

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;

The screenshot shows the SQL Server Enterprise Manager interface. The top pane displays a query in the 'SQLQuery_2' window, which is connected to the 'QhatuPERU' database. The query is as follows:

```
1 SELECT
2     A.CodArticulo,
3     A.DescripcionArticulo,
4     CAST(A.StockActual * CAST(A.PrecioProveedor AS DECIMAL(18,2)) AS DECIMAL(18,2)) AS
5 FROM ARTICULO A;
6
7 GO
```

The bottom pane shows the 'Results' tab with a table containing 14 rows of data. The columns are 'CodArticulo', 'DescripcionArticulo', and 'ValorInventario'.

| | CodArticulo | DescripcionArticulo | ValorInventario |
|----|-------------|-----------------------|-----------------|
| 1 | 1 | Aceite Primor | 5625.00 |
| 2 | 2 | Leche Gloria Entera | 3360.00 |
| 3 | 3 | Agua San Luis | 1800.00 |
| 4 | 4 | Pan Simbo Blanco | 2380.00 |
| 5 | 5 | Pollo San Fernando | 1900.00 |
| 6 | 6 | Filete de Bonito | 1896.00 |
| 7 | 7 | Manzana Delicia | 1260.00 |
| 8 | 8 | Tomate Italiano | 1120.00 |
| 9 | 9 | Helado Laive Vainilla | 2682.00 |
| 10 | 10 | Papas Lays Clásicas | 2280.00 |
| 11 | 11 | Chocolate Sublime | 1875.00 |
| 12 | 12 | Atún Costeño | 2704.00 |
| 13 | 13 | Cereal Nestlé Fitness | 2856.00 |
| 14 | 14 | Eideos Don Vittorio | 3332.00 |

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;

```
5
6 -- Consulta 2: Calcular el total monetario del inventario
7 SELECT CAST(SUM(A.StockActual * CAST(A.PrecioProveedor AS DECIMAL(18,2))) AS DECIMAL(18,2)) AS TotalInventario
8 FROM ARTICULO A;
9
10 -- Consulta 3: Obtener CodLinea y PrecioProveedor promedio
11 SELECT A.CodLinea,
12        CAST(AVG(CAST(A.PrecioProveedor AS DECIMAL(18,2))) AS DECIMAL(18,2)) AS PrecioPromedio
```

Results Messages

| | TotalInventario |
|---|-----------------|
| 1 | 347657.50 |

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

Consulta 3

Enunciado: Obtener CodLinea 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;
```

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'SERVERS' tree is expanded to show the 'Articulos' database. The 'Tables' folder is selected, and a list of tables is shown: CodProveedor (PK, int, not null), DescripcionArticulo (varchar(40)), PrecioProveedor (money, null), StockActual (decimal(18,2), null), StockMinimo (decimal(18,2), null), and Descontinuado (bit, null). The 'Articulos' table is selected. The main window displays a query window with the following SQL code:

```
-- Consulta 2: Calcular el total monetario del inventario
SELECT CAST(SUM(A.StockActual * CAST(A.PrecioProveedor AS DECIMAL(18,2))) AS DECIMAL(18,2)) AS TotalInventario
FROM ARTICULO A;

-- Consulta 3: Obtener CodLinea y PrecioProveedor promedio
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;

-- Consulta 4: Obtener articulos descontinuados
SELECT COUNT(*) AS articulosDescontinuados
FROM ARTICULO
```

The results grid shows the following data:

| | CodLinea | PrecioPromedio |
|----|----------|----------------|
| 1 | 1 | 12.50 |
| 2 | 2 | 4.20 |
| 3 | 3 | 5.50 |
| 4 | 4 | 6.00 |
| 5 | 5 | 5.50 |
| 6 | 6 | 15.50 |
| 7 | 7 | 4.50 |
| 8 | 8 | 5.20 |
| 9 | 9 | 14.50 |
| 10 | 10 | 5.50 |
| 11 | 11 | 5.50 |
| 12 | 12 | 5.20 |
| 13 | 13 | 12.50 |
| 14 | 14 | 5.50 |

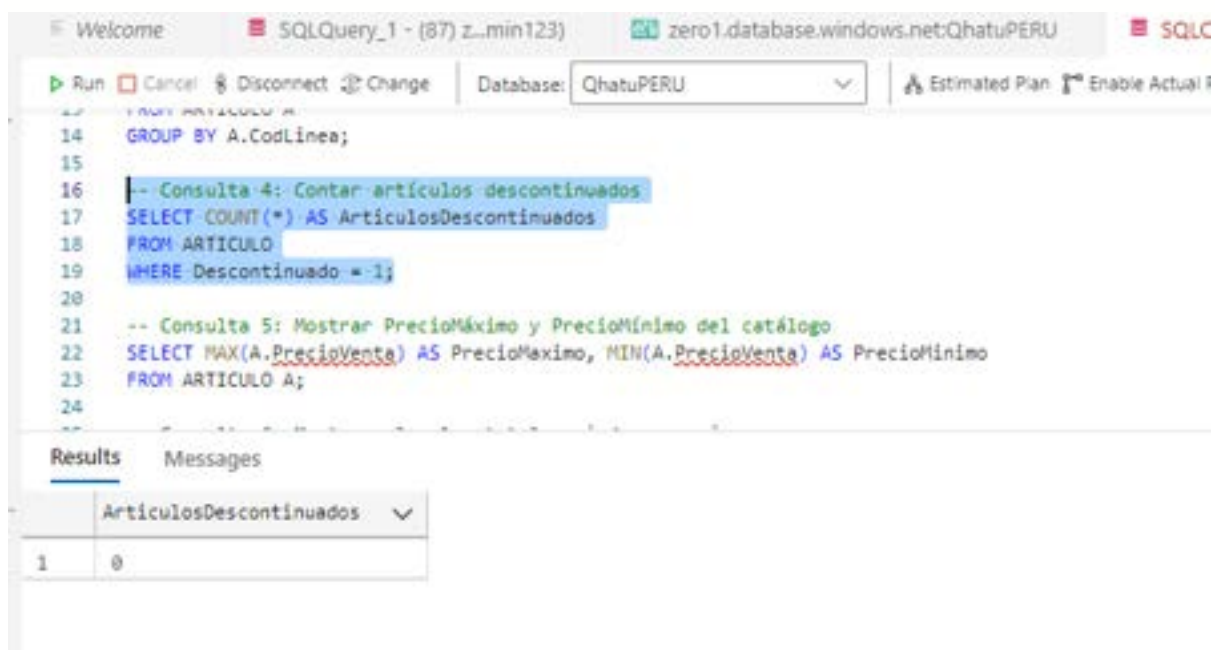
Explicación: Agrupa por línea de producto y calcula el precio promedio de proveedor.

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;
```

Run Cancel Disconnect Change Database: QhataPERU Estimated Plan Enable Actual Plan

```

20:
21: -- Consulta 5: Mostrar PrecioMáximo y PrecioMínimo del catálogo
22: SELECT MAX(A.PrecioProveedor) AS PrecioMaximo, MIN(A.PrecioProveedor) AS PrecioMinimo
23: FROM ARTICULO A;
24:
25: -- Consulta 6: Mostrar el valor total enviado por guía
26: SELECT GD.NumGuia,
27:        CAST(SUM(GD.CantidadEnviada * CAST(GD.PrecioVenta AS DECIMAL(18,2))) AS DECIMAL(18,2)) AS ValorTo
28: FROM GUIA_DETALLE GD
29: GROUP BY GD.NumGuia;
30:
31: -- Consulta 7: Para cada Compañía, mostrar TotalSolicitado

```

Results Messages

| | PrecioMaximo | PrecioMinimo |
|---|--------------|--------------|
| 1 | 159.00 | 1.50 |

Explicación: Obtiene los valores extremos del precio del proveedor 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;

```

```

23 FROM ARTICULO A;
24
25 -- Consulta 6: Mostrar el valor total enviado por guía
26 SELECT GD.NumGuia,
27        CAST(SUM(GD.CantidadEnviada * CAST(GD.PrecioVenta AS DECIMAL(18,2))) AS DECIMAL(18,2)) AS ValorTotal
28 FROM GUIA_DETALLE GD
29 GROUP BY GD.NumGuia;
30
31 -- Consulta 7: Para cada CodArtículo, mostrar TotalSolicitado
32 SELECT OD.CodArtículo, SUM(OD.CantidadSolicitada) AS TotalSolicitado
33 FROM ORDEN_DETALLE OD
34 GROUP BY OD.CodArtículo;

```

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 | 2002.50 |
| 9 | 5009 | 826.00 |
| 10 | 5010 | 846.50 |
| 11 | 5011 | 1444.00 |
| 12 | 5012 | 1076.60 |
| 13 | 5013 | 1357.30 |

Activar Win

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

Consulta 7

Enunciado: Para cada CodArtículo, mostrar TotalSolicitado.

Consulta SQL:

```

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

```

Run Cancel Disconnect Change Database: QhatsPERU Estimated Plan Enable Actual Plan Page Enable SQLCMD 1

```

26 SELECT GD.NumGuia,
27        CAST(SUM(GD.CantidadEnviada * CAST(GD.PrecioVenta AS DECIMAL(18,2))) AS DECIMAL(18,2)) AS ValorTotal
28 FROM GUIA_DETALLE GD
29 GROUP BY GD.NumGuia;
30
31 -- Consulta 7: Para cada CodArtículo, mostrar TotalSolicitado
32 SELECT OD.CodArtículo, SUM(OD.CantidadSolicitada) AS TotalSolicitado
33 FROM ORDEN_DETALLE OD
34 GROUP BY OD.CodArtículo;
35
36 -- Consulta 8: Contar órdenes únicas que incluyen cada artículo
37 SELECT OD.CodArtículo, COUNT(DISTINCT OD.NumOrden) AS OrdenesUnicas

```

Results Messages

| | CodArtículo | TotalSolicitado |
|----|-------------|-----------------|
| 1 | 1 | 100 |
| 2 | 2 | 200 |
| 3 | 3 | 300 |
| 4 | 4 | 80 |
| 5 | 5 | 50 |
| 6 | 6 | 30 |
| 7 | 7 | 70 |
| 8 | 8 | 90 |
| 9 | 9 | 45 |
| 10 | 10 | 150 |
| 11 | 11 | 200 |
| 12 | 12 | 120 |
| 13 | 13 | 60 |
| 14 | 14 | 170 |

Activar Windows
Ve a Configuración para activar

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

Consulta 8

Enunciado: Contar órdenes únicas que incluyen cada artículo.

Consulta SQL:

```

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

```


Run Cancel Disconnect Change Database: QhatuPERU Estimated

Enable Actual Plan Parse Enable SQLCMD To Notebook

```

32 SELECT OD.CodArticulo, SUM(OD.CantidadSolicitada) AS TotalSolicitado
33 FROM ORDEN_DETALLE OD
34 GROUP BY OD.CodArticulo;
35
36 -- Consulta 8: Contar órdenes únicas que incluyen cada artículo
37 SELECT OD.CodArticulo, COUNT(DISTINCT OD.NumOrden) AS OrdenesUnicas
38 FROM ORDEN_DETALLE OD
39 GROUP BY OD.CodArticulo;
40
41 -- Consulta 9: Calcular promedio de días de orden con FechaIngreso
42 SELECT CAST(AVG(DATEDIFF(DAY, O.FechaIngreso, GETDATE())) AS DECIMAL(10,2)) AS
43 FROM ORDEN O;

```

Results Messages

| | CodArticulo | OrdenesUnicas |
|----|-------------|---------------|
| 1 | 1 | 1 |
| 2 | 2 | 1 |
| 3 | 3 | 1 |
| 4 | 4 | 1 |
| 5 | 5 | 1 |
| 6 | 6 | 1 |
| 7 | 7 | 1 |
| 8 | 8 | 1 |
| 9 | 9 | 1 |
| 10 | 10 | 1 |
| 11 | 11 | 1 |
| 12 | 12 | 1 |
| 13 | 13 | 1 |

Activar Windows
Ve a Configuración para activar Windows.

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;

```

The screenshot shows a SQL query editor with two queries. Query 9 is highlighted in blue and calculates the average days from the order date to the receipt date. Query 10 is highlighted in green and sums the quantity sent by transportist. The results of Query 9 are shown in a table below the queries.

```

39  GROUP BY OD.CodArticulo;
40
41  -- Consulta 9: Calcular promedio de días de orden con FechaIngreso
42  SELECT CAST(AVG(DATEDIFF(DAY, O.FechaIngreso, GETDATE())) AS DECIMAL(10,2)) AS Promedi
43  FROM ORDEN_COMPRA O;
44
45  -- Consulta 10: Sumar CantidadEnviada por CodTransportista
46  SELECT GE.CodTransportista, SUM(GD.CantidadEnviada) AS TotalEnviado
47  FROM GUIA_ENVIO GE
48  JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
49  GROUP BY GE.CodTransportista;

```

Results Messages

| | PromedioDias |
|---|--------------|
| 1 | 684.00 |

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;

```

The screenshot shows the SQL Server Enterprise Manager interface. The top bar indicates the connection to 'zero1.database.windows.net:QhatuPERU'. The 'Database' dropdown is set to 'QhatuPERU'. The query editor displays the following SQL code:

```

42 SELECT CAST(AVG(DATEDIFF(DAY, O.FechaIngreso, GETDATE())) AS DECIMAL(10,2)) AS Proc
43 FROM ORDEN_COMPRA O;
44
45 -- Consulta 10: Sumar CantidadEnviada por CodTransportista
46 SELECT GE.CodTransportista, SUM(GD.CantidadEnviada) AS TotalEnviado
47 FROM GUIA_ENVIO GE
48 JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
49 GROUP BY GE.CodTransportista;
50
51 -- Consulta 11: Mostrar NomLínea y CantArtículos
52 SELECT L.NomLínea, COUNT(A.CodArtículo) AS CantArtículos

```

Below the query editor, the 'Results' tab is active, displaying the output of the query. The results are shown in a table with two columns: 'CodTransportista' and 'TotalEnviado'.

| | CodTransportista | TotalEnviado |
|----|------------------|--------------|
| 1 | 1 | 150 |
| 2 | 2 | 190 |
| 3 | 3 | 40 |
| 4 | 4 | 80 |
| 5 | 5 | 95 |
| 6 | 6 | 160 |
| 7 | 7 | 115 |
| 8 | 8 | 95 |
| 9 | 9 | 180 |
| 10 | 10 | 95 |
| 11 | 11 | 120 |
| 12 | 12 | 94 |
| 13 | 13 | 77 |
| 14 | 14 | 117 |

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;

```

The screenshot shows the SQL Server Enterprise Manager interface. The top bar indicates the connection to 'zero1.database.windows.net:QhatuPERU'. The query editor displays the following SQL code:

```

54 JOIN ARTICULO A ON L.CodLinea = A.CodLinea
55 GROUP BY L.NomLinea;
56
57 -- Consulta 12: Mostrar CodLinea y StockTotal
58 SELECT A.CodLinea, SUM(A.StockActual) AS StockTotal
59 FROM ARTICULO A
60 GROUP BY A.CodLinea;
61
62 -- Consulta 13: Para cada NumOrden, calcular CostoTotal
63 SELECT OO.NumOrden, SUM(OO.PrecioCompra * OO.CantidadSolicitada) AS CostoTotal
64 FROM ORDEN_DETALLE OO
65 GROUP BY OO.NumOrden;

```

The 'Results' tab is active, showing the output of the first query. The results are as follows:

| | CodLinea | StockTotal |
|----|----------|------------|
| 1 | 1 | 450 |
| 2 | 2 | 800 |
| 3 | 3 | 1200 |
| 4 | 4 | 350 |
| 5 | 5 | 200 |
| 6 | 6 | 120 |
| 7 | 7 | 280 |
| 8 | 8 | 350 |
| 9 | 9 | 180 |
| 10 | 10 | 600 |
| 11 | 11 | 750 |
| 12 | 12 | 520 |
| 13 | 13 | 240 |
| 14 | 14 | 680 |

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;

```

zero1.database.windows.net:QhatuPERU SQLQuery_2 - (85) z...min123) 9+ ...

Run Cancel Disconnect Change Database: QhatuPERU Estimated Plan

Enable Actual Plan Parse Enable SQLCMD To Notebook

