Rapport
TP 8

Optimisation par index Bitmap et fragmentation horizontale

Réalisé en binôme par :
BENSALAH Kawthar / ABBACI Khaled
Numero du binôme : 22
Master 2 IL - Groupe 1
USTHB 2019/2020

TP 8

Objectif du TP

Optimisation par index Bitmap et fragmentation horizontale

Introduction

À travers ce TP, nous allons découvrire deux techniques d'optimisation des entrepôts de données : **l'indexation** (par index Bitmap et B-arbre) et **la fragmentation** (horizontale) .

Réponse 1

- **Activation des options autotrace et timing de oracle :**

```
SQL> set timing on;
SQL> set autotrace on explain;
SQL>
```

- **Ecriture et exécution de la requête R1 qui donne le nombre d'une banque donnée :**

```
SQL> alter system flush shared_pool;

Système modifié.

Ecoulé : 00 :00 :00.22
SQL> alter system flush buffer_cache;

Système modifié.

Ecoulé : 00 :00 :00.06
SQL>
SQL> Select count(NumAgence) as NbAgences
  2  From DAgence where NomBanque like 'BNA 1';

 NBAGENCES
----------
      1268

Ecoulé : 00 :00 :00.18

Plan d'exécution
----------------------------------------------------------
Plan hash value: 211357158

--------------------------------------------------------------------------------
| Id  | Operation          | Name    | Rows  | Bytes | Cost (%CPU)| Time     |
--------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |         |     1 |     7 |    30   (0)| 00:00:01 |
|   1 |  SORT AGGREGATE    |         |     1 |     7 |            |          |
|*  2 |   TABLE ACCESS FULL| DAGENCE |  1230 |  8610 |    30   (0)| 00:00:01 |
--------------------------------------------------------------------------------

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

   2 - filter("NOMBANQUE"='BNA 1')

SQL>
```

Réponse 2

-Création d'un index B-Arbre de la table DAgence sur l'attribut NomBanque :

```
SQL> CREATE Index Index_Banque on DAgence(NomBanque) TABLESPACE DefaultTBS2 ;

Index créé.

Ecoulé : 00 :00 :00.05
SQL>
```

Réponse 3

- Réexécution de la requête R1 :

```
SQL> alter system flush shared_pool;

Système modifié.

Ecoulé : 00 :00 :00.22
SQL> alter system flush buffer_cache;

Système modifié.

Ecoulé : 00 :00 :00.05
SQL>
```

```
SQL> Select count(NumAgence) as NbAgences
  2  From DAgence where NomBanque like 'BNA 1';

 NBAGENCES
----------
      1268

Ecoulé : 00 :00 :00.20

Plan d'exécution
----------------------------------------------------------
Plan hash value: 2419097964

-------------------------------------------------------------------------------
--
| Id  | Operation         | Name         | Rows  | Bytes | Cost (%CPU)| Time
 |
-------------------------------------------------------------------------------
--
|   0 | SELECT STATEMENT  |              |     1 |     7 |     3   (0)| 00:00:01
 |
|   1 |  SORT AGGREGATE   |              |     1 |     7 |            |
 |
|*  2 |   INDEX RANGE SCAN| INDEX_BANQUE |  1230 |  8610 |     3   (0)| 00:00:01
 |
-------------------------------------------------------------------------------
--

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

   2 - access("NOMBANQUE"='BNA 1')

SQL>
```

- **Temps d'exécution :** 00 : 00 : 00.20

- **Examination du plan d'exécution :** utilisation de l'index B-Arbre "Index Banque".

Remarque : D'après le plan d'exécution, l'index B-Arbre "Index Banque" a été utilisé ce qui a beaucoup amélioré le temps de réponse de la requête R1.

