RENDU TD2

OPTIMISATION

Entrepôt de données et Big Data

Par Chamy KACI et Asma GUERMOUCHE

Master 1 IASD

UNIVERSITÉ
DE MONTPELLIER

I.Partie 1

1.1.Cout de plans d'execution logiques

Question 1

la requête permet d'obtenir la liste des noms des étudiants inscrits au module ayant pour intitulé "EDBD" ;

Question 2

1. Jointure : On lit 70*4200 lignes, on produit 4200 lignes
   Jointure : On lit 200*4200 lignes On produit 4200 lignes
   Selection : one lit 4200 lignes, on produit 4200 * 10 %
   Coût(E/S) : 294000E+4200S+840000E+4200S+4200S+420S =1147020 lignes E/S

2. Selection : On lit 70 lignes, on produit une ligne
   Jointure : on lit 4200 * 1 lignes on produit 420 lignes
   Jointure : on lit 420 * 200 lignes on produit 420 lignes
   Coût(E/S) : 70E+1S+4200E+420S+84000E+420S=89111 lignes E/S

3. Jointure : On lit 70*200=14000 lignes, on produit 70*200=14000 lignes
   Jointure : On lit 14000*4200 lignes, on produit 4200 lignes
   Selection : On lit 4200 lignes on produit 420 lignes
   Coût(E/S) :14000E+14000S+(14000*4200)E+4200S+4200E+420S= 58836820 lignes E/S

Question 3

Le plan 2 est le plan d'exécution optimal parmis les plans proposés, la selection est plus sélective par rapport à la jointure.

1.2.Définition de plans d'éxecutions logiques

Les plans d'exécution sont placés en annexe, parce qu'ils prennent une place consé- quente dans la mise en page.

1. La requête 1 permet d'afficher le responsable du module dont l'intitulé commence par HMIN et dont l'étudiant du nom de DUPONT y est inscrit i

2. Les 3 plans d'execution logiques : voir Plan d'exécution logique n°1 (a)1,(b)2 et (c)3 .

3. le plan optimal parmi les 3 : Les stratégies sont ici classées de la plus coûteuse à la plus optimale. Cepen- dant la stratégie 1(c)3 est peu susceptible (ou pas du tout) d'être utilisée par un SGBD puisque leur approche heuristique tend plutôt à éviter les approches parallé- lisées. Dans ce cas la stratégie 1(b)2 est probablement celle qui sera choisie à la place.

1.3.Réécriture de plans d'exécution logiques

1. Oui, elles sont équivalentes. On peut descendre ou faire monter les conditions sur les sélections et on retrouve les mêmes expressions.

2. La seconde expression est meilleure étant donnée que la jointure se fait sur des tables réduites.

1.4.Tous les plans d'éxecution logiques

1. Les 2 plans d'éxecution logiques : voir annex, Plan d'exécution logique n°2 (a) 4 et (b) 5.

2. Le plus optimal : Plan d'exécution logique n°2 (b) 5. justification :
   Le plan(c) est le plus optimisés et meilleur puisque l'on fait descendre la sélection sur le genre plus bas, mais nous forçons donc une stratégie parallélisée.

   1) La sélection sur le réalisateur dès la première étape nous permet de nous restreindre sur leur oeuvre. Il est assez logique de penser que le nombre de films qu'ils ont réalisé est très petit par rapport au nombre total de films, de même pour les acteurs qui ont joué dans leurs films par rapport au nombre total d'acteurs.

   2) La sélection description="Comédie" permet de restreindre encore plus le domaine, puisque nous sommes maintenant restreints aux films des frères Cohen qui sont des comédies.

   3) La sélection nationalité="France" peut sembler tardive, mais elle ne l'est pas. En effet le nombre d'acteurs qui ont joué dans les comédies des frères Cohen est probablement beaucoup plus petit que le nombre d'acteurs français donc faire la sélection de la nationalité plus tard est plus avantageux.

II.Partie 2

1.Les plans d'exécution sous ORACLE

1.1 Selection

Question 1 :

Il y a 2 scripts SQL, l'un nommé script_table.sql, chargé de créer les trois tables suivantes : ville, region, et departement.
Le second, nommé script_remplissage.sql, insère des tuples dans les tables précédemment créées.

Question 2 :

Requete :
select nom
from ville
where insee='1293' ;
comme le type de insee est un varchar alors quand on exécute la requête précédente avec " "
where insee=1293 " ça nous renvoie une erreure .
La sortie de console est disponible en Figure 6 et 7. Pour une seule table, et sans index sur la colonne insee, la seule méthode d'accès logique est TABLE ACCESS FULL.

Question 3 :

ALTER TABLE ville ADD CONSTRAINT PK_ville PRIMARY KEY (insee);

Question 4 :

Les différences observées par rapport à la question 2:Quand on compare le résultat de cette requête au résultat de la requête de la 2ème question on remarque que : la table d'exécution de la question 2 contient la ligne 'table access full' en revanche, celle de la question 4 contient 'table access by index rowid' ainsi que « INDEX UNIQUE SCAN .
Explication :
Dans la 2ème question, comme insee n'est pas une clé primaire alors on fait un parcours sequentiel dans la table, alors que dans le cas ou c'est une clée primaire (question 4) on fait un accès direct via l'adresse et l'index(la clé primaire), qui ce dernier est plus interéssant en terme de temps . Voir figure 8.

1.2 Jointure

Question 5 :

Requête :
Select departement.nom
from departement, ville
where ville.insee='1293' and ville.dep='departement.id '.
Voir figure 9 .

Question 6 :

Requête :
Select departement.nom
from departement, ville
where ville.dep='departement.id'

Voir figure 10.

comme on peut le remarquer dans la la requête de la question 6: On fait un parcours séquentiel aux deux tables: ville et departement (table acces full) et une jointure par hachage (hash join) .On a comme résultat 29811 rows et un nombre d'octets enorme transférés par Oracle comparé à la question 5. dans la question 5: on fait un accès direct par adresse et par index aux deux tables et on fait une jointure par boucle imbriqué (nested loops) . un seul row est selectionnée (correspondant à la ville ayant comme id 1293)
Concernant les statistiques: On peut remarquer qu'ils sont elevés dans la question 6 comparé la question 5.

1.3 Modification du comportement de l'optimiseur

Question 7:

requête:
/*+ use_nl(ville departement)*/ Select departement.nom
from departement, ville
where ville.dep='departement.id'
Dans la plan d'exécution on a une ligne NESTED LOOPS
L'optimiseur fait un accès sequentiel eux deux tables. On remarque que le nombre de lignes obtenues par l'accès à la table ville est beaucoup moins que dans le cas ou c'est une jointure par hachage (287 lignes au lieu de 29811)
Concernant les statistiques: On a quasiment les mêmes statistiques la question 6, sauf pour les appel récursifs (159 recursive calls au lieux de 72), le nombre de d'accès aux données en ram (22615 consistent gets au lieu de 2783) et le nombre d'operations de tri en mémoire (14 sorts au lieu de 12). Voir figure 11.

1.4 Utilisation d'index

Question 8:

requête:
Create index idx_dep_ville on ville(dep)
Quand on ré-execute la requete de la question 5, On obtient la même table d'exécution obtenu avant l'ajout de l'index, on revanche on remarque une différence dans les statistiques au niveau des deux élèments 'récursive calls' et 'consistents gets' qui sont plus elevés par rapport à la question 5. Voir figure 12 et 13.

Question 9:

requête:
Select ville.nom, departement.nom, region.nom
from ville, region, departement

where ville.dep=departement.id and departement.reg=region.id ;
le plan d'exécution contient 3 accès sequentiels (un pour chaque table) ainsi que 2 jointure de hachage (jointure entre ville et departement et entre departement et region), en terme de lignes selectionnées et de coup la première jointure donne plus de lignes que la deuxième et son coup est beaucoup plus elevé .
en terme de statistiques:
on a 58 appels récursifs, 2700 accès à une données consistante en ram et un nombre enorme de bytes transférés par oracle.
Voir figure 14 .

Question 10:

requête:
Create index idx_reg_dep on departement(dep)
Quand On ré-execute la requete de la question 9 on obtient le plan d'exécution on remarque que l'optimiseur a opté pour une jointure par tri-fusion (Merge join and sort join) ainsi qu'un accès sequentiel à la table departement dans l'ordre de l'index .
Concernant les statistqiues on remarque qu'il y a beaucoup moins d'appels récursifs mais la taille du fichier log produit est plus grande (redo size).
Voir figure 16.

Question 11:

requête:
Select ville.nom, departement.nom, region.nom
from ville, region, departement
where ville.dep=departement.id and departement.reg=region.id and region.id=91 ;
L'optimiseur a opté pour un accès par adresse à la table ville et region ainsi que departement sauf que pour cette dernière on remarque que c'est " access by index rowid batched ", c'est à dire l'optimiseur récupère les identifiants de ligne de l'index dans des blocs puis accède aux lignes dans l'ordre des blocs. On constate aussi qu'il a fait 3 jointure par boucles imbriqués
Concernant les statistiques: Aucun appel récursif.
Voir figure 15.

Question 12:

requête:
Select ville.nom from ville, departement
where ville.dep=departement.id and departement.id LIKE '7%' ;
On remarque dans le plan d'exécution l'absence de loop, et ça est du au fait qu'on a ajouté l'index . voir figure 17

1.5 Les statistiques des tables

Question 13 :

Oui, Les statistiques correspondent aux données des tables. Voir figure 18

Question 14 :

Avant l'exécution de la commande il n'y a pas de statistiques pour les tables crées récemment. Après l'exécution il y a des statistiques sur les minimas et maximas des colonnes, la densité, le nombre de buckets (probablement en rapport avec les index), et d'autres statistiques. Donc oui, pour certaines requêtes dèja effectuées, l'optimiseur a opté pour d'autres operateurs . Voir figure 19, 20 et 21 .

