Index et optimisation

Exemple de plan d'exécution - 1

```
SELECT empno
FROM bigemp
WHERE empno=100000;
```

Estimation du nombre de lignes retournées

Estimation du nombre de blocs accédés. N'est pas un facteur fiable de coût réel

```
-------------------------------------------------------------------------------
| Id  | Operation         | Name       | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |            |     1 |     6 |     2   (0)| 00:00:01 |
|*  1 |  INDEX UNIQUE SCAN| PK_BIGEMP  |     1 |     6 |     2   (0)| 00:00:01 |
-------------------------------------------------------------------------------

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

   1 - access("EMPNO"=100000)
```

Condition d'égalité avec une valeur de clé primaire

Exemple de plan d'exécution - 2

```
SELECT empno
FROM bigemp
WHERE job='ANALYST';
```

```
200002 lignes selectionnees.

Ecoule : 00 :00 :01.37

Plan d'execution
----------------------------------------------------------
Plan hash value: 2368838273

-------------------------------------------------------------------------------
| Id  | Operation          | Name   | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |        |   200K|  2734K|  2713   (1)| 00:00:01 |
|*  1 |  TABLE ACCESS FULL| BIGEMP |   200K|  2734K|  2713   (1)| 00:00:01 |
-------------------------------------------------------------------------------

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

   1 - filter("JOB"='ANALYST')
```

Colonne avec histogram

Exemple de plan d'exécution - 3

```
SELECT empno
FROM bigemp
WHERE job='CLERK';
```

```
400004 lignes selectionnees.

Ecoule : 00 :00 :02.67

Plan d'execution
----------------------------------------------------------
Plan hash value: 2368838273

-------------------------------------------------------------------------------
| Id  | Operation          | Name    | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |         |   400K|  5468K|  2713   (1)| 00:00:01 |
|*  1 |  TABLE ACCESS FULL| BIGEMP  |   400K|  5468K|  2713   (1)| 00:00:01 |
-------------------------------------------------------------------------------

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

   1 - filter("JOB"='CLERK')
```

Colonne avec histogram
Distribution déséquilibrée
(200K pour ANALYST)

Histogrammes

```
SELECT table_name, column_name, histogram
  2  FROM user_tab_columns
  3  WHERE table_name='BIGEMP';

TABLE_NAME           COLUMN_NAME          HISTOGRAM
-------------------- -------------------- ---------------
BIGEMP               EMPNO                NONE
BIGEMP               ENAME                FREQUENCY
BIGEMP               JOB                  FREQUENCY
BIGEMP               MGR                  NONE
BIGEMP               HIREDATE             FREQUENCY
BIGEMP               SAL                  FREQUENCY
BIGEMP               COMM                 FREQUENCY
BIGEMP               DEPTNO               NONE
```

```
SELECT table_name, column_name, endpoint_number, endpoint_value
  2  FROM user_histograms
  3  WHERE table_name='BIGEMP';

TABLE_NAME           COLUMN_NAME          ENDPOINT_NUMBER ENDPOINT_VALUE
-------------------- -------------------- --------------- --------------
BIGEMP               JOB                           200002     3,3909E+35
BIGEMP               JOB                           600006     3,4943E+35
BIGEMP               JOB                           900009     4,0113E+35
BIGEMP               JOB                          1000010     4,1705E+35
BIGEMP               JOB                          1400014     4,3229E+35
BIGEMP               COMM                          100001             0
BIGEMP               COMM                          200002           300
BIGEMP               COMM                          300003           500
BIGEMP               COMM                          400004          1400
```

Exemple de plan d'exécution - 4

```
SELECT empno
FROM bigemp
WHERE sal=1250;

-------------------------------------------------------------------------------
| Id  | Operation          | Name   | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |        |   200K|  1953K|  2715   (1)| 00:00:01 |
|*  1 |  TABLE ACCESS FULL| BIGEMP |   200K|  1953K|  2715   (1)| 00:00:01 |
-------------------------------------------------------------------------------


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


   1 - filter("SAL"=1250)
```

Estimation (colonne sans index et avec histogramme) : valeur issue de la table USER_TAB_HISTOGRAMS

Question 1.a

```
SELECT empno
FROM bigemp
WHERE empno>100000;

-------------------------------------------------------------------------------
| Id  | Operation             | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |           |  1398K|  8195K|  1399   (1)| 00:00:01 |
|*  1 |  INDEX FAST FULL SCAN | PK_BIGEMP |  1398K|  8195K|  1399   (1)| 00:00:01 |
-------------------------------------------------------------------------------

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

   1 - filter("EMPNO">100000)
```

