CREACIÓN DE TABLAS AVANZADAS (CREATE TABLE)

La estructura más importante de una base de datos relacional es la tabla. En una base de datos multiusuario, las tablas principales son típicamente creadas una vez por el administrador de la base de datos y utilizadas luego día tras día.

La sentencia define una nueva tabla en la base de datos y la prepara para aceptar datos. Las diferentes cláusulas de la sentencia especifican los elementos de la definición de la tabla. En la práctica, la creación de una nueva tabla es relativamente sencilla. Cuando se ejecuta una sentencia CREATE TABLE, uno se convierte en el propietario de la tabla recién creada, a la cual se le da el nombre especificado en la sentencia. El nombre de la tabla debe ser un nombre SQL legal y no debe entrar en conflicto con el nombre de alguna otra tabla ya existente. La tabla recién creada está vacía, pero el DBMS la prepara para aceptar datos añadidos con la sentencia INSERT.

1. DEFINICIONES DE COLUMNAS

Las columnas de la tabla recién creada se definen en el cuerpo de la sentencia CREATE TABLE. Las definiciones de columnas aparecen en una lista, separada por comas, incluida entre paréntesis. El orden de las definiciones de las columnas determina el orden de izquierda a derecha de las columnas en la tabla. Cada definición de una columna especifica el *nombre de la columna*, que se utilizará para referirse a la columna en sentencias SQL posteriormente. Cada columna de la tabla debe tener un nombre único, pero los nombres pueden ser iguales a los de las columnas de otras tablas. El número máximo de columnas por tabla es de 1024. La longitud máxima de una línea (registro, es decir la suma de las longitudes de las columnas) es de 8060 byte (sin contar los datos de texto o imagen).

CREATE TABLE <NombreTabla>
(<NombreColumna> <TipoColumna> [Restricciones]
{,<NombreColumna> <TipoColumna> [Restricciones]}
[, <Restricciones>])
[ON <GrupoArchivo]

<NombreTabla>	Puede ser de la forma *BaseDatos.propietario.Tabla*
<NombreColumna>	Nombre de la columna que debe ser única en la tabla. Puede haber 250 columnas por tabla.
<TipoColumna>	Tipo de datos válido en el SGBD o definido por el usuario.
<Restricciones>	**[CONSTRAINT …] (Mas adelante se desarrolla…)** **Reglas de integridad (UNIQUE, IDENTITY, REFERENCES, NULL, DEFAULT ….).**
<GrupoArchivo.>	Grupo o esquema de archivos sobre el cual se creará la tabla.

EJEMPLO: Define la Tabla *OFICINAS* y sus columnas.

```
CREATE TABLE OFICINAS (
        OFICINA INTEGER NOT NULL,
        CIUDAD VARCHAR (15) NOT NULL,
        REGION VARCHAR (10) NOT NULL,
        DIR INTEGER,
        OBJETIVO MONEY,
        VENTAS MONEY NOT NULL)
```

EJEMPLO: Define la Tabla *PEDIDOS* y sus columnas.

```
CREATE TABLE PEDIDOS (
        NUM_PEDIDO INTEGER NOT NULL,
        FECHA_PEDIDO DATETIME NOT NULL,
        CLIE INTEGER NOT NULL,
        REP INTEGER,
        FAB CHAR (3) NOT NULL,
        PRODUCTO CHAR (5) NOT NULL,
        CANT INTEGER NOT NULL,
        IMPORTE MONEY NOT NULL)
```

El estándar especifica que una columna puede contener valores NULL a menos que específicamente se declare NOT NULL.

2. RESTRICCIONES

DEFINICIÓN DE VALORES POR OMISIÓN (DEFAULT <valor_constante>)

Ejemplo que define la Tabla *OFICINAS* con valores por omisión:

```
CREATE TABLE OFICINAS
(OFICINA INTEGER NOT NULL,
CIUDAD VARCHAR (15) NOT NULL,
REGION VARCHAR (10) NOT NULL DEFAULT 'Este',
DIR INTEGER DEFAULT 106,
OBJETIVO MONEY DEFAULT NULL,
VENTAS MONEY NOT NULL DEFAULT 0.00)
```

Con esta definición de tabla, sólo es necesario especificar el número de oficina y la ciudad, cuando se inserte una nueva oficina. La región por omisión es el "Este", el director de la oficina es DIR 106, el objetivo es NULL y las ventas son cero. El campo objetivo tomaría el valor por omisión NULL incluso sin la especificación DEFAULT NULL.

