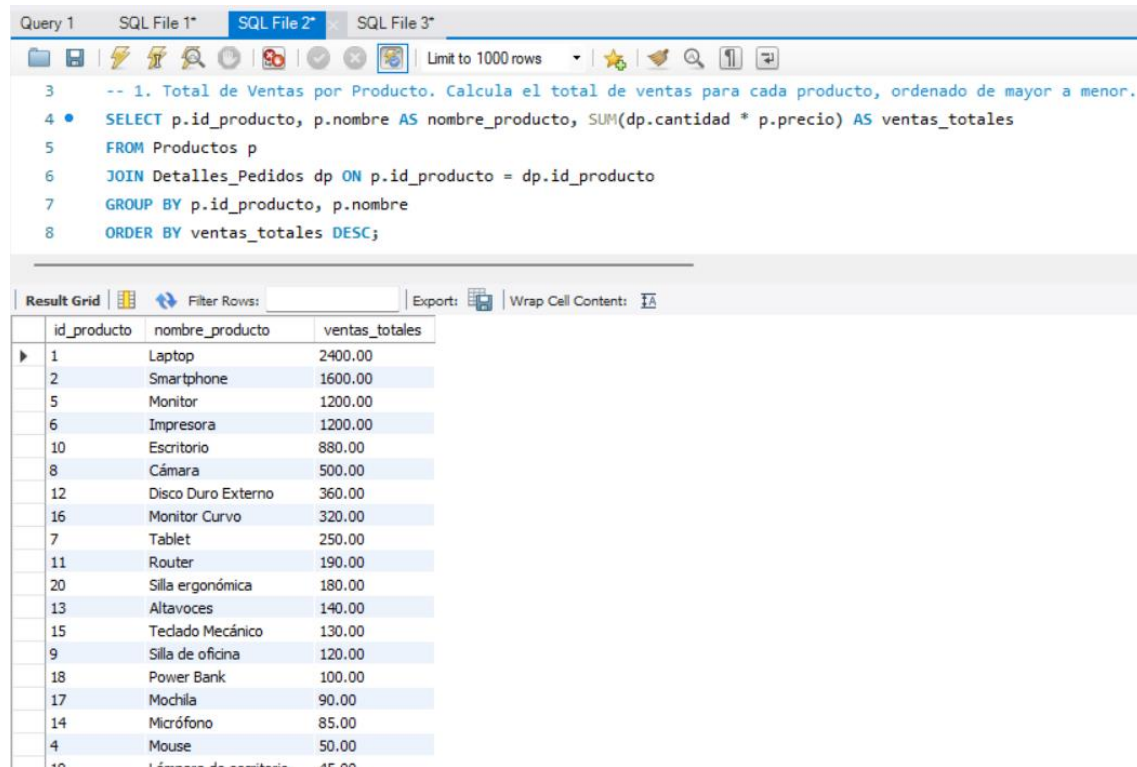


Evaluación Continua – Adrián Sobrevela 15/05/2024

Pregunta 1.



The screenshot shows a SQL IDE with a query editor and a result grid. The query is as follows:

```
-- 1. Total de Ventas por Producto. Calcula el total de ventas para cada producto, ordenado de mayor a menor.
SELECT p.id_producto, p.nombre AS nombre_producto, SUM(dp.cantidad * p.precio) AS ventas_totales
FROM Productos p
JOIN Detalles_Pedidos dp ON p.id_producto = dp.id_producto
GROUP BY p.id_producto, p.nombre
ORDER BY ventas_totales DESC;
```

The result grid displays the following data:

	id_producto	nombre_producto	ventas_totales
▶	1	Laptop	2400.00
	2	Smartphone	1600.00
	5	Monitor	1200.00
	6	Impresora	1200.00
	10	Escritorio	880.00
	8	Cámara	500.00
	12	Disco Duro Externo	360.00
	16	Monitor Curvo	320.00
	7	Tablet	250.00
	11	Router	190.00
	20	Silla ergonómica	180.00
	13	Altavoces	140.00
	15	Teclado Mecánico	130.00
	9	Silla de oficina	120.00
	18	Power Bank	100.00
	17	Mochila	90.00
	14	Micrófono	85.00
	4	Mouse	50.00
	19	Teclado inalámbrico	45.00

SELECT: Selecciona el ID del producto, el nombre del producto y calcula la suma de las ventas totales (cantidad * precio) para cada producto.

FROM: Especifica las tablas de donde se seleccionan los datos: Productos y Detalles_Pedidos.

JOIN: Une las tablas Productos y Detalles_Pedidos usando el ID del producto.

