Archivos
Algorítmica
Clásica

CONCEPTOS DE BASES DE DATOS

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 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

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

CORTE DE CONTROL

```
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);
While (reg .codProv <> valorAlto) do     begin
        codProvAct:=reg.codProv;
        votosProvincia:=0;
        writeLn('Provincia ', codProvAct);
        while (reg.codProv=codProvAct) do  begin   {corta la ejecución cuando cambia pcia}
                codLocAct:= reg.codLoc;
                hogLocalidad:=0;
                write(' Localidad ', codLocAct);
                while (reg.codProv=codProvAct) and (reg. codLoc=codLocAct) do begin  {corta la ejecución cuando
                   cambia pcia o la  localidad}
                        hogLocalidad := hogLocalidad + reg.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

Se cuenta con un archivo maestro donde figuran todos los productos que maneja una zapatería. De cada producto se almacena: código, descripción, precio y stock.

Diariamente se genera un archivo detalle con las ventas del día, se almacena código de producto y cantidad vendida.

Al finalizar el día se debe actualizar el stock de los productos vendidos en el archivo maestro.

Tanto el archivo maestro como el detalle se encuentran ordenados por código de producto

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);
    regm.cod=valoralto;
    leer(det, regd);
    while (regd.cod <> valoralto) do begin {Se procesan todos los registros del
    archivo detalle}
    aux := regd.cod;
    total := 0;
    while (aux = regd.cod) do begin {suma el total vendido para = producto}
            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

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

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

MERGE

Se cuenta con tres archivos detalles que almacenan productos con los siguientes datos: código del producto, precio unitario y cantidad.

Generar un archivo maestro donde el producto aparezca una sola vez con la cantidad total.

Los tres archivos se encuentran ordenados por código de producto.

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;
    adet = array[1..3] of arc_productos;
    ardet = array[1..3] of producto;
var
    det:adet;
    rdet:ardet;
    mae: arc_productos;
    min, 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 det: adet; var rdet:ardet;var min:producto);
Var
      posMin :int
Begin
      posMin:=1;
      min := rdet[1];
      for i:=2 to 3 do
      begin
            if(rdet[i].codigo < min.codigo)then
            begin
                  min:= rdet[i];
                  posMin:=i;
            end;
      end;
      leer(det[posMin],rdet[posMin]);
end;
```

MERGE

```
begin
    assign (mae, 'maestro');
    for i:=1 to 3 do begin
        writeln('ESCRIBA UN NOMBRE PARA EL ARCHIVO: ');
        read(nombreDet);
        assign(det[i],nombreDet);
    end;
    rewrite (mae);
    for i:=1 to 3 do begin
        reset (det[i]);
        leer (det[i], regd[i]);
  end
  minimo (det, rdet, 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 (det, rdet, min);
    end;
    write (mae, prod);
  end;
  close(mae);
  for i:=1 to 3 do
    close (det[i])
End.
```

A TENER EN CUENTA

- Utilizar siempre el procedimiento leer para estos algoritmos
- Para Merge o Actualización Maestro Detalle con más de un archivo detalle utilizar siempre el procedimiento mínimo