Prácticas de Administración de Bases de Datos

Grado en Ingeniería Informática

PRÁCTICA 5

Procesamiento y optimización de consultas

Ejercicios

1. Tras una actualización de la base de datos (desde la versión 10.2.0.4), se ha informado de que algunas consultas no parecen ser tan eficientes como anteriormente. Se le pide al administrador que mientras no se encuentren posibles optimizaciones, cambie algún parámetro del optimizador de consultas para que funcione igual que en la versión anterior. ¿Cómo podría realizarse?.

   ALTER SYSTEM SET optimizer_features_enable='10.2.0.4';

   SHOW PARAMETER optimizer_features_enable

2. ¿Qué tendría que hacerse para que el optimizador no usara estadísticas dinámicas en la sesión actual?.

   ALTER SESSION SET optimizer_dynamic_sampling=0;

3. Un proceso realiza consultas a la base de datos y requiere bajos tiempos de respuesta, aunque no necesita todos los datos en un primer momento. ¿Cómo podría cambiarse el comportamiento del optimizador para conseguirlo?.

   ALTER SESSION SET OPTIMIZER_MODE=FIRST_ROWS_1;

4. Examina el contenido del script *catplan.sql*.

5. Lanza EXPLAIN PLAN (guardándolo con el `statement_id` *sentencia1*) para la siguiente consulta:

   ```
   SELECT phone_number
     FROM hr.employees
    WHERE phone_number LIKE '%16%';
   ```

   Muestra el plan mediante UTLXPLS.SQL y mediante DBMS_XPLAN.DISPLAY.
   ¿Qué diferencias hay entre hacerlo de una u otra manera?.

   Muestra el plan con el mayor nivel de detalle y también con el menor nivel de detalle.

   EXPLAIN PLAN SET statement_id = 'sentencia1' FOR SELECT phone_number
   FROM hr.employees
   WHERE phone_number LIKE '%16%';

//La ruta puede cambiar dependiendo de nuestra instalación
@C:\app\usuario\product\11.2.0\dbhome_1\RDBMS\ADMIN\UTLXPLS.SQL

SELECT PLAN_TABLE_OUTPUT
FROM TABLE(DBMS_XPLAN.DISPLAY(null, 'sentencia1'));

SELECT PLAN_TABLE_OUTPUT
FROM TABLE(DBMS_XPLAN.DISPLAY(null, 'sentencia1', 'ALL'));

```
SELECT PLAN_TABLE_OUTPUT
FROM TABLE(DBMS_XPLAN.DISPLAY(null, 'sentencia1', 'BASIC'));
```

6. Lanza EXPLAIN PLAN para la siguiente consulta:

```
SELECT *
  FROM hr.employees
  ORDER BY last_name;
```

¿Cambia algo del plan si modificas para la actual sesión el valor del parámetro `DB_FILE_MULTIBLOCK_READ_COUNT` a `1`? ¿Por qué?.

```
SHOW PARAMETER DB_FILE_MULTIBLOCK_READ_COUNT

EXPLAIN PLAN FOR SELECT *
FROM hr.employees
 ORDER BY last_name;

SELECT PLAN_TABLE_OUTPUT
FROM TABLE(DBMS_XPLAN.DISPLAY());
```

```
-------------------------------------------------------------------------------
| Id  | Operation          | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |           |   107 |  7383 |     4 (25)| 00:00:01 |
|   1 |  SORT ORDER BY     |           |   107 |  7383 |     4  (25)| 00:00:01 |
|   2 |   TABLE ACCESS FULL| EMPLOYEES |   107 |  7383 |     3   (0)| 00:00:01 |
-------------------------------------------------------------------------------
```

```
ALTER SESSION SET DB_FILE_MULTIBLOCK_READ_COUNT=1;

--Ahora el plan de ejecución queda--
```

```
-------------------------------------------------------------------------------
| Id  | Operation          | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |           |   107 |  7383 |     7 (15)| 00:00:01 |
|   1 |  SORT ORDER BY     |           |   107 |  7383 |     7  (15)| 00:00:01 |
|   2 |   TABLE ACCESS FULL| EMPLOYEES |   107 |  7383 |     6   (0)| 00:00:01 |
-------------------------------------------------------------------------------
```

```
ALTER SESSION SET DB_FILE_MULTIBLOCK_READ_COUNT=128;
```

7. Lanza EXPLAIN PLAN para la siguiente consulta:

```
SELECT *
  FROM   sh.customers
  WHERE  cust_city='Los Angeles'
  AND    cust_state_province='CA';
```

