SQL DANS UN LANGAGE DE PROG

SEANCE N°5

1°)

```
MILLANR> CREATE TABLE LignesCommande (idCommande NUMBER, idProduit NUMBER,
         CONSTRAINT pk_LignesCommande PRIMARY KEY (idCommande, idProduit),
         CONSTRAINT fk_LignesCommande_Commande FOREIGN KEY (idCommande) REFERENCES Commandes(idCommande),
         CONSTRAINT fk_LignesCommande_Produit FOREIGN KEY (idProduit) REFERENCES Produits(idProduit))
[2022-11-09 09:49:19] completed in 182 ms
MILLANR> INSERT INTO Clients (SELECT * FROM Palleja.OPT3_Clients)
[2022-11-09 09:49:20] 10,001 rows affected in 180 ms
MILLANR> INSERT INTO Produits (SELECT * FROM Palleja.OPT3_Produits)
[2022-11-09 09:49:20] 20 rows affected in 145 ms
MILLANR> INSERT INTO Commandes (SELECT * FROM Palleja.OPT3_Commandes)
[2022-11-09 09:49:22] 100,000 rows affected in 669 ms
MILLANR> INSERT INTO LignesCommande (SELECT * FROM Palleja.OPT3_LignesCommande)
[2022-11-09 09:49:25] 284,040 rows affected in 2 s 858 ms
MILLANR> COMMIT
[2022-11-09 09:49:26] completed in 271 ms
```

2°)

SEX	VILLECLIENT
H	Montpellier

Ecoulé : 00 :00 :00.01

Plan d'exécution

```
Plan hash value: 4036073249

-----------------------------------------------------------------------------
| Id  | Operation          | Name    | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |         |     1 |    36 |    23   (0)| 00:00:01 |
|*  1 |  TABLE ACCESS FULL| CLIENTS |     1 |    36 |    23   (0)| 00:00:01 |
-----------------------------------------------------------------------------

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

   1 - filter("NOMCLIENT"='Palleja')

Note
-----
   - dynamic sampling used for this statement (level=2)
```

Statistiques

0	recursive calls
0	db block gets
84	consistent gets
0	physical reads
0	redo size
413	bytes sent via SQL*Net to client
338	bytes received via SQL*Net from client
2	SQL*Net roundtrips to/from client
0	sorts (memory)
0	sorts (disk)

1 rows processed

SEX	VILLECLIENT
H	Montpellier

Ecoulé : 00 :00 :00.00

Plan d'exécution

```
Plan hash value: 4036073249

-----------------------------------------------------------------------------
| Id  | Operation          | Name    | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |         |     1 |    36 |    23   (0)| 00:00:01 |
|*  1 |  TABLE ACCESS FULL| CLIENTS |     1 |    36 |    23   (0)| 00:00:01 |
-----------------------------------------------------------------------------

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

   1 - filter("NOMCLIENT"='Palleja')

Note
-----
   - dynamic sampling used for this statement (level=2)
```

Statistiques

0	recursive calls
0	db block gets
84	consistent gets
0	physical reads
0	redo size
413	bytes sent via SQL*Net to client
338	bytes received via SQL*Net from client
2	SQL*Net roundtrips to/from client
0	sorts (memory)
0	sorts (disk)

1 rows processed

3°)

A-

SELECT *
FROM clients
WHERE idClient='1000'

IDCLIENT	NOMCLIENT	PRENOMCLIENT	SEX	DATENAIS	VILLECLIENT	TELEPHONECLIENT
1000	Gomes	Maxime	H	05/07/64	Montgeron	06.91.73.00.99

Ecoulé : 00 :00 :00.00

Plan d'exécution

```
Plan hash value: 571565002

-------------------------------------------------------------------------------------------
| Id  | Operation                    | Name        | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |             |     1 |    81 |     1   (0)| 00:00:01 |
|   1 |  TABLE ACCESS BY INDEX ROWID | CLIENTS     |     1 |    81 |     1   (0)| 00:00:01 |
|*  2 |   INDEX UNIQUE SCAN          | PK_CLIENTS  |     1 |       |     1   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------

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

   2 - access("IDCLIENT"=1000)
```

Statistiques

8	recursive calls
0	db block gets
6	consistent gets
2	physical reads
124	redo size
743	bytes sent via SQL*Net to client
338	bytes received via SQL*Net from client
2	SQL*Net roundtrips to/from client
0	sorts (memory)
0	sorts (disk)

