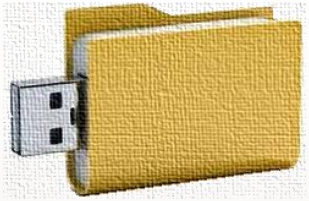
Analista Programador Universitario

Estructura de Datos

Unidad V: Archivos



Facultad de Ingeniería
Universidad Nacional de Jujuy

Índice

- Definición de archivos
  - registros, claves, bloques
- Medios de almacenamiento
  - soportes secuenciales y direccionables
- Acceso
  - secuencial y directo
- Organización
  - secuencial, directo y secuencial indexado
- Operaciones
  - creación, consulta, actualización, clasificación, reorganización, destrucción, reunión, fusión, rotura

Definición (1)

- Desde el punto de vista físico, un archivo es una sucesión de bytes almacenados en un soporte secundario o medio de almacenamiento (disco rígido, DVD, cintas magnéticas, pendrives, etc.).
- Desde el punto de vista lógico, un archivo es una colección de registros que representa las entidades de una situación específica (clientes, productos, facturas, etc.).

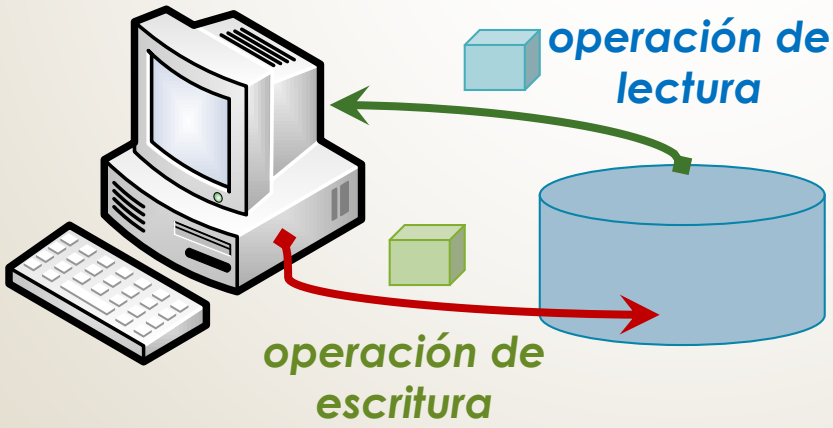
Definición (2)

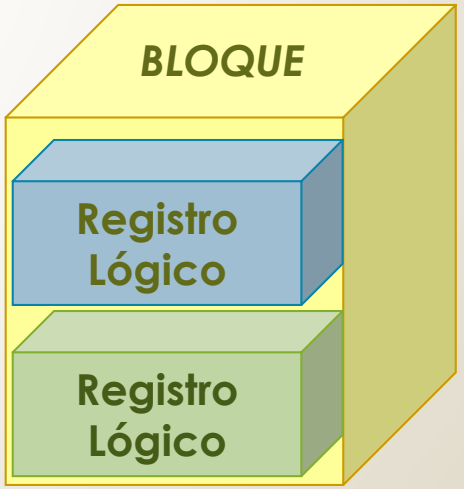
- Registro lógico
  - un registro es un conjunto de campos, lógicamente relacionados, que representan entidades del mundo real.
- Clave o llave
  - se trata de un campo o conjunto de campos que permiten identificar de forma única un registro de archivo. Por ejemplo, el número de legajo de un empleado.

El campo legajo permite identificar de forma única a cada empleado.

Definición (3)

- Registro físico o bloque
  - Se refiere a la cantidad de información (1 o más registros lógicos) que se transfiere en una operación de lectura/escritura sobre el medio de almacenamiento.
- Factor de bloqueo
  - Se refiere al número de registros lógicos que contiene un registro físico.





Definición (4)

- Partes de un registro
  - Parte maestra: información sujeta a pocos cambios, se utiliza como histórico, identificación y referencia.
  - Parte secundaria: información sujeta a cambios constantes
- Tipos de archivos
  - Archivos maestros: se trata de archivos con información general de poco variación, se utilizan para referencia, identificación y estadísticas. Por ejemplo: archivos de empleados, alumnos, etc.
  - Archivos de transacciones: contienen información acerca de las operaciones de la organización, se trata de archivos temporales que permiten actualizar los archivos maestros.
  - Archivos de trabajo: se trata de archivos auxiliares utilizados durante el procesamiento de los archivos anteriores.

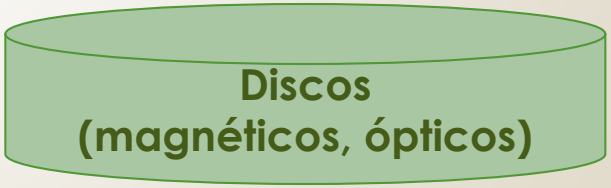
Definición (5)

- Características
  - Los archivos se almacenan en soportes secundarios que proporcionan gran capacidad de almacenamiento.
  - La información se almacena en forma permanente.
  - Un archivo puede ser utilizado por diferentes programas.
  - La unidad básica de transferencia entre programas y archivos es el registro.
  - Durante una operación de lectura/escritura se procesa cierta cantidad de registros, pero no el archivo completo.