```

60 GROUP BY A.CodLines;
61
62 -- Consulta 13: Para cada NumOrden, calcular CostoTotal
63 SELECT OD.NumOrden, SUM(OD.PrecioCompra * OD.CantidadSolicitada) AS CostoTotal
64 FROM ORDEN_DETALLE OD
65 GROUP BY OD.NumOrden;
66
67 -- Consulta 14: Mostrar NumGuía y PromedioEnviado
68 SELECT GD.NumGuía, AVG(GD.CantidadEnviada) AS PromedioEnviado
69 FROM GUIA_DETALLE GD
70 GROUP BY GD.NumGuía;
71

```

Results Messages

| | NumOrden | CostoTotal |
|----|----------|------------|
| 1 | 1001 | 2090.00 |
| 2 | 1002 | 994.00 |
| 3 | 1003 | 949.00 |
| 4 | 1004 | 603.00 |
| 5 | 1005 | 1240.50 |
| 6 | 1006 | 1124.00 |
| 7 | 1007 | 1547.00 |
| 8 | 1008 | 2871.00 |
| 9 | 1009 | 1106.00 |
| 10 | 1010 | 1142.00 |
| 11 | 1011 | 1969.00 |
| 12 | 1012 | 1439.00 |
| 13 | 1013 | 1801.00 |
| 14 | 1014 | 1810.50 |

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;

zero1.database.windows.net:QhatuPERU SQLQuery_2 - (85) z...min123) 9+ ...

Run Cancel Disconnect Change Database: QhatuPERU Estimated Plan

Enable Actual Plan Parse Enable SQLCMD To Notebook

```

70 GROUP BY GD.NumGuia;
71
72 -- Consulta 15: Mostrar CodProveedor y TotalArticulos
73 SELECT A.CodProveedor, COUNT(A.CodArticulo) AS TotalArticulos
74 FROM ARTICULO A
75 GROUP BY A.CodProveedor;
76
77 -- Consulta 16: Mostrar número de órdenes de entrega por día
78 SELECT CAST(FechaSalida AS DATE) AS Fecha, COUNT(DISTINCT NumGuia) AS NumOrdenes
79 FROM GUIA_ENVIO
80 GROUP BY CAST(FechaSalida AS DATE)

```

Results Messages

| | CodProveedor | TotalArticulos |
|----|--------------|----------------|
| 1 | 1 | 9 |
| 2 | 2 | 1 |
| 3 | 3 | 3 |
| 4 | 4 | 2 |
| 5 | 5 | 1 |
| 6 | 6 | 2 |
| 7 | 7 | 1 |
| 8 | 8 | 1 |
| 9 | 9 | 5 |
| 10 | 10 | 1 |
| 11 | 11 | 1 |
| 12 | 12 | 1 |
| 13 | 14 | 1 |
| 14 | 18 | 1 |

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

Consulta 16

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

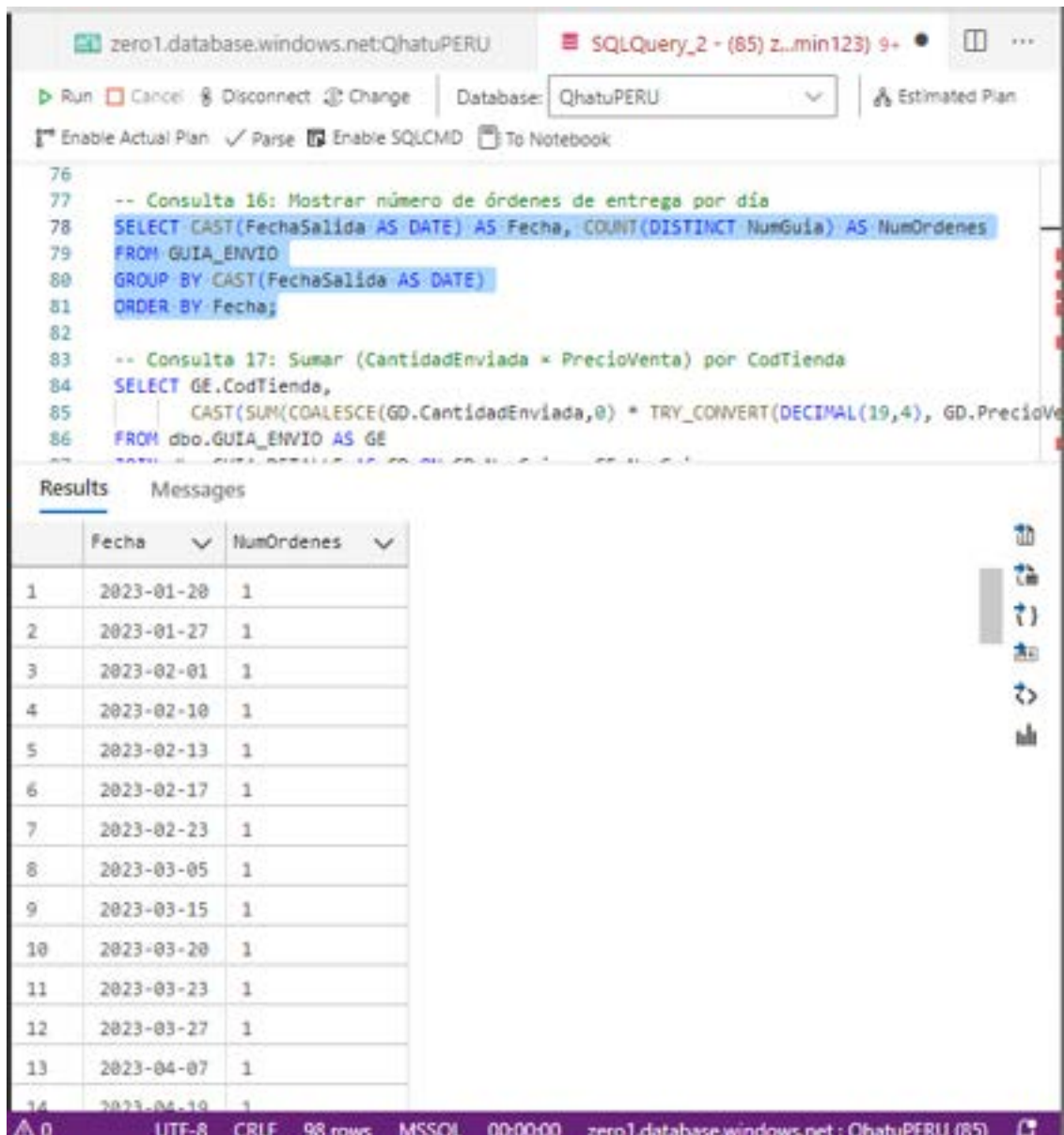
Consulta SQL:

```

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

```


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



The screenshot shows a SQL Server Enterprise Manager window with a query editor and a results pane. The query editor displays two SQL queries. The first query, labeled 'Consulta 16', is a SELECT statement that groups data by the date part of 'FechaSalida' and counts the number of distinct 'NumGuia' values. The second query, labeled 'Consulta 17', is a SELECT statement that calculates the sum of 'CantidadEnviada' multiplied by 'PrecioVenta' for each 'CodTienda'.

Consulta 16:

```
-- Consulta 16: Mostrar número de órdenes de entrega por día
SELECT CAST(FechaSalida AS DATE) AS Fecha, COUNT(DISTINCT NumGuia) AS NumOrdenes
FROM GUIA_ENVIO
GROUP BY CAST(FechaSalida AS DATE)
ORDER BY Fecha;
```

Consulta 17:

```
-- Consulta 17: Sumar (CantidadEnviada * PrecioVenta) por CodTienda
SELECT GE.CodTienda,
       CAST(SUM(COALESCE(GD.CantidadEnviada,0) * TRY_CONVERT(DECIMAL(19,4), GD.PrecioVe
FROM dbo.GUIA_ENVIO AS GE
```

The results pane shows the output of the first query, displaying a table with two columns: 'Fecha' and 'NumOrdenes'. The table contains 14 rows of data, showing the number of orders for each date.

| | Fecha | NumOrdenes |
|----|------------|------------|
| 1 | 2023-01-20 | 1 |
| 2 | 2023-01-27 | 1 |
| 3 | 2023-02-01 | 1 |
| 4 | 2023-02-10 | 1 |
| 5 | 2023-02-13 | 1 |
| 6 | 2023-02-17 | 1 |
| 7 | 2023-02-23 | 1 |
| 8 | 2023-03-05 | 1 |
| 9 | 2023-03-15 | 1 |
| 10 | 2023-03-20 | 1 |
| 11 | 2023-03-23 | 1 |
| 12 | 2023-03-27 | 1 |
| 13 | 2023-04-07 | 1 |
| 14 | 2023-04-10 | 1 |

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

Consulta 17

Enunciado: Sumar (CantidadEnviada × PrecioVenta) por CodTienda.

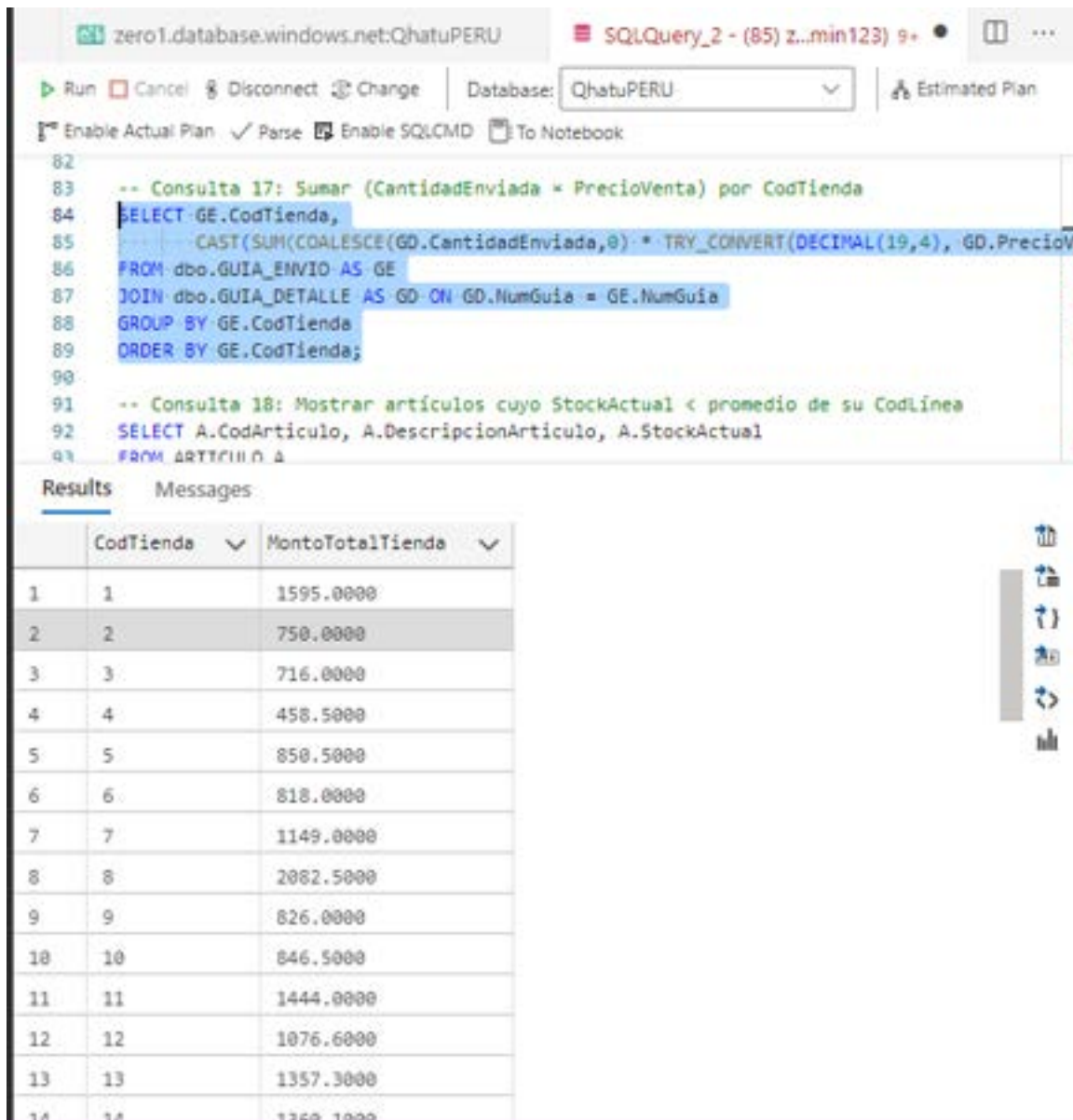
Consulta SQL:

```
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;

```



The screenshot shows a SQL Server Enterprise Manager window with a query editor and a results grid. The query editor displays a SQL query for 'Consulta 17' which calculates the total sales amount per store. The results grid shows 14 rows of data with columns 'CodTienda' and 'MontoTotalTienda'.

Query:

```

-- Consulta 17: Sumar (CantidadEnviada * PrecioVenta) por CodTienda
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;

```

Results:

| | CodTienda | MontoTotalTienda |
|----|-----------|------------------|
| 1 | 1 | 1595.0000 |
| 2 | 2 | 750.0000 |
| 3 | 3 | 716.0000 |
| 4 | 4 | 458.5000 |
| 5 | 5 | 850.5000 |
| 6 | 6 | 818.0000 |
| 7 | 7 | 1149.0000 |
| 8 | 8 | 2082.5000 |
| 9 | 9 | 826.0000 |
| 10 | 10 | 846.5000 |
| 11 | 11 | 1444.0000 |
| 12 | 12 | 1076.6000 |
| 13 | 13 | 1357.3000 |
| 14 | 14 | 1360.1000 |

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 CantArticulos
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 | Panas 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(OD.CantidadSolicitada) AS TotalSolicitado
105 FROM ORDEN_detalle O
106 JOIN ORDEN_DETALLE OD ON O.NumOrden = OD.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;

```

```

109  -- Consulta 21: Asignar posición por línea ordenada por precio
110  SELECT CodLinea, CodArticulo, PrecioProveedor,
111         ROW_NUMBER() OVER (PARTITION BY CodLinea ORDER BY PrecioProveedor 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
118  FROM ORDEN_DETALLE
119  GROUP BY NumOrden;
120

```

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:

```

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 | 3087.00 | 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 |

Activar M
Ve a Confia

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);

```



```

129 -- Consulta 24: Calcular promedio móvil para stock
130 SELECT CodArticulo, CodLinea, StockActual,
131        (StockActual +
132         COALESCE(LAG(StockActual, 1) OVER (PARTITION BY CodArticulo ORDER BY CodLinea), 0) +
133         COALESCE(LAG(StockActual, 2) OVER (PARTITION BY CodArticulo ORDER BY CodLinea), 0)) / 3 AS PromedioMovil
134 FROM articulo;
135
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;

```

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 |

Activar Windows
Ve a Configuración para activar Windows.

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 CantidadPorLinea 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 CantidadPorLinea a cada artículo.

Consulta SQL:

```

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

```

FROM ARTICULO;

```
141
142 -- Consulta 26: Añadir columna CantidadPorLinea 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 |

Activar Win
Ve a Configuraci

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;
```

```

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()) AS DECIMAL(18,2) AS Porcentaje
150 FROM ARTICULO
151 GROUP BY CodProveedor;
152
153 -- Consulta 28: Mostrar solo los 3 artículos más caros por línea
154 SELECT *
155 FROM (

```

| | CodProveedor | MontoProveedor | Porcentaje |
|----|--------------|----------------|------------|
| 1 | 1 | 31619.00 | 9.89 |
| 2 | 2 | 3360.00 | 8.87 |
| 3 | 3 | 14118.00 | 4.86 |
| 4 | 4 | 18980.00 | 5.46 |
| 5 | 5 | 2682.00 | 8.77 |
| 6 | 6 | 4222.00 | 1.21 |
| 7 | 7 | 2492.00 | 8.72 |
| 8 | 8 | 3332.00 | 8.96 |
| 9 | 9 | 10000.00 | 2.90 |
| 10 | 10 | 3036.00 | 1.13 |
| 11 | 11 | 2280.00 | 8.66 |
| 12 | 12 | 1875.00 | 8.54 |
| 13 | 14 | 2704.00 | 8.78 |

Activar Windows
Ve a Configuración para activar Windows.