Réponse 4

- Suppression de l'index B-Arbre :

```
SQL> Drop Index Index_Banque;

Index supprimé.

Ecoulé : 00 :00 :00.24
SQL>
```

- Création d'un index bitmap sur la même table :

```
SQL> Create Bitmap Index Index_Banque on DAgence(NomBanque) TABLESPACE DefaultTBS2;

Index créé.

Ecoulé : 00 :00 :00.09
SQL>
```

Réponse 5

- Réexécution de la requête R1 :

```
SQL> alter system flush shared_pool;

Système modifié.

Ecoulé : 00 :00 :00.20
SQL> alter system flush buffer_cache;

Système modifié.

Ecoulé : 00 :00 :00.05
SQL>
```

```
SQL> Select count(NumAgence) as NbAgences
  2  From DAgence where NomBanque like 'BNA 1';

 NBAGENCES
----------
      1268

Ecoulé : 00 :00 :00.20

Plan d'exécution
----------------------------------------------------------
Plan hash value: 2006586747

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

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

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

|   0 | SELECT STATEMENT             |              |     1 |     7 |     1   (0)
| 00:00:01 |

|   1 |  SORT AGGREGATE              |              |     1 |     7 |
|          |

|   2 |   BITMAP CONVERSION COUNT    |              |  1230 |  8610 |     1   (0)
| 00:00:01 |

|*  3 |    BITMAP INDEX SINGLE VALUE| INDEX_BANQUE |       |       |
|          |

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


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

   3 - access("NOMBANQUE"='BNA 1')

SQL>
```

- **Temps d'exécution :** 00 : 00 : 00.20

- **Examination du plan d'exécution :** utilisation de l'index bitmap "Index Banque".

- ***Comparaison entre les 3 temps d'exécution :***

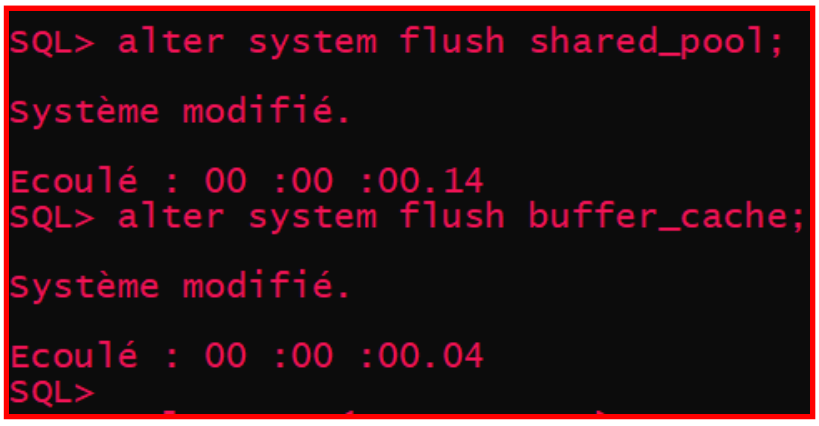
On remarque que les temps d'exécution avec ou sans l'utilisation des index bitmap et b-arbre sont presque égaux.

- Dans la question 1, le SGBD a exploité la table DAgence pour exécuter la requête.
- Après la création de l'index B-Arbre, le SGBD a utilisé cet index pour exécuter la requête.

- Après la suppression de l'index B-Arbre et la création de l'index bitmap, le SGBD a utilisé cet index pour exécuter la requête.

Réponse 6

- Ecriture de la requête R2 qui donne le montant versé global dans des comptes d'épargne :



```
SQL> Select sum(FO.MontantV) as MontV
  2  From FOperation FO, DTypeCompte DTC
  3  where FO.CodeTypeCompte = DTC.CodeType
  4  and DTC.LibType = 'Epargne';

     MONTV
----------
8897457388

Ecoulé : 00 :00 :00.36

Plan d'exécution
----------------------------------------------------------
Plan hash value: 577913969

--------------------------------------------------------------------------------
---
| Id  | Operation           | Name        | Rows  | Bytes | Cost (%CPU)| Time
   |
--------------------------------------------------------------------------------
---
|   0 | SELECT STATEMENT    |             |     1 |    19 |   903   (2)| 00:00:1
1 |
|   1 |  SORT AGGREGATE     |             |     1 |    19 |            |
   |
|*  2 |   HASH JOIN         |             |   304K|  5656K|   903   (2)| 00:00:1
1 |
|*  3 |    TABLE ACCESS FULL| DTYPECOMPTE |     1 |    11 |     3   (0)| 00:00:0
1 |
|   4 |    TABLE ACCESS FULL| FOPERATION  |   609K|  4763K|   897   (2)| 00:00:1
1 |
--------------------------------------------------------------------------------
---

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

Réponse 7

- Création d'un index bitmap de jointure entre FOperation et DTypeCompte :