Medios de Almacenamiento

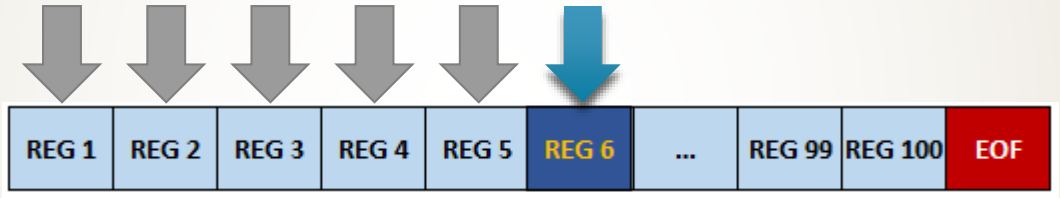
- Los medios de almacenamiento constituyen el soporte físico en el que se almacenan los datos de forma permanente.
  - Soportes secuenciales: los registros del archivo se almacenan uno a continuación de otro. El acceso al registro *k*, implica el acceso a los *k-1* registros anteriores.

  - Soportes direccionables: estos soportes permiten acceder directamente a los registros de un archivo en base al valor de un campo clave. Estos soportes también admiten el acceso secuencial.

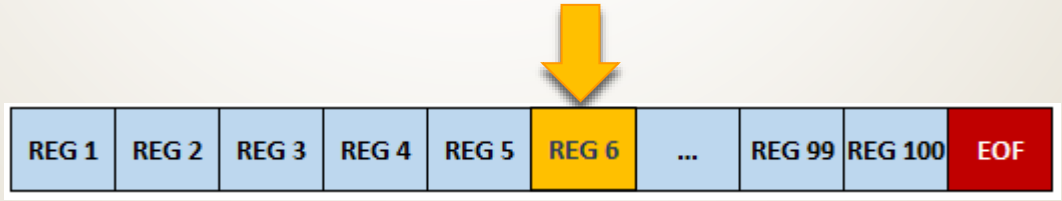


Acceso

- Según el tipo de soporte y la forma en la que se procesen los registros de un archivo, el acceso puede ser:
  - Secuencial, cuando el procesamiento de los registros se realiza de acuerdo al orden de almacenamiento de éstos.



  - Directo, cuando el procesamiento de un registro determinado se realiza directamente, sin necesidad de acceder a los registros previos.



Organización

- La organización de un archivo define la disposición de los registros en un medio de almacenamiento, es decir, la forma en la que se estructuran los datos en un archivo.
  - Secuencial
  - Directo
  - Secuencial Indexado

Organización. Secuencial (1)

- Los registros se disponen uno a continuación de otro en el medio de almacenamiento, siendo necesario pasar por *n-1* registros para acceder al registro *n.*
  - El orden físico de los registros determina el orden de acceso a los mismos.
  - Todos los medios de almacenamiento soportan el acceso secuencial.
  - Existe un registro especial llamado EOF (*end of file*) que indica el final del archivo.

Organización. Secuencial (2)

```
TIPOS
     t_registro=REGISTRO
                    campo1: t_dato
                    campo2: t_dato
                    …
                    campon: t_dato
                FIN_REGISTRO
     t_archivo=ARCHIVO SECUENCIAL de t_registro

VARIABLES
     info: t_archivo
```

Organización. Directo (1)

- El orden físico de los registros no coincide con el orden lógico.
- El acceso es directo, se realiza mediante la posición o lugar relativo.
- La clave de registro se transforma en la posición relativa del registro en el soporte.
- Características
  - Se almacenan en soportes direccionables.
  - Los registros contienen un campo especial (*clave*) que permite identificar de forma única cada registro.
  - Existe una correspondencia entre las claves y las posiciones del soporte (función de conversión de claves o función *hash*).

Organización. Directo (2)

- La función de conversión de claves f(k) puede generar colisiones.
- Una colisión ocurre cuando para 2 claves de registro se genera la misma dirección física (posición relativa dentro del archivo).

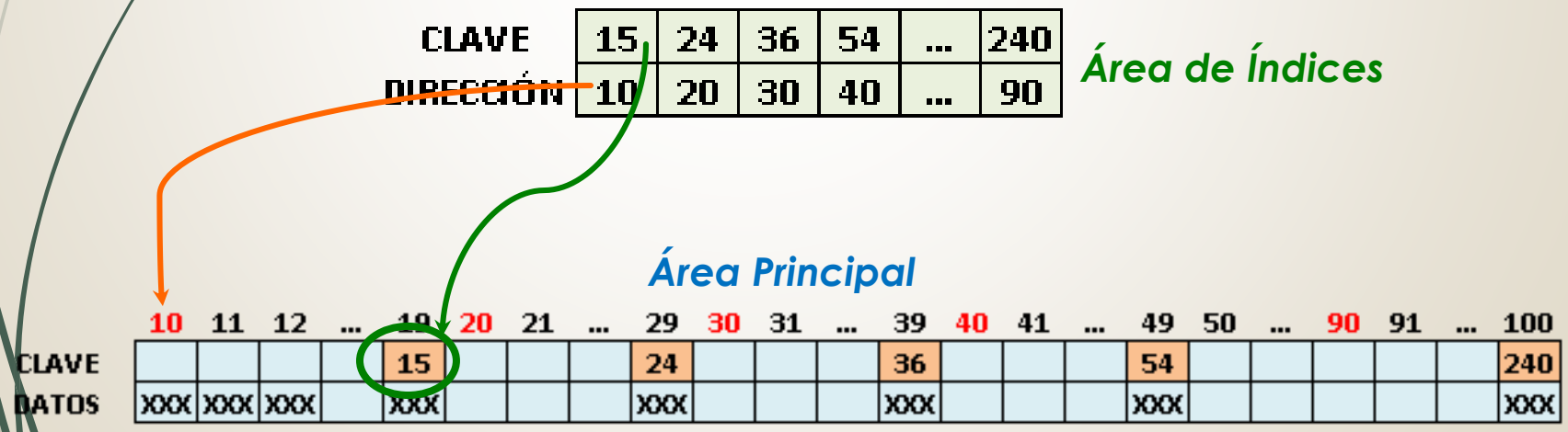
Organización. Directo (3)

```
TIPOS
     t_registro=REGISTRO
                    *campo1: t_dato //CAMPO CLAVE
                    campo2: t_dato
                    …
                    campon: t_dato
                FIN_REGISTRO
     t_archivo=ARCHIVO DIRECTO de t_registro

VARIABLES
     datos: t_archivo
```

