

**Ficheros y Bases de Datos**  
**Curso 2007-8.**  
**Ingeniería Técnica de Informática, UCM**  
**Segundo parcial. Junio.**

Nombre:

Se debe entregar esta hoja

- 1) (3 puntos) Considérese un fichero secuencial indexado denso multinivel (dos niveles) con los campos A:Byte(2), B:String(10), C:String(30), cuya clave es el campo A, que admite duplicados. El formato de los caracteres es ASCII y las cadenas de caracteres son de longitud fija. El primer nivel tiene 1000 entradas, el segundo nivel contiene, en media, 200 entradas por cada entrada del índice de primer nivel y 100 entradas en el fichero de datos por cada valor de la clave. En la unidad de disco se tiene que las direcciones de bloque son de 6 bytes, la frecuencia de rotación del disco es de 10800 rpm, el tiempo de búsqueda es de 12 ms, hay 512 sectores por pista, 4000 pistas por cara, dos sectores por bloque, 1024 bytes por sector, 8 platos de doble cara y las escrituras necesitan el doble de tiempo que las lecturas. Se dispone de asignación enlazada y mapas de bits de existencia para los bloques. Se pide:
- a) (0,5) Calcular el tamaño del bloque y de la unidad de disco en bytes, y los tiempos de lectura y escritura de bloque.

**Solución:**

Tamaño del bloque: 25 sectores/bloque\*1.024 bytes/sector = 2.048 bytes.

Tamaño del disco: 8 platos \* 2 caras/plato \* 4000 pistas/cara \* 512 sectores/pista \*

1.024 bytes/sector =  $2^3+2^1+2^2*1.000+2^9*2^{10}=2^{3+1+9+10}*1.000=2^{25}*1.000 = 32.000 \text{ MB} \approx 32 \text{ GB}$ .

Tiempo medio operación E/S=Tiempo de búsqueda + Tiempo de latencia + Tiempo de transmisión

Tiempo de búsqueda=12ms

Tiempo de latencia= $\frac{1}{2}$ \*Inversa de la frecuencia =  $\frac{1}{2} \frac{1}{10.800 \frac{1}{\text{min}} \frac{1 \text{ min}}{60.000 \text{ ms}}} = 2,78 \text{ ms}$

Tiempo de transmisión= $\frac{1}{512}$  \* Tiempo de rotación (el doble del tiempo de latencia) \* 2

(2 sectores/bloque)=0,022ms

Tiempo medio operación lectura=1+2,78+0,022=14,802 ms

Tiempo medio operación escritura=2\* Tiempo medio operación lectura =29.604 ms

- b) (1,0) Describir la estructura y tamaño de la información almacenada en los ficheros de datos y de índice.

**Solución:**

Primer nivel del fichero de índices: 1.000 registros <A (2 bytes), Dirección bloque (6 bytes)>. Tamaño =  $8*1.000 = 8.000$  bytes.

Segundo nivel del fichero de índices: 1.000\*200 registros <A (2 bytes), Dirección bloque (6 bytes)>. Tamaño = 8\*1.000\*200= 1.600.000 bytes.  
 Fichero de datos: 200.000\*100 registros <A (2 bytes) , B (10 bytes) , C (30 bytes)>.  
 Tamaño = 42\*200.000\*100= 8,4\*10<sup>8</sup> bytes.

- c) (1,0) Calcular el factor de bloqueo en los dos niveles del fichero de índice y en el fichero de datos. Asumir que cada nivel del índice está implementado en un fichero independiente.

**Solución:**

*Fichero de índice:*

La estructura de los dos niveles del índice es igual: un campo para la clave (A, de 2 bytes), otro para la dirección de bloque correspondiente a la clave (6 bytes), mapa de bits de existencia y dirección de bloque por la asignación enlazada. Por tanto, el factor de bloqueo N se calcula como:

$$N \times (2+6) \times 8 + N + 6 \times 8 \leq 2.048 \times 8$$

En esta inecuación se usa el bit como unidad y por ello todos los sumandos se multiplican por 8, salvo el mapa de bits, que ya va expresado en esta unidad.

$$N \leq 251,32$$

Por lo tanto, escogemos el entero N = 251.