Utilisation de l'index de la clé primaire

Le parcours de l'index seul est nécessaire, les données sont dans l'index (index couvrant). Un fast full scan est utilisé car les données n'ont pas besoin d'être parcourues dans l'ordre de l'index

Index Fast full scan/Range scan ou full scan

- Range scan ou full scan : leture des blocs de l'index un à la fois dans l'ordre de la clé de l'index
- Fast full scan : lecture des blocs dans l'ordre où ils apparaissent sur le disque et lecture possible de plusieurs blocs à la fois en fonction du paramétrage de la base, ce qui peut réduire les temps d'exécution des requêtes

Question 1.b

```
SELECT empno
FROM bigemp
WHERE empno<100000;

-------------------------------------------------------------------------------
| Id  | Operation         | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |           |  1297 |  7782 |     7   (0)| 00:00:01 |
|*  1 |  INDEX RANGE SCAN | PK_BIGEMP |  1297 |  7782 |     7   (0)| 00:00:01 |
-------------------------------------------------------------------------------


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

   1 - access("EMPNO"<100000)
```

Index couvrant, parcours de l'index uniquement. Requête sélective, parcours de valeurs adjacentes → range scan

Question 1.c

```
SELECT empno
FROM bigemp
WHERE empno between 100000 and 200000;
```

```
-------------------------------------------------------------------------------
| Id  | Operation         | Name       | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |            |  1402 |  8412 |     8   (0)| 00:00:01 |
|*  1 |  INDEX RANGE SCAN | PK_BIGEMP  |  1402 |  8412 |     8   (0)| 00:00:01 |
-------------------------------------------------------------------------------

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

   1 - access("EMPNO">=100000 AND "EMPNO"<=200000)
```

Index couvrant. Requête sélective, parcours de valeurs adjacentes

Transformation en inégalités

Question 1.d

```
SELECT empno
FROM bigemp
WHERE empno IN (100000,200000,300000);

-------------------------------------------------------------------------------
| Id  | Operation          | Name       | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |            |     3 |    18 |     5   (0)| 00:00:01 |
|   1 |  INLIST ITERATOR   |            |       |       |            |          |
|*  2 |   INDEX UNIQUE SCAN| PK_BIGEMP  |     3 |    18 |     5   (0)| 00:00:01 |
-------------------------------------------------------------------------------

Predicate Information (identified by operation id):

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

   2 - access("EMPNO"=100000 OR "EMPNO"=200000 OR "EMPNO"=300000)
```

Condition d'égalité avec une valeur de clé primaire

Itère sur l'opération suivante (2) pour chaque valeur de la liste

Question 1.e

```
SELECT empno
FROM bigemp
WHERE mod(empno,2)=0;
```

```
-----------------------------------------------------------------------------
| Id  | Operation             | Name       | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |            | 14000 | 84000 |  1407   (1)| 00:00:01 |
|*  1 |  INDEX FAST FULL SCAN | PK_BIGEMP  | 14000 | 84000 |  1407   (1)| 00:00:01 |
-----------------------------------------------------------------------------

Predicate Information (identified by operation id):

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

   1 - filter(MOD("EMPNO",2)=0)
```

Index couvrant. Parcours non ordonné

Question 1.f

```
SELECT empno
FROM bigemp
WHERE lower(ename) like 's%';
```

```
-------------------------------------------------------------------------
| Id  | Operation          | Name   | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |        |   200K|  2343K|  2718   (1)| 00:00:01 |
|*  1 |  TABLE ACCESS FULL| BIGEMP |   200K|  2343K|  2718   (1)| 00:00:01 |
-------------------------------------------------------------------------


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

   1 - filter(LOWER("ENAME") LIKE 's%')
```

Requête peu sélective, index sur une colonne avec peu de valeurs distinctes

Question 1.g

```
SELECT empno
FROM bigemp
WHERE empno>100000 and lower(ename) like 's%';

-------------------------------------------------------------------------------
| Id  | Operation          | Name   | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |        |   199K|  2341K|  2718   (1)| 00:00:01 |
|*  1 |  TABLE ACCESS FULL| BIGEMP |   199K|  2341K|  2718   (1)| 00:00:01 |
-------------------------------------------------------------------------------


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

   1 - filter(LOWER("ENAME") LIKE 's%' AND "EMPNO">100000)
```