Organización. Indexado (1)

- Un archivo es secuencial indexado si:
  - posee un campo clave que permite identificar de forma única sus registros
  - se almacena en un soporte direccionable
  - existe un archivo de índices que contiene las claves del archivo y las direcciones físicas correspondientes.



Organización. Indexado (2)

- Partes de un archivo indexado
  - Área de datos o primaria: contiene los registros en forma secuencial
  - Área de índices: contiene los niveles de índices, pueden existir varios índices (niveles de indexación)
  - Área de desbordamiento o excedentes: para actualizaciones.

Organización. Indexado (3)

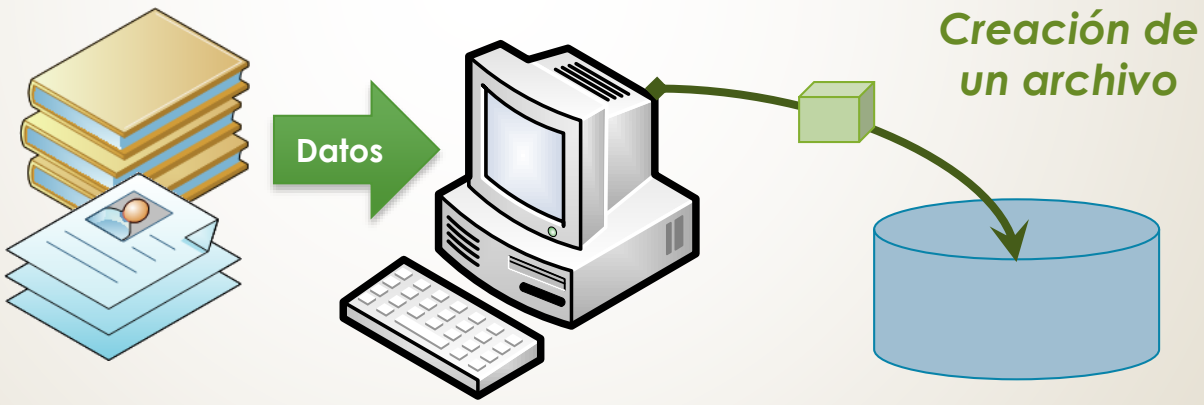
```
TIPOS
     t_registro=REGISTRO
                    *campo1: t_dato //CAMPO CLAVE
                    campo2: t_dato
                    …
                    campon: t_dato
                FIN_REGISTRO
     t_registro2=REGISTRO
                    campo1: t_dato
                    posición: t_dato
                FIN_REGISTRO
     t_archivo=ARCHIVO INDEXADO de t_registro
     t_archivo2=ARCHIVO SECUENCIAL de t_registro2
VARIABLES
     datos: t_archivo
     indice: t_archivo2
```

Operaciones sobre archivos

- Creación
- Consulta
- Actualización (altas, bajas, modificaciones)
- Clasificación
- Reorganización
- Destrucción
- Reunión, Fusión
- Rotura (estallido)