DEFINICIONES DE CLAVE PRIMARIA Y AJENA (PRIMARY KEY – FOREIGN KEY).

Además de la definición de las columnas de una tabla, la sentencia CREATE TABLE permite identificar la clave primaria de la tabla y las relaciones de la tabla con otras tablas de la base de datos.

La cláusula **PRIMARY KEY** especifica la columna o columnas que forman la clave primaria de la tabla. Esta columna (o combinación de columnas) sirve como identificador único para cada fila de la tabla. El DBMS requiere automáticamente que el valor de clave primaria sea único en cada fila de la tabla. Además, la definición de columna para todas las columnas que forman la clave primaria debe especificar que la columna es NOT NULL.

La cláusula **FOREIGN KEY** especifica una clave ajena en la tabla y la relación que crea con otra tabla (padre) de la base de datos. La cláusula especifica:

- La columna o columnas que forman la clave ajena, las cuáles son columnas de la tabla que está siendo creada.
- La tabla que es referenciada por la clave ajena, es la tabla padre en la relación; la tabla que está siendo definida es la hija.
- Un nombre opcional para la relación. El nombre no se utiliza en ninguna sentencia SQL, pero puede aparecer en mensajes de error y es necesaria si se desea poder suprimir la clave ajena posteriormente.
- Cómo debe tratar el DBMS un valor NULL en una o más columnas de la clave ajena, cuando la compare con las filas de la tabla padre.
- Una restricción de comprobación opcional que restrinja los datos de la tabla para que sus filas encuentren una condición de búsqueda especificada.

En general, la definición de las restricciones sobre las columnas de una tabla se hace con las instrucciones **CREATE TABLE** y **ALTER TABLE**, o por la interfaz Enterprise Manager. Se guarda en las tablas de sistema **syscomments**, **sysreferences** y **sysconstraints**. Se pueden obtener datos sobre las restricciones por los procedimientos almacenados, **sp_help** <NombreTabla> y **sp_helpconstraint** <NombreTabla>.

SINTAXIS PARA ESPECIFICAR LAS RESTRICCIONES EN CREATE TABLE

CREATE TABLE nombre_tabla
(<NombreColumna> <TipoColumna> **[RestriccionesColumna]**
{,<NombreColumna> <TipoColumna> **[RestriccionesColumna]**}
{, <RestriccionesTabla>})
[ON <GrupoArchivo>]

[RestriccionesColumna]

```
[CONSTRAINT nombreRestricción]
  {
      [ { PRIMARY KEY | UNIQUE }
          [ CLUSTERED | NONCLUSTERED]
          { ( columna[,...n] ) }
          [ WITH FILLFACTOR = factorRelleno]
          [ON { grupoArchivos | partition_scheme | "default" }]

      ]
      | FOREIGN KEY
              [(columna[,...n])]
              REFERENCES tablaReferencia
  [(columnaReferencia[,...n])]
              [NOT FOR REPLICATION]
      | CHECK [NOT FOR REPLICATION]
          (expresiónLógica)
  }
```

CONSTRAINT

Es una palabra reservada de SQL que indica el principio de la definición de una restricción PRIMARY KEY, UNIQUE, FOREIGN KEY o CHECK. Las restricciones son propiedades especiales que exigen la integridad de los datos y crean tipos especiales de índices para la tabla y sus columnas.

nombreRestricción: es el nombre de una restricción. Los nombres de restricción deben ser únicos en una base de datos.

PRIMARY KEY

Es una restricción que exige la **integridad de entidad** para una o varias columnas dadas a través de un índice único. Sólo se puede crear una restricción PRIMARY KEY por cada tabla.

UNIQUE

Es una restricción que proporciona la **integridad de entidad** para una o varias columnas dadas a través de un índice único. Una tabla puede tener varias restricciones UNIQUE.

CLUSTERED | NONCLUSTERED

Son palabras clave que indican que se ha creado un índice agrupado o no agrupado para la restricción PRIMARY KEY o UNIQUE. De forma predeterminada, el valor de las restricciones PRIMARY KEY es CLUSTERED, y el de las restricciones UNIQUE es NONCLUSTERED. Sólo se puede especificar CLUSTERED para una única restricción de una instrucción CREATE TABLE. Si especifica CLUSTERED para una restricción UNIQUE y especifica también una restricción PRIMARY KEY, el valor predeterminado de PRIMARY KEY es NONCLUSTERED.

