VARIABLES Y CONSTANTES.

- PL/SQL permite declarar e inicializar variables y/o constantes.
- Las variables pueden ser de cualquiera de los tipos vistos en SQL o de otros tipos que veremos.
- Todas las variables han de ser declaradas antes de ser utilizadas
- Las variables se pueden utilizar:
  - Para almacenar temporalmente datos de entrada hasta que ya no se necesiten
  - Para realizar operaciones sobre los datos
  - Facilidad de mantenimiento: %ROWTYPE,%TYPE
- Para declarar una variable utilizaremos el siguiente formato:
  Nombre_variable [CONSTANT] tipo [NOT NULL]
  [{:=| DEFAULT } expresión];
- Las palabras clave NOT NULL especifican que la variable no podrá tomar valor nulo.
- La palabra clave DEFAULT permite inicializar una variable a la vez que se define.
- Expresión podrá ser un literal, otra variable o el resultado de una expresión que puede incluir operadores y funciones.
- Para los identificadores de las variables se deben utilizar las mismas reglas que para los objetos de SQL
  - Longitud máxima: 30 caracteres
  - Letras, números y caracteres solo $, _, #
  - El nombre debe empezar por una letra
  - No puede ser una palabra reservada de Oracle
- Para asignar valores por defecto podemos utilizar indistintamente := o DEFAULT
- Una variable no inicializada queda con valor NULL hasta asignarle valor en la ejecución
- Las constantes y variables declaradas como NOT NULL deben ser inicializadas
- Dos objetos pueden tener el mismo nombre siempre y cuando estén declarados en distintos bloques
- No se debe poner el mismo nombre a un identificador de variable que a la columna de la que vaya a recibir los datos. Esto evita confusiones y facilita el mantenimiento del software

Ejemplos:

```
importe    NUMBER (9);
nombre     VARCHAR2(20) NOT NULL;
descuento NUMBER(4,2) NOT NULL DEFAULT 10.27;
nombre char(20) NOT NULL := `MIGUEL';
```

casado BOOLEAN DEFAULT FALSE;

- Podemos utilizar el atributo %TYPE para dar a una variable el tipo de dato de otra variable o de una columna de la base de datos.

Nom_variable tabla.columna%type;

Ejemplo:

importe NUMBER (9);
total importe %TYPE; → Declarada total como NUMBER(9)
nuevonombre emp.nombre%TYPE -> Declara nuevonombre del mismo tipo que la columna nombre de emp

- La principal ventaja del uso de este atributo es la facilidad de mantenimiento de los programas, ya que ante cualquier cambio de tipos en la base de datos no hay que hacer ningún cambio en el código Pl/SQL

- %TYPE también puede utilizarse para declara una variable del mismo tipo que otra declarada previamente.

- **Si una columna tiene la restricción NOT NULL, la variable definida de ese tipo mediante el atributo %TYPE, no asume dicha restricción, pudiendo, por lo tanto, ponerla a NULL, a no ser que se especifique lo contrario en la declaración.**

Ejemplo: planta vecinos.piso %TYPE NOT NULL;
Declarada como NUMBER(2) y no
admitiendo valores nulos.

- También podemos utilizar %ROWTYPE que crea una variable tipo registro para cargar los valores de toda una fila.
Mifila emp%ROWTYPE

Variable tabla%ROWTYPE;

- Se pueden declarar constantes utilizando la palabra reservada CONSTANT, pero teniendo en cuenta que la constante se debe inicializar en el momento de la declaración.

- El valor de una constante, una vez determinada no modificarse. La declaración de constantes facilita la modificación de programas que contienen datos constantes.

- La declaración de una constante es igual a la de una variable, salvo que hay que utilizar CONSTANT e inicializarla.

Ejemplo: iva CONSTANT NUMBER(2) DEFAULT 16;

- En el ejemplo anterior, si no declaramos la constante 'iva', un cambio en el porcentaje del impuesto obligaría a cambiarlo en todas las expresiones donde apareciese.

- Cualquier variable o constante debe ser declarada antes de ser referenciada en otra declaración o sentencia, ya que en caso contrario se produciría un error.

  Ejemplo: total importe %TYPE; → Error
  importe NUMBER(9,2);