**Listar los productos con precio entre 20€ y 50 €.**

select id,name,price\_list

from product\_template

where price\_list between 20 and 50;

**Listar todos los productos vendidos.**

SELECT pt.id, pt.name

FROM product\_template pt

JOIN product\_product pp ON pt.id = pp.product\_tmpl\_id

JOIN sale\_order\_line sol ON pp.id = sol.product\_id;

**Listas los productos del almacén.**

select id,name

from stock\_quant

join on product\_product on stock\_quant.id=product\_product.id

join product\_template on product\_product.id=product\_template.id;

**Listar los productos vendidos que NO están en el almacén agrupados por fecha de venta.MAAAL**

select id,name

from product\_template

join sale\_order\_line on id.sale\_order\_line=id.product\_template

where sale\_order\_line.id is NULL

group by

**Listar los productos vendidos con un precio entre 20€ y 50 €.**

select id,name

from product\_template

join sale\_order\_line on id.sale\_order\_line=id.product\_template

where price\_list between 20 and 50;

**Listar los productos vendidos en un mes determinado con precio entre 20€ y 50 €.**

SELECT pt.id, pt.name

FROM product\_template pt

JOIN product\_product pp ON pt.id = pp.product\_tmpl\_id

JOIN sale\_order\_line sol ON pp.id = sol.product\_id

WHERE sol.price\_unit BETWEEN 20 AND 50;

**Calcular el total de las ventas realizadas de los productos con precio entre 20€ y 50 € en un mes determinado.**

La función EXTRACT se utiliza para extraer partes específicas de una fecha o una marca de tiempo en PostgreSQL. En tu consulta, se utiliza para obtener el mes y el año de la fecha de pedido (so.date\_order) y filtrar los resultados para el mes de diciembre del año 2023.

En el contexto de Odoo, sale\_order\_line representa las líneas individuales de productos en una orden de venta (sale\_order). Cada línea de pedido (sale\_order\_line) está vinculada a una orden de venta específica (sale\_order) a través del campo order\_id. S **in embargo, a menudo, la información principal sobre la orden de venta, como la fecha de la orden,**el cliente, etc., **se almacena en la tabla principal sale\_order.**

select sum(sol.price\_total)

from sale\_order\_line sol

join sale\_order so on...-->join solo de sol a so

join product\_product on ...-->el join este y el de abajo son independientes del de arriba

join product\_templae on ...

where product\_template.price\_list between 20 and 50

and EXTRACT(MONTH FROM sale\_order.date\_order)=12 AND EXTRACT(YEAR FROM so.date\_order)=2023

ChatGPT

¡Claro! A continuación, te proporcionaré una serie de ejercicios más avanzados utilizando las tablas mencionadas en Odoo. Puedes ejecutar estas consultas en DbVisualizer o cualquier otro cliente de base de datos compatible con PostgreSQL.

Listar los productos con precio entre 20€ y 50 €:

sql

Copy code

SELECT id, name, list\_price

FROM product\_template

WHERE list\_price BETWEEN 20 AND 50;

Listar todos los productos vendidos:

sql

Copy code

SELECT product\_template.name, sale\_order\_line.price\_unit, sale\_order\_line.product\_uom\_qty

FROM sale\_order\_line

JOIN product\_template ON sale\_order\_line.product\_id = product\_template.id;

Listar los productos del almacén:

sql

Copy code

SELECT product\_template.name, stock\_quant.qty

FROM stock\_quant

JOIN product\_template ON stock\_quant.product\_id = product\_template.id;

Listar los productos vendidos que NO están en el almacén agrupados por fecha de venta:

sql

Copy code

SELECT sale\_order.date\_order, product\_template.name, sale\_order\_line.product\_uom\_qty

FROM sale\_order\_line

JOIN product\_template ON sale\_order\_line.product\_id = product\_template.id

LEFT JOIN stock\_quant ON sale\_order\_line.product\_id = stock\_quant.product\_id

JOIN sale\_order ON sale\_order\_line.order\_id = sale\_order.id

WHERE stock\_quant.id IS NULL;

Listar los productos vendidos con precio entre 20€ y 50 €:

sql

Copy code

SELECT product\_template.name, sale\_order\_line.price\_unit, sale\_order\_line.product\_uom\_qty

FROM sale\_order\_line

JOIN product\_template ON sale\_order\_line.product\_id = product\_template.id

WHERE sale\_order\_line.price\_unit BETWEEN 20 AND 50;

Listar los productos vendidos en un mes determinado con precio entre 20€ y 50 €:

sql

Copy code

SELECT product\_template.name, sale\_order\_line.price\_unit, sale\_order\_line.product\_uom\_qty, sale\_order.date\_order

FROM sale\_order\_line

JOIN product\_template ON sale\_order\_line.product\_id = product\_template.id

JOIN sale\_order ON sale\_order\_line.order\_id = sale\_order.id

WHERE sale\_order.date\_order >= '2024-01-01' AND sale\_order.date\_order < '2024-02-01'