1 rows processed

Il utilise bien l'index pk_client

B-

Espace de travail

Saisissez les instructions SQL, PL/SQL et SQL*Plus.

```
SELECT /*+ no_index(Clients pk_Clients) */ *
FROM Clients
WHERE idClient='1000'
```

Exécuter | Charger script | Enregistrer script | Annuler

IDCLIENT	NOMCLIENT	PRENOMCLIENT	SEX	DATENAIS	VILLECLIENT	TELEPHONECLIENT
1000	Gomes	Maxime	H	05/07/64	Montgeron	06.91.73.00.99

Ecoulé : 00 :00 :00.01

Plan d'exécution

```
Plan hash value: 4036073249

-----------------------------------------------------------------------------
| Id  | Operation         | Name    | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |         |     1 |    81 |     2   (0)| 00:00:01 |
|*  1 |  TABLE ACCESS FULL| CLIENTS |     1 |    81 |     2   (0)| 00:00:01 |
-----------------------------------------------------------------------------

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

   1 - filter("IDCLIENT"=1000)
```

Statistiques

8	recursive calls
0	db block gets
87	consistent gets
0	physical reads
0	redo size
743	bytes sent via SQL*Net to client
338	bytes received via SQL*Net from client
2	SQL*Net roundtrips to/from client
0	sorts (memory)
0	sorts (disk)

1 rows processed

On constate que grâce au hint Oracle n'utilise pas les index et donc fait un parcours de la table Client.
Il y a alors 162 blocks

C°)

Espace de travail

Saisissez les instructions SQL, PL/SQL et SQL*Plus.

```
SET AUTOTRACE TRACEONLY
SELECT *
FROM Clients
WHERE idClient <> '1000'
```

Exécuter | Charger script | Enregistrer script | Annuler

10000 ligne(s) sélectionnée(s).

Ecoulé : 00 :00 :00.11

Plan d'exécution

```
Plan hash value: 4036073249

-----------------------------------------------------------------------------
| Id  | Operation         | Name    | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |         | 10155 |   803K|    23   (0)| 00:00:01 |
|*  1 |  TABLE ACCESS FULL| CLIENTS | 10155 |   803K|    23   (0)| 00:00:01 |
-----------------------------------------------------------------------------

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

   1 - filter("IDCLIENT"<>1000)

Note
-----
   - dynamic sampling used for this statement (level=2)
```

Statistiques

0	recursive calls
0	db block gets
747	consistent gets
0	physical reads
0	redo size
583899	bytes sent via SQL*Net to client
7664	bytes received via SQL*Net from client
668	SQL*Net roundtrips to/from client
0	sorts (memory)
0	sorts (disk)

10000 rows processed

L'optimiseur n'utilise pas pk_Client car l'optimiseur n'utilise pas les index sur les différences.

Saisissez les instructions SQL, PL/SQL et SQL*Plus.

```
SET AUTOTRACE TRACEONLY
SELECT /*+ index(Clients pk_Clients) */ *
FROM Clients
WHERE idClient <> '1000'
```

Exécuter | Charger script | Enregistrer script | Annuler

10000 ligne(s) sélectionnée(s).

Ecoulé : 00 :00 :00.12

Plan d'exécution

```
Plan hash value: 3864469006

-----------------------------------------------------------------------------------------
| Id  | Operation                   | Name       | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT            |            | 10155 |   803K|    66   (0)| 00:00:01 |
|   1 |  TABLE ACCESS BY INDEX ROWID| CLIENTS    | 10155 |   803K|    66   (0)| 00:00:01 |
|*  2 |   INDEX FULL SCAN           | PK_CLIENTS |   508 |       |    26   (0)| 00:00:01 |
-----------------------------------------------------------------------------------------

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

   2 - filter("IDCLIENT"<>1000)

Note
-----
   - dynamic sampling used for this statement (level=2)
```

Statistiques

11	recursive calls
0	db block gets
1514	consistent gets
0	physical reads
0	redo size
583870	bytes sent via SQL*Net to client
7664	bytes received via SQL*Net from client
668	SQL*Net roundtrips to/from client
0	sorts (memory)
0	sorts (disk)

10000 rows processed

En forçant l'utilisation de l'index la recherche est plus longue que sans l'index. Donc oui l'optimiseur avait raison de ne pas utiliser l'index car sans l'index il gagne 1 secondes.
Il y a aussi le même nombre de bloc sur les deux.