```
SQL> Create Bitmap Index Intex_Bitmap on Foperation(DTC.LibType)
  2  FROM FOperation FO,
  3  DTypeCompte DTC
  4  Where (FO.CodeTypeCompte=DTC.CodeType) TABLESPACE DefaultTBS2;

Index créé.

Ecoulé : 00 :00 :01.80
SQL>
```

Réponse 8

- Réexécution de la requête R2

```
SQL> alter system flush shared_pool;

Système modifié.

Ecoulé : 00 :00 :00.21
SQL> alter system flush buffer_cache;

Système modifié.

Ecoulé : 00 :00 :00.05
SQL>
```

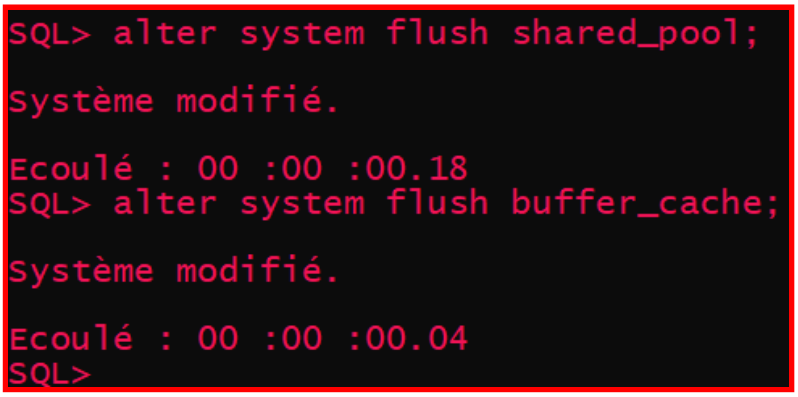
```
SQL> Select sum(FO.MontantV) as MontV
  2  From FOperation FO, DTypeCompte DTC
  3  where FO.CodeTypeCompte = DTC.CodeType
  4  and DTC.LibType = 'Epargne';

     MONTV
----------
8897457388

Ecoulé : 00 :00 :00.52

Plan d'exécution
----------------------------------------------------------
Plan hash value: 577913969

-------------------------------------------------------------------------------
---
| Id  | Operation           | Name        | Rows  | Bytes | Cost (%CPU)| Time
   |
-------------------------------------------------------------------------------
---
|   0 | SELECT STATEMENT    |             |     1 |    19 |   903   (2)| 00:00:1
1 |
|   1 |  SORT AGGREGATE     |             |     1 |    19 |            |
   |
|*  2 |   HASH JOIN         |             |   304K|  5656K|   903   (2)| 00:00:1
1 |
|*  3 |    TABLE ACCESS FULL| DTYPECOMPTE |     1 |    11 |     3   (0)| 00:00:0
1 |
|   4 |    TABLE ACCESS FULL| FOPERATION  |   609K|  4763K|   897   (2)| 00:00:1
1 |
-------------------------------------------------------------------------------
---
```

- ***Comparaison entre les 2 temps d'exécution :***

On remarque que l'index bitmap de jointure n'a pas été exploité pour diminuer le temps d'exécution de la requête. Dans les deux cas, le SGBD utilise les tables DTypeCompte et FOperation et n'utilise pas l'index de jointure car il a le choix d'exploiter les index ou pas lors de l'exécution des requêtes.

Réponse 9

- Ecriture de la requête R3 qui donne le montant versé global dans la wilaya d'alger :



```
SQL> Select sum(FO.MontantV) as MontV
  2  From FOperation FO, DAgence DA
  3  where FO.NumAgence = DA.NumAgence
  4  and DA.CodeWilaya = 16;

     MONTV
----------
 247631103

Ecoulé : 00 :00 :00.41

Plan d'exécution
----------------------------------------------------------
Plan hash value: 862488194