Index non couvrants et requête peu sélective

Question 1.h

```
SELECT empno
FROM bigemp
WHERE lower(job) IN ('analyst','clerk');
```

```
-------------------------------------------------------------------------
| Id  | Operation          | Name   | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |        |   600K|  8203K|  2721   (1)| 00:00:01 |
|*  1 |  TABLE ACCESS FULL| BIGEMP |   600K|  8203K|  2721   (1)| 00:00:01 |
-------------------------------------------------------------------------

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

   1 - filter(LOWER("ENAME") LIKE 's%'  OR LOWER("JOB")='clerk'
```

Index non couvrants et requête peu sélective

Question 1.h

```
SELECT empno
FROM bigemp2
WHERE lower(job) IN ('analyst','clerk');
```

```
-------------------------------------------------------------------------
| Id  | Operation          | Name    | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |         |   600K|  8203K|  2756   (1)| 00:00:01 |
|*  1 |  TABLE ACCESS FULL| BIGEMP2 |   600K|  8203K|  2756   (1)| 00:00:01 |
-------------------------------------------------------------------------

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

1 - filter(LOWER("ENAME")='analyst' OR LOWER("JOB")='clerk')
```

Index non couvrants et requête peu sélective

Question 1.h, avec COUNT

```
SELECT count(*)
FROM bigemp
WHERE lower(job) IN ('analyst','clerk');
```

```
-------------------------------------------------------------------------------
| Id  | Operation                     | Name                | Rows | Bytes | Cost |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |                     |    1 |     8 |   63 |
|   1 |  SORT AGGREGATE               |                     |    1 |     8 |      |
|   2 |   INLIST ITERATOR             |                     |      |       |      |
|   3 |    BITMAP CONVERSION COUNT    |                     |  600K|  4687K|   63 |
|*  4 |     BITMAP INDEX SINGLE VALUE| IDX_BT_BIGEMP_LJOB  |      |       |      |
-------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------
   4 - access(LOWER("JOB")='analyst' OR LOWER("JOB")='clerk')
```

Question 1.h, avec COUNT

```
SELECT count(*)
FROM bigemp2
WHERE lower(job) IN ('analyst','clerk');
```

```
-------------------------------------------------------------------------------
| Id  | Operation                    | Name                | Rows | Bytes | Cost |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |                     |    1 |     8 | 1015 |
|   1 |  SORT AGGREGATE              |                     |    1 |     8 |      |
|   2 |   INDEX FAST FULL SCAN       | IDX_BIGEMP_LJOB     | 600K | 4687K | 1015 |
-------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------
   4 - access(LOWER("JOB")='analyst' OR LOWER("JOB")='clerk')
```

Autre configuration

- bigemp3

  CREATE BITMAP INDEX idx_bt_bigemp3_ljob on bigemp3(lower(job));

  CREATE BITMAP INDEX idx_bt_bigemp3_lename on bigemp3(lower(ename));

- bigemp4

  CREATE INDEX idx_bigemp4_ljob on bigemp4(lower(job));

  CREATE INDEX idx_bigemp4_lename on bigemp4(lower(ename));

Autre configuration

```
SELECT empno
FROM bigemp3
WHERE lower(job) IN ('analyst', 'clerk')
   AND lower(ename)='blake';
```

```
-----------------------------------------------------------------------------------------
| Id  | Operation                                | Name                  | Rows | Bytes | Cost |
-----------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                         |                       |42858 |  837K | 2514 |
|   1 |  TABLE ACCESS BY INDEX ROWID BATCHED|BIGEMP3                  |42858 |  837K | 2514 |
|   2 |   BITMAP CONVERSION TO ROWIDS            |                       |      |       |      |
|   3 |    BITMAP AND                            |                       |      |       |      |
|*  4 |     BITMAP INDEX SINGLE VALUE            |IDX_BT_BIGEMP3_LENAME|      |       |      |
|   5 |     BITMAP OR                            |                       |      |       |      |
|*  6 |      BITMAP INDEX SINGLE VALUE           | IDX_BT_BIGEMP3_LJOB |      |       |      |
|*  7 |      BITMAP INDEX SINGLE VALUE           | IDX_BT_BIGEMP3_LJOB |      |       |      |
-----------------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
    4 - access(LOWER("ENAME")='blake')
    6 - access(LOWER("JOB")='analyst')
    7 - access(LOWER("JOB")='clerk')
```

