Estructuras de datos:
Archivos tipo registro
Estructura Jerárquica: fichero de [registro.campo]

1. Archivo de registros.

Un **archivo** o fichero de registros es un conjunto de datos estructurados o colección de entidades básicas denominadas **registros**, de igual estructura en tamaño y tipo. Los registros constan a su vez de entidades de nivel mas bajo denominados **campos**.

Acceso a dispositivo de disco:
Es la técnica de Organización y gestión de datos estructurados sobre dispositivos de almacenamiento secundario

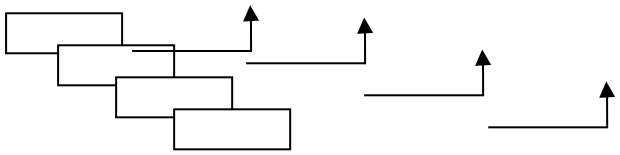
Fichero:
Estructuras o Colecciones de datos almacenados en dispositivos de memoria auxiliar y permanente. Los datos son Escritos / Leídos desde ficheros y alojados en la memoria RAM solo para su procesamiento (Administrar Información).

2. Campos.

Los elementos de datos se agrupan en unidades básicas llamados campos (fields). Estos miembros de datos elementales son por ejemplo: nombre, código de empleado, sueldo, edad, fecha de nacimiento, etc.

Definición. Un campo esta definido por su identificador, tamaño, y tipo de dato (entero, real, cadena, lógico, array, etc). Los campos suelen ser de longitud fija en la mayoría de los lenguajes.

Codigo Apellidos Basico, Fecha_Ingreso

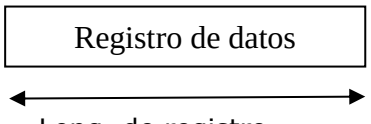


Campos (Fields)

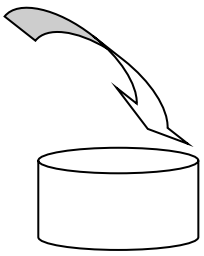
3. Registros.

Un registro es una colección de información, normalmente relativa a una entidad en particular, (empleados, artículos, etc.). Un registro es una colección de campos definidos con un tipo de datos (Regla de Modelo de Datos) y por tanto son heterogéneos. Los campos en un registro al estar lógicamente relacionados, pueden ser tratados como una unidad. Por ultimo, los registros tienen la misma estructura y tamaño.

Los registros organizados en campos se denominan **registros lógicos**.



Registro (Record)



Fichero de Registros

4. Gestión de Ficheros de registros.

4.1. Las operaciones más usuales en los registros son:

- Consulta: Lectura/Escritura del contenido de un registro
- Modificación: alterar el contenido del registro
- Inserción: Añadir un nuevo registro al archivo (incluye funciones de archivo)
- Borrado: Suprimir un registro del archivo. (incluye funciones de archivo)

4.2. Operaciones sobre ficheros.

Las operaciones sobre archivos se realizan mediante programas en donde los archivos se identifican por un nombre externo al que están asociados y una ruta de acceso en el dispositivo de almacenamiento. (el nombre de dispositivo incluye la unidad de disco, una vía de acceso o path y el nombre.ext del fichero, en plataforma Windows).

La mayoría de programas ejecutan las siguientes operaciones sobre los ficheros.

- Crear archivos (create).
- Abrir o arrancar (open) un archivo que fue creado con anterioridad.
- Transferir datos desde(leer/read) o a (escribir/write) el dispositivo definido por la función abrir/open.
- Cerrar el fichero (close) después que el programa ha terminado de utilizarlo.
- Borrar el fichero (Delete) un fichero existente.
- Renombrar nombres de ficheros.
- Copiar archivos (backup), etc

Gestión de Archivos de registros: Para poder gestionar un archivo mediante un programa es preciso declarar el archivo, y la estructura del registro.

Antes de que cualquier dato sea escrito o leído de un archivo, el archivo debe existir en el dispositivo correspondiente.

La declaración se realiza así:

Sintaxis:

```
tipo
      registro: <tipo_registro>
            <tipo_dato>: nombre_campo1
            <tipo_dato>: nombre_campo2

            <tipo_dato>: nombre_campo_n
      fin_registro

      archivo_<organizacion> de <tipo_registro>: <tipo_archivo>

var
      <tipo_registro>:  var_tipoRegistro
      <tipo_archivo> :  var_tipoArchivo
```

Ejemplo comentado. Gestión de archivos de registros

- La definición de la estructura de un registro de empleados, seria:

```
tipo
      registro: regEmpleado
            entero : codigo   //4 digitos
            cadena : nombre   //15 caracteres
            real   : sueldo   //(6,2) 3 enteros, 2 decimales
      fin_registro

      archivo_s de regEmpleado: idArchivo

var
      regEmpleado:  empleado
      idArchivo   :  idf
```


IMPORTANTE

Los campos de un registro, además de las reglas de Modelo de Datos (tipos de datos) deben tener reglas de restricción, para implementar la integridad de datos.
Asi por ejemplo, Por regla de Modelo de Datos, existen 24 trabajadores y no 24.5. O bien reforzando el modelo de datos para el campo precio de tipo real, por regla de restricción el precio de un articulo es 4.5 US$ y no -4.5 US$ (precio negativo).

