

# AMPLIACIÓN de BASES DE DATOS

(Profesor : Héctor Gómez Gauchía)

## Práctica 3: Optimizacion de Consultas

*Fecha entrega:* **24-04-2014 después de la clase de Laboratorio**

*Resultado:* Los ficheros indicados en cada apartado, todos en un directorio comprimido.

*Modo de entrega:* Subir un zip con el directorio al Campus Virtual.

### APARTADO 1.A

Se desea hacer un estudio comparativo de varias consultas , algunas vistas en la Teoría. Para ello, generamos el *Plan de Ejecución* de las consultas de Oracle, que se genera con el comando EXPLAIN.

→ **Entrega** un fichero *prac31.docx* con lo siguiente:

- Una tabla con los datos de los planes para las consultas siguientes, una fila por consulta.
  - Las columnas son: Coste (total), Num. Filas(total), Num. Operaciones.
  - Qué observas en la relación de cada operación, su Coste y sus Filas utilizadas?
- Para cada consulta:
  - El resultado con las operaciones del explain plan
  - Dibuja el árbol del Plan de Ejecución: incluye el núm. De operación y su nombre en cada nodo
  - Explica brevemente porqué ha ejecutado esas operaciones y no otras.
  - Responde a las *preguntas que hay después de las ejecuciones*.

Para ejecutarlas sigue estos pasos: (ver NOTA al final del enunciado sobre ejecutar en sql Developer)

**1)-** Se necesita crear una tabla donde Oracle almacena los resultados de las 'explicaciones'. Debe *estar ya creada*. Si al hacer los siguientes pasos no la encuentra, ejecuta este archivo: *utlxplan.sql*

**2)-** Para tener una mascara de edición que formatee el resultado del plan , **ejecuta:** (sale vacía ahora)  
*MASCPLAN13.sql*

**3)- Ejecuta** las siguientes explicaciones siguiendo estos pasos:

(las tres instrucciones para cada consulta : borrar lo que había, explicar consulta, ver el resultado)

```
-- CONSULTA 1 --
delete plan_table;                                /* primero borra las tuplas */

EXPLAIN PLAN
  INTO plan_table
  FOR (select * from cliente) union (select * from moroso);
→ Ejecutar MASCPLAN13.sql

-- CONSULTA 2 --
delete plan_table;    /* borra las tuplas de explicación anterior*/

EXPLAIN PLAN
  INTO plan_table
  FOR (select * from cliente where DNI < '00000005') union
      (select * from moroso where NombreC = 'Client E');
→Ejecutar MASCPLAN13.sql

-- CONSULTA 3 -- anidados ----
delete plan_table;

EXPLAIN PLAN
  INTO plan_table
  FOR select * from cliente where DNI in
      (select DNI from moroso where NombreC = 'Client E');
Ejecutar MASCPLAN13.sql
```

```
-- CONSULTA 4 - que pasa en esta consulta? ----
delete plan_table;

EXPLAIN PLAN
SET STATEMENT_ID= 'mipruxx'
INTO plan_table
FOR (select * from cliente) union (select * from invierte);
→ Ejecutar MASCPPLAN13.sql

-- CONSULTA 5 - compara con la 3 ----
delete plan_table;

EXPLAIN PLAN
SET STATEMENT_ID= 'mipru05'
INTO plan_table
FOR (select * from cliente where dni in
      (select dni from invierte));
→ Ejecutar MASCPPLAN13.sql
-----
```

→ *Contestar a las siguientes preguntas sobre los resultados anteriores:*

- Porqué en -- CONSULTA 2 -- el acceso será más eficiente que en -- CONSULTA 1-- ?
- Porqué en -- CONSULTA 2 -- no accede por el índice a moroso ?
- Porqué en -- CONSULTA 3 -- no accede por el índice a CLIENTE ?
- Porqué en -- CONSULTA 3 -- sí accede por el índice a MOROSO ?