0.1 Annex

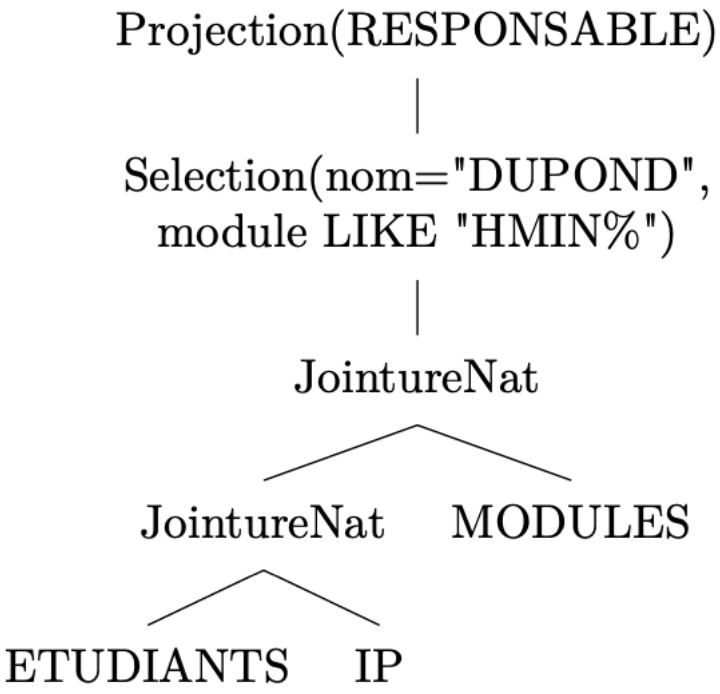


Figure 1: Plan d'exécution logique n°1 (a)

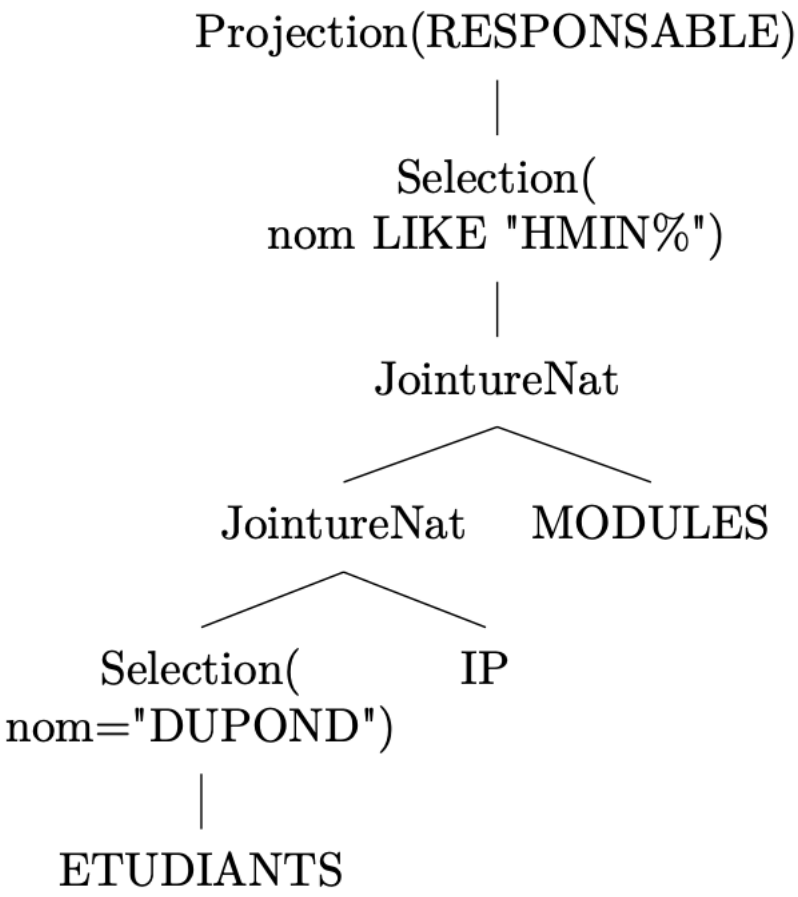


Figure 2: Plan d'exécution logique n°1 (b)

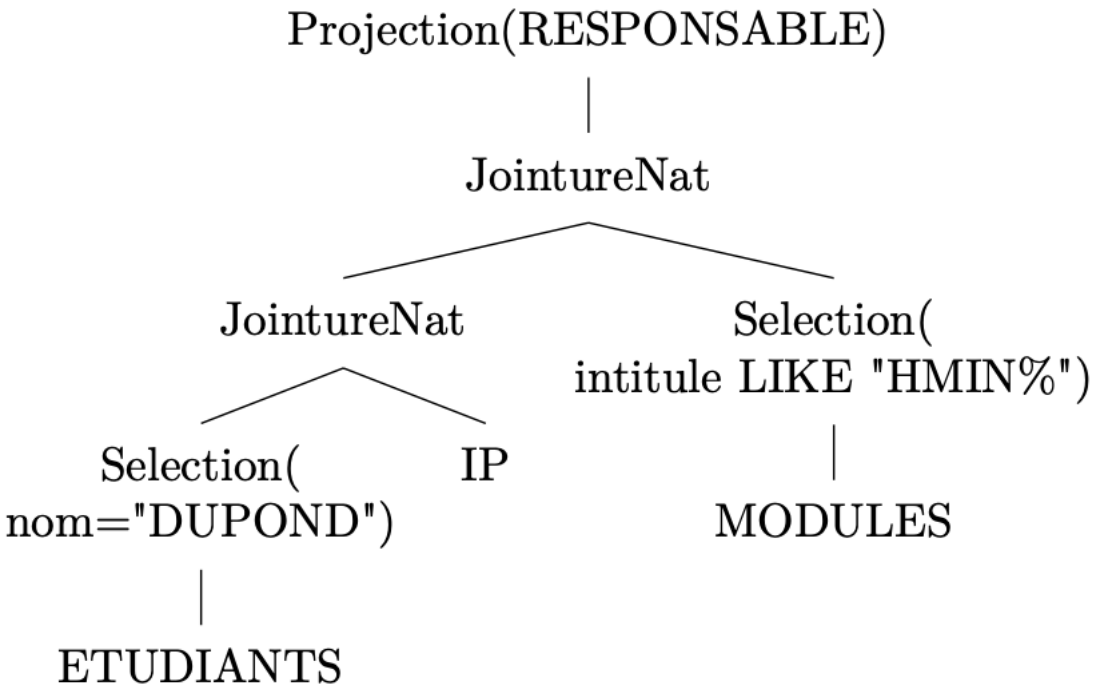


Figure 3: Plan d'exécution logique n°1 (c)

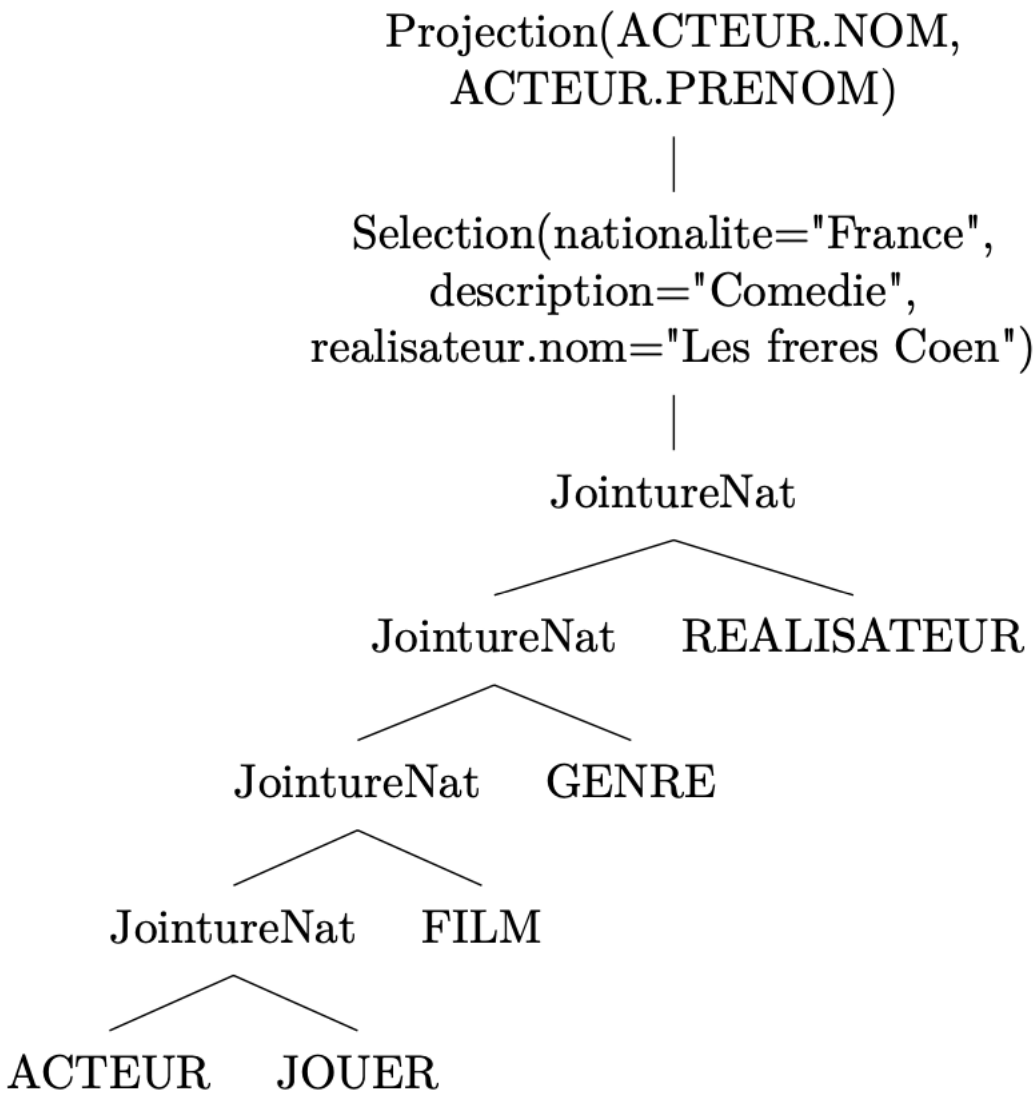


Figure 4: Plan d'exécution logique n°2 (a)

