Operaciones de mantenimiento de datos

En este capítulo veremos las instrucciones SQL para ingresar nuevos datos (INSERT), eliminar registros (DELETE) y modificar (UPDATE) los datos existentes.

Esta página se ha dejado en blanco intencionalmente.

Capítulo 5
Operaciones de mantenimiento de datos

Contenido

Esta página se ha dejado en blanco intencionalmente.

Operaciones de mantenimiento de datos

Una vez creadas las tablas de la base de datos se debe cargar la data, lo que normalmente se lleva a cabo desde una aplicación cliente que contiene formularios para que el usuario pueda cargar los datos con facilidad y comodidad. Estos formularios enmascaran las sentencias SQL que se deben ejecutar para llevar a cabo la tarea de mantenimiento de datos.

Inserción de filas (la instrucción INSERT)

Sintáxis

```
INSERT [INTO] nombre_tabla[ ( lista_de_columnas ) ]
   VALUES( lista_de_valores )
```

- **lista_de_columnas** es la relación de columnas en las que se almacenarán los valores especificados en *lista_de_valores*.
- **lista_de_valores** es la relación de los valores a almacenar en la fila a insertar.

Los elementos de ambas listas van separados por comas.

Ejercicio 32: Inserción de una fila con lista de valores completa

1. En su query, ejecute las siguientes instrucciones:

```
USE ExtPro
go

INSERT INTO Curso(nombreCurso, idCurso,
   profesor, numVacantes, numMatriculados, precioCurso)
   VALUES('Linux Nivel I - Grupo A', 'LX1A',
   'Gonzáles Villegas, Julio', 15, 0, 250)
go
```

En este caso, cuando se entrega valores para cada una de las columnas de una tabla, el orden de los valores en la lista de valores debe coincidir con el orden de las columnas en la lista de columnas.

2. También se puede insertar una fila con lista de valores completa usando la siguiente sintáxis:

```
INSERT INTO Curso
   VALUES('EX1A', 'MS Excel 2003 Nivel I - Grupo A',
   15, 0, 'Quispe Tineo, César', 150)
go
```

En este caso, cuando en la instrucción no se establece la lista de columnas, pero se está entregando la lista de valores completa, los valores de la lista se ordenan según el orden natural de las columnas en la tabla.

3. También es posible insertar una fila con lista de valores completa que contengan valores NULL, siempre y cuando la columna correspondiente los permita.

```
INSERT INTO Curso
   VALUES('EX1B', 'MS Excel 2003 Nivel I - Grupo B',
   15, 0, NULL, 150)
go
```

En este caso, no se conoce el nombre del profesor, pero como la columna permite nulos registramos NULL como valor de dicha columna.

Ejercicio 33: Inserción de una fila con lista de valores incompleta

1. Cuando la lista de valores está incompleta es obligatorio establecer en qué columna va cada valor de la lista, por lo que la lista de columnas es obligatoria en la instrucción INSERT.

```
INSERT INTO Curso(idCurso, nombreCurso, precioCurso,
   numMatriculados, numVacantes)
   VALUES('EX2A', 'MS Excel 2003 Nivel II - Grupo A',
   200, 0, 15)
go
```

   El orden de las columnas en la lista de columnas, y el orden de los valores en la lista de valores se deben corresponder.

Uso de DEFAULT VALUES

Si cada una de las columnas de la tabla tiene definido un valor por defecto, o permite valores nulos, o tiene la propiedad IDENTITY, podemos insertar filas sin tener que especificar los valores a insertar tal como se muestra en la sintáxis siguiente:

```
INSERT [INTO] nombre_tabla DEFAULT VALUES
```

Esta orden inserta una fila de la siguiente manera:

- Si la columna permite nulos, inserta NULL en dicha columna.
- Si la columna tiene definido un valor por defecto, inserta el valor por defecto.
- Si la columna tiene la propiedad IDENTITY, inserta el valor de identidad correspondiente.

La propiedad IDENTITY

Permite definir una columna numérica entera en la que el valor de la columna es autogenerado a medida que se van insertando filas. Es útil cuando se desea emplear claves primarias numéricas.

Ejercicio 34: Uso de la propiedad IDENTITY

1. Cree una tabla que tenga la propiedad IDENTITY.

```
USE ExtPro
go

CREATE TABLE pruebaIdentidad(
   codigo int IDENTITY(1,1) PRIMARY KEY,
   nombre varchar(15) not null,
   telefono varchar(15) null )
go
```

2. Inserte filas en la tabla ejecutando las siguientes instrucciones:

```
INSERT INTO pruebaIdentidad(nombre, telefono)
   VALUES('Gustavo', '99981234')

INSERT INTO pruebaIdentidad(nombre, telefono)
   VALUES('Ricardo', '2211234')

INSERT INTO pruebaIdentidad(nombre, telefono)
   VALUES('Gloria', '3493212')
go

SELECT * FROM pruebaIdentidad
go
```

En IDENTITY(1,1), el primer argumento establece el primer valor de identidad a generar, y el segundo argumento es el incremento a utilizar para generar los siguientes valores.

