17 - Tipo de dato (texto)

Ya explicamos que al crear una tabla debemos elegir la estructura adecuada, esto es, definir los campos y sus tipos más precisos, según el caso.

Para almacenar TEXTO usamos cadenas de caracteres.
Las cadenas se colocan entre comillas simples.
Podemos almacenar letras, símbolos y dígitos con los que no se realizan operaciones matemáticas, por ejemplo, códigos de identificación, números de documentos, números telefónicos.

Tenemos los siguientes tipos:

1. **varchar(x):** define una cadena de caracteres de longitud variable en la cual determinamos el máximo de caracteres con el argumento "x" que va entre paréntesis.
   Si se omite el argumento coloca 1 por defecto. Su rango va de 1 a 8000 caracteres.
2. **char(x):** define una cadena de longitud fija determinada por el argumento "x". Si se omite el argumento coloca 1 por defecto. Su rango es de 1 a 8000 caracteres.
   Si la longitud es invariable, es conveniente utilizar el tipo char; caso contrario, el tipo varchar.
   Ocupa tantos bytes como se definen con el argumento "x".
   "char" viene de character, que significa caracter en inglés.
3. **text:** guarda datos binarios de longitud variable, puede contener hasta 2000000000 caracteres. No admite argumento para especificar su longitud.
4. **nvarchar(x):** es similar a "varchar", excepto que permite almacenar caracteres Unicode, su rango va de 0 a 4000 caracteres porque se emplean 2 bytes por cada caracter.
5. **nchar(x):** es similar a "char" excpeto que acepta caracteres Unicode, su rango va de 0 a 4000 caracteres porque se emplean 2 bytes por cada caracter.
6. **ntext:** es similar a "text" excepto que permite almacenar caracteres Unicode, puede contener hasta 1000000000 caracteres. No admite argumento para especificar su longitud.

En general se usarán los 3 primeros.

Si intentamos almacenar en un campo una cadena de caracteres de mayor longitud que la definida, aparece un mensaje indicando tal situación y la sentencia no se ejecuta.

Por ejemplo, si definimos un campo de tipo varchar(10) y le asignamos la cadena 'Aprenda PHP' (11 caracteres), aparece un mensaje y la sentencia no se ejecuta.

Si ingresamos un valor numérico (omitiendo las comillas), lo convierte a cadena y lo ingresa como tal.

Por ejemplo, si en un campo definido como varchar(5) ingresamos el valor 12345, lo toma como si hubiésemos tipeado '12345', igualmente, si ingresamos el valor 23.56, lo convierte a '23.56'. Si el valor numérico, al ser convertido a cadena supera la longitud definida, aparece un mensaje de error y la sentencia no se ejecuta.

Es importante elegir el tipo de dato adecuado según el caso, el más preciso.

Para almacenar cadenas que varían en su longitud, es decir, no todos los registros tendrán la misma longitud en un campo determinado, se emplea "varchar" en lugar de "char".

Por ejemplo, en campos que guardamos nombres y apellidos, no todos los nombres y apellidos tienen la misma longitud.

Para almacenar cadenas que no varían en su longitud, es decir, todos los registros tendrán la misma longitud en un campo determinado, se emplea "char".

Por ejemplo, definimos un campo "codigo" que constará de 5 caracteres, todos los registros tendrán un código de 5 caracteres, ni más ni menos.

Para almacenar valores superiores a 8000 caracteres se debe emplear "text".

