sur la structure mémoire System Global Area, ou SGA (v$sga ou show SGA, v$sgainfo). La SGA est divisée en différentes sous-structures, qui chacune possède un

rôle clé, dans l'efficacité et la sécurité des accès aux données. Ainsi la mémoire partagée (shared pool) correspond au cache des dernières requêtes (library cache) et du dictionnaire de données (dictionary cache). Le cache de données (data buffer cache) est la zone de transit et d'écriture pour ce qui concerne les blocs de données. Le tampon de journalisation (redo log buffer) conserve l'information sur les dernières transactions en cours.
Vous répondrez à des interrogations telles que :

1. quelle est la taille (en Mo) allouée à la mémoire partagée (shared pool),

```
select name, bytes/1024/1024, resizeable from v$sgainfo ;
environ 400 Mo shared pool
environ 850 Mo Data Buffer
environ 8 Mo Rego Log Buffer
environ 3 Go SGA total
```

2. quelle est la taille (en Mo) allouée au tampon de données (data buffer cache),
3. quelle est la taille (en Mo) allouée au tampon de journalisation (redo log buffer),
4. quelle est la taille totale (en Go) allouée à la SGA

```
select sum(value)/1024/1024/1024 enGo from v$sga;
-- donne 3 Go
-- voir aussi
 select name, value/1024/1024 enMo from v$sga;
NAME    ENM
-------------------- ----------
 Fixed Size      8,39035797
 Variable Size      2176,00027
 Database Buffers     880
 Redo Buffers        7,609375

-- et egalement show SGA
```

Pour aller dans plus de détail, vous consulterez éventuellement la vue v$sgastat.

3.2 Mémoire "LibraryCache"

3.2.1 Indicateur de performance concernant la mémoire partagée

La mémoire partagée contient la "LibraryCache" et le "DictionaryCache". Nous évaluons ici la performance de la "LibraryCache". La vue système v$librarycache donne des informations sur les derniers ordres (SQL, Trigger et Procédure PL/SQL, ...) exécutés. L'objectif de l'indicateur à construire est d'évaluer le nombre d'exécutions qui n'ont pas nécessité de charger l'ordre en mémoire (pins) au regard du nombre de rechargements (reloads). Vous interrogerez v$librarycache pour construire le "library cache hit ratio" en question (somme de ("pins" - reloads) / somme de "pins"). Vous vérifierez que sa valeur est très proche de 1. Qu'est ce que vous pouvez en conclure si cette valeur est proche de 1 ? Que faire si elle est très inférieure à 1 ?

```
-- indicateur library cache hit ratio
 select  sum(pins - reloads)/sum(pins) as LCH_Ratio from v$librarycache ;

  LCH_RATIO
  ----------
  ,995954833
--  ici tres proche de 1, a defaut il faut augmenter la taille de la shared pool (parametre shared_pool_size)
```

3.2.2 Les derniers ordres du cache de requêtes

Tous les ordres SQL (mais pas que) transitent par la mémoire "LibraryCache". Vous exploiterez les vues système afférentes à cette zone mémoire : v$sqlarea, v$sql, v$sqltext et v$sql_plan pour tracer et évaluer les dernières requêtes traitées par le système. Différents exemples de requête (allant dans ce sens) sont donnés. Vous construirez une première procédure nommée User_Activity permettant d'afficher des informations sur les accès aux données d'un usager dont le schéma utilisateur est passé en paramètre d'entrée. Vous construirez une seconde procédure Costly_Cursors qui permet de renvoyer les 10 requêtes (appelées cursors dans le contexte de la "LibraryCache") les plus coûteuses tous usagers confondus. Le coût est fonction du nombre de blocs accédés sur disque, du temps de calcul au niveau du processeur (cpu_time), du temps de traitement total (elapsed_time). Attention dans v$sqlarea, les

données sont cumulées au fur et mesure que le même ordre est exécuté (le nombre d'exécutions est donné pat l'attribut executions). Nous n'allons pas lister l'ensemble des attributs de ces vues systèmes, mais seulement les attributs exploités dans les exemples :

1. SQL_ID : identifiant du curseur
2. SQL_TEXT : ordre SQL / PL/SQL
3. DISK_READS : nombre de blocs en accès disque
4. PARSING_SCHEMA_NAME : schéma utilisateur
5. CPU_TIME : Temps CPU (en microsecondes) utilisé pour le curseur pour le cycle parse/execute/fetch/
6. ELAPSED_TIME : Temps écoulé (en microsecondes) utilisé pour le curseur pour le cycle parse/execute/fetch/
7. EXECUTIONS : nombre de fois où l'ordre a été lancé
8. BUFFER_GETS : nombre de blocs exploités depuis le cache de données

```
set linesize 200
col req for a80
select sql_id, substr(sql_text,1,80) as req, disk_reads from v$sqlarea where
    parsing_schema_name =user;

col parsing_schema_name for a16
select parsing_schema_name, sql_id, substr(sql_text,1,80) as req from v$sqlarea order by
    first_load_time ;

select parsing_schema_name, sql_id, substr(sql_text,1,80) as req from v$sqlarea where
    parsing_schema_name<>'SYS';


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 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 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 ='P00000009432'
AND rownum <50
order by 3 desc;
```