Ejercicio 35: Inserción de la fecha del sistema

1. La función **getdate()** entrega la fecha y hora del sistema. La puede utilizar para insertar la fecha y hora del sistema en una columna de tipo fecha-hora.

```
INSERT INTO Matricula(idCurso, idAlumno, fecha,
   precio, numCuotas)
   VALUES('PB1A', 1002, getdate(), 250, 3)
go

SELECT * FROM Matricula
go
```

Ejercicio 36: Inserción de un valor de fecha específico

Para enviar un dato fecha al servidor, la fecha se envía como una cadena con formato de fecha. Por lo general, cuando la conexión es a un servidor SQL remoto, no sabemos cuál es el formato de fecha predeterminado que está utilizando el servidor, por lo que en ocasiones podemos tener problemas con la manipulación de las fechas.

1. Cree una tabla para hacer algunas pruebas con las fechas:

```
CREATE TABLE PruebaFechas(
   fecha datetime not null )
go
```

2. Ahora, inserte la fecha 26 de noviembre de 1980 mediante la siguiente instrucción:

```
-- Prueba 1: insertar 26 de noviembre de 1980
INSERT INTO PruebaFechas VALUES( '26 Nov 1980' )
go

SELECT * FROM PruebaFechas
go
```

Observe que la fecha se registró correctamente.

3. Ahora, inserte la fecha 3 de enero de 1979

```
-- Prueba 2: insertar 3 de enero de 1979
INSERT INTO PruebaFechas VALUES( '03/01/1979' )
go

SELECT * FROM PruebaFechas
go
```

Note que se ha insertado incorrectamente. La fecha registrada es 1 de marzo de 1979. El servidor tiene como formato predeterminado de fecha 'mm/dd/aaaa'.

4. Ahora, inserte la fecha 26 de noviembre de 1980 con la siguiente instrucción:

```
-- Prueba 3: insertar 26 de noviembre de 1980
INSERT INTO PruebaFechas VALUES( '26/11/1980' )
go
```

5. Se genera el siguiente mensaje de error:

```
Msg 242, Level 16, State 3, Line 1
The conversion of a char data type to a datetime data type
resulted in an out-of-range datetime value.
The statement has been terminated.
```

El valor de fecha se ha leido como el día 11 del mes 26 del año 1980, lo que produce un valor de fecha fuera de rango (fecha inválida).

Cuando se envía fechas al servidor se recomienda informar al servidor en qué formato se le está enviando las fechas para que las lea correctamente.

6. Infórmele al servidor acerca del formato de sus fechas:

```
SET DATEFORMAT dmy
go
```

7. Ahora, trate de insertar nuevamente la fecha 26 de noviembre de 1980 usando el formato acostumbrado para nosotros:

```
-- Prueba 4: insertar 26 de noviembre de 1980
INSERT INTO PruebaFechas VALUES( '26/11/1980' )
go

SELECT * FROM PruebaFechas
go
```

La fecha se inserta correctamente.

Actualización de datos (la instrucción UPDATE)

Sintáxis

```
UPDATE nombre_tabla
   SET    columnaX = expresiónX,
          columnaP = expresiónP, ...
   [ WHERE condición_de_las_filas_a_actualizar ]
```

- **columnaX**, **columnaP** son las columnas cuyo contenido se actualizará.
- **expresiónX**, **expresiónP** establecen los nuevos valores a almacenar en las columnas *columnaX* y *columnaP* respectivamente.
- **condición_de_las_filas_a_actualizar** es una expresión lógica que determina las filas en las que la actualización se debe llevar a cabo.

Ejercicio 37: Uso de UPDATE

1. Ejecute las siguientes instrucciones en su query:

```
USE ExtPro
go

SELECT * FROM Curso
go
```

2. Note que el curso EX1B no tiene asignado profesor. Asignaremos un profesor para dicho curso.

```
UPDATE Curso
   SET profesor = 'Chuquín Espinoza, William'
   WHERE idCurso = 'EX1B'
go

SELECT * FROM Curso
go
```

Eliminación de filas (la instrucción DELETE)

Sintáxis

```
DELETE [FROM] nombre_tabla
   [ WHERE condición_de_las_filas_a_eliminar ]
```

- **condición_de_las_filas_a_eliminar** es una expresión lógica que determina las filas en las que la eliminación se debe llevar a cabo.

Ejercicio 38: Uso de DELETE

1. Ejecute las siguientes instrucciones en su query:

```
USE ExtPro
go

SELECT * FROM Matricula
go
```

2. Elimine la matrícula del alumno 1001 en el curso PB1A:

```
DELETE FROM Matricula
   WHERE idAlumno = 1001 AND idCurso = 'PB1A'
go

SELECT * FROM Matricula
go
```

Eliminación de todas las filas de una tabla (la instrucción TRUNCATE TABLE)

Si bien la sentencia DELETE utilizada sin la cláusula WHERE permite eliminar todas las filas de una tabla, se recomienda el uso de TRUNCATE TABLE porque esta instrucción se ejecuta mas rápido debido a que no registra la eliminación de cada fila en el log de transacciones de la base de datos como lo hace DELETE. La sintáxis es:

```
TRUNCATE TABLE nombre_tabla
```