-------------------------------------------------------------------------------
--
| Id  | Operation           | Name       | Rows  | Bytes | Cost (%CPU)| Time
  |
-------------------------------------------------------------------------------
--
|   0 | SELECT STATEMENT    |            |     1 |    18 |   931   (2)| 00:00:12
  |
|   1 |  SORT AGGREGATE     |            |     1 |    18 |            |
  |
|*  2 |   HASH JOIN         |            | 12701 |   223K|   931   (2)| 00:00:12
  |
|*  3 |    TABLE ACCESS FULL| DAGENCE    |   256 |  2048 |    30   (0)| 00:00:01
  |
|   4 |    TABLE ACCESS FULL| FOPERATION |   609K|  5953K|   897   (2)| 00:00:11
  |
-------------------------------------------------------------------------------
--
```

Réponse 10

- Création d'un index bitmap de jointure entre FOperation et DAgence :

```
SQL> Create Bitmap Index Intex_Bitmap2 on Foperation(DA.NumAgence)
  2  FROM FOperation FO,
  3  DAgence DA
  4  Where (FO.NumAgence=DA.NumAgence) TABLESPACE DefaultTBS2;

Index créé.

Ecoulé : 00 :00 :02.09
SQL>
```

Réponse 11

- Réexécution de la requête R3 :

```
SQL> alter system flush shared_pool;

Système modifié.

Ecoulé : 00 :00 :00.19
SQL> alter system flush buffer_cache;

Système modifié.

Ecoulé : 00 :00 :00.05
SQL>
```

```
SQL> Select sum(FO.MontantV) as MontV
  2  From FOperation FO, DAgence DA
  3  where FO.NumAgence = DA.NumAgence
  4  and DA.CodeWilaya = 16;

     MONTV
----------
 247631103

Ecoulé : 00 :00 :00.32

Plan d'exécution
----------------------------------------------------------
Plan hash value: 862488194

-------------------------------------------------------------------------------
--
| Id  | Operation           | Name       | Rows  | Bytes | Cost (%CPU)| Time
 |
-------------------------------------------------------------------------------
--
|   0 | SELECT STATEMENT    |            |     1 |    18 |   931   (2)| 00:00:12
 |
|   1 |  SORT AGGREGATE     |            |     1 |    18 |            |
 |
|*  2 |   HASH JOIN         |            | 12701 |   223K|   931   (2)| 00:00:12
 |
|*  3 |    TABLE ACCESS FULL| DAGENCE    |   256 |  2048 |    30   (0)| 00:00:01
 |
|   4 |    TABLE ACCESS FULL| FOPERATION |   609K|  5953K|   897   (2)| 00:00:11
 |
-------------------------------------------------------------------------------
--

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

- ***Comparaison entre les 2 temps d'exécution :***

Selon le plan d'exécution, on remarque que le SGBD a choisi d'exploiter les deux tables DAgence et FOperation au lieu d'utiliser l'index de jointure créé. Le temps d'exécution donc reste le même.

Réponse 12

-Création d'un table FOpération2 identique à FOpération avec partitions :

```
SQL> CREATE TABLE FOperation2 (
  2      NumClient Number(10),
  3      NumAgence Number(10),
  4      CodeTypeCompte Number(10),
  5      CodeTemps Number(10),
  6      NbOperationR Number(10),
  7      NbOperationV Number(10),
  8      MontantR number(10,2),
  9      MontantV number(10,2),
 10      CONSTRAINT FK_O12
 11          FOREIGN KEY (NumClient)
 12          REFERENCES DClient(NumClient),
 13      CONSTRAINT FK_O22
 14          FOREIGN KEY (NumAgence)
 15          REFERENCES DAgence(NumAgence),
 16      CONSTRAINT FK_O32
 17          FOREIGN KEY (CodeTypeCompte)
 18          REFERENCES DTypeCompte(CodeType),
 19      CONSTRAINT FK_O42
 20          FOREIGN KEY (CodeTemps)
 21          REFERENCES DTemps(CodeTemps),
 22      CONSTRAINT PK_O2
 23          PRIMARY KEY (NumClient, NumAgence,CodeTypeCompte, CodeTemps)
 24  )
 25  PARTITION BY range(NumAgence)
 26  (
 27  partition P1 values LESS THAN (4000),
 28  partition P2 values LESS THAN (7000),
 29  partition P3 values LESS THAN (10000),
 30  partition P4 values LESS THAN (MAXVALUE)
 31  );

Table créée.

Ecoulé : 00 :00 :00.31
SQL>
```