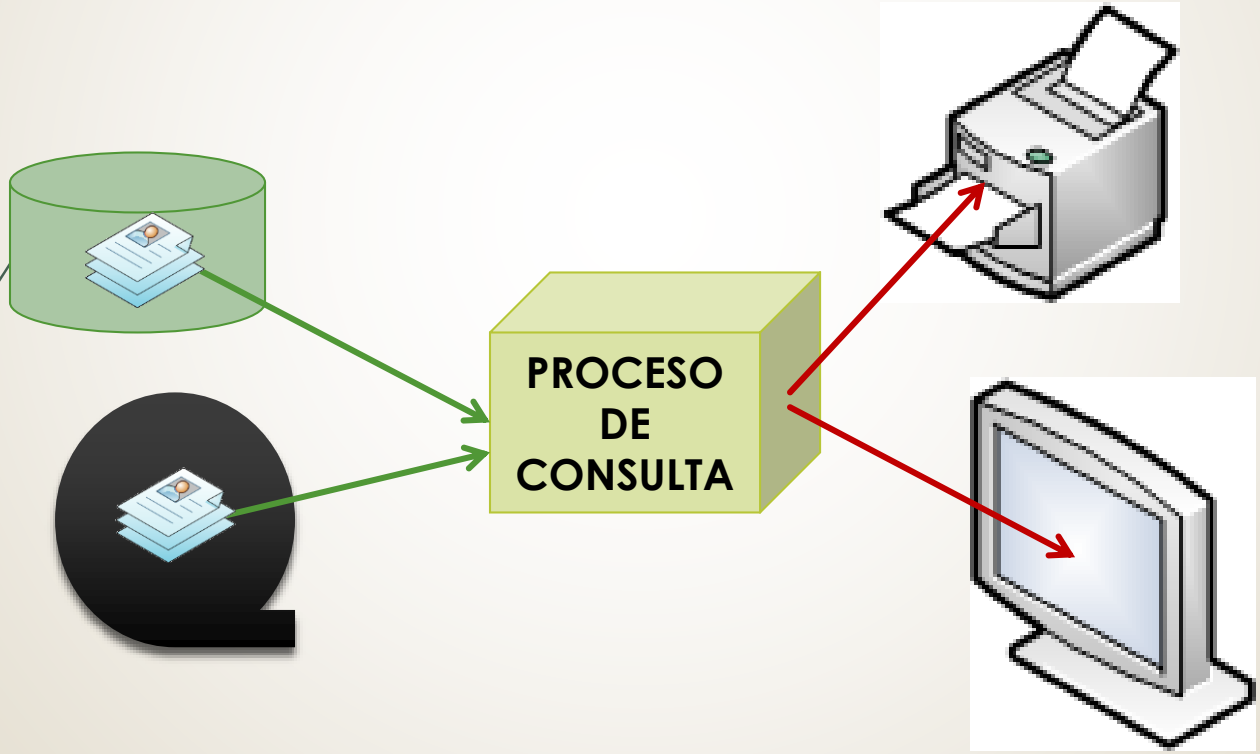
Operaciones. Creación

- Es la primera operación que se realiza sobre un archivo.
- Debe definirse la estructura de datos, el soporte y la organización del archivo.



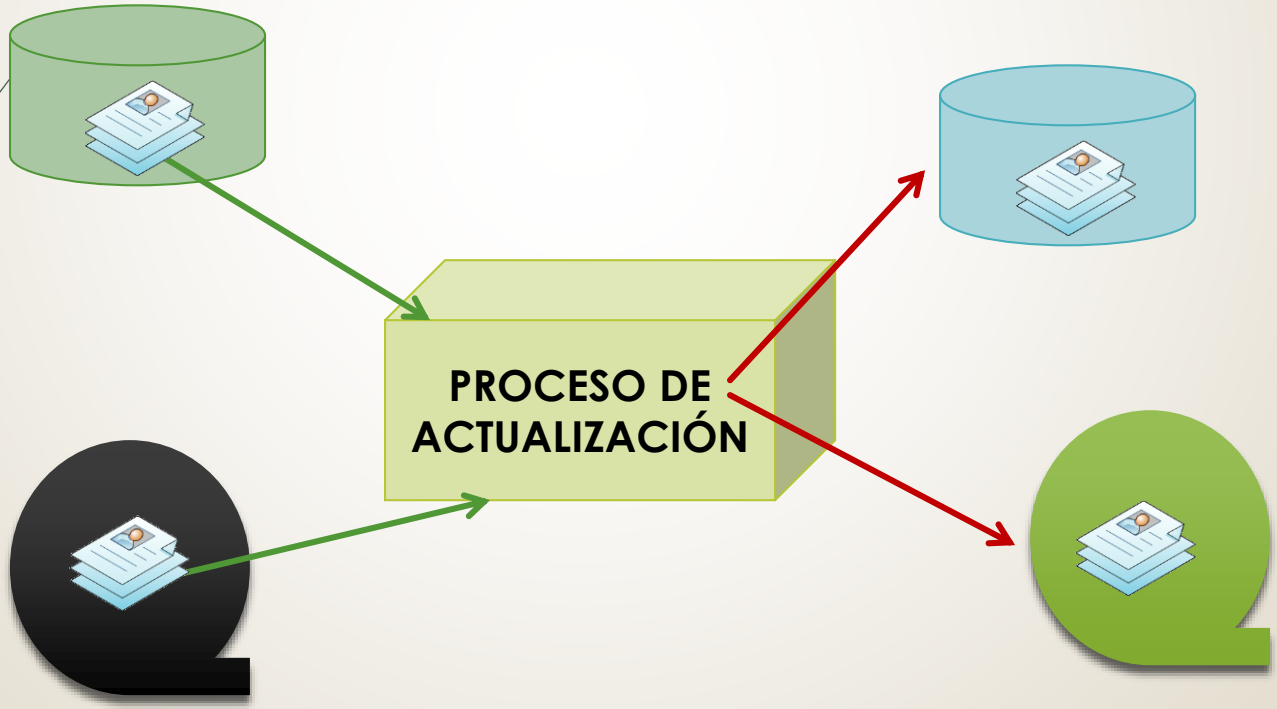
Operaciones. Consulta

- Permite conocer el contenido de uno o varios registros de un archivo.



