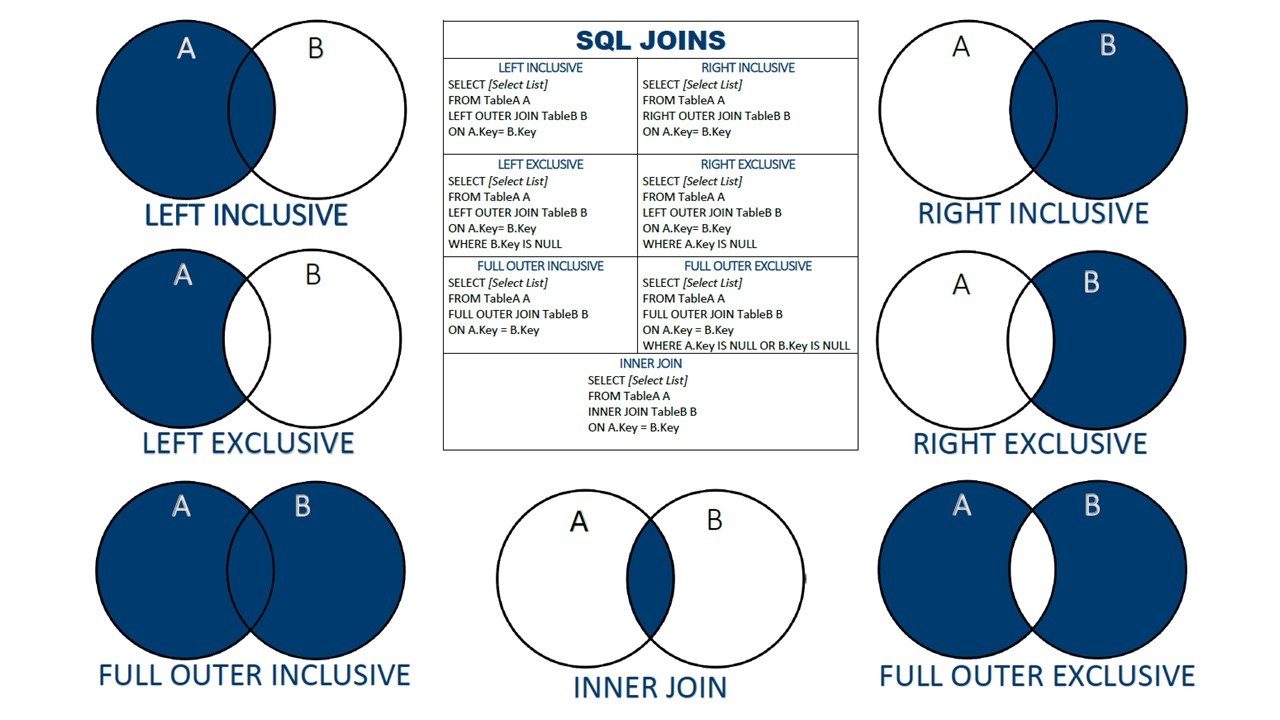
**Conocimientos SQL**

**(Arevalo Vaca Brayan Israel)**

Funcionamiento de la sentencia JOIN: La sentencia JOIN o (UNION) permite unir los registros de 2 o mas tablas en la base de datos, esta unión es posible debido a una condición que brinda la similitud o relación de los campos en ambas tablas. Dando como resultado una tabla que contiene los datos de interés con ambas tablas.



Referencia: https://www.linkedin.com/pulse/sql-inner-join-tutorial-matt-l

* **Inner Join:** Esta sentencia nos da como resultado solo las filas de las tablas donde haya match en la sentencia. Tomando de referencia la imagen anterior, Solo mostrara las filas donde el campo de la columna ‘key’ sea igual tanto en la tabla A como en la tabla B.
* **Left outer Join/Left join:** Sentencia que nos da como resultado las filas de la tabla izquierda “A” y los registros que coinciden de la tabla B (aquellas columnas que no hacen match de la tabla derecha se incluirán valores NULL).
* **Right outer Join/ Right join:** Hace la misma sentencia que la Sentencia left join, pero ahora con la tabla B o derecha. Sentencia que nos da como resultado las filas de la tabla derecha “B” y los registros que coinciden de la tabla “A” (Aquellas columnas que no hacen match con la tabla izquierda se incluirán valores NULL).
* **Left exclusive:** Esta sentencia es similar que la sentencia left join pero incluyendo un WHERE en la sentencia. Se obtiene todos los datos de la tabla “A” o izquierda, permitiendo excluir las filas donde hay match con la tabla “B” o derecha (colocando NULL en las columnas donde hay match entre las tablas).
* **Right exclusive:** Esta sentencia es similar a la sentencia left exclusive, pero para la tabla derecha o “B”. Se obtiene todos los datos de la tabla “B” o derecha y permite excluir las filas donde hay match con la tabla “A” o izquierda.
* **Full outer join:** Esta sentencia Brinda todas las filas de la tabla izquierda “A” y la tabla derecha “B” incluyendo filas donde sus columnas coinciden o no.
* **Full outer exclusive:** Esta sentencia brinda todas las filas de ambas tablas excepto las filas donde las columnas de las tablas coinciden.

**¿Cuál es el funcionamiento principal del TRIGGER y qué propósito tienen?**

Los trigger son herramientas que nos permite automatizar querys, son activados antes o después de un evento específicos como INSERT/UPDATE/DELETE de alguna tabla, con el propósito de mantener la integridad de los datos, aplicación de reglas por la empresa o por automatización de sentencias repetitivas.

