

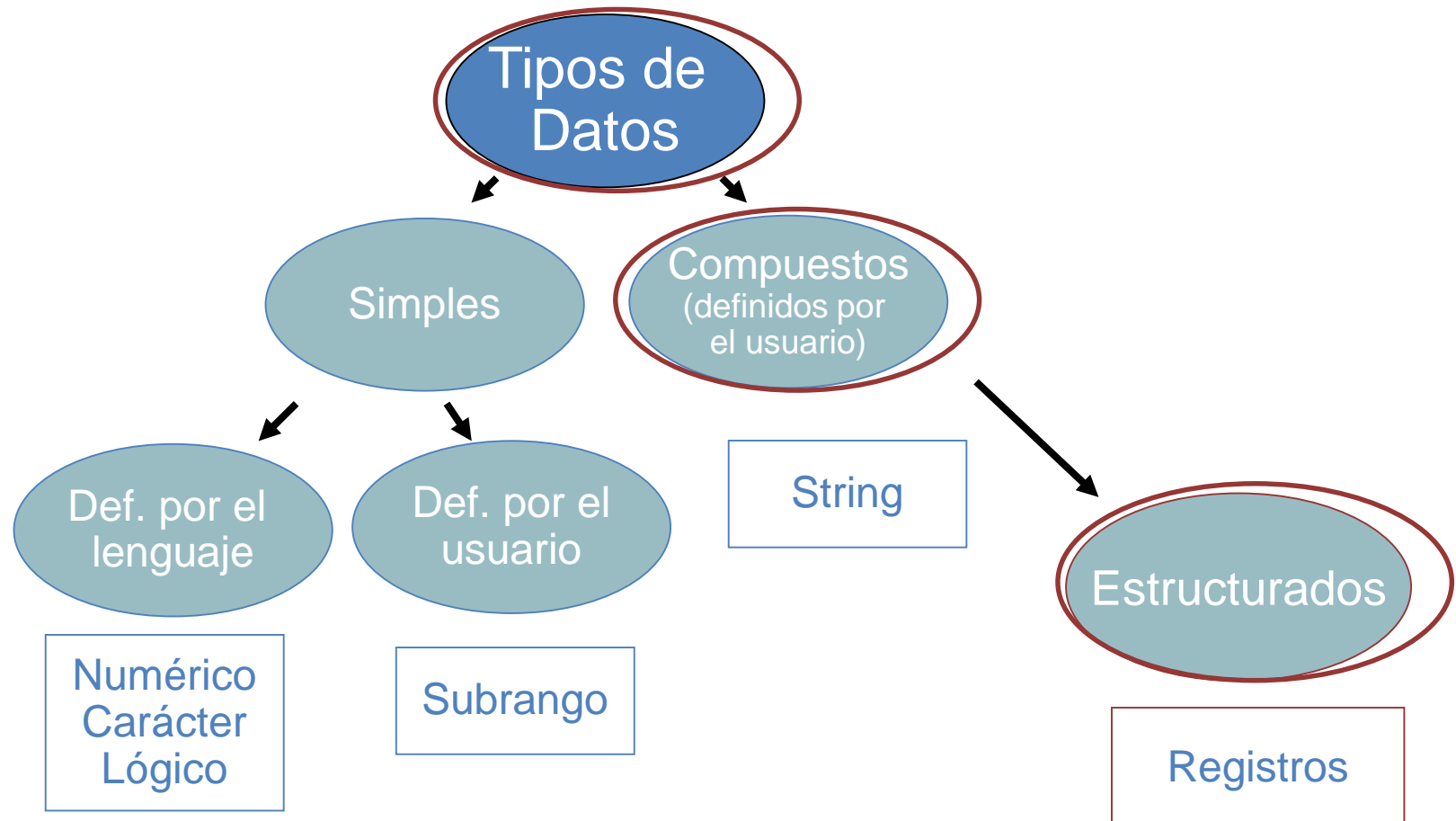
1 Tipo de dato REGISTRO

Concepto

Declaración en Pascal

Ejercitación

Trabajaremos con el tipo de dato estructurado REGISTRO



Tipo de dato Registro

Una de las primeras estructuras de datos básica, con la que vamos a trabajar es la de registro.

El **registro** es uno de los tipos de datos estructurados, que permiten agrupar diferentes clases de datos en una estructura única.

Pensemos en los siguientes ejemplos...