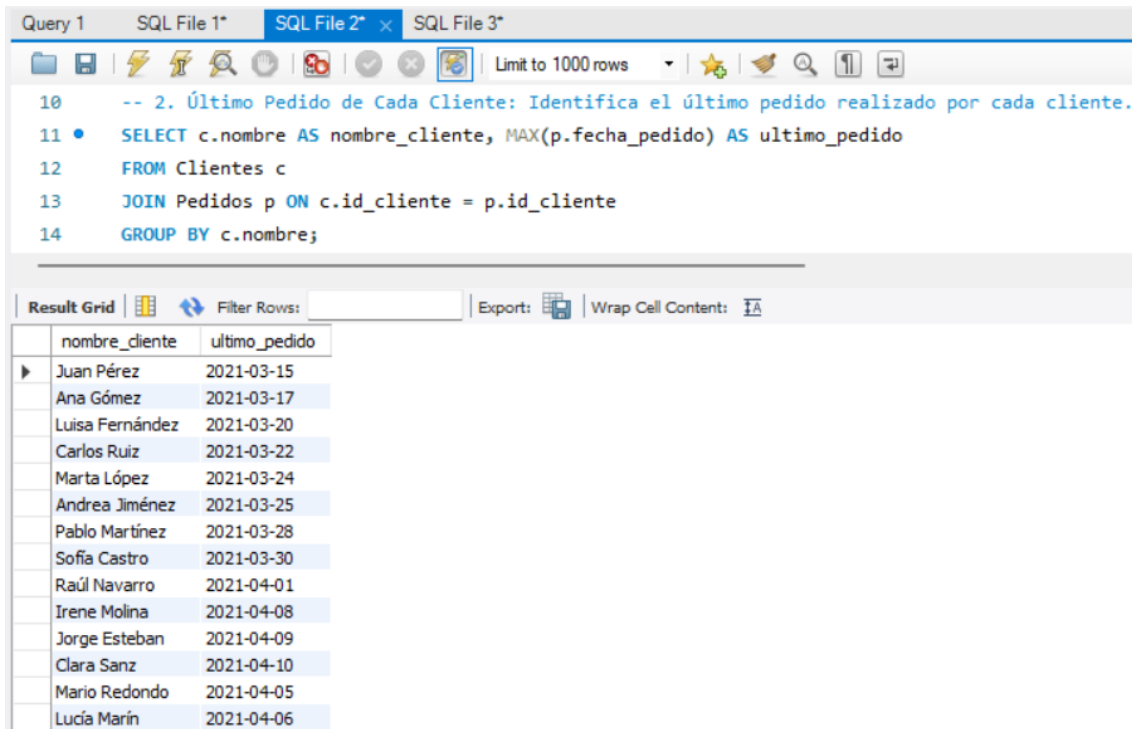
GROUP BY: Agrupa los resultados por ID de producto y nombre de producto.

SUM(): Calcula la suma de las ventas totales multiplicando la cantidad vendida por el precio de cada producto.

ORDER BY: Ordena los resultados por las ventas totales de forma descendente.

Esta consulta devuelve el total de ventas por producto, calculando las ventas totales (cantidad * precio) y ordenando los resultados de mayor a menor.

Pregunta 2.



The screenshot shows a SQL IDE interface with a query editor and a results grid. The query editor contains the following SQL code:

```
10 -- 2. Último Pedido de Cada Cliente: Identifica el último pedido realizado por cada cliente.
11 • SELECT c.nombre AS nombre_cliente, MAX(p.fecha_pedido) AS ultimo_pedido
12 FROM Clientes c
13 JOIN Pedidos p ON c.id_cliente = p.id_cliente
14 GROUP BY c.nombre;
```

The results grid displays the following data:

nombre_cliente	ultimo_pedido
Juan Pérez	2021-03-15
Ana Gómez	2021-03-17
Luisa Fernández	2021-03-20
Carlos Ruiz	2021-03-22
Marta López	2021-03-24
Andrea Jiménez	2021-03-25
Pablo Martínez	2021-03-28
Sofía Castro	2021-03-30
Raúl Navarro	2021-04-01
Irene Molina	2021-04-08
Jorge Esteban	2021-04-09
Clara Sanz	2021-04-10
Mario Redondo	2021-04-05
Lucía Marín	2021-04-06

SELECT: Selecciona las columnas que queremos mostrar en el resultado.

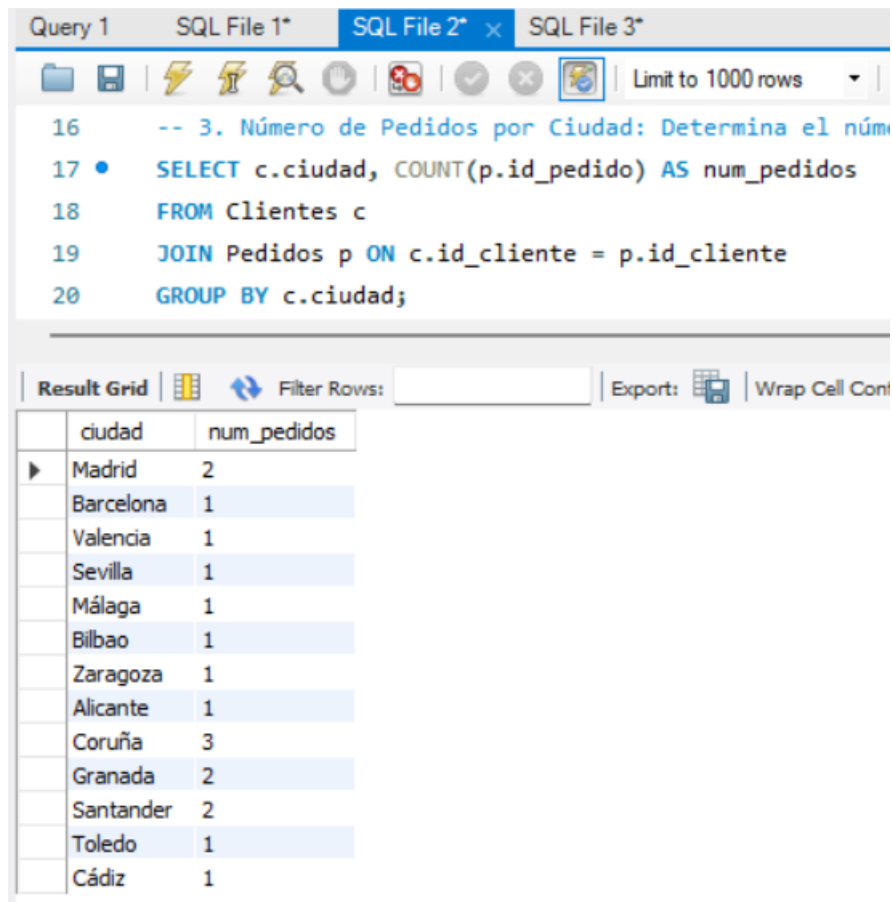
FROM: Especifica las tablas de donde se seleccionan los datos: Clientes y Pedidos.

