

**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 a SQL Server Management Studio window with the following details:

- Title Bar:** zero1.database.windows.net:QhatuPERU - SQLQuery\_2 - (85) z\_min123
- Toolbar:** Run, Cancel, Disconnect, Change, Database: QhatuPERU, Estimated Plan.
- Text Editor:** A query window containing the following SQL code:

```
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
```
- Results Grid:** A table titled "Results" showing the output of the query. The columns are CodArticulo, DescripcionArticulo, and ValorInventario. The data is as follows:

	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 Bimbo 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	Melado Laiive 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	Fideos Don Vittorio	2332.00

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

TotalInventario
347657.50

**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;
```

The screenshot shows the SSMS interface. The Object Explorer on the left lists database objects like CodProveedor, DescripcionArticulo, PrecioProveedor, StockActual, StockMinimo, StockMaximo, and others. The central pane contains three queries:

- Query1:** `SELECT CAST(SUM(A.StockActual * CAST(A.PrecioProveedor AS DECIMAL(18,2))) AS DECIMAL(18,2)) AS TotalInventario FROM ARTICULO A;`
- Query2:** `-- 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;`
- Query3:** `-- Consulta 4: Contar articulos discontinuos
SELECT COUNT(*) AS articulosDiscontinuos
FROM ARTICULO;`

The results grid under the second query shows the following data:

CodLinea	PrecioPromedio
1	12.50
2	4.50
3	1.50
4	6.00
5	6.00
6	15.00
7	4.50
8	1.50
9	14.00
10	3.00
11	2.50
12	3.00
13	11.50
14	8.00

**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;
```

The screenshot shows the SSMS interface with the following details:

- Top Bar:** Welcome, SQLQuery\_1 - (87) z\_min123, zero1.database.windows.net\QhatuPERU, SQL.
- Toolbar:** Run, Cancel, Disconnect, Change, Database: QhatuPERU, Estimated Plan, Enable Actual.
- Text Editor:** Contains the SQL code for Query 4, followed by a comment for Query 5 and its code.
- Results Tab:** Shows the output of the query: ArticulosDescontinuados (0).

**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: QchatUPERU 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 ValorTotal
28: FROM GUIA_DETALLE GD
29: GROUP BY GD.NumGuia;
30:
31: -- Consulta 7: Para cada Artículo 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.CodArticulo, SUM(OD.CantidadSolicitada) AS TotalSolicitado
33  FROM ORDEN_DETALLE OD
34  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	1876.60
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.CodArticulo,
    SUM(OD.CantidadSolicitada) AS TotalSolicitado
FROM ORDEN_DETALLE OD
GROUP BY OD.CodArticulo;

```

```

Run Cancel Disconnect Change Database ChatUPERU Estimated Plan Enable Actual Plan Parse Enable SQLCMD T
26 SELECT OD.NumGuia,
27      CAST(SUM(OD.CantidadEnviada * CAST(OD.PrecioVenta AS DECIMAL(18,2))) AS DECIMAL(18,2)) AS ValorTotal
28 FROM GUIA_DETALLE OD
29 GROUP BY OD.NumGuia;
30
31 -- Consulta 7: Para cada CodArticulo, mostrar TotalSolicitado
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

```

**Results** Messages

CodArticulo	TotalSolicitado
1	100
2	200
3	300
4	80
5	50
6	30
7	70
8	90
9	45
10	150
11	200
12	120
13	60
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.CodArticulo,
    COUNT(DISTINCT OD.NumOrden) AS OrdenesUnicas
FROM ORDEN_DETALLE OD
GROUP BY OD.CodArticulo;

```

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 PromedioDias
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 the SSMS interface with two queries in the query window:

```
zero1.database.windows.net:QhatuPERU          SQLQuery_2 - (85) z...min123 9+ ...
Run Cancel Disconnect Change Database: QhatuPERU
Enable Actual Plan Parse Enable SQLCMD To Notebook
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(18,2)) AS Promedio
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;
```

The Results tab displays the output of the second query:

PromedioDias
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 a SQL Server Management Studio window with the following details:

- Connection:** zero1.database.windows.net:QhatuPERU
- Query ID:** SQLQuery\_2 - (85) z...min123
- Database:** QhatuPERU
- Buttons:** Run, Cancel, Disconnect, Change, Database dropdown, Estimated Plan, Enable Actual Plan, Parse, Enable SQLCMD, To Notebook.
- Code (Listed Lines):**

```
42 SELECT CAST(AVG(DATEDIFF(DAY, O.FechaIngreso, GETDATE())) AS DECIMAL(10,2)) AS Prc  
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.NomLinea, COUNT(A.CodArticulo) AS CantArticulos
```
- Results Tab:** Shows a table with the following data:

	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	112

**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 NomLinea y CantArticulos
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**

	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 a SQL Server Management Studio (SSMS) window. At the top, there's a toolbar with buttons for Run, Cancel, Disconnect, Change, Database (set to QhatuPERU), and Estimated Plan. Below the toolbar, the database name is zero1.database.windows.net:QhatuPERU. The query window contains several numbered lines of SQL code. Lines 58 and 60 are highlighted in blue, indicating they are the ones being executed or selected.

```

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 OD.NumOrden, SUM(OD.PrecioCompra * OD.CantidadSolicitada) AS CostoTotal
64 FROM ORDEN_DETALLE_OD
65

```

The results pane shows a table with two columns: CodLinea and StockTotal. The data is 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;
```

The screenshot shows the SSMS interface with the following details:

- Query Editor:** The code is pasted into the query editor window.
- Results Tab:** The "Results" tab is selected, showing the output of the query.
- Table Output:** The results are presented in a table with two columns:
  - NumOrden:** Order ID (values from 1 to 14).
  - CostoTotal:** Total Cost (values ranging from 603.00 to 2871.00).
- Toolbar:** Standard SSMS toolbar with icons for Run, Cancel, Disconnect, Change, Database, Estimated Plan, and other options.
- Status Bar:** Shows the connection information (zero1.database.windows.net:QhatuPERU), the database name (QhatuPERU), and the number of rows affected (14).

**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;

```

```

-- Consulta 14: Mostrar NumGuía y PromedioEnviado
67  SELECT GD.NumGuia, AVG(GD.CantidadEnviada) AS PromedioEnviado
68  FROM GUIA_DETALLE GD
69  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;
```

The screenshot shows a SQL Server Management Studio (SSMS) interface. At the top, there's a connection bar with 'zero1.database.windows.net:QhatuPERU' and a status bar indicating 'SQLQuery\_2 - (85) z...min123) 9+' and an estimated plan icon. Below the status bar are several buttons: Run, Cancel, Disconnect, Change, Database dropdown set to 'QhatuPERU', Estimated Plan, Enable Actual Plan, Parse, Enable SQLCMD, and To Notebook.

The main area contains two queries:

```
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)
```

Below the queries, there are two tabs: 'Results' and 'Messages'. The 'Results' tab is selected and displays a table with the following data:

	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 Management Studio (SSMS) interface. In the top-left corner, the connection details are shown as zero1.database.windows.net:QhatuPERU. The top-right corner displays the title bar as SQLQuery\_2 - (85) z...min123 9+. Below the title bar, there are several buttons: Run, Cancel, Disconnect, Change, Database: QhatuPERU, Estimated Plan, Enable Actual Plan, Parse, Enable SQLCMD, and To Notebook. The main area contains a multi-line text editor with the following SQL code:

```
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)
81 ORDER BY Fecha;
82
83 -- Consulta 17: Sumar (CantidadEnviada × PrecioVenta) por CodTienda
84 SELECT GE.CodTienda,
85      CAST(SUM(COALESCE(GD.CantidadEnviada,0) * TRY_CONVERT(DECIMAL(19,4), GD.PrecioVe
86 FROM dbo.GUIA_ENVIO AS GE
```

Below the code editor, there are two tabs: Results and Messages. The Results tab is selected, showing a table with the following data:

	Fecha	NumOrdenes
1	2023-01-28	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

At the bottom of the results grid, there are several status indicators: A (All), 0 rows, UTF-8, CRLF, 98 rows, MSSQL, 00:00:00, zero1.database.windows.net:QhatuPERU (85), and a refresh icon.

