GESTION DE BASES DE DATOS

SQL – DDL

CREATE: Crear nuevas bases de datos o tablas

- CREATE DATABASE NombreBaseDatos
- CREATE TABLE NombreTabla

CREATE TABLE Tabla1 (Campo1 tipodatos1 restriccion1, Campo2 tipodatos2 restriccion2, …, CampoN tipodatosN restriccionN)

Restricciones de los campos:

[NOT NULL]
[DEFAULT valor_predeterminado]
[AUTO_INCREMENT][PRIMARY KEY] [Referencia]
[PRIMARY KEY] (columna1, columna2…)
[UNIQUE]
[CONSTRAINT nombre] FOREIGN KEY [nombre_índice][referencia]

Creación de tabla incluyendo relación:

CAMPO	TIPO DATOS
ID_COMPRA	INT / AUTO_INCREMENT / PRIMARY KEY
ARTICULO	VARCHAR (30)
CANTIDAD	INT (5)
CLIENTE	INT / RELACIONADO CON LA TABLA CLIENTE ELIMINADOSE EN CASCADA

✓*CREATE TABLE COMPRAS (ID_COMPRAS INT AUTO_INCREMENT PRIMARY KEY, ARTICULO VARCHAR(20), CANTIDAD INT(5), CLIENTE INT, CONSTRAINT REL1 FOREIGN KEY (CLIENTE) REFERENCES CLIENTES (ID_CLIENTE) ON DELETE CASCADE ON UPDATE CASCADE)*

ALTER: Modificar tablas agregando/modificando definición de campos

Añadir una columna:

ALTER TABLE Tabla1 ADD COLUMN NombreColumna tipodato;

Eliminar una columna:

ALTER TABLE Tabla1 DROP COLUMN NombreColumna;

Modificar una columna:

ALTER TABLE Tabla1 MODIFY COLUMN NombreColumna tipodato;

Añadir a la tabla Compras la columna Precio, DECIMAL:

ALTER TABLE Compras **ADD COLUMN** Precio DECIMAL

Eliminar de la tabla de Compras la columna Precio:

ALTER TABLE Compras **DROP COLUMN** Precio

Modificar de la tabla Comprar la columna de Cantidad que es INT por DECIMAL:

ALTER TABLE Compras **MODIFY COLUMN** Cantidad DECIMAL

GESTION DE BASES DE DATOS

Añadir el valor por defecto (0) de la columna Cantidad de la tabla Compras:

ALTER TABLE Compras **ALTER COLUMN** Cantidad **SET** DEFAULT 0

Eliminar la clave principal de la tabla Clientes (ID_Cliente):

ALTER TABLE Clientes **DROP PRIMARY KEY**

Añadir como clave primaria en la tabla Clientes el ID_Cliente:

ALTER TABLE Clientes **ADD PRIMARY KEY** (ID_Cliente)

Eliminar la clave ajena de la tabla Compras:

ALTER TABLE Clientes **DROP FOREING KEY REL1**

Añadir la relación entre Compras y Clientes:

ALTER TABLE Compras **ADD CONSTRAINT REL2 FOREING KEY** (Cliente) **REFERENCES** Clientes(ID_Cliente)

DROP: Eliminar bases o tablas

Eliminar una base de datos:

```
DROP DATABASE NombreBaseDatos
```

DROP TABLE Compras

Eliminar una tabla:

```
DROP TABLE NombreTabla;
```

DROP DATABASE NombreBaseDatos

SQL DML

Sintaxis para introducir un registro:

INSERT INTO Tabla (Campo1, Campo2, ...) **VALUES** (valor1,valor2,...)

INSERT INTO Tabla **VALUES** (valor1,valor2,...)

INSERT INTO Clientes **VALUES** (NULL, 'Maria', 'Garcia Castillo', 35, 'Valencia')

INSERT INTO Clientes (Código, Nombre, Apellidos, Edad, Población) **VALUES** (NULL, 'María', 'García Castillo', 35, 'Valencia')

Insertar un valor por defecto:

INSERT INTO Clientes **VALUES** (NULL, 'Sonia', 'Carretero Ruíz', 28, DEFAULT)

No sabemos algún campo o no queremos rellenarlo:

INSERT INTO Clientes **VALUES** (NULL, 'Sergio', 'Miralles Saez', NULL, ' Alaquàs')

Solo se requiere rellenar los campos necesarios:

INSERT INTO Clientes (Código, Nombre, Apellidos, Edad) **VALUES** (NULL, 'Silvia', 'Ramírez Vilches', 29)

UPDATE – Actualización de tablas

Sintaxis:

UPDATE Tabla **SET** NuevoValor **WHERE** Criterio;

Actualizar las personas que viven en Alaquàs ahora viven en Torrent:

UPDATE Clientes **SET** Población = "Torrent" **WHERE** Población = "Alaquàs"

Actualizar edad de todos los clientes que tienen entre 30 y 50 años, añadiéndole 3 años más:

UPDATE Clientes **SET** Edad = Edad + 3 **WHERE** Edad **BETWEEN** 30 AND 50

Creación de nuevas tablas a partir de una tabla con criterios especificados

CREATE TABLA NuevaTabla **SELECT** CamposNuevaTabla **FROM** TablaPrincipal **WHERE** Criterio;

Crear una tabal a partir de la tabla Clientes, con los clientes que son de Valencia. La nueva tabla se llama Clientes_Valencia:

CREATE TABLE Clientes_Valencia **SELECT** * **FROM** Clientes **WHERE** Población = "Valencia"

Igual pero solo con el Código, Nombre y Apellidos:

CREATE TABLE Clientes_Valencia **SELECT** Código, Nombre, Apellidos **FROM** Clientes **WHERE** Población = "Valencia"

DELETE – Eliminar registros dependiendo del criterio

Sintaxis:

DELETE FROM Tabla **WHERE** Criterio;

Eliminar los clientes de Valencia de la tabla Clientes:

DELETE FROM Clientes **WHERE** Población = "Valencia"

Eliminar de la tabla Clientes_Valencia los que se llamen José y tengan + 45 años.