AND sale\_order\_line.price\_unit BETWEEN 20 AND 50;

Calcular el total de las ventas realizadas de los productos con precio entre 20€ y 50 € en un mes determinado:

sql

Copy code

SELECT SUM(sale\_order\_line.price\_unit \* sale\_order\_line.product\_uom\_qty) AS total\_ventas

FROM sale\_order\_line

JOIN product\_template ON sale\_order\_line.product\_id = product\_template.id

JOIN sale\_order ON sale\_order\_line.order\_id = sale\_order.id

WHERE sale\_order.date\_order >= '2024-01-01' AND sale\_order.date\_order < '2024-02-01'

AND sale\_order\_line.price\_unit BETWEEN 20 AND 50;

Estos ejercicios implican el uso de joins, condiciones de fecha y funciones agregadas. Asegúrate de adaptar las consultas según la estructura específica de tu base de datos y de cambiar las fechas según tus necesidades.

User

pero si me has puesto las mismas. Ponme otras consultas diferentes con algo mas de dificultad

ChatGPT

Mis disculpas por la repetición. Aquí te proporciono algunas consultas adicionales con un nivel de dificultad mayor:

**Listar los productos con el valor total de existencias en el almacén (valor = cantidad en almacén \* precio unitario):**

select product\_template.name, stock\_quant.qty\* product\_template.price\_list as valorTotal

from product\_template pt

join stock\_quant on product\_id....

sql

Copy code

SELECT product\_template.name, stock\_quant.qty \* product\_template.list\_price AS valor\_existencias

FROM stock\_quant

JOIN product\_template ON stock\_quant.product\_id = product\_template.id;

**Listar los productos que han sido vendidos más de una vez y mostrar la cantidad total vendida:**

sql

Copy code

SELECT product\_template.name, SUM(sale\_order\_line.product\_uom\_qty) AS cantidad\_total\_vendida

FROM sale\_order\_line

JOIN product\_template ON sale\_order\_line.product\_id = product\_template.id

GROUP BY product\_template.name

HAVING COUNT(sale\_order\_line.id) > 1;

**Calcular el promedio de tiempo que los productos pasan en el almacén antes de ser vendidos (diferencia entre fecha de venta y fecha de entrada en el almacén):**

sql

Copy code

SELECT product\_template.name, AVG(EXTRACT(EPOCH FROM (sale\_order.date\_order - stock\_quant.create\_date))) AS tiempo\_promedio

FROM sale\_order\_line

JOIN product\_template ON sale\_order\_line.product\_id = product\_template.id

JOIN stock\_quant ON sale\_order\_line.product\_id = stock\_quant.product\_id

JOIN sale\_order ON sale\_order\_line.order\_id = sale\_order.id

GROUP BY product\_template.name;

**Listar los productos que han tenido una fluctuación significativa en el precio (por ejemplo, variación de más del 20% en el último año):**

sql

Copy code

SELECT product\_template.name, MAX(product\_template.list\_price) AS precio\_maximo, MIN(product\_template.list\_price) AS precio\_minimo

FROM product\_template

WHERE EXTRACT(YEAR FROM NOW()) - EXTRACT(YEAR FROM product\_template.write\_date) = 1

GROUP BY product\_template.name

HAVING MAX(product\_template.list\_price) / MIN(product\_template.list\_price) > 1.2;

Estas consultas implican operaciones más avanzadas, como cálculos basados en fechas, funciones de agregación, y condiciones más complejas. Ajusta las consultas según tu esquema de base de datos y requerimientos específicos.

User

podrias ponerme otros 10 ejemplos de una dificultad similar. No olvides poner consultas relacionadas(algunas, no todas) con las tablas de COMPRAS(purchase\_order)

ChatGPT

Claro, aquí tienes otros 10 ejemplos de consultas más avanzadas, algunas de las cuales involucran la tabla de compras (purchase\_order):

**Listar los productos que tienen un stock mínimo inferior a la cantidad vendida en el último mes:**

sql

Copy code

SELECT product\_template.name, stock\_quant.qty AS stock\_actual, MIN(stock\_valuation\_layer.stock\_value) AS stock\_minimo

FROM stock\_quant

JOIN product\_template ON stock\_quant.product\_id = product\_template.id

JOIN stock\_valuation\_layer ON stock\_quant.product\_id = stock\_valuation\_layer.product\_id

WHERE stock\_quant.qty < (

SELECT SUM(sale\_order\_line.product\_uom\_qty)

FROM sale\_order\_line

JOIN sale\_order ON sale\_order\_line.order\_id = sale\_order.id

WHERE EXTRACT(MONTH FROM NOW() - INTERVAL '1 month') = EXTRACT(MONTH FROM sale\_order.date\_order)

)

GROUP BY product\_template.name;

**Calcular el valor total de las compras realizadas en el último trimestre:**

sql

Copy code

SELECT SUM(purchase\_order.amount\_total) AS total\_compras

FROM purchase\_order

WHERE EXTRACT(QUARTER FROM NOW() - INTERVAL '3 months') = EXTRACT(QUARTER FROM purchase\_order.date\_order);

