UNIVERSIDAD PERUANA LOS ANDES

FACULTAD INGENIERIA INGENIERÍA DE SISTEMAS Y COMPUTACIÓN

MANUAL DE CONSULTAS OBTENER, AGRUPAR Y ANALIZAR INFORMACIÓN

ASIGNATURA: BASE DE DATOS II

DOCENTE: FERNÁNDEZ BEJARANO RAUL

ESTUDIANTE: Bonifacio Hilario Erick

CÓDIGO: S01238F

HUANCAYO-2025

INTRODUCCIÓN

El presente manual tiene como finalidad recopilar y ejecutar una serie de **consultas SQL** aplicadas a la base de datos **QhatuPERU**, desarrollada en Microsoft SQL Server.
Estas consultas permiten **obtener, agrupar y analizar información** relacionada con artículos, órdenes, guías de envío, proveedores y transportistas.
A través del uso de **instrucciones SELECT, funciones de agregación y cláusulas como GROUP BY, HAVING y JOIN**, se busca reforzar el conocimiento práctico en la manipulación de datos relacionales, herramienta esencial para el desarrollo de sistemas empresariales eficientes.

OBJETIVO GENERAL

Desarrollar y comprender el funcionamiento de diferentes **consultas SQL** que permitan realizar operaciones de selección, filtrado, agrupación y cálculo dentro de la base de datos **QhatuPERU**, con el propósito de fortalecer las habilidades analíticas en la gestión de datos.

OBJETIVOS ESPECÍFICOS

1. Aplicar funciones de agregación (`SUM`, `AVG`, `COUNT`, `MAX`, `MIN`) en distintos contextos de consulta.
2. Realizar combinaciones de tablas mediante `JOIN` para relacionar información entre entidades.
3. Utilizar `GROUP BY` y `HAVING` para generar reportes agrupados y filtrados.
4. Analizar los resultados obtenidos y explicar el propósito de cada consulta.

CONSULTAS SQL

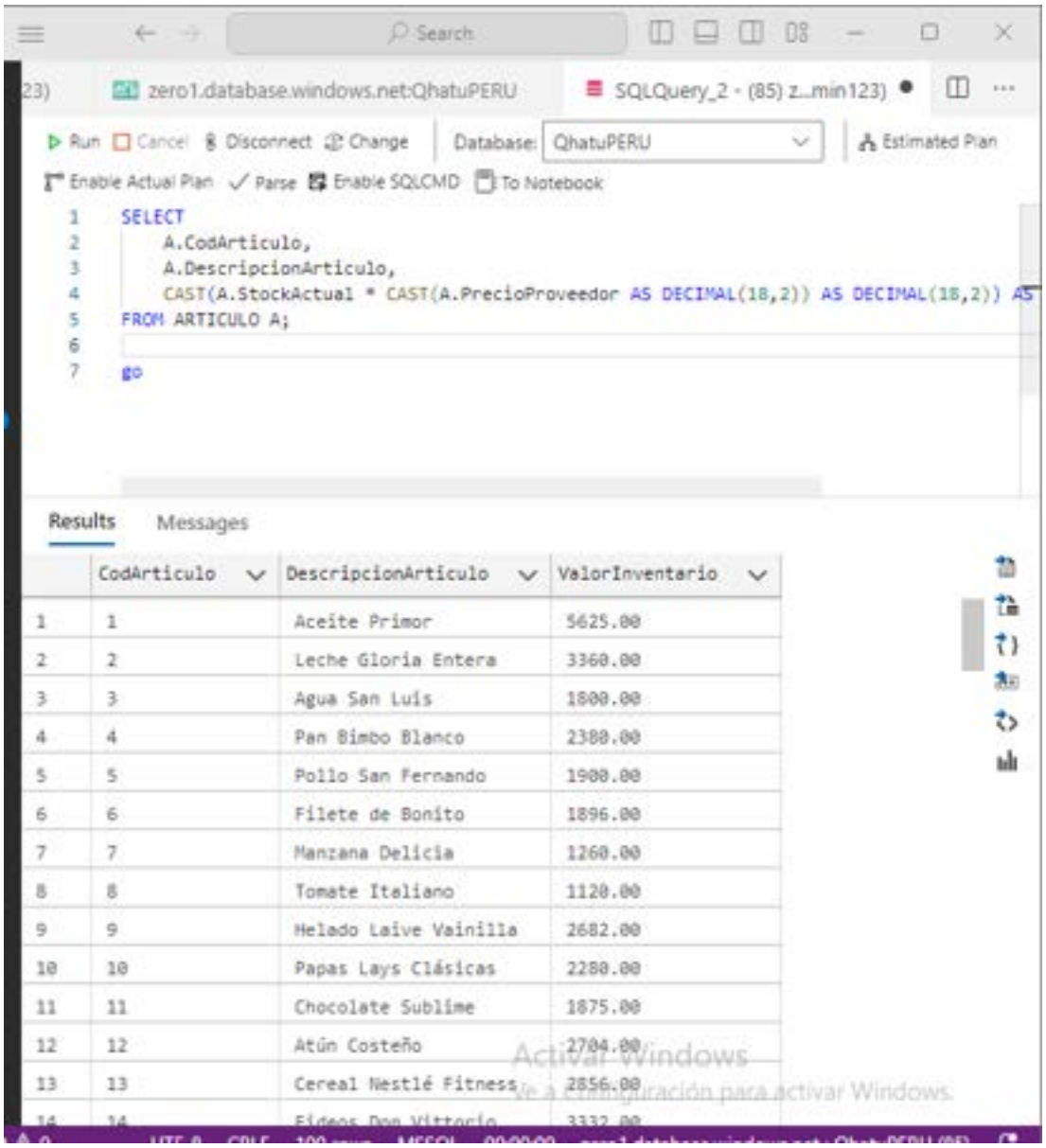
CONSULTA 1

Enunciado: Mostrar CodArticulo, DescripcionArticulo y ValorInventario.

Consulta SQL:

```
SELECT
    A.CodArticulo,
    A.DescripcionArticulo,
    CAST(A.StockActual * CAST(A.PrecioProveedor AS DECIMAL(18,2)) AS
DECIMAL(18,2)) AS ValorInventario
```

```
FROM ARTICULO A;
```



Explicación: Multiplica stock por precio proveedor por fila; se usa CAST para asegurar precisión decimal.

Consulta 2

Enunciado: Calcular el total monetario del inventario.

Consulta SQL:

```
SELECT
    CAST(SUM(A.StockActual * CAST(A.PrecioProveedor AS
DECIMAL(18,2))) AS DECIMAL(18,2)) AS TotalInventario
```

```
FROM ARTICULO A;
```

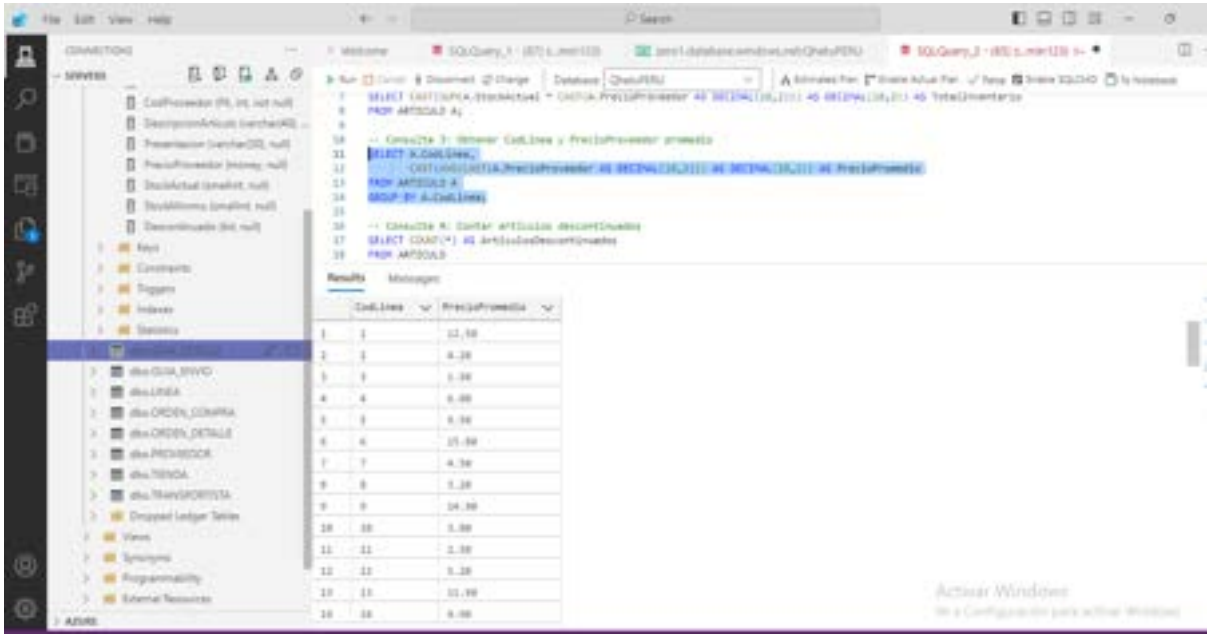
Explicación: Usa SUM para obtener el valor total del inventario en base a stock y precio proveedor.

Consulta 3

Enunciado: Obtener CodLínea y PrecioProveedor promedio.

Consulta SQL:

```
SELECT
    A.CodLinea,
    CAST(AVG(CAST(A.PrecioProveedor AS DECIMAL(18,2))) AS
DECIMAL(18,2)) AS PrecioPromedio
FROM ARTICULO A
GROUP BY A.CodLinea;
```



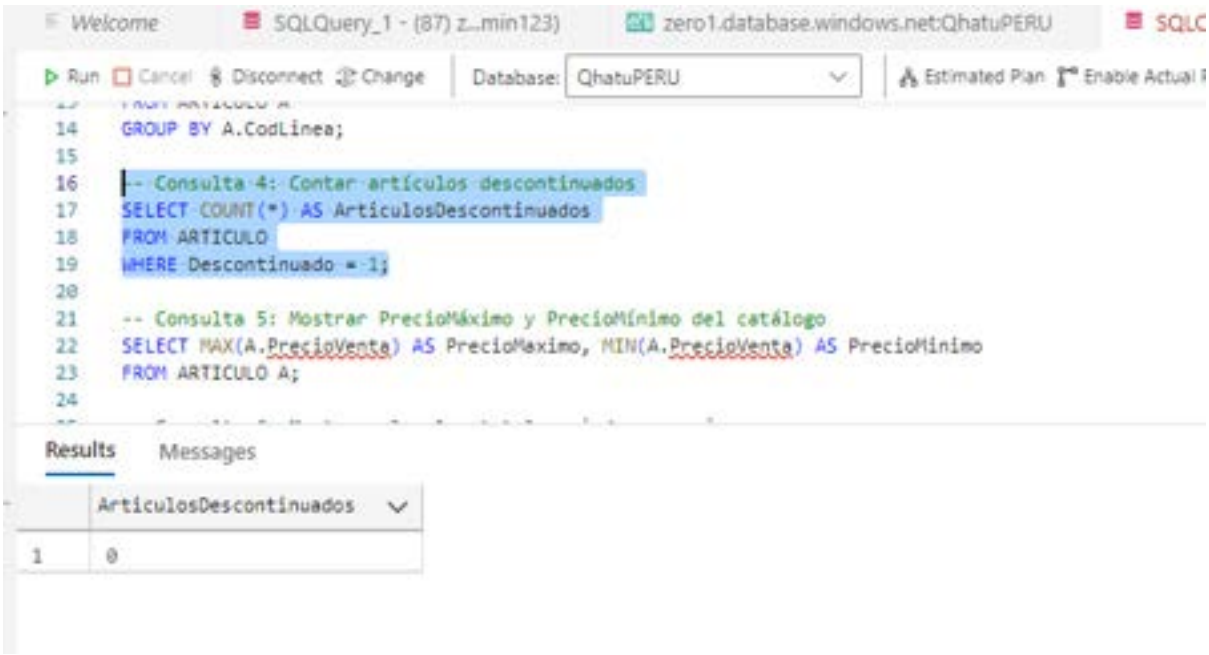
Explicación: Agrupa por línea de producto y calcula el precio promedio de proveedor.

Consulta 4

Enunciado: Contar artículos descontinuados.

Consulta SQL:

```
SELECT
    COUNT(*) AS ArticulosDescontinuados
FROM ARTICULO
WHERE Descontinuado = 1;
```



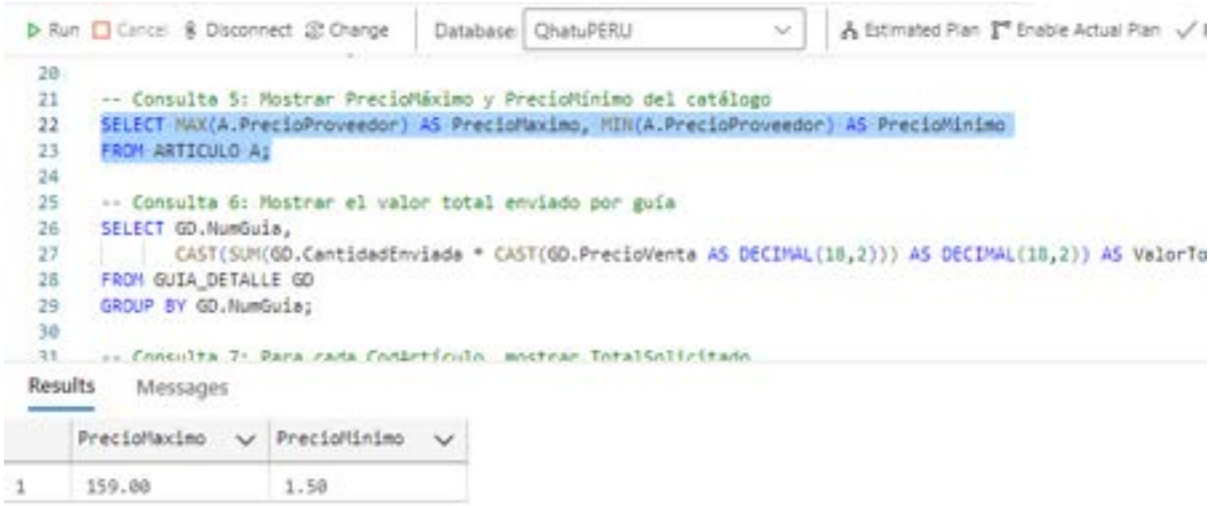
Explicación: Cuenta cuántos artículos están marcados como descontinuados.

Consulta 5

Enunciado: Mostrar PrecioMáximo y PrecioMínimo del catálogo.

Consulta SQL:

```
SELECT
    MAX(A. PrecioProveedor) AS PrecioMaximo,
    MIN(A. PrecioProveedor) AS PrecioMinimo
FROM ARTICULO A;
```



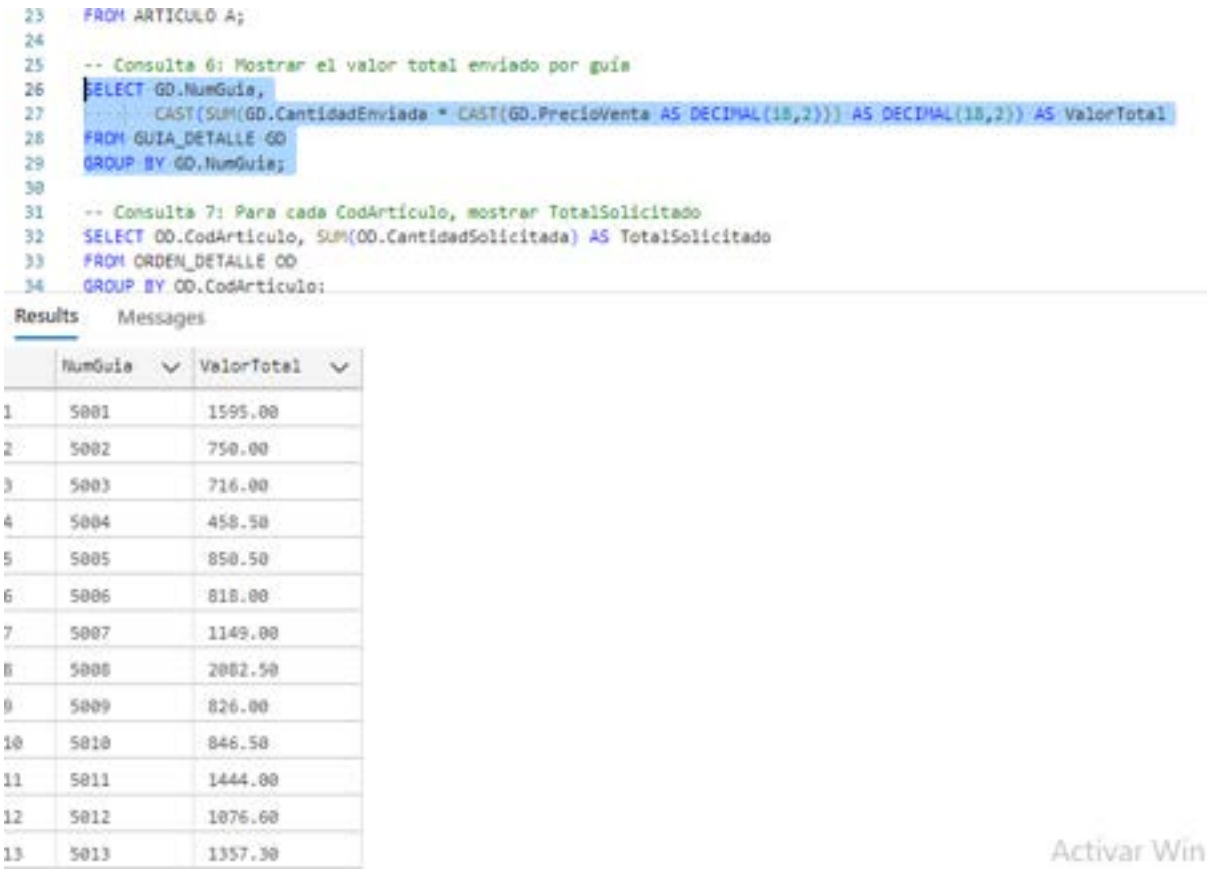
Explicación: Obtiene los valores extremos del precio del provedor en la tabla ARTICULO.

Consulta 6

Enunciado: Mostrar el valor total enviado por guía.

