Nombre: Karen Daniela Holguin Cruz
Fecha: día mes año
Profesor:
Materia:
Institución:
Curso:
Nota:

## 1. Clave Primaria (Primary Key)

- Una clave primaria es un campo (o conjuntos de campos) que identifica de manera única cada registro en una tabla. No puede haber duplicados ni valores nulos

- La clave primaria es esencial en una base de datos relacional porque permite identificar de forma única cada registro. Es como una cédula para una persona. Una base de datos no puede funcionar correctamente sin una buena definición de claves primarias, ya que ellas aseguran la integridad de los datos.

```
CREATE TABLE Estudiantes (
ID INT PRIMARY KEY,
Nombre VARCHAR (50)
);

CREATE TABLE Inscripciones(
EstudianteID INT,
CursoID INT,
PRIMARY KEY (EstudianteID, CursoID)
);

CREATE TABLE Productos(
CodigoProducto INT,
Nombre VARCHAR (100),
CONSTRAINT PK_Productos PRIMARY KEY
);

CREATE TABLE Clientes (
ID INT PRIMARY KEY IDENTITY,
Nombre VARCHAR (100)
);

CREATE TABLE Empleados (
Cedula VARCHAR (20) PRIMARY KEY,
Nombre VARCHAR (100)
);
```

## 2. Clave foránea y Delete

- Una clave foranea (foreign key) es un campo que crea una relación entre dos tablas. El comando DELETE se usa para eliminar registros de una tabla y se puede combinar con la clave foranea para manejar relaciones

- La clave foránea sirve para mantener relaciones entre tablas Es como un vínculo entre estudiantes y cursos y el comando DELETE nos permite eliminar datos, pero debemos tener cuidado con las relaciones si borramos un dato padre, puede afectar a los hijos si no gestionamos bien la clave foranea

```
CREATE TABLE Departamentos(
ID INT PRIMARY KEY,
Nombre VARCHAR (50)
);

CREATE TABLE Empleados(
ID INT PRIMARY KEY,
Nombre VARCHAR (50),
DepartamentoID INT
FOREIGN KEY REFERENCES Departamento
);

DELETE FROM Empleados WHERE ID=1;

CREATE TABLE Cursos(
ID INT PRIMARY key,
Nombre VARCHAR (50),
);

CREATE TABLE ordenes(
ID INT PRIMARY KEY,
ClienteID INT,
FOREIGN key (clienteID)
);

DELETE FROM Inscripciones WHERE CursoID=2;
);
```

3. Truncate

- Truncate elimina todos los registros de una tabla, pero de forma rápida y sin registrar cada fila individualmente como DELETE No se puede usar con Claves foraneas activas

- El comando TRUNCATE es como un borrado masivo. Sirve Cuando necesitamos vaciar completamente una tabla, pero sin eliminar una estructura. Es más eficiente que DELETE Cuando no hay relaciones foraneas y se requiere velocidad.

```
TRUNCATE TABLE LogsSistema;
TRUNCATE TABLE TempResultados;
TRUNCATE TABLE Inscripciones;
TRUNCATE TABLE Inventario;
TRUNCATE TABLE productos;
```

DIFERENCIAS clave ENTRE DELETE y TRUNCATE

| Características | DELETE | TRUNCATE |
|---|---|---|
| Registro en log | Si | Mínimo |
| Se puede usar WHERE | Si | NO |
| Más rapido | NO | Si |
| Reinicia IDENTITY | NO | Si (por defecto) |
| Afecta fk | Respecta restricciones | Error si hay fk |

4. Primera forma Normal

•La primera forma Normal o 1fN en la Normalizacion de bases de datos establece que todos los valores de cada columna deben ser atomicos, es decir, invisibles. Ademas no deben existir grupos repetidos ni listas de valores en una misma columna. Aplicar esta forma mejora la consistencia y facilita sus consultas.

| ID | Nombre | Teléfonos |
|---|---|---|
| 1 | Juan pérez | 3201345G910, 3109876543 |

Ejemplo Normalizado

| ID | Nombre | Teléfonos |
|---|---|---|
| 1 | Juan pérez | 3201345G910 |
| 1 | Juan pérez | 3109876543 |

5. funciones distintas aplicadas

```
SELECT DISTINCT Nombre from estudiantes;
SELECT * FROM Estudiantes WHERE Nombre LIKE
SELECT * FROM Estudiantes ORDER BY Nombre ASC;
SELECT Nombre, COUNT AS cantidad Telefonos
FROM telefonos
GROUP By nombres;
SELECT COUNT AS total Registros FROM Estudiantes;
```

6. Normalización en base de Datos

•La normalización en base de datos es el proceso de organizar datos para reducir la redundancia y mejorar la integridad Consiste en aplicar una serie de reglas conocidas como formas Normales: Primera (1fN) segunda (2fN) Tercera (3fN) Su objetivo principal es evitar datos duplicados, facilita el mantenimiento y asegura relaciones claras

Sin Normalizar

| pedidoID | clienteNombre | clienteGmail | producto |
|---|---|---|---|
| 1 | Ana Torres | ana@gmail.com | Teclado |
| 2 | Ana Torres | ana@gmail.com | House |

Tablas Normalizadas

| clienteID | Nombre | Gmail |
|---|---|---|
| 1 | Ana Torres | ana@gmail.com |

| PedidosID | clienteID | Productos |
|---|---|---|
| 1 | 1 | Teclado |
| 2 | 1 | House |

## 7.5 funciones

```
SELECT SUM (precio) AS totalVentas FROM pedidos;
SELECT MAX (precio) AS precioMaximo FROM productos;
SELECT MIN (precio) AS precioMinimo FROM productos;
SELECT AVG (precio) AS Promedio FROM pedidos
SELECT CONCAT (nombre, Apellido) AS NombreCompleto FROM clientes;
```

## 8. Como hacer un UPDATE

•El comando UPDATE en SQL se utiliza para modificar datos existentes en una tabla. Es muy útil cuando queremos corregir un error o actualizar una información o modificar varios registros al mismo tiempo es importante usarlo con where para evitar actualizar todos los registros por accidente

```
UPDATE nombre-tabla
SET Columna1 = valor1 Columna 2 = valor 2
WHERE condición;
```

### 1. Actualizar el nombre de un cliente

```
UPDATE Clientes
SET Nombre = 'Juan perez'
WHERE ID = 1;
```

### 2. Cambiar el correo de un estudiante específico

```
UPDATE Estudiantes
SET Email = nuevoemail@gmail.com
WHERE Cedula = '1234567890';
```

### 3. Aumentar el precio de un producto en un 10%.

```
UPDATE productos
SET precio = precio * 1.10
WHERE categoria = 'Electronica'
```

### 4. Actualizar varios Campos a la vez

```
UPDATE Empleados
SET Cargo = 'Supervisor', Salario = 3200000
WHERE ID = 4;
```

### 5. Marcar un pedido como entregado

```
UPDATE pedidos
SET Estado = 'Entregado'
WHERE pedidoID = 27;
```

### 6. Reducir inventario de un producto

```
UPDATE Inventario
SET Cantidad = Cantidad - 5
WHERE productoID = 10;
```

### 7. Borrar información confidencial remplazado con Null

```
UPDATE usuarios
SET Telefono = Null
WHERE Baja = 'si'
```