**Listar los productos que han sido comprados pero no vendidos en el último semestre:**

sql

Copy code

SELECT product\_template.name

FROM product\_template

JOIN purchase\_order\_line ON product\_template.id = purchase\_order\_line.product\_id

LEFT JOIN sale\_order\_line ON product\_template.id = sale\_order\_line.product\_id

WHERE purchase\_order\_line.create\_date >= NOW() - INTERVAL '6 months' AND sale\_order\_line.id IS NULL;

Calcular el promedio de días que tarda un producto en ser vendido desde su compra:

sql

Copy code

SELECT product\_template.name, AVG(EXTRACT(EPOCH FROM (sale\_order.date\_order - purchase\_order\_line.create\_date))) AS tiempo\_promedio

FROM sale\_order\_line

JOIN product\_template ON sale\_order\_line.product\_id = product\_template.id

JOIN purchase\_order\_line ON product\_template.id = purchase\_order\_line.product\_id

JOIN sale\_order ON sale\_order\_line.order\_id = sale\_order.id

GROUP BY product\_template.name;

**Listar los productos que han tenido un aumento significativo en las compras en comparación con el mismo período del año pasado:**

sql

Copy code

SELECT product\_template.name, SUM(purchase\_order\_line.product\_qty) AS cantidad\_actual, SUM(purchase\_order\_line.product\_qty) / SUM(purchase\_order\_line.product\_qty\_last\_year) AS aumento\_porcentual

FROM purchase\_order\_line

JOIN product\_template ON purchase\_order\_line.product\_id = product\_template.id

WHERE EXTRACT(YEAR FROM NOW()) = EXTRACT(YEAR FROM purchase\_order\_line.create\_date)

AND EXTRACT(MONTH FROM NOW()) = EXTRACT(MONTH FROM purchase\_order\_line.create\_date)

GROUP BY product\_template.name

HAVING SUM(purchase\_order\_line.product\_qty) / SUM(purchase\_order\_line.product\_qty\_last\_year) > 1.5;

**Listar los proveedores que han suministrado productos vendidos en el último mes:**

sql

Copy code

SELECT DISTINCT purchase\_order.partner\_id, res\_partner.name

FROM purchase\_order

JOIN purchase\_order\_line ON purchase\_order.id = purchase\_order\_line.order\_id

JOIN sale\_order\_line ON purchase\_order\_line.product\_id = sale\_order\_line.product\_id

JOIN res\_partner ON purchase\_order.partner\_id = res\_partner.id

WHERE EXTRACT(MONTH FROM NOW() - INTERVAL '1 month') = EXTRACT(MONTH FROM sale\_order\_line.create\_date);

**Calcular el valor total de los productos vendidos en el último mes que fueron comprados más de una vez:**

sql

Copy code

SELECT SUM(sale\_order\_line.price\_unit \* sale\_order\_line.product\_uom\_qty) AS total\_ventas

FROM sale\_order\_line

JOIN product\_template ON sale\_order\_line.product\_id = product\_template.id

JOIN purchase\_order\_line ON product\_template.id = purchase\_order\_line.product\_id

WHERE EXTRACT(MONTH FROM NOW() - INTERVAL '1 month') = EXTRACT(MONTH FROM sale\_order\_line.create\_date)

GROUP BY product\_template.name

HAVING COUNT(purchase\_order\_line.id) > 1;

**Listar los productos comprados que no han sido vendidos en el último trimestre:**

sql

Copy code

SELECT product\_template.name

FROM product\_template

JOIN purchase\_order\_line ON product\_template.id = purchase\_order\_line.product\_id

LEFT JOIN sale\_order\_line ON product\_template.id = sale\_order\_line.product\_id

WHERE purchase\_order\_line.create\_date >= NOW() - INTERVAL '3 months' AND sale\_order\_line.id IS NULL;

**Calcular la diferencia de precio promedio entre las compras y las ventas de un producto en el último año:**

sql

Copy code

SELECT product\_template.name, AVG(sale\_order\_line.price\_unit - purchase\_order\_line.price\_unit) AS diferencia\_promedio

FROM product\_template

JOIN purchase\_order\_line ON product\_template.id = purchase\_order\_line.product\_id

JOIN sale\_order\_line ON product\_template.id = sale\_order\_line.product\_id

WHERE EXTRACT(YEAR FROM NOW()) = EXTRACT(YEAR FROM sale\_order\_line.create\_date)

GROUP BY product\_template.name;

**Listar los productos comprados que tienen un stock actual superior al stock mínimo en el almacén:**

sql

Copy code

SELECT product\_template.name, stock\_quant.qty AS stock\_actual, product\_template.stock\_minimo

FROM product\_template

JOIN stock\_quant ON product\_template.id = stock\_quant.product\_id

WHERE stock\_quant.qty > product\_template.stock\_minimo;