Ln 147, Col 1 (258 selected) | Spaces: 4 | UTF-8 | CRLF | 77 rows | MSSQL | 00.00.00 | sql1.database.windows.net | ChetuPERU (70) | Q

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;

```

```

153 -- Consulta 28: Mostrar solo los 3 artículos más caros por línea
154 SELECT *
155 FROM (
156     SELECT CodLinea, CodArticulo, PrecioProveedor,
157            ROW_NUMBER() OVER (PARTITION BY CodLinea ORDER BY PrecioProveedor DESC) AS Pos
158     FROM ARTICULO
159 ) X
160 WHERE X.Pos <= 3;
161
162 -- Consulta 29: Mostrar transportista y su DenseRank por TotalEnviado
163 SELECT CodTransportista, SUM(GD.CantidadEnviada) AS TotalEnviado,
164        DENSE_RANK() OVER (ORDER BY SUM(GD.CantidadEnviada) DESC) AS Posicion
165 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 |

Activar Windows
Ve a Configuración

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;

```

```

161
162 -- Consulta 29: Mostrar transportista y su DenseRank por TotalEnviado
163 SELECT CodTransportista, SUM(GD.CantidadEnviada) AS TotalEnviado,
164        DENSE_RANK() OVER (ORDER BY SUM(GD.CantidadEnviada) DESC) AS Posicion
165 FROM GUIA_ENVIO GE
166 JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
167 GROUP BY CodTransportista;
168
169 -- Consulta 30: Mostrar por guía la suma acumulada por tienda hasta esa guía
170 SELECT CodTienda, NumGuia,
171        SUM(CantidadEnviada * PrecioVenta) AS TotalGuia

```

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 |
| 13 | 35 | 117 | 13 |

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:

```

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;

```

SQLQuery_1 - (76) z...min123) s

Run Cancel Disconnect Change Database: QhatuPERU Estimated Plan

Enable Actual Plan Parse Enable SQLCMD To Notebook

```

165 FROM GUIA_ENVIO GE
166 JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
167 GROUP BY CodTransportista;
168
169 -- Consulta 30: Mostrar por guía la suma acumulada por tienda hasta esa guía
170
171 SELECT GE.CodTienda, GE.NumGuia,
172        SUM(GD.CantidadEnviada * GD.PrecioVenta) AS TotalGuia,
173        SUM(SUM(GD.CantidadEnviada * GD.PrecioVenta)) OVER (PARTITION BY GE.CodTienda
174 FROM GUIA_ENVIO GE
175 JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
176 GROUP BY GE.CodTienda, GE.NumGuia;
177

```

Results Messages

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

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;
```


The screenshot shows a SQL Server Enterprise Manager interface. The top toolbar includes buttons for Run, Cancel, Disconnect, Change, and Database. The Database dropdown is set to 'QhatuPERU'. Below the toolbar, there are checkboxes for 'Enable Actual Plan', 'Parse', 'Enable SQLCMD', and 'To Notebook'. The main area displays a SQL query for 'Consulta 31: Mostrar Fecha y columnas CodTienda_1, CodTienda_2, con TotalEnviado'. The query uses a PIVOT function to transform data from rows to columns based on 'CodTienda' values (1, 2, 3). The results pane at the bottom shows a table with 10 rows and 5 columns: Fecha, Tienda_1, Tienda_2, and Tienda_3. The data shows various dates and quantities for each store.

```

179 -- Consulta 31: Mostrar Fecha y columnas CodTienda_1, CodTienda_2, con TotalEnviado
180
181 SELECT Fecha,
182        COALESCE([1], 0) AS Tienda_1,
183        COALESCE([2], 0) AS Tienda_2,
184        COALESCE([3], 0) AS Tienda_3
185 FROM (
186        SELECT CAST(GE.FechaSalida AS DATE) AS Fecha, GE.CodTienda, COALESCE(GD.CantidadEnviada, 0) AS CantidadEnviada
187        FROM GUIA_ENVIO GE
188        JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
189 ) AS SourceTable
190 PIVOT (
191        SUM(CantidadEnviada)
192        FOR CodTienda IN ([1], [2], [3])
193 ) AS PVT;
194

```

| | Fecha | Tienda_1 | Tienda_2 | Tienda_3 |
|----|------------|----------|----------|----------|
| 1 | 2023-01-20 | 150 | 0 | 0 |
| 2 | 2023-01-27 | 0 | 190 | 0 |
| 3 | 2023-02-10 | 0 | 0 | 40 |
| 4 | 2023-02-17 | 0 | 0 | 0 |
| 5 | 2023-03-05 | 0 | 0 | 0 |
| 6 | 2023-03-15 | 0 | 0 | 0 |
| 7 | 2023-03-23 | 0 | 0 | 0 |
| 8 | 2023-04-07 | 0 | 0 | 0 |
| 9 | 2023-04-19 | 0 | 0 | 0 |
| 10 | 2023-04-30 | 0 | 0 | 0 |

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;

```

```

197  -- Consulta 32: Mostrar CodArtículo y columnas con cantidades por tienda 1.
198  SELECT *
199  FROM (
200      SELECT GD.CodArtículo, GE.CodTienda, GD.CantidadEnviada
201      FROM GUIA_DETALLE GD
202      JOIN GUIA_ENVIO GE ON GD.NumGuia = GE.NumGuia
203  ) SRC
204  PIVOT (SUM(CantidadEnviada) FOR CodTienda IN ([1],[2],[3])) AS PVT;
205
206  -- Consulta 33: Mostrar AñoMes y tiendas como columnas con suma de PrecioVe
207  SELECT *
208  FROM (
209      SELECT FORMAT(GE.FechaSalida, 'yyyy-MM') AS AñoMes, GE.CodTienda,
210             GD.PrecioVenta * GD.CantidadEnviada AS Monto
211      FROM GUIA_ENVIO GE
212      JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
213  ) SRC
214  PIVOT (SUM(Monto) FOR CodTienda IN ([1],[2],[3])) AS PVT;
215
216  -- Consulta 34: Mostrar CodArtículo con columnas para cada Estado

```

Results

Messages

| | | | |
|-------------|---|---|---|
| CodArtículo | 1 | 2 | 3 |
|-------------|---|---|---|

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;

```

205

206 -- Consulta 33: Mostrar AñoMes y tiendas como columnas con suma de PrecioVen

207 SELECT *

208 FROM (

209 SELECT FORMAT(GE.FechaSalida, 'yyyy-MM') AS AñoMes, GE.CodTienda,

210 GD.PrecioVenta * GD.CantidadEnviada AS Monto

211 FROM GUIA_ENVIO GE

212 JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia

213) SRC

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

215

216 -- Consulta 34: Mostrar CodArtículo con columnas para cada Estado

217 SELECT *

218 FROM (

219 SELECT OD.CodArtículo, OD.Estado, OD.CantidadSolicitada

Results Messages

| | AñoMes ▾ | 1 ▾ | 2 ▾ | 3 ▾ |
|---|----------|---------|--------|------|
| 1 | 2023-01 | 1595.00 | 750.00 | NULL |

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.CodArtículo,
        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.CodArtículo, 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],[Cancelado])

-- 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.CodArtículo, GD.CantidadEnviada
    FROM CITA_SANTO GE
    JOIN DETALLE_DETALLE GD ON GE.CodArtículo = GD.CodArtículo
) SRC
PIVOT (SUM(CantidadEnviada) FOR CodTienda IN (' + @cols + ')) SRC
GROUP BY CodArtículo, CodTienda;
'
```

Messages

| | | | |
|-------------|-----------|-----------|-----------|
| CodArtículo | 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;
```

☒ Enable Actual Plan
 ☒ Parse
 ☒ Enable SQLCMD
 ☐ To Notebook

```

223 PIVOT (SUM(CantidadSolicitada) FOR Estado IN ([Pendiente],[Entregado],[Cancelado]))
224 -- Consulta 35: Contar artículos por presentación y proveedor
225 SELECT A.Presentacion, A.CodProveedor, COUNT(*) AS TotalArticulos
226 FROM ARTICULO A
227 GROUP BY A.Presentacion, A.CodProveedor;
228
229 -- Consulta 36: Generar PIVOT dinámico para todas las tiendas
230 DECLARE @cols NVARCHAR(MAX), @sql NVARCHAR(MAX);
231 SELECT @cols = STRING_AGG(QUOTENAME(CodTienda), ',') FROM TIENDA;
232 SET @sql = '
233 SELECT * FROM (
234     SELECT GE.CodTienda, GD.CodArticulo, GD.CantidadEnviada
235     FROM GUIA_ENVIO GE
236     JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
237 ) SRC
238 PIVOT (SUM(CantidadEnviada) FOR CodTienda IN (' + @cols + ')) AS P;';
239 EXEC sp_executesql @sql;
240
241 -- Consulta 37: Mostrar meses y columnas por transportista con totales enviados
242 SELECT *
243 FROM (
244     SELECT FORMAT(GE.FechaSalida, 'yyyy-MM') AS Mes, GE.CodTransportista, GD.Cantidad
  