Examina el plan resultante. ¿Cambia algo del plan si modificas para la actual sesión el valor del parámetro `OPTIMIZER_DYNAMIC_SAMPLING` a `10`?.

```
EXPLAIN PLAN FOR
SELECT *
FROM  sh.customers
WHERE         cust_city='Los Angeles'
AND    cust_state_province='CA';
```

```
-------------------------------------------------------------------------------
| Id  | Operation         | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |           |     1 |   181 |   261   (1)| 00:00:04 |
|*  1 |  TABLE ACCESS FULL| CUSTOMERS |     1 |   181 |   261   (1)| 00:00:04 |
-------------------------------------------------------------------------------

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

   1 - filter("CUST_CITY"='Los Angeles' AND "CUST_STATE_PROVINCE"='CA')
```

ALTER SESSION SET OPTIMIZER_DYNAMIC_SAMPLING=10;

--ahora el plan queda…--

```
-------------------------------------------------------------------------------
| Id  | Operation         | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |           |   932 |  164K|   261   (1)| 00:00:04 |
|*  1 |  TABLE ACCESS FULL| CUSTOMERS |   932 |  164K|   261   (1)| 00:00:04 |
-------------------------------------------------------------------------------

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

   1 - filter("CUST_CITY"='Los Angeles' AND "CUST_STATE_PROVINCE"='CA')

Note
-----
   - dynamic sampling used for this statement (level=10)
```

Ha actualizado estadísticas

8. En cierto código se ha encontrado la siguiente consulta:

```
SELECT /*+ FULL(e) */ employee_id, last_name
  FROM hr.employees e
  WHERE last_name LIKE '%Smith%';
```

Analiza su plan de ejecución actual, si el plan de ejecución hace uso de algún índice y cómo podría mejorarse.

EXPLAIN PLAN FOR
SELECT /*+ FULL(e) */ employee_id, last_name
FROM hr.employees e
WHERE last_name LIKE '%Smith%';

```
-------------------------------------------------------------------------------
| Id  | Operation         | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |           |     5 |    60 |     3   (0)| 00:00:01 |
|*  1 |  TABLE ACCESS FULL| EMPLOYEES |     5 |    60 |     3   (0)| 00:00:01 |
-------------------------------------------------------------------------------

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

   1 - filter("LAST_NAME" LIKE '%Smith%')
```

EXPLAIN PLAN FOR
SELECT employee_id, last_name
FROM hr.employees e
WHERE last_name LIKE '%Smith%';

```
-------------------------------------------------------------------------------------------
| Id  | Operation             | Name             | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |                  |     5 |    60 |     3  (34)| 00:00:01 |
|   1 |  VIEW  JOIN           | index$_join$_001 |     5 |    60 |     3  (34)| 00:00:01 |
```

```
|*  2 |   HASH                  |                  |       |       |           |          |
|   3 |    INDEX  FAST FULL SCAN| EMP_EMP_ID_PK    |     5 |    60 |     1   (0)| 00:00:01 |
|*  4 |    INDEX  FAST FULL SCAN| EMP_NAME_IX      |     5 |    60 |     1   (0)| 00:00:01 |
-----------------------------------------------------------------------------------------------

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

   2 - access(ROWID=ROWID)
   4 - filter("LAST_NAME" LIKE '%Smith%')
```

EXPLAIN PLAN FOR
SELECT /*+ INDEX(e EMP_NAME_IX) */ employee_id, last_name FROM hr.employees e
WHERE last_name LIKE '%Smith%';

```
-----------------------------------------------------------------------------------------------
| Id  | Operation                    | Name         | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |              |     5 |    60 |     2   (0)| 00:00:01 |
|   1 |  TABLE ACCESS BY  INDEX ROWID| EMPLOYEES    |     5 |    60 |     2   (0)| 00:00:01 |
|*  2 |   INDEX FULL SCAN            | EMP_NAME_IX  |     5 |       |     1   (0)| 00:00:01 |
-----------------------------------------------------------------------------------------------
```

9. En cierto código se ha encontrado la siguiente consulta:

```
SELECT /*+ NO_USE_HASH(o l) */
       o.customer_id, l.unit_price * l.quantity
  FROM oe.orders o, oe.order_items l
 WHERE l.order_id = o.order_id;
```

Analiza su plan de ejecución y estudia si podría mejorarse.