[WITH FILLFACTOR = factorRelleno]

Especifica cuánto se debe llenar cada página de índice de SQL Server utilizada para almacenar los datos de índice. Los valores de *factorRelleno* especificados por el usuario pueden estar entre 1 y 100; el valor predeterminado es 0. Un factor de relleno pequeño crea el índice con más espacio disponible para las nuevas entradas de índice sin tener que asignar nuevo espacio.

[ON { grupoArchivos | partition_scheme | "default" }]

Especifica el esquema de partición o el grupo de archivos en que se almacena la tabla. Si se especifica *partition_scheme*, la tabla será una tabla con particiones cuyas particiones se almacenan en un conjunto de uno o más grupos de archivos especificados en *partition_scheme*. Si se especifica *grupoArchivos*, la tabla se almacena en el grupo de archivos con nombre. El grupo de archivos debe existir en la base de datos. Si se especifica "*default*" o si ON no se especifica en ninguna parte, la tabla se almacena en el grupo de archivos predeterminado. El mecanismo de almacenamiento de una tabla según se especifica en CREATE TABLE no se puede modificar posteriormente.

Esto se puede especificar también en una restricción PRIMARY KEY o UNIQUE. Estas restricciones crean índices. Si se especifica *grupoArchivos*, el índice se almacena en el grupo de archivos con nombre. Si se especifica "*default*" o si ON no se especifica en ninguna parte, el índice se almacena en el mismo grupo de archivos que la tabla. Si la restricción PRIMARY KEY o UNIQUE crea un índice clúster, las páginas de datos de la tabla se almacenan en el mismo grupo de archivos que el índice. Si se especifica CLUSTERED o la restricción crea un índice clúster, y se especifica un elemento *partition_scheme* distinto del *partition_scheme* o *grupoArchivos* de la definición de tabla, o viceversa, únicamente se respeta la definición de restricción y se omite el resto.

FOREIGN KEY...REFERENCES

Es una restricción que proporciona **integridad referencial** para los datos de la columna o columnas. Las restricciones FOREIGN KEY requieren que cada valor de la columna exista en la columna de referencia correspondiente de la tabla a la que se hace referencia. Las restricciones FOREIGN KEY pueden hacer referencia sólo a columnas que sean restricciones PRIMARY KEY o UNIQUE en la tabla de referencia.

tablaRef Es el nombre de la tabla a la que hace referencia la restricción FOREIGN KEY.

(columnaRef[,...n]) Es una columna o lista de columnas de la tabla a la que hace referencia la restricción FOREIGN KEY.

CHECK

Es una restricción que exige la **integridad del dominio** al limitar los valores posibles que se pueden escribir en una o varias columnas.

EJEMPLO: crear una tabla llamada PRODUCTOS cuyo precio por producto debe ser superior a cero.

```
CREATE TABLE PRODUCTOS
(ID_FAB VARCHAR(3),
ID_PRODUCTO VARCHAR(5),
DESCRIPCION VARCHAR(20),
PRECIO MONEY,
EXISTENCIAS SMALLINT,
CONSTRAINT CLAVE_PRODUCTOS
        PRIMARY KEY (ID_FAB,ID_PRODUCTO),
CONSTRAINT PK_PRECIO
        CHECK (precio>0))
```

NOT FOR REPLICATION

Palabras clave que se utilizan para impedir que se exija la restricción CHECK durante el proceso de distribución utilizado por la duplicación.
La cláusula NOT FOR REPLICATION significa que la restricción se fuerza en las modificaciones de los usuarios, pero no en el proceso de duplicación. La restricción NOT FOR REPLICATION CHECK se aplica tanto a la imagen anterior como posterior de un registro actualizado para impedir que se agreguen o eliminen registros del intervalo duplicado. Se comprueban todos los borrados e inserciones; si éstos se encuentran en el intervalo duplicado, se rechazan. Cuando esta restricción se utiliza con una columna de identidad, SQL Server permite que la tabla no tenga que reinicializar los valores de columna de identidad cuando un usuario de duplicación la actualiza.

expresiónLógica: es una expresión lógica que devuelve TRUE o FALSE.

Columna: Es una columna o lista de columnas, entre paréntesis, que se utiliza en las restricciones de tabla para indicar las columnas que se están utilizando en la definición de la restricción.

n: es un marcador de posición que indica que el elemento anterior se puede repetir n veces.