JOIN: Une las tablas Clientes y Pedidos usando el ID de cliente.

GROUP BY: Agrupa los resultados por el nombre del cliente.

MAX(): Devuelve el valor máximo de la fecha de pedido para cada cliente, lo que representa su último pedido.

Pregunta 3.



The screenshot shows a SQL IDE interface with a query editor and a results grid. The query editor contains the following SQL code:

```
-- 3. Número de Pedidos por Ciudad: Determina el número de pedidos por ciudad.
SELECT c.ciudad, COUNT(p.id_pedido) AS num_pedidos
FROM Clientes c
JOIN Pedidos p ON c.id_cliente = p.id_cliente
GROUP BY c.ciudad;
```

The results grid displays the following data:

ciudad	num_pedidos
Madrid	2
Barcelona	1
Valencia	1
Sevilla	1
Málaga	1
Bilbao	1
Zaragoza	1
Alicante	1
Coruña	3
Granada	2
Santander	2
Toledo	1
Cádiz	1

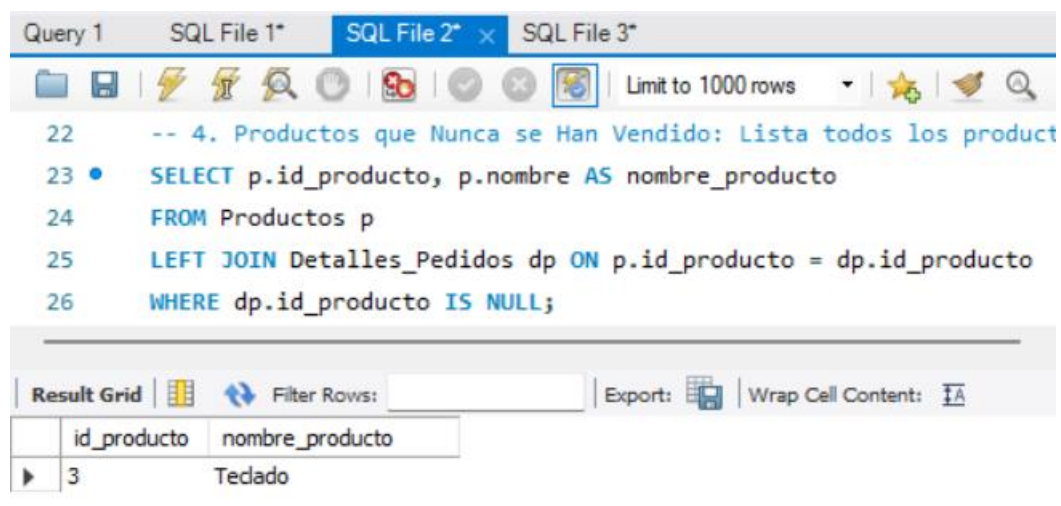
SELECT: Selecciona las columnas que queremos mostrar en el resultado. En este caso, seleccionamos la columna "ciudad" de la tabla Clientes y usamos la función **COUNT()** para contar el número de pedidos (usando la columna "id_pedido" de la tabla Pedidos), renombrando el resultado como "num_pedidos".

FROM: Especifica las tablas de donde se seleccionan los datos: Clientes y Pedidos.

JOIN: Une las tablas Clientes y Pedidos usando el ID del cliente.

GROUP BY: Agrupa los resultados por la columna "ciudad" en la tabla Clientes. Esto significa que obtendremos un solo resultado por cada ciudad.

Pregunta 4.