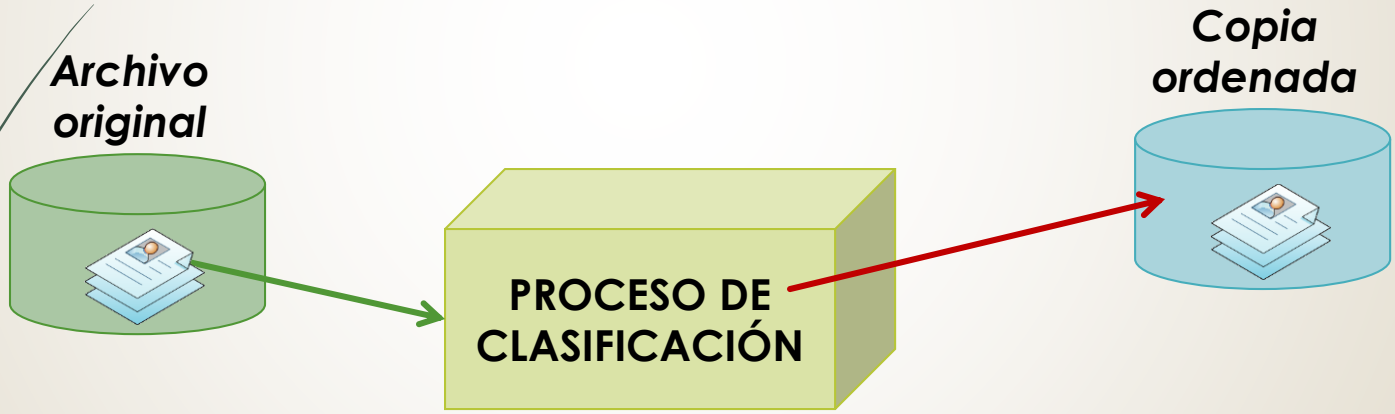
Operaciones. Actualización

- Las operaciones de actualización consisten en agregar nuevos registros al archivo y modificar o eliminar los registros existentes (borrado lógico/borrado físico)



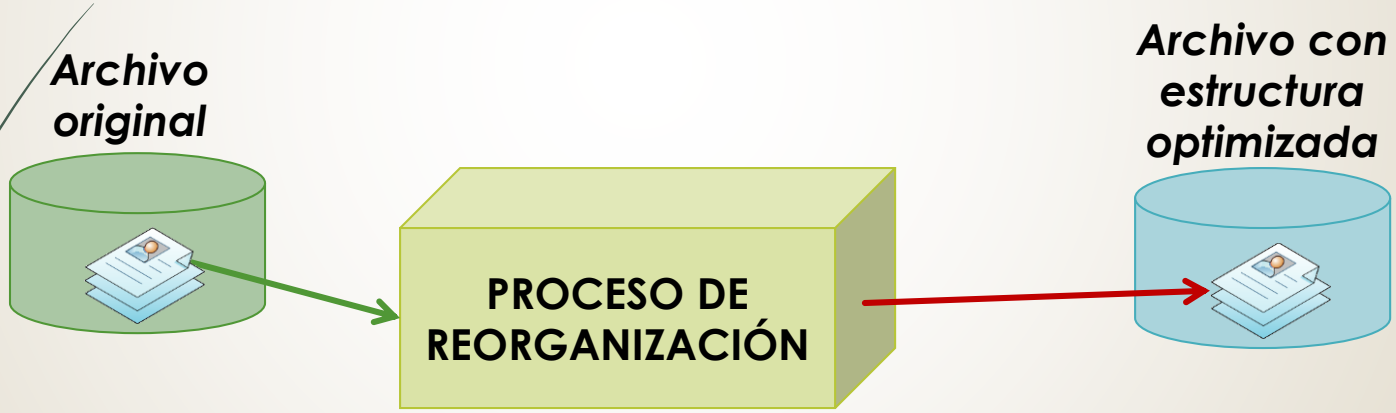
Operaciones. Clasificación

- La clasificación u ordenación permite clasificar el contenido de un archivo de acuerdo a un criterio específico.



Operaciones. Reorganización

- La reorganización consiste en generar un nuevo archivo (copia del original) a fin de optimizar su estructura (índices, áreas de desbordamiento).