Listing 1 – exemples

```
-- pour User_Activity
set serveroutput on
set linesize 200

create or replace procedure User_Activity (Uname in varchar)
as
cursor c is select substr(sql_text,1,120) as req, executions from v$sqlarea where parsing_schema_name = Uname;
ex exception;
begin
if sql%rowcount is null
then raise ex ;
end if ;
for t in c
```

```
loop
dbms_output.put_line(rpad(t.req,120,' ~')||' **** '||t.executions) ;
end loop ;
exception
when ex then dbms_output.put_line('usager inconnu') ;
when others then dbms_output.put_line(SQLERRM) ;
end ;
/

exec User_Activity (User)

-- le parti pris est de diviser par le nombre d'executions pour chaque requete car les valeurs sont cumulees
-- nous evitons ainsi de voir une requete elementaire mais executee 1000 fois comme couteuse

create or replace procedure Costly_Cursors
as
cursor c is select substr(sql_text,1,120) as req, cpu_time/executions as cpu_time, buffer_gets/executions as buffer,
disk_reads/executions as disk, executions, parsing_schema_name from v$sqlarea
where parsing_schema_name not in ('SYS', 'C##_SVC_SHINKEN') and rownum <= 10 order by 2 desc ;
begin
for t in c
loop
dbms_output.put_line(rpad(t.req,120,' ~')||'  '||rpad(t.parsing_schema_name,12)||'  '||round(t.cpu_time/1000000,4)||' sec,
avg disk '||round(t.disk,1)||' avg buffer '||round(t.buffer,1)||' nbre exec '||t.executions) ;
end loop ;
exception
when others then dbms_output.put_line(SQLERRM) ;
end ;
/

exec Costly_Cursors
```

3.3 Tampon de données "data buffer cache"

3.3.1 Indicateur 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 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 2 – processus

Vous chercherez à expliquer en quoi il consiste.

```
-- indicateur buffer hit ratio doit etre le plus proche possible de 1
1-(PHY.VALUE/(CONS.VALUE+DB.VALUE-PHY.VALUE))
----------------------------------------------
   ,913720646

-- si tres proche de 1 : indique que tres peu de blocs ont ete lus sur le disque
-- acces disque : 10 ms, acces tampon donnees : de l'ordre de la nano-seconde
- db block gets  : nombre de blocs lus
- consistent gets : nombre de blocs recherches dans les Rollback Segments
- physical reads : nombre de blocs lus sur disque.
-- il est a noter que le ratio est tres correct sans etre excellent, c'est lie a l'usage
particulier de la base avec chaque etudiant travaillant dans son espace propre.
rien n'est quasi partage, ce qui n'est pas habituel pour une BD
```

3.3.2 En savoir plus sur les blocs du 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.
Le numéro de classe est un indicateur du type de bloc. Quelques valeurs possibles et leurs significations

sont listées ici : (1 = 'data block'), (2 = 'sort block'), (3, 'save undo block'), (4, 'segment header'), (5, 'save undo header'), (6, 'free list'), (7, 'extent map'), (18, 'undo block') ....

```
-- il s'agit de renvoyer le numero du bloc, le numero du fichier qui contient ce bloc,
le numero du type (class) du bloc, le fait que ce bloc soit en cours de mise \'a jour ou non,
le nom de l'objet associe et son proprietaire quand ce bloc est en cours de modification (dirty  = 'Y')
```

Comment lister tous vos blocs de données qui se situent en mémoire cache ? Comment connaître le nom des objets associés à ces blocs ? Comment identifier l'usager qui est le plus gros consommateur du cache de donnees ?

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

select owner
from v$bh, dba_objects where objd = object_id group by owner having count(block#)
 = (select max(count(block#)) from v$bh, dba_objects where objd = object_id group by owner);

select owner
from v$bh, dba_objects where objd = object_id group by owner
having count(block#) >=ALL (select count(block#) from v$bh, dba_objects where objd = object_id group by owner);


select distinct decode(class#,
     1, 'data block'           ,
     2, 'sort block'           ,
     3, 'save undo block'      ,
     4, 'segment header'       ,
     5, 'save undo header'     ,
     6, 'free list'            ,
     7, 'extent map'           ,
     8, '1st level bitmap block',
     9, '2nd level bitmap block',
    10, '3rd level bitmap block',
    11, 'bitmap block'         ,
    12, 'bitmap index block'   ,
    13, 'file header block'    ,
    14, 'unused'               ,
    15, 'system undo header'   ,
    16, 'system undo block'    ,
    17, 'undo header'          ,
    18, 'undo block')
from v$bh ;
```

3.4 Processus

Une requête vous est donnée pour identifier les processus d'arrière-plan interagissant avec les structures mémoire et les fichiers de données. Cette requête exploite deux vues systèmes associées aux processus Oracle. La vue v$bgprocess (bg pour backgroung) contient les informations requises pour caractériser les processus d'arrière-plan. Vous chercherez à retrouver les processus d'arrière-plan décrits en cours. Que retrouve t'on comme processus dans v$process qui ne sont pas listés dans v$bgprocess ?