Consulta SQL:

```
SELECT
    GD.NumGuia,
    CAST(SUM(GD.CantidadEnviada * CAST(GD.PrecioVenta AS
DECIMAL(18,2))) AS DECIMAL(18,2)) AS ValorTotal
FROM GUIA_DETALLE GD
GROUP BY GD.NumGuia;
```



```
FROM ARTICULO A;

-- Consulta 6: Mostrar el valor total enviado por guía
SELECT GD.NumGuia,
       CAST(SUM(GD.CantidadEnviada * CAST(GD.PrecioVenta AS DECIMAL(18,2))) AS DECIMAL(18,2)) AS ValorTotal
FROM GUIA_DETALLE GD
GROUP BY GD.NumGuia;

-- Consulta 7: Para cada CodArticulo, mostrar TotalSolicitado
SELECT OD.CodArticulo, SUM(OD.CantidadSolicitada) AS TotalSolicitado
FROM ORDEN_DETALLE OD
GROUP BY OD.CodArticulo;
```

Results | Messages

	NumGuia	ValorTotal
1	5001	1595.00
2	5002	750.00
3	5003	716.00
4	5004	458.50
5	5005	850.50
6	5006	818.00
7	5007	1149.00
8	5008	2082.50
9	5009	826.00
10	5010	846.50
11	5011	1444.00
12	5012	1076.60
13	5013	1357.30

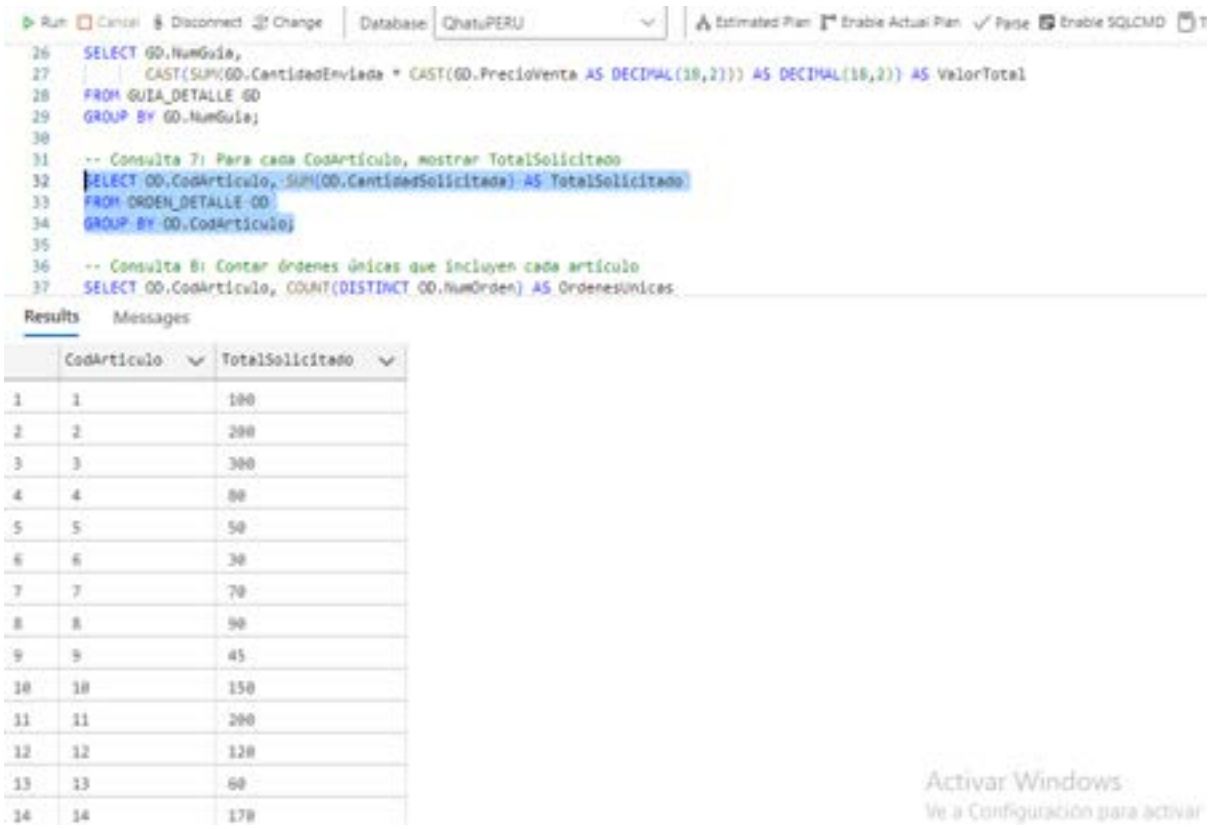
Explicación: Calcula el valor total de productos enviados por cada guía.

Consulta 7

Enunciado: Para cada CodArtículo, mostrar TotalSolicitado.

Consulta SQL:

```
SELECT
    OD.CodArticulo,
    SUM(OD.CantidadSolicitada) AS TotalSolicitado
FROM ORDEN_DETALLE OD
GROUP BY OD.CodArticulo;
```



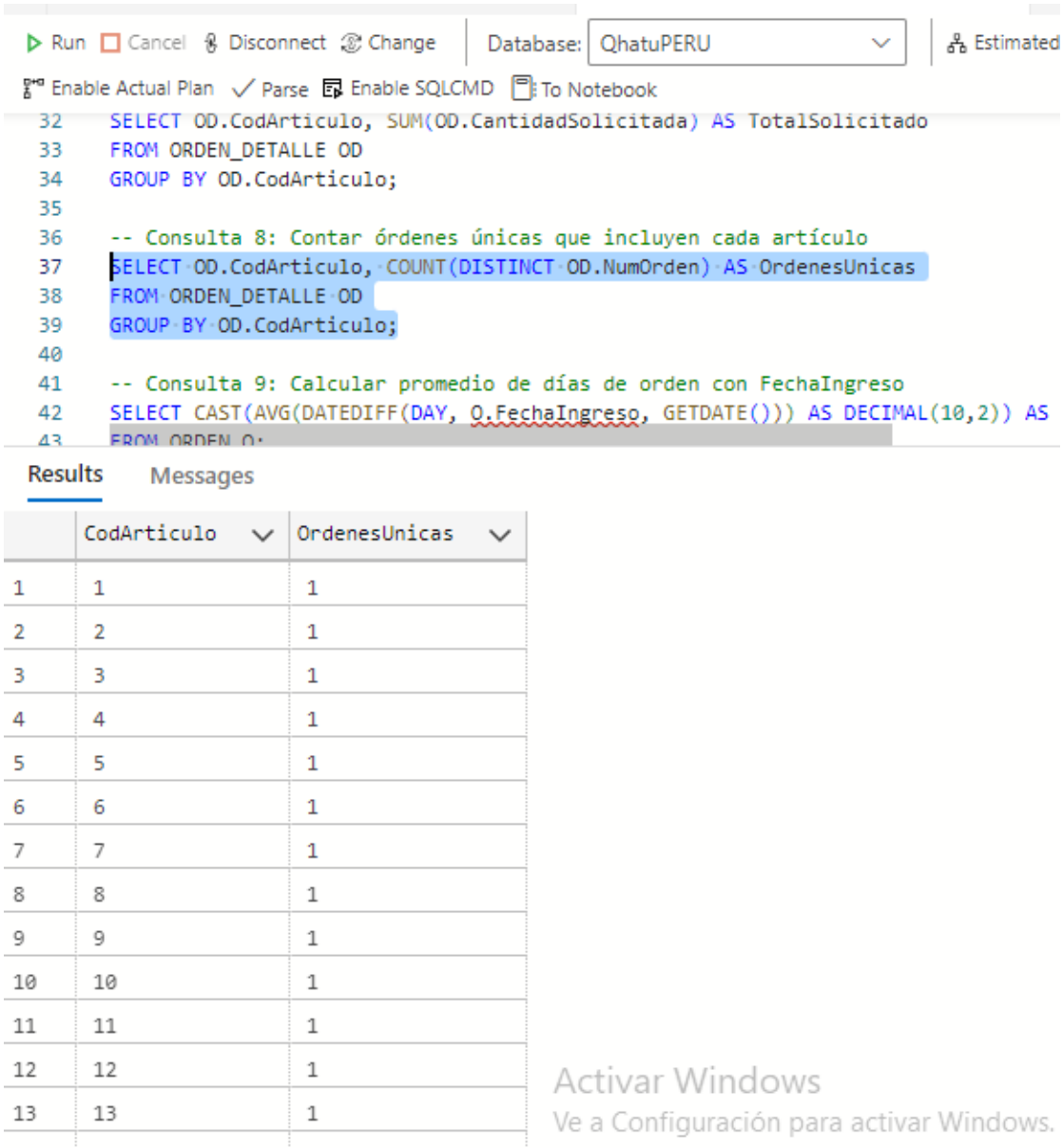
Explicación: Suma las cantidades solicitadas agrupadas por código de artículo.

Consulta 8

Enunciado: Contar órdenes únicas que incluyen cada artículo.

Consulta SQL:

```
SELECT
    OD.CodArticulo,
    COUNT(DISTINCT OD.NumOrden) AS OrdenesUnicas
FROM ORDEN_DETALLE OD
GROUP BY OD.CodArticulo;
```



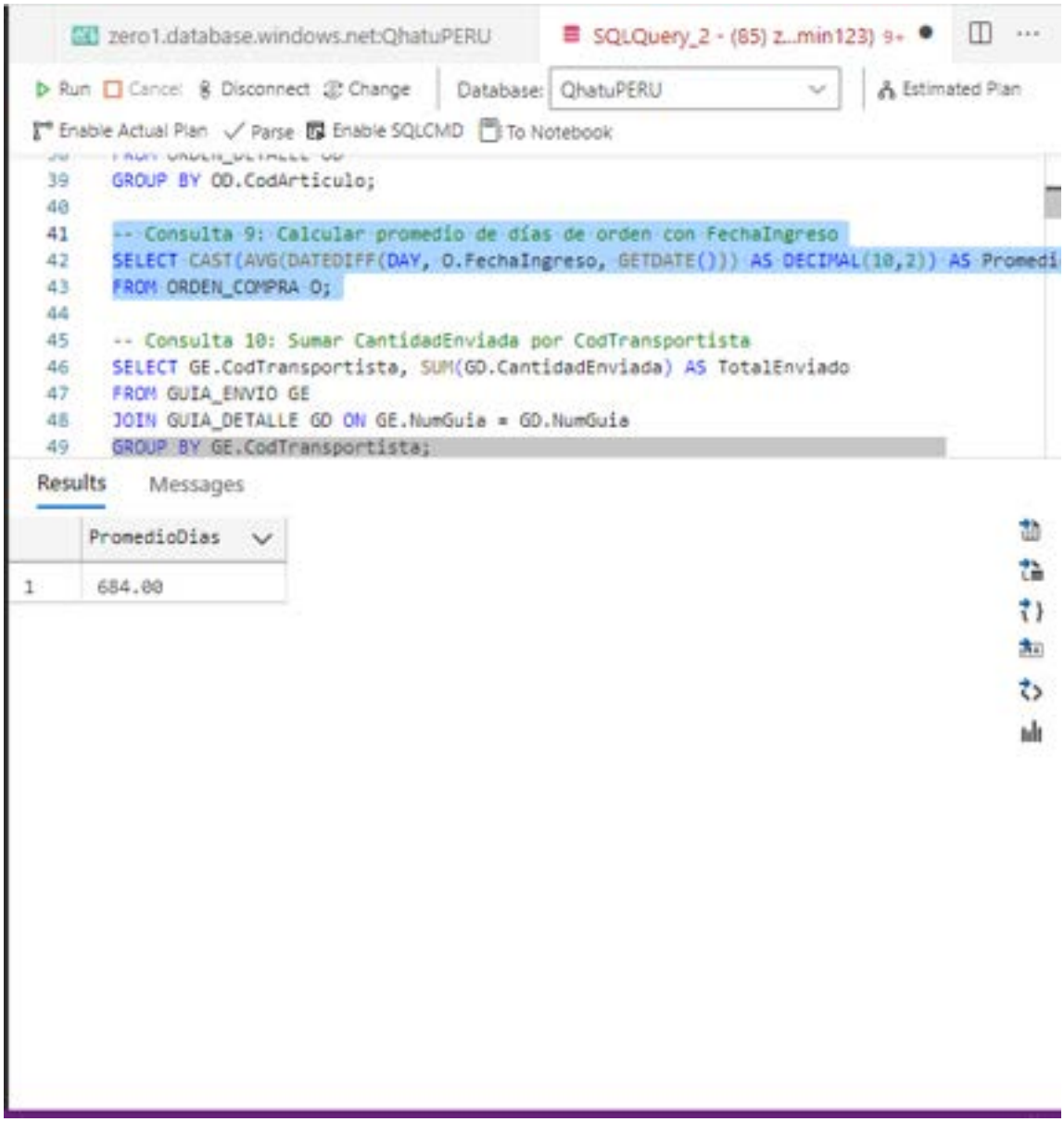
Explicación: Muestra cuántas órdenes distintas incluyen un mismo artículo.

Consulta 9

Enunciado: Calcular promedio de días de orden con FechaIngreso.

Consulta SQL:

```
SELECT
    CAST(AVG(DATEDIFF(DAY, O.FechaIngreso, GETDATE())) AS
DECIMAL(10,2)) AS PromedioDias
FROM ORDEN_COMPRA O;
```



Explicación: Calcula los días promedio transcurridos desde la fecha de ingreso de la orden (orden_detalle).

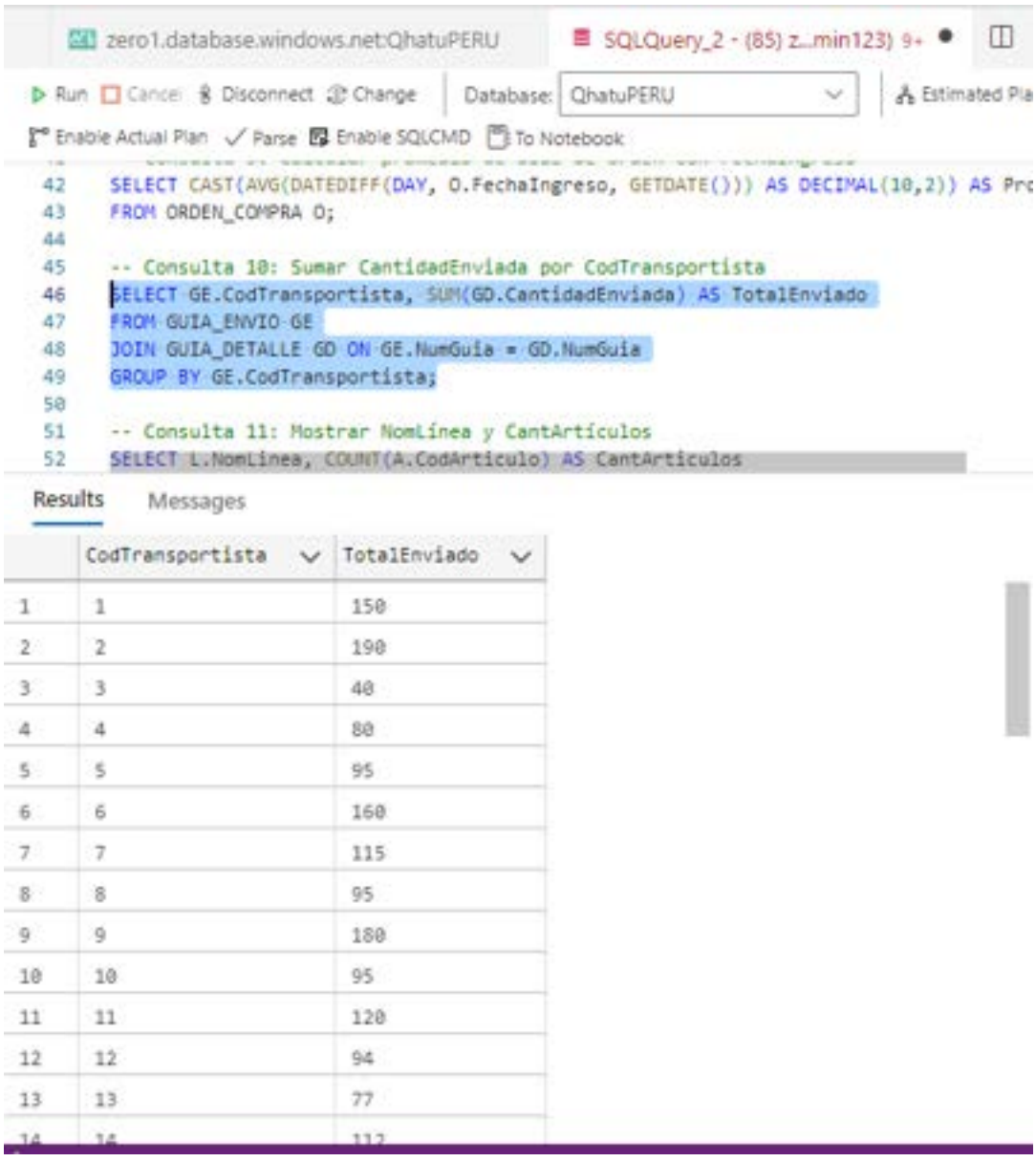
Consulta 10

Enunciado: Sumar CantidadEnviada por CodTransportista.

Consulta SQL:

```
SELECT
    GE.CodTransportista,
    SUM(GD.CantidadEnviada) AS TotalEnviado
```

```
FROM GUIA_ENVIO GE
JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
GROUP BY GE.CodTransportista;
```



Explicación: Suma las cantidades enviadas agrupadas por transportista responsable.

Consulta 11

Enunciado: Mostrar NomLínea y CantArtículos.

Consulta SQL:

```
SELECT
    L.NomLinea,
    COUNT(A.CodArticulo) AS CantArticulos
FROM LINEA L
JOIN ARTICULO A ON L.CodLinea = A.CodLinea
GROUP BY L.NomLinea;
```

```
50
51    -- Consulta 11: Mostrar NomLínea y CantArtículos
52    SELECT L.NomLinea, COUNT(A.CodArticulo) AS CantArticulos
53    FROM LINEA L
54    JOIN ARTICULO A ON L.CodLinea = A.CodLinea
55    GROUP BY L.NomLinea;
56
57    -- Consulta 12: Mostrar CodLínea y StockTotal
58    SELECT A.CodLinea, SUM(A.StockActual) AS StockTotal
```

Results Messages

	NomLinea	CantArticulos
1	Abarrotes	1
2	Aceites	1
3	Agua Mineral	1
4	Aguaymanto	1
5	Ají Amarillo	1
6	Ají Panca	1
7	Ajíes	1
8	Anticuchos	1
9	Arroz	1
10	Azúcar	1
11	Barras	1
12	Bebé	1
13	Bebidas	1

Explicación: Agrupa artículos por línea y muestra la cantidad en cada grupo.

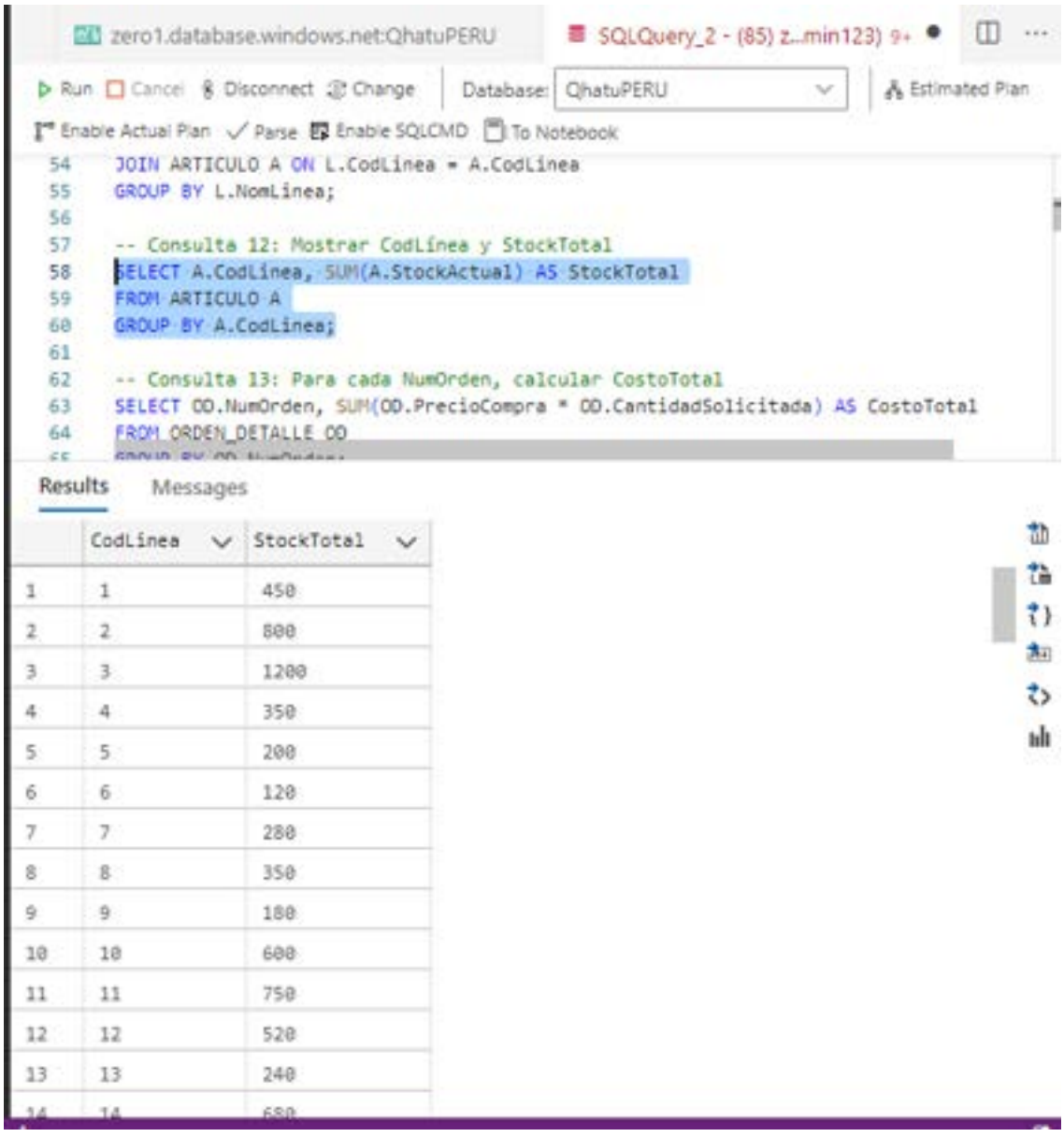
Consulta 12

Enunciado: Mostrar CodLínea y StockTotal.