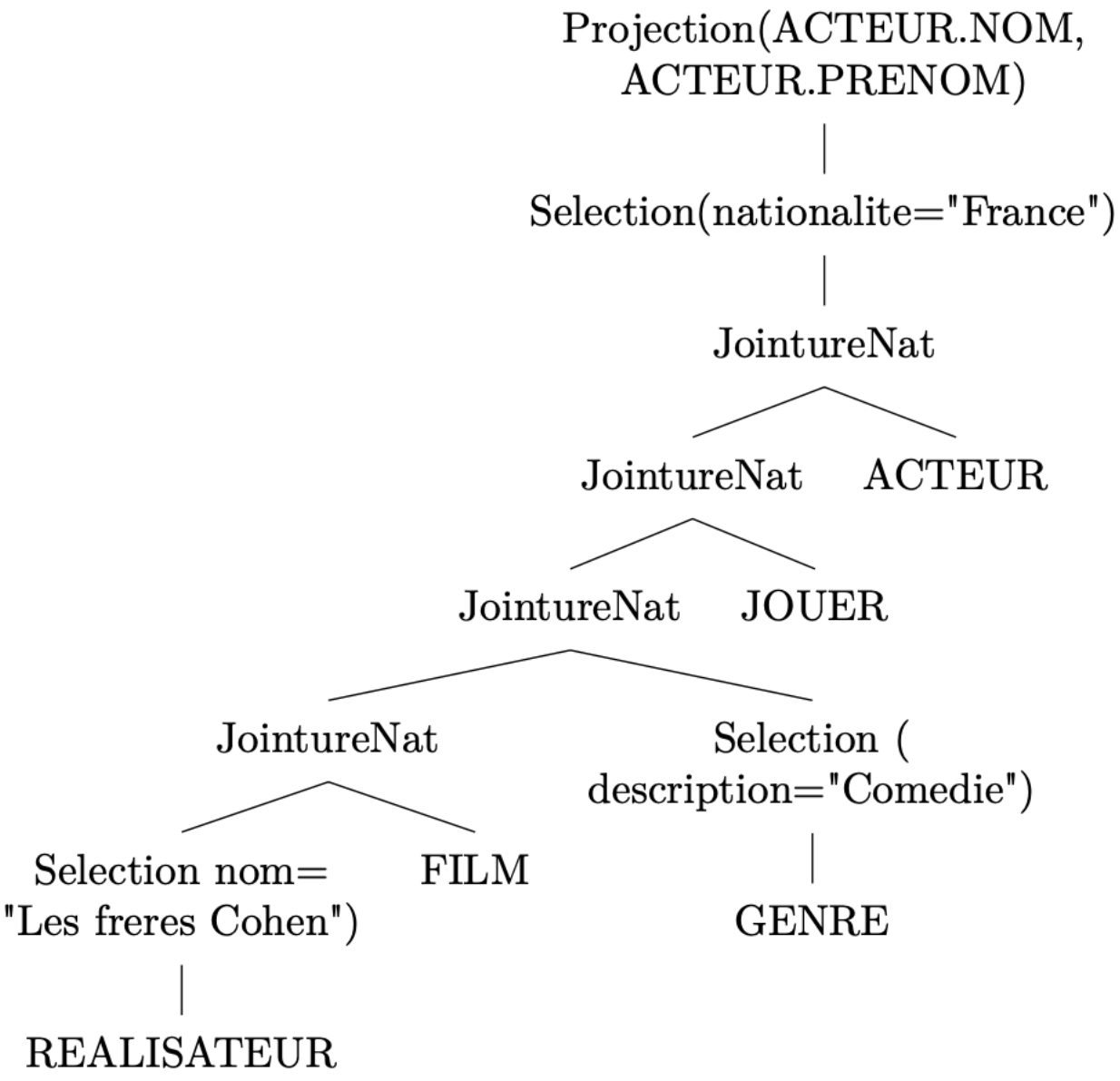


Figure 5: Plan d'exécution logique n°2 (b)

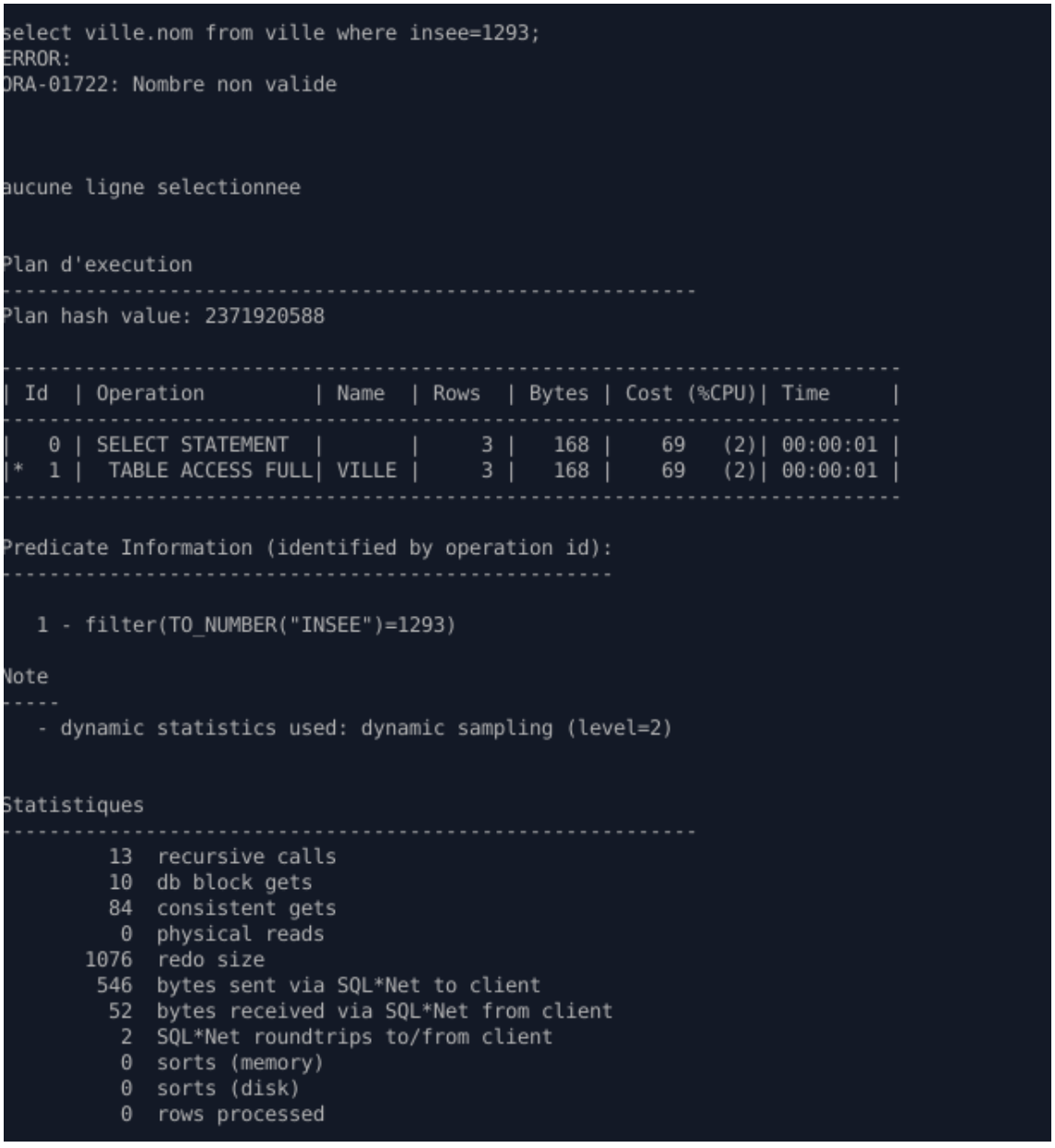
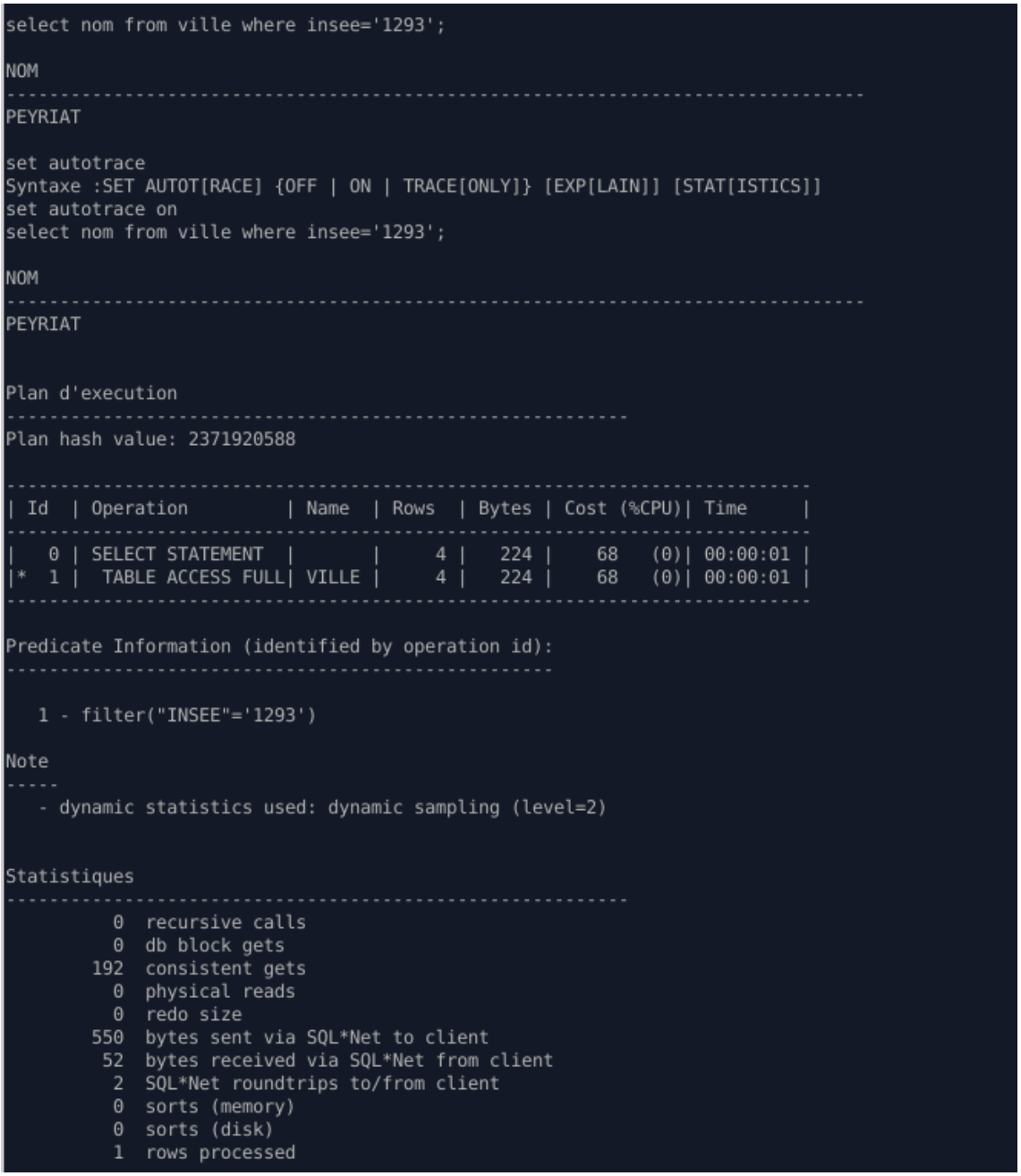