```
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 4 – processus

```
Le DBWR (db writer), le LGWR (log writer) ou encore le PMON (Process monitor) et le SMON (System Monitor) sont retrouves

Dans v$process, nous allons aussi retrouver les processus client et les processus serveur
```

3.5 Lien entre les structures logiques et physiques

Vous traiterez les questions suivantes :

1. consulter les tablespaces définis (dba_tablespaces)

```
select tablespace_name, block_size, initial_extent from dba_tablespaces;

TABLESPACE_NAME         BLOCK_SIZE INITIAL_EXTENT
------------------------------ ---------- --------------
SYSTEM      8192    65536
SYSAUX      8192    65536
UNDOTBS1      8192    65536
TEMP      8192  1048576
USERS      8192    65536
DATA_ETUD      8192    65536
```

2. consulter l'emplacement des fichiers de données (v$datafile) ;

```
v$datafile n'est pas le plus informatif mais peut etre exploite en jointure avec dba_data_files par ex.
select file#, name from v$dbfile ;
SQL> select file_name, file_id, blocks, maxblocks, bytes/1024/1024/1024, maxbytes/1024/1024/1024 from dba_data_files;

FILE_NAME     FILE_ID BLOCKS MAXBLOCKS BYTES/1024/1024/1024 MAXBYTES/1024/1024/1024
------------------------------------------------------------------------------- ---------- ---------- ---------- -------------------- ------
/data/bases/prodpeda/CMASTER/oracle/CMASTER/PMASTER/undotbs01.dbf  10 240960   4194302     1,83837891     31,9999847
/data/bases/prodpeda/CMASTER/oracle/CMASTER/PMASTER/sysaux01.dbf   9    1756160   4194302     13,3984375     31,9999847
/data/bases/prodpeda/CMASTER/oracle/CMASTER/PMASTER/users01.dbf   11  10880   4194302     ,083007813     31,9999847
/data/bases/prodpeda/CMASTER/oracle/CMASTER/PMASTER/system01.dbf   8  96000   4194302     ,732421875     31,9999847
/data/bases/prodpeda/CMASTER/oracle/CMASTER/PMASTER/DATA_ETUD.DBF  12    1106560   1310720     8,44238281     10

-- on notera que le fichier data_etud occupe 8,44 Go sur 10 Go max.
-- Il va falloir commencer a supprimer des tables et autres objets des schemas
```

3. consulter l'emplacement des fichiers journaux (v$logfile) ;

```
select group#, member from v$logfile;

1 membre par groupe - multiplexage possible
-- ecriture sequentielle d'un fichier de groupe a l'autre
-- puis recyclage du fichier si entierement rempli
```

4. consulter l'emplacement des fichiers de contrôle (v$controlfile) ;

```
select name , block_size, file_size_blks from v$controlfile ;

-- le bloc est a 16 Ko et le fichier est multiplexe 3 fois
```

5. faire le lien entre espace de table et fichier de données : select tablespace_name, file_name from dba_data_files. Combien de fichiers sont asservis à chaque tablespace ? ;

```
col tablespace_name for a10
col file_name for a80
select tablespace_name, file_name from dba_data_files ;

TABLESPACE FILE_NAME
---------- --------------------------------------------------------------------------------
UNDOTBS1   /data/bases/prodpeda/CMASTER/oracle/CMASTER/PMASTER/undotbs01.dbf
SYSAUX    /data/bases/prodpeda/CMASTER/oracle/CMASTER/PMASTER/sysaux01.dbf
USERS    /data/bases/prodpeda/CMASTER/oracle/CMASTER/PMASTER/users01.dbf
SYSTEM    /data/bases/prodpeda/CMASTER/oracle/CMASTER/PMASTER/system01.dbf
DATA_ETUD  /data/bases/prodpeda/CMASTER/oracle/CMASTER/PMASTER/DATA_ETUD.DBF

-- 1 seul fichier par tablespace
il est a noter que le tablespace temp est associe \'a un fichier qui est liste
dans v$tempfile ou dba_temp_files. Il n'est pas vu comme un fichier de donnees,
il sert surtout en cas de pb pour restaurer la bd dans un etat coherent
```