CONCEPTOS DE BASES DE DATOS

Archivos - Algorítmica clásica

GENERACIÓN DE ARCHIVOS-MERGE

Este algoritmo permite generar un archivo resumiendo información obtenida de múltiples archivos.

- No existe el archivo maestro.
- Se tienen 1 o n detalles.

ACTUALIZACIÓN DE ARCHIVOS-MAESTRO/DETALLE

Este algoritmo permite actualizar la información de un archivo archivo maestro a partir de múltiples detalles. Se puede disponer de 1 o n detalles

El archivo maestro resume información sobre un dominio determinado.

Cada archivo detalle contiene información que genera modificaciones sobre la información almacenada en el archivo maestro.

- *Ejemplo Maestro: Archivo con información del personal de la facultad de ingeniería.*

- Ejemplo Detalle: Archivo con las licencias solicitadas por dicho personal

ARCHIVOS-CORTE DE CONTROL

Este algoritmo permite presentar la información de un archivo en forma organizada de acuerdo a la estructura del archivo origen.

Ejemplo: Se posee un archivo que tiene información sobre productos de una cadena de electrodomésticos. Cada registro contiene la siguiente información: #electrod, tipo de electrodoméstico, marca. nombre, cantidad en stock. Se pide listar la cantidad electrodomesticos por tipo y marca. El archivo está ordenado por tipo de electrodomésticos y marca.

ARCHIVOS-CORTE DE CONTROL

Se debe contabilizar los hogares de bajos recursos en el territorio nacional. Cada registro contiene información de provincia, localidad, barrio y cantidad hogares. La totalización debe realizarse por localidad y provincia.

```
Program hogaresPlanHelp
Type
    const valorAlto = 1000000;
    RegistroHogares = Record
                              Codigo_provincia: integer;
                              Codigo_localidad: integer;
                              Barrio: integer;
                              Cantidad: integer;
                  end;
     tArchivo = File of RegistroHogares
Var
   archivo: tArchivo;
  hogProvincia, hogLocalidad , cod_provincia,cod_localidad: integer;
   reg: RegistroHogares;
```

```
Begin
Assign(archivo, ‘hogares.dd'); Reset(archivo);
WriteLn(‘Plan HELP 2020’);WriteLn;
Leer(archivo, reg);
With reg do
  While (codProv <> valorAlto) do     begin
    codProvAct:=codProv;    votosProvincia:=0;
    writeLn; writeLn('Provincia ', codProvAct);
   while (codProv=codProvAct) do  begin                {corta la ejecución cuando cambia pcia}
      codLocAct:=codLoc;
      hogLocalidad:=0;
      write('     Localidad ', codLocAct);
    while (codProv=codProvAct) and    (codLoc=codLocAct) do begin {corta la ejecución cuando cambia pcia
    o la localidad}
                              hogLocalidad := hogLocalidad +cantidad;
                               Leer(archivo, reg)
                end;
      hogProvincia := hogProvincia + hogLocalidad ;
      writeLn('Hogares Plan Help localidad: ', hogLocalidad );
      end;
    writeLn(‘Hogares Plan Help Pcia: ', hogProvincia )
    end;
Close(archivo); WriteLn; Write('Oprima tecla de ingreso para finalizar...'); ReadLn end.
```

ACTUALIZACIÓN DE ARCHIVOS-MAESTRO/DETALLE

```
const valoralto = '9999';
type   str4 = string[4];
       producto = record
                  cod: str4;
                  descripcion: string[30];
                  pu: real;
                  stock: integer;
       end;
       venta_prod = record
                  cod: str4;
                  cant_vendida: integer;
       end;
       detalle = file of venta_prod;
       maestro = file of producto;
```

ACTUALIZACIÓN DE ARCHIVOS-MAESTRO/DETALLE

```
var
        mae: maestro;
        regm: producto;
        det: detalle;
        regd: venta_prod;
        total: integer;
        aux: str4;

procedure leer(var archivo: detalle; var dato: venta_prod);
begin
    if (not(EOF(archivo))) then
       read (archivo, dato)
     else dato.cod := valoralto;
end;
```

ACTUALIZACIÓN DE ARCHIVOS-MAESTRO/DETALLE

```
begin {programa principal}
    assign(mae, 'maestro');
    assign(det, 'detalle');
    reset(mae);
    reset(det);
    read(mae, regm);
    leer(det, regd);
{Se procesan todos los registros del archivo detalle}
while (regd.cod <> valoralto) do begin
   aux := regd.cod;
   total := 0;
{totaliza el total vendido para = producto}
    while (aux = regd.cod) do begin
          total := total + regd.cant_vendida;
          leer(det, regd);
    end;
```

Siempre el corte de control principal es la finalización del/ de los detalles???

ACTUALIZACIÓN DE ARCHIVOS-MAESTRO/DETALLE

```
{se busca el producto detalle en el maestro}
        while (regm.cod <> aux) do
                read (mae, regm);
{se modifica el stock del producto con la cantidad total vendida de ese producto}
        regm.cant := regm.cant – total;
        {se reubica el puntero en el maestro}
        seek(mae, filepos(mae)-1);
        {se actualiza el maestro}
        write(mae, regm);
        if (not(EOF(mae))) then
                read(mae, regm);
  end;
  close(det);
  close(mae);
end.
```

Que modificaciones habría que hacer para usar n detalles???

MERGE

```
program ejemplo;
        const valor_alto = '9999';
        type str4 = string[4];
        producto = record
                codigo: str4;
                pu: real;
                cant: integer;
        end;
        arc_productos = file of producto;
var
        det1, det2, det3, mae: arc_productos;
        min, regd1, regd2, regd3, prod: producto;
procedure leer (var archivo: arc_productos;var dato: producto);
begin
        if (not(EOF(archivo))) then
                read (archivo, dato)
        else  dato.codigo := valor_alto;
end;
```

MERGE

```
procedure minimo(var det1, det2, det3: arc_productos; var r1, r2, r3, min: producto);
begin
if (r1.codigo<=r2.codigo) and (r1.codigo<=r3.codigo)then begin
        min := r1;
        leer(det1, r1);
end
else
        if (r2.cod <= r3.cod) then begin
                min := r2; leer(det2, r2);
        end
        else begin
                min := r3; leer(det3, r3)
        end;
 end;
```

MERGE

```
begin
  assign (mae, 'maestro');
  assign (det1, 'detalle1');
  assign (det2, 'detalle2');
  assign (det3, 'detalle3');
  rewrite (mae); reset (det1);
  reset (det2); reset (det3);
  leer (det1, regd1);
  leer (det2, regd2);
  leer (det3, regd3);
  minimo (det1, det2, det3,regd1, regd2, regd3, min);
```

MERGE

```
while (min.codigo <> valoralto) do begin
   prod.codigo:= min.codigo;
   prod.pu=min.pu;
   prod.cant := 0;
    while (min.codigo = prod.codigo) do       begin
            prod.cant := prod.cant + min.cant;
            minimo (det1, det2, det3,regd1, regd2, regd3, min);
    end;
    write (mae, prod);
end;
close(mae); close(det1); close(det2); close(det3);
```