Creación de tablas

La creación de tablas se realiza mediante la sentencia *CREATE TABLE*. La sintaxis de esta sentencia es la siguiente:

```
CREATE TABLE <table_name > (
<column_name > <type > [< default value >] [< column constraints >],
: : :
<column_name > <type > [< default value >] [< column constraints >],
<table constraint >,
: : :
<table constraint >
);
```

Por ejemplo:

```
CREATE TABLE empleados (
                    enombre char (15) NOT NULL ,
                    ecod integer NOT NULL ,
                    efnac date ,
                    dcod integer
                    );
```

Crea la tabla empleados con 4 columnas. La tabla no tendrá ninguna fila, hasta que no se ejecute una sentencia de inserción de datos(*insert*)

Tipos de datos

Hemos visto que en la sentencia CREATE TABLE se especifican los tipos de datos para las columnas. Esto establece una restricción porque limita por dominio los datos que pueden almacenarse en cada columna. Los tipos de datos soportados dependen mucho del motor de base de datos que se esté utilizando, no todos los motores soportan los mismos tipos de datos. La siguiente tabla 1 lista algunos tipos del estándar de SQL.
Hay un tipo valor especial que SQL utiliza cuando una columna en alguna fila no tiene un valor asignado ya sea porque todavía no se sabe qué valor debería ir o porque específicamente se necesita que no tenga valor asignado. Ese valor es el valor **NULL**. Por ejemplo, si una columna debiera guardar la clave de otra tabla para relacionarlas y resulta que no hay ninguna fila de la otra tabla relacionada entonces el valor será **NULL**.

lógico	BOOLEAN			
character	CHAR	VARCHAR		
numérico	NUMERIC	DECIMAL	INTEGER	SMALLINT
numérico aproximado	FLOAT	REAL	DOUBLE PRECISION	
fecha y tiempo	DATE	TIME	TIMESTAMP	
intervalo	INTERVAL			

Table 1: Tipos de datos

Modificación de tablas

La sentencia *ALTER TABLE* es la que nos permite:

- Agregar columnas.
- Cambiar la definición de columnas.
- Agregar o borrar *constraints*.

La sintaxis para agregar una columna es la siguiente

```
ALTER TABLE table ADD ( column datatype [ DEFAULT expr ]);
```

La modificación de columnas tiene algunas variantes según el motor:

- Modificar Columna SQL Server

```
ALTER TABLE table ALTER COLUMN column datatype
```

- Modificar Columna PostgreSQL

```
ALTER TABLE table ALTER COLUMN column TYPE datatype
ALTER TABLE table ALTER COLUMN column SET NOT NULL ;
```

- Modificar Columna Oracle

```
ALTER TABLE table MODIFY column datatype ;
```

Ejemplo agregar clave primaria

```
ALTER TABLE tabla ADD CONSTRAINT pktbl PRIMARY KEY ( idTabla )
```

Ejemplo agregar una clave foránea

```
ALTER TABLE Hincha ADD CONSTRAINT FK_Hincha_Club FOREIGN KEY (ClubId ) REFERENCES Club ( ClubId )
```

Ejemplo agregar una constraint

```
ALTER TABLE club ADD CHECK ( fechafundacion > '25/01/1900 ');
ALTER TABLE club ADD CONSTRAINT fechafundacioncorrecta CHECK ( fechafundacion > '25/01/1900 ')
;
```

Insert

La instrucción INSERT permite agregar filas en una tabla; es la única sentencia que provee SQL para agregar filas. Existen 2 Formas de ejecutar el INSERT:

1. Usando la cláusula **VALUES** cuya sintaxis es la siguiente:

```
INSERT INTO tabla_nombre [( column [, column ...]) ]
VALUES ( value [, value ...]) ;
```

Se crea una nueva fila en la tabla. La nueva fila contendrá los valores determinados por las expresiones en la lista VALUES.

Ejemplos de uso:

```
INSERT INTO empleados
VALUES (1, 'Juan Perez ', '04/04/98 ', 100)
```

```
INSERT INTO deptos (dcod , ddescr )
VALUES (50 , ' CONTABILIDAD ')
```

```
INSERT INTO deptos DEFAULT VALUES ;
```

En el primer caso se insertan en la tabla de *empleados* los valores especificados según el orden en el que se definieron las columnas en la tabla al momento de su creación. Por lo cual el primer elemento de VALUES es el código de departamento, luego el nombre, la fecha de ingreso, etc. En el segundo caso se especifica a qué campos corresponde cada valor, así el 50 es el código de departamento mientras que *CONTABILIDAD* es el nombre de este. La última sentencia crea una fila con los valores por defecto en todas las columnas

2. Usando la cláusula **SELECT** (agrega un conjunto de filas mediante un solo INSERT). La sintaxis es:

```
INSERT INTO <tabla_nombre >
[(< columna >, : : :, <columna >)]
<SELECT statement >;
```

Supongamos que a partir de la tabla de empleados queremos guardar los datos de los gerentes en otra tabla llamada *gerentes*. En dicho caso hacemos lo siguiente

```
INSERT INTO gerentes (gcod , gnombre , gsalario )
SELECT ecod , enombre , esalario
FROM empleados WHERE ecargo = 'GERENTE ';
```

La cantidad de columnas y tipos que devuelve el SELECT deben coincidir con la cantidad y tipo de las columnas de la Tabla.

Update

La sentencia UPDATE modifica filas existentes en una tabla. La sintaxis es la siguiente:

```
UPDATE <tabla_nombre > [[ AS] <alias >]
SET <columna >=< expression >, : : :, <columna >=< expression >
[ WHERE <condicion >]
```

Supongamos que queremos que el empleado con código 7782 se incorpore al departamento con código 20, entonces deberíamos hacer lo siguiente:

```
UPDATE empleados
SET dcod = 20
WHERE ecod = 7782;
```

Si bien el UPDATE se aplica a solo una tabla la condición puede afectar a varias tablas.

Delete

La manera de eliminar una o varias filas de una tabla es utilizando la sentencia DELETE:

```
DELETE FROM <tabla_nombre > [[ AS] <alias >]
[ WHERE <condicion >];
```

Para eliminar el departamento de *finanzas* de la tabla *departamentos*:

```
DELETE FROM departamentos
WHERE ddescr = 'FINANZAS ';
```

La sentencia DELETE sin parte WHERE borra todas las filas, pero la tabla permanece creada (sin filas)

Eliminación de tablas

Para eliminar una tabla de la base de datos se utiliza la sentencia *DROP TABLE*

```
DROP TABLE <table_name >
```

Por ejemplo:

```
DROP TABLE empleados ;
```

En este ejemplo se borra la tabla empleados, no solo los datos sino la estructura completa.