

Crea una base de datos

CREATE DATABASE *laEmpresa*;

USE *laEmpresa*;(nos posiciona en la base de datos creada)

GO

Para crear una tabla

CREATE TABLE Empleado (
documento INT PRIMARY KEY,
apellido varchar(20) not null,
);

ALTER TABLE :

Modifica una definición de tabla,
agrega o quita columnas y restricciones.

Agrega nueva columna:

ALTER TABLE Empleado **ADD** nombre VARCHAR(20) NULL ;

Quita una columna:

ALTER TABLE Empleado **DROP COLUMN** column_a;

Cambia el tipo de datos :

ALTER TABLE Empleado

ALTER COLUMN column_a **DECIMAL** (5, 2) ;

Agrega una columna con una restricción :

ALTER TABLE doc_exc **ADD** column_b VARCHAR(20) NULL **CONSTRAINT**
exb_unique **UNIQUE** ;

Agrega una restricción no comprobada:

ALTER TABLE doc_exd **WITH NOCHECK ADD CONSTRAINT** exd_check
CHECK (column_a > 1)

Agrega una restricción default:

ALTER TABLE doc_exz

ADD CONSTRAINT col_b_def **DEFAULT** 50 **FOR** column_b ;

ALTER TABLE: MODIFICA UNA TABLA

Agrega una columna

ALTER TABLE Empleado **ADD** nombre VARCHAR(20) NULL;

GO

ALTER TABLE Empleado **ADD** codEmpleado INT **IDENTITY**(1,1);

GO

Modifica una columna

ALTER TABLE Empleado **ALTER COLUMN** column apellido VARCHAR(30)
NOT NULL;

GO

Borra una columna

ALTER TABLE Empleado **DROP COLUMN** codEmpleado;

EDAD:

ALTER TABLE persona

ADD CONSTRAINT CK_persona_fecha CHECK

(DATEDIFF(YY, fecha_nacimiento, getdate()) < 100); verifica la edad, que sea menor a 100

- **DATEDIFF(YY, '1981-03-20', GETDATE()):** La función **DATEDIFF** calcula la diferencia entre dos fechas. En este caso, el primer argumento **YY** indica que la diferencia se calculará en años. El segundo argumento es la fecha de nacimiento '1981-03-20', y el tercer argumento es la fecha actual **GETDATE()**.

- **AS 'EDAD':** Asigna un alias al resultado de la función **DATEDIFF**, que en este caso se llama 'EDAD'. Esto significa que el resultado de la consulta será una columna llamada **EDAD**.

SELECT DATEDIFF(YY, '1981-03-20', getdate()) as 'EDAD'; (solo es una consulta)

SELECT DATEDIFF(YY, fecha_nacimiento, GETDATE()) AS 'EDAD' FROM persona WHERE id = 1; -- Puedes ajustar esta condición según sea necesario, para saber la edad de una persona específica.

Fecha nacimiento:

ALTER TABLE persona

ADD CONSTRAINT DF_persona_fecha_nac DEFAULT getdate() FOR

fecha_nacimiento;

DEFAULT getdate() FOR fecha_nacimiento: Esta parte define una restricción de valor por defecto para la columna **fecha_nacimiento**. **getdate()** es una función que devuelve la fecha y hora actuales del sistema. Por lo tanto, si al insertar una fila en la tabla **persona** no proporcionas un valor para **fecha_nacimiento**, el valor por defecto será la fecha y hora actuales en el momento de la inserción

*****La función getdate() muestra la fecha actual**

SELECT getdate();

DROP TABLE

Elimina la definición de una tabla y todos los datos, índices, restricciones y especificaciones de permisos asociados. Realiza un borrado físico de la tabla.

DROP TABLE [database_name . [schema_name] . | schema_name .]

table_name [,...n] [;]

DROP TABLE empleados; (borra una tabla)

SELECT* from persona; (permite ver los campos de una tabla)

INSERT

Agrega una o varias filas nuevas a una tabla o una vista.

INSERT [INTO] objeto [(lista de columnas)] VALUES (({ DEFAULT | NULL | expresión } [,...n]) [,...n])

INSERT INTO rr.hhEmpleado **VALUES** (20111333, 'GOMEZ');

INSERT INTO Empleado (documento, apellido) **VALUES** (20111333, 'GOMEZ');

UPDATE

La instrucción **UPDATE** en SQL se utiliza para **modificar los datos existentes en una tabla**.

Puedes usar **UPDATE** para cambiar los valores de una o más columnas en una o más filas.

UPDATE objeto SET nombre columna = { expresión | DEFAULT | NULL } [,...n] [FROM { } [,...n]] [WHERE { }]

UPDATE Empleado **SET** apellido ='PEREZ' **WHERE** apellido ='Achong';(cambio el apellido Achong por el de Perez)

La instrucción **DELETE** quita una o varias filas de una tabla o vista.

Ej: 1) **DELETE** [FROM] tablix **WHERE** col1 = 100

2) **DELETE FROM** Sales.SalesPersonQuotaHistory

WHERE SalesPersonID

IN (SELECT SalesPersonID **FROM** Sales.SalesPerson **WHERE** SalesYTD > 2500000.00)

La instrucción **TRUNCATE TABLE** es un método rápido y no registrado para eliminar todas las filas de una tabla. **TRUNCATE TABLE** es funcionalmente equivalente a la instrucción **DELETE** sin una cláusula **WHERE**. Sin embargo, **TRUNCATE TABLE** es más rápida y utiliza menos recursos de registro de sistema y de transacciones.