EJEMPLO: sentencia avanzada CREATE TABLE para la Tabla PEDIDOS, que incluye la definición de su clave primaria y de las tres claves ajenas que contiene:

```
CREATE TABLE PEDIDOS1
(NUM_PEDIDO INTEGER NOT NULL CONSTRAINT clave PRIMARY KEY,
FECHA_PEDIDO DATETIME  NOT NULL,
CLIE VARCHAR(4) NOT NULL CONSTRAINT PEDIDOPOR FOREIGN KEY
REFERENCES CLIENTES,
REP VARCHAR(3),
FAB VARCHAR (3) NOT NULL,
PRODUCTO VARCHAR (5) NOT NULL,
CANT INTEGER NOT NULL,
IMPORTE MONEY NOT NULL,
CONSTRAINT TOMADOPOR FOREIGN KEY (REP)
REFERENCES REPVENTAS (NUM_EMPL),
CONSTRAINT ESPOR FOREIGN KEY (FAB, PRODUCTO)
REFERENCES PRODUCTOS (ID_FAB,ID_PRODUCTO))
```

La figura muestra las tres relaciones creadas por esta sentencia y los nombres que se les asigna. En general es buena idea asignar un nombre de restricción, ya que ayuda a clarificar la relación creada por la clave ajena. Por ejemplo, cada pedido fue remitido por el cliente cuyo número aparece en la columna CLIE de la Tabla PEDIDOS. La relación creada por esta columna ha recibido el nombre de PEDIDOPOR.

Cuando el DBMS procesa la sentencia CREATE TABLE, compara cada definición de clave ajena con la definición de la tabla referenciada. El DBMS se asegura que la clave ajena y la clave primaria de la tabla referenciada concuerdan en el número de columnas que contienen y en sus tipos de datos. La tabla referenciada debe estar ya definida en la base de datos para que esta comparación tenga éxito.

Tabla CLIENTES

NUM_CLIE	EMPRESA
21003	Acme Mfg.

Tabla REPVENTAS

NUM_CLIE	NOMBRE
105	Bill Adams

Tabla PRODUCTOS

ID_FAB	ID_PRODUCTO	DESCRIPCION
ACI	41004	Art. Tipo 4

PEDIDOPOR **TOMADOPOR** **ESPOR**

tabla PEDIDOS

NUM_PEDIDO	FECHA_PEDIDO	CLIE	REP	FAB	PRODUCTO	CANT	IMPORTE
112963	12/17/1989	2103	105	ACI	41004	28	$3,276.00

EJEMPLO: supongamos que queremos crear una tabla llamada CATEGORÍA cuya clave principal es cod_cat.

```
Create table CATEGORÍA
(cod_cat varchar(2) NOT NULL,
Etiqueta varchar(30) NULL,
CONSTRAINT pk_categ PRIMARY KEY CLUSTERED (cod_cat))
```

EJEMPLO: se pretende que las columnas Designación y Precio deben tener valores únicos en la tabla ARTICULOS.

```
Create table ARTICULOS
(num_art varchar(2) PRIMARY KEY,
designacion_art  varchar(3),
precio integer,
constraint pk_desig UNIQUE NONCLUSTERED (designacion_art),
constraint pk_precio UNIQUE NONCLUSTERED (precio))
```

PROPIEDAD IDENTITY: permite crear una columna de identidad en una tabla. Es una propiedad opcional, que se puede usar con las instrucciones CREATE TABLE y ALTER TABLE de Transact-SQL, detrás de la definición opcional de una restricción ([CONSTRAINT]) de columna. Formato: **`IDENTITY [(seed , increment)]`**
Deber ser asignada a una columna numérica entera (tinyint, smallint, int, decimal(p,0) o numeric(p,0)), en la creación o en la modificación de la tabla, **y permite que el sistema genere valores automáticamente para esta columna (similar al tipo de dato denominado Autonumeración de Access)**. Los valores serán generados, de manera correlativa, al insertar cada nueva fila, partiendo del valor inicial especificado (por defecto 1) y aumentando/disminuyendo el valor de cada nueva fila en un incremento/decremento dado (por defecto 1). **¡Sólo puede haber una columna IDENTITY por tabla! EJEMPLO:**

```
CREATE TABLE PEDIDOS
    (num_pedido int identity (1000,1),
    numero_cli int,
    fecha_pdo datetime,
    estado_pdo varchar(2))
```

En las creaciones de línea (INSERT), no se precisa un valor para *num_pedido*. La primera inserción asignará el num_pedido 1000, a la segunda el 1001, etc.

SET IDENTITY_INSERT: Permite insertar valores explícitos en la columna identidad de una tabla. Formato:

```
SET IDENTITY_INSERT [[database_name.]schema_name.]table_name {ON|OFF}
```

EJEMPLO: SET IDENTITY_INSERT PEDIDOS ON