Réponse 13

- Remplissage de la table FOperation2

```
SQL> begin
  2    for i in (
  3      SELECT NumClient, NumAgence, CodeTypeCompte, CodeTemps, NbOperationR, NbOperationV, MontantR, MontantV
  4      FROM FOperation)
  5      loop
  6          insert into FOperation2 values (i.NumClient, i.NumAgence, i.CodeTypeCompte,
  7          i.CodeTemps, i.NbOperationR, i.NbOperationV, i.MontantR, i.MontantV);
  8    end loop;
  9  COMMIT;
 10  end;
 11  /

Procédure PL/SQL terminée avec succès.

Ecoulé : 00 :00 :54.16
SQL>
```

Réponse 14

- Ecriture de la requête R4 qui donne le montant versé global dans l'agence N°12014 :

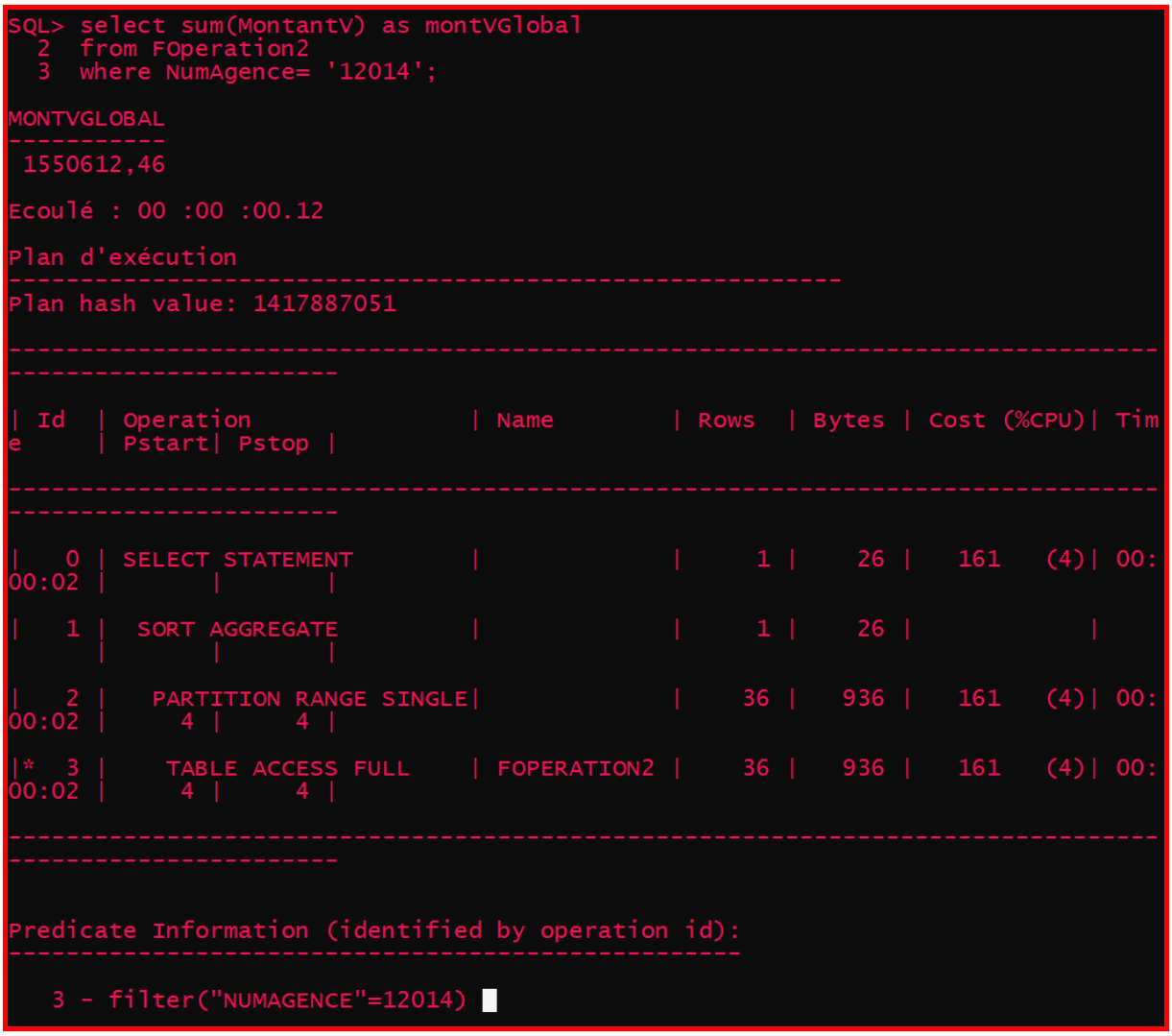
```
SQL> alter system flush shared_pool;

Système modifié.

Ecoulé : 00 :00 :00.22
SQL> alter system flush buffer_cache;

Système modifié.

Ecoulé : 00 :00 :00.85
```

```
SQL> select sum(MontantV) as montVGlobal
  2  from FOperation
  3  where NumAgence= '12014';

MONTVGLOBAL
-----------
 1550612,46

Ecoulé : 00 :00 :00.37

Plan d'exécution
----------------------------------------------------------
Plan hash value: 2432283172

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

--------------------------------------------------------------------------------
-
|   0 | SELECT STATEMENT   |            |     1 |    10 |   895   (1)| 00:00:11
|
|   1 |  SORT AGGREGATE    |            |     1 |    10 |            |
|
|*  2 |   TABLE ACCESS FULL| FOPERATION |    50 |   500 |   895   (1)| 00:00:11
|

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


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

   2 - filter("NUMAGENCE"=12014)

SQL>
```