The screenshot shows a SQL IDE interface with a query editor and a results grid. The query editor contains the following SQL code:

```
-- 4. Productos que Nunca se Han Vendido: Lista todos los product
SELECT p.id_producto, p.nombre AS nombre_producto
FROM Productos p
LEFT JOIN Detalles_Pedidos dp ON p.id_producto = dp.id_producto
WHERE dp.id_producto IS NULL;
```

The results grid displays the following data:

id_producto	nombre_producto
3	Teclado

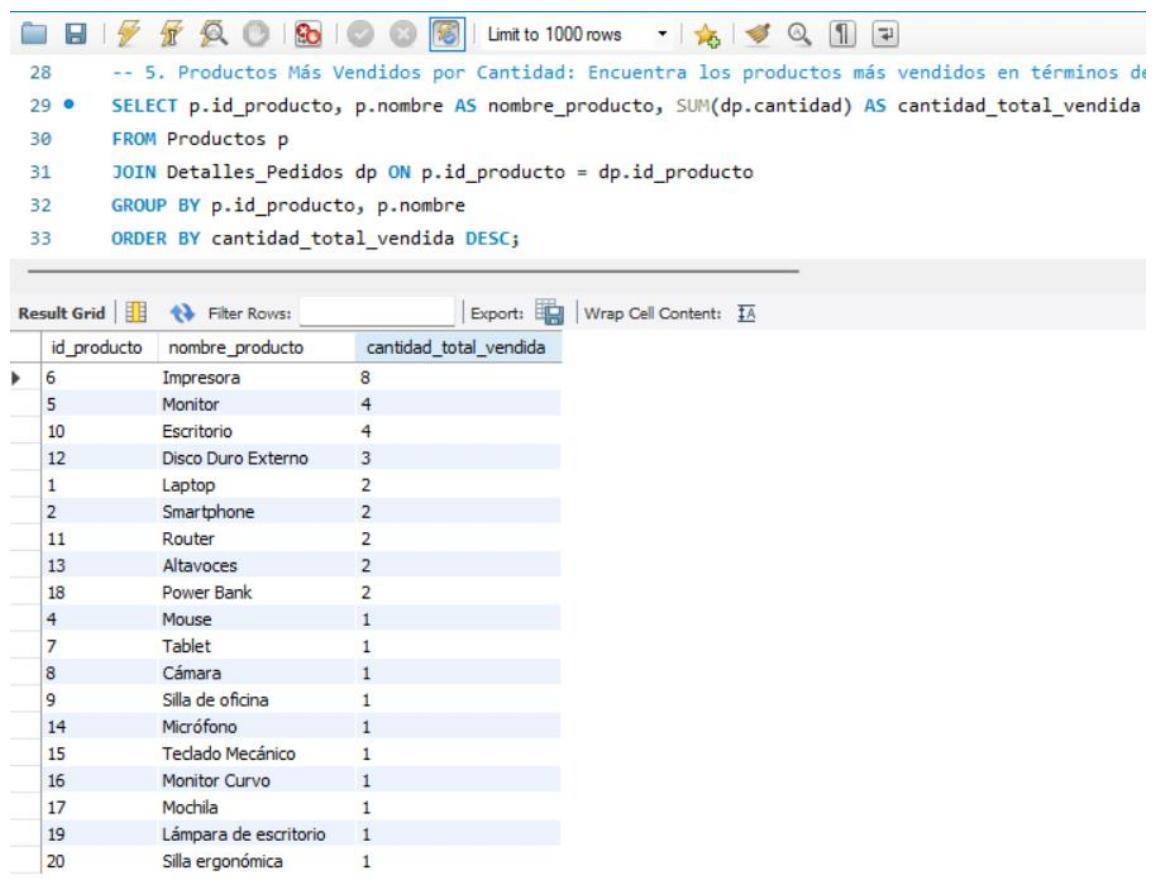
SELECT: Selecciona las columnas que queremos mostrar en el resultado. En este caso, seleccionamos el ID del producto y el nombre del producto de la tabla Productos.

FROM: Especifica la tabla de donde se seleccionan los datos: Productos.

LEFT JOIN: Unimos la tabla Productos con la tabla Detalles_Pedidos usando el ID del producto. Utilizamos un LEFT JOIN para asegurarnos de incluir todos los productos, incluso aquellos que no tienen detalles de pedidos.

WHERE: Filtramos los resultados para incluir solo aquellos productos que nunca han sido parte de un pedido. Esto lo hacemos buscando los registros en Detalles_Pedidos donde el ID del producto es NULL, lo que indica que no hay registros de pedidos para ese producto.

Pregunta 5.



```
28 -- 5. Productos Más Vendidos por Cantidad: Encuentra los productos más vendidos en términos de
29 • SELECT p.id_producto, p.nombre AS nombre_producto, SUM(dp.cantidad) AS cantidad_total_vendida
30 FROM Productos p
31 JOIN Detalles_Pedidos dp ON p.id_producto = dp.id_producto
32 GROUP BY p.id_producto, p.nombre
33 ORDER BY cantidad_total_vendida DESC;
```

	id_producto	nombre_producto	cantidad_total_vendida
▶	6	Impresora	8
	5	Monitor	4
	10	Escritorio	4
	12	Disco Duro Externo	3
	1	Laptop	2
	2	Smartphone	2
	11	Router	2
	13	Altavoces	2
	18	Power Bank	2
	4	Mouse	1
	7	Tablet	1
	8	Cámara	1
	9	Silla de oficina	1
	14	Micrófono	1
	15	Teclado Mecánico	1
	16	Monitor Curvo	1
	17	Mochila	1
	19	Lámpara de escritorio	1
	20	Silla ergonómica	1

SELECT: Selecciona las columnas que queremos mostrar en el resultado. En este caso, seleccionamos el ID del producto, el nombre del producto y calculamos la suma de la cantidad vendida de cada producto.