Empleado



Nombre
DNI
Fecha Nac.
NroLegajo
Sexo
Sueldo
Antigüedad

Alumno



Nombre
Nro. Alumno
Datos Personales
Materias que cursa
Materias aprobadas

Mensaje



Origen
Destino
Fecha envío
Mensaje

Producto



Código
Marca
Nombre
Precio
Fecha de vencimiento

Tipo de dato Registro

Una manera natural y lógica de agrupar la información de cada empleado en una sola estructura es declarar un tipo **REGISTRO** asociando el conjunto de datos de cada producto.

Cada dato que compone el Registro se denomina **CAMPO**.

Por lo tanto, se podrá definir al tipo Registro del producto como:

*Campos del
registro*



Código
Nombre
Fecha Vencimiento
Marca
Precio



Tipo de dato Registro



Nombre
Nro. Alumno
Datos Personales
Materias que cursa
Materias aprobadas



Origen
Destino
Fecha envío
Mensaje



Nombre
DNI
Fecha Nac.
NroLegajo
Sexo
Sueldo
Antigüedad

Tipo de dato Registro – Características

Un registro es una estructura de datos que cumple con:

- Los valores pueden ser de diferentes tipos, esto convierte a un registro en una estructura de datos **heterogénea**.
- El almacenamiento ocupado por un registro es fijo, por esto, un registro es una estructura **estática**.
- El **acceso** a sus componentes (campos) es **directo**. Debe referenciarse a través de su nombre.



Nombre
DNI
Fecha Nac.
NroLegajo
Sexo
Sueldo
Antigüedad



Nombre
Nro. Alumno
Datos Personales
Materias que cursa
Materias aprobadas



Origen
Destino
Fecha envío
Mensaje



Código
Marca
Nombre
Precio
Fecha de vencimiento

Tipo de dato Registro – Declaración en Pascal

- Se identifica el nombre del tipo como registro (RECORD).
- Se especifica el **nombre** y **tipo** de los campos individuales que componen el tipo. La lista de campos se encierra entre las palabras claves record y end.
- Cada uno de los campos tiene un identificador. Los campos pueden ser nombrados individualmente, como variables ordinarias.
- Los campos pueden ser de cualquier tipo predefinido o definidos por el usuario (estáticos).

Type

```
nombre = record  
    Campo1: tipoA;  
    Campo2: tipoB;  
    .....  
end;
```

Var

```
nombre-variable:nombre;
```

Tipo de dato Registro – Declaración en Pascal



Código
Marca
Nombre
Precio
Fecha de
vencimiento

Observemos el campo fecha

Type

```
cadena15 = string [15];
```

```
producto= Record
```

```
    codigo: integer;
```

```
    Nombre: cadena15;
```

```
    Marca: cadena15;
```

```
    FechaVenc?????
```

```
    Precio: real
```

```
End;
```

Var

```
    prod: producto;
```


Tipo de dato Registro – Declaración en Pascal



Código
Marca
Nombre
Precio
Fecha de
vencimiento

El tipo fecha puede ser definido como un registro

Type

```
fecha = record
    día: 1..31;
    mes: 1..12;
    año: 1900..2100;
end;
```

Type

```
cadena15 = string [15];
fecha = record
    día: 1..31;
    mes: 1..12;
    año: 1900..2100;
end;
producto= Record
    codigo: integer;
    nombre: cadena15;
    marca: cadena15;
    fechaVenc: fecha;
    Precio: real
End;
```

Var

```
prod: producto;
```

¿Cuál es la ocupación en memoria para prod?

Tipo de dato Registro – Acceso a los campos

- Para acceder a los campos de un registro, se necesita especificar tanto el nombre del registro como el del campo que interesa.
- Esto se denomina calificar al campo.
- En Pascal se hace de la siguiente forma:

Nombre-Registro.nombre-campo

Prod.nombre

Prod.precio

Prod.fechaVenc.dia

prod.marca

Type

```
cadena15 = string [15];  
fecha    = record  
    día: 1..31;  
    mes: 1..12;  
    año: 1900..2100;  
end;  
producto= Record  
    codigo: integer;  
    nombre: cadena15;  
    marca: cadena15;  
    fechaVenc: fecha;  
    Precio: real  
End;
```

Var

```
prod: producto;
```

Operaciones aplicadas a los campos de un registro

- Dado que el Registro es un tipo de dato estructurado, las operaciones deberán aplicarse a cada uno de los campos que lo componen.
- Como los campos de un registro son variables de algún tipo, las operaciones posibles sobre un campo son las permitidas para el tipo de dato correspondiente.