Figure 6: Sortie de console Question 2



```
select nom from ville where insee='1293';

NOM
--------------------------------------------------------------------------------
PEYRIAT

set autotrace
Syntaxe :SET AUTOT[RACE] {OFF | ON | TRACE[ONLY]} [EXP[LAIN]] [STAT[ISTICS]]
set autotrace on
select nom from ville where insee='1293';

NOM
--------------------------------------------------------------------------------
PEYRIAT


Plan d'execution
----------------------------------------------------------
Plan hash value: 2371920588

-------------------------------------------------------------------------------
| Id  | Operation         | Name  | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |       |     4 |   224 |    68   (0)| 00:00:01 |
|*  1 |  TABLE ACCESS FULL| VILLE |     4 |   224 |    68   (0)| 00:00:01 |
-------------------------------------------------------------------------------

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

   1 - filter("INSEE"='1293')

Note
-----
   - dynamic statistics used: dynamic sampling (level=2)


Statistiques
----------------------------------------------------------
          0  recursive calls
          0  db block gets
        192  consistent gets
          0  physical reads
          0  redo size
        550  bytes sent via SQL*Net to client
         52  bytes received via SQL*Net from client
          2  SQL*Net roundtrips to/from client
          0  sorts (memory)
          0  sorts (disk)
          1  rows processed
```

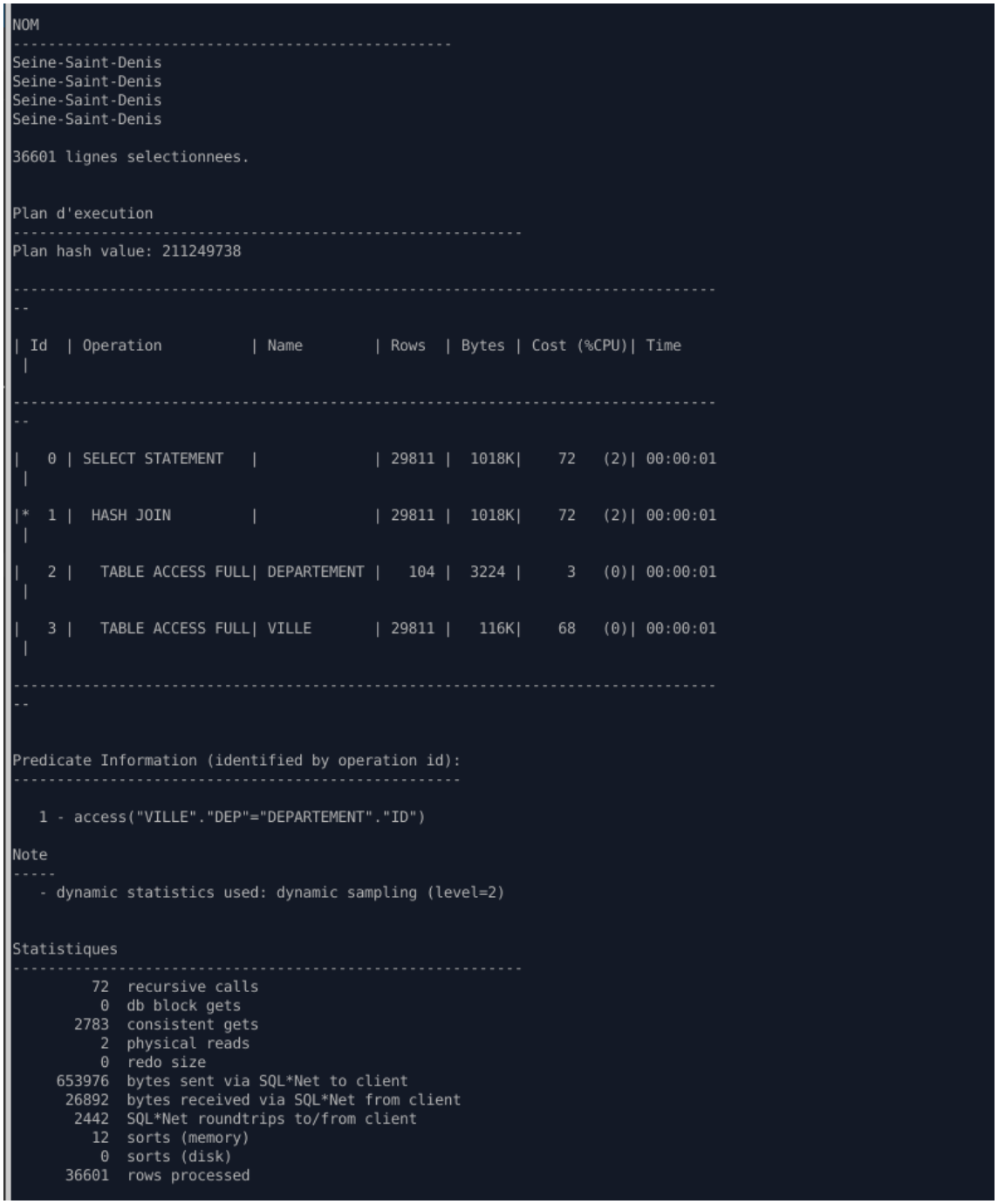
Figure 7: Sortie de console Question 2

```
Validation effectuee.

select ville.nom from ville where insee='1293';

NOM
--------------------------------------------------------------------------------
PEYRIAT

set autotrace on
select ville.nom from ville where insee='1293';

NOM
--------------------------------------------------------------------------------
PEYRIAT


Plan d'execution
----------------------------------------------------------
Plan hash value: 2371920588

--------------------------------------------------------------------------------
| Id  | Operation         | Name  | Rows  | Bytes | Cost (%CPU)| Time     |
--------------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |       |     3 |   168 |    68   (0)| 00:00:01 |
|*  1 |  TABLE ACCESS FULL| VILLE |     3 |   168 |    68   (0)| 00:00:01 |
--------------------------------------------------------------------------------

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

   1 - filter("INSEE"='1293')

Note
-----
   - dynamic statistics used: dynamic sampling (level=2)


Statistiques
----------------------------------------------------------
          0  recursive calls
          0  db block gets
        192  consistent gets
          0  physical reads
          0  redo size
        550  bytes sent via SQL*Net to client
         52  bytes received via SQL*Net from client
          2  SQL*Net roundtrips to/from client
          0  sorts (memory)
          0  sorts (disk)
          1  rows processed
```

Figure 8: Sortie de console Question 4

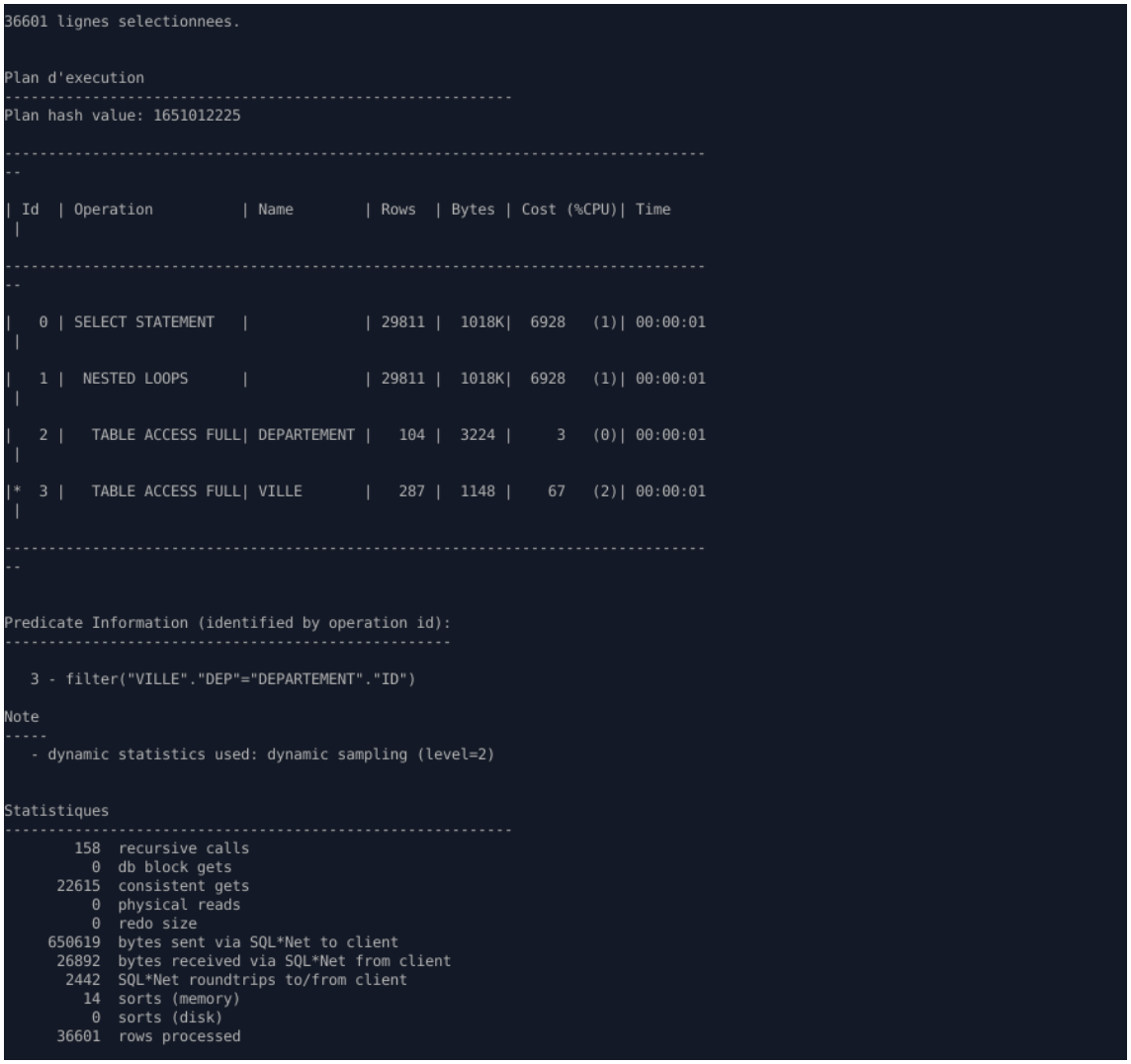
```
select departement.nom from departement, ville where ville.insee='1293' and ville.dep = departement.id ;

NOM
--------------------------------------------------
Ain


Plan d'execution
----------------------------------------------------------
Plan hash value: 2507270265

-------------------------------------------------------------------------------
-------------

| Id  | Operation                    | Name          | Rows  | Bytes | Cost (%CP
U)| Time     |

-------------------------------------------------------------------------------
-------------

|   0 | SELECT STATEMENT             |               |     1 |    39 |     3   (
0)| 00:00:01 |

|   1 |  NESTED LOOPS                |               |     1 |    39 |     3   (
0)| 00:00:01 |

|   2 |   TABLE ACCESS BY INDEX ROWID| VILLE         |     1 |     8 |     2   (
0)| 00:00:01 |

|*  3 |    INDEX UNIQUE SCAN         | SYS_C00362565 |     1 |       |     1   (
0)| 00:00:01 |

|   4 |   TABLE ACCESS BY INDEX ROWID| DEPARTEMENT   |     1 |    31 |     1   (
0)| 00:00:01 |

|*  5 |    INDEX UNIQUE SCAN         | SYS_C00362562 |     1 |       |     0   (
0)| 00:00:01 |

-------------------------------------------------------------------------------
-------------


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

   3 - access("VILLE"."INSEE"='1293')
   5 - access("VILLE"."DEP"="DEPARTEMENT"."ID")


Statistiques
----------------------------------------------------------
         0  recursive calls
         0  db block gets
         5  consistent gets
         0  physical reads
         0  redo size
       546  bytes sent via SQL*Net to client
        52  bytes received via SQL*Net from client
         2  SQL*Net roundtrips to/from client
         0  sorts (memory)
         0  sorts (disk)
         1  rows processed
```

