| **Campo con atributo Identity** |  |
| --- | --- |

Un campo numérico puede tener un atributo extra "identity". Los valores de un campo con este atributo genera valores secuenciales que se inician en 1 y se incrementan en 1 automáticamente.

Se utiliza generalmente en campos correspondientes a códigos de identificación para generar valores únicos para cada nuevo registro que se inserta.

Sólo puede haber un campo "identity" por tabla.

Para que un campo pueda establecerse como "identity", éste debe ser entero (también puede ser de un subtipo de entero o decimal con escala 0, tipos que estudiaremos posteriormente).

Para que un campo genere sus valores automáticamente, debemos agregar el atributo "identity" luego de su definición al crear la tabla:

create table libros(

codigo int identity,

titulo varchar(40) not null,

autor varchar(30),

editorial varchar(15),

precio float

);

Cuando un campo tiene el atributo "identity" no se puede ingresar valor para él, porque se inserta automáticamente tomando el último valor como referencia, o 1 si es el primero.

Para ingresar registros omitimos el campo definido como "identity", por ejemplo:

insert into libros (titulo,autor,editorial,precio)

values('El aleph','Borges','Emece',23);

Este primer registro ingresado guardará el valor 1 en el campo correspondiente al código.

Si continuamos ingresando registros, el código (dato que no ingresamos) se cargará automáticamente siguiendo la secuencia de autoincremento.

No está permitido ingresar el valor correspondiente al campo "identity", por ejemplo:

insert into libros (codigo,titulo,autor,editorial,precio)

values(5,'Martin Fierro','Jose Hernandez','Paidos',25);

generará un mensaje de error.

"identity" permite indicar el valor de inicio de la secuencia y el incremento, pero lo veremos posteriormente.

Un campo definido como "identity" generalmente se establece como clave primaria.

Un campo "identity" no es editable, es decir, no se puede ingresar un valor ni actualizarlo.

Un campo de identidad no permite valores nulos, aunque no se indique especificamente. Si ejecutamos el procedimiento "sp\_columns()" veremos que en el campo "codigo" en la columna "TYPE\_NAME" aparece "int identity" y en la columna "IS\_NULLABLE" aparece "NO".

Los valores secuenciales de un campo "identity" se generan tomando como referencia el último valor ingresado; si se elimina el último registro ingresado (por ejemplo 3) y luego se inserta otro registro, SQL Server seguirá la secuencia, es decir, colocará el valor "4".

**Servidor de SQL Server instalado en forma local.**

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

if object\_id('libros') is not null

drop table libros;

create table libros(

codigo int identity,

titulo varchar(40) not null,

autor varchar(30),

editorial varchar(15),

precio float

);

go

insert into libros (titulo,autor,editorial,precio)

values('El aleph','Borges','Emece',23);

select \* from libros;

insert into libros (titulo,autor,editorial,precio)

values('Uno','Richard Bach','Planeta',18);

insert into libros (titulo,autor,editorial,precio)

values('Aprenda PHP','Mario Molina','Siglo XXI',45.60);

insert into libros (titulo,autor,editorial,precio)

values('Alicia en el pais de maravillas','Lewis Carroll','Paidos',15.50);

select \* from libros;

-- Intentemos ingresar un valor para el campo "codigo" (genera error):

insert into libros (codigo,titulo,autor,editorial,precio)

values(5,'Martin Fierro','Jose Hernandez','Paidos',25);

-- Intentemos cambiar el valor de código de un registro (genera error):

update libros set codigo=9

where titulo='Uno';

exec sp\_columns libros;

-- Eliminemos el último registro:

delete from libros

where autor='Lewis Carroll';

-- Ingresamos un quinto registro

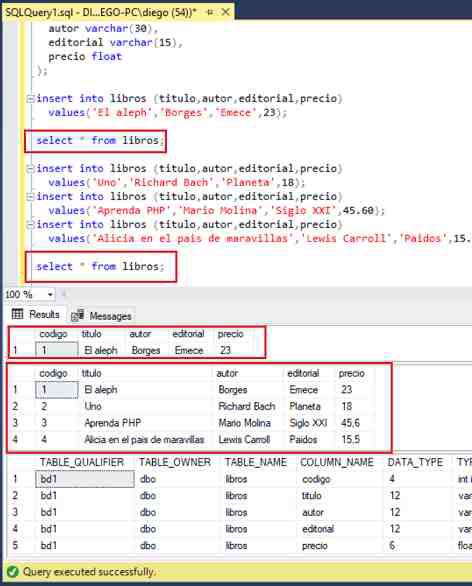
insert into libros (titulo, autor, editorial, precio)

values('Martin Fierro','Jose Hernandez','Paidos',25);