Consulta SQL:

```
SELECT
    A.CodLinea,
```

```
    SUM(A.StockActual) AS StockTotal
FROM ARTICULO A
GROUP BY A.CodLinea;
```



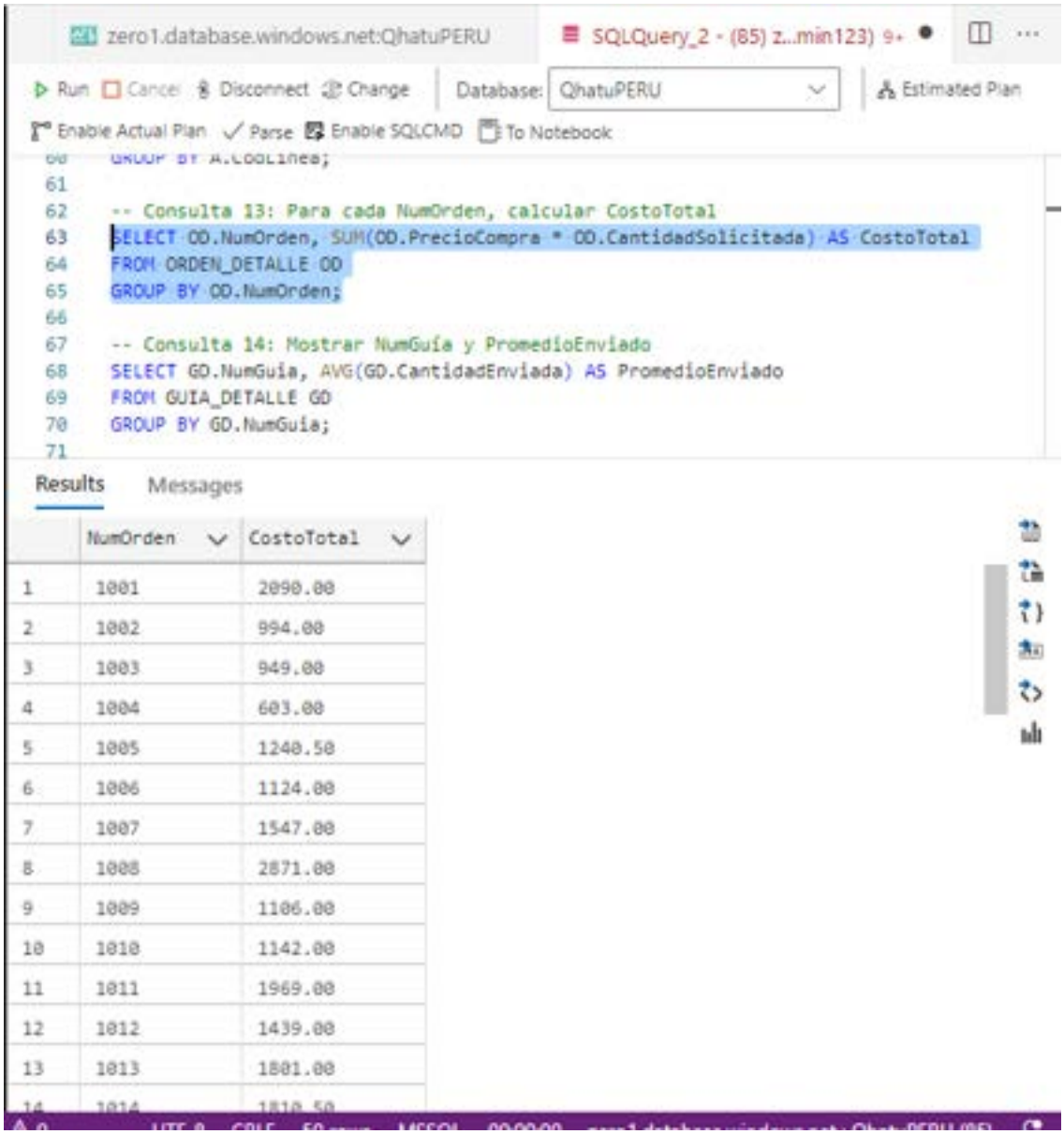
Explicación: Suma el stock actual agrupado por línea.

Consulta 13

Enunciado: Para cada NumOrden, calcular CostoTotal = SUM(PrecioCompra * Cantidad).

Consulta SQL:

```
SELECT
    OD.NumOrden,
    SUM(OD.PrecioCompra * OD.CantidadSolicitada) AS CostoTotal
FROM ORDEN_DETALLE OD
GROUP BY OD.NumOrden;
```



Explicación: Calcula el costo total de cada orden sumando el valor de sus productos.

Consulta 14

Enunciado: Mostrar NumGuía y PromedioEnviado.

Consulta SQL:

```
SELECT
    GD.NumGuia,
    AVG(GD.CantidadEnviada) AS PromedioEnviado
FROM GUIA_DETALLE GD
GROUP BY GD.NumGuia;
```

```
67  -- Consulta 14: Mostrar NumGuía y PromedioEnviado
68  SELECT GD.NumGuia, AVG(GD.CantidadEnviada) AS PromedioEnviado
69  FROM GUIA_DETALLE GD
70  GROUP BY GD.NumGuia;
71
72  -- Consulta 15: Mostrar CodProveedor y TotalArtículos
73  SELECT A.CodProveedor, COUNT(A.CodArticulo) AS TotalArticulos
74  FROM ARTICULO A
```

Results Messages

	NumGuia	PromedioEnviado
1	5001	75
2	5002	95
3	5003	20
4	5004	40
5	5005	47
6	5006	80
7	5007	57
8	5008	47
9	5009	90
10	5010	47
11	5011	60
12	5012	47
13	5013	38
14	5014	56

Explicación: Calcula el promedio de artículos enviados por cada guía.

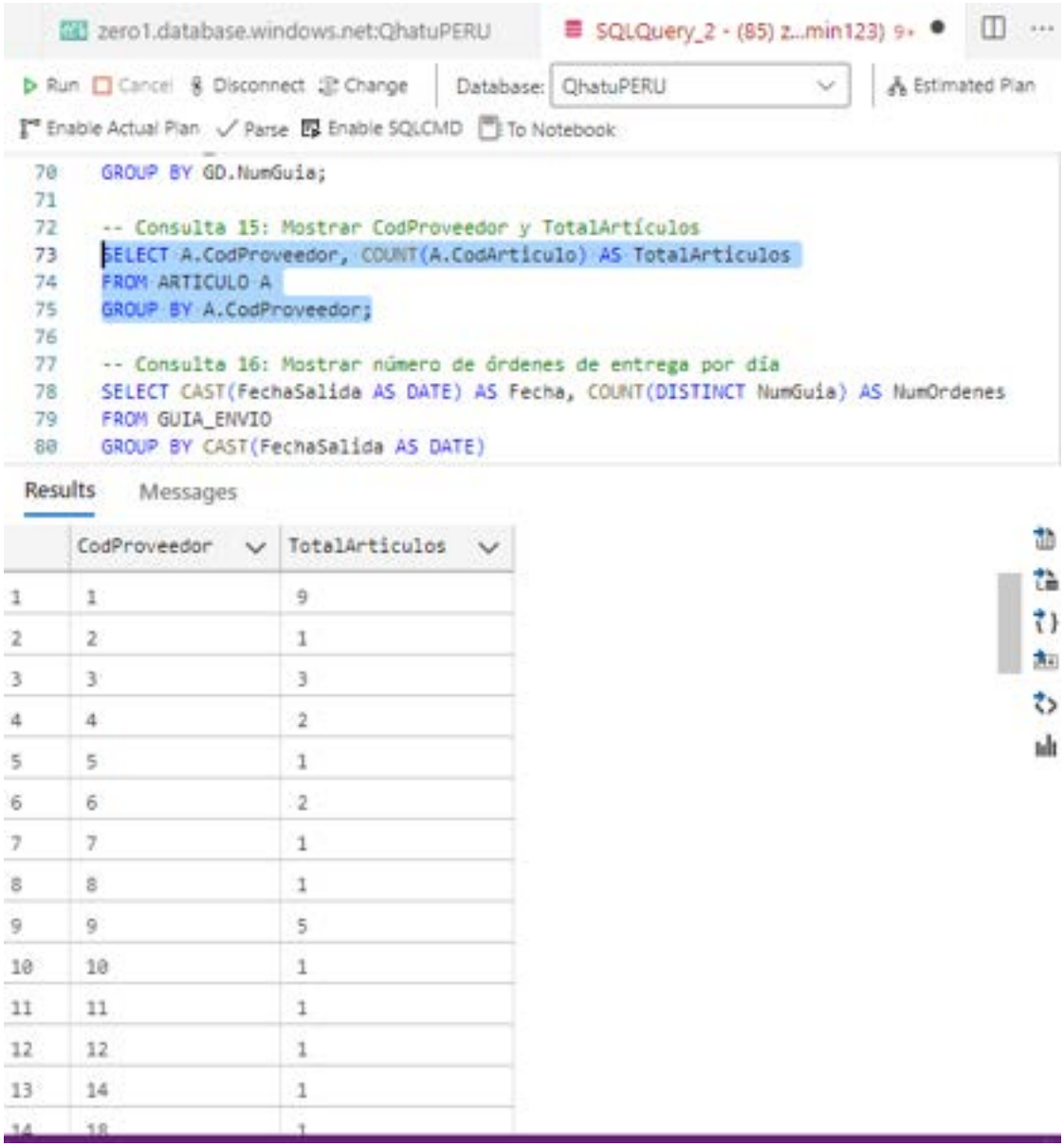
Consulta 15

Enunciado: Mostrar CodProveedor y TotalArtículos que ha suministrado.

Consulta SQL:

```
SELECT
    A.CodProveedor,
    COUNT(A.CodArticulo) AS TotalArticulos
```

```
FROM ARTICULO A
GROUP BY A.CodProveedor;
```



Explicación: Muestra la cantidad de artículos registrados por cada proveedor.

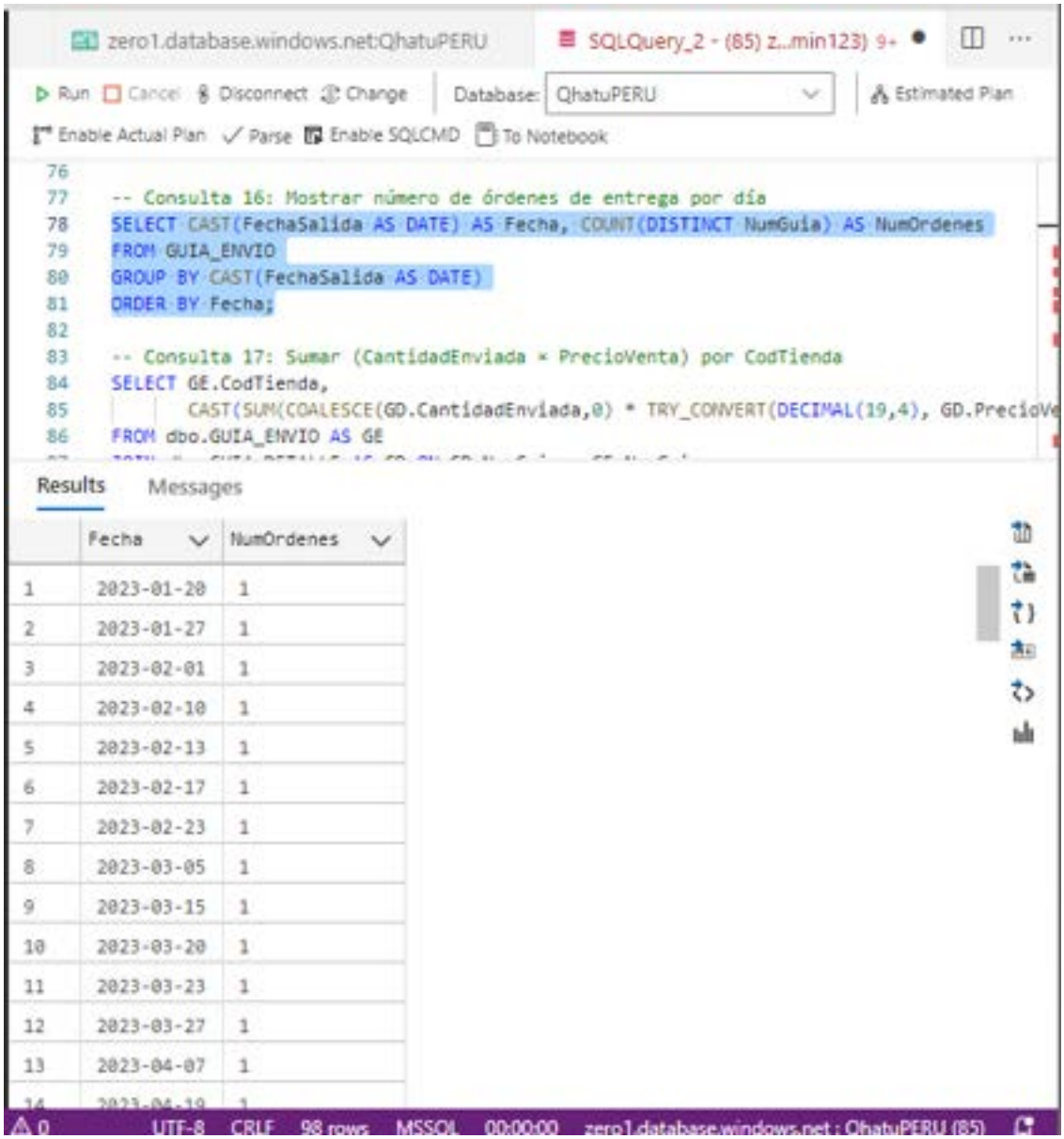
Consulta 16

Enunciado: Mostrar el número de órdenes de entrega por día (sin hora).

Consulta SQL:

```
SELECT
    CAST(FechaSalida AS DATE) AS Fecha,
    COUNT(DISTINCT NumGuia) AS NumOrdenes
FROM GUIA_ENVIO
```

```
GROUP BY CAST(FechaSalida AS DATE)
ORDER BY Fecha;
```



Explicación: Agrupa las guías por fecha (sin hora) y cuenta las órdenes emitidas en cada día.

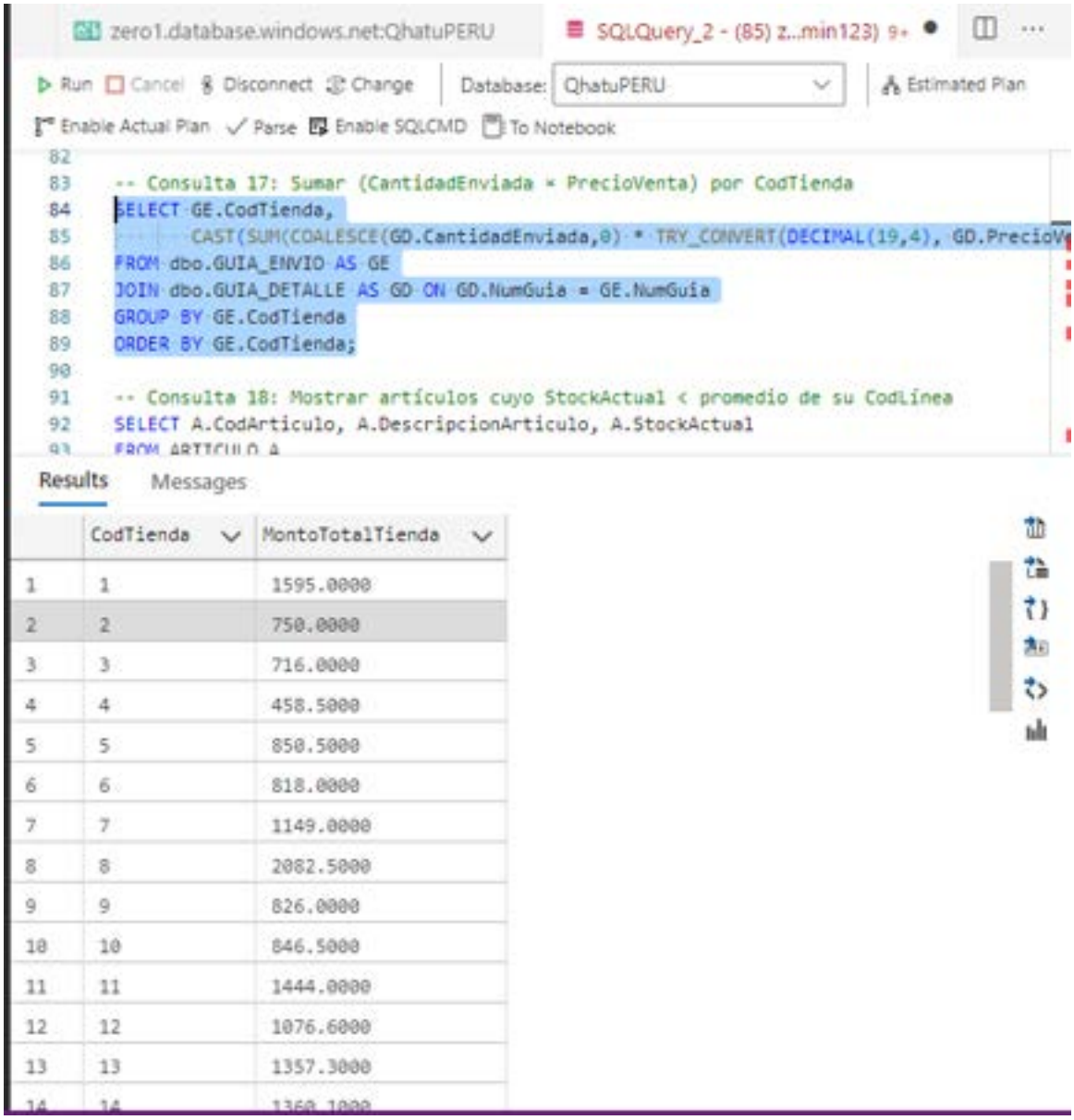
Consulta 17

Enunciado: Sumar (CantidadEnviada × PrecioVenta) por CodTienda.

Consulta SQL:

```
SELECT
    GE.CodTienda,
```

```
    CAST(SUM(COALESCE(GD.CantidadEnviada,0) *
TRY_CONVERT(DECIMAL(19,4), GD.PrecioVenta)) AS DECIMAL(19,4)) AS
MontoTotalTienda
FROM dbo.GUIA_ENVIO AS GE
JOIN dbo.GUIA_DETALLE AS GD ON GD.NumGuia = GE.NumGuia
GROUP BY GE.CodTienda
ORDER BY GE.CodTienda;
```



Explicación: Calcula el total de ventas por tienda; TRY_CONVERT garantiza precisión al convertir precios.

Consulta 18

Enunciado: Mostrar artículos cuyo StockActual < promedio de su CodLínea.