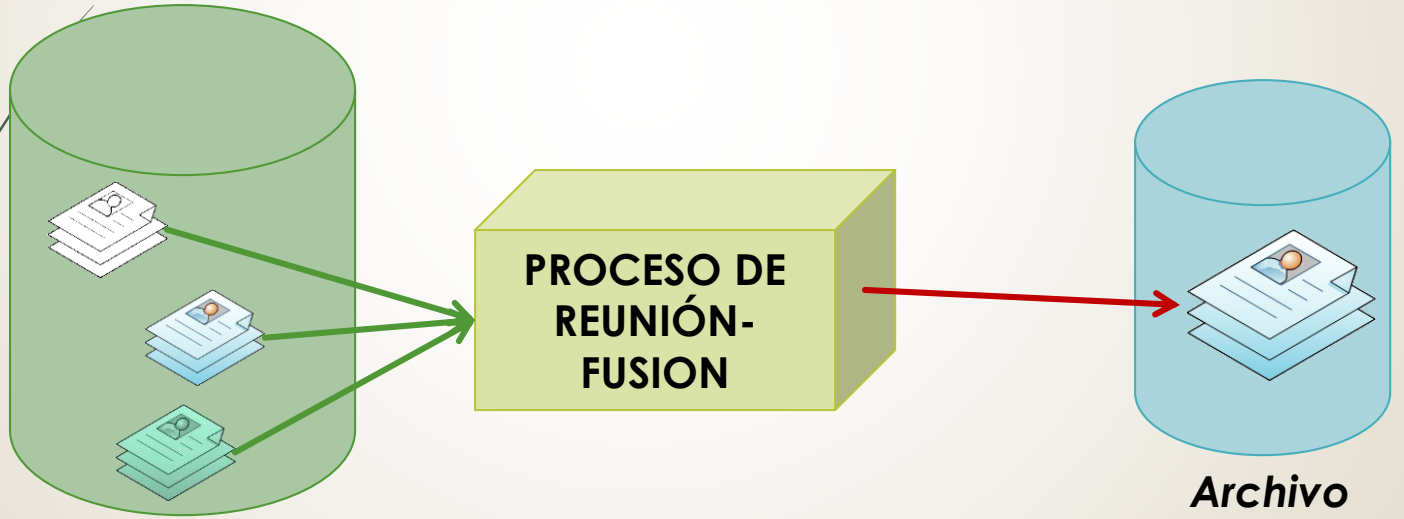


Operaciones. Destrucción

- Cuando se destruye un archivo este se borra del almacenamiento, y por tanto ya no se podrá acceder a sus registros.

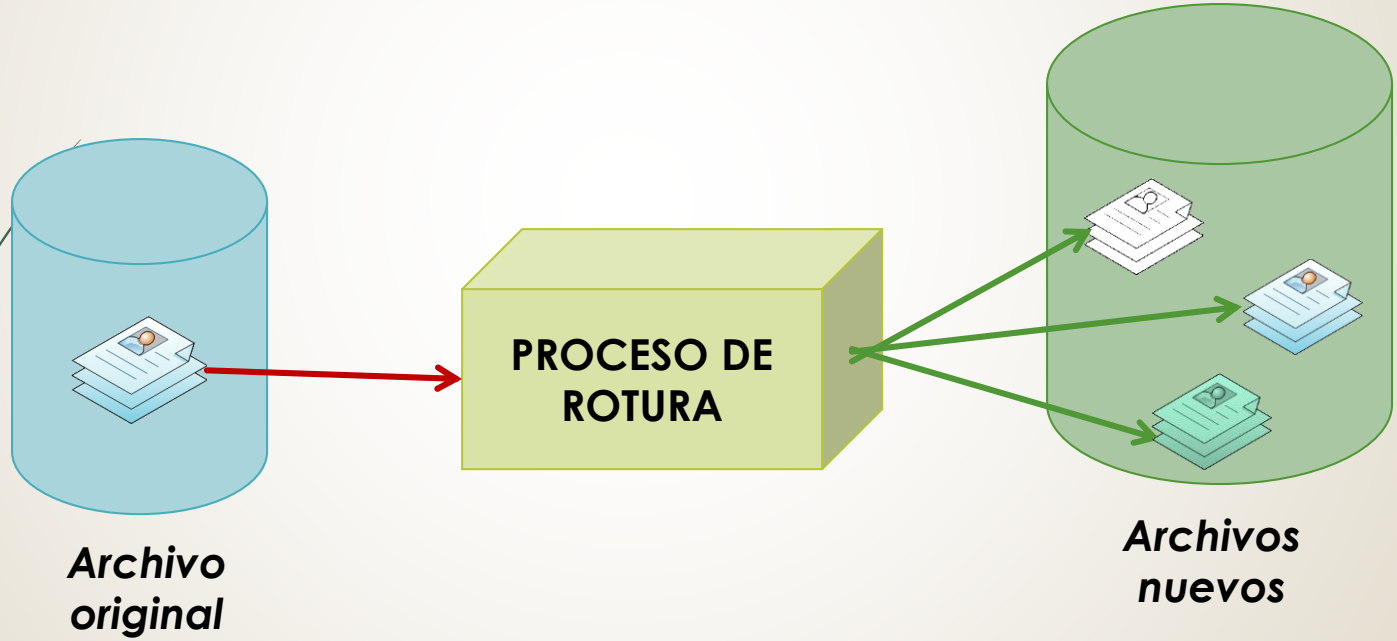
Operaciones. Reunión-Fusión

- La reunión permite obtener un archivo a partir de varios (que pueden tener estructura diferente).
- La fusión permite mezclar archivos con la misma estructura según un criterio definido.



Operaciones. Rotura

- La rotura permite obtener varios archivos a partir de un mismo archivo inicial.



Archivos en C

- En C los archivos se clasifican en:
  - Archivos de texto, en los que la información se almacena como cadenas de caracteres.
  - Archivos binarios, la información se almacena como secuencias de bytes.
- Declaración de Archivos en C

```
typedef FILE *parchivo;
…
main()
{ parchivo datos;
   …
}
```

Archivos en C. Operaciones

- Operaciones habituales sobre archivos

Operación	Función	Descripción
Apertura de archivos	*fopen()*	Abre un archivo y define el modo de apertura
Cierre de Archivos	*fclose()*	Cierra un archivo
Lectura de datos	*fread()*	Lee datos de un archivo
	fscanf()	Lee un dato de un archivo según un formato
	fgetc()	Lee un carácter de un archivo
Escritura de datos	*fwrite()*	Escribe datos en un archivo
Limpiar Buffers	*fflush()*	Borra el contenido del buffer especificado
Final de archivo	*feof()*	Detecta el final de archivo
Posicionar puntero	*fseek()*	Ubica el puntero en una posición específica del archivo
Posición del puntero	*ftell()*	Devuelve la posición actual del puntero en el archivo
Iniciar puntero	*rewind*	Ubica el puntero al inicio del archivo
Eliminar archivos	*remove()*	Elimina el archivo especificado
Renombrar archivos	*rename()*	Renombra el archivo especificado