D-

Espace de travail

Saisissez les instructions SQL, PL/SQL et SQL*Plus.

```
SET AUTOTRACE TRACEONLY
SELECT *
FROM Commandes
WHERE idCommande > 60000
```

Exécuter | Charger script | Enregistrer script | Annuler

40000 ligne(s) sélectionnée(s).

Ecoulé : 00 :00 :00.66

Plan d'exécution

```
Plan hash value: 2577155004

-------------------------------------------------------------------------------
| Id  | Operation         | Name       | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |            | 22588 |  1257K|   137   (1)| 00:00:02 |
|*  1 |  TABLE ACCESS FULL| COMMANDES  | 22588 |  1257K|   137   (1)| 00:00:02 |
-------------------------------------------------------------------------------

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

   1 - filter("IDCOMMANDE">60000)

Note
-----
   - dynamic sampling used for this statement (level=2)
```

Statistiques

10	recursive calls
2	db block gets
3260	consistent gets
506	physical reads
652	redo size
1271623	bytes sent via SQL*Net to client
29664	bytes received via SQL*Net from client
2668	SQL*Net roundtrips to/from client
0	sorts (memory)
0	sorts (disk)

40000 rows processed

L'optimiseur n'utilise pas l'index pk_Client car il y a trop de tuple à retourne

Saisissez les instructions SQL, PL/SQL et SQL*Plus.

```
SET AUTOTRACE TRACEONLY
SELECT *
FROM Commandes
WHERE idCommande > 99000
```

Exécuter | Charger script | Enregistrer script | Annuler

1000 ligne(s) sélectionnée(s).

Ecoulé : 00 :00 :00.02

Plan d'exécution

```
Plan hash value: 3363529193

-------------------------------------------------------------------------------------------
| Id  | Operation                   | Name          | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT            |               |  1000 | 57000 |    57   (0)| 00:00:01 |
|   1 |  TABLE ACCESS BY INDEX ROWID| COMMANDES     |  1000 | 57000 |    57   (0)| 00:00:01 |
|*  2 |   INDEX RANGE SCAN          | PK_COMMANDES  |  1000 |       |     6   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------

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

   2 - access("IDCOMMANDE">99000)

Note
-----
   - dynamic sampling used for this statement (level=2)
```

Statistiques

7	recursive calls
1	db block gets
937	consistent gets
0	physical reads
268	redo size
32230	bytes sent via SQL*Net to client
1064	bytes received via SQL*Net from client
68	SQL*Net roundtrips to/from client
0	sorts (memory)
0	sorts (disk)

1000 rows processed

Maintenant l'optimiser utilise l'index pk_Clients car moins de lignes sont sélectionner

E-

Saisissez les instructions SQL, PL/SQL et SQL*Plus.

```
SET AUTOTRACE TRACEONLY
SELECT *
FROM Commandes
WHERE idCommande > 97562
```

Exécuter | Charger script | Enregistrer script | Annuler

2438 ligne(s) sélectionnée(s).

Ecoulé : 00 :00 :00.03

Plan d'exécution

```
Plan hash value: 3363529193

-----------------------------------------------------------------------------------------
| Id  | Operation                   | Name          | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT            |               |  2438 |   135K|   136   (0)| 00:00:02 |
|   1 |  TABLE ACCESS BY INDEX ROWID| COMMANDES     |  2438 |   135K|   136   (0)| 00:00:02 |
|*  2 |   INDEX RANGE SCAN          | PK_COMMANDES  |  2438 |       |    13   (0)| 00:00:01 |
-----------------------------------------------------------------------------------------

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

   2 - access("IDCOMMANDE">97562)

Note
-----
   - dynamic sampling used for this statement (level=2)
```

Statistiques

0	recursive calls
0	db block gets
2127	consistent gets
0	physical reads
0	redo size
77837	bytes sent via SQL*Net to client
2120	bytes received via SQL*Net from client
164	SQL*Net roundtrips to/from client
0	sorts (memory)
0	sorts (disk)

2438 rows processed

L'index est utilise à partir de '97562' donc 2438 lignes à sélectionner.
Le nombre de bloc lu a la limite est donc 270K

4°)