EXPLAIN PLAN FOR
SELECT /*+ NO_USE_HASH(o l) */
o.customer_id, l.unit_price * l.quantity FROM oe.orders o, oe.order_items l
WHERE l.order_id = o.order_id;

```
-----------------------------------------------------------------------------------------------
| Id  | Operation                | Name              | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT         |                   |   665 | 13300 |     8 (38)| 00:00:01 |
|   1 |  MERGE JOIN              |                   |   665 | 13300 |     8 (38)| 00:00:01 |
|   2 |   SORT JOIN              |                   |   105 |   840 |     4 (50)| 00:00:01 |
|   3 |    VIEW                  | index$_join$_001  |   105 |   840 |     3 (34)| 00:00:01 |
|*  4 |     HASH JOIN            |                   |       |       |           |          |
|   5 |      INDEX FAST FULL SCAN| ORDER_PK          |   105 |   840 |     1   (0)| 00:00:01 |
|   6 |      INDEX FAST FULL SCAN| ORD_CUSTOMER_IX   |   105 |   840 |     1   (0)| 00:00:01 |
|*  7 |   SORT JOIN              |                   |   665 |  7980 |     4 (25)| 00:00:01 |
|   8 |    TABLE ACCESS FULL     | ORDER_ITEMS       |   665 |  7980 |     3   (0)| 00:00:01 |
-----------------------------------------------------------------------------------------------

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

   4 - access(ROWID=ROWID)
   7 -  access("L"."ORDER_ID"="O"."ORDER_ID")
       filter("L"."ORDER_ID"="O"."ORDER_ID")
```

EXPLAIN PLAN FOR
SELECT o.customer_id, l.unit_price * l.quantity
FROM oe.orders o, oe.order_items l
WHERE l.order_id = o.order_id;

```
-----------------------------------------------------------------------------------------------
| Id  | Operation                | Name              | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT         |                   |   665 | 13300 |     6 (17)| 00:00:01 |
|*  1 |  HASH JOIN               |                   |   665 | 13300 |     6 (17)| 00:00:01 |
|   2 |   VIEW                   | index$_join$_001  |   105 |   840 |     3 (34)| 00:00:01 |
```

```
|*   3 |    HASH JOIN                 |                  |       |       |            |          |
|    4 |     INDEX FAST FULL SCAN     | ORDER_PK         |   105 |   840 |     1   (0)| 00:00:01 |
|    5 |     INDEX FAST FULL SCAN     | ORD_CUSTOMER_IX  |   105 |   840 |     1   (0)| 00:00:01 |
|    6 |   TABLE ACCESS FULL          | ORDER_ITEMS      |   665 |  7980 |     3   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------------

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

   1 - access("L"."ORDER_ID"="O"."ORDER_ID")
   3 - access(ROWID=ROWID)
```

10. Para la siguiente consulta:

```
SELECT *
     FROM hr.employees e, hr.departments d, hr.job_history j
    WHERE e.department_id = d.department_id
      AND e.hire_date = j.start_date;
```

a) Examina su plan de ejecución.
b) Examina los posibles planes resultantes de emplear *LEADING*.