Archivos en C. Apertura

- Apertura de archivos

```
puntero_archivo=fopen("ruta archivo","modo");
```

Modo de apertura (archivos de texto)	Modo de apertura (archivos binarios)	Operación
"r"	"rb"	Apertura en modo de sólo lectura. El archivo debe existir.
"w"	"wb"	Apertura en modo de sólo escritura. Si el archivo existe, se reescribirá (pierde el contenido anterior). Si el archivo no existe, lo crea.
"a"	"ab"	Apertura en modo de agregar. Si el archivo existe, los datos se agregan al final del archivo, en caso contrario, el archivo se crea.
"r+"	"rb+"	Apertura en modo de lectura/escritura. El archivo debe existir.
"w+"	"wb+"	Apertura en modo de lectura/escritura. Si el archivo existe, se reescribirá (pierde el contenido anterior).
"a+"	"ab+"	Apertura en modo de lectura/agregar. Si el archivo no existe lo crea.

Archivos en C. Cierre

- Cierre de archivos
  - `fclose(puntero_archivo)` : cierra el archivo especificado como argumento.

Archivos en C. Lectura

- Lectura de archivos
  - **`fread`**: lee información almacenada en un archivo.

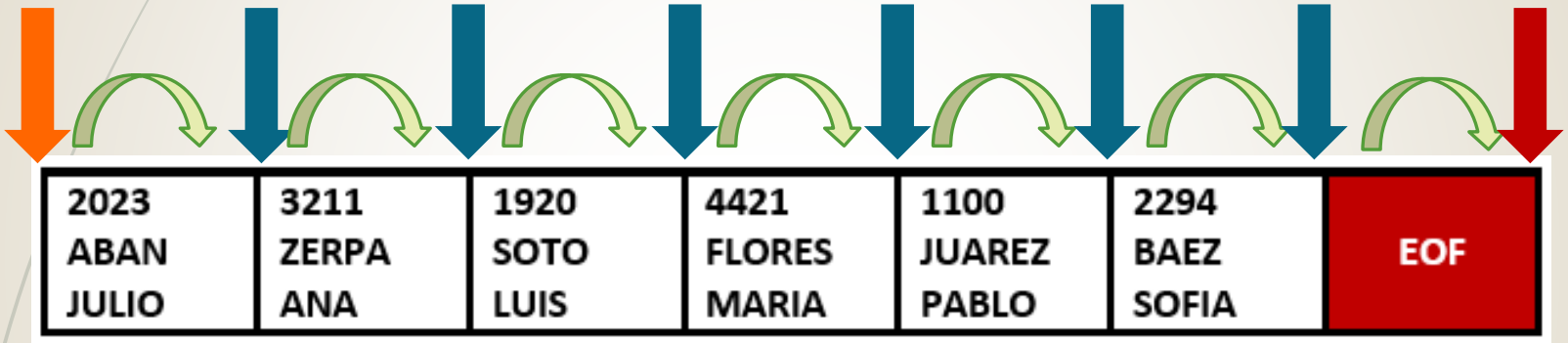
```
fread(registro, tamaño registro, cant. registros, archivo)
```

```
fread(&reg_entrada,sizeof(reg_entrada),1,puntero_archivo);
```

  - *reg_entrada* se refiere a la variable tipo registro que almacenará los datos leídos desde archivo
  - *sizeof(registro_entrada)* indica el tamaño o cantidad de información leída desde el archivo
  - 1 especifica que se leerá sólo un registro del archivo
  - *puntero_archivo* especifica el nombre del puntero que hace referencia al archivo.

Archivos en C. Lectura

- Lectura de archivos



Archivos en C. Escritura

- Escritura de archivos
  - **fwrite**: escribe información en un archivo.

```
fwrite(registro, tamaño registro, cant. registro, archivo)
```

```
fwrite(&reg_salida,sizeof(reg_salida),1,puntero_archivo);
```

  - *reg_salida* se refiere a la variable que almacena los datos a escribir en el archivo
  - *sizeof(registro_salida)* indica el tamaño o cantidad de información que se escribirá en el archivo
  - 1 especifica que se escribirá sólo un registro en el archivo
  - *puntero_archivo* especifica el nombre del puntero que hace referencia al archivo.