Réponse 15

- Modification de la requête R4 pour utiliser FOpération2 :

```
SQL> alter system flush shared_pool;

Système modifié.

Ecoulé : 00 :00 :00.18
SQL> alter system flush buffer_cache;

Système modifié.
```



```
SQL> select sum(MontantV) as montVGlobal
  2  from FOperation2
  3  where NumAgence= '12014';

MONTVGLOBAL
-----------
 1550612,46

Ecoulé : 00 :00 :00.12

Plan d'exécution
----------------------------------------------------------
Plan hash value: 1417887051

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

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

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

|   0 | SELECT STATEMENT        |             |     1 |    26 |   161   (4)| 00:
00:02 |       |       |

|   1 |  SORT AGGREGATE         |             |     1 |    26 |            |
      |       |       |

|   2 |   PARTITION RANGE SINGLE|             |    36 |   936 |   161   (4)| 00:
00:02 |     4 |     4 |

|*  3 |    TABLE ACCESS FULL    | FOPERATION2 |    36 |   936 |   161   (4)| 00:
00:02 |     4 |     4 |

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


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

   3 - filter("NUMAGENCE"=12014)
```

- ***Comparaison entre les 2 temps d'exécution :***

On remarque que le temps d'exécution de la requête R4 a diminué (de 00:00:00.37 à 00:00:00:12) et ceci après la fragmentation horizontale de la table FOperation. En effet, dans la deuxième requête le SGBD n'a pas exploité toute la table FOperation2 mais uniquement la partition concernée par la requête (Partition 4).

Conclusion

Après la réalisation de ce TP, nous avons découvert deux techniques d'optimisation des requêtes entrepôts :
L'indexation par la création des index b-arbre ou bitmap qui permettent de récupérer les données assez rapidement. Le SGBD a donc le choix d'utiliser ces index ou non pour améliorer le temps de réponse des requêtes.
La fragmentation horizontale des tables qui permet réduire la complexité des requêtes exécutées sur un entrepôt de données car des sous ensembles sont plus facilement gérables qu'une table en entier.