Consulta SQL:

```
SELECT
    A.CodArticulo,
    A.DescripcionArticulo,
    A.StockActual
FROM ARTICULO A
WHERE A.StockActual < (
    SELECT AVG(A2.StockActual)
    FROM ARTICULO A2
    WHERE A2.CodLinea = A.CodLinea
);
```

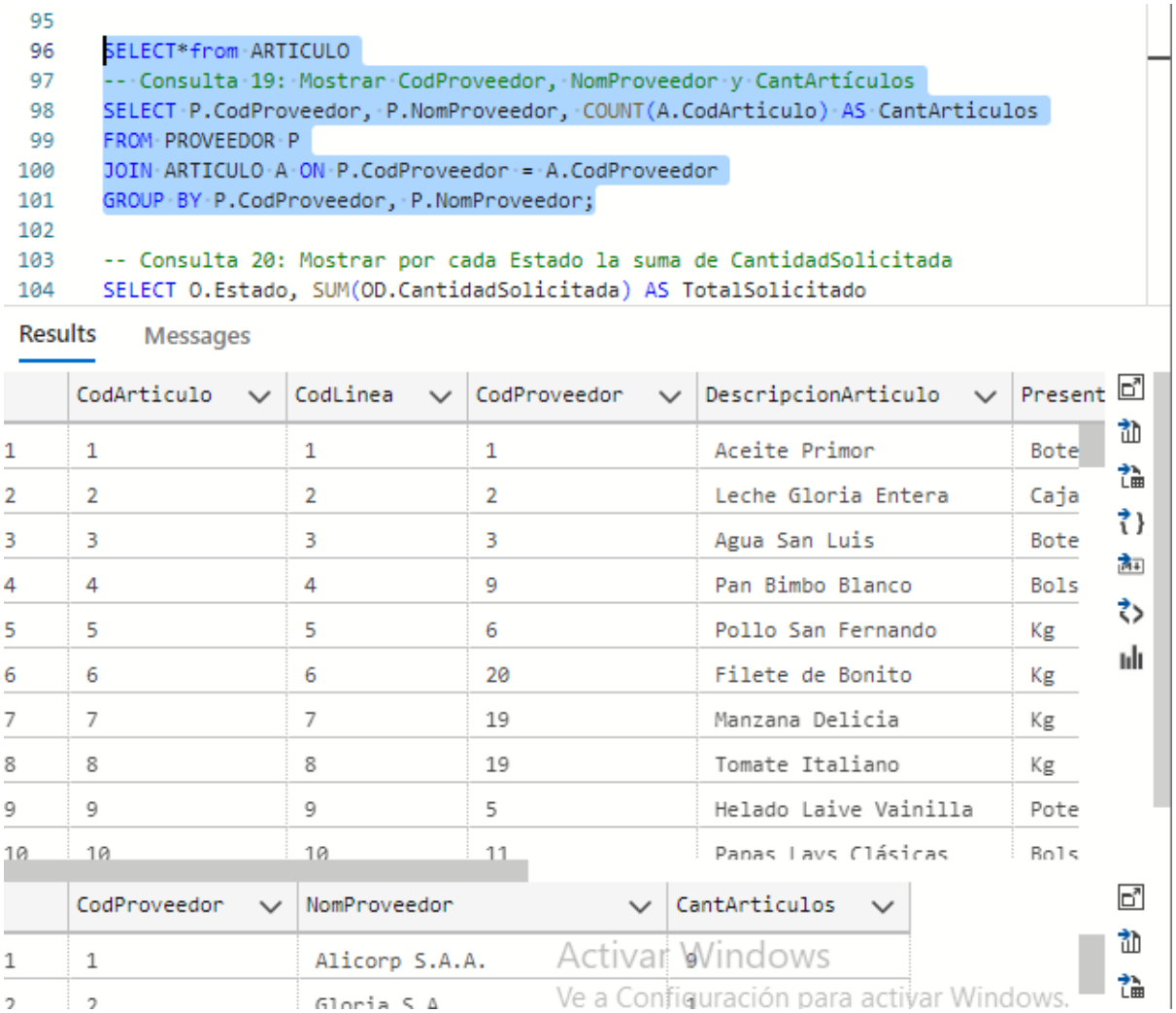
Explicación: Compara el stock actual del artículo con el promedio de su línea.

Consulta 19

Enunciado: Mostrar CodProveedor, NomProveedor y CantArtículos.

Consulta SQL:

```
SELECT
    P.CodProveedor,
    P.NomProveedor,
    COUNT(A.CodArticulo) AS CantArticulos
FROM PROVEEDOR P
JOIN ARTICULO A ON P.CodProveedor = A.CodProveedor
GROUP BY P.CodProveedor, P.NomProveedor;
```



```
95
96  SELECT*from ARTICULO
97  -- Consulta 19: Mostrar CodProveedor, NomProveedor y CantArtículos
98  SELECT P.CodProveedor, P.NomProveedor, COUNT(A.CodArticulo) AS CantArticulos
99  FROM PROVEEDOR P
100 JOIN ARTICULO A ON P.CodProveedor = A.CodProveedor
101 GROUP BY P.CodProveedor, P.NomProveedor;
102
103 -- Consulta 20: Mostrar por cada Estado la suma de CantidadSolicitada
104 SELECT O.Estado, SUM(OD.CantidadSolicitada) AS TotalSolicitado
```

Results Messages

	CodArticulo	CodLinea	CodProveedor	DescripcionArticulo	Present
1	1	1	1	Aceite Primor	Bote
2	2	2	2	Leche Gloria Entera	Caja
3	3	3	3	Agua San Luis	Bote
4	4	4	9	Pan Bimbo Blanco	Bols
5	5	5	6	Pollo San Fernando	Kg
6	6	6	20	Filete de Bonito	Kg
7	7	7	19	Manzana Delicia	Kg
8	8	8	19	Tomate Italiano	Kg
9	9	9	5	Helado Laive Vainilla	Pote
10	10	10	11	Papas Lays Clásicas	Bols

	CodProveedor	NomProveedor	CantArticulos
1	1	Alicorp S.A.A.	9
2	2	Gloria S.A.	1

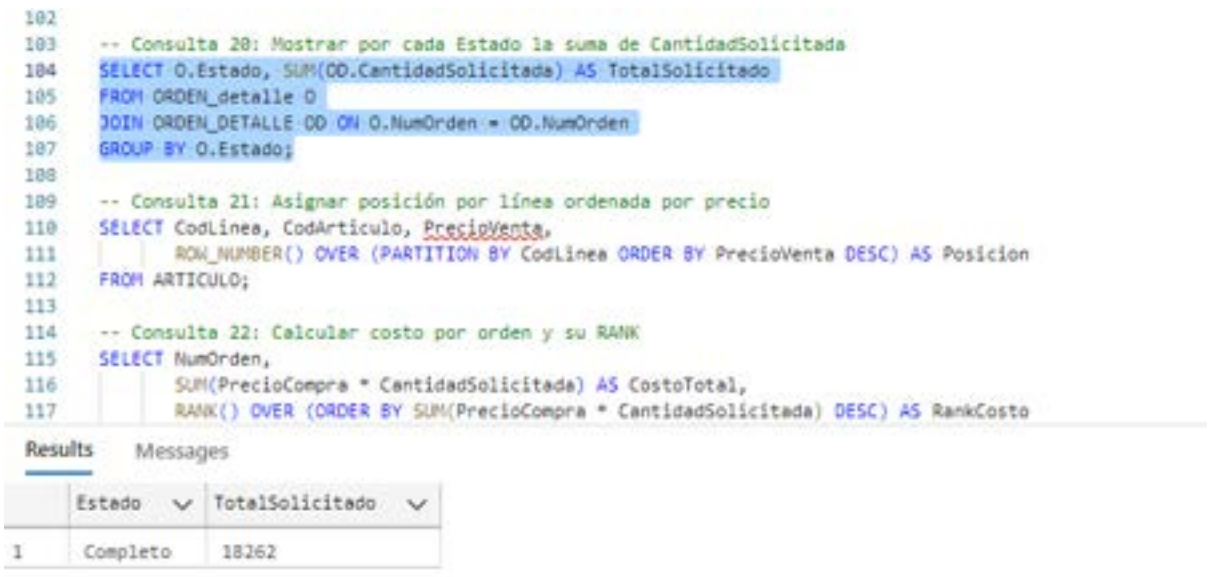
Explicación: Relaciona proveedores con los artículos que suministran y cuenta su cantidad.

Consulta 20

Enunciado: Mostrar por cada Estado la suma de CantidadSolicitada.

Consulta SQL:

```
SELECT
    O.Estado,
    SUM(OD.CantidadSolicitada) AS TotalSolicitado
FROM ORDEN _DETALLE O
JOIN ORDEN_DETALLE OD ON O.NumOrden = OD.NumOrden
GROUP BY O.Estado;
```



```
102
103   -- Consulta 20: Mostrar por cada Estado la suma de CantidadSolicitada
104   SELECT O.Estado, SUM(CO.CantidadSolicitada) AS TotalSolicitado
105   FROM ORDEN_detalle O
106   JOIN ORDEN_DETALLE CO ON O.NumOrden = CO.NumOrden
107   GROUP BY O.Estado;
108
109   -- Consulta 21: Asignar posición por línea ordenada por precio
110   SELECT CodLinea, CodArticulo, PrecioVenta,
111          ROW_NUMBER() OVER (PARTITION BY CodLinea ORDER BY PrecioVenta DESC) AS Posicion
112   FROM ARTICULO;
113
114   -- Consulta 22: Calcular costo por orden y su RANK
115   SELECT NumOrden,
116          SUM(PrecioCompra * CantidadSolicitada) AS CostoTotal,
117          RANK() OVER (ORDER BY SUM(PrecioCompra * CantidadSolicitada) DESC) AS RankCosto
```

Results Messages

	Estado	TotalSolicitado
1	Completo	18262

Explicación: Suma las cantidades solicitadas agrupadas por el estado de la orden.

CLÁUSULA OVER

Consulta 21

Enunciado: Asignar posición por línea ordenada por precio.

Consulta SQL:

```
SELECT
    CodLinea,
    CodArticulo,
    PrecioVenta,
    ROW_NUMBER() OVER (PARTITION BY CodLinea ORDER BY PrecioVenta
DESC) AS Posicion
FROM ARTICULO;
```

```sql
-- Consulta 21: Asignar posición por línea ordenada por precio
SELECT CodLinea, CodArticulo, PrecioProveedor,
       ROW_NUMBER() OVER (PARTITION BY CodLinea ORDER BY PrecioProveedor DESC) AS Posicion
FROM ARTICULO;

-- Consulta 22: Calcular costo por orden y su RANK
SELECT NumOrden,
       SUM(PrecioCompra * CantidadSolicitada) AS CostoTotal,
       RANK() OVER (ORDER BY SUM(PrecioCompra * CantidadSolicitada) DESC) AS RankCosto
FROM ORDEN_DETALLE
GROUP BY NumOrden;
```

Results Messages

	CodLinea	CodArticulo	PrecioProveedor	Posicion
1	1	1	12.50	1
2	2	2	4.20	1
3	3	3	1.50	1
4	4	4	6.80	1
5	5	5	9.50	1
6	6	6	15.80	1
7	7	7	4.50	1
8	8	8	3.20	1

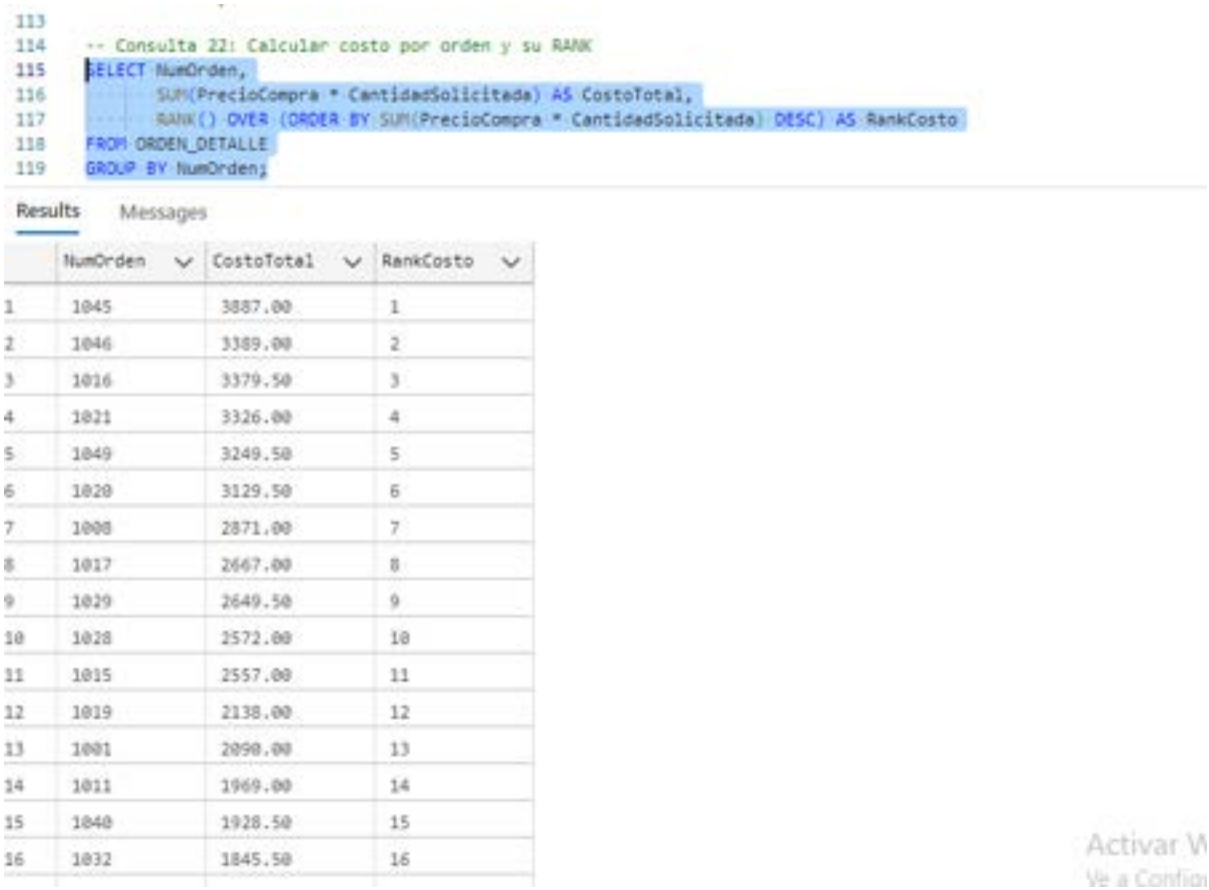
Explicación: Asigna un número de orden a los artículos según su precio dentro de cada línea.

Consulta 22

Enunciado: Calcular costo por orden y su RANK (RankCosto).

Consulta SQL:

```sql
SELECT
    NumOrden,
    SUM(PrecioCompra * CantidadSolicitada) AS CostoTotal,
    RANK() OVER (ORDER BY SUM(PrecioCompra * CantidadSolicitada)
DESC) AS RankCosto
FROM ORDEN_DETALLE
GROUP BY NumOrden;
```



```
113
114  -- Consulta 22: Calcular costo por orden y su RANK
115  SELECT NumOrden,
116         SUM(PrecioCompra * CantidadSolicitada) AS CostoTotal,
117         RANK() OVER (ORDER BY SUM(PrecioCompra * CantidadSolicitada) DESC) AS RankCosto
118  FROM ORDEN_DETALLE
119  GROUP BY NumOrden;
```

Results Messages

	NumOrden	CostoTotal	RankCosto
1	1045	3887.00	1
2	1046	3389.00	2
3	1016	3379.50	3
4	1021	3326.00	4
5	1049	3249.50	5
6	1020	3129.50	6
7	1008	2871.00	7
8	1017	2667.00	8
9	1029	2649.50	9
10	1028	2572.00	10
11	1015	2557.00	11
12	1019	2138.00	12
13	1001	2090.00	13
14	1011	1969.00	14
15	1040	1928.50	15
16	1032	1845.50	16

Explicación: Calcula el costo total de cada orden y le asigna un rango según su valor.

Consulta 23

Enunciado: Mostrar TotalDía y AcumuladoVentas ordenado por fecha.

Consulta SQL:

```
SELECT
    CAST(FechaSalida AS DATE) AS Fecha,
    SUM(GD.CantidadEnviada * GD.PrecioVenta) AS TotalDia,
    SUM(SUM(GD.CantidadEnviada * GD.PrecioVenta)) OVER (ORDER BY
CAST(FechaSalida AS DATE)) AS Acumulado
FROM GUIA_ENVIO GE
JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
GROUP BY CAST(FechaSalida AS DATE);
```

```
-- Consulta 23: Mostrar TotalDia y AcumuladoVentas ordenado por fecha
SELECT CAST(FechaSalida AS DATE) AS Fecha,
       SUM(GD.CantidadEnviada * GD.PrecioVenta) AS TotalDia,
       SUM(SUM(GD.CantidadEnviada * GD.PrecioVenta)) OVER (ORDER BY CAST(FechaSalida AS DATE)) AS Acumulado
FROM GUIA_ENVIO GE
JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
GROUP BY CAST(FechaSalida AS DATE);

-- Consulta 24: Calcular promedio móvil para stock
SELECT CodArticulo, FechaMovimiento,
```

Results Messages

	Fecha	TotalDia	Acumulado
1	2023-01-20	1595.00	1595.00
2	2023-01-27	750.00	2345.00
3	2023-02-10	716.00	3061.00
4	2023-02-17	458.50	3519.50
5	2023-03-05	850.50	4370.00
6	2023-03-15	818.00	5188.00
7	2023-03-23	1149.00	6337.00
8	2023-04-07	2082.50	8419.50
9	2023-04-19	826.00	9245.50
10	2023-04-30	846.50	10092.00
11	2023-05-13	1444.00	11536.00
12	2023-05-24	1076.60	12612.60

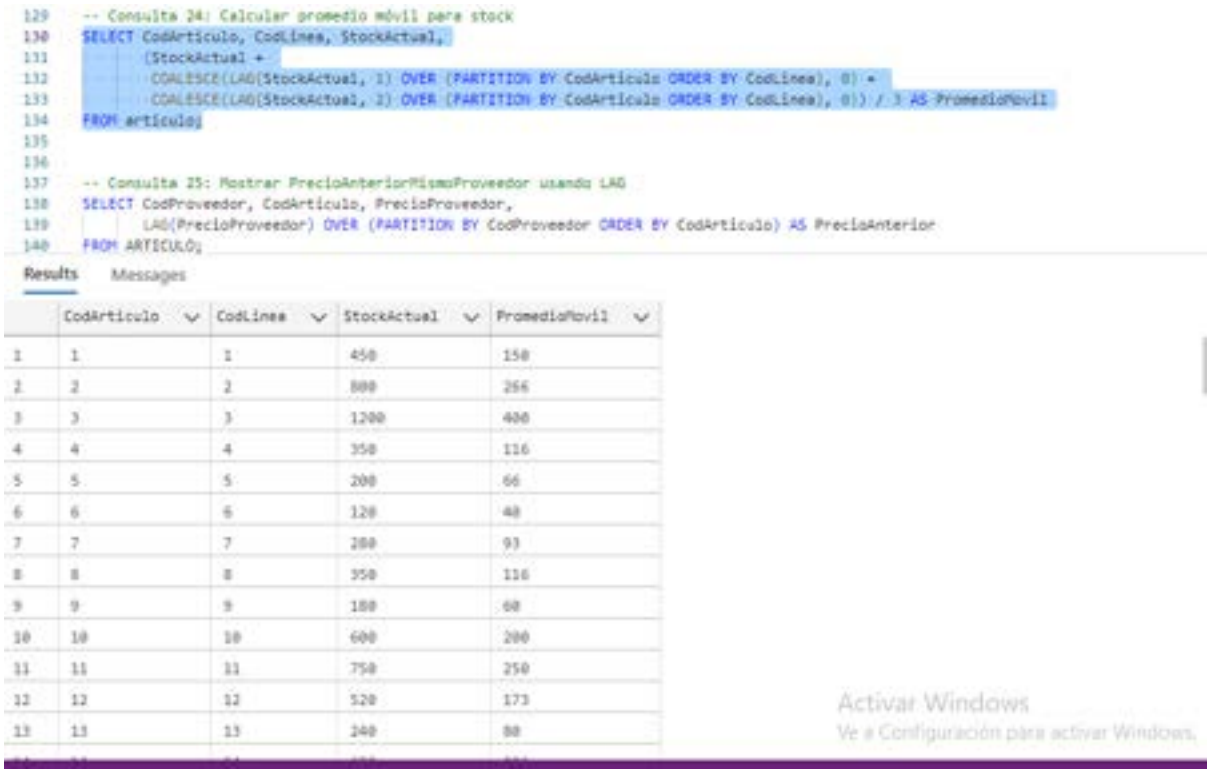
Explicación: Calcula el total diario y su acumulado progresivo en el tiempo.

Consulta 24

Enunciado: Calcular promedio móvil para stock.