```

Results Messages

| | Presentacion | CodProveedor | TotalArticulos |
|---|---------------|--------------|----------------|
| 1 | Bolsa 1kg | 1 | 3 |
| 2 | Bolsa 5kg | 1 | 1 |
| 3 | Botella 1L | 1 | 1 |
| 4 | Botella 900ml | 1 | 1 |

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:

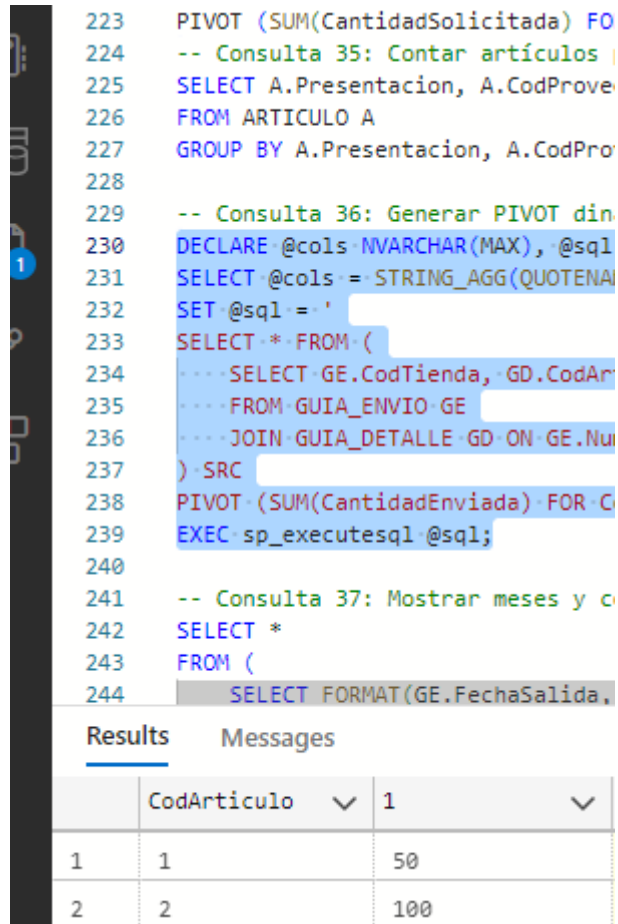
```

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;

```



The screenshot shows a SQL query window with the following code:

```

223 PIVOT (SUM(CantidadSolicitada) FO
224 -- Consulta 35: Contar artículos |
225 SELECT A.Presentacion, A.CodProve
226 FROM ARTICULO A
227 GROUP BY A.Presentacion, A.CodPro
228
229 -- Consulta 36: Generar PIVOT din
230 DECLARE @cols NVARCHAR(MAX), @sql
231 SELECT @cols = STRING_AGG(QUOTENA
232 SET @sql = '
233 SELECT * FROM (
234     SELECT GE.CodTienda, GD.CodAr
235     FROM GUIA_ENVIO GE
236     JOIN GUIA_DETALLE GD ON GE.Nu
237 ) SRC
238 PIVOT (SUM(CantidadEnviada) FOR C
239 EXEC sp_executesql @sql;
240
241 -- Consulta 37: Mostrar meses y c
242 SELECT *
243 FROM (
244     SELECT FORMAT(GE.FechaSalida,

```

Below the query window, the 'Results' tab is active, showing a table with the following data:

| | CodArticulo | 1 |
|---|-------------|-----|
| 1 | 1 | 50 |
| 2 | 2 | 100 |

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.

```

239 EXEC sp_executesql @sql;
240
241 -- Consulta 37: Mostrar meses y columnas por
242 SELECT *
243 FROM (
244     SELECT FORMAT(GE.FechaSalida, 'yyyy-MM')
245     FROM GUIA_ENVIO GE
246     JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.
247 ) SRC
248 PIVOT (SUM(CantidadEnviada) FOR CodTransport
249
250 -- Consulta 38: Contar proveedores por rango
251 SELECT *
252 FROM (
253     SELECT P.NomProveedor,
254     CASE
255         WHEN COUNT(A.CodArticulo) < 5
256         WHEN COUNT(A.CodArticulo) BET

```

Results Messages

| | Mes | 1 | 2 |
|---|---------|-----|-----|
| 1 | 2023-01 | 150 | 190 |

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;

```

Run Cancel Disconnect Change Database: QhātuPERU Estimated Plan

Enable Actual Plan Parse Enable SQLCMD To Notebook

```

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;
270
271 -- Consulta 40: Mostrar Mes y columnas por tienda (CASE alternativo)
272 SELECT FORMAT(GE.FechaSalida,'yyyy-MM') AS Mes,
273        SUM(CASE WHEN CodTienda = 1 THEN GD.CantidadEnviada END) AS Tienda1,
274        SUM(CASE WHEN CodTienda = 2 THEN GD.CantidadEnviada END) AS Tienda2,
275        SUM(CASE WHEN CodTienda = 3 THEN GD.CantidadEnviada END) AS Tienda3
276 FROM GUIA_ENVIO GE
277 JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
278 GROUP BY FORMAT(GE.FechaSalida,'yyyy-MM');
279
280 -- Consulta 41: Mostrar CodLínea y CantArtículos donde CantArtículos > 10
281 SELECT A.CodLinea, COUNT(A.CodArticulo) AS CantArticulos

```

Results Messages

| CodArticulo | CodTienda | VolumenTotal |
|-------------|-----------|--------------|
| 1 | 1 | 50 |
| 2 | 1 | 100 |
| 3 | 2 | 150 |
| 4 | 2 | 40 |
| 5 | 3 | 25 |
| 6 | 3 | 15 |

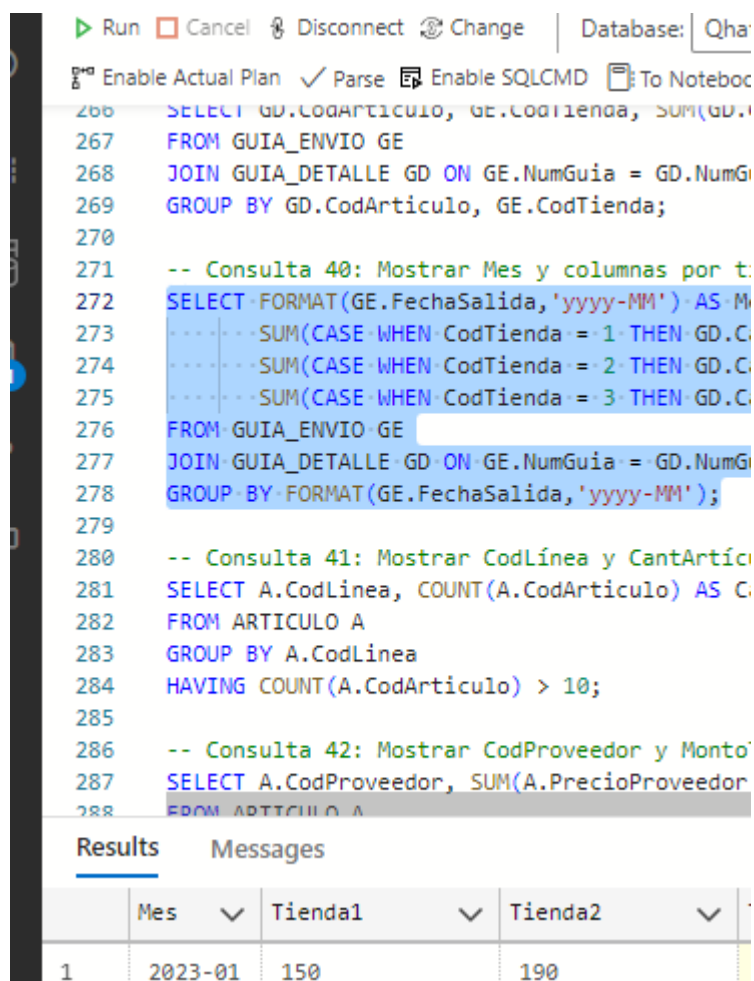
Explicación: Muestra el volumen de artículos enviados a cada tienda.

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');
```