A-

Espace de travail

Saisissez les instructions SQL, PL/SQL et SQL*Plus.

```
SET AUTOTRACE ON;
```

Exécuter | Charger script | Enregistrer script | Annuler

SP2-0863 : Toutes les commandes iSQL*Plus ont été exécutées

B-

Espace de travail

Saisissez les instructions SQL, PL/SQL et SQL*Plus.

```
SELECT *
FROM Clients
WHERE nomClient='Claude'
```

Exécuter | Charger script | Enregistrer script | Annuler

IDCLIENT	NOMCLIENT	PRENOMCLIENT	SEX	DATENAIS	VILLECLIENT	TELEPHONECLIENT
2850	Claude	Martin	H	28/03/42	Smarves	06.36.51.98.11
4015	Claude	Émilie	F	11/04/54	Embreville	06.94.77.46.23
4614	Claude	Alexandra	F	06/05/75	Taulanne	06.65.06.43.50
4660	Claude	Chaïma	F	09/03/84	Magné	06.55.36.19.92
4259	Claude	Pauline	F	16/10/90	Olmeto	06.57.21.91.66
5380	Claude			11/06/64	Trémouille	06.71.71.29.58
6247	Claude	Thibault	H	04/06/71		06.44.01.04.17
6799	Claude	Guillaume	H	27/08/92		06.58.29.85.42
6873	Claude	Constant	H	17/03/43		06.94.62.95.58
6953	Claude	Bastien	H	24/05/88		06.32.98.53.11
6483	Claude	Léo	H	17/09/44		06.29.69.63.06
7566	Claude	Léonard	H	02/10/52		06.85.14.55.57
7821	Claude	Antonin	H	08/02/74		06.89.36.86.38
7334	Claude	Florian	H	09/12/45		06.65.42.43.23
7856	Claude	Davy	H	13/09/43		06.46.81.87.74
8510	Claude	Jordan	H	27/07/72		06.16.94.47.75
8694	Claude	Hugo	H	01/10/46		06.92.19.06.29
9490	Claude			12/09/59		06.08.22.95.00
9959	Claude			19/07/65		06.70.84.17.95

19 ligne(s) sélectionnée(s).

19 ligne(s) sélectionnée(s).

Ecoulé : 00 :00 :00.01

Plan d'exécution

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

Plan hash value: 4036073249

-------------------------------------------------------------------------------
| Id  | Operation         | Name    | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |         |    19 |  1539 |     23   (0)| 00:00:01 |
|*  1 |  TABLE ACCESS FULL| CLIENTS |    19 |  1539 |     23   (0)| 00:00:01 |
-------------------------------------------------------------------------------

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

   1 - filter("NOMCLIENT"='Claude')

Note
-----
   - dynamic sampling used for this statement (level=2)
```

Statistiques

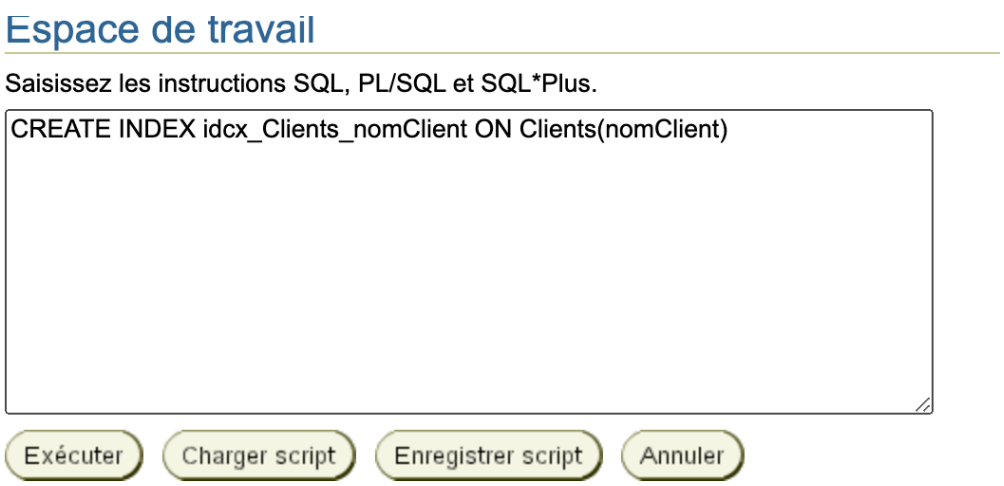
```
----------------------------------------------------------
```

19	recursive calls
0	db block gets
158	consistent gets
0	physical reads
0	redo size
1596	bytes sent via SQL*Net to client
349	bytes received via SQL*Net from client
3	SQL*Net roundtrips to/from client
0	sorts (memory)
0	sorts (disk)

19 rows processed

Il y a 3078 bloc lue en mémoire.

C-



D-

19 ligne(s) sélectionnée(s).

Ecoulé : 00 :00 :00.00

Plan d'exécution