Combinaison des bitmap

Autre configuration

```
SELECT empno
FROM bigemp4
WHERE lower(job) IN ('analyst', 'clerk')
  AND lower(ename)='blake';

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

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

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

|   0 | SELECT STATEMENT   |         | 42858 |   837K|  2750   (1)|

|*  1 |  TABLE ACCESS FULL| BIGEMP4 | 42858 |   837K|  2750   (1)|

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


Predicate Information (identified by operation id):

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

   1 - filter(LOWER("ENAME")='blake' AND (LOWER("JOB")='analyst' OR

       LOWER("JOB")='clerk'))
```

Pas d'utilisation d'index. Index non couvrants et requête non sélective

Autre configuration

```
SELECT count(*)
FROM bigemp3
WHERE lower(job) IN ('analyst', 'clerk')
  AND lower(ename)='blake';
```

```
-----------------------------------------------------------------------------------------
| Id  | Operation                              | Name                  | Rows  | Bytes | Cost |
-----------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                       |                       |     1 |    14 |   92 |
|   1 |  SORT AGGREGATE                        |                       |     1 |    14 |      |
|   2 |   BITMAP CONVERSION COUNT              |                       | 42858 |   585K|   92 |
|   3 |    BITMAP AND                          |                       |       |       |      |
|*  4 |     BITMAP INDEX SINGLE VALUE          | IDX_BIGEMP3_LENAME    |       |       |      |
|   5 |      BITMAP OR                         |                       |       |       |      |
|*  6 |       BITMAP INDEX SINGLE VALUE        | IDX_BT_BIGEMP3_LJOB   |       |       |      |
|*  7 |       BITMAP INDEX SINGLE VALUE        | IDX_BT_BIGEMP3_LJOB   |       |       |      |
-----------------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------
   4 - access(LOWER("ENAME")='blake')
   6 - access(LOWER("JOB")='analyst')
   7 - access(LOWER("JOB")='clerk')
```

Autre configuration

```
SELECT count(*)
FROM bigemp
WHERE lower(job) IN ('analyst', 'clerk')
  AND lower(ename)='blake';
```

- Avec bigemp3
  temps écoulé : 00 :00 :00.01
  plan d'exécution en 7 étapes
  coût estimé : 14
- Avec bigemp4
  temps écoulé : 00 :00 :00.11
  plan d'exécution en 10 étapes
  coût estimé : 1828

Question 2.a

```
SELECT deptno
FROM bigdept
WHERE lower(loc) like 'd%';
```

```
-------------------------------------------------------------------------
| Id  | Operation             | Name                       | Rows  | Bytes |
-------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |                            | 99900 |  1365K|
|*  1 |  INDEX FAST FULL SCAN | IDX_BIGDEPT_DEPTNO_LLOC    | 99900 |  1365K|
-------------------------------------------------------------------------

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

   1 - filter(LOWER("LOC") LIKE 'd%')
```

Toutes les données utiles sont dans la clé de l'index : index couvrant. La condition porte sur la 2e colonne de l'index, parcours de l'index sans ordre (en inversant l'ordre des colonnes dans l'index→index range scan)

Question 2.b

```
SELECT deptno, dname
FROM bigdept
WHERE lower(loc) like 'd%';
```

```
-------------------------------------------------------------------------
| Id  | Operation          | Name    | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |         | 99900 |  2341K|   447   (1)| 00:00:01 |
|*  1 |  TABLE ACCESS FULL | BIGDEPT | 99900 |  2341K|   447   (1)| 00:00:01 |
-------------------------------------------------------------------------


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


   1 - filter(LOWER("LOC") LIKE 'd%')
```

Requête moyennement sélective et un accès à la ligne complète est nécessaire

Question 2.c

```
SELECT deptno
FROM bigdept
WHERE lower(loc) like '%a%';

-------------------------------------------------------------------
| Id  | Operation             | Name                    | Rows  | Bytes |
-------------------------------------------------------------------
|   0 | SELECT STATEMENT      |                         | 20000 |   273K|
|*  1 |  INDEX FAST FULL SCAN | IDX_BIGDEPT_DEPTNO_LLOC | 20000 |   273K|
-------------------------------------------------------------------

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

   1 - filter(LOWER("LOC") LIKE '%a%' AND LOWER("LOC") IS NOT NULL)
```