**¿Qué es y para qué sirve un STORED PROCEDURE?**

Bloque de código precompilado que se guarda en la base de datos para que se pueda utilizar una y otra vez solo llamándola para ejecutarla. Estos procedimientos se les puede pasar parámetros con la finalidad de poder funcionar a base de los valores proporcionado. Si hacemos una analogía, es una función en el paragdima de la programación funcional.

SCRIPTS

**Crear tablas:**

-- INIT database

CREATE TABLE productos (

idProducto INT AUTO\_INCREMENT KEY,

nombre VARCHAR(40),

precio DECIMAL(16,2)

);

CREATE TABLE ventas (

idVenta INT(6) AUTO\_INCREMENT KEY,

idProducto INT(6),

cantidad INT(6),

FOREIGN KEY (idProducto) REFERENCES productos(idProducto) ON DELETE CASCADE ON UPDATE CASCADE

);

**Insertar productos en la tabla:**

INSERT INTO productos (nombre, precio)

VALUES ('LAPTOP', 3000),

('PC', 4000),

('MOUSE', 100),

('TECLADO', 150),

('MONITOR', 2000),

('MICROFONO', 350),

('AUDIFONOS', 450);

INSERT INTO ventas (idVenta, idProducto, cantidad)

VALUES (1, 5, 8),

(2, 1, 15),

(3, 6, 13),

(4, 6, 4),

(5, 2, 3),

(6, 5, 1),

(7, 4, 5),

(8, 2, 5),

(9, 6, 2),

(10, 1, 8);

**Productos que obtuvieron solo 1 venta:**

SELECT

p.idProducto,

p.nombre AS Producto,

COUNT(v.idVenta) AS Cantidad\_Ventas

FROM

productos AS p

INNER JOIN

ventas AS v

ON v.idProducto = p.idProducto

GROUP BY

p.idProducto, p.nombre

HAVING

COUNT(v.idVenta) = 1;



**Cantidad de productos vendidos:**

-- QUERY database

SELECT

p.idProducto,

p.nombre AS Producto,

SUM(v.cantidad) AS Cantidad\_Productos\_Vendidos

FROM

productos AS p

INNER JOIN

ventas AS v

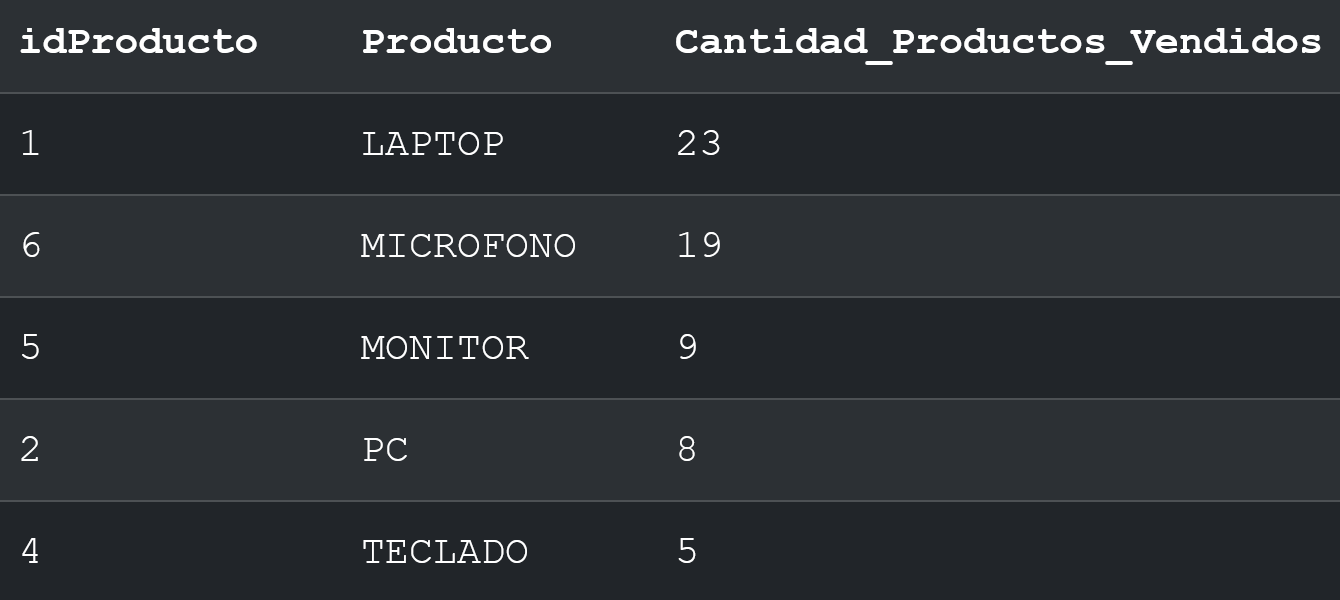
ON v.idProducto = p.idProducto

GROUP BY

p.idProducto, p.nombre

ORDER BY

Cantidad\_Productos\_Vendidos DESC;



**Total de percepciones por producto:**

-- QUERY database

SELECT

p.idProducto,

p.nombre AS Producto,

COALESCE(SUM(v.cantidad), 0) AS Cantidad\_Productos\_Vendidos,

COALESCE(SUM(v.cantidad \* p.precio), 0) AS Total\_VENDIDO

FROM

productos p

LEFT JOIN

ventas v ON p.idProducto = v.idProducto

GROUP BY

p.idProducto, p.nombre

ORDER BY

Cantidad\_Productos\_Vendidos DESC;