La instrucción **DELETE** quita una a una las filas y graba una entrada en el registro de transacciones por cada fila eliminada. Ej: **TRUNCATE** table

NULL = NULL ? Funcion

ISNULL()

Ej:

Create table persona (id int, apellido varchar(20))

Insert into persona (id, apellido) **Values** (1,null)

Create table historico_persona (id int, apellido varchar(20))

Insert into historico_persona (id, apellido) **Values** (2,null)

select * from persona a inner join historico_persona h on a.apellido = h.apellido

FOREING KEY (definida en la tabla)

CONSTRAINT FK_emp_dept **FOREIGN KEY** (department_id) REFERENCES departments(department_id),

CONSTRAINT UQ_emp_email **UNIQUE**(email)); Garantiza que los valores en la columna dni o email sean únicos, evitando duplicados.

-- Tabla modelo

CREATE TABLE modelo

(

id_modelo INT NOT NULL,

descripcion VARCHAR(100) NOT NULL,

id_marca INT NOT NULL,

CONSTRAINT PK_modelo **PRIMARY KEY** (id_modelo, id_marca),

CONSTRAINT FK_modelo_marca **FOREIGN KEY** (id_marca) REFERENCES marca(id_marca)

);

ALTER TABLE viaje

ADD CONSTRAINT CK_viaje_sucursal_origen_destino

CHECK (id_sucursal_origen <> id_sucursal_destino);

/*

Sintaxis

ALTER TABLE: Modifica la estructura de la tabla viaje.

ADD CONSTRAINT: Añade una nueva restricción con el nombre CK_viaje_sucursal_origen_destino.

CHECK (id_sucursal_origen <> id_sucursal_destino):
Verifica que el valor de id_sucursal_origen sea diferente al de id_sucursal_destino,
evitando así que un viaje tenga la misma sucursal como origen y destino.
*/

Fecha

```
ALTER TABLE camion_chofer
ADD CONSTRAINT DF_camion_chofer_fecha_asignacion
DEFAULT GETDATE() FOR fecha_asignacion;
*** DEFAULT GETDATE() FOR fecha_asignacion: Establece GETDATE() como valor predeterminado para la columna
fecha_asignacion,
*** insertando la fecha y hora actual del sistema cuando no se proporciona un valor.
```

- **DEFAULT** se usa para establecer un valor automático en una columna cuando no se proporciona un valor específico durante la inserción.
- Esto ayuda a asegurar que las columnas tengan un valor incluso si se olvida proporcionar uno explícitamente.

```
ALTER TABLE empleados
ADD CONSTRAINT DF_empleados_nombre
DEFAULT 'Desconocido' FOR nombre;
```

```
ALTER TABLE pedidos
ADD CONSTRAINT DF_pedidos_fecha
DEFAULT GETDATE() FOR fecha_pedido;
```

```
ALTER TABLE productos
ADD CONSTRAINT DF_productos_precio DEFAULT 0.00 FOR precio;
```

```
ALTER TABLE usuarios
ADD CONSTRAINT DF_usuarios_activo DEFAULT 1 FOR activo;
```

HORA:

```
-- Usando FORMAT
SELECT FORMAT(GETDATE(), 'HH:mm:ss') AS HoraSinMilisegundos;
```

```
-- Usando CONVERT
SELECT CONVERT(VARCHAR(8), GETDATE(), 108) AS HoraSinMilisegundos;
```

```
CREATE TABLE eventos (
    id INT PRIMARY KEY,
    descripcion NVARCHAR(255),
    hora_evento TIME
);
```

```
INSERT INTO eventos (id, descripcion, hora_evento)
VALUES (1, 'Evento de prueba', CONVERT(TIME, GETDATE(), 0));
```

Para insertar la hora actual en la columna hora_evento, puedes utilizar la función CONVERT o CAST para formatear la hora sin milisegundos al insertar los datos:

```
INSERT INTO eventos (id, descripcion, hora_evento)
VALUES (1, 'Evento de prueba', CONVERT(TIME, GETDATE(), 0));
```

CHECK:

Una **restricción CHECK** en SQL es una regla que defines para asegurar que los datos en una columna (o en varias columnas) cumplan con ciertas condiciones o criterios específicos.

Dentro de una tabla

```
CREATE TABLE Empleados (
```

```

id INT PRIMARY KEY,
nombre NVARCHAR(100),
salario DECIMAL(10, 2) CHECK (salario > 0) -- Restricción CHECK
);
ALTER TABLE Contratos
ADD CONSTRAINT CHK_Fecha_Salida_Valid CHECK (fecha_salida IS NULL OR fecha_salida >
fecha_ingreso);

***para asegurar que la hora de entrada es distinta y mayor al de salida
ALTER TABLE RegistroHoras ADD CONSTRAINT CHK_Horas_Validas CHECK (hora_entrada
<= hora_salida);

ALTER TABLE Contratos
ADD CONSTRAINT CHK_Fecha_Salida_Valid CHECK (fecha_salida IS NULL OR fecha_salida >
fecha_ingreso);

***dentro de una tabla
CREATE TABLE RegistroHoras (
id INT PRIMARY KEY,
hora_entrada TIME,
hora_salida TIME,
-- Definir la restricción CHECK
CONSTRAINT CHK_Horas_Validas CHECK (hora_entrada <= hora_salida)
);

```