DELETE FROM Clientes_Valencia **WHERE** Nombre = "Jose" **AND** Edad > 45

INSERT INTO – Anexar datos de otra tabla

Sintaxis:

INSERT INTO TablaAAnexar (campo1,campo2…) **SELECT** (campo1,campo2…) **FROM** TablaDeDondeSeAnexa **WHERE** Criterio;

Anexar a tabla Clientes los datos e la tabla Clientes_Valencia, todos los campos:

INSERT INTO Clientes **SELECT** * **FROM** Clientes_Valencia

Solo anexar campos Nombre y Edad:

INSERT INTO Clientes (Nombre, Edad) **SELECT** Nombre, Edad **FROM** Clientes Valencia

Anexar los campos Nombre, Apellidos y Edad, pero solo los que sean mayores de 50:

INSERT INTO Clientes (Nombre, Apellidos, Edad) **SELECT** Nombre, Apellidos, Edad **FROM** Clientes_Valencia **WHERE** Edad > 50

GESTION DE BASES DE DATOS

TRIGGERS

Sintaxis:

```
CREATE TRIGGER <NombreTrigger>
        {BEFORE|AFTER}
        {INSERT|UPDATE|DELETE}
ON <NombreTabla>
        FOR EACH ROW
        <sentenciaSQL>
```

2 palabras clave:

- OLD: Valor anterior al disparo.
- NEW: Valor posterior al disparo.

Según la operación se utilizarán 1 o 2:

- INSERT: Solo NEW.
- UPDATE: OLD y NEW.
- DELETE: Solo OLD.

Cada vez que alguien inserte un nuevo cliente en la tabla Clientes, en la tabla Auditoría_clientes se insertara una nueva entrada:

CREATE TRIGER AudClient_Insertar

AFTER INSERT

ON Clientes

FOR EACH ROW

INSERT INTO Auditoría_clientes (Nombre_nuevo, Seccion_nuevo, usuario, Fecha_modif, Proceso, Id_cliente)

VALUES (NEW.Nombre, NEW.Seccion, CURRENT_USER(), NOW, NEW.Accion, NEW.Id_cliente)

Cada vez que alguien actualice un nuevo cliente en la tabla Clientes, en la tabla Auditoría_clientes se insertara una nueva entrada:

CREATE TRIGER AudClient_Modificar

BEFORE UPDATE

ON Clientes

FOR EACH ROW

INSERT INTO Auditoría_clientes (Nombre_ant, Seccion_ant, Nombre_nuevo, Seccion_nuevo, usuario, Fecha_modif, Proceso, Id_cliente)

VALUES (OLD.Nombre, OLD.Seccion, NEW.Nombre, NEW.Seccion, CURRENT_USER(), NOW, NEW.Accion, NEW.Id_cliente)

Cada vez que alguien borre un nuevo cliente en la tabla Clientes, en la tabla Auditoría_clientes se insertara una nueva entrada.

```
CREATE TRIGER AudClient_Eliminar
    BEFORE DELETE
ON Clientes
    FOR EACH ROW
INSERT INTO Auditoría_clientes (Nombre_ant, Seccion_ant, usuario, Fecha_modif, Proceso, Id_cliente)
VALUES (OLD.Nombre, OLD.Seccion, CURRENT_USER(), NOW, OLD.Id_cliente)
```

PROCEDIMIENTOS

Sintaxis para instrucción SQL:

```
CREATE PROCEDURE NombrePA(parámetros)
<InstrucciónSQL>
```

Ejecución en la base de datos:

```
CALL NombrePA(parámetros);
```

Procedimientos sin parámetros – Consulta de Clientes de valencia:

```
CREATE PROCEDURE Clientes_Valencia()
    SELECT * FROM Clientes WHERE Población = "Valencia"
```

Procedimientos con parámetros – Consulta de Clientes de valencia:

```
CREATE PROCEDURE Act_Precio(Precio_Nue DECIMAL(5,2), Cod_Art INT(6))
    UPDATE Articulos SET Precio = Precio_Nuev WHERE Codigo_articulo = Cod_Art
```

Procedimiento con más de una instrucción SQL

Sintaxis:

```
DELIMITER //
CREATE PROCEDURE NombrePA(parámetros)
    BEGIN
        <CUERPO DEL BLOQUE>;
    END;//
DELIMITER ;
```

Declaración de variables en Procedimientos – Calcular la edad según el año de nacimiento:

```
DELIMITER //
CREATE PROCEDURE EdadPersona(Año_nac INT)
    BEGIN
        DECLARE Año_act INT DEFAULT 2021;
        DECLARE Edad INT;
        SET Edad = Año_act – Año_nac;
        SELECT Edad;
    END; //
DELIMITER ;
```

Triggers condicionales – Actualizar precio, pero controlando que no sea negativo ni > 900

```
DELIMITER //
CREATE TRIGGER Act_Precio BEFORE UPDATE ON Articulos FOR EACH ROW
    BEGIN
        IF (New.Precio<0) THEN
            SET New.Precio = 1;
        ELSEIF (New.Precio > 900) THEN
            SET New.Precio = 900;
        END IF;
    END; //
DELIMITER ;
```