Consulta SQL:

```
SELECT CodArticulo, CodLinea, StockActual,
    (StockActual +
     COALESCE(LAG(StockActual, 1) OVER (PARTITION BY CodArticulo ORDER BY CodLinea), 0) +
     COALESCE(LAG(StockActual, 2) OVER (PARTITION BY CodArticulo ORDER BY CodLinea), 0)) / 3 AS PromedioMovil
FROM articulo;
```



```
-- Consulta 24: Calcular promedio móvil para stock
SELECT CodArticulo, CodLinea, StockActual,
       (StockActual +
        COALESCE(LAG(StockActual, 1) OVER (PARTITION BY CodArticulo ORDER BY CodLinea), 0) +
        COALESCE(LAG(StockActual, 2) OVER (PARTITION BY CodArticulo ORDER BY CodLinea), 0)) / 3 AS PromedioMovil
FROM articulo;


-- Consulta 25: Mostrar PrecioAnteriorMismoProveedor usando LAG
SELECT CodProveedor, CodArticulo, PrecioProveedor,
       LAG(PrecioProveedor) OVER (PARTITION BY CodProveedor ORDER BY CodArticulo) AS PrecioAnterior
FROM ARTICULO;
```

Results | Messages

	CodArticulo	CodLinea	StockActual	PromedioMovil
1	1	1	450	150
2	2	2	800	266
3	3	3	1200	400
4	4	4	350	116
5	5	5	200	66
6	6	6	120	40
7	7	7	280	93
8	8	8	350	116
9	9	9	180	60
10	10	10	600	200
11	11	11	750	250
12	12	12	520	173
13	13	13	240	80

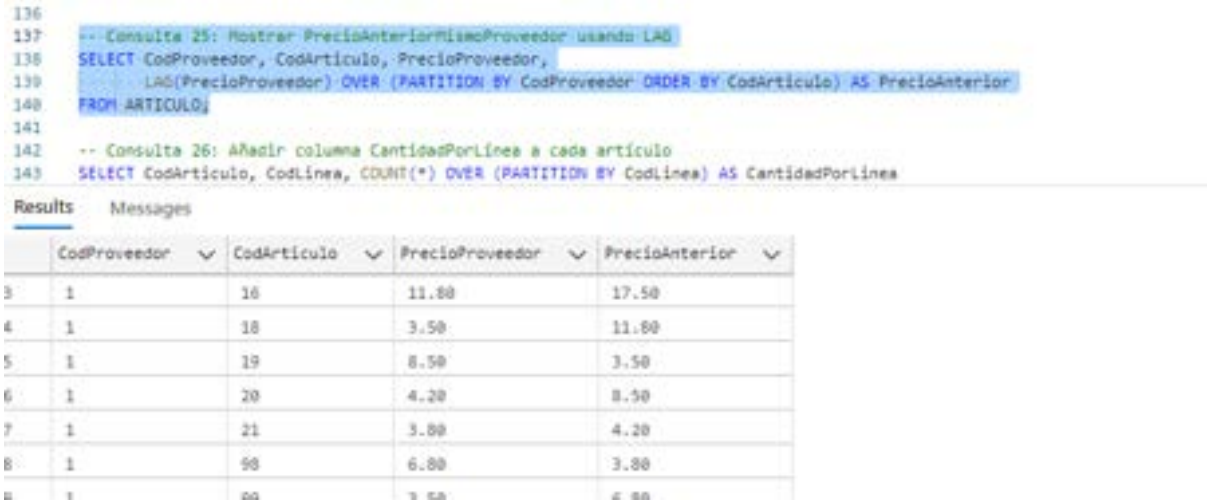
Explicación: Calcula un promedio móvil de stock para los últimos registros de cada artículo.

Consulta 25

Enunciado: Mostrar PrecioAnteriorMismoProveedor usando LAG.

Consulta SQL:

```
SELECT
    CodProveedor,
    CodArticulo,
    PrecioProveedor,
    LAG(PrecioProveedor) OVER (PARTITION BY CodProveedor ORDER BY
CodArticulo) AS PrecioAnterior
FROM ARTICULO;
```



```
136
137   -- Consulta 25: Mostrar PrecioAnteriorMismoProveedor usando LAG
138   SELECT CodProveedor, CodArticulo, PrecioProveedor,
139          LAG(PrecioProveedor) OVER (PARTITION BY CodProveedor ORDER BY CodArticulo) AS PrecioAnterior
140   FROM ARTICULO;
141
142   -- Consulta 26: Añadir columna CantidadPorLínea a cada artículo
143   SELECT CodArticulo, CodLinea, COUNT(*) OVER (PARTITION BY CodLinea) AS CantidadPorLinea
```

Results Messages

	CodProveedor	CodArticulo	PrecioProveedor	PrecioAnterior
3	1	16	11.80	17.50
4	1	18	3.50	11.80
5	1	19	8.50	3.50
6	1	20	4.20	8.50
7	1	21	3.80	4.20
8	1	90	6.80	3.80

Explicación: Obtiene el precio anterior del mismo proveedor para comparar variaciones.

Consulta 26

Enunciado: Añadir columna CantidadPorLínea a cada artículo.

Consulta SQL:

```
SELECT
    CodArticulo,
    CodLinea,
    COUNT(*) OVER (PARTITION BY CodLinea) AS CantidadPorLinea
```

FROM ARTICULO;

```
141
142   -- Consulta 26: Añadir columna CantidadPorLínea a cada artículo
143   SELECT CodArticulo, CodLinea, COUNT(*) OVER (PARTITION BY CodLinea) AS CantidadPorLinea
144   FROM ARTICULO;
145
146   -- Consulta 27: Mostrar MontoProveedor y PorcentajeDelTotal
147   SELECT CodProveedor,
148          SUM(PrecioProveedor * StockActual) AS MontoProveedor,
149          CAST(SUM(PrecioProveedor * StockActual) * 100.0 / SUM(SUM(PrecioProveedor * StockActual)) OVER(
```

Results Messages

	CodArticulo	CodLinea	CantidadPorLinea
1	1	1	1
2	2	2	1
3	3	3	1
4	4	4	1
5	5	5	1
6	6	6	1
7	7	7	1
8	8	8	1
9	9	9	1
10	10	10	1
11	11	11	1
12	12	12	1
13	13	13	1

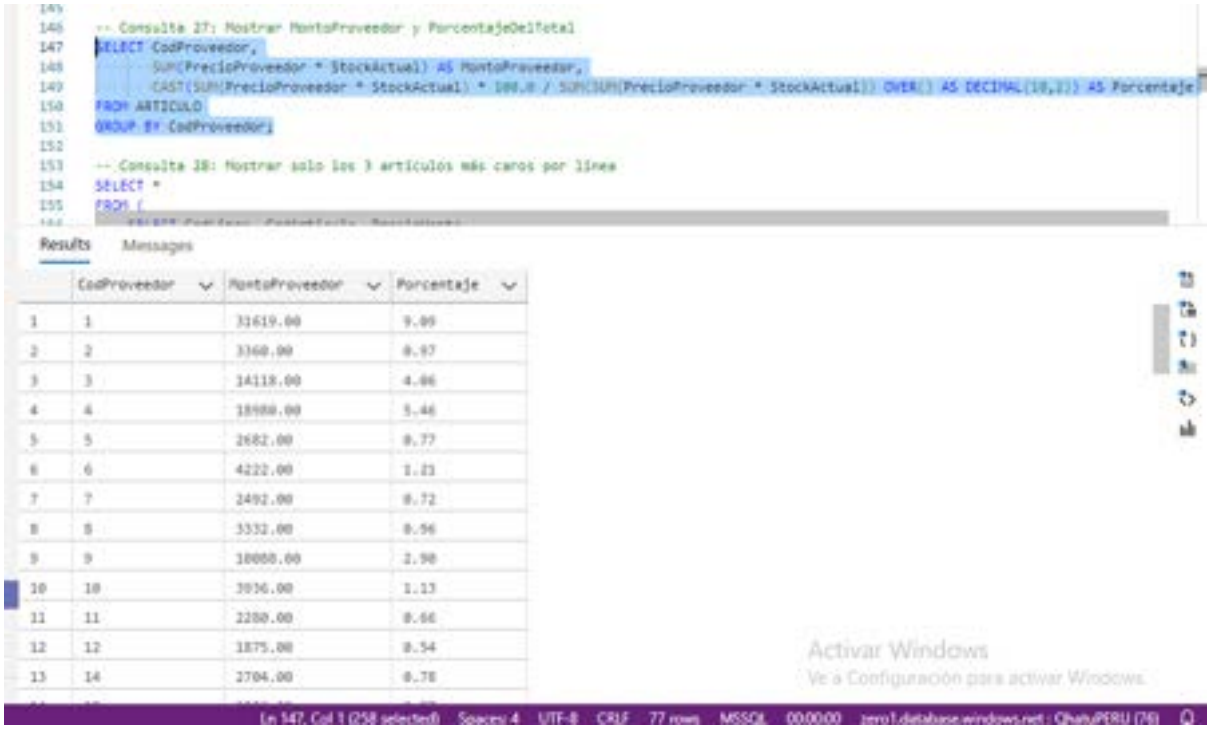
Explicación: Usa OVER para calcular el número total de artículos por línea sin agrupar.

Consulta 27

Enunciado: Mostrar MontoProveedor y PorcentajeDelTotal.

Consulta SQL:

```
SELECT
    CodProveedor,
    SUM(PrecioProveedor * StockActual) AS MontoProveedor,
    CAST(SUM(PrecioProveedor * StockActual) * 100.0 /
SUM(SUM(PrecioProveedor * StockActual)) OVER() AS DECIMAL(10,2)) AS
Porcentaje
FROM ARTICULO
GROUP BY CodProveedor;
```



Explicación: Calcula el valor total aportado por cada proveedor y su porcentaje global.

Consulta 28

Enunciado: Mostrar solo los 3 artículos más caros por línea.

Consulta SQL:

```
SELECT  *
FROM (
  SELECT
    CodLinea, CodArticulo, PrecioProveedor
,
    ROW_NUMBER() OVER (PARTITION BY CodLinea ORDER BY PrecioProveedor
DESC) AS Pos
  FROM ARTICULO
) X
WHERE X.Pos <= 3;
```

```sql
-- Consulta 28: Mostrar solo los 3 artículos más caros por línea
SELECT *
FROM (
    SELECT CodLinea, CodArticulo, PrecioProveedor,
           ROW_NUMBER() OVER (PARTITION BY CodLinea ORDER BY PrecioProveedor DESC) AS Pos
    FROM ARTICULO
) X
WHERE X.Pos <= 3;

-- Consulta 29: Mostrar transportista y su DenseRank por TotalEnviado
SELECT CodTransportista, SUM(GD.CantidadEnviada) AS TotalEnviado,
       DENSE_RANK() OVER (ORDER BY SUM(GD.CantidadEnviada) DESC) AS Posicion
FROM GUIA_ENVIO GE
```

Results Messages

	CodLinea	CodArticulo	PrecioProveedor	Pos
1	1	1	12.50	1
2	2	2	4.20	1
3	3	3	1.50	1
4	4	4	6.80	1
5	5	5	9.50	1
6	6	6	15.80	1
7	7	7	4.50	1
8	8	8	3.20	1
9	9	9	14.90	1
10	10	10	3.80	1
11	11	11	2.50	1
12	12	12	5.20	1
13	13	13	11.90	1

Explicación: Selecciona los tres artículos con mayor precio en cada línea.

Consulta 29

Enunciado: Mostrar transportista y su DenseRank por TotalEnviado.

Consulta SQL:

```
SELECT
    CodTransportista,
    SUM(GD.CantidadEnviada) AS TotalEnviado,
    DENSE_RANK() OVER (ORDER BY SUM(GD.CantidadEnviada) DESC) AS
Posicion
FROM GUIA_ENVIO GE
JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
GROUP BY CodTransportista;
```

```sql
-- Consulta 29: Mostrar transportista y su DenseRank por TotalEnviado
SELECT CodTransportista, SUM(GD.CantidadEnviada) AS TotalEnviado,
    DENSE_RANK() OVER (ORDER BY SUM(GD.CantidadEnviada) DESC) AS Posicion
FROM GUIA_ENVIO GE
JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
GROUP BY CodTransportista;

-- Consulta 30: Mostrar por guía la suma acumulada por tienda hasta esa guía
SELECT CodTienda, NumGuia,
```

Results Messages

	CodTransportista	TotalEnviado	Posicion
1	2	190	1
2	22	182	2
3	9	180	3
4	46	167	4
5	6	160	5
6	1	150	6
7	21	145	7
8	34	137	8
9	30	132	9
10	33	125	10
11	11	120	11
12	16	117	12

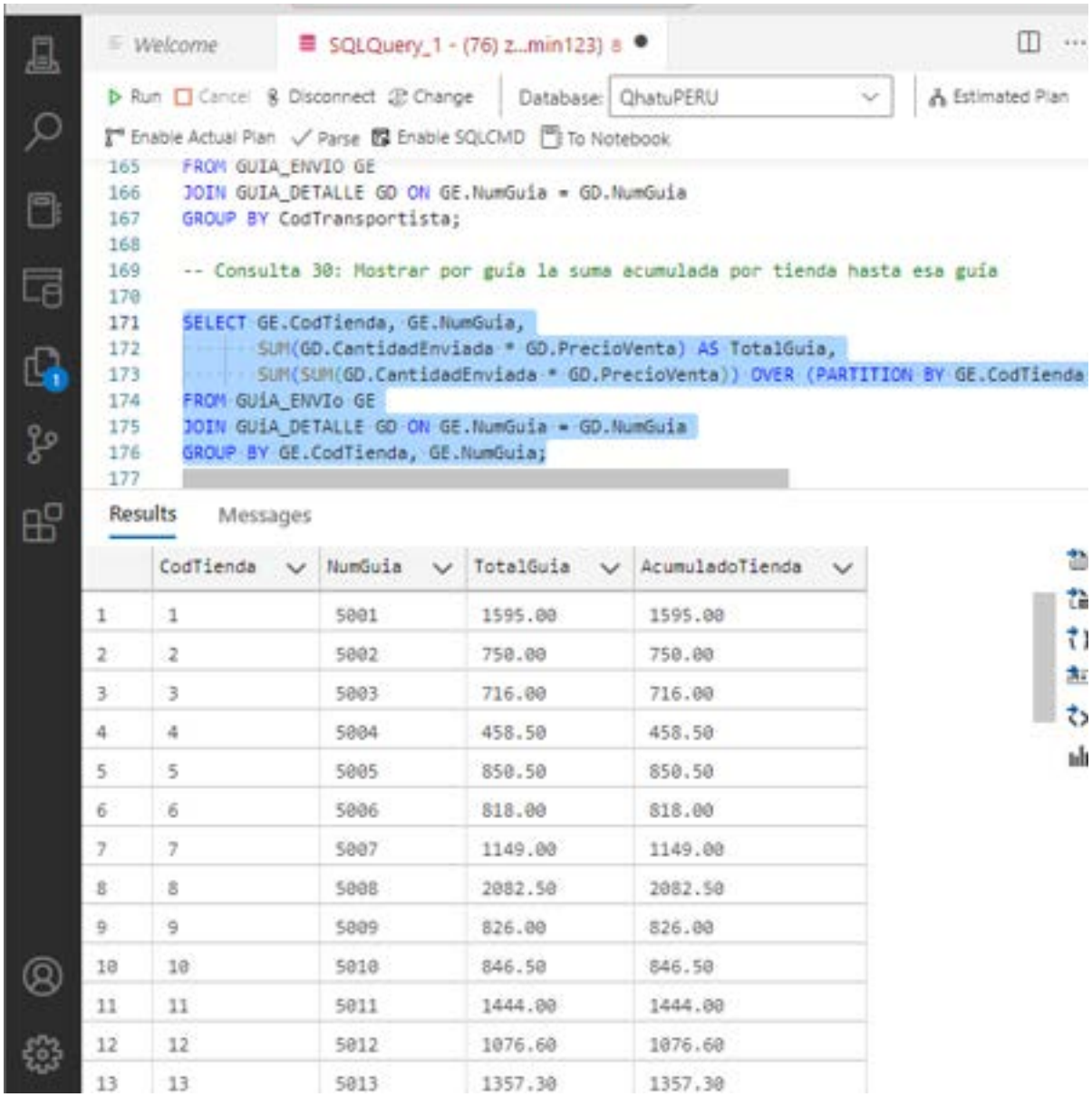
Explicación: Ordena transportistas según volumen enviado sin saltar posiciones repetidas.

Consulta 30

Enunciado: Mostrar por guía la suma acumulada por tienda hasta esa guía.

Consulta SQL:

```sql
SELECT GE.CodTienda, GE.NumGuia,
    SUM(GD.CantidadEnviada * GD.PrecioVenta) AS TotalGuia,
    SUM(SUM(GD.CantidadEnviada * GD.PrecioVenta)) OVER (PARTITION BY
GE.CodTienda ORDER BY GE.NumGuia) AS AcumuladoTienda
FROM GUiA_ENVIo GE
JOIN GUiA_DETALLE GD ON GE.NumGuia = GD.NumGuia
GROUP BY GE.CodTienda, GE.NumGuia;
```



Explicación: Calcula acumulado de envíos por tienda en orden de guías.

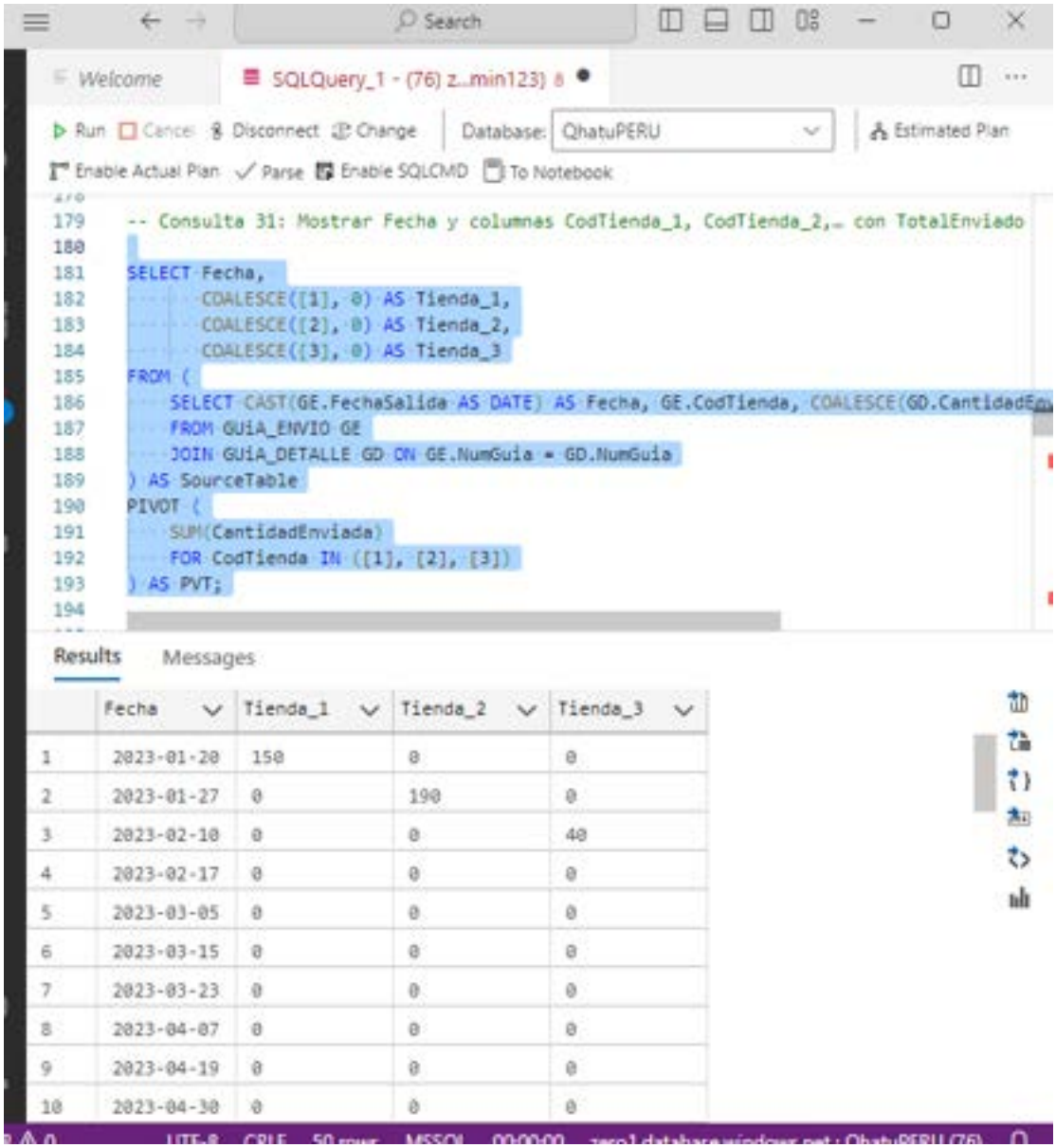
OPERADOR PIVOT

Consulta 31

Enunciado: Mostrar Fecha y columnas CodTienda_1, CodTienda_2, ... con TotalEnviado por día.

Consulta SQL:

```
SELECT Fecha,
     COALESCE([1], 0) AS Tienda_1,
     COALESCE([2], 0) AS Tienda_2,
     COALESCE([3], 0) AS Tienda_3
FROM (
   SELECT CAST(GE.FechaSalida AS DATE) AS Fecha, GE.CodTienda,
COALESCE(GD.CantidadEnviada, 0) AS CantidadEnviada
   FROM GUiA_ENVIO GE
   JOIN GUiA_DETALLE GD ON GE.NumGuia = GD.NumGuia
) AS SourceTable
PIVOT (
   SUM(CantidadEnviada)
   FOR CodTienda IN ([1], [2], [3])
) AS PVT;
```



Explicación: Convierte los valores de tienda en columnas para cada fecha.