Figure 9: Sortie de console Question 5



```
NOM
------------------------------------------------
Seine-Saint-Denis
Seine-Saint-Denis
Seine-Saint-Denis
Seine-Saint-Denis

36601 lignes selectionnees.


Plan d'execution
----------------------------------------------------------
Plan hash value: 211249738

-------------------------------------------------------------------------------
--
| Id  | Operation          | Name        | Rows  | Bytes | Cost (%CPU)| Time
 |

-------------------------------------------------------------------------------
--
|   0 | SELECT STATEMENT   |             | 29811 |  1018K|    72   (2)| 00:00:01
 |
|*  1 |  HASH JOIN         |             | 29811 |  1018K|    72   (2)| 00:00:01
 |
|   2 |   TABLE ACCESS FULL| DEPARTEMENT |   104 |  3224 |     3   (0)| 00:00:01
 |
|   3 |   TABLE ACCESS FULL| VILLE       | 29811 |   116K|    68   (0)| 00:00:01
 |

-------------------------------------------------------------------------------
--


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

   1 - access("VILLE"."DEP"="DEPARTEMENT"."ID")

Note
-----
   - dynamic statistics used: dynamic sampling (level=2)


Statistiques
----------------------------------------------------------
         72  recursive calls
          0  db block gets
       2783  consistent gets
          2  physical reads
          0  redo size
     653976  bytes sent via SQL*Net to client
      26892  bytes received via SQL*Net from client
       2442  SQL*Net roundtrips to/from client
         12  sorts (memory)
          0  sorts (disk)
      36601  rows processed
```

Figure 10: Sortie de console Question 6



```
36601 lignes selectionnees.


Plan d'execution
----------------------------------------------------------
Plan hash value: 1651012225

-------------------------------------------------------------------------------
--
| Id  | Operation          | Name        | Rows  | Bytes | Cost (%CPU)| Time
 |
-------------------------------------------------------------------------------
--
|   0 | SELECT STATEMENT   |             | 29811 |  1018K|  6928   (1)| 00:00:01
 |
|   1 |  NESTED LOOPS      |             | 29811 |  1018K|  6928   (1)| 00:00:01
 |
|   2 |   TABLE ACCESS FULL| DEPARTEMENT |   104 |  3224 |     3   (0)| 00:00:01
 |
|*  3 |   TABLE ACCESS FULL| VILLE       |   287 |  1148 |    67   (2)| 00:00:01
 |
-------------------------------------------------------------------------------
--


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

   3 - filter("VILLE"."DEP"="DEPARTEMENT"."ID")

Note
-----
   - dynamic statistics used: dynamic sampling (level=2)


Statistiques
----------------------------------------------------------
        158  recursive calls
          0  db block gets
      22615  consistent gets
          0  physical reads
          0  redo size
     650619  bytes sent via SQL*Net to client
      26892  bytes received via SQL*Net from client
       2442  SQL*Net roundtrips to/from client
         14  sorts (memory)
          0  sorts (disk)
      36601  rows processed
```

Figure 11: Sortie de console Question 7

```
select departement.nom from departement, ville where ville.insee='1293' and ville.dep = departement.id ;

NOM
--------------------------------------------------
Ain


Plan d'execution
----------------------------------------------------------
Plan hash value: 2507270265

--------------------------------------------------------------------------------
-------------

| Id  | Operation                    | Name          | Rows  | Bytes | Cost (%CP
U)| Time     |

--------------------------------------------------------------------------------
-------------

|   0 | SELECT STATEMENT             |               |     1 |    39 |     3   (
0)| 00:00:01 |

|   1 |  NESTED LOOPS                |               |     1 |    39 |     3   (
0)| 00:00:01 |

|   2 |   TABLE ACCESS BY INDEX ROWID| VILLE         |     1 |     8 |     2   (
0)| 00:00:01 |

|*  3 |    INDEX UNIQUE SCAN         | SYS_C00362565 |     1 |       |     1   (
0)| 00:00:01 |

|   4 |   TABLE ACCESS BY INDEX ROWID| DEPARTEMENT   |     1 |    31 |     1   (
0)| 00:00:01 |

|*  5 |    INDEX UNIQUE SCAN         | SYS_C00362562 |     1 |       |     0   (
0)| 00:00:01 |

--------------------------------------------------------------------------------
-------------


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

   3 - access("VILLE"."INSEE"='1293')
   5 - access("VILLE"."DEP"="DEPARTEMENT"."ID")


Statistiques
----------------------------------------------------------
         46  recursive calls
          0  db block gets
         76  consistent gets
          0  physical reads
          0  redo size
        546  bytes sent via SQL*Net to client
         52  bytes received via SQL*Net from client
          2  SQL*Net roundtrips to/from client
          5  sorts (memory)
          0  sorts (disk)
          1  rows processed
```

