**1. CONOCIMIENTOS SQL**

1.1) Describe el funcionamiento general de la sentencia JOIN.  
R: La función principal es unir dos o más tablas de una base de datos que tienen una columna que las relaciona.

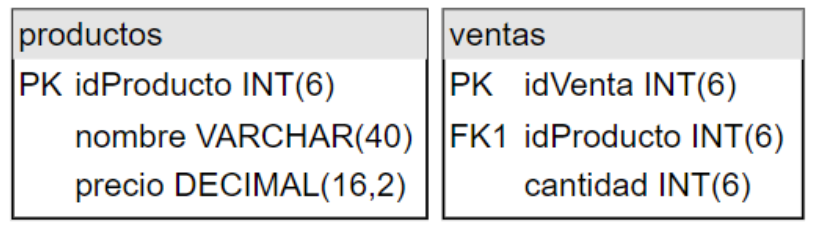
1.2) ¿Cuáles son los tipos de JOIN y cuál es el funcionamiento de los mismos?

1. **INNER JOIN:** regresa las filas que tienen coincidencia en **ambas** tablas
2. **LEFT JOIN**: regresa todas las filas de la primera tabla y solo las que tienen coincidencia de la segunda tabla
3. **RIGHT JOIN:** lo inverso, regresa todas las filas de la segunda tabla y solo las que tienen coincidencia de la primera.
4. **CROSS JOIN:** devuelve todos los registros de ambas tablas

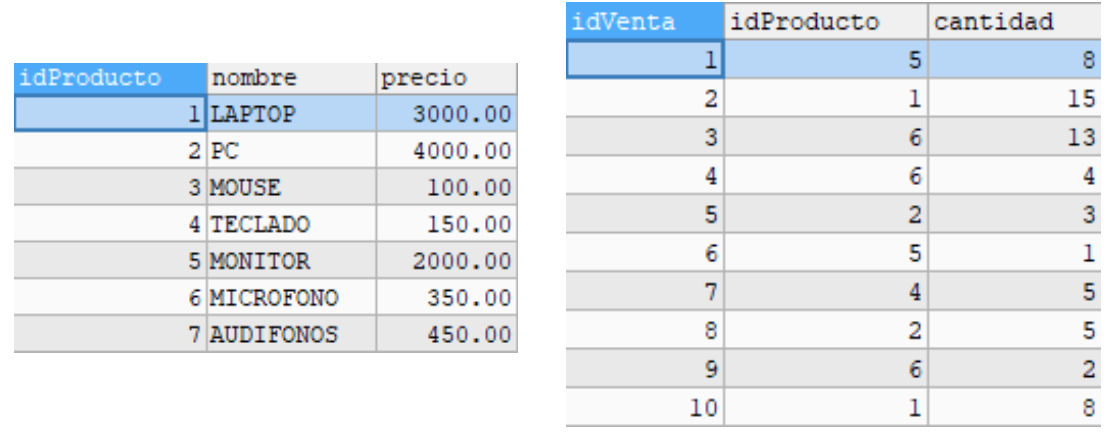
1.3) ¿Cuál el funcionamiento general de los TRIGGER y qué propósito tienen?  
R: Es una manera de automatizar acciones en la base de datos, por ejemplo añadir un registro a una tabla automáticamente al detectar un cambio en otra tabla, su propósito principal es la automatización de estas acciones que se pueden aprovechar para registro de cambios o sincronización automática entre tablas por poner algunos ejemplos.

1.4) ¿Qué es y para qué sirve un STORED PROCEDURE?  
R: Sería lo similar a las funciones en programación, pero en este caso, código SQL que usamos repetidamente, este código queda guardado para poder reutilizarlo cuando lo queramos, podemos hacer store procedures para diferentes acciones como insertar, actualizar, hacer consultas o eliminar.

Considerando que las siguientes tablas:



Tienen los siguientes datos insertados:



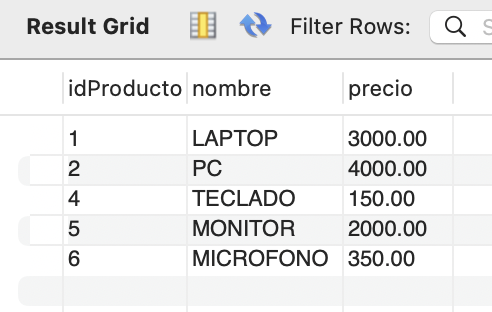
Hacer las consultas necesarias para:

1.5) Traer todos los productos que tengan una venta.

si esto se refiere a “Al menos una venta”:

**SELECT DISTINCT** p.idProducto, p.nombre, p.precio

**FROM** productos p **INNER** **JOIN** ventas v **ON** p.idProducto = v.idProducto;



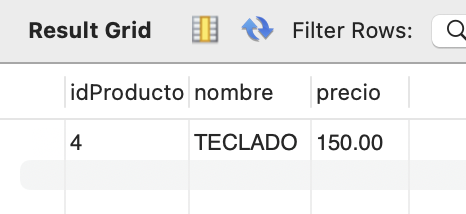
Pero si se refiere a “una y solo una venta”:

**SELECT** p.idProducto, p.nombre, p.precio

**FROM** productos p **INNER** **JOIN** ventas v **ON** p.idProducto = v.idProducto

**GROUP** **BY** p.idProducto, p.nombre, p.precio

**HAVING** **COUNT**(v.idVenta) = 1;

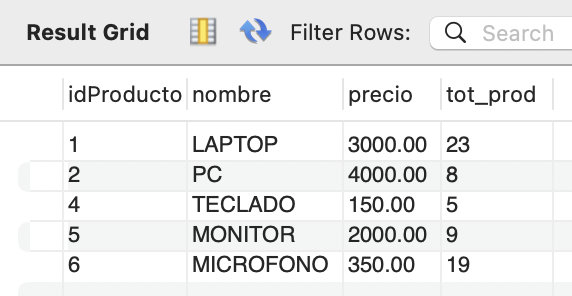


1.6) Traer todos los productos que tengan ventas y la cantidad total de productos vendidos.

**SELECT** p.idProducto, p.nombre, p.precio, **SUM**(v.cantidad) **AS** tot\_prod

**FROM** productos p **INNER** **JOIN** ventas v **ON** p.idProducto = v.idProducto

**GROUP** **BY** p.idProducto, p.nombre, p.precio;



1.7) Traer todos los productos (independientemente de si tienen ventas o no) y la suma total ($) vendida por producto.

**SELECT** p.idProducto, p.nombre, p.precio,

**IFNULL**(**SUM**(v.cantidad),0) **AS** tot\_prod,

**IFNULL**(**SUM**(v.cantidad) \* p.precio,0) **AS** tot\_venta

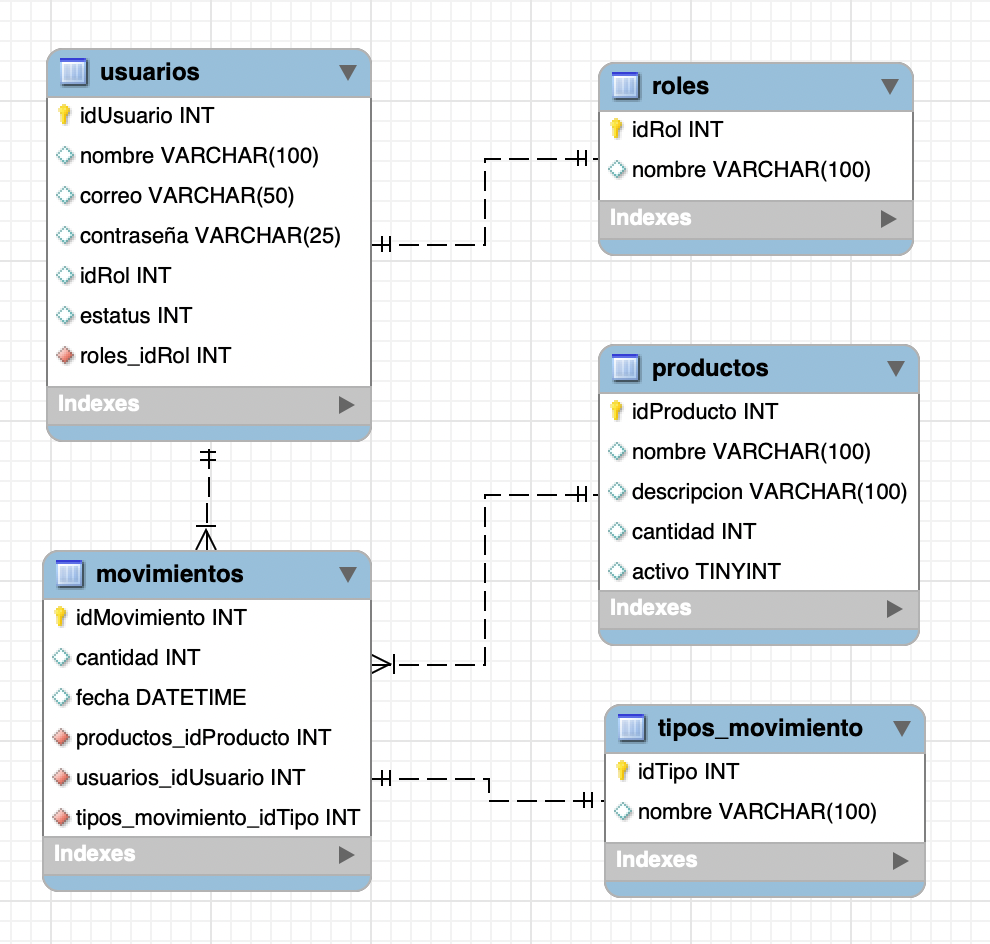
**FROM** productos p **LEFT** **JOIN** ventas v **ON** p.idProducto = v.idProducto

**GROUP** **BY** p.idProducto, p.nombre, p.precio;

(sorry por el color, se activó el modo nocturno)

2. **EJERCICIO PRÁCTICO: BD**

2.1) Crea un diagrama relacional de BD para el escenario descrito anteriormente.



2.2) Hacer el script para crear las tablas del punto anterior (en el punto 4.2 se especificará dónde debe anexarse

dicho script).