TP Optimisation de requêtes

1. Préambule

Le schéma comprend les tables Commune, Departement et Region. Le schéma relationnel est indiqué de manière succincte.

- COMMUNE (**CODEINSEE**, CODEPOSTAL, NUMDEP, NOMCOMMAJ, NOMCOMMIN, LATITUDE, LONGITUDE)
- DEPARTEMENT(**NUMDEP**, CHEFLIEU, NUMREG, NOMDEPMAJ, NOMDEPMIN)
- REGION(**NUMREG**, CHEFLIEU, NOMREGMAJ)

1.1 Disposer des trois tables et d'index uniques associés aux trois clés primaires

Vous disposez déjà des tables COMMUNE et DEPARTEMENT. Vous définirez la table REGION à partir de l'instruction :

```
create table REGION as select * from P00000009432.REGION ;
-- ne pas oublier de rajouter la contrainte de cle primaire et donc d'index unique
alter table REGION add constraint REGIONPK primary key(numreg) ;
```

Vérifiez que tous les index uniques, associés aux contraintes de clé primaire COMMUNEPK, DEPARTEMENTPK et REGIONPK, sont bien définis (consulter les vues USER_INDEXES et USER_CONSTRAINTS)

1.2 Exercices sur machine

Testez tous les plans d'exécution à partir de l'interpréteur sqlplus d'Oracle. Utilisez également les directives pour guider/forcer le plan d'exécution et ainsi évaluer le coût de différentes requêtes conduisant aux mêmes résultats. Des exemples d'utilisation de directives sont données dans le cours. Pour l'ensemble des exercices, exploitez les commandes :

1. set autotrace on (résultats, plan d'exécution et statistiques) ou set autotrace traceonly (seulement plan d'exécution)
2. explain plan for select ... puis
   set linesize 300
   set pagesize 100
   select * from table(dbms_xplan.display()) ;
3. SELECT /*+ GATHER_PLAN_STATISTICS */ ...
   puis
   SELECT * FROM
   TABLE(DBMS_XPLAN.display_cursor(format=>'ALLSTATS LAST +cost +outline')) ;

2. Exercice 1

Huit requêtes SQL sont données, et vous en obtiendrez les plans d'exécution.

```
-- query 1
select nomcommaj, codeinsee from commune, departement
where codeinsee = cheflieu
minus
select nomcommaj, codeinsee from commune, region
where codeinsee = cheflieu;

-- query 2
select nomcommaj, codeinsee from commune where
codeinsee in (select cheflieu from departement)
and codeinsee not in (select cheflieu from region);

-- query 3
select nomcommaj, codeinsee from commune where
exists (select null from departement where codeinsee=cheflieu)
and not exists (select null from region where codeinsee=cheflieu);

-- query 4
select nomcommaj, codeinsee from commune, departement
where codeinsee = cheflieu
and codeinsee not in (select cheflieu from region);

--query 5
select nomcommaj, codeinsee from commune left join departement on codeinsee = cheflieu
where cheflieu is not null
and codeinsee not in (select cheflieu from region);

--query 6
select nomcommaj, codeinsee from commune join departement on codeinsee = cheflieu
minus
select nomcommaj, codeinsee from commune join region on codeinsee = cheflieu;

--query 7
select nomcommaj, codeinsee from commune left join departement on codeinsee = cheflieu
where decode(cheflieu,null,'non','oui') = 'oui'
and codeinsee not in (select cheflieu from region);

--query 8
select nomcommaj, codeinsee from commune, (select cheflieu from departement) d
where codeinsee = d.cheflieu
minus
select nomcommaj, codeinsee from commune, (select cheflieu from region) r
where codeinsee = r.cheflieu;
```

1. Est ce que ces requêtes vous semblent équivalentes (c.a.d. donnent le même résultat à partir des mêmes données) ? Donnez la sémantique naturelle associée à chacune de ces requêtes. Vous pouvez vous aider avec la construction d'un arbre algébrique.
2. Est ce que l'optimiseur fait des choix différents pour chaque requête concernant le plan d'exécution ? Comment le savoir ?
3. Renvoyez le plan d'exécution qui vous semble le plus performant et commentez le. À cet effet, décrivez les opérations dans l'ordre d'enchaînement de ces opérations. Quelle est la table exploitée en premier lieu. Citez un opérateur physique mobilisé et indiquer son rôle. Est ce que les estimations en terme de tuples résultants sont correctes ?
4. Que pourriez vous ajouter au schéma pour améliorer les performances ? Tester vos ajouts en matière d'index.
5. Que pourraient apporter les directives exploitées ci-dessous à la requête ?