Index couvrant. Parcours de l'index sans ordre (même en inversant l'ordre des colonnes de l'index car la condition ne permet pas une recherche avec ordre)

Question 2.d

```
SELECT deptno
FROM bigdept
WHERE lower(loc) like '%s';

-----------------------------------------------------------------------
| Id  | Operation             | Name                       | Rows  | Bytes |
-----------------------------------------------------------------------
|   0 | SELECT STATEMENT      |                            | 20000 |   273K|
|*  1 |  INDEX FAST FULL SCAN | IDX_BIGDEPT_DEPTNO_LLOC    | 20000 |   273K|
-----------------------------------------------------------------------


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


   1 - filter(LOWER("LOC") LIKE '%s' AND LOWER("LOC") IS NOT NULL)
```

Index couvrant. Parcours de l'index sans ordre (voir diapo précédente)

Question 2.e

```
SELECT deptno
FROM bigdept
WHERE deptno>98;

-------------------------------------------------------------------------
| Id  | Operation             | Name                       | Rows  | Bytes |
-------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |                            |   400K|  2343K|
|*  1 |  INDEX FAST FULL SCAN| IDX_BIGDEPT_DEPTNO_LLOC |   400K|  2343K|
-------------------------------------------------------------------------


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

   1 - filter("DEPTNO">98)
```

Résultat similaire en forçant l'utilisation de PK_BIGDEPT

Question 2.f

```
SELECT dname
FROM bigdept
WHERE lower(loc) like 'd%';
```

```
-------------------------------------------------------------------------
| Id  | Operation          | Name    | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |         | 99900 |  1756K|   447   (1)| 00:00:01 |
|*  1 |  TABLE ACCESS FULL | BIGDEPT | 99900 |  1756K|   447   (1)| 00:00:01 |
-------------------------------------------------------------------------


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

   1 - filter(LOWER("LOC") LIKE 'd%')
```

L'index IDX_BIGDEPT_DEPTNO_LLOC n'est pas couvrant, et sa 1ère colonne n'est pas utilisée

Question 2.g

```
SELECT lower(loc), dname
FROM bigdept
ORDER BY lower(loc);
```

```
-------------------------------------------------------------------------
| Id  | Operation          | Name    | Rows  | Bytes |TempSpc| Cost (%CPU)|
-------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |         |   400K|  7031K|       |  2724   (1)|
|   1 |  SORT ORDER BY     |         |   400K|  7031K|    10M|  2724   (1)|
|   2 |   TABLE ACCESS FULL| BIGDEPT |   400K|  7031K|       |   445   (1)|
-------------------------------------------------------------------------
```

Espace temporaire nécessaire pour faire le tri

Question 2.h

```
SELECT deptno, lower(loc)
FROM bigdept
ORDER BY deptno, lower(loc);
```

```
-------------------------------------------------------------------------------
| Id  | Operation             | Name                        | Rows  | Bytes |TempSpc|
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |                             |   400K|  5468K|       |
|   1 |  SORT ORDER BY        |                             |   400K|  5468K|  9424K|
|   2 |   INDEX FAST FULL SCAN| IDX_BIGDEPT_DEPTNO_LLOC     |   400K|  5468K|       |
-------------------------------------------------------------------------------
```

Index couvrant

Question 2.i

```
SELECT deptno, lower(loc)
FROM bigdept
ORDER BY lower(loc), deptno;
```

```
-------------------------------------------------------------------------------
| Id  | Operation             | Name                       | Rows  | Bytes |TempSpc|
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |                            |   400K|  5468K|       |
|   1 |  SORT ORDER BY        |                            |   400K|  5468K|  9424K|
|   2 |   INDEX FAST FULL SCAN| IDX_BIGDEPT_DEPTNO_LLOC    |   400K|  5468K|       |
-------------------------------------------------------------------------------
```

Même requête qu'avant, mais les critères de tri ne sont pas dans l'ordre de l'index

Autre question

```
SELECT count(*)
FROM bigemp2;
```

```
-------------------------------------------------------------------------------
| Id  | Operation             | Name              | Rows  | Cost (%CPU)| Time
|
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |                   |     1 |  1086   (1)| 00:00:01
|   1 |  SORT AGGREGATE       |                   |     1 |            |
|   2 |   INDEX FAST FULL SCAN| PK_BIGEMP2        |  1400K|  1086   (1)| 00:00:01
-------------------------------------------------------------------------------
```