- **Asignación**
- **Comparación**
- **Lectura/escritura**

Type

```
cadena15 = string [15];  
fecha    = record  
    día: 1..31;  
    mes: 1..12;  
    año: 1900..2100;  
end;  
producto= Record  
    codigo: integer;  
    nombre: cadena15;  
    marca: cadena15;  
    fechaVenc: fecha;  
    Precio: real  
End;
```

Var

```
prod: producto;
```

Operaciones aplicadas a los campos de un registro

Asignación

prod.precio := 20

prod.nombre := 'Yerba'

prod.marca := 'SanCor'

prod.fechaVenc.año:=2020

Type

```
cadena15 = string [15];  
fecha = record  
    día: 1..31;  
    mes: 1..12;  
    año: 1900..2100;  
end;  
producto= Record  
    codigo: integer;  
    nombre: cadena15;  
    marca: cadena15;  
    fechaVenc: fecha;  
    Precio: real  
End;
```

Var

```
prod: producto;
```

Operaciones aplicadas a los campos de un registro

Comparación

if prod.precio=20 then

**if (prod.precio > 30) and
 (prod.precio <50) then**

while (prod.marca= 'SanCor') do begin

if prod.fechaVenc.año=2018 then

Type

```
cadena15 = string [15];  
fecha    = record  
    día: 1..31;  
    mes: 1..12;  
    año: 1900..2100;  
end;  
producto= Record  
    codigo: integer;  
    nombre: cadena15;  
    marca: cadena15;  
    fechaVenc: fecha;  
    Precio: real  
End;
```

Var

```
prod: producto;
```

Operaciones aplicadas a los campos de un registro

Lectura

```
Read (prod.marca);  
Read (prod.codigo);  
Read (prod.fechavenc.mes);
```

Escritura

```
write (prod.marca);  
write (prod.nombre);  
write (prod.fechavenc.año);
```

Type


```
cadena15 = string [15];  
fecha    = record  
    día: 1..31;  
    mes: 1..12;  
    año: 1900..2100;  
end;  
producto= Record  
    codigo: integer;  
    nombre: cadena15;  
    marca: cadena15;  
    fechaVenc: fecha;  
    Precio: real  
End;
```

Var

```
prod: producto;
```

Observar que el orden en que se leen los campos del registro podría no coincidir con el orden especificado en la declaración del tipo empleado ¿Por qué?

Operaciones permitidas para el tipo Registro



La única operación permitida entre registros es la de Asignación, **siempre y cuando**, los dos registros estén definidos del mismo tipo.

Type

```
cadena15 = string [15];  
fecha = record  
    día: 1..31;  
    mes: 1..12;  
    año: 1900..2100;  
end;
```

```
producto= Record  
    codigo: integer;  
    nombre: cadena15;  
    Marca: cadena15;  
    FechaVenc: fecha;  
    Precio: real  
End;
```

Var

```
    prod1, prod2: producto;
```

Begin

```
.....  
    prod2 := prod1;  
.....
```

¿Cómo se comparan variables de tipo registro?

No pueden realizarse comparaciones entre registros completos, es decir para saber si dos registros son iguales, se debe evaluar cada uno de los campos.

Type

```
fecha = record
    día: 1..31;
    mes: 1..12;
    año: 1900..2100;
end;
```

Para comparar dos registros puede realizarse una función de la forma:

```
function iguales (f1, f2 : fecha) : boolean;
begin
    iguales:= (f1.dia=f2.dia) and (f1.mes=f2.mes) and (f1.año = f2.año);
end;
```

Comentarios sobre el parámetro de la función...

Ejemplo para la lectura del registro Producto

¿Por qué conviene tener un procedimiento de lectura?

Type

```
cadena15 = string [15];  
fecha = record  
    día: 1..31;  
    mes: 1..12;  
    año: 1900..2100;  
end;  
  
producto= Record  
    codigo: integer;  
    nombre: cadena15;  
    marca: cadena15;  
    FechaVenc: fecha;  
    Precio: real  
End;
```

Var

```
prod: producto;
```

```
Procedure LeerProducto (Var prod: producto);  
begin  
    Readln (prod.codigo);  
    Readln (prod.nombre);  
    Readln (prod.marca);  
    Readln (prod.fechaVenc.dia);  
    Readln (prod.fechaVenc.mes);  
    Readln (prod.fechaVenc.año);  
    Readln (prod.precio);  
end;
```

*¿Podríamos pensar en un
procedimiento de lectura para el
registro fecha?*

Comentarios sobre el parámetro del procedimiento...

Ejercitación



Se leen 100 datos correspondientes a los productos de un supermercado. Obtener un listado con los nombres de los productos con precio entre 25 y 50 pesos e informar la cantidad de productos de marca 'Ala'.