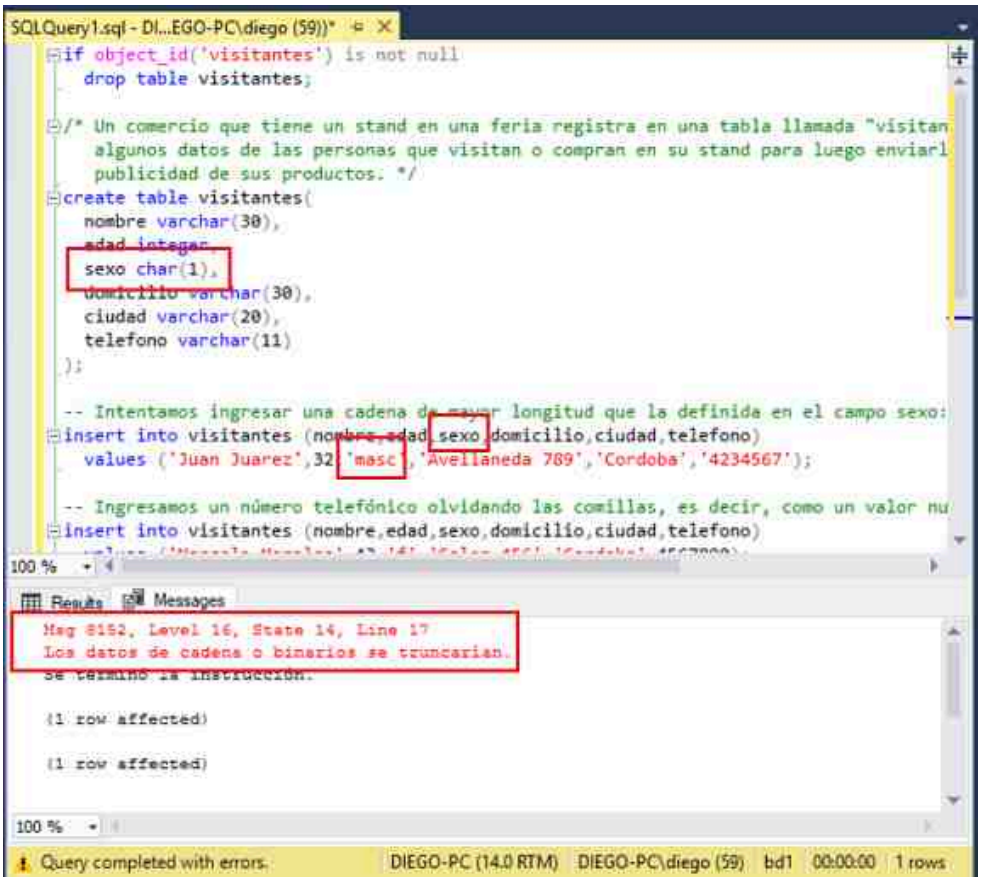
```
Tipo            Bytes de almacenamiento
_____________________________________
varchar(x)      0 a 8K
char(x)         0 a 8K
text            0 a 2GB

nvarchar(x)     0 a 8K
nchar(x)        0 a 8K
ntext           0 a 2GB
```

Servidor de SQL Server instalado en forma local.

Ingresemos el siguiente lote de comandos en el SQL Server Management Studio:

```
if object_id('visitantes') is not null
  drop table visitantes;

/* Un comercio que tiene un stand en una feria registra en una tabla llamada "visitantes"
   algunos datos de las personas que visitan o compran en su stand para luego enviarle
   publicidad de sus productos. */
create table visitantes(
  nombre varchar(30),
  edad integer,
  sexo char(1),
  domicilio varchar(30),
  ciudad varchar(20),
  telefono varchar(11)
);

go

-- Intentamos ingresar una cadena de mayor longitud que la definida
-- en el campo sexo (se genera un error):
insert into visitantes (nombre,edad,sexo,domicilio,ciudad,telefono)
  values ('Juan Juarez',32,'masc','Avellaneda 789','Cordoba','4234567');

-- Ingresamos un número telefónico olvidando las comillas, es decir,
-- como un valor numérico (lo transforma a cadena):
insert into visitantes (nombre,edad,sexo,domicilio,ciudad,telefono)
  values ('Marcela Morales',43,'f','Colon 456','Cordoba',4567890);

select * from visitantes;
```

Tenemos como resultado:


SQLQuery1.sql - DI...EGO-PC\diego (59))*
if object_id('visitantes') is not null
drop table visitantes;
/* Un comercio que tiene un stand en una feria registra en una tabla llamada "visitan
algunos datos de las personas que visitan o compran en su stand para luego enviarl
publicidad de sus productos. */
create table visitantes(
nombre varchar(30),
edad integer,
sexo char(1),
domicilio varchar(30),
ciudad varchar(20),
telefono varchar(11)
);
-- Intentamos ingresar una cadena de mayor longitud que la definida en el campo sexo:
insert into visitantes (nombre,edad,sexo,domicilio,ciudad,telefono)
values ('Juan Juarez',32,'masc','Avellaneda 789','Cordoba','4234567');
-- Ingresamos un número telefónico olvidando las comillas, es decir, como un valor nu
insert into visitantes (nombre,edad,sexo,domicilio,ciudad,telefono)
100 %
Results
Messages
Msg 8152, Level 16, State 14, Line 17
Los datos de cadena o binarios se truncarían.
Se terminó la instrucción.
(1 row affected)
(1 row affected)
100 %
Query completed with errors.
DIEGO-PC (14.0 RTM)
DIEGO-PC\diego (59)
bd1
00:00:00
1 rows