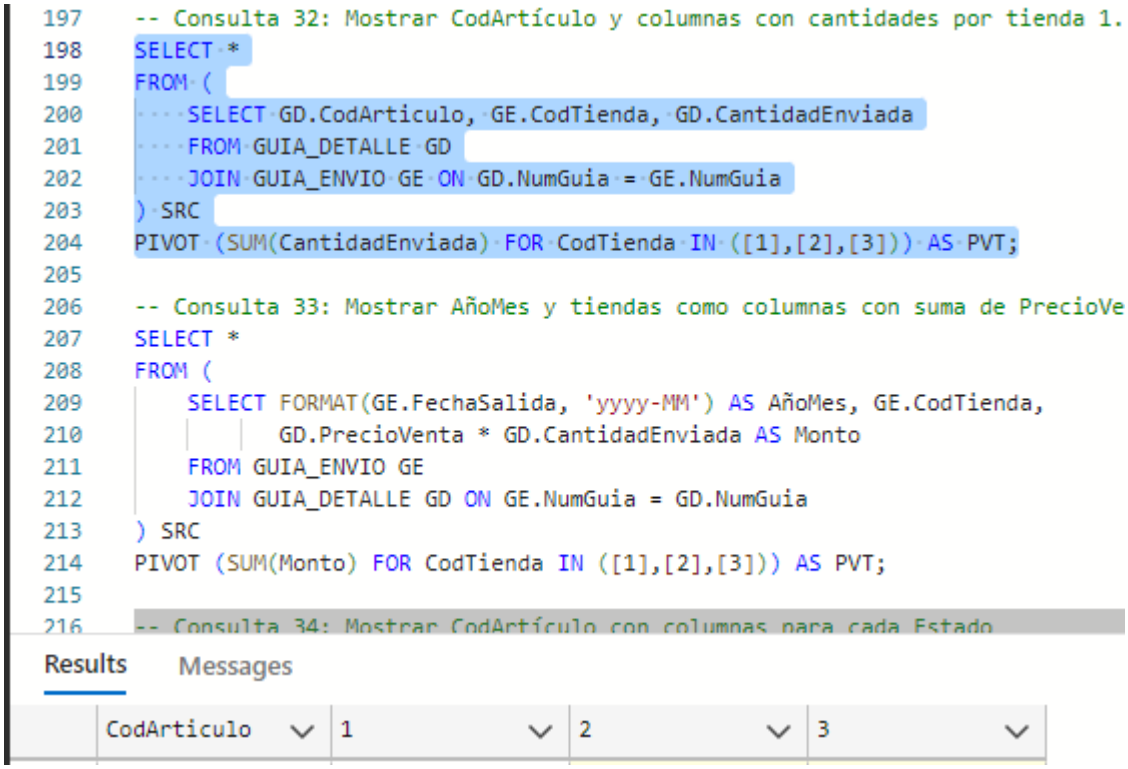
Consulta 32

Enunciado: Mostrar CodArtículo y columnas con cantidades por tienda 1..3.

Consulta SQL:

```
SELECT *
FROM (
    SELECT GD.CodArticulo, GE.CodTienda, GD.CantidadEnviada
    FROM GUIA_DETALLE GD
```

```
    JOIN GUIA_ENVIO GE ON GD.NumGuia = GE.NumGuia
) SRC
PIVOT (
    SUM(CantidadEnviada) FOR CodTienda IN ([1],[2],[3])
) AS PVT;
```



Explicación: Genera columnas por tienda mostrando cuántos artículos fueron enviados.

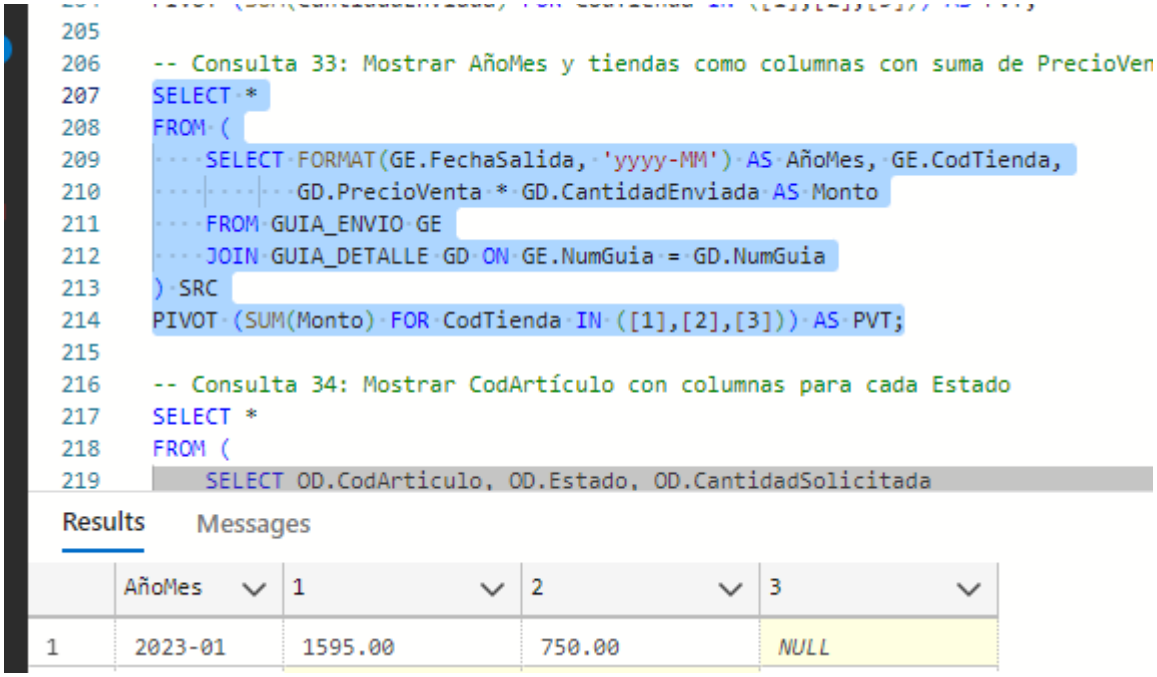
Consulta 33

Enunciado: Mostrar AñoMes y tiendas como columnas con suma de PrecioVenta * Cantidad.

Consulta SQL:

```
SELECT *
FROM (
    SELECT
        FORMAT(GE.FechaSalida, 'yyyy-MM') AS AñoMes,
        GE.CodTienda,
        GD.PrecioVenta * GD.CantidadEnviada AS Monto
    FROM GUIA_ENVIO GE
    JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
```

```
) SRC
PIVOT (
    SUM(Monto) FOR CodTienda IN ([1],[2],[3])
) AS PVT;
```



Explicación: Muestra el monto total vendido por mes y tienda en columnas pivotadas.

Consulta 34

Enunciado: Mostrar CodArtículo con columnas para cada Estado.

Consulta SQL:

```
SELECT *
FROM (
    SELECT
        OD.CodArticulo,
        O.Estado,
        OD.CantidadSolicitada
    FROM ORDEN_DETALLE OD
    JOIN ORDEN O ON OD.NumOrden = O.NumOrden
) SRC
PIVOT (
    SUM(CantidadSolicitada) FOR Estado IN
([Pendiente],[Entregado],[Cancelado])
```

```
) AS PVT;
```

```
-- Consulta 34: Mostrar CodArtículo con columnas para cada Estado
SELECT *
FROM (
    SELECT OD.CodArticulo, OD.Estado, OD.CantidadSolicitada
    FROM ORDEN_DETALLE OD
    JOIN ORDEN_DETALLE ON OD.NumOrden = Od.NumOrden
) SRC
PIVOT (SUM(CantidadSolicitada) FOR Estado IN ([Pendiente],[Entregado],[Cancelac
-- Consulta 35: Contar artículos por presentación y proveedor
SELECT A.Presentacion, A.CodProveedor, COUNT(*) AS TotalArticulos
FROM ARTICULO A
GROUP BY A.Presentacion, A.CodProveedor;

-- Consulta 36: Generar PIVOT dinámico para todas las tiendas
DECLARE @cols NVARCHAR(MAX), @sql NVARCHAR(MAX);
SELECT @cols = STRING_AGG(QUOTENAME(CodTienda), ',') FROM TIENDA;
SET @sql = '
SELECT * FROM (
    SELECT GE.CodTienda, GD.CodArticulo, GD.CantidadEnviada
```

Messages

CodArticulo	Pendiente	Entregado	Cancelado

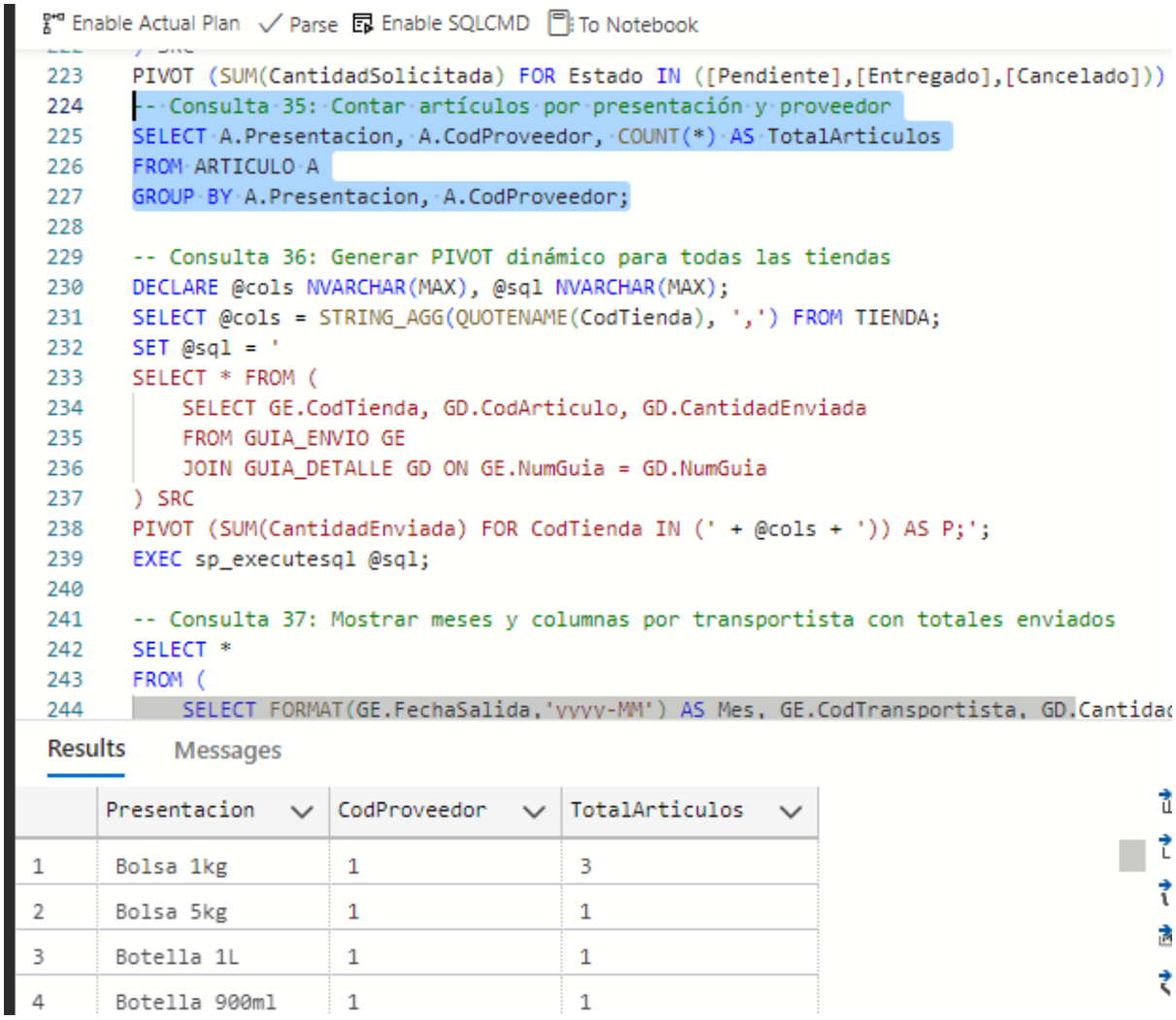
Explicación: Convierte los estados de las órdenes en columnas para comparar cantidades.

Consulta 35

Enunciado: Contar artículos por presentación y proveedor.

Consulta SQL:

```
SELECT
    A.Presentacion,
    A.CodProveedor,
    COUNT(*) AS TotalArticulos
FROM ARTICULO A
GROUP BY A.Presentacion, A.CodProveedor;
```



Explicación: Agrupa por tipo de presentación y proveedor, contando los artículos en cada caso.

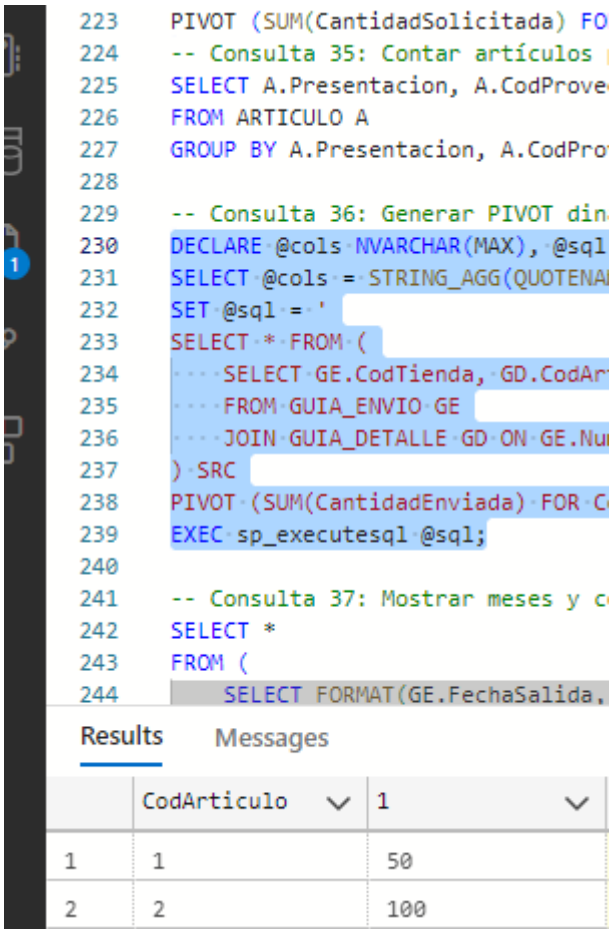
Consulta 36

Enunciado: Generar PIVOT dinámico para todas las tiendas (ejemplo de patrón).

Consulta SQL:

```
DECLARE @cols NVARCHAR(MAX), @sql NVARCHAR(MAX);
SELECT @cols = STRING_AGG(QUOTENAME(CodTienda), ',') FROM TIENDA;
SET @sql = '
SELECT * FROM (
    SELECT GE.CodTienda, GD.CodArticulo, GD.CantidadEnviada
    FROM GUIA_ENVIO GE
    JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
```

```
) SRC
PIVOT (SUM(CantidadEnviada) FOR CodTienda IN (' + @cols + ')) AS
P;';
EXEC sp_executesql @sql;
```



Explicación: Usa SQL dinámico para generar automáticamente un PIVOT con todas las tiendas registradas.

Consulta 37

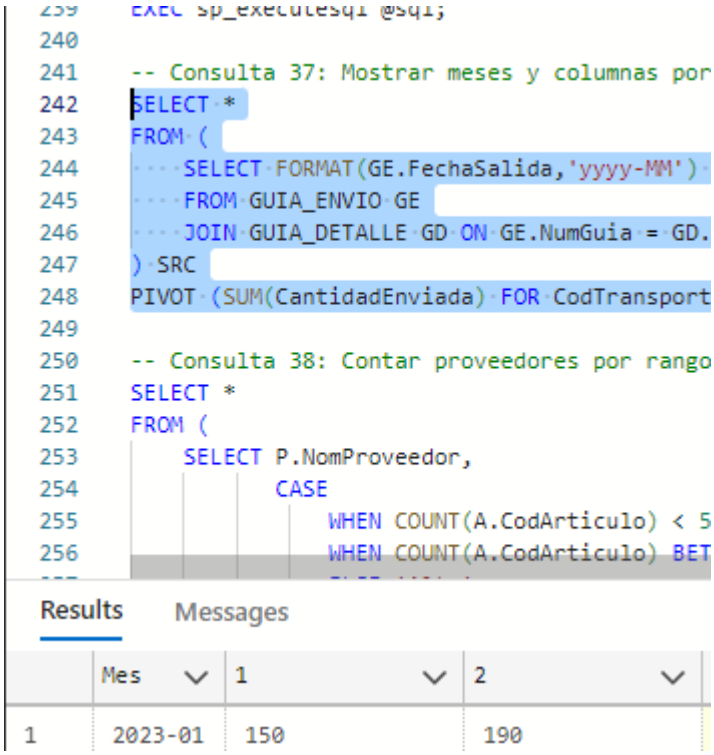
Enunciado: Mostrar meses y columnas por transportista con totales enviados.

Consulta SQL:

```
SELECT *
FROM (
    SELECT
        FORMAT(GE.FechaSalida,'yyyy-MM') AS Mes,
        GE.CodTransportista,
```

```
        GD.CantidadEnviada
    FROM GUIA_ENVIO GE
    JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
) SRC
PIVOT (
    SUM(CantidadEnviada) FOR CodTransportista IN ([1],[2],[3])
) AS PVT;
```

Explicación: Convierte los transportistas en columnas mostrando la cantidad enviada por mes.



Consulta 38

Enunciado: Contar proveedores por rango de variedad de artículos pivotado por columnas.

Consulta SQL:

```
SELECT *
FROM (
    SELECT
        P.NomProveedor,
        CASE
            WHEN COUNT(A.CodArticulo) < 5 THEN 'Baja'
```

```
            WHEN COUNT(A.CodArticulo) BETWEEN 5 AND 10 THEN 'Media'
            ELSE 'Alta'
        END AS RangoVariedad
    FROM PROVEEDOR P
    JOIN ARTICULO A ON P.CodProveedor = A.CodProveedor
    GROUP BY P.NomProveedor
) SRC
PIVOT (
    COUNT(NomProveedor) FOR RangoVariedad IN ([Baja],[Media],[Alta])
) AS PVT;
```

```
248  PIVOT (SUM(CantidadEnviada) FOR CodTransportista IN ([1],[2],[3])) AS PVT;
249
250  -- Consulta 38: Contar proveedores por rango de variedad de artículos
251  SELECT *
252  FROM (
253      SELECT P.NomProveedor,
254          CASE
255              WHEN COUNT(A.CodArticulo) < 5 THEN 'Baja'
256              WHEN COUNT(A.CodArticulo) BETWEEN 5 AND 10 THEN 'Media'
257              ELSE 'Alta'
258          END AS RangoVariedad
259      FROM PROVEEDOR P
260      JOIN ARTICULO A ON P.CodProveedor = A.CodProveedor
261      GROUP BY P.NomProveedor
262  ) SRC
263  PIVOT (COUNT(NomProveedor) FOR RangoVariedad IN ([Baja],[Media],[Alta])) AS PVT;
264
265  -- Consulta 39: Mostrar CodArtículo y volumen total por tienda
266  SELECT GD.CodArticulo, GE.CodTienda, SUM(GD.CantidadEnviada) AS VolumenTotal
267  FROM GUIA_ENVIO GE
268  JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
269  GROUP BY GD.CodArticulo, GE.CodTienda;
```

Results Messages

	Baja	Media	Alta
1	74	3	0

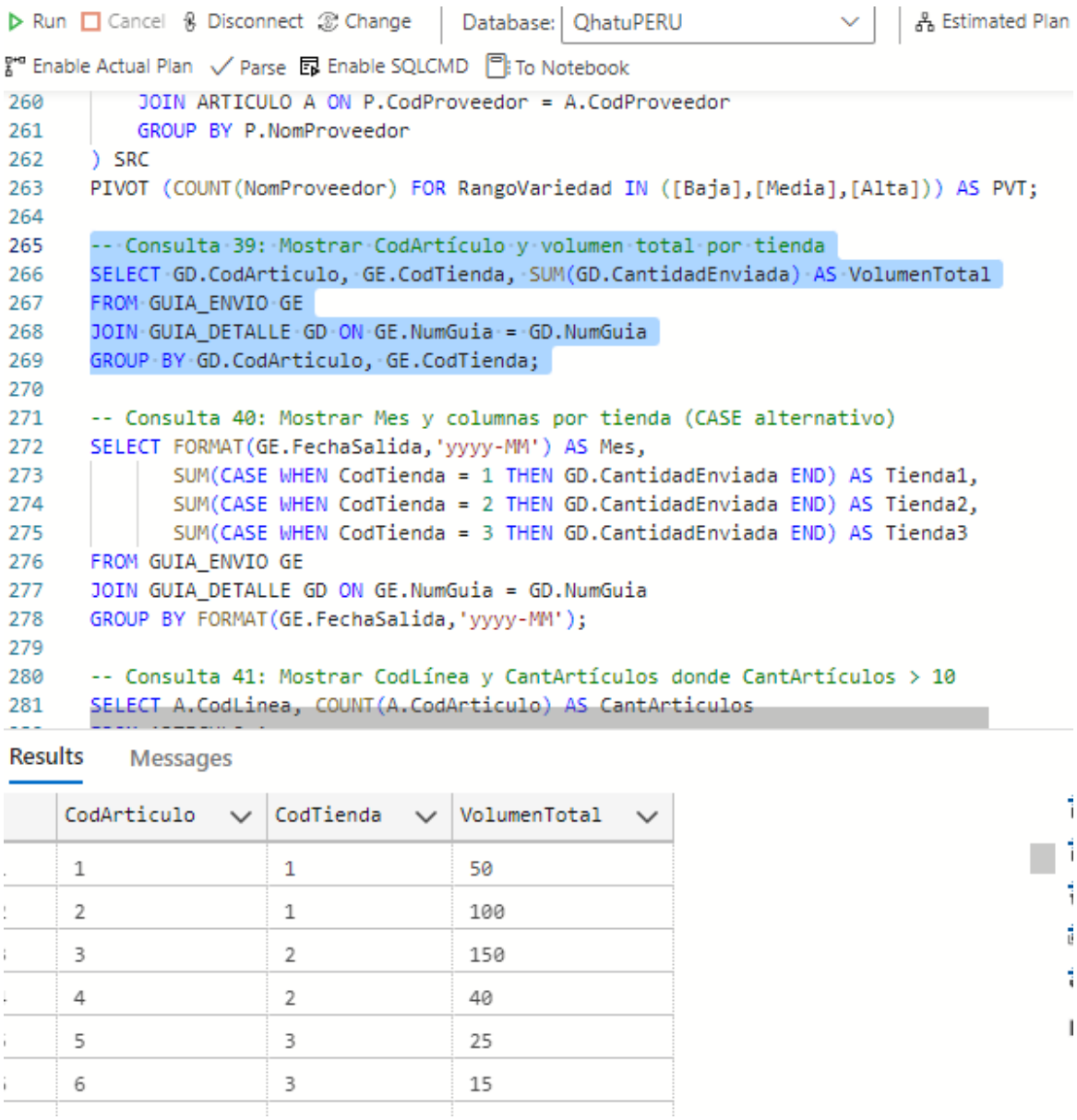
Explicación: Clasifica proveedores por nivel de variedad y los muestra como columnas.