EXPLAIN PLAN FOR
SELECT * FROM hr.employees e, hr.departments d, hr.job_history j
WHERE e.department_id = d.department_id AND e.hire_date = j.start_date;

```
-------------------------------------------------------------------------------------------------
| Id  | Operation                     | Name         | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |              |    11 |  1331 |    10 (20)| 00:00:01 |
|*  1 |  HASH JOIN                    |              |    11 |  1331 |    10 (20)| 00:00:01 |
|   2 |   TABLE ACCESS FULL           | JOB_HISTORY  |    10 |   310 |     3  (0)| 00:00:01 |
|   3 |   MERGE JOIN                  |              |   106 |  9540 |     6 (17)| 00:00:01 |
|   4 |    TABLE ACCESS BY INDEX ROWID| DEPARTMENTS  |    27 |   567 |     2  (0)| 00:00:01 |
|   5 |     INDEX FULL SCAN           | DEPT_ID_PK   |    27 |       |     1  (0)| 00:00:01 |
|*  6 |    SORT JOIN                  |              |   107 |  7383 |     4 (25)| 00:00:01 |
|   7 |     TABLE ACCESS FULL         | EMPLOYEES    |   107 |  7383 |     3  (0)| 00:00:01 |
-------------------------------------------------------------------------------------------------

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

   1 - access("E"."HIRE_DATE"="J"."START_DATE")
   6 -   access("E"."DEPARTMENT_ID"="D"."DEPARTMENT_ID")
       filter("E"."DEPARTMENT_ID"="D"."DEPARTMENT_ID")
```

```
EXPLAIN PLAN FOR
SELECT /*+ LEADING(e d) */
        FROM hr.employees e, hr.departments d, hr.job_history j
       WHERE e.department_id =          d.department_id

        AND e.hire_date = j.start_date;
```

```
-------------------------------------------------------------------------------
| Id  | Operation            | Name         | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT     |              |    11 |  1331 |    10  (10)| 00:00:01 |
|*  1 |  HASH JOIN           |              |    11 |  1331 |    10  (10)| 00:00:01 |
|   2 |   TABLE ACCESS FULL  | JOB_HISTORY  |    10 |   310 |     3   (0)| 00:00:01 |
|*  3 |   HASH JOIN          |              |   106 |  9540 |     7  (15)| 00:00:01 |
|   4 |    TABLE ACCESS FULL | EMPLOYEES    |   107 |  7383 |     3   (0)| 00:00:01 |
|   5 |    TABLE ACCESS FULL | DEPARTMENTS  |    27 |   567 |     3   (0)| 00:00:01 |
-------------------------------------------------------------------------------

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

   1 - access("E"."HIRE_DATE"="J"."START_DATE")
   3 - access("E"."DEPARTMENT_ID"="D"."DEPARTMENT_ID")
```

```
EXPLAIN PLAN FOR
SELECT /*+ LEADING(e j) */
        FROM hr.employees e, hr.departments d, hr.job_history j
       WHERE e.department_id =          d.department_id

         AND e.hire_date = j.start_date;
```

```
-------------------------------------------------------------------------------
| Id  | Operation            | Name         | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT     |              |    11 |  1331 |    10  (10)| 00:00:01 |
|*  1 |  HASH JOIN           |              |    11 |  1331 |    10  (10)| 00:00:01 |
|*  2 |   HASH JOIN          |              |    11 |  1100 |     7  (15)| 00:00:01 |
|   3 |    TABLE ACCESS FULL | EMPLOYEES    |   107 |  7383 |     3   (0)| 00:00:01 |
|   4 |    TABLE ACCESS FULL | JOB_HISTORY  |    10 |   310 |     3   (0)| 00:00:01 |
|   5 |   TABLE ACCESS FULL  | DEPARTMENTS  |    27 |   567 |     3   (0)| 00:00:01 |
-------------------------------------------------------------------------------

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

   1 - access("E"."DEPARTMENT_ID"="D"."DEPARTMENT_ID")
   2-- access("E"."HIRE_DATE"="J"."START_DATE")
```

```
Explain plan for
SELECT /*+ LEADING(d j) */
        FROM hr.employees e, hr.departments d, hr.job_history j WHERE e.department_id =
       d.department_id AND e.hire_date = j.start_date;
```

```
-------------------------------------------------------------------------------
| Id  | Operation             | Name         | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |              |    11 |  1331 |    32   (4)| 00:00:01 |
|*  1 |  HASH JOIN            |              |    11 |  1331 |    32   (4)| 00:00:01 |
|   2 |   TABLE ACCESS FULL   | EMPLOYEES    |   107 |  7383 |     3   (0)| 00:00:01 |
|   3 |   MERGE JOIN CARTESIAN|              |   270 | 14040 |    28   (0)| 00:00:01 |
|   4 |    TABLE ACCESS FULL  | DEPARTMENTS  |    27 |   567 |     3   (0)| 00:00:01 |
|   5 |    BUFFER SORT        |              |    10 |   310 |    25   (0)| 00:00:01 |
|   6 |     TABLE ACCESS FULL | JOB_HISTORY  |    10 |   310 |     1   (0)| 00:00:01 |
-------------------------------------------------------------------------------

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

   1 - access("E"."DEPARTMENT_ID"="D"."DEPARTMENT_ID" AND
              "E"."HIRE_DATE"="J"."START_DATE")
```

11. Para la siguiente consulta:

```
SELECT *
  FROM hr.employees, hr.departments
  WHERE employees.department_id = departments.department_id;
```

Evalúa si es ventajoso emplear *USE_MERGE* o *NO_USE_MERGE*.