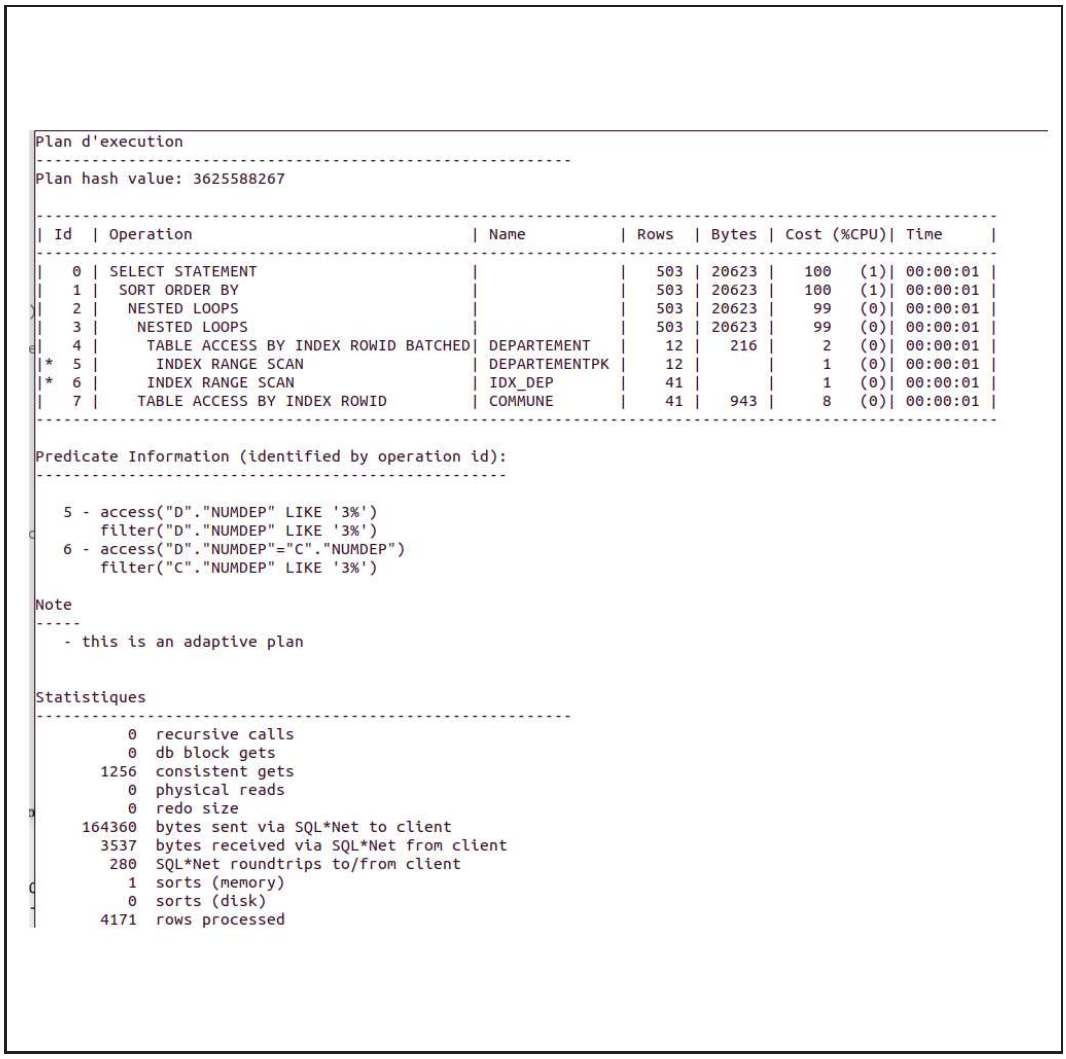
```
set pagesize 100
set linesize 200
select nomcommaj, codeinsee from commune where
exists (select /*+ hash_sj */ null from departement where codeinsee=cheflieu)
and not exists (select /*+ nl_aj */ null from region where codeinsee=cheflieu);
```