*Fichero de datos:*

La estructura del fichero de datos es: campos de datos (A, de 2 bytes, B de 10 y C de 30), mapa de bits de existencia y dirección de bloque por la asignación enlazada. Por tanto, el factor de bloqueo N se calcula como:

$$N \times (2+10+30) \times 8 + N + 6 \times 8 \leq 2.048 \times 8$$

$$N \leq 48,47$$

Por lo tanto, escogemos el entero N = 48.

- d) (0,5) ¿Cuál es el número medio de accesos de entrada/salida necesario para la lectura de un registro del fichero de datos con un determinado valor clave, con y sin índice?

**Solución:**

*Con índice:*

$$\text{Primer nivel: } \left\lceil \frac{1}{2} \times \frac{1.000 \text{registros}}{251} \right\rceil = 2 \text{ bloques}$$

$$\text{Segundo nivel: } \left\lceil \frac{1}{2} \times \frac{200 \text{registros}}{251} \right\rceil = 1 \text{ bloque}$$

Fichero de datos = 1 bloque

Total = 2+1+1 = 4 bloques = 4 operaciones de lectura.

*Sin índice:*

$$\left\lceil \frac{1}{2} \times \frac{20.000.000 \text{registros}}{48} \right\rceil = 208.334 \text{ bloques}$$

Total = 208.334 bloques = 208.334 operaciones de lectura.

2) (2 puntos) Dada la siguiente planificación decir si es secuenciable (1 punto) y, si lo es, determinar una planificación secuencial equivalente (1 punto).

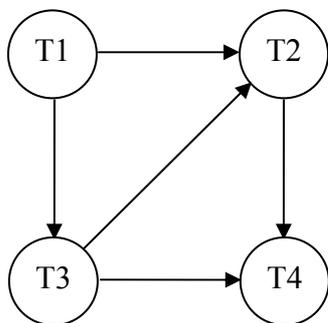
T1: LOCK A  
T3: LOCK B  
T1: UNLOCK A  
T1: LOCK C  
T3: UNLOCK B  
T2: LOCK B  
T3: LOCK A  
T2: UNLOCK B  
T1: UNLOCK C  
T4: LOCK B  
T3: UNLOCK A  
T2: LOCK C  
T4: LOCK A  
T4: UNLOCK B  
T2: UNLOCK C  
T4: UNLOCK A

**Solución:**

Para determinar si la planificación es secuenciable recurrimos al test de secuencialidad. Construimos el grafo:

T1: UNLOCK A -> T3: LOCK A  
T3: UNLOCK B -> T2: LOCK B  
T2: UNLOCK B -> T4: LOCK B  
T1: UNLOCK C -> T2: LOCK C  
T3: UNLOCK A -> T4: LOCK A

Estas dependencias forman el grafo:



Aplicando el algoritmo para generar planificaciones secuenciales equivalentes, se obtiene la única posible:

T1 -> T3 -> T2 -> T4

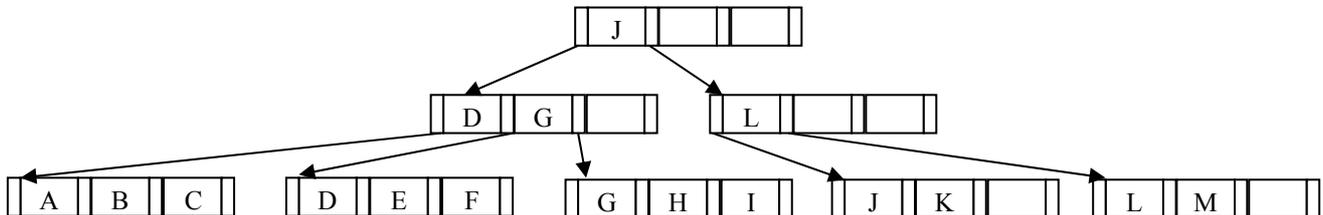
- 3) (2 puntos) Dado un índice B+ donde se almacenan 13 claves (primeras 13 letras) y con  $n=4$ .
- a) (0,75) Crear un árbol B+ con estos valores, indicando las restricciones que se deben cumplir.

**Solución:**

Se necesitan al menos  $\left\lceil \frac{13}{3} \right\rceil = 5$

Restricciones:

- Nodo raíz entre 1 y 4 hijos.
- Nodo interno entre  $\left\lceil \frac{4}{2} \right\rceil = 2$  y 4 hijos.
- Nodo hoja entre  $\left\lceil \frac{4-1}{2} \right\rceil = 2$  y  $4-1=3$  valores.



- b) (0,5) ¿Cuál es el número de accesos medio para localizar un registro del fichero de datos apuntado por el índice anterior si el índice es primario? ¿y si es secundario?

**Solución:**

Índice primario: 3 niveles del índice + 1 acceso al bloque de datos = 4

Índice secundario: igual que antes pero 1 acceso extra a cajón si el fichero de datos admite repetidos.

- c) (0,75) Si el tamaño del bloque es 1024 bytes, el tamaño de la clave es 128 bytes, las direcciones de bloque son de 4 bytes y se utilizan mapas de bits de existencia. ¿Cuál sería el valor de  $n$  para un árbol B+ donde cada nodo ocupara un bloque?

**Solución:**

Para calcular el grado máximo de salida hay que analizar cuántos valores de la clave caben en un bloque. Estudiamos su estructura:

$N$  punteros a bloques (4 bytes),  $N-1$  valores de la clave (128 bytes) y  $N-1$  bits para el mapa de bits de existencia:

$$N \cdot 4 \cdot 8 + (N-1) \cdot 128 \cdot 8 + N-1 \leq 1.024 \cdot 8$$

$$N \cdot 32 + N \cdot 1.024 - 1024 + N - 1 \leq 8.192$$

$$N \cdot (32 + 1.024 + 1) \leq 8.192 + 1.024 + 1$$

$$N \leq \lfloor 8,7 \rfloor = 8, \text{ que es el grado de salida máximo}$$

- 4) (3 puntos) Dadas las tablas VENTAS\_TOT, VENTAS y MOTOS descritas más adelante, codifica:
- a) (0,75 puntos) Escribe el código SQL para crear las tres tablas.
    - o El tipo de datos para las fechas es el tipo DATE de Oracle, para el resto puedes utilizar el tipo de datos que consideres oportuno.
    - o Las sentencias DDL deben incluir al menos las sentencias de gestión de integridad referencial
    - o Incluye las restricciones adicionales que consideres oportunas.

**Solución:**

```
CREATE TABLE VENTAS_TOT (
  dnicliente char(9),
  idcomercial integer,
  numfact integer primary key,
  fecha date,
  marca varchar(10),
  modelo varchar(10),
  importe number(9,2) check ( importe > 0.0 ),
  tipoiva integer check( tipoiva IN (7,16) )
);
```

```
CREATE TABLE MOTOS (
  marca varchar(10),
  modelo varchar(10),
  CC integer,
  año date,
  existencias integer check( existencias >= 0),
  PRIMARY KEY (marca, modelo)
);
```

```
CREATE TABLE VENTAS (
  numfact integer PRIMARY KEY,
  dnicliente char(9),
  fecha date,
  marca varchar(10),
  modelo varchar(10),
  importe number(9,2) check( importe > 0.0 ),
  FOREIGN KEY (marca,modelo) REFERENCES Motos(marca,modelo)
);
```

- b) (0,75 puntos) Codifica un bloque PL/SQL anónimo que pase las compras almacenadas en la tabla VENTAS\_TOT a la tabla VENTAS, tomando sólo los campos necesarios para ésta última. Utiliza un cursor. Suponemos que la tabla VENTAS ya existe y contiene datos.

**Solución:**

```
-- Bloque anonimo con cursor explicito
DECLARE
  CURSOR c_ej1 IS
    SELECT DNICLIENTE,NUMFACT,FECHA,MARCA,MODELO,IMPORTE
    FROM VENTAS_TOT;
BEGIN
  -- Estructura "cursor for loop" para la insercion de datos
  FOR c_reg IN c_ej1 LOOP
```

```

INSERT INTO VENTAS
VALUES (c_reg.DNICLIENTE,
        c_reg.NUMFACT,
        c_reg.FECHA,
        c_reg.MARCA,
        c_reg.MODELO,
        c_reg.IMPORTE);
END LOOP;
END;
/

```

- c) (1,5 puntos) Escribe también el código de un disparador asociado a la tabla VENTAS que haga lo siguiente:
- o Cada vez que se realice la venta de una moto, reste una unidad a las existencias de motocicletas del modelo correspondiente.
  - o En caso de que esas existencias sean cero antes de restarle la unidad, el disparador debe mostrar un mensaje advirtiéndolo y no realizar la resta. Si además las existencias de todos los modelos de motos son cero, se debe mostrar un mensaje que diga “MOTOS AGOTADAS”.
  - o En cualquier caso, el disparador siempre debe mostrar un mensaje en pantalla con la marca y el modelo comprados acompañados del número de unidades de dicha motocicleta vendidas en el último año (este dato se consultará de VENTAS\_TOT).

```

VENTAS_TOT (DNICLIENTE, IDCOMERCIAL, NUMFACT,
            FECHA, MARCA, MODELO, IMPORTE, TIPOIVA)

```

```

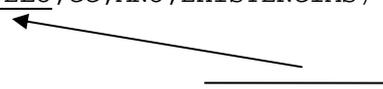
MOTOS (MARCA, MODELO, CC, AÑO, EXISTENCIAS)

```

```

VENTAS (NUMFACT, DNICLIENTE, FECHA, MARCA, MODELO, IMPORTE)

```



### Solución:

```

-- Disparador asociado a la tabla VENTAS
CREATE OR REPLACE TRIGGER resta_moto AFTER INSERT ON VENTAS
FOR EACH ROW
DECLARE
ant_exist MOTOS.EXISTENCIAS%TYPE; -- "Antiguas" existencias
unidades INT;                    -- Unidades vendidas.
quedan INT;                      -- Para ver si quedan motos.
BEGIN
-- Comprobamos las antiguas existencias (cursor implícito)
SELECT EXISTENCIAS INTO ant_exist
FROM MOTOS
WHERE MARCA = :new.MARCA AND MODELO = :new.MODELO;
IF ant_exist = 0 THEN
-- Mensaje
DBMS_OUTPUT.PUT_LINE('La ' || :new.MARCA || ', ' || :new.MODELO ||
' está agotada');
-- Y vemos si quedan existencias de las demás.
SELECT SUM(EXISTENCIAS) INTO quedan FROM MOTOS;
IF quedan = 0 THEN
-- No quedan motos

```

```

        DBMS_OUTPUT.PUT_LINE('MOTOS AGOTADAS');
    END IF;
ELSE
    -- Actualizacion
    UPDATE MOTOS
        SET EXISTENCIAS = EXISTENCIAS - 1
        WHERE MARCA = :new.MARCA AND MODELO = :new.MODELO;
END IF;
-- Mensaje con el total de unidades
SELECT COUNT(*) INTO unidades FROM VENTAS_TOT
WHERE MARCA = :new.MARCA AND MODELO = :new.MODELO AND
        FECHA > sysdate - 365;
DBMS_OUTPUT.PUT_LINE('Unidades vendidas de '||:new.MARCA||
        ' '||:new.MODELO||': '||unidades);
END;
/

```