EXPLAIN PLAN FOR
SELECT *
FROM hr.employees, hr.departments
WHERE employees.department_id = departments.department_id;

```
-------------------------------------------------------------------------------------------------
| Id  | Operation                     | Name        | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |             |   106 |  9540 |     6 (17)| 00:00:01 |
|   1 |  MERGE JOIN                   |             |   106 |  9540 |     6 (17)| 00:00:01 |
|   2 |   TABLE ACCESS BY INDEX ROWID| DEPARTMENTS |    27 |   567 |     2   (0)| 00:00:01 |
|   3 |    INDEX FULL SCAN            | DEPT_ID_PK  |    27 |       |     1   (0)| 00:00:01 |
|*  4 |   SORT JOIN                   |             |   107 |  7383 |     4 (25)| 00:00:01 |
|   5 |    TABLE ACCESS FULL          | EMPLOYEES   |   107 |  7383 |     3   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------------

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

   4 -      access("EMPLOYEES"."DEPARTMENT_ID"="DEPARTMENTS"."DEPARTMENT_ID")
       filter("EMPLOYEES"."DEPARTMENT_ID"="DEPARTMENTS"."DEPARTMENT_ID")
```

EXPLAIN PLAN FOR
SELECT /*+ USE_MERGE(employees departments) */ *
FROM hr.employees, hr.departments
WHERE employees.department_id = departments.department_id;

```
-------------------------------------------------------------------------------------------------
| Id  | Operation                     | Name        | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |             |   106 |  9540 |     6 (17)| 00:00:01 |
|   1 |  MERGE JOIN                   |             |   106 |  9540 |     6 (17)| 00:00:01 |
|   2 |   TABLE ACCESS BY INDEX ROWID| DEPARTMENTS |    27 |   567 |     2   (0)| 00:00:01 |
|   3 |    INDEX FULL SCAN            | DEPT_ID_PK  |    27 |       |     1   (0)| 00:00:01 |
|*  4 |   SORT JOIN                   |             |   107 |  7383 |     4 (25)| 00:00:01 |
|   5 |    TABLE ACCESS FULL          | EMPLOYEES   |   107 |  7383 |     3   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------------

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

   4 -      access("EMPLOYEES"."DEPARTMENT_ID"="DEPARTMENTS"."DEPARTMENT_ID")
       filter("EMPLOYEES"."DEPARTMENT_ID"="DEPARTMENTS"."DEPARTMENT_ID")
```

EXPLAIN PLAN FOR
SELECT /*+ NO_USE_MERGE(employees departments) */ * FROM
hr.employees, hr.departments
WHERE employees.department_id = departments.department_id;

```
-----------------------------------------------------------------------------------
| Id  | Operation          | Name        | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |             |   106 |  9540 |     7 (15)| 00:00:01 |
|*  1 |  HASH JOIN         |             |   106 |  9540 |     7 (15)| 00:00:01 |
|   2 |   TABLE ACCESS FULL| DEPARTMENTS |    27 |   567 |     3   (0)| 00:00:01 |
|   3 |   TABLE ACCESS FULL| EMPLOYEES   |   107 |  7383 |     3   (0)| 00:00:01 |
-----------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------
   1 - access("EMPLOYEES"."DEPARTMENT_ID"="DEPARTMENTS"."DEPARTMENT_ID")
```

12. En cierto código se ha encontrado la siguiente consulta:

```
SELECT /*+ NO_USE_MERGE(e d) */ *
   FROM hr.employees e, hr.departments d
   WHERE e.department_id = d.department_id
   ORDER BY d.department_id;
```