Figure 12: Sortie de console Question 8

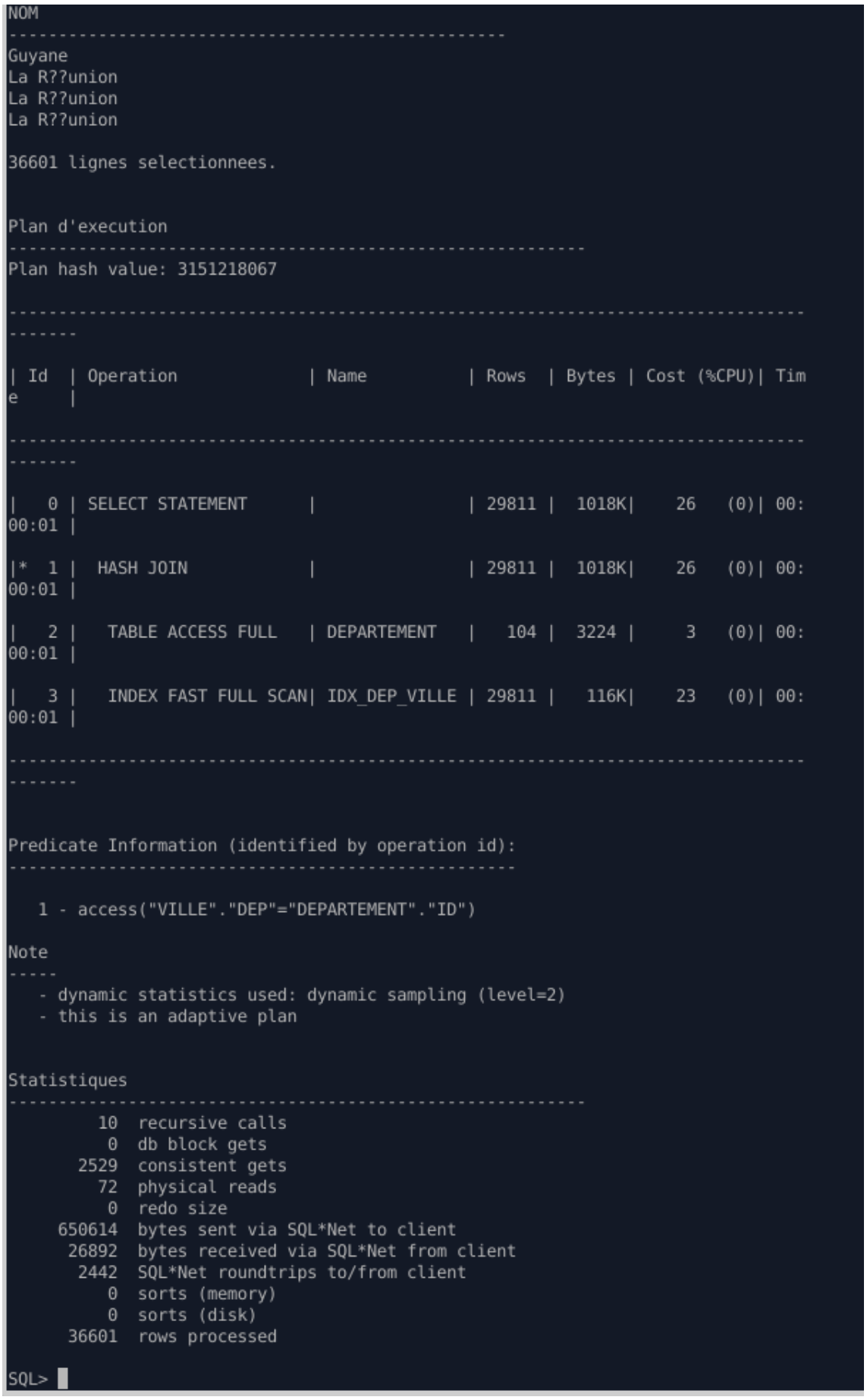
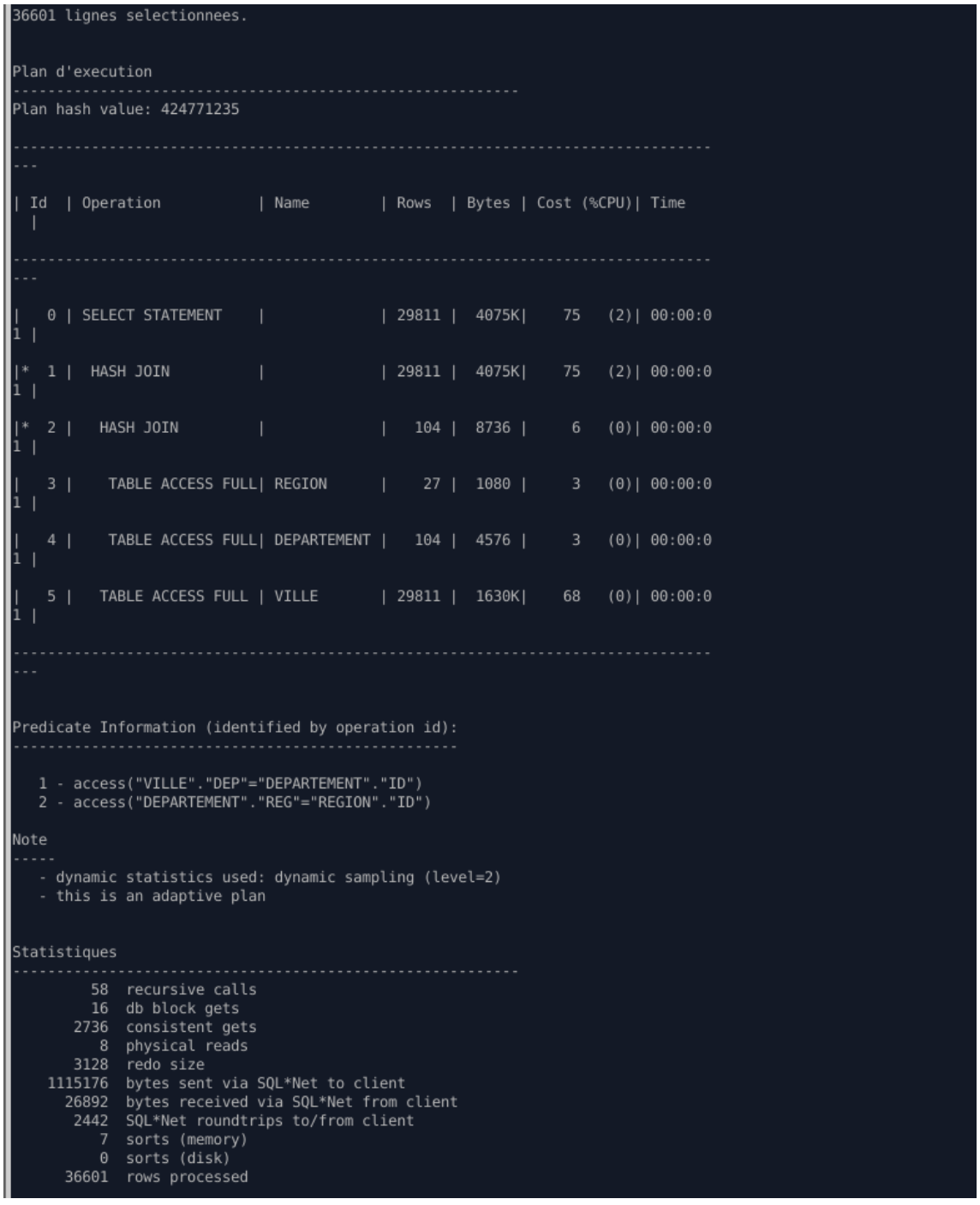
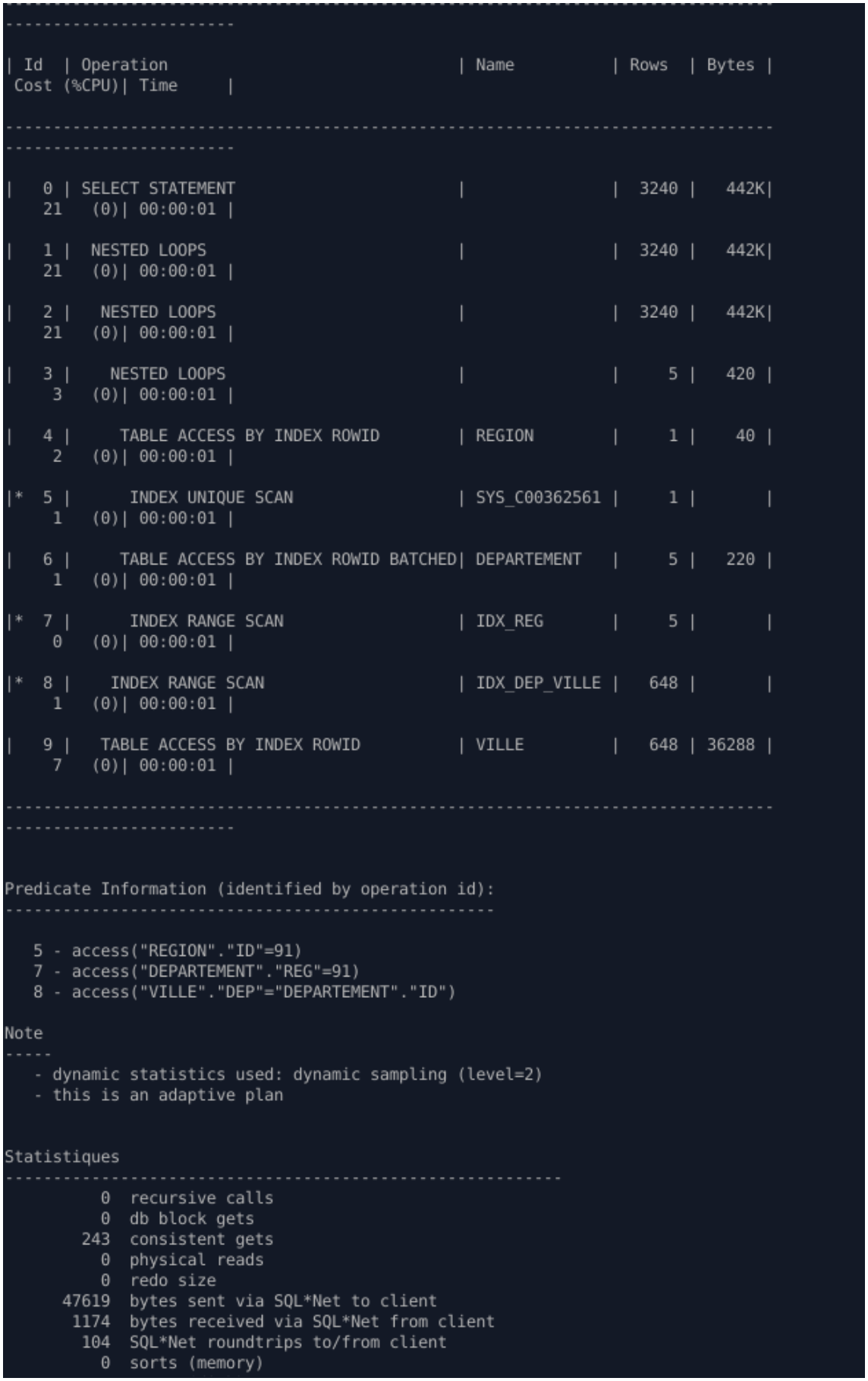


Figure 13: Sortie de console Question 8



```
36601 lignes selectionnees.


Plan d'execution
----------------------------------------------------------
Plan hash value: 424771235

--------------------------------------------------------------------------------
---
| Id  | Operation           | Name        | Rows  | Bytes | Cost (%CPU)| Time
  |

--------------------------------------------------------------------------------
---

|   0 | SELECT STATEMENT    |             | 29811 |  4075K|    75   (2)| 00:00:0
1 |

|*  1 |  HASH JOIN          |             | 29811 |  4075K|    75   (2)| 00:00:0
1 |

|*  2 |   HASH JOIN         |             |   104 |  8736 |     6   (0)| 00:00:0
1 |

|   3 |    TABLE ACCESS FULL| REGION      |    27 |  1080 |     3   (0)| 00:00:0
1 |

|   4 |    TABLE ACCESS FULL| DEPARTEMENT |   104 |  4576 |     3   (0)| 00:00:0
1 |

|   5 |   TABLE ACCESS FULL | VILLE       | 29811 |  1630K|    68   (0)| 00:00:0
1 |

--------------------------------------------------------------------------------
---


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

   1 - access("VILLE"."DEP"="DEPARTEMENT"."ID")
   2 - access("DEPARTEMENT"."REG"="REGION"."ID")

Note
-----
   - dynamic statistics used: dynamic sampling (level=2)
   - this is an adaptive plan


Statistiques
----------------------------------------------------------
         58  recursive calls
         16  db block gets
       2736  consistent gets
          8  physical reads
       3128  redo size
    1115176  bytes sent via SQL*Net to client
      26892  bytes received via SQL*Net from client
       2442  SQL*Net roundtrips to/from client
          7  sorts (memory)
          0  sorts (disk)
      36601  rows processed
```