Consulta 39

Enunciado: Mostrar CodArtículo y volumen total por tienda.

Consulta SQL:

```
SELECT
    GD.CodArticulo,
    GE.CodTienda,
    SUM(GD.CantidadEnviada) AS VolumenTotal
FROM GUIA_ENVIO GE
JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
GROUP BY GD.CodArticulo, GE.CodTienda;
```



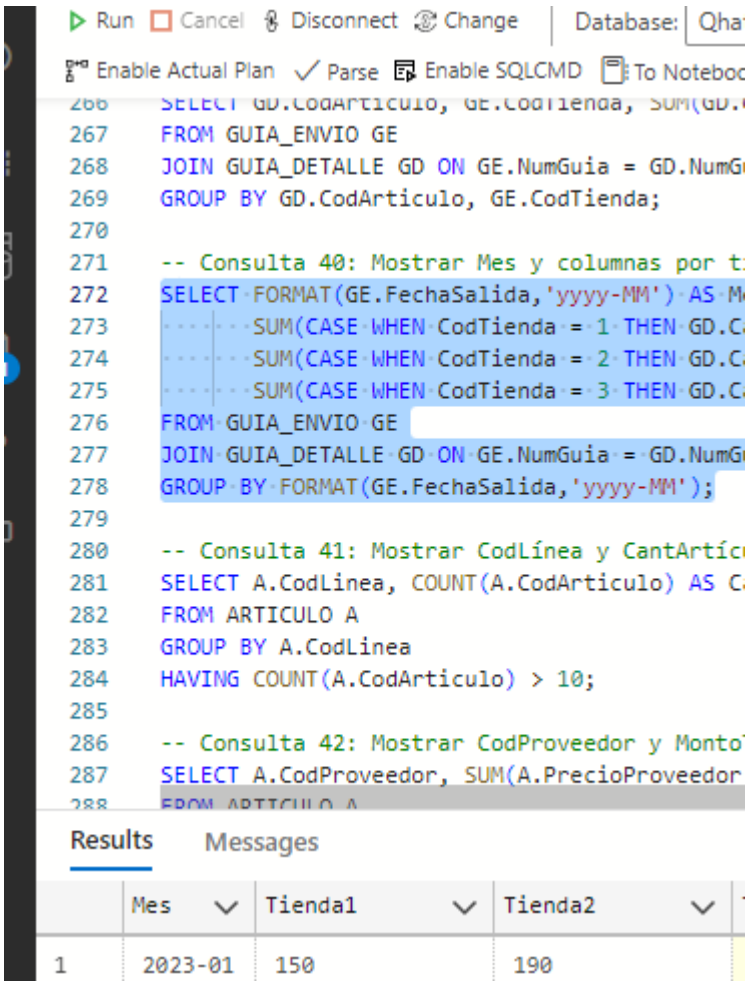
Explicación: Muestra el volumen de artículos enviados a cada tienda.

Consulta 40

Enunciado: Mostrar Mes y columnas por tienda (CASE alternativo).

Consulta SQL:

```
SELECT
    FORMAT(GE.FechaSalida,'yyyy-MM') AS Mes,
    SUM(CASE WHEN CodTienda = 1 THEN GD.CantidadEnviada END) AS
Tienda1,
    SUM(CASE WHEN CodTienda = 2 THEN GD.CantidadEnviada END) AS
Tienda2,
    SUM(CASE WHEN CodTienda = 3 THEN GD.CantidadEnviada END) AS
Tienda3
FROM GUIA_ENVIO GE
JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
GROUP BY FORMAT(GE.FechaSalida,'yyyy-MM');
```



Explicación: Reproduce un PIVOT utilizando condiciones CASE para generar columnas manuales.

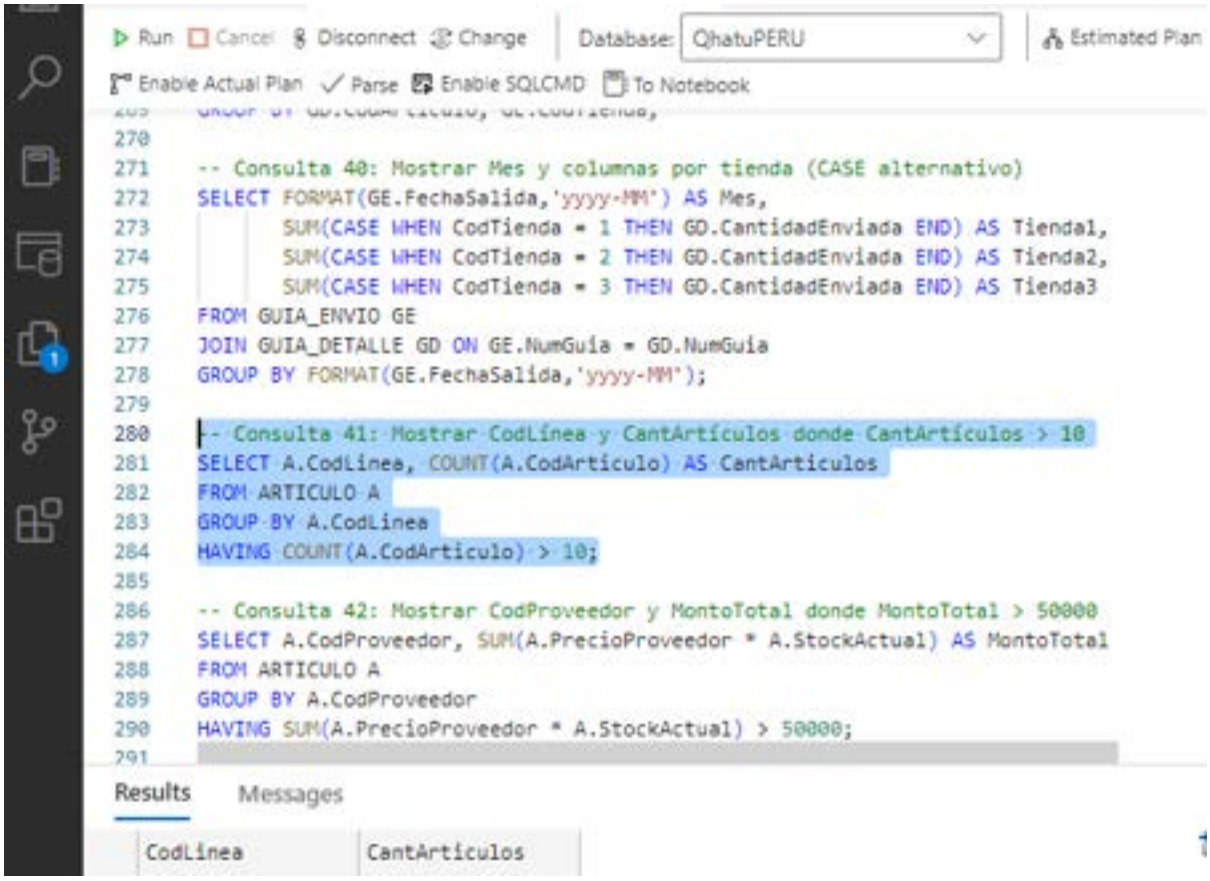
CLÁUSULA HAVING

Consulta 41

Enunciado: Mostrar CodLínea y CantArtículos donde CantArtículos > 10.

Consulta SQL:

```
SELECT
    A.CodLinea,
    COUNT(A.CodArticulo) AS CantArticulos
FROM ARTICULO A
GROUP BY A.CodLinea
HAVING COUNT(A.CodArticulo) > 10;
```



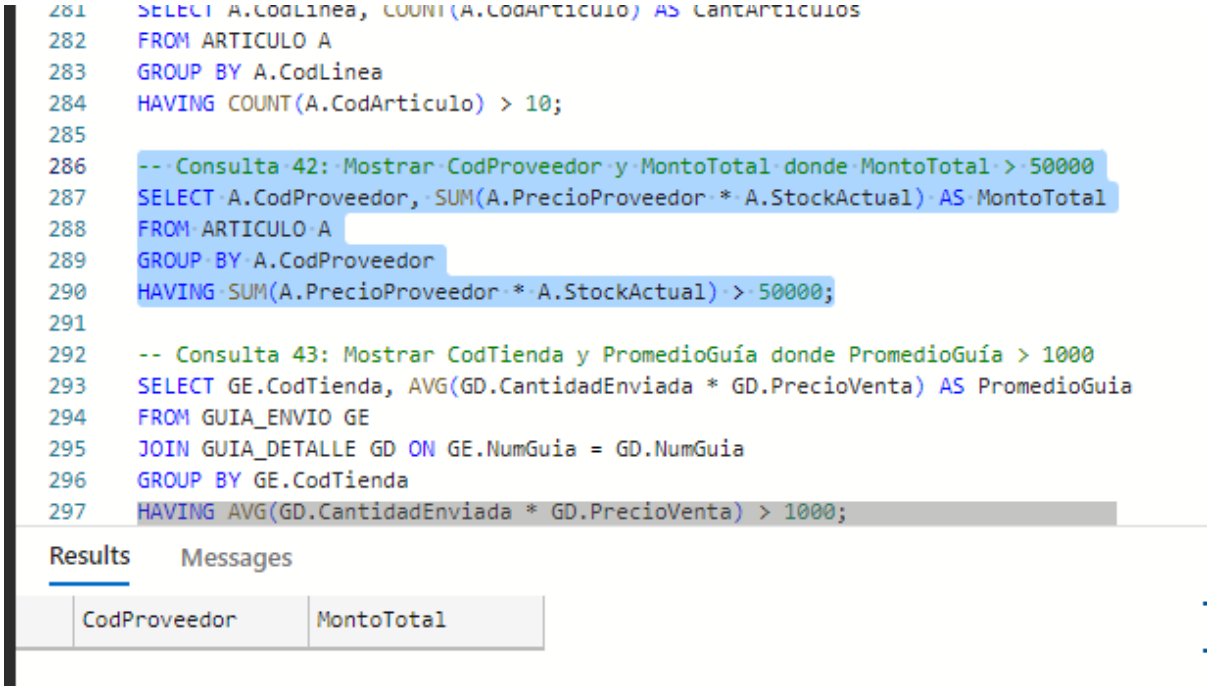
Explicación: Filtra líneas que superan las diez unidades de artículos registrados.

Consulta 42

Enunciado: Mostrar CodProveedor y MontoTotal donde MontoTotal > 50000.

Consulta SQL:

```
SELECT
    A.CodProveedor,
    SUM(A.PrecioProveedor * A.StockActual) AS MontoTotal
FROM ARTICULO A
GROUP BY A.CodProveedor
HAVING SUM(A.PrecioProveedor * A.StockActual) > 50000;
```



Explicación: Calcula el monto total por proveedor y muestra los que superan 50 mil.

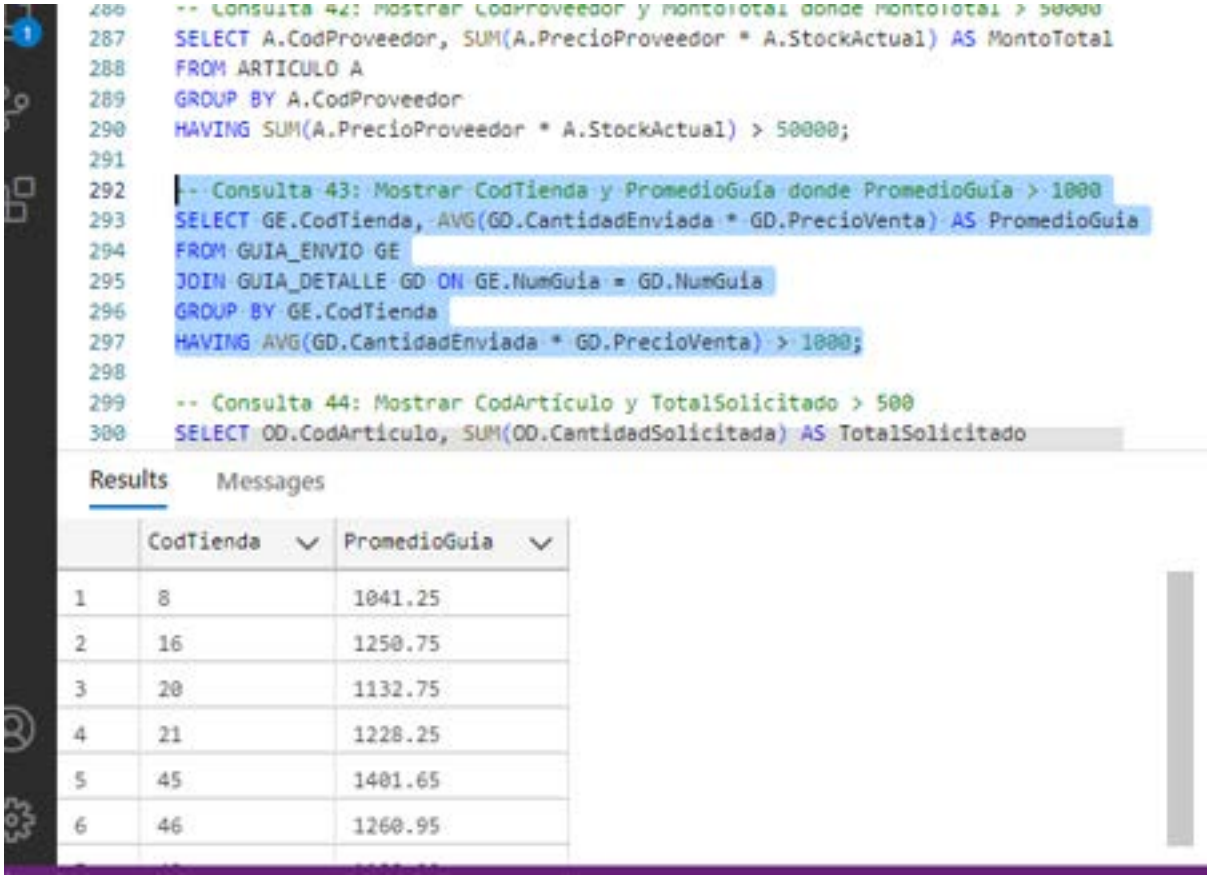
Consulta 43

Enunciado: Mostrar CodTienda y PromedioGuía donde PromedioGuía > 1000.

Consulta SQL:

```
SELECT
    GE.CodTienda,
```

```
    AVG(GD.CantidadEnviada * GD.PrecioVenta) AS PromedioGuia
FROM GUIA_ENVIO GE
JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
GROUP BY GE.CodTienda
HAVING AVG(GD.CantidadEnviada * GD.PrecioVenta) > 1000;
```



Explicación: Muestra tiendas con promedio de guías mayores a 1000 en ventas.

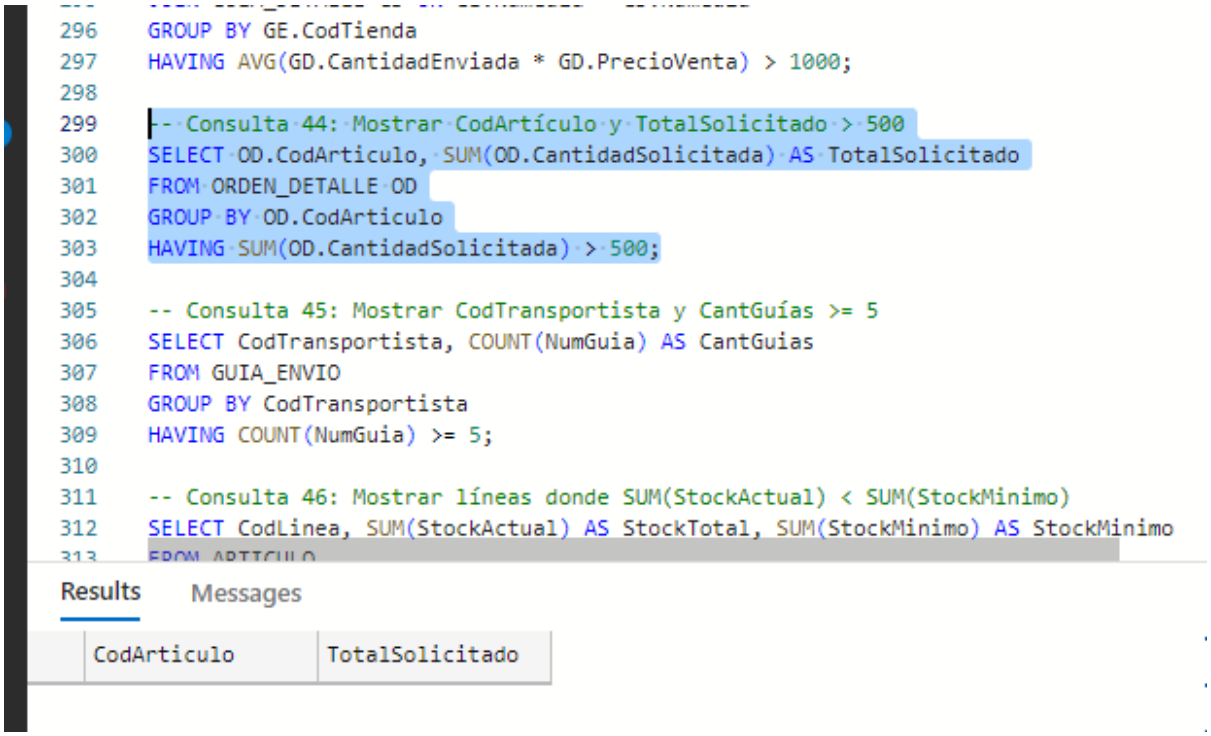
Consulta 44

Enunciado: Mostrar CodArtículo y TotalSolicitado > 500.