Analiza su plan de ejecución y estudia si podría mejorarse.

```
EXPLAIN PLAN FOR
SELECT /*+ NO_USE_MERGE(e d) */ *
    FROM hr.employees e, hr.departments d
    WHERE e.department_id = d.department_id
    ORDER BY d.department_id;
```

```
-------------------------------------------------------------------------------
| Id  | Operation           | Name        | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT    |             |   106 |  9540 |     8  (25)| 00:00:01 |
|   1 |  SORT ORDER BY      |             |   106 |  9540 |     8  (25)| 00:00:01 |
|*  2 |   HASH JOIN         |             |   106 |  9540 |     7  (15)| 00:00:01 |
|   3 |    TABLE ACCESS FULL| DEPARTMENTS |    27 |   567 |     3   (0)| 00:00:01 |
|   4 |    TABLE ACCESS FULL| EMPLOYEES   |   107 |  7383 |     3   (0)| 00:00:01 |
-------------------------------------------------------------------------------

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

   2 - access("E"."DEPARTMENT_ID"="D"."DEPARTMENT_ID")
```

```
EXPLAIN PLAN FOR
SELECT *
    FROM hr.employees e, hr.departments d
    WHERE e.department_id = d.department_id

    ORDER BY d.department_id;
```

```
-----------------------------------------------------------------------------------------------
| Id  | Operation                     | Name        | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |             |   106 |  9540 |     6  (17)| 00:00:01 |
|   1 |  MERGE JOIN                   |             |   106 |  9540 |     6  (17)| 00:00:01 |
|   2 |   TABLE ACCESS BY INDEX ROWID| DEPARTMENTS |    27 |   567 |     2   (0)| 00:00:01 |
|   3 |    INDEX FULL SCAN            | DEPT_ID_PK  |    27 |       |     1   (0)| 00:00:01 |
|*  4 |   SORT JOIN                   |             |   107 |  7383 |     4  (25)| 00:00:01 |
|   5 |    TABLE ACCESS FULL          | EMPLOYEES   |   107 |  7383 |     3   (0)| 00:00:01 |
-----------------------------------------------------------------------------------------------

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

   4 -     access("E"."DEPARTMENT_ID"="D"."DEPARTMENT_ID")
       filter("E"."DEPARTMENT_ID"="D"."DEPARTMENT_ID")
```

13. Dada la siguiente consulta:

```
SELECT e1.last_name, e1.salary, v.avg_salary
   FROM hr.employees e1,
        (SELECT department_id, avg(salary) avg_salary
            FROM hr.employees e2
            GROUP BY department_id) v
   WHERE e1.department_id = v.department_id
     AND e1.salary > v.avg_salary
   ORDER BY e1.last_name
```

Compara su plan de ejecución con el correspondiente a usar *MERGE* en esa misma consulta.

```
EXPLAIN PLAN FOR
SELECT e1.last_name, e1.salary, v.avg_salary FROM
     hr.employees e1,
          (SELECT department_id, avg(salary) avg_salary FROM
               hr.employees e2
               GROUP BY department_id) v
     WHERE e1.department_id = v.department_id AND e1.salary
        > v.avg_salary
     ORDER BY e1.last_name
```

```
-------------------------------------------------------------------------------
| Id  | Operation             | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |           |    17 |   527 |     9 (34)| 00:00:01 |
|   1 |  SORT ORDER BY        |           |    17 |   527 |     9 (34)| 00:00:01 |
|*  2 |   HASH JOIN           |           |    17 |   527 |     8 (25)| 00:00:01 |
|   3 |    VIEW               |           |    11 |   176 |     4 (25)| 00:00:01 |
|   4 |     HASH GROUP BY     |           |    11 |    77 |     4 (25)| 00:00:01 |
|   5 |      TABLE ACCESS FULL| EMPLOYEES |   107 |   749 |     3  (0)| 00:00:01 |
|   6 |    TABLE ACCESS FULL  | EMPLOYEES |   107 |  1605 |     3  (0)| 00:00:01 |
-------------------------------------------------------------------------------

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

   2 -    access("E1"."DEPARTMENT_ID"="V"."DEPARTMENT_ID")
       filter("E1"."SALARY">"V"."AVG_SALARY")
```

```
EXPLAIN PLAN FOR
SELECT /*+ MERGE(v) */ e1.last_name, e1.salary, v.avg_salary FROM hr.employees e1,
          (SELECT department_id, avg(salary) avg_salary FROM
               hr.employees e2
               GROUP BY department_id) v
     WHERE e1.department_id = v.department_id AND e1.salary
        > v.avg_salary
     ORDER BY e1.last_name

SELECT PLAN_TABLE_OUTPUT
     FROM TABLE(DBMS_XPLAN.DISPLAY(null, null, 'ALL'));
```

```
-------------------------------------------------------------------------------------
| Id  | Operation             | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |           |   165 |  5610 |     9 (34)| 00:00:01 |
|   1 |  SORT ORDER BY        |           |   165 |  5610 |     9 (34)| 00:00:01 |
|*  2 |   FILTER              |           |       |       |           |          |
|   3 |    HASH GROUP BY      |           |   165 |  5610 |     9 (34)| 00:00:01 |
|*  4 |     HASH JOIN         |           |  3296 |   109K|     7 (15)| 00:00:01 |
|   5 |       TABLE ACCESS FULL| EMPLOYEES |   107 |  2889 |     3  (0)| 00:00:01 |
|   6 |       TABLE ACCESS FULL| EMPLOYEES |   107 |   749 |     3  (0)| 00:00:01 |
-------------------------------------------------------------------------------------

Query Block Name / Object Alias (identified by operation id):
-------------------------------------------------------------

   1 - SEL$F5BB74E1
   5 - SEL$F5BB74E1 / E1@SEL$1
   6 - SEL$F5BB74E1 / E2@SEL$2

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

   2  -filter("E1"."SALARY">SUM("SALARY")/COUNT("SALARY"))
   4  -access("E1"."DEPARTMENT_ID"="DEPARTMENT_ID")

Column Projection Information (identified by operation id):
-----------------------------------------------------------

   1  -(#keys=1) NLSSORT("E1"."LAST_NAME",'nls_sort=''SPANISH''')[210],
       "E1"."LAST_NAME"[VARCHAR2,25], "E1"."SALARY"[NUMBER,22],
       SUM("SALARY")/COUNT("SALARY")[22]
   2  -"E1"."SALARY"[NUMBER,22], "E1"."LAST_NAME"[VARCHAR2,25],
       COUNT("SALARY")[22], SUM("SALARY")[22]
   3  -(#keys=4) "DEPARTMENT_ID"[NUMBER,22], ROWID[ROWID,10],
       "E1"."SALARY"[NUMBER,22], "E1"."LAST_NAME"[VARCHAR2,25],
       COUNT("SALARY")[22], SUM("SALARY")[22]
   4  -(#keys=1) "DEPARTMENT_ID"[NUMBER,22], ROWID[ROWID,10],
       "E1"."LAST_NAME"[VARCHAR2,25], "E1"."SALARY"[NUMBER,22],
       "SALARY"[NUMBER,22]
   5  -ROWID[ROWID,10], "E1"."LAST_NAME"[VARCHAR2,25],
       "E1"."SALARY"[NUMBER,22], "E1"."DEPARTMENT_ID"[NUMBER,22]
   6  -"SALARY"[NUMBER,22], "DEPARTMENT_ID"[NUMBER,22]

LAST_NAME                     SALARY AVG_SALARY
 ------------------------- ---------- ----------
Abel                           11000  8955,88235
Bell                            4000  3475,55556
Bernstein                       9500  8955,88235
...
Vollman                         6500  3475,55556
Weiss                           8000  3475,55556
Zlotkey                        10500  8955,88235
 38 filas seleccionadas
```