Archivos en C. Posicionamiento

- Final de archivo
  - **feof**: determina si se alcanzó el final de un archivo.

```
feof(puntero_archivo);
```

- Ubicación del Puntero
  - **ftell**: indica en que posición del archivo está el puntero.

```
ftell(puntero_archivo);
```

- Posicionamiento del Puntero
  - **fseek**: ubica el puntero en una posición específica.

```
fseek(puntero_archivo,desplazamiento,punto de referencia);
```

- Iniciar el Puntero
  - **rewind**: ubica el puntero al comienzo del archivo.

```
rewind(puntero_archivo);
```

Archivos en C. Manipulación

- Renombrado de archivos
  - **rename**: renombra el archivo especificado con el nuevo nombre. Si la operación se realiza correctamente, la función retorna valor 0.

```
rename("nombre_actual","nombre_nuevo");
```

- Eliminación de archivos
  - **remove**: elimina del disco el archivo especificado. Si la operación se realiza correctamente, la función retorna valor 0.

```
remove("nombre_archivo");
```

Archivos en C. Ejemplo (1)

```
#include <iostream>
#include <stdlib.h>
#include <string.h>
#include <stdio.h>

using namespace std;

const int MAX=25;

typedef FILE *parchivo;

typedef char tcad[MAX];

typedef struct tprod
                { int codigo;
                  tcad descripcion;
                  float precio;
                };
```

Definición del tipo archivo (puntero a archivo)

Definición del registro que se almacenará en el archivo

Archivos en C. Ejemplo (2)

```
void alta(parchivo prod)
{ tprod p;
  char rta;
  prod=fopen("prod.txt","ab+");
  do
  { cargar(p);
    fwrite(&p,sizeof(p),1,prod);
    cout << "Ingresar mas datos S/N: ";
    cin >> rta;
  }while(rta!='N' && rta!='n');
  fclose(prod);
}
```

Abrir el archivo en modo lectura/escritura (agregar)

Escribir un registro en el archivo

Cerrar el archivo

```
void cargar(tprod &p)
{
  cout << "Codigo: ";
  cin >> p.codigo;
  cout << "Descripcion: ";
  fflush(stdin);
  gets(p.descripcion);
  cout << "Precio: ";
  cin >> p.precio;
}
```

Archivos en C. Ejemplo (3)

```
void listar(parchivo prod)
{  tprod p;
   prod=fopen("prod.txt","rb");
   while (!feof(prod))
   { fread(&p,sizeof(p),1,prod);
     if (!feof(prod))
        mostrar(p);
   }
   fclose(prod);
}
```

Abrir el archivo en modo lectura

Verificar final del archivo

Leer un registro del archivo

Cerrar el archivo

```
void mostrar(tprod p)
{
  cout << "Codigo: " << p.codigo << endl;
  cout << "Descripcion: " << p.descripcion << endl;
  cout << "Precio: " << p.precio << endl;
}
```

Archivos en C. Ejemplo (4)

```
void buscar(parchivo prod, int cod)
{ bool encontrado=false;
  tprod p;
  prod=fopen("prod.txt","rb");
  while (!feof(prod)&& !encontrado)
  { fread(&p,sizeof(p),1,prod);
    if (p.codigo==cod)
    { mostrar(p);
      encontrado=true;
    }
  }
  if (encontrado==false)
    cout << "NO ENCONTRADO" << endl;
  fclose(prod);
}
```

Abrir el archivo en modo lectura

Verificar final del archivo

Leer un registro del archivo

Cerrar el archivo

Archivos en C. Ejemplo (5)

```
void modificar(parchivo prod, int cod)
{ bool encontrado=false;
  tprod p;
  prod=fopen("prod.txt","rb+");
  while (!feof(prod)&& !encontrado)
  { fread(&p,sizeof(p),1,prod);
    if (p.codigo==cod)
    { mostrar(p);
      encontrado=true; }
  }
  if (encontrado==false)
    cout << "NO ENCONTRADO" << endl;
  else
   { cargar(p);
     fseek(prod,-sizeof(p),SEEK_CUR);
     fwrite(&p,sizeof(p),1,prod);
   }
  fclose(prod);
}
```

Abrir el archivo en modo lectura/escritura

Leer un registro del archivo

Ubicar el puntero al inicio del registro a modificar

Escribir el registro modificado en el archivo

Cerrar el archivo

Archivos en C. Ejemplo (6)

```
void borrar(parchivo prod, int cod)
{ tprod p;
  parchivo temp;
  prod=fopen("prod.txt","rb");
  temp=fopen("temporal.txt","wb");
  if (prod==NULL)
   { cout << "ARCHIVO INEXISTENTE" << endl;
     fclose(prod);
     fclose(temp);
   }
  else
  { while (!feof(prod))
    { fread(&p,sizeof(p),1,prod);
      if (p.codigo!=cod && !feof(prod))
        fwrite(&p,sizeof(p),1,temp);
    }
    fclose(prod);
    fclose(temp);
    if (remove("prod.txt")==0)
      rename("temporal.txt","prod.txt");
    else
      cout << "PROBLEMAS AL BORRAR REGISTRO" << endl;
  }
}
```

Abrir el archivo en modo lectura

Abrir el archivo en modo escritura

Cerrar archivos

Leer un registro del archivo

Escribir el registro en el temporal

Cerrar archivos

Eliminando archivo

Renombrar archivo temporal

Bibliografía

- Sznajdleder, Pablo Augusto. Algoritmos a fondo. Alfaomega. 2012.
- López Román, Leobardo. Programación estructurada y orientada a objetos. Alfaomega. 2011.
- De Giusti et al. Algoritmos, datos y programas, conceptos básicos. Editorial Exacta, 1998.
- Joyanes Aguilar, Luis. Fundamentos de Programación. Mc Graw Hill. 1996.
- Joyanes Aguilar, Luis. Programación en Turbo Pascal. Mc Graw Hill. 1990.