Se puede utilizar la declaración del tipo producto vista anteriormente...

Type

```
cadena15 = string [15];  
fecha    = record  
    día: 1..31;  
    mes: 1..12;  
    año: 1900..2100;  
end;  
  
producto= Record  
    codigo: integer;  
    nombre: cadena15;  
    marca: cadena15;  
    fechaVenc: fecha;  
    precio: real  
End;
```



Se leen 100 datos correspondientes a los productos de un supermercado. Obtener un listado con los nombres de los productos con precio entre 25 y 50 pesos e informar la cantidad de productos de marca 'Ala'.

```
Inicializar cantidad
Repetir 100
  Leer producto
  si precio entre 25 y 50 entonces
    muestro nombre
  si marca = Ala then incremento cantidad
Fin
Mostrar cantidad
```

```
Procedure LeerFecha (Var f:fecha);
begin
  Readln (f.dia);
  Readln (f.mes);
  Readln (f.año);
end;
```

```
Procedure LeerProducto (Var prod: producto);
begin
  Readln (prod.codigo);
  Readln (prod.nombre);
  Readln (prod.marca);
  LeerFecha (prod.fechaVenc);
  Readln (prod.precio);
end;
```

Program ejemplo;

Type

```
cadena15 = string [15];
fecha = record
  día: 1..31;
  mes: 1..12;
  año: 1900..2100;
end;
producto = Record
  codigo: integer;
  nombre: cadena15;
  marca: cadena15;
  fechaVenc: fecha;
  Precio: real
End;
```

{implementación módulo LeerFecha}

{implementación módulo LeerProducto}

```
var prod:producto; cant, i: integer;
```

```
begin {Programa principal}
```

```
  cant:= 0;
```

```
  for i:= 1 to 100 do begin
```

```
    LeerProducto (prod);
```

```
    if (prod.precio >=25 and prod.precio <=50)
```

```
      then Writeln (prod.nombre);
```

```
    if (prod.marca = 'Ala') then cant := cant +1;
```

```
  end;
```

```
  write (cant)
```

```
end.
```

Ejercitación



Se leen datos correspondientes a los productos de un supermercado. La lectura finaliza con código igual a -1. Obtener un listado con los nombres de los productos con precio entre 25 y 50 pesos e informar la cantidad de productos de marca 'Ala'.

Se puede utilizar la declaración del tipo producto vista anteriormente...

Type

```
cadena15 = string [15];  
fecha    = record  
    día: 1..31;  
    mes: 1..12;  
    año: 1900..2100;  
end;
```

```
producto= Record  
    codigo: integer;  
    nombre: cadena15;  
    marca: cadena15;  
    fechaVenc: fecha;  
    precio: real  
End;
```

Se leen datos correspondientes a los productos de un supermercado. La lectura finaliza con código igual a -1. Obtener un listado con los nombres de los productos con precio entre 25 y 50 y marca 'Ala'.

```
Inicializar cantidad
Leer producto
Mientras producto <> -1
    si precio entre 25 y 50 entonces
        muestro nombre
    si marca = Ala then incremento cantidad
    leer otro producto
Fin
Mostrar cantidad
```

```
Procedure LeerFecha (Var f: fecha);
begin
    Readln (f.dia);
    Readln (f.mes);
    Readln (f.año);
end;
```

```
Procedure LeerProducto2 (Var prod: producto);
begin
    Readln (prod.codigo);
    if (prod.código <> -1) then begin
        Readln (prod.nombre);
        Readln (prod.marca);
        LeerFecha (prod.fechaVenc);
        Readln (prod.precio);
    end;
end;
```

Program ejemplo;

Type

```
cadena15 = string [15];
fecha = record
    día: 1..31;
    mes: 1..12;
    año: 1900..2100;
end;
producto = Record
    codigo: integer;
    nombre: cadena15;
    marca: cadena15;
    fechaVenc: fecha;
    Precio: real;
End;
```

{implementación módulo LeerFecha}

{implementación módulo LeerProducto}

```
var prod: producto; cant: integer;
```

```
begin {Programa principal}
```

```
    cant:= 0;
```

```
    LeerProducto2 (prod);
```

```
    While (prod.codigo <> -1) do begin
```

```
        if (prod.precio >=25 and prod.precio <=50)
```

```
            then Writeln (prod.nombre);
```

```
        if (prod.marca = 'Ala') then cant := cant +1;
```

```
        LeerProducto2 (prod);
```

```
    end;
```

```
    write (cant)
```

```
end.
```