14. Dada la siguiente consulta:

```
SELECT h.customer_id, l.unit_price * l.quantity
  FROM oe.orders h, oe.order_items l
  WHERE l.order_id = h.order_id;
```

Compara su plan de ejecución con el correspondiente a usar *USE_NL* en esa misma consulta.

EXPLAIN PLAN FOR

```
SELECT h.customer_id, l.unit_price * l.quantity
    FROM oe.orders h, oe.order_items l
    WHERE l.order_id = h.order_id;
```

```
-------------------------------------------------------------------------------------------
| Id  | Operation               | Name              | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT        |                   |   665 | 13300 |     6  (17)| 00:00:01 |
|*  1 |  HASH JOIN              |                   |   665 | 13300 |     6  (17)| 00:00:01 |
|   2 |   VIEW                  | index$_join$_001  |   105 |   840 |     3  (34)| 00:00:01 |
|*  3 |    HASH JOIN            |                   |       |       |            |          |
|   4 |     INDEX FAST FULL SCAN| ORDER_PK          |   105 |   840 |     1   (0)| 00:00:01 |
|   5 |     INDEX FAST FULL SCAN| ORD_CUSTOMER_IX   |   105 |   840 |     1   (0)| 00:00:01 |
|   6 |   TABLE ACCESS FULL     | ORDER_ITEMS       |   665 |  7980 |     3   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------

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

   1 - access("L"."ORDER_ID"="H"."ORDER_ID")
   3 - access(ROWID=ROWID)
```

```
EXPLAIN PLAN FOR
SELECT /*+ USE_NL(l h) */ h.customer_id, l.unit_price * l.quantity
FROM oe.orders h, oe.order_items l WHERE l.order_id = h.order_id;
```

```
-------------------------------------------------------------------------------------------
| Id  | Operation               | Name              | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT        |                   |   665 | 13300 |   148   (2)| 00:00:02 |
|   1 |  NESTED LOOPS           |                   |   665 | 13300 |   148   (2)| 00:00:02 |
|   2 |   VIEW                  | index$_join$_001  |   105 |   840 |     3  (34)| 00:00:01 |
|*  3 |    HASH JOIN            |                   |       |       |            |          |
|   4 |     INDEX FAST FULL SCAN| ORDER_PK          |   105 |   840 |     1   (0)| 00:00:01 |
|   5 |     INDEX FAST FULL SCAN| ORD_CUSTOMER_IX   |   105 |   840 |     1   (0)| 00:00:01 |
|*  6 |   TABLE ACCESS FULL     | ORDER_ITEMS       |     6 |    72 |     1   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------

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

   3 - access(ROWID=ROWID)
   6 - filter("L"."ORDER_ID"="H"."ORDER_ID")
```