Run Cancel Disconnect Change Database: Qha

Enable Actual Plan Parse Enable SQLCMD To Notebook

```
266 SELECT GD.CodArticulo, GE.CodTienda, SUM(GD.C
267 FROM GUIA_ENVIO GE
268 JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGu
269 GROUP BY GD.CodArticulo, GE.CodTienda;
270
271 -- Consulta 40: Mostrar Mes y columnas por t
272 SELECT FORMAT(GE.FechaSalida, 'yyyy-MM') AS M
273     SUM(CASE WHEN CodTienda = 1 THEN GD.C
274     SUM(CASE WHEN CodTienda = 2 THEN GD.C
275     SUM(CASE WHEN CodTienda = 3 THEN GD.C
276 FROM GUIA_ENVIO GE
277 JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGu
278 GROUP BY FORMAT(GE.FechaSalida, 'yyyy-MM');
279
280 -- Consulta 41: Mostrar CodLínea y CantArtícu
281 SELECT A.CodLinea, COUNT(A.CodArticulo) AS C
282 FROM ARTICULO A
283 GROUP BY A.CodLinea
284 HAVING COUNT(A.CodArticulo) > 10;
285
286 -- Consulta 42: Mostrar CodProveedor y Monto
287 SELECT A.CodProveedor, SUM(A.PrecioProveedor
288 FROM ARTICULO A
```

Results Messages

| | Mes | Tienda1 | Tienda2 |
|---|---------|---------|---------|
| 1 | 2023-01 | 150 | 190 |

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

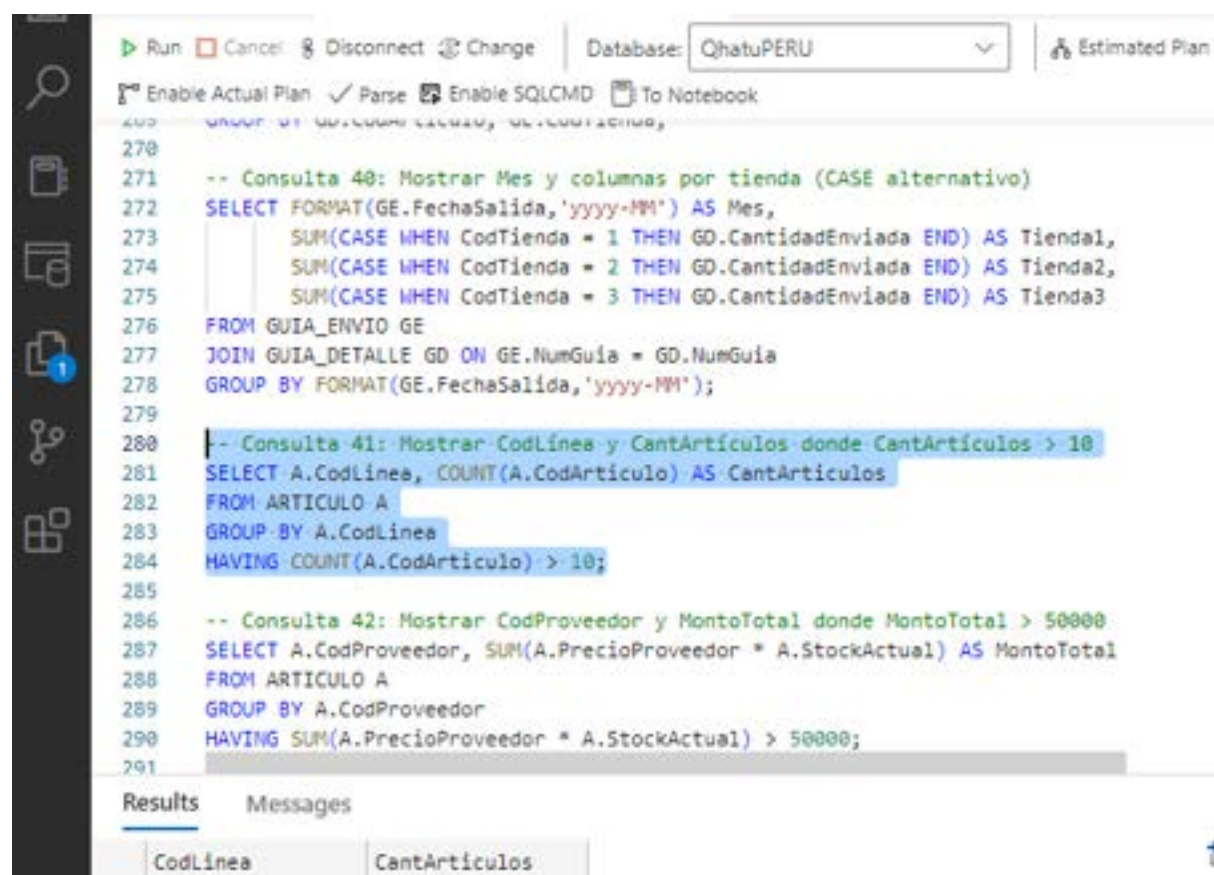
CLÁUSULA HAVING

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;
```

```
281 SELECT A.CodLinea, COUNT(A.CodArticulo) AS CantArticulos
282 FROM ARTICULO A
283 GROUP BY A.CodLinea
284 HAVING COUNT(A.CodArticulo) > 10;
285
286 -- Consulta 42: Mostrar CodProveedor y MontoTotal donde MontoTotal > 50000
287 SELECT A.CodProveedor, SUM(A.PrecioProveedor * A.StockActual) AS MontoTotal
288 FROM ARTICULO A
289 GROUP BY A.CodProveedor
290 HAVING SUM(A.PrecioProveedor * A.StockActual) > 50000;
291
292 -- Consulta 43: Mostrar CodTienda y PromedioGuía donde PromedioGuía > 1000
293 SELECT GE.CodTienda, AVG(GD.CantidadEnviada * GD.PrecioVenta) AS PromedioGuía
294 FROM GUIA_ENVIO GE
295 JOIN GUIA_DETALLE GD ON GE.NumGuía = GD.NumGuía
296 GROUP BY GE.CodTienda
297 HAVING AVG(GD.CantidadEnviada * GD.PrecioVenta) > 1000;
```

Results Messages

| CodProveedor | MontoTotal |
|--------------|------------|
| | |

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

Consulta 43

Enunciado: Mostrar CodTienda y PromedioGuía donde PromedioGuía > 1000.

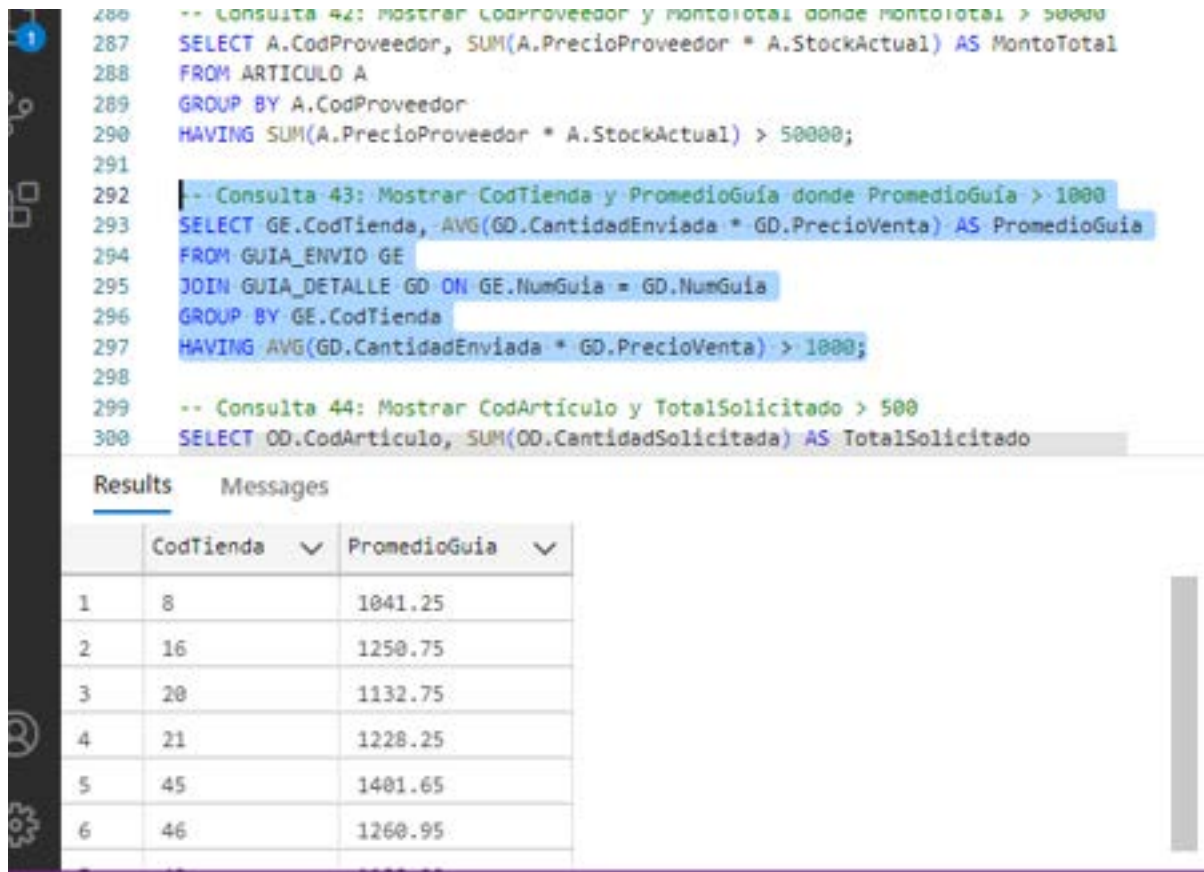
Consulta SQL:

```
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;

```



```

286 -- Consulta 42: Mostrar CodProveedor y MontoTotal donde MontoTotal > 50000
287 SELECT A.CodProveedor, SUM(A.PrecioProveedor * A.StockActual) AS MontoTotal
288 FROM ARTICULO A
289 GROUP BY A.CodProveedor
290 HAVING SUM(A.PrecioProveedor * A.StockActual) > 50000;
291
292 -- Consulta 43: Mostrar CodTienda y PromedioGuia donde PromedioGuia > 1000
293 SELECT GE.CodTienda, AVG(GD.CantidadEnviada * GD.PrecioVenta) AS PromedioGuia
294 FROM GUIA_ENVIO GE
295 JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
296 GROUP BY GE.CodTienda
297 HAVING AVG(GD.CantidadEnviada * GD.PrecioVenta) > 1000;
298
299 -- Consulta 44: Mostrar CodArtículo y TotalSolicitado > 500
300 SELECT OD.CodArtículo, SUM(OD.CantidadSolicitada) AS TotalSolicitado

```

| | CodTienda | PromedioGuia |
|---|-----------|--------------|
| 1 | 8 | 1041.25 |
| 2 | 16 | 1250.75 |
| 3 | 20 | 1132.75 |
| 4 | 21 | 1228.25 |
| 5 | 45 | 1401.65 |
| 6 | 46 | 1260.95 |

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

Consulta 44

Enunciado: Mostrar CodArtículo y TotalSolicitado > 500.

Consulta SQL:

```

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

```

HAVING SUM(OD.CantidadSolicitada) > 500;

```
296 GROUP BY GE.CodTienda
297 HAVING AVG(GD.CantidadEnviada * GD.PrecioVenta) > 1000;
298
299 -- Consulta 44: Mostrar CodArtículo y TotalSolicitado > 500
300 SELECT OD.CodArtículo, SUM(OD.CantidadSolicitada) AS TotalSolicitado
301 FROM ORDEN_DETALLE OD
302 GROUP BY OD.CodArtículo
303 HAVING SUM(OD.CantidadSolicitada) > 500;
304
305 -- Consulta 45: Mostrar CodTransportista y CantGuías >= 5
306 SELECT CodTransportista, COUNT(NumGuia) AS CantGuías
307 FROM GUIA_ENVIO
308 GROUP BY CodTransportista
309 HAVING COUNT(NumGuia) >= 5;
310
311 -- Consulta 46: Mostrar líneas donde SUM(StockActual) < SUM(StockMinimo)
312 SELECT CodLinea, SUM(StockActual) AS StockTotal, SUM(StockMinimo) AS StockMinimo
313 FROM ARTICULO
```

Results Messages

| CodArtículo | TotalSolicitado |
|-------------|-----------------|
|-------------|-----------------|

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 CantGuías
FROM GUIA_ENVIO
GROUP BY CodTransportista
HAVING COUNT(NumGuia) >= 5;
```

```

302 GROUP BY OD.CodArticulo
303 HAVING SUM(OD.CantidadSolicitada) > 500;
304
305 -- Consulta 45: Mostrar CodTransportista y CantGuías >= 5
306 SELECT CodTransportista, COUNT(NumGuia) AS CantGuías
307 FROM GUIA_ENVIO
308 GROUP BY CodTransportista
309 HAVING COUNT(NumGuia) >= 5;
310
311 -- Consulta 46: Mostrar líneas donde SUM(StockActual) < SUM(StockMinimo)
312 SELECT CodLinea, SUM(StockActual) AS StockTotal, SUM(StockMinimo) AS StockMinimo
313 FROM ARTICULO
314 GROUP BY CodLinea
315 HAVING SUM(StockActual) < SUM(StockMinimo);
316
317 -- Consulta 47: Mostrar proveedores donde MAX(PrecioProveedor) > 100
318 SELECT CodProveedor, MAX(PrecioProveedor) AS PrecioMaximo
319 FROM ARTICULO

```

Results Messages

| CodTransportista | CantGuías |
|------------------|-----------|
|------------------|-----------|

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);

```

```

307 FROM GUIA_ENVIO
308 GROUP BY CodTransportista
309 HAVING COUNT(NumGuia) >= 5;
310
311 -- Consulta 46: Mostrar líneas donde SUM(StockActual) < SUM(StockMinimo)
312 SELECT CodLinea, SUM(StockActual) AS StockTotal, SUM(StockMinimo) AS StockMinimo
313 FROM ARTICULO
314 GROUP BY CodLinea
315 HAVING SUM(StockActual) < SUM(StockMinimo);
316
317 -- Consulta 47: Mostrar proveedores donde MAX(PrecioProveedor) > 100
318 SELECT CodProveedor, MAX(PrecioProveedor) AS PrecioMaximo
319 FROM ARTICULO

```

Results Messages

| | CodLinea | StockTotal | StockMinimo |
|---|----------|------------|-------------|
| 1 | 1 | 450 | 50 |
| 2 | 2 | 800 | 100 |
| 3 | 3 | 1200 | 150 |
| 4 | 4 | 350 | 40 |
| 5 | 5 | 200 | 30 |
| 6 | 6 | 120 | 20 |

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;

```



```

315     HAVING SUM(StockActual) > SUM(StockMinimo);
316
317 -- Consulta 47: Mostrar proveedores donde MAX(PrecioProveedor) > 100
318 SELECT CodProveedor, MAX(PrecioProveedor) AS PrecioMaximo
319 FROM ARTICULO
320 GROUP BY CodProveedor
321 HAVING MAX(PrecioProveedor) > 100;
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 CodLinea donde MAX(Precio) - MIN(Precio) > 20
331 SELECT CodLinea, MAX(PrecioProveedor) - MIN(PrecioProveedor) AS RangoPrecio

```

Results Messages

| | CodProveedor | PrecioMaximo |
|---|--------------|--------------|
| 1 | 49 | 159.00 |

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 CodLínea, MAX(Precioproveedor) - MIN(Precioproveedor) AS RangoPrecio |
| 332 | FROM ARTICULO |
| 333 | GROUP BY CodLínea |
| 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 |
| 338 | FROM ARTICULO |

| Results | | Messages | |
|---------|-----------|---------------|----------|
| | CodTienda | PromedioEnvio | NumGuías |
| 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
    CodLínea,
    MAX(PrecioVenta) - MIN(PrecioVenta) AS RangoPrecio
FROM ARTICULO
GROUP BY CodLínea
HAVING MAX(PrecioVenta) - MIN(PrecioVenta) > 20;
```

```
327 -- Consultas 48 y 49: Mostrar CodLinea donde MAX(Precio) - MIN(Precio) < 20
328 HAVING AVG(GD.CantidadEnviada) > 50 AND COUNT(GE.NumGuia) <= 10;
329
330 -- Consulta 49: Mostrar CodLinea 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 Pr
338 FROM ARTICULO
339 GROUP BY CodProveedor
340 HAVING AVG(StockActual) < 20 AND COUNT(CodArticulo) > 5;
341
342 go
```

Results

Messages

| CodLinea | RangoPrecio |
|----------|-------------|
|----------|-------------|

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.