-- El campo código se guardó el valor secuencial sin considerar que el valor "4" ya no existe:

select \* from libros;

Tenemos como resultado (previo comentamos los comandos que generan error):



**Primer problema:**

Una farmacia guarda información referente a sus medicamentos en una tabla llamada "medicamentos".

1- Elimine la tabla,si existe:

if object\_id('medicamentos') is not null

drop table medicamentos;

2- Cree la tabla con un campo "codigo" que genere valores secuenciales automáticamente:

create table medicamentos(

codigo int identity,

nombre varchar(20) not null,

laboratorio varchar(20),

precio float,

cantidad integer

);

3- Visualice la estructura de la tabla "medicamentos":

exec sp\_columns medicamentos;

4- Ingrese los siguientes registros:

insert into medicamentos (nombre, laboratorio,precio,cantidad)

values('Sertal','Roche',5.2,100);

insert into medicamentos (nombre, laboratorio,precio,cantidad)

values('Buscapina','Roche',4.10,200);

insert into medicamentos (nombre, laboratorio,precio,cantidad)

values('Amoxidal 500','Bayer',15.60,100);

5- Verifique que SQL Server generó valores para el campo "código" de modo automático:

select \* from medicamentos;

6- Intente ingresar un registro con un valor para el campo "codigo"

7- Intente actualizar un valor de código (aparece un mensaje de error)

8- Elimine el registro con codigo "3" (1 registro eliminado)

9- Ingrese un nuevo registro:

insert into medicamentos (nombre, laboratorio,precio,cantidad)

values('Amoxilina 500','Bayer',15.60,100);

10- Seleccione todos los registros para ver qué valor guardó SQL Server en el campo código:

select \* from medicamentos;

| **Tipo de dato (fecha y hora)** |  |
| --- | --- |

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 valores de tipo FECHA Y HORA SQL Server dispone de dos tipos:

1) datetime: puede almacenar valores desde 01 de enero de 1753 hasta 31 de diciembre de 9999.

2) smalldatetime: el rango va de 01 de enero de 1900 hasta 06 de junio de 2079.

Las fechas se ingresan entre comillas simples.

Para almacenar valores de tipo fecha se permiten como separadores "/", "-" y ".".

SQL Server reconoce varios formatos de entrada de datos de tipo fecha. Para establecer el orden de las partes de una fecha (dia, mes y año) empleamos "set dateformat". Estos son los formatos:

-mdy: 4/15/96 (mes y día con 1 ó 2 dígitos y año con 2 ó 4 dígitos),

-myd: 4/96/15,

-dmy: 15/4/1996

-dym: 15/96/4,

-ydm: 96/15/4,

-ydm: 1996/15/4,

Para ingresar una fecha con formato "día-mes-año", tipeamos:

set dateformat dmy;

El formato por defecto es "mdy".

Todos los valores de tipo "datetime" se muestran en formato "año-mes-día hora:minuto:segundo .milisegundos", independientemente del formato de ingreso que hayamos seteado.

Podemos ingresar una fecha, sin hora, en tal caso la hora se guarda como "00:00:00". Por ejemplo, si ingresamos '25-12-01' (año de 2 dígitos), lo mostrará así: '2001-12-25 00:00:00.000'.

Podemos ingresar una hora sin fecha, en tal caso, coloca la fecha "1900-01-01". Por ejemplo, si ingresamos '10:15', mostrará '1900-01-01 10:15.000'.

Podemos emplear los operadores relacionales vistos para comparar fechas.

