AMPLIACIÓN de BASES DE DATOS

(Profesor: Héctor Gómez Gauchía)

Práctica 4 SEMANA 9 - Optimización de Consultas

***Resultado*:**

- En un archivo reune Instrucciones, Resultados de las ejecuciones de los árboles y tus respuestas a las preguntas.
- Haz lista de dudas concretas que te queden sobre tus respuestas para consultar con el profesor

***Modo de entrega**:*

- Sube un archivo PDF a la tarea del CV: *Prác4Semana9*
- Los conceptos de esta práctica se evalúan en una presentación oral con todas las prácticas, después de terminar el perido de docencia online.

APARTADO 3

Preparación:

- Vamos a usar más volumen de datos para que los valores estadísticos tengan más sentido:
  -> Si el tablespace se te llena mira la nota al final del apartado **.**

- Importa las tablas PELISAHORA y PELISHIST en formato excel de la carpeta /BDpelis/ :
  PASOS: en SQLDeveloper, sobre carpeta *Tablas*: con B.dcho + importar Datos
  - *Vista previa*: Deja la cabecera y resto igual. Escoger el fichero. (avanzar con <siguiente>)
  - *Método de importación*: todo como está, salvo el nombre de la tabla, usa el mismo del fichero
  - *Selección de columnas*: dejarlas todas tal cual,
    - Conserva el mismo nombre del atributo/ columna.
  - *Definición de columnas*: Asignar tipo y tamaño a cada columna
    - El tipo que escojas debe ser en las dos tablas igual: puedes dejar los valores por defecto excepto cuando en las dos tablas NO son iguales

- Crea una Clave primaria en cada tabla para el atributo ID con ALTER TABLE (asígnale un nombre, como PK_pelisactual)

- Crear índice único B+ con CREATE INDEX sobre el Título para agilizar las búsquedas y las uniones.

→ Ejecutar los tres pasos del apartado anterior (borrar lo que había, explicar consulta y ver el resultado con *MASCPLAN-14.SQL*), para hacer una explicación de estas consultas *y completa cada apartado*:

```
-- CONSULTA C1 -

select PELISAHORA.ID
from PELISAHORA, PELISHIST
where PELISAHORA.ID = PELISHIST.ID;

-- CONSULTA C2 --
select PELISAHORA.DESCRIPCION
from PELISAHORA, PELISHIST
where PELISAHORA.DESCRIPCION = PELISHIST.DESCRIPCION;

-- CONSULTA C3 --
select PELISAHORA.TITULO
from PELISAHORA, PELISHIST
where PELISAHORA.TITULO = PELISHIST.TITULO;

-- CONSULTA C4 --
select PELISAHORA.TITULO
from PELISAHORA
where PELISAHORA.TITULO in (select PELISHIST.TITULO from PELISHIST);
```

```
-- CONSULTA C5 --
select PELISAHORA.TITULO
from PELISAHORA
where PELISAHORA.TITULO in
       (select PELISHIST.TITULO from PELISHIST
        where PELISAHORA.TITULO = PELISHIST.TITULO);
```

a)- *Para comparar la eficiencia en coste y filas manejadas* crea esta tabla *en un documento* con los datos de los planes para las consultas indicadas, una fila por consulta.
NOTA: El *coste total* de un plan es el coste de la operación raíz, porque indica el coste acumulado de sus hijos (cada hijo tiene su coste particular). No pasa igual con el total de las filas y bytes utilizados, que es la suma de todas las operaciones.
NOTA: Cómo saber si tenemos *una buena estimación*: Una medida simple es comprobar que el Num. Filas (Cardinality) es igual o muy parecido a las filas reales que tenemos en las tablas. Si no es así, es que la estimación está fallando y no es fiable.
NOTA: si alguna operación no está en la teoría, puedes buscar en internet, ej.: Oracle index fast full.

CONSULTA	**Coste (total)**	**Num. Filas (total)**	**Num. Operaciones**	**Bytes (total)**
C1				
C2				
. . . .				

a.1) Sobre C1: ¿Porqué no accede a la tabla PELISHIST?

a.2) Sobre C1: ¿Cuál es el criterio principal para escoger un plan de ejecución u otro?

a.3) Compara C1 y C2: ¿Qué problema hay con los índices para que el coste sea tal alto en C2?

a.4) Compara C2 y C3: ¿Porqué, en C2, accede a las tablas completas?

a.5) Compara C2 y C3: ¿Porqué, en C2, el coste es tal alto?

a.6) Compara C2 y C3: ¿En C2, porqué el Hash Join solo usa 142 filas si ha leído las 521 filas de pelishist?

a.7) Compara C3 y C4: ¿Porqué ambas NO hacen las mismas operaciones de bajo nivel?

a.8) Compara C4 y C5: ¿ Porqué ambas SI hacen las mismas operaciones de bajo nivel?

b)- *Explica* cuál de las consultas es más eficiente si tienes en cuenta:
b.1) Sólo el *coste* y las *filas usadas*.
b.2) Sólo el *coste* y los *bytes usados.*

c)- *Crear diferentes tipos de índices dependiendo de las situaciones:*

c.1 Queremos acceder rápidamente por el género. Asume que tiene pocos valores distintos. - ¿Qué índice conviene crear? - Créalo. - Obtener el plan sobre esta consulta a ver si usa el índice: Queremos la pelis de la tabla pelisahora que sean del género Drama y tengan más de un 35% de drama.

c.2 Vamos a hacer muchas consultas como la del ejemplo, así que queremos agilizarlas mediante un índice. EJ.:

```
select titulo, round(Drama)
from pelisahora
where round(Drama) = 50 ;
```

Para responder sigue estos pasos:

1.- Obtener el plan de ejecución con esa consulta

2.- Crear el índice adecuado

3.- Obtener el plan de ejecución y comprobar que, ahora, usa dicho índice.

4.- Genera el plan de la consulta con la condición "> 10". ¿Qué diferencia hay en el plan? ¿porqué ?

d)- (OPCIONAL) *Crea consultas diferentes sobre las tablas de películas, que provoquen en Oracle las operaciones siguientes*: (consulta en teoría qué características las provocan)
d.1) Full table scan, **d.2)** Index unique scan, **d.3)** Full Index scan y **d.4)** Cartesian join.

Crea otras consultas que provoquen en Oracle (ver Reglas de Optimización) que:

d.5) - NO use los índices

d.5) - SÍ use índices

e)- Prueba otro modo de conseguir los datos del plan usando el paquete XPLAN (basado en estimaciones, como el explain plan) :

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

NOTA: Para ampliar tu tablespace: (desde el usuario *ADMINUSER*)

```
alter DATABASE DATAFILE 'ESPACIOU-tuyo' autoextend ON
next 521k maxsize 25M;
```

APARTADO 4.- Consultas jerárquicas

(Entregar las instrucciones necesarias en sql y los resultado de las consulta)

Queremos hacer un diccionario que relacione jerárquicamente los conceptos. Ej.: el concepto "select" con todos los tipos de select. Para ello hacer lo siguiente:

a)- 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)- 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 'mascplanXX.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. (es una "inserción condicionada").

e) Ejecuta de nuevo la *consulta jerárquica* y comprueba que sale correctamente.