3. Exercice 2 (vu en TD)

Un plan d'exécution est donné. Il a été obtenu après définition d'un index non unique sur l'attribut numdep de la table Commune.

```
create index idx_dep on commune (numdep);
```

Les attributs projetés sont codeinsee, nomcommaj, numdep (de la table Departement), nomdepmaj, numreg.



```
Plan d'execution
----------------------------------------------------------
Plan hash value: 3625588267

-------------------------------------------------------------------------------------------------
| Id  | Operation                              | Name          | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                       |               |   503 | 20623 |   100   (1)| 00:00:01 |
|   1 |  SORT ORDER BY                         |               |   503 | 20623 |   100   (1)| 00:00:01 |
|   2 |   NESTED LOOPS                         |               |   503 | 20623 |    99   (0)| 00:00:01 |
|   3 |    NESTED LOOPS                        |               |   503 | 20623 |    99   (0)| 00:00:01 |
|   4 |     TABLE ACCESS BY INDEX ROWID BATCHED| DEPARTEMENT   |    12 |   216 |     2   (0)| 00:00:01 |
|*  5 |      INDEX RANGE SCAN                  | DEPARTEMENTPK |    12 |       |     1   (0)| 00:00:01 |
|*  6 |     INDEX RANGE SCAN                   | IDX_DEP       |    41 |       |     1   (0)| 00:00:01 |
|   7 |    TABLE ACCESS BY INDEX ROWID         | COMMUNE       |    41 |   943 |     8   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

   5 - access("D"."NUMDEP" LIKE '3%')
       filter("D"."NUMDEP" LIKE '3%')
   6 - access("D"."NUMDEP"="C"."NUMDEP")
       filter("C"."NUMDEP" LIKE '3%')

Note
-----
   - this is an adaptive plan


Statistiques
----------------------------------------------------------
          0  recursive calls
          0  db block gets
       1256  consistent gets
          0  physical reads
          0  redo size
     164360  bytes sent via SQL*Net to client
       3537  bytes received via SQL*Net from client
        280  SQL*Net roundtrips to/from client
          1  sorts (memory)
          0  sorts (disk)
       4171  rows processed
```

FIGURE 1 – Plan d'exécution

1. Quels sont les index mobilisés ?
2. Quel est l'opérateur physique de jointure choisi par l'optimiseur ? Discutez le choix de cet opérateur. Est ce que les estimations sur le nombre de tuples retournés sont correctes ? Comment faire pour vérifier que certaines de ces estimations sont incorrectes ?
3. Donner la sémantique naturelle de la requête SQL qui a donné lieu à ce plan
4. Proposer la requête SQL qui vous semble à l'origine de ce plan.
5. Proposer une écriture de la requête intégrant des directives permettant de changer d'opérateur physique de jointure, et de table guidant la jointure.

4. Exercice 3

Une requête de consultation est donnée sous sa forme algébrique.

$\Pi_{nomCommaj,nomDepMaj,nomRegMaj,d.chefLieu,r.chefLieu}$ (Commune c $\bowtie$ Departement d $\bowtie$ Region r)

1. donner la sémantique naturelle de cette requête
2. donner une écriture SQL de cette requête
3. évaluer et expliquer le plan d'éxecution proposé par l'optimiseur (tracer les opérations réalisées, indiquer si les index sont mobilisés et lesquels, indiquer si les tables sont parcourues dans leur globalité)
4. exploiter autotrace ou gather_plan_statistics pour exécuter la requête et non pas seulement estimer son exécution. Donnez quelques explications des statistiques obtenues : nombre de blocs parcourus depuis le cache de données, nombre de blocs parcourus depuis le disque, temps de calcul, temps total pour la restitution des résultats
5. expliquer pourquoi la requête de consultation suivante, également donnée sous sa forme algébrique, pourrait donner lieu à un plan d'exécution moins coûteux
   $\Pi_{nomCommaj,d.numDep,r.numReg}$ (Commune c $\bowtie$ Departement d $\bowtie$ Region r)