Figure 14: Sortie de console Question 9



```
-----------------------

| Id  | Operation                           | Name          | Rows  | Bytes |
 Cost (%CPU)| Time     |

-------------------------------------------------------------------------------
-----------------------

|   0 | SELECT STATEMENT                    |               |  3240 |   442K|
    21   (0)| 00:00:01 |

|   1 |  NESTED LOOPS                       |               |  3240 |   442K|
    21   (0)| 00:00:01 |

|   2 |   NESTED LOOPS                      |               |  3240 |   442K|
    21   (0)| 00:00:01 |

|   3 |    NESTED LOOPS                     |               |     5 |   420 |
     3   (0)| 00:00:01 |

|   4 |     TABLE ACCESS BY INDEX ROWID     | REGION        |     1 |    40 |
     2   (0)| 00:00:01 |

|*  5 |      INDEX UNIQUE SCAN              | SYS_C00362561 |     1 |       |
     1   (0)| 00:00:01 |

|   6 |     TABLE ACCESS BY INDEX ROWID BATCHED| DEPARTEMENT   |     5 |   220 |
     1   (0)| 00:00:01 |

|*  7 |      INDEX RANGE SCAN               | IDX_REG       |     5 |       |
     0   (0)| 00:00:01 |

|*  8 |    INDEX RANGE SCAN                 | IDX_DEP_VILLE |   648 |       |
     1   (0)| 00:00:01 |

|   9 |   TABLE ACCESS BY INDEX ROWID       | VILLE         |   648 | 36288 |
     7   (0)| 00:00:01 |

-------------------------------------------------------------------------------
-----------------------


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

   5 - access("REGION"."ID"=91)
   7 - access("DEPARTEMENT"."REG"=91)
   8 - access("VILLE"."DEP"="DEPARTEMENT"."ID")

Note
-----
   - dynamic statistics used: dynamic sampling (level=2)
   - this is an adaptive plan


Statistiques
----------------------------------------------------------
          0  recursive calls
          0  db block gets
        243  consistent gets
          0  physical reads
          0  redo size
      47619  bytes sent via SQL*Net to client
       1174  bytes received via SQL*Net from client
        104  SQL*Net roundtrips to/from client
          0  sorts (memory)
```

Figure 15: Sortie de console Question 11

```
----------------------------------------------------------
Plan hash value: 2242427799

-------------------------------------------------------------------------------
------------

| Id  | Operation                     | Name        | Rows  | Bytes | Cost (%CPU
)| Time     |

-------------------------------------------------------------------------------
------------

|   0 | SELECT STATEMENT              |             | 29811 |  4075K|    75   (3
)| 00:00:01 |

|*  1 |  HASH JOIN                    |             | 29811 |  4075K|    75   (3
)| 00:00:01 |

|   2 |   MERGE JOIN                  |             |   104 |  8736 |     6  (17
)| 00:00:01 |

|   3 |    TABLE ACCESS BY INDEX ROWID| DEPARTEMENT |   104 |  4576 |     2   (0
)| 00:00:01 |

|   4 |     INDEX FULL SCAN           | IDX_REG     |   104 |       |     1   (0
)| 00:00:01 |

|*  5 |    SORT JOIN                  |             |    27 |  1080 |     4  (25
)| 00:00:01 |

|   6 |     TABLE ACCESS FULL         | REGION      |    27 |  1080 |     3   (0
)| 00:00:01 |

|   7 |   TABLE ACCESS FULL           | VILLE       | 29811 |  1630K|    68   (0
)| 00:00:01 |

-------------------------------------------------------------------------------
------------


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

   1 - access("VILLE"."DEP"="DEPARTEMENT"."ID")
   5 - access("DEPARTEMENT"."REG"="REGION"."ID")
       filter("DEPARTEMENT"."REG"="REGION"."ID")

Note
-----
   - dynamic statistics used: dynamic sampling (level=2)
   - this is an adaptive plan


Statistiques
----------------------------------------------------------
         11  recursive calls
          0  db block gets
       2643  consistent gets
          0  physical reads
          0  redo size
    1115176  bytes sent via SQL*Net to client
      26892  bytes received via SQL*Net from client
       2442  SQL*Net roundtrips to/from client
          1  sorts (memory)
          0  sorts (disk)
      36601  rows processed
```

Figure 16: Sortie de console Question 10

```
VILLEFULLET
VILLEMAIN
VILLIERS-EN-BOIS
VILLIERS-EN-PLAINE
VILLIERS-SUR-CHIZE
VITRE
VOUHE
VOUILLE
VOULTEGON
XAINTRAY

4278 lignes selectionnees.


Plan d'execution
----------------------------------------------------------
Plan hash value: 1694568309

--------------------------------------------------------------------------------
--------------------

| Id  | Operation                           | Name          | Rows  | Bytes | Co
st (%CPU)| Time     |

--------------------------------------------------------------------------------
--------------------

|   0 | SELECT STATEMENT                    |               |  3118 |   170K|
 35   (0)| 00:00:01 |

|   1 |  TABLE ACCESS BY INDEX ROWID BATCHED| VILLE         |  3118 |   170K|
 35   (0)| 00:00:01 |

|*  2 |   INDEX RANGE SCAN                  | IDX_DEP_VILLE |  3118 |       |
 10   (0)| 00:00:01 |

--------------------------------------------------------------------------------
--------------------


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

   2 - access("VILLE"."DEP" LIKE '7%')
       filter("VILLE"."DEP" IS NOT NULL AND "VILLE"."DEP" LIKE '7%')

Note
-----
   - dynamic statistics used: dynamic sampling (level=2)


Statistiques
----------------------------------------------------------
          0  recursive calls
          0  db block gets
        609  consistent gets
          0  physical reads
          0  redo size
     171957  bytes sent via SQL*Net to client
       3187  bytes received via SQL*Net from client
        287  SQL*Net roundtrips to/from client
          0  sorts (memory)
          0  sorts (disk)
       4278  rows processed

SQL>
```

Figure 17: Sortie de console Question 12

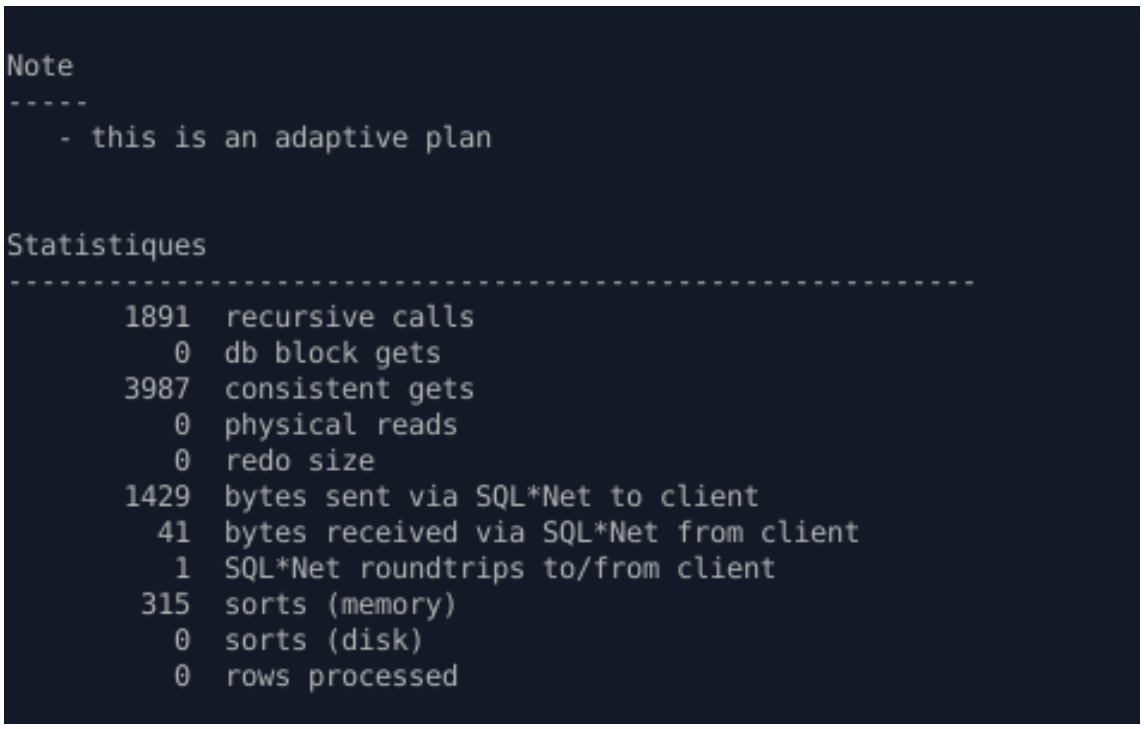
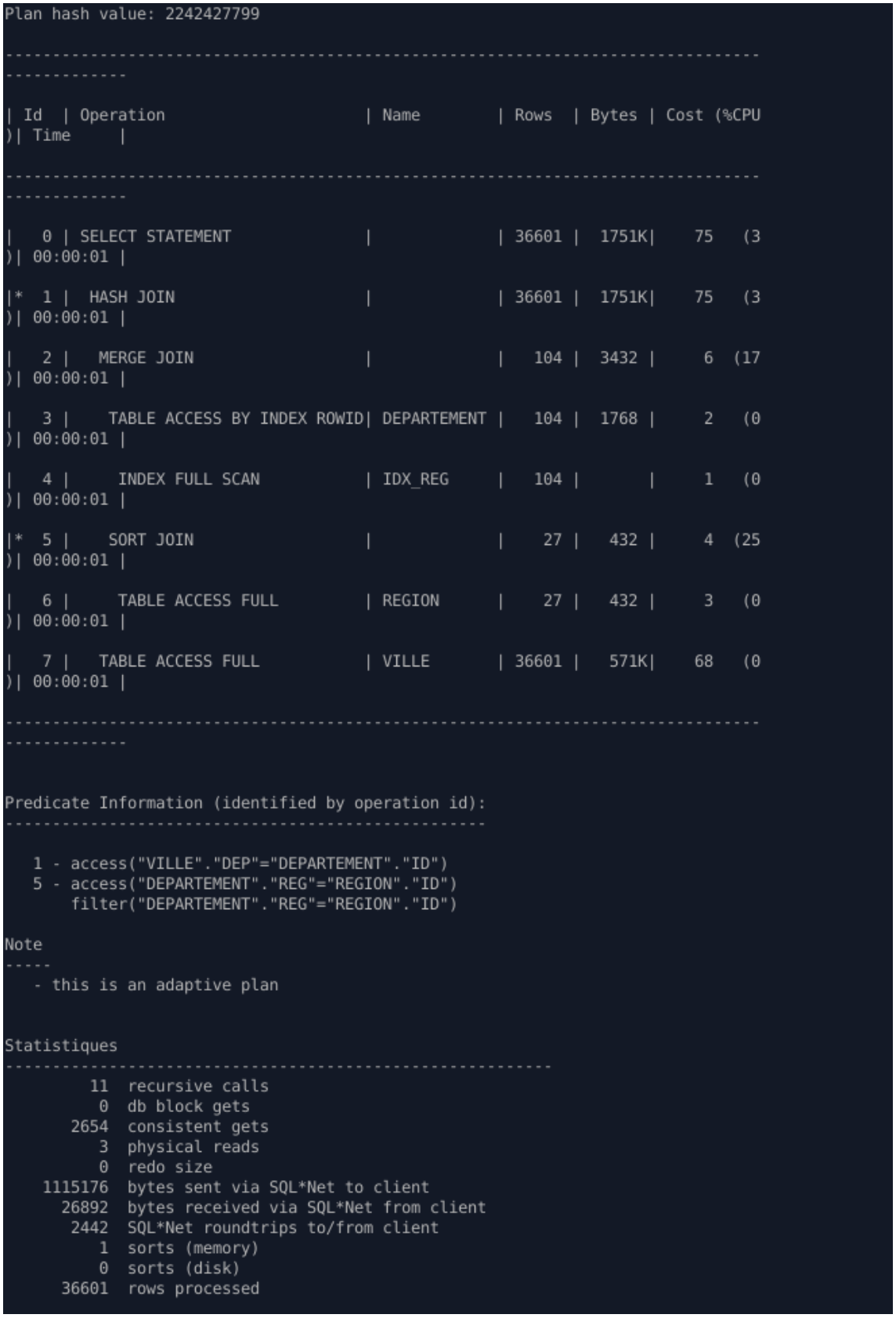


