

Bajas

CONCEPTOS DE BASES DE DATOS

TIPOS DE BAJAS

Se puede eliminar información de un archivo mediante:

- ▶ Baja Física: borrar efectivamente la información del archivo.
- ▶ Baja lógica: se realiza algún tipo de marca en la información indicando que la misma es obsoleta (está borrada). Para evitar desaprovechamiento del disco: técnica de reutilización de espacio.

ELIMINAR INFORMACIÓN

Bajas Físicas:

- ▶ Nuevo archivo.
- ▶ Reacomodar archivo.(Copiar el último registro al lugar que desea borrar la información y luego truncar el archivo)

Bajas Lógicas:

El elemento a borrar se lo distingue de algún modo como borrado.

Ventaja: Performance, se busca y se marca (lecturas hasta el elemento y una escritura).

Desventaja: Espacio en disco, no se recupera el espacio borrado por ende el archivo crece continuamente.

BAJA FÍSICA

```
Procedure BajaFisica(var a:archivo; apellido,nombre: String);
Var
    posBorrar:integer; rp,aux:per; apellido, nombre: st20;
begin
    reset(a);
    readln(apellido); readln(nombre);
    rp.ape= 'zzz'; rp.nom='zzz';
    while (not eof(a) and ((rp.ape != apellido) and (rp.nom!=nombre)))
        then
            read(a,rp);
    if(rp.ape = apellido) and (rp.nom=nombre)then begin
        posBorrar:=filepos(a)-1;
        seek(a,filesize(a)-1);
        read(a,aux);
        seek(a, posBorrar);
        write(a,aux);
        seek(a,filesize(a)-1);
        truncate(a);
    end,
    close(a);
end;
Bajas
```

Type

```
persona= record
    ape:st20;
    nom:st20;
    fecNac:longint;
end;
```

archivo=file of persona;

BAJA LÓGICA

```
Procedure BajaLogica(var a: archivo; apellido,nombre: String);
```

```
Var
```

```
    posBorrar:integer; rp:persona;
```

```
    apellido, nombre: st20;
```

```
begin
```

```
    reset(a);
```

```
    readln(apellido); readln(nombre);
```

```
    rp.ape= 'zzz'; rp.nom='zzz';
```

```
    while (not eof(a) and ((rp.ape != apellido) and (rp.nom!=nombre)))
```

```
        then
```

```
            read( a,rp);
```

```
    If(rp.ape = apellido) and (rp.nom=nombre)then begin
```

```
        posBorrar:=filepos(a)-1;    guardo pos a borrar
```

```
        rp.ape="@@"    en el apellido guardo la marca
```

```
        seek(a, posBorrar);    me posiciono
```

```
        write(a,rp);    escribo el registro con la marca en esa pos
```

```
    end;
```

```
    close(a);
```

```
end;
```

```
Bajas
```

Type

```
persona= record
```

```
    ape:st20;
```

```
    nom:st20;
```

```
    fecNac:longint;
```

```
end;
```

```
archivo=file of persona;
```

TÉCNICA PARA RECUPERACIÓN DE ESPACIO

Se debe realizar un programa que permita la creación de un archivo conteniendo razas vacunas o la apertura de uno existente para ver su contenido en pantalla o actualizarlo (altas, bajas o modificaciones de registros).

Se detallarán los procedimientos que realizan altas y bajas de razas.

Las bajas se realizan 'apilando' registros borrados y las altas reutilizando registros borrados. El registro 0 se usa como cabecera de la pila de registros borrados.

Si el registro cabecera tiene valor -1 implica que no hay registros borrados, en cambio si tiene valor n , n un número de registro válido del archivo, implica que el próximo registro a reutilizar es el n .

AGREGAR RAZAS REUTILIZANDO ESPACIO

Type

tRaza= String[50];

tArchRaza= File of tRaza;

Procedure Procesar(var a: tArchRaza);

Var

o: Byte; *{opción}*

cl: Byte; *{contador de lineas}*

r, rb: tRaza; *{raza y raza a buscar}*

sLibre: tRaza; *{string con próximo registro libre}*

nLibre, cod: number; *{numero del próximo registro libre, y
código de error al transformar datos }*

AGREGAR RAZAS REUTILIZANDO ESPACIO

```
begin
  Reset(a); Read(a, sLibre); {lee la cabecera}
  Val(sLibre, nLibre, cod); {convierte de string a number}
  If (nLibre = -1) then
    Seek(a, FileSize(a))
  else begin
    Seek(a, nLibre); Read(a, sLibre); {lee la posición a reutilizar}
    Seek(a, 0); Write(a, sLibre); {Actualiza la cabecera}
    Seek(a, nLibre);
  end;
  WriteLn('Raza Vacuna: ');
  ReadLn(r); {Lee la raza de teclado}
  Write(a, r); {Guarda la raza}
  Close(a)
end;
```


ELIMINAR RAZA REUTILIZANDO ESPACIO

```
begin
  Reset(a);
  Write('Nombre de la raza a dar de baja: ');
  ReadLn(rb); {Raza eliminar}
  Read(a, sLibre) {lee la cabecera}
  r= 'zzz';
  While (not((r=rb) or EoF(a))) Read(a, r); {busca}
  If r=rb then begin {se encuentra la raza}
    nLibre:=FilePos(a)-1;
    Seek(a, nLibre); Write(a, sLibre); {Grabamos el contenido de la cabecera}
    Str(nLibre, sLibre); {Convierte de number a string}
    Seek(a, 0); Write(a, sLibre); {Se actualiza la cabecera}
  end
else Begin {no se encuentra la raza}
  WriteLn; WriteLn('No existe la raza. ');
  Write('Oprima Entrar para continuar...'); ReadLn
end;
Close(a)
```