```
Plan hash value: 2280551947

-----------------------------------------------------------------------------------------------------
| Id  | Operation                   | Name                  | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT            |                       |    19 |  1539 |    18   (0)| 00:00:01 |
|   1 |  TABLE ACCESS BY INDEX ROWID| CLIENTS               |    19 |  1539 |    18   (0)| 00:00:01 |
|*  2 |   INDEX RANGE SCAN          | IDCX_CLIENTS_NOMCLIENT|    19 |       |     1   (0)| 00:00:01 |
-----------------------------------------------------------------------------------------------------

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

   2 - access("NOMCLIENT"='Claude')

Note
-----
   - dynamic sampling used for this statement (level=2)
```

Statistiques

30	recursive calls
0	db block gets
99	consistent gets
1	physical reads
0	redo size
1596	bytes sent via SQL*Net to client
349	bytes received via SQL*Net from client
3	SQL*Net roundtrips to/from client
0	sorts (memory)
0	sorts (disk)

19 rows processed

Maintenant, le temps est de 0 alors que sans index il était de 1. Il y a le même nombre de bloc lue en mémoire
Finalement les Index sont très bien pour la lecture mais pas dutout pour l'écriture.

5°)

Ecoulé : 00 :00 :01.21

Plan d'exécution

```
Plan hash value: 534811400

-------------------------------------------------------------------------------
| Id  | Operation          | Name       | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | UPDATE STATEMENT   |            | 78675 |   998K|   137   (1)| 00:00:02 |
|   1 |  UPDATE            | COMMANDES  |       |       |            |          |
|   2 |   TABLE ACCESS FULL| COMMANDES  | 78675 |   998K|   137   (1)| 00:00:02 |
-------------------------------------------------------------------------------

Note
-----
   - dynamic sampling used for this statement (level=2)
```

Statistiques

149	recursive calls
209947	db block gets
1226	consistent gets
7	physical reads
52351604	redo size
727	bytes sent via SQL*Net to client
509	bytes received via SQL*Net from client
4	SQL*Net roundtrips to/from client
1	sorts (memory)
0	sorts (disk)

A-

Le temps d'execution de la commande est de 1.21s

B-

Espace de travail

Saisissez les instructions SQL, PL/SQL et SQL*Plus.

```
CREATE INDEX idx_Commandes_montantCommande ON
Commandes(montantCommande)
```

Exécuter | Charger script | Enregistrer script | Annuler

Index créé.

Ecoulé : 00 :00 :00.15

C-

Espace de travail

Saisissez les instructions SQL, PL/SQL et SQL*Plus.

```
UPDATE Commandes
SET MONTANTCOMMANDE=MONTANTCOMMANDE-10
```

Exécuter | Charger script | Enregistrer script | Annuler

100000 ligne(s) mise(s) à jour.

Ecoulé : 00 :00 :04.22

Plan d'exécution

```
Plan hash value: 534811400

-----------------------------------------------------------------------------------
| Id  | Operation           | Name       | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------
|   0 | UPDATE STATEMENT    |            | 78675 |   998K|   137   (1)| 00:00:02 |
|   1 |  UPDATE             | COMMANDES  |       |       |            |          |
|   2 |   TABLE ACCESS FULL | COMMANDES  | 78675 |   998K|   137   (1)| 00:00:02 |
-----------------------------------------------------------------------------------

Note
-----
   - dynamic sampling used for this statement (level=2)
```

Statistiques

276	recursive calls
540347	db block gets
2615	consistent gets
209	physical reads
74478892	redo size
729	bytes sent via SQL*Net to client
509	bytes received via SQL*Net from client
4	SQL*Net roundtrips to/from client
1	sorts (memory)
0	sorts (disk)

100000 rows processed

Le fait de crée un index augmente le temps d’update