Consulta SQL:

```
SELECT
    OD.CodArticulo,
    SUM(OD.CantidadSolicitada) AS TotalSolicitado
FROM ORDEN_DETALLE OD
GROUP BY OD.CodArticulo
```

```
HAVING SUM(OD.CantidadSolicitada) > 500;
```



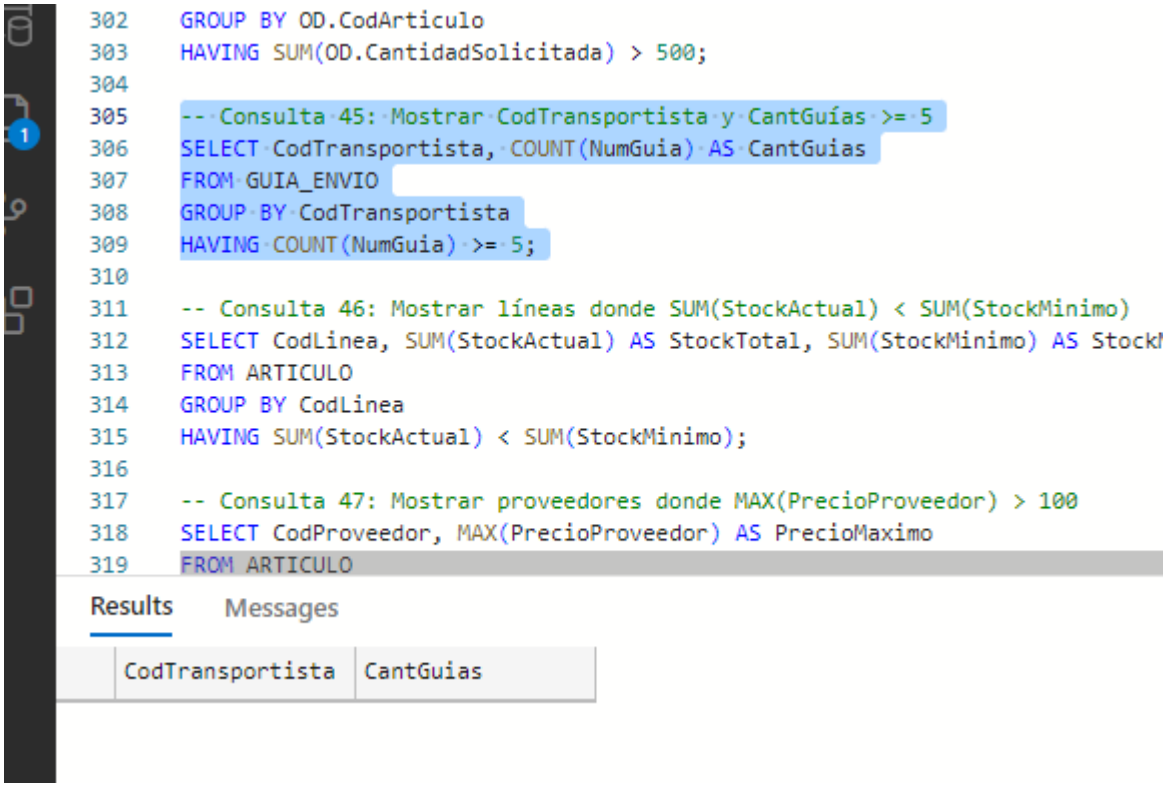
Explicación: Filtra los artículos que han sido solicitados más de 500 veces.

Consulta 45

Enunciado: Mostrar CodTransportista y CantGuías >= 5.

Consulta SQL:

```
SELECT
    CodTransportista,
    COUNT(NumGuia) AS CantGuias
FROM GUIA_ENVIO
GROUP BY CodTransportista
HAVING COUNT(NumGuia) >= 5;
```



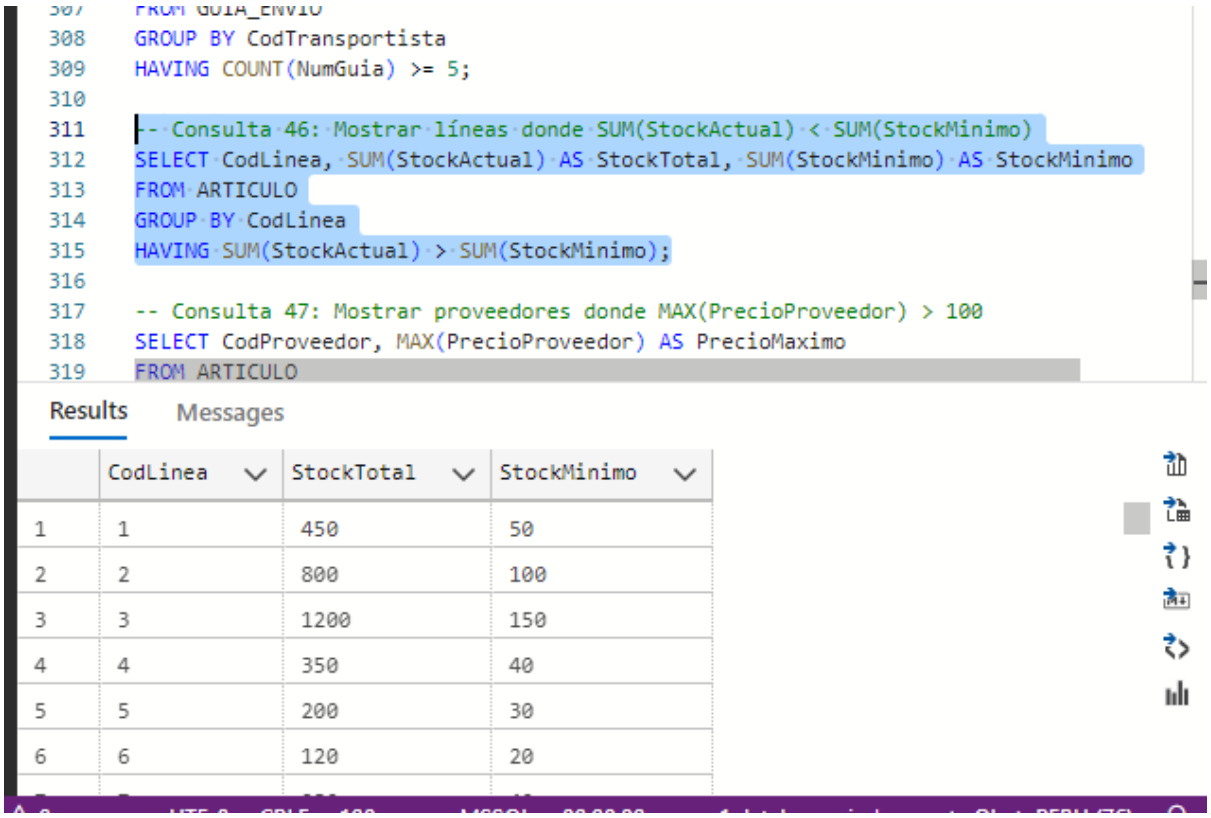
Explicación: Muestra transportistas que han gestionado cinco o más guías.

Consulta 46

Enunciado: Mostrar líneas donde SUM(StockActual) < SUM(StockMínimo * NumArtículosPorLínea).

Consulta SQL:

```
SELECT
    CodLinea,
    SUM(StockActual) AS StockTotal,
    SUM(StockMinimo) AS StockMinimo
FROM ARTICULO
GROUP BY CodLinea
HAVING SUM(StockActual) < SUM(StockMinimo);
```



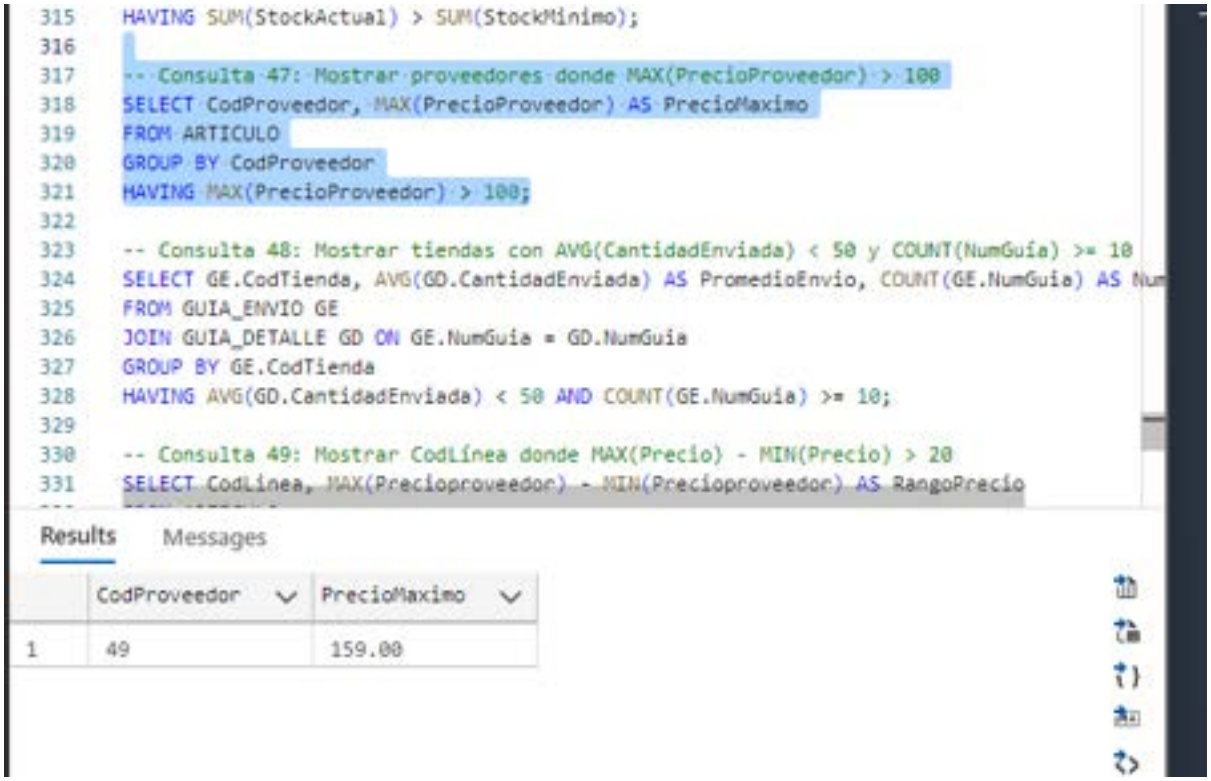
Explicación: Identifica líneas de productos con stock por debajo del mínimo esperado.

Consulta 47

Enunciado: Mostrar proveedores donde MAX(PrecioProveedor) > 100.

Consulta SQL:

```
SELECT
    CodProveedor,
    MAX(PrecioProveedor) AS PrecioMaximo
FROM ARTICULO
GROUP BY CodProveedor
HAVING MAX(PrecioProveedor) > 100;
```



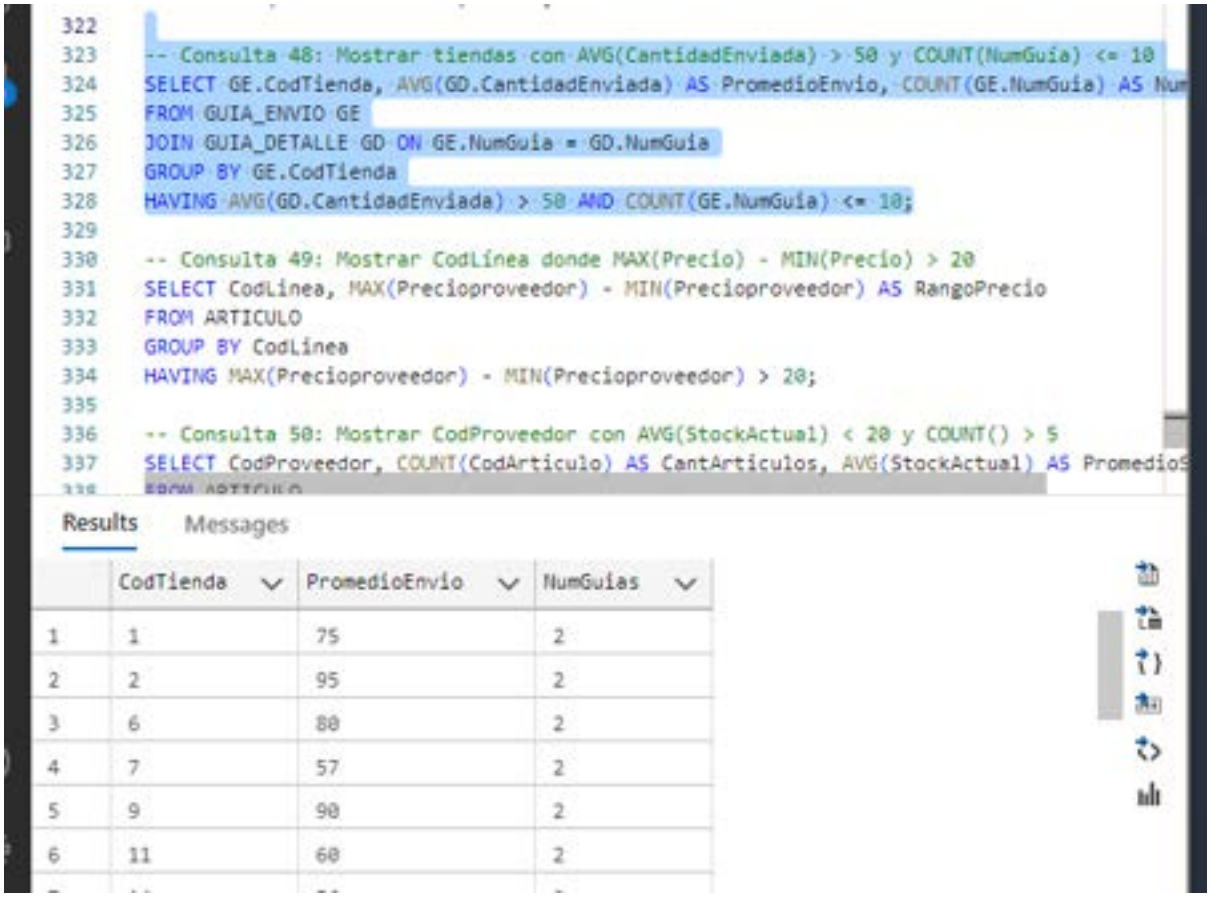
Explicación: Lista proveedores que ofrecen productos con precios superiores a 100.

Consulta 48

Enunciado: Mostrar tiendas con AVG(CantidadEnviada) >50 y COUNT(NumGuía) >= 10.

Consulta SQL:

```
SELECT
    GE.CodTienda,
    AVG(GD.CantidadEnviada) AS PromedioEnvio,
    COUNT(GE.NumGuia) AS NumGuias
FROM GUIA_ENVIO GE
JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
GROUP BY GE.CodTienda
HAVING AVG(GD.CantidadEnviada) > 50 AND COUNT(GE.NumGuia) >= 10;
```



```
322
323  -- Consulta 48: Mostrar tiendas con AVG(CantidadEnviada) > 50 y COUNT(NumGuia) <= 10
324  SELECT GE.CodTienda, AVG(GD.CantidadEnviada) AS PromedioEnvio, COUNT(GE.NumGuia) AS Num
325  FROM GUIA_ENVIO GE
326  JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
327  GROUP BY GE.CodTienda
328  HAVING AVG(GD.CantidadEnviada) > 50 AND COUNT(GE.NumGuia) <= 10;
329
330  -- Consulta 49: Mostrar CodLínea donde MAX(Precio) - MIN(Precio) > 20
331  SELECT CodLinea, MAX(Precioproveedor) - MIN(Precioproveedor) AS RangoPrecio
332  FROM ARTICULO
333  GROUP BY CodLinea
334  HAVING MAX(Precioproveedor) - MIN(Precioproveedor) > 20;
335
336  -- Consulta 50: Mostrar CodProveedor con AVG(StockActual) < 20 y COUNT() > 5
337  SELECT CodProveedor, COUNT(CodArticulo) AS CantArticulos, AVG(StockActual) AS PromedioS
```

Results Messages

	CodTienda	PromedioEnvio	NumGuias
1	1	75	2
2	2	95	2
3	6	80	2
4	7	57	2
5	9	90	2
6	11	60	2

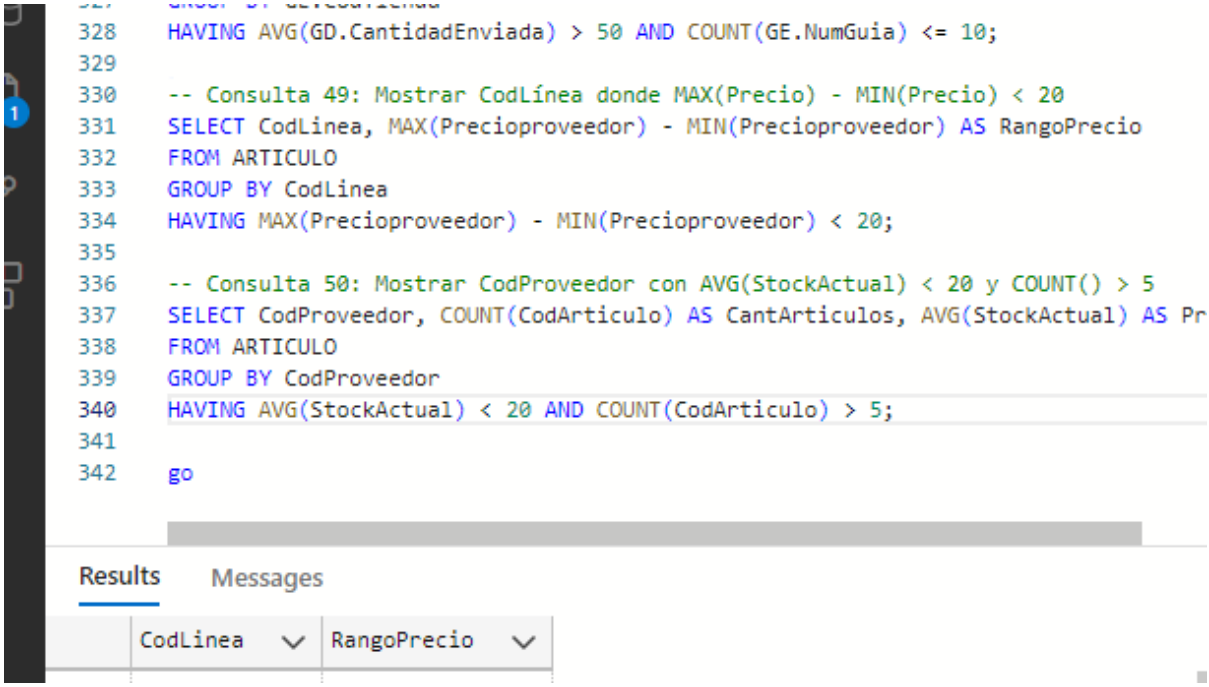
Explicación: Muestra tiendas con bajo promedio de envío pero con alta frecuencia de guías.

Consulta 49

Enunciado: Mostrar CodLínea donde MAX(Precio) - MIN(Precio) > 20.

Consulta SQL:

```
SELECT
    CodLinea,
    MAX(PrecioVenta) - MIN(PrecioVenta) AS RangoPrecio
FROM ARTICULO
GROUP BY CodLinea
HAVING MAX(PrecioVenta) - MIN(PrecioVenta) > 20;
```



Explicación: Calcula la diferencia entre el precio máximo y mínimo por línea.

Consulta 50

Enunciado: Mostrar CodProveedor con COUNT() artículos donde AVG(StockActual) < 20 y COUNT() > 5.

Consulta SQL:

```
SELECT
    CodProveedor,
    COUNT(CodArticulo) AS CantArticulos,
    AVG(StockActual) AS PromedioStock
FROM ARTICULO
GROUP BY CodProveedor
HAVING AVG(StockActual) < 20 AND COUNT(CodArticulo) > 5;
```

```
334    HAVING MAX(Precioproveedor) - MIN(Precioproveedor) < 20;
335
336    -- Consulta 50: Mostrar CodProveedor con AVG(StockActual) < 2 y COUNT() > 5
337    SELECT CodProveedor, COUNT(CodArticulo) AS CantArticulos, AVG(StockActual) AS Promed
338    FROM ARTICULO
339    GROUP BY CodProveedor
340    HAVING AVG(StockActual) < 20 AND COUNT(CodArticulo) > 5;
341
342    go
```

Results Messages

	CodProveedor	CantArticulos	PromedioStock

Explicación: Filtra proveedores con bajo promedio de stock y más de cinco artículos registrados.