L'index permet de compter sans avoir besoin d'accéder aux données

Autre question

```
SELECT sum(sal)
FROM bigemp;
```

```
-----------------------------------------------------------------------------
| Id  | Operation          | Name    | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |         |     1 |     4 |  2713   (1)| 00:00:01 |
|   1 |  SORT AGGREGATE    |         |     1 |     4 |            |          |
|   2 |   TABLE ACCESS FULL| BIGEMP  |  1400K|  5468K|  2713   (1)| 00:00:01 |
-----------------------------------------------------------------------------
```

L'accès aux données est nécessaire

Question 8

```
SELECT empno
FROM bigemp
WHERE sal = (SELECT sal
             FROM bigemp
             WHERE empno=100007934);
```

En mettant IN au lieu de =, on a une étape de plus dans le plan d'exécution

Question 9

- Sur un COUNT l'index bitmap est plus rapide que l'index B-Tree
- COUNT(*) préconisé pour compter le nombre de lignes, permet de ne pas accéder à la table
- COUNT(colonne) compte le nombre de valeurs non renseignées de la colonne et peut impliquer un accès à la table, il est transformé en COUNT(*) si la colonne est spécifiée NOT NULL

Question 10

```
SELECT noclient, nocmd, count(*)
FROM cmd
GROUP BY noclient, nocmd
HAVING noclient=100597;
```

Mettre dans HAVING une condition qui pourrait être dans le WHERE dégrade les performances : la clause HAVING étant évaluée après les opérations d'agrégation, mettre les conditions sur les lignes dans le WHERE permet de profiter d'éventuels index et réduit le volume à traiter durant l'opération d'agrégation

Et dans d'autres SGBD ?

Postgres

- Nous allons voir quelques différences, ce n'est pas exhaustif et peut changer avec les versions
- Postgres ne crée pas automatiquement d'index pour la clé primaire
- Postgres n'a pas d'index bitmap (il utilise une structure bitmap pour faire des tris, mais c'est autre chose)

Postgres : Index basé sur une fonction

```
explain SELECT empno
        FROM bigemp.bigemp
        WHERE job='ANALYST' ;

                    QUERY PLAN
-----------------------------------------------------------
 Seq Scan on bigemp  (cost=0.00..30585.17 rows=198989 width=4)
   Filter: ((job)::text = 'ANALYST'::text)
```

```
CREATE INDEX idx_bigemp_ljob
on bigemp(lower(job));
```

Seq scan, équivalent de Full Table Scan pour Oracle

Postgres : Index basé sur une fonction

```
explain SELECT empno
        FROM bigemp.bigemp
        WHERE lower(job)='analyst' ;
                              QUERY PLAN
-----------------------------------------------------------------------------
 Bitmap Heap Scan on bigemp  (cost=4042.59..20112.43 rows=198989
width=4)
   Recheck Cond: (lower((job)::text) = 'analyst'::text)
   ->  Bitmap Index Scan on idx_bigemp_ljob  (cost=0.00..3992.85
rows=198989 width=0)
         Index Cond: (lower((job)::text) = 'analyst'::text)
```

Bitmap Index Scan : utilise une structure bitmap en mémoire pour trier les pointeurs de ligne et visiter les lignes dans la table dans l'ordre de leur emplacement physique.

MySQL

- Des index sont créés automatiquement sur la clé primaire et les clés étrangères
- Pas d'index bitmap
- Création d'index de fonction impossible

```
CREATE INDEX idx_bigemp_job
ON bigemp.BIGEMP(lower(job)); → impossible
```

MySQL

```
CREATE INDEX idx_bigemp_job
ON bigemp.BIGEMP(job);
```

```
explain SELECT empno
        FROM bigemp.BIGEMP
        WHERE job='ANALYST';
```

```
+--+-------------+---------+------------------+-------+---------+-------------------------+
| id | select_type | table     | key                | ref     | rows    | Extra                   |
+--+-------------+---------+------------------+-------+---------+-------------------------+
| 1 | SIMPLE       | BIGEMP | idx_bigemp_job | const  | 392022 | Using where; Using index|
+--+-------------+---------+------------------+-------+---------+-------------------------+
```

MySQL : EXPLAIN

- select_type : SIMPLE : SELECT simple, n'utilisant ni UNION, ni sous-requête
- key : index utilisé
- ref : colonne ou constante utilisée pour sélectionner les lignes de la table
- Extra : Using index ; Using where ; : l'index est utilisé pour rechercher les informations