*Entregar: ver al principio del apartado 1.*

#### APARTADO 1.B- (incluye lo solicitado en el fichero de entrega prac31.docx)

→ *Ejecutar las tres instrucciones de la explicación de la consulta: borrar lo que había, explicar consulta y ver el resultado, para:*

- a)-** Hazlo de las dos formas, sin 'distinct' y con 'distinct':
  - Incluye en la entrega los resultados del explain de ambas
  - Qué diferencias hay?

```
select distinct NombreC
from Cliente, Compras, Invierte
where Cliente.DNI = Invierte.DNI and
      Invierte.NombreE = 'Empresa 55' and
      Compras.DNI = Cliente.DNI and
      Compras.Importe >1000;
```

- b)-** Escribe la explicación de otra consulta *equivalente* a la de **a)** usando una subconsulta anidada.

→ *Compara los dos resultados haciendo lo siguiente:*

- c)-** Dibuja los dos árboles del plan de ejecución, para **a)** con *distinct* y para **b)**
- d)-** Explica cual sería más eficiente sin tener en cuenta el coste, basándote en esta información:
  - *Criterios:* Cuántas operaciones y de qué tipo son
  - *DATOS a tener en cuenta:* Hay muchas menos filas en Cliente que en Inversiones
  - Nested loop es más costoso que Filter.
  - Tipos de acceso: mas costoso secuencial (full) que con índices (dos tipos: más costoso *unique* que *range*).
- e)-** Explicar cual es más eficiente si ahora tienes en cuenta el coste y la filas usadas (cardinalidad)?
- f) -** Que instrucción ha ejecutado oracle para hacer el 'distinct'?

**g) Ejecutar en *sql Developer*** con "Rastreo Automático (F6)" y con "Explicación del plan (F10)" la misma consultas y compara los resultados. Qué diferencias hay? Son ambas funciones iguales?

*Entregar: incluye en el fichero prac31.docx lo pedido en cada sección*

## APARTADO 2.-

- a)- Se desea crear una tabla diccionario DICCION, que tenga los siguientes atributos:  
PalID , será como máximo de 20 caracteres. Identifica la palabra.  
Descripción , de 50 caracteres  
PadreID, de 20 char. Representa un concepto más genérico que PalID, en las filas insertadas en b)- se ve que 'select compuesta' es el PadreID de 'select jerarquica' y de 'select correlativa'
- b)- Se desean incluir las siguientes filas: (estas comillas son del word, no válidas en oracle)
- ```
( 'select jerarquica', 'estructura tabla en arbol', 'select compuesta' );  
( 'fecha sistema', 'es la fecha que tiene el ordenador', 'fecha' );  
( 'fecha', 'tipo de dato , en oracle : DATE', 'nada' );  
( 'select compuesta', 'consultas con varias partes', 'select' );  
( 'select simple', 'consultas con una sola instruccion', 'select' );  
( 'select', 'hacer consulta', 'nada' );  
( 'sql', 'lenguaje de consultas estructuradas', 'nada' );  
( 'select correlativa', 'coordina resultado subconsulta', 'select compuesta' );
```
- c)- Hacer una *consulta jerárquica* conectada por PalID y PadreID que empiece con la palabra 'select' (es como la de 'masplan.sql')
- d)- Insertar una fila con valores PalID = 'select anidada', descripción = 'consulta dentro de consulta' y PadreID = 'select compuesta'. Lo importante de esta inserción es que se quiere hacer solo en el caso que el padre exista, es decir solo hacerla en caso de que una consulta de PalID='select compuesta' devuelve algo. Si no devuelve nada no se debe crear.

*Entregar: Fichero prac32.sql con las instrucciones necesarias en sql.  
Si has tenido que hacer algo diferente a lo pedido, explícalo brevemente.*

## APARTADO 3

Vamos a probar con tablas con más filas: Hector.pelisahora y Hector.pelishist

- Hacer los planes de ejecución y justificar porque se usan esas operaciones para las dos consultas siguientes:

- a) (select \* from hector.pelisahora where ID in  
(select ID from hector.pelishist where ID like '%1%'));
- b) select \* from hector.pelisahora where ID in  
(select descripcion from hector.pelishist where ID like '%1%'));

*Entregar: Fichero prac33.docx con la explicación*

## APARTADO 4 Sobre estadísticas en Oracle: se describirá el enunciado más adelante

**NOTA sobre ejecución en SQL Developer:** Hay dos modos de ejecutar *MASPLAN13.sql*:  
- consulta(icono: triángulo) resultado sale en forma de tabla

- script (icon: triángulo dentro de rectángulo con rayas) sale en forma de texto. Así se puede cortar y pegar o salvar como fichero de texto.