**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 Management Studio (SSMS) window with the following details:

- Top Bar:** Shows the connection information "zero1.database.windows.net:QhatUPERU" and the query ID "SQLQuery\_2 - (85) z...min123) 9+".
- Toolbar:** Includes buttons for Run, Cancel, Disconnect, Change, Database dropdown set to "QhatUPERU", Estimated Plan, Enable Actual Plan, Parse, Enable SQLCMD, and To Notebook.
- Text Editor:** Displays the SQL code for Consulta 17, which calculates the total amount per store by multiplying quantity shipped by unit price and summing them up. It also includes a comment for Consulta 18.
- Results Grid:** A table titled "Results" showing the output of the query. The columns are "CodTienda" and "MontoTotalTienda". The data is as follows:

	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
);
```

The screenshot shows a SQL query being run in a database environment. The code is as follows:

```
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 NumeroOrdenes
79 FROM GUIA_ENVIO
80 GROUP BY CAST(FechaSalida AS DATE);
81 ORDER BY Fecha;
82
83 -- Consulta 17: Sumar (CantidadEnvuada * PrecioVenta) por CodTienda
84 SELECT GE.CodTienda,
85        CAST(SUM(COALESCE(GD.CantidadEnvuada,0) * TRY_CONVERT(DECIMAL(10,4), GD.PrecioVenta)) AS DECIMAL(10,4)) AS MontoTotalTienda
86 FROM dbo.GUIA_ENVIO AS GE
```

The results pane shows a table with three columns: CodArticulo, DescripcionArticulo, and StockActual. The table is currently empty, indicating no results have been returned yet.

**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 Iavis Clásicas	Bols

	CodProveedor	NomProveedor		CantArticulos	
1	1	Alicorp S.A.A.	Activar Windows	9	
2	2	Gloria S.A.	Ve a Configuración para activar Windows.		

**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 linea 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	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	1000	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	2098.00	13
14	1011	1969.00	14
15	1040	1928.50	15
16	1032	1845.50	16

Activar VI  
Ve a Configuración

**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);

```

```

121  -- Consulta 23: Mostrar TotalDia y AcumuladoVentas ordenado por fecha
122  SELECT CAST(FechaSalida AS DATE) AS Fecha,
123      SUM(GD.CantidadEnviada * GD.PrecioVenta) AS TotalDia,
124      SUM(SUM(GD.CantidadEnviada * GD.PrecioVenta)) OVER (ORDER BY CAST(FechaSalida AS DATE)) AS Acumulado
125  FROM GUIA_ENVIO GE
126  JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
127  GROUP BY CAST(FechaSalida AS DATE);
128
129  -- Consulta 24: Calcular promedio móvil para stock
130  SELECT CodArticulo, FechaEnvio
131      ,((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

```

#### 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	2882.50	9219.50
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	1876.60	12612.60
13	2023-06-06	1463.00	14075.60

Activar Windo  
Visita [Centro de ayuda](#)

**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;

```

```

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 articulos
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	250
3	3	3	1200	400
4	4	4	350	116
5	5	5	200	66
6	6	6	120	40
7	7	7	200	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	340	90

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	98	6.80	3.80
9	1	99	10.50	6.80

**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 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() AS DECIMAL(10,2)) AS Porcentaje

```

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;

```

```

149  -- Consulta 27: Mostrar MontoProveedor y PorcentajeDelTotal
150  SELECT CodProveedor,
151      SUM(PrecioProveedor * StockActual) AS MontoProveedor,
152      CAST(SUM(PrecioProveedor * StockActual) * 100.0 / SUM(SUM(PrecioProveedor * StockActual)) OVER() AS DECIMAL(18,2)) AS Porcentaje
153  FROM ARTICULO
154  GROUP BY CodProveedor;
155
156  -- Consulta 28: Mostrar solo los 3 artículos más caros por línea
157  SELECT *
158  FROM (
159

```

**Results** Messages

CodProveedor	MontoProveedor	Porcentaje
1	31619.00	9.09
2	3160.00	0.93
3	14118.00	4.06
4	18988.00	5.46
5	2682.00	0.77
6	4222.00	1.23
7	2492.00	0.72
8	3332.00	0.96
9	10008.00	2.98
10	3936.00	1.13
11	2280.00	0.66
12	1875.00	0.54
13	2704.00	0.78

Activar Windows  
Ve a Configuración para activar Windows.

(en 147, Col 1 (256 selected)) Spaces: 4 UTF-8 CR/LF 77 rows MSSQL 00:00:00 jenniferdatabase.windows.net : OcharuPRU (76)

**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(GD.CantidadEnviada * GD.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	25	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;

```

The screenshot shows the SