Archivos Hash con resolusión por Cubetas

Andrés Arcia

Para crear un archivo con resolución por cubetas, hay que crear todos los espacios que podrían ocupar los futuros registros.

Algunas variables a considerar son:
tam_arch_maestro: Tamaño del archivo maestro, en número de cubetas.
num_reg_cubetas: Número de registros por cubeta.

Registro_Maestro
Tipo_Dato data
bool disp
bool borrado

crear_archivo()
0. Preparar Registro "R" (disp = true, borrado = false)
1. Abrir Maestro
2. R.P. i =0 hasta tam_arch_maestro-1
    2.1. R.P. j=0 hasta num_reg_cubetas-1
        2.1.1. ESCRIBIR (Maestro, R, i*num_reg_cubetas + j)
        2.1.2. // También pudo haber sido un escribir al final del archivo

insertar(Registro_Maestro Rins)

```
Abrir(maestro)
Rins.disp = falso
Rins.borrado = falso
pos = hash(Rins.clave)
Apartar un arreglo de Registro_Maestro de tamaño num_reg_cubetas de nombre blq_regs
LEER_BLQ(maestro, pos, blq_regs, num_reg_cubetas)
i=0
R.M. (blq_regs[i].disp = falso  ^  i < num_reg_cubetas)
        i++
Si (i = num_reg_cubetas)
        // Arrojar una excepción indicando que no hay más espacio
        // Activar mecanismo de desborde: por encadenamiento, sondeo lineal, encadenamiento con
        // nodo tonto.
SI NO
        ESCRIBIR(maestro, pos*num_reg_cubetas + i, Rins)
cerrar (maestro)
```

buscar(clave)

Durante la búsqueda de una clave, puede ser que exista un registro eliminado en medio de un par de registros con información activa. Este registro eliminado tiene el campo **borrado** en el valor verdadero, indicando que después de él es posible que existan más registros con información.

```
Abrir maestro
pos = hash(clave)
Apartar un arreglo de Registro_Maestro de tamaño num_reg_cubetas de nombre blq_regs
LEER_BLQ(maestro, pos, blq_regs, num_reg_cubetas)
i=0
R.M. (i < num_reg_cubetas  ^ (blq_regs[i].borrado = verdadero  v  blq_regs[i].disp = falso))
        Si (clave = blq_regs[i].clave)
                Si (blq_regs[i].borrado != verdadero)
                        retornar blq_regs[i]
                SINO
                        retornar NO_SE_ENCONTRO
        SI NO
                i++
// Aqui puede buscarse en el desborde
Retornar NO_SE_ENCONTRO
```

eliminar(clave)

Esta rutina de eliminar, hace que, si se encuentra la clave que se desea eliminar, al campo **borrado** del registro se le asigna el valor verdadero. Con ésta acción, ahorramos tiempo en la eliminación, pues siempre se hacen la misma cantidad de operaciones para eliminar un registro.

```
Abrir maestro
pos = hash(clave)
Apartar un arreglo de Registro_Maestro de tamaño num_reg_cubetas de nombre blq_regs
LEER_BLQ(maestro, pos, blq_regs, num_reg_cubetas)
i=0
R.M. (i < num_reg_cubetas  ^ (blq_regs[i].borrado = verdadero  v  blq_regs[i].disp = falso))
        Si (clave = blq_regs[i].clave)
                Si (blq_regs[i].borrado = falso)
                        blq_regs[i].disp = verdadero
                        blq_regs[i].borrado = verdadero
                        ESCRIBIR(maestro, blq_regs[i],  pos * num_reg_cubetas + i)
                        retornar EXITO
```

```
            SI NO
                retornar NO_SE_ENCONTRO
        SI NO
            i++
// Aqui puede buscarse en el desborde
// También hay que chequear que el último registro del maestro apunte al primero de la cadena
// de desborde en el archivo desborde.

Retornar NO_SE_ENCONTRO
```

Mezcla de Archivos

Caso 1:

Un par de archivos ordenados: maestro_a y maestro_b deben fusionarse en un tercer archivo maestro. Se supone que ambos archivos estan ordenados en forma creciente y el operador '<', '>' y '=' están bien definidos para el tipo registro.

```
Abrir maestro_a
Abrir maestro_b
Abrir maestro
LEER(maestro_a, reg_a)
LEER(maestro_b, reg_b)
R.M. (maestro_a != EOF  ^ maestro_b != EOF)
        Si (reg_a  <  reg_b)
            ESCRIBIR(maestro, reg_a)
            reg_tmp = reg_a
            R.M. (reg_tmp == reg_a)
                LEER(maestro_a, reg_a)
        SI NO
            Si (reg_a > reg_b)
                ESCRIBIR(maestro, reg_b)
                reg_tmp = reg_b
                R.M. (reg_tmp == reg_b)
                    LEER(maestro_b, reg_b)
            SI NO
                ESCRIBIR(maestro, reg_b)
                R.M. (reg_a == reg_b   ^  maestro_b != EOF)
                    LEER(maestro_b, reg_b)
                reg_tmp = reg_a
                R.M. (reg_a == reg_tmp  ^  maestro_a != EOF)
                    LEER(maestro_a, reg_a)
```

```
Si (maestro_a = EOF  ^  maestro_b = EOF)
        Si (reg_a < reg_b)
                Si (reg_a != reg_tmp)
                        ESCRIBIR(maestro, reg_a)
                Si (reg_b != reg_tmp)
                        ESCRIBIR(maestro, reg_b)
        SI NO
                Si (reg_b < reg_a)
                        Si (reg_b != reg_tmp)
                                ESCRIBIR(maestro, reg_b)
                        Si (reg_a != reg_tmp)
                                ESCRIBIR(maestro, reg_a)
                SI NO
                        Si (reg_a != reg_tmp)
                                ESCRIBIR(maestro, reg_a)
SI NO
Si maestro_a = EOF
        R.M. maestro_b != EOF
                LEER(maestro_b, reg_b)
                Si (reg_a < reg_b)
                        ESCRIBIR(maestro, reg_a)
                SI NO
                        ESCRIBIR(maestro, reg_b)
SI NO
        Si maestro_b = EOF
                ESCRIBIR(maestro, reg_b)
                R.M. maestro_a != EOF
                        LEER(maestro_a, reg_a)
                        Si (reg_a < reg_b)
                                ESCRIBIR(maestro, reg_a)
                        SI NO
                                ESCRIBIR(maestro, reg_b)
```

Caso 2:

Un archivo maestro y un archivo desborde según el método de encadenamiento al desborde,

```
Abrir maestro
Abrir temporal
Abrir desborde
R.M. (maestro != EOF)
        LEER_SECUENCIAL(maestro, reg)
        ESCRIBIR_AL_FINAL(temporal, reg)
        Si (reg.prox != -1)
                R.M. (reg.prox != -1)
                        LEER_DIRECTO(desborde, reg.prox, reg) // archivo, posicion y variable deposit
                        ESCRIBIR_AL_FINAL(temporal, reg)
cerrar maestro, desborde, temporal
eliminar maestro
eliminar desborde
renombrar temporal como maestro
```