Tipo Bytes de almacenamiento

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

datetime 8

smalldatetime 4

**Servidor de SQL Server instalado en forma local.**

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

if object\_id('empleados') is not null

drop table empleados;

create table empleados(

nombre varchar(20),

documento char(8),

fechaingreso datetime

);

go

-- Seteamos el formato de la fecha para que guarde día, mes y año:

set dateformat dmy;

insert into empleados values('Ana Gomez','22222222','12-01-1980');

insert into empleados values('Bernardo Huerta','23333333','15-03-81');

insert into empleados values('Carla Juarez','24444444','20/05/1983');

insert into empleados values('Daniel Lopez','25555555','2.5.1990');

-- Note que el formato de visualización es "y-m-d".

select \* from empleados;

-- Mostramos los datos de los empleados cuya fecha de ingreso es anterior a '01-01-1985':

select \* from empleados where fechaingreso<'01-01-1985';

-- Actualizamos el nombre a "Maria Carla Juarez'

-- del empleado cuya fecha de ingreso es igual a '20/05/1983':

update empleados set nombre='Maria Carla Juarez' where fechaingreso='20.5.83';

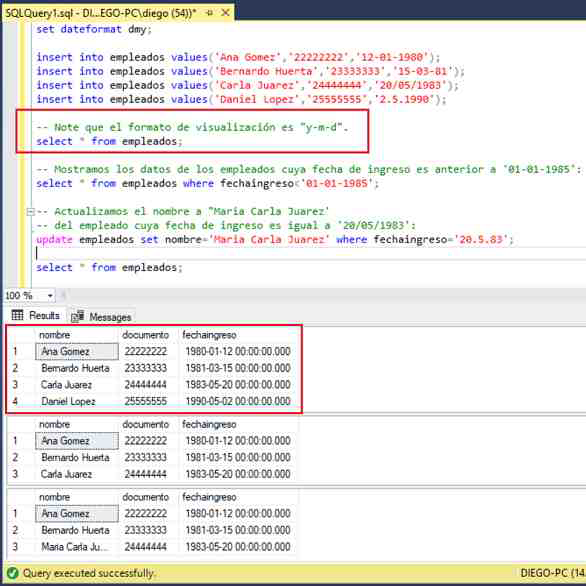
select \* from empleados;

-- Borramos los empleados cuya fecha de ingreso es distinta a '20.5.83':

delete from empleados where fechaingreso<>'20/05/1983';

select \* from empleados;

Tenemos como resultado:



**Primer problema:**

Una facultad almacena los datos de sus alumnos en una tabla denominada "alumnos".

1- Elimine la tabla, si existe:

if object\_id('alumnos') is not null

drop table alumnos;

2- Cree la tabla eligiendo el tipo de dato adecuado para cada campo:

create table alumnos(

apellido varchar(30),

nombre varchar(30),

documento char(8),

domicilio varchar(30),

fechaingreso datetime,

fechanacimiento datetime

);

3- Setee el formato para entrada de datos de tipo fecha para que acepte valores "día-mes-año":

set dateformat 'dmy';

4- Ingrese un alumno empleando distintos separadores para las fechas:

insert into alumnos values('Gonzalez','Ana','22222222','Colon 123','10-08-1990','15/02/1972');

5- Ingrese otro alumno empleando solamente un dígito para día y mes y 2 para el año:

insert into alumnos values('Juarez','Bernardo','25555555','Sucre 456','03-03-1991','15/02/1972');

6- Ingrese un alumnos empleando 2 dígitos para el año de la fecha de ingreso y "null" en

"fechanacimiento":

insert into alumnos values('Perez','Laura','26666666','Bulnes 345','03-03-91',null);

7- Intente ingresar un alumno con fecha de ingreso correspondiente a "15 de marzo de 1990" pero en

orden incorrecto "03-15-90":

insert into alumnos values('Lopez','Carlos','27777777','Sarmiento 1254','03-15-1990',null);

aparece un mensaje de error porque lo lee con el formato día, mes y año y no reconoce el mes 15.

8- Muestre todos los alumnos que ingresaron antes del '1-1-91'.

1 registro.

9- Muestre todos los alumnos que tienen "null" en "fechanacimiento":

select \*from alumnos where fechanacimiento is null;

1 registro.

10- Intente ingresar una fecha de ingreso omitiendo los separadores:

insert into alumnos values('Rosas','Romina','28888888','Avellaneda 487','03151990',null);

No lo acepta.

11- Setee el formato de entrada de fechas para que acepte valores "mes-dia-año".

12- Ingrese el registro del punto 7.