Figure 18: Sortie de console Question 13

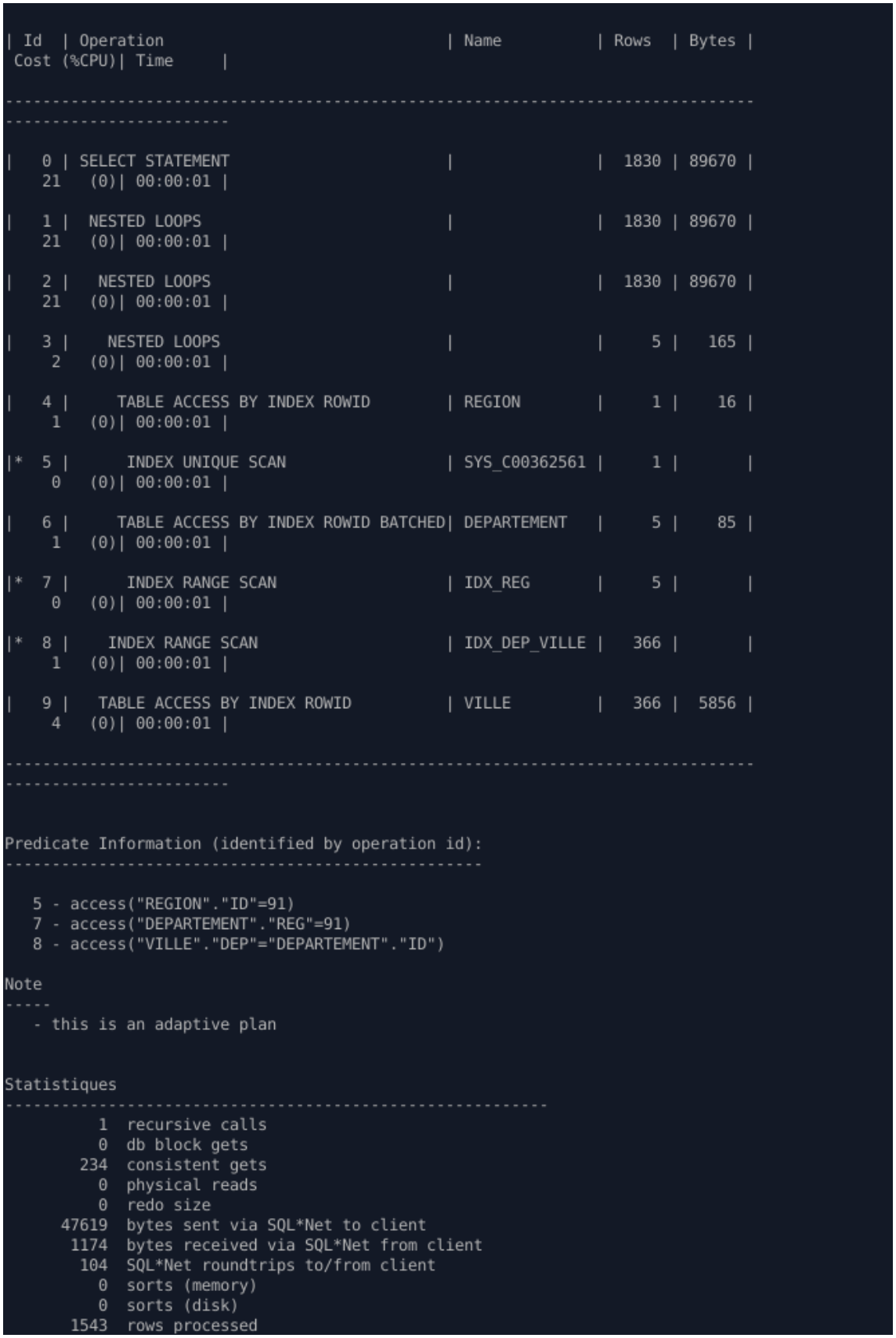
```
36601 lignes selectionnees.


Plan d'execution
----------------------------------------------------------
Plan hash value: 3151218067

-------------------------------------------------------------------------------
-------

| Id  | Operation             | Name          | Rows  | Bytes | Cost (%CPU)| Tim
e     |

-------------------------------------------------------------------------------
-------

|   0 | SELECT STATEMENT      |               | 36601 |   607K|    25   (4)| 00:
00:01 |

|*  1 |  HASH JOIN            |               | 36601 |   607K|    25   (4)| 00:
00:01 |

|   2 |   TABLE ACCESS FULL   | DEPARTEMENT   |   104 |  1456 |     3   (0)| 00:
00:01 |

|   3 |   INDEX FAST FULL SCAN| IDX_DEP_VILLE | 36601 |   107K|    21   (0)| 00:
00:01 |

-------------------------------------------------------------------------------
-------


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

   1 - access("VILLE"."DEP"="DEPARTEMENT"."ID")

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


Statistiques
----------------------------------------------------------
         1  recursive calls
         0  db block gets
      2518  consistent gets
         0  physical reads
         0  redo size
    650614  bytes sent via SQL*Net to client
     26892  bytes received via SQL*Net from client
      2442  SQL*Net roundtrips to/from client
         0  sorts (memory)
         0  sorts (disk)
     36601  rows processed

SQL>
```

Figure 19: Sortie de console Question 14



```
Plan hash value: 2242427799

-------------------------------------------------------------------------------
------------

| Id  | Operation                     | Name        | Rows  | Bytes | Cost (%CPU
)| Time     |

-------------------------------------------------------------------------------
------------

|   0 | SELECT STATEMENT              |             | 36601 |  1751K|    75   (3
)| 00:00:01 |

|*  1 |  HASH JOIN                    |             | 36601 |  1751K|    75   (3
)| 00:00:01 |

|   2 |   MERGE JOIN                  |             |   104 |  3432 |     6  (17
)| 00:00:01 |

|   3 |    TABLE ACCESS BY INDEX ROWID| DEPARTEMENT |   104 |  1768 |     2   (0
)| 00:00:01 |

|   4 |     INDEX FULL SCAN           | IDX_REG     |   104 |       |     1   (0
)| 00:00:01 |

|*  5 |    SORT JOIN                  |             |    27 |   432 |     4  (25
)| 00:00:01 |

|   6 |     TABLE ACCESS FULL         | REGION      |    27 |   432 |     3   (0
)| 00:00:01 |

|   7 |   TABLE ACCESS FULL           | VILLE       | 36601 |   571K|    68   (0
)| 00:00:01 |

-------------------------------------------------------------------------------
------------


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

   1 - access("VILLE"."DEP"="DEPARTEMENT"."ID")
   5 - access("DEPARTEMENT"."REG"="REGION"."ID")
       filter("DEPARTEMENT"."REG"="REGION"."ID")

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


Statistiques
----------------------------------------------------------
         11  recursive calls
          0  db block gets
       2654  consistent gets
          3  physical reads
          0  redo size
    1115176  bytes sent via SQL*Net to client
      26892  bytes received via SQL*Net from client
       2442  SQL*Net roundtrips to/from client
          1  sorts (memory)
          0  sorts (disk)
      36601  rows processed
```

Figure 20: Sortie de console Question 14



```
| Id  | Operation                             | Name          | Rows  | Bytes |
 Cost (%CPU)| Time     |

-------------------------------------------------------------------------------
----------------------

|   0 | SELECT STATEMENT                      |               |  1830 | 89670 |
    21   (0)| 00:00:01 |

|   1 |  NESTED LOOPS                         |               |  1830 | 89670 |
    21   (0)| 00:00:01 |

|   2 |   NESTED LOOPS                        |               |  1830 | 89670 |
    21   (0)| 00:00:01 |

|   3 |    NESTED LOOPS                       |               |     5 |   165 |
     2   (0)| 00:00:01 |

|   4 |     TABLE ACCESS BY INDEX ROWID       | REGION        |     1 |    16 |
     1   (0)| 00:00:01 |

|*  5 |      INDEX UNIQUE SCAN                | SYS_C00362561 |     1 |       |
     0   (0)| 00:00:01 |

|   6 |     TABLE ACCESS BY INDEX ROWID BATCHED| DEPARTEMENT   |     5 |    85 |
     1   (0)| 00:00:01 |

|*  7 |      INDEX RANGE SCAN                 | IDX_REG       |     5 |       |
     0   (0)| 00:00:01 |

|*  8 |    INDEX RANGE SCAN                   | IDX_DEP_VILLE |   366 |       |
     1   (0)| 00:00:01 |

|   9 |   TABLE ACCESS BY INDEX ROWID         | VILLE         |   366 |  5856 |
     4   (0)| 00:00:01 |

-------------------------------------------------------------------------------
----------------------


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

   5 - access("REGION"."ID"=91)
   7 - access("DEPARTEMENT"."REG"=91)
   8 - access("VILLE"."DEP"="DEPARTEMENT"."ID")

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


Statistiques
----------------------------------------------------------
          1  recursive calls
          0  db block gets
        234  consistent gets
          0  physical reads
          0  redo size
      47619  bytes sent via SQL*Net to client
       1174  bytes received via SQL*Net from client
        104  SQL*Net roundtrips to/from client
          0  sorts (memory)
          0  sorts (disk)
       1543  rows processed
```

Figure 21: Sortie de console Question 14