FROM: Especifica la tabla de donde se seleccionan los datos: Productos.

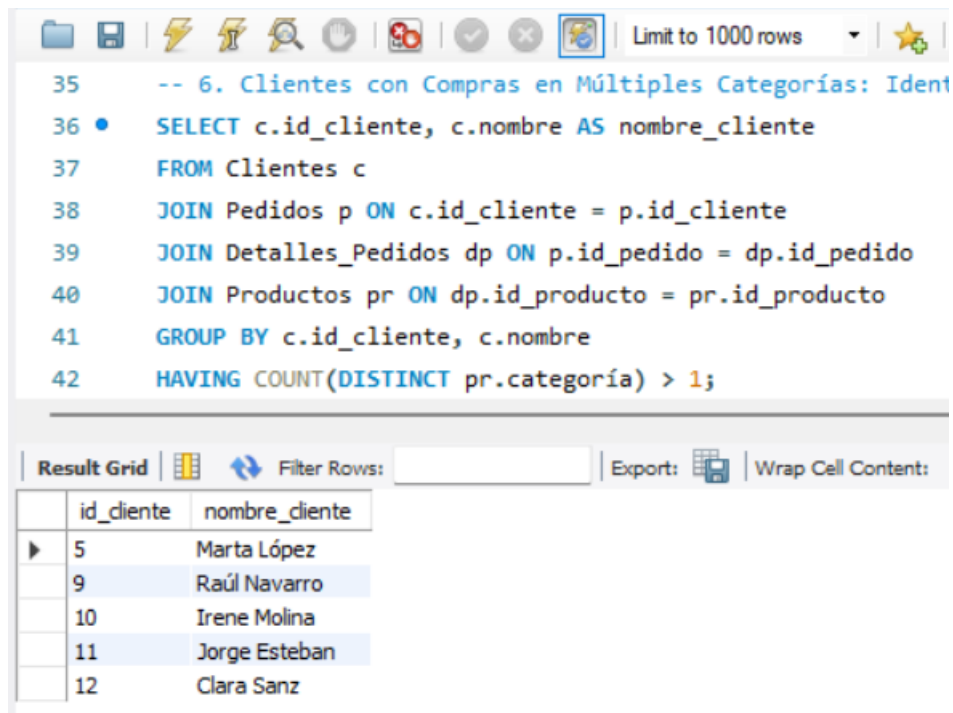
JOIN: Unimos la tabla Productos con la tabla Detalles_Pedidos usando el ID del producto.

GROUP BY: Agrupamos los resultados por el ID del producto y el nombre del producto. Esto nos permite obtener una suma total de la cantidad vendida para cada producto.

SUM(): Calculamos la suma de la cantidad vendida de cada producto.

ORDER BY: Ordenamos los resultados por la cantidad total vendida de forma descendente, para que los productos más vendidos aparezcan primero en el resultado.

Pregunta 6.



```
35  -- 6. Clientes con Compras en Múltiples Categorías: Ident
36  •  SELECT c.id_cliente, c.nombre AS nombre_cliente
37      FROM Clientes c
38      JOIN Pedidos p ON c.id_cliente = p.id_cliente
39      JOIN Detalles_Pedidos dp ON p.id_pedido = dp.id_pedido
40      JOIN Productos pr ON dp.id_producto = pr.id_producto
41      GROUP BY c.id_cliente, c.nombre
42      HAVING COUNT(DISTINCT pr.categoría) > 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	id_cliente	nombre_cliente
▶	5	Marta López
	9	Raúl Navarro
	10	Irene Molina
	11	Jorge Esteban
	12	Clara Sanz

SELECT: Selecciona las columnas que queremos mostrar en el resultado. En este caso, seleccionamos el ID del cliente y su nombre.

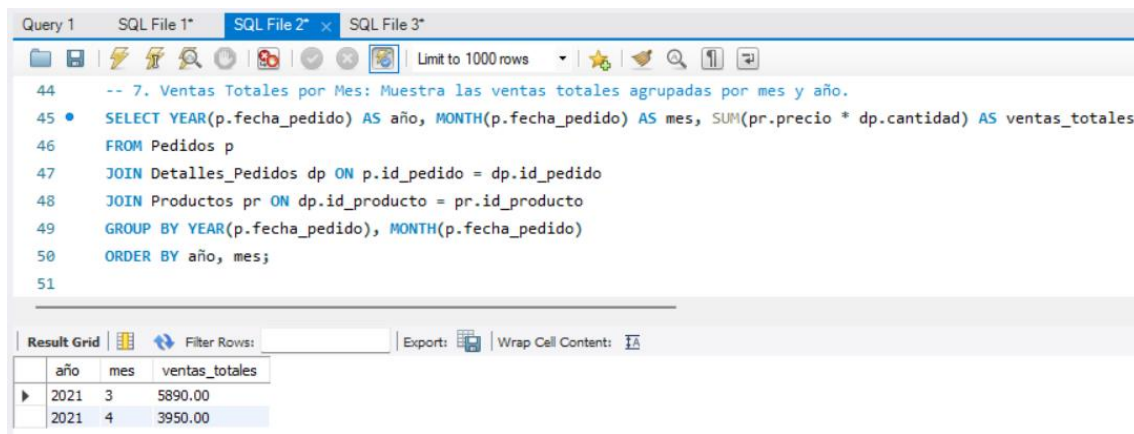
FROM: Especifica las tablas de donde se seleccionan los datos: Clientes, Pedidos, Detalles_Pedidos y Productos.

JOIN: Unimos las tablas necesarias para relacionar clientes, pedidos, detalles de pedidos y productos.

GROUP BY: Agrupamos los resultados por el ID del cliente y su nombre. Esto nos permite contar el número de categorías diferentes en las que un cliente ha realizado compras.

HAVING: Esta cláusula se utiliza junto con GROUP BY para filtrar los resultados según una condición agregada. En este caso, estamos filtrando los clientes que tienen más de una categoría de productos distintas en sus compras.

Pregunta 7.



The screenshot shows a SQL IDE with a query editor and a results grid. The query is as follows:

```
-- 7. Ventas Totales por Mes: Muestra las ventas totales agrupadas por mes y año.
SELECT YEAR(p.fecha_pedido) AS año, MONTH(p.fecha_pedido) AS mes, SUM(pr.precio * dp.cantidad) AS ventas_totales
FROM Pedidos p
JOIN Detalles_Pedidos dp ON p.id_pedido = dp.id_pedido
JOIN Productos pr ON dp.id_producto = pr.id_producto
GROUP BY YEAR(p.fecha_pedido), MONTH(p.fecha_pedido)
ORDER BY año, mes;
```

The results grid displays the following data:

año	mes	ventas_totales
2021	3	5890.00
2021	4	3950.00



A zoomed-in view of the results grid from the previous screenshot, showing the same data:

año	mes	ventas_totales
2021	3	5890.00
2021	4	3950.00

SELECT: Selecciona las columnas que queremos mostrar en el resultado. En este caso, seleccionamos el año y el mes de la fecha de pedido, así como la suma de las ventas totales (precio del producto multiplicado por la cantidad vendida).

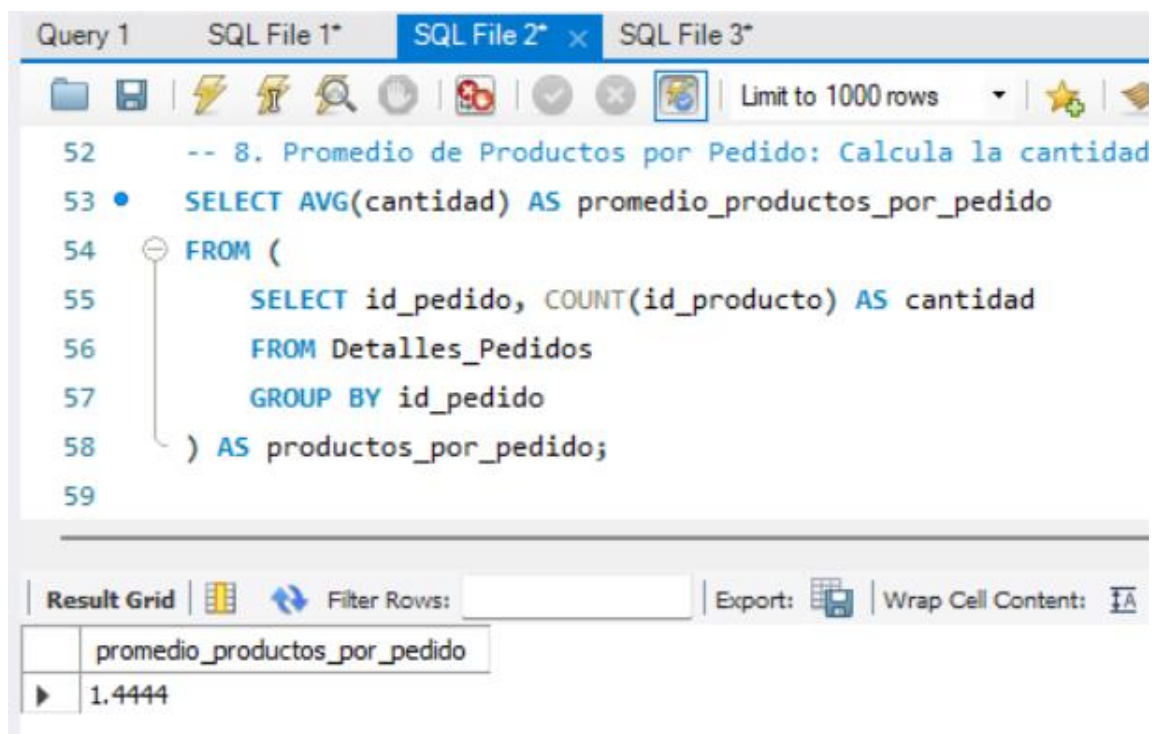
FROM: Especifica las tablas de donde se seleccionan los datos: Pedidos, Detalles_Pedidos y Productos.

JOIN: Unimos las tablas necesarias para relacionar pedidos, detalles de pedidos y productos.

GROUP BY: Agrupamos los resultados por el año y el mes de la fecha de pedido. Esto nos permite sumar las ventas totales para cada mes y año.

ORDER BY: Ordenamos los resultados por año y mes para mostrarlos en orden cronológico.

Pregunta 8.



The screenshot shows a SQL IDE interface with a query editor and a result grid. The query editor contains the following SQL code:

```
52  -- 8. Promedio de Productos por Pedido: Calcula la cantidad
53  •  SELECT AVG(cantidad) AS promedio_productos_por_pedido
54  FROM (
55      SELECT id_pedido, COUNT(id_producto) AS cantidad
56      FROM Detalles_Pedidos
57      GROUP BY id_pedido
58  ) AS productos_por_pedido;
59
```

The result grid shows the following data:

promedio_productos_por_pedido
1.4444

SELECT: Selecciona la función **AVG()** para calcular el promedio de la cantidad de productos por pedido.

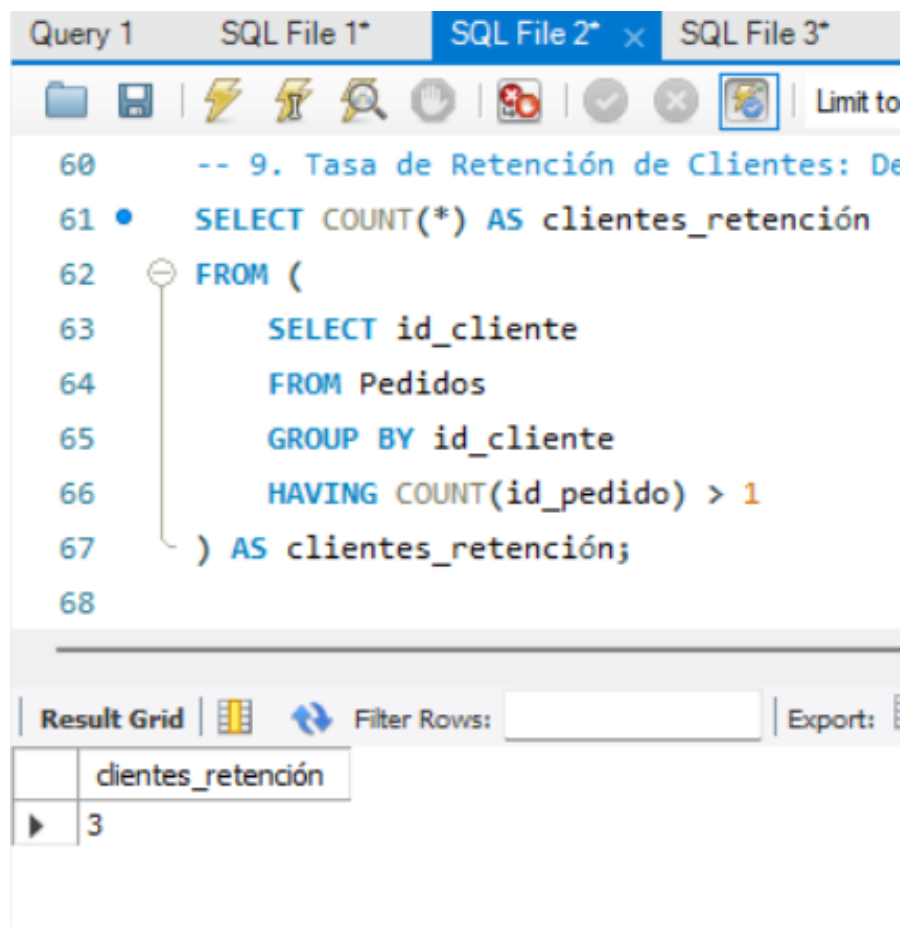
FROM: Utilizamos una subconsulta para obtener la cantidad de productos por pedido.

SELECT (subconsulta): Dentro de la subconsulta, seleccionamos el ID del pedido y contamos la cantidad de productos por pedido utilizando **COUNT()**.

GROUP BY (subconsulta): Agrupamos los resultados de la subconsulta por el ID del pedido.

AVG(): Calculamos el promedio de la cantidad de productos por pedido utilizando la función **AVG()** sobre los resultados de la subconsulta.

Pregunta 9.



The screenshot shows a SQL IDE window with multiple tabs: 'Query 1', 'SQL File 1*', 'SQL File 2*' (selected), and 'SQL File 3*'. The toolbar includes icons for file operations, execution, and a 'Limit to' dropdown. The SQL editor contains the following code:

```
60      -- 9. Tasa de Retención de Clientes: De
61 •    SELECT COUNT(*) AS clientes_retención
62      FROM (
63          SELECT id_cliente
64          FROM Pedidos
65          GROUP BY id_cliente
66          HAVING COUNT(id_pedido) > 1
67      ) AS clientes_retención;
68
```

Below the editor is a 'Result Grid' section with a 'Filter Rows' input and an 'Export' button. The result grid displays a single row with the column 'clientes_retención' and the value '3'.

clientes_retención
3

SELECT: Selecciona la función **COUNT()** para contar el número de clientes que han realizado pedidos en más de una ocasión.

FROM: Utilizamos una subconsulta para obtener los clientes que han realizado más de un pedido.

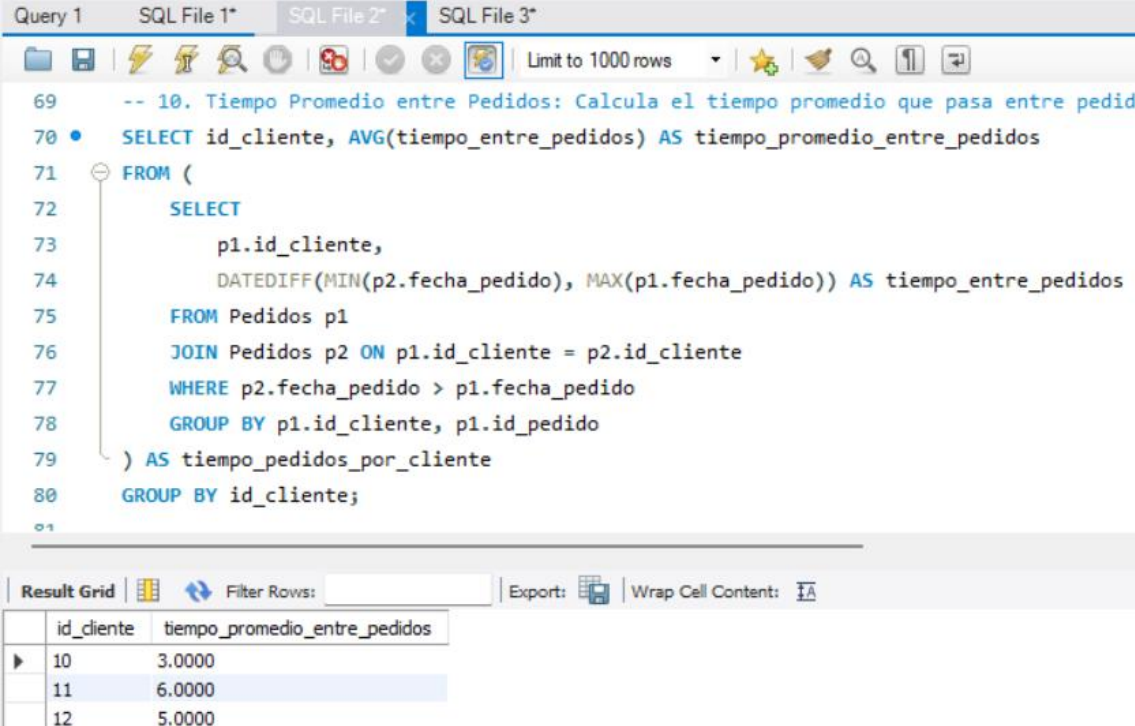
SELECT (subconsulta): Dentro de la subconsulta, seleccionamos el ID del cliente.

GROUP BY (subconsulta): Agrupamos los resultados de la subconsulta por el ID del cliente.

HAVING (subconsulta): Utilizamos la cláusula HAVING para filtrar solo aquellos clientes que tienen más de un pedido, es decir, que han realizado pedidos en más de una ocasión.

COUNT(): Contamos el número de clientes que cumplen con la condición de tener más de un pedido.

Pregunta 10.



The screenshot shows a SQL IDE with a query editor and a result grid. The query is as follows:

```
69 -- 10. Tiempo Promedio entre Pedidos: Calcula el tiempo promedio que pasa entre pedid
70 • SELECT id_cliente, AVG(tiempo_entre_pedidos) AS tiempo_promedio_entre_pedidos
71 FROM (
72     SELECT
73         p1.id_cliente,
74         DATEDIFF(MIN(p2.fecha_pedido), MAX(p1.fecha_pedido)) AS tiempo_entre_pedidos
75     FROM Pedidos p1
76     JOIN Pedidos p2 ON p1.id_cliente = p2.id_cliente
77     WHERE p2.fecha_pedido > p1.fecha_pedido
78     GROUP BY p1.id_cliente, p1.id_pedido
79 ) AS tiempo_pedidos_por_cliente
80 GROUP BY id_cliente;
```

The result grid shows the following data:

id_cliente	tiempo_promedio_entre_pedidos
10	3.0000
11	6.0000
12	5.0000

1.Subconsulta para calcular el tiempo entre pedidos por cliente:

Se realiza una subconsulta para calcular la diferencia de tiempo en días entre cada pedido consecutivo de cada cliente.

Se selecciona el ID del cliente de la tabla Pedidos como 'p1.id_cliente'.

Se utiliza la función DATEDIFF() para calcular la diferencia en días entre la fecha del siguiente pedido ('MIN(p2.fecha_pedido)') y la fecha del pedido actual ('MAX(p1.fecha_pedido)').

Se unen las tablas de pedidos consigo mismas ('JOIN Pedidos p2') para comparar cada pedido con el siguiente pedido del mismo cliente.

Se agrega una condición en la cláusula WHERE para asegurar que la fecha del siguiente pedido sea posterior a la fecha del pedido actual.

Se agrupan los resultados por el ID del cliente ('p1.id_cliente') y el ID del pedido actual ('p1.id_pedido').

2.Consulta principal para calcular el promedio del tiempo entre pedidos por cliente:

Se utiliza la subconsulta como una tabla derivada ('AS tiempo_pedidos_por_cliente').

Se selecciona el ID del cliente ('id_cliente') y se calcula el promedio de las diferencias de tiempo entre pedidos utilizando la función AVG().

Finalmente, se agrupan los resultados por el ID del cliente ('id_cliente') para obtener el promedio del tiempo entre pedidos para cada cliente.