Operaciones de consulta de registros. Leer y escribir registros

- La lectura o consulta del contenido del registro seria mediante el procedimiento:

```
      llamar_a leerRegistro(empleado)
```

y su implementación:

```
procedimiento leerRegistro(S reg_empleado: emp )
var
inicio
      leer(emp.codigo)
      leer(emp.nombre)
      leer(emp.sueldo)
fin_procedimeinto
```

- La escritura o consulta de visualización del contenido del registro seria mediante el procedimiento:

```
      llamar_a escribirRegistro(empleado)
```

y su implementación:

```
procedimiento escribirRegistro(E reg_empleado: emp )
var
inicio
      escribir(emp.codigo)
      escribir(emp.nombre)
      escribir(emp.sueldo)
fin_procedimeinto
```

Operaciones sobre ficheros: Transferencia de datos (Modos de acceso)

- La operación de agregar nuevos registros a un fichero, implica la transferencia en modo añadir (append) de datos al fichero. Esta operación de inserción siempre es complementada por las funciones de Archivos, asi:

```
crear(idf, "nomina.dat")
abrir(idf, a, "nomina.dat")   //a – modo añadir (escritura)


si idf = NULL entonces
      escribir("error de acceso")
      salir()
fin_si

llamar_a leerRegistro(empleado)
escribirArchivo(idf, empleado)
cerrar(idf)
```

- La operación de transferencia o lectura de un fichero de registros, implica el acceso al dispositivo de disco en modo lectura (función read), asi:

```
inicio
      abrir(idf, l, "empleados.dat")      //l – modo lectura
      si idf = NULL entonces
            escribir("error de acceso")
            salir()
      fin_si

      mientras no fda(idf) hacer
            leerArchivo(idf, empleado)
            llamar_a escribirRegistro(empleado)
      fin_mientras

      cerrar(idf)
fin
```

- o bien:

```
inicio
      abrir(idf, "empleados.dat")

      leerArchivo(idf, empleado)
      mientras no fda(idf) hacer
            llamar_a escribirRegistro(empleado)
            leerArchivo(idf, empleado)
      fin_mientras

      cerrar(idf)
fin
```

Implementación en Código c++

Aplicación 1. Acceso a dispositivo en modo escritura

```
/*
  Name: Escribir_registro.cpp
  Author: J.Medianero.A
  Date: 20/12/09
  Description: Acceso a fichero de registros
*/
#include <cstdlib>
#include <iostream>

struct regEmpleado{
       int codigo;
       char nombre[20];
       float sueldo;
       };

void leerRegistro(struct regEmpleado *);
using namespace std;

int main(int argc, char* argv[]){
    FILE *idf;
    struct regEmpleado empleado;

    idf = fopen("nomina.dat", "a");
    if(idf ==NULL){
         printf("Error de acceso\n");
         system("PAUSE");
         return 1;
    }else{
         leerRegistro(&empleado);
         fprintf(idf, "%4d %20s %6.2f", empleado.codigo,
                                        empleado.nombre,
                                        empleado.sueldo);
    }

    fclose(idf);
    //system("NotePad.exe nomina.dat");

    system("PAUSE");
    return EXIT_SUCCESS;
}//:~



//definir rutina: leerRegistro
void leerRegistro(struct regEmpleado *emp){
    cout << "Codigo: "; cin >> emp->codigo;
    cout << "Nombre: "; cin >> emp->nombre;
    cout << "Sueldo: "; cin >> emp->sueldo;
}
```

Aplicación 2. Acceso a dispositivo en modo lectura

```
/*
  Name: Leer_registro.cpp
  Author: J.MedianeroA.
  Date: 20/12/09
  Description: Acceso a Ficheros de registros
*/
#include <cstdlib>
#include <iostream>
#include <iomanip>

struct regEmpleado{
       int codigo;
       char nombre[20];
       float sueldo;
       };

using namespace std;
void escribirRegistro(struct regEmpleado);

int main(int argc, char* argv[]){
    FILE *idf;
    struct regEmpleado empleado;

    idf = fopen("nomina.dat", "r");
    if(idf ==NULL){
         printf("Error de acceso\n");
         system("PAUSE");
         return 1;//abandonar rutina
    }else{

          while(feof(idf)== 0){
              //lectura de campos con anchos de posiciones pre-establecidas
              fscanf(idf, "%4d %20s %6f", &empleado.codigo,
                                          &empleado.nombre,
                                          &empleado.sueldo );
              escribirRegistro(empleado);
         }
    }//fin_si
    fclose(idf);

    system("PAUSE");
    return EXIT_SUCCESS;

}//:~


//definir procedimeinto escribirRegistro
void escribirRegistro(struct regEmpleado emp){
     cout << setw(4);                  //establece ancho de 4 posiciones
     cout << emp.codigo << " ";
     cout << setw(20);                 //establece ancho de 20 posiciones
     cout << emp.nombre << " ";
     cout << setw(6);                  //establece ancho de 6 posiciones
     //activa formato de coma flotante con dos decimales
     cout << fixed << setprecision(2);
     cout << emp.sueldo << endl;
}
```