15. Dada la siguiente consulta:

```
SELECT employee_id, department_id
  FROM hr.employees
  WHERE department_id > 50;
```

Estudia si su plan de ejecución puede mejorarse mediante el uso del índice *emp_department_ix*.

```
EXPLAIN PLAN FOR
SELECT employee_id, department_id
   FROM hr.employees
   WHERE department_id > 50;
```

```
-------------------------------------------------------------------------------------------
| Id  | Operation               | Name              | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT        |                   |    50 |   350 |     3  (34)| 00:00:01 |
|*  1 |  VIEW                   | index$_join$_001  |    50 |   350 |     3  (34)| 00:00:01 |
|*  2 |   HASH JOIN             |                   |       |       |            |          |
|*  3 |    INDEX RANGE SCAN     | EMP_DEPARTMENT_IX |    50 |   350 |     2  (50)| 00:00:01 |
|   4 |    INDEX  FAST FULL SCAN| EMP_EMP_ID_PK     |    50 |   350 |     1   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------
```

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

   1 - filter("DEPARTMENT_ID">50)
   2 - access(ROWID=ROWID)
   3 - access("DEPARTMENT_ID">50)
```

```
EXPLAIN PLAN FOR
SELECT /*+ INDEX (employees emp_department_ix)*/ employee_id, department_id FROM
   hr.employees
   WHERE department_id > 50;
```

```
-------------------------------------------------------------------------------------------------------
| Id  | Operation                    | Name               | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------------------
|   0 | SELECT  STATEMENT            |                    |    50 |   350 |     6   (0)| 00:00:01 |
|   1 |  TABLE  ACCESS BY INDEX ROWID| EMPLOYEES          |    50 |   350 |     6   (0)| 00:00:01 |
|*  2 |    INDEX RANGE SCAN          |  EMP_DEPARTMENT_IX |    50 |       |     1   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------------------

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

   2 - access("DEPARTMENT_ID">50)
```

16. En cierto código se ha encontrado la siguiente consulta:

```
SELECT /*+ NO_INDEX(employees emp_empid) */ employee_id
  FROM hr.employees
  WHERE employee_id > 200;
```

Analiza su plan de ejecución e indica si convendría realizar algún cambio.

```
EXPLAIN PLAN FOR
SELECT /*+ NO_INDEX(employees emp_empid) */ employee_id
   FROM hr.employees
   WHERE employee_id > 200;
```

```
-------------------------------------------------------------------------------------
| Id  | Operation        | Name          | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT |               |     6 |    24 |     1   (0)| 00:00:01 |
|*  1 |  INDEX RANGE SCAN| EMP_EMP_ID_PK |     6 |    24 |     1   (0)| 00:00:01 |
-------------------------------------------------------------------------------------

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

   1 - access("EMPLOYEE_ID">200)
```

```
EXPLAIN PLAN FOR
SELECT employee_id
   FROM hr.employees
   WHERE employee_id > 200;
```

```
-------------------------------------------------------------------------------------
| Id  | Operation         | Name           | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |                |     6 |    24 |     1   (0)| 00:00:01 |
|*  1 |  INDEX RANGE SCAN | EMP_EMP_ID_PK  |     6 |    24 |     1   (0)| 00:00:01 |
-------------------------------------------------------------------------------------

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

   1 - access("EMPLOYEE_ID">200)
```

EXPLAIN PLAN FOR
SELECT /*+ NO_INDEX(employees emp_emp_id_pk) */ employee_id
FROM hr.employees
WHERE employee_id > 200;

```
-------------------------------------------------------------------------------
| Id  | Operation          | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |           |     6 |    24 |     3   (0)| 00:00:01 |
|*  1 |  TABLE ACCESS FULL | EMPLOYEES |     6 |    24 |     3   (0)| 00:00:01 |
-------------------------------------------------------------------------------

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

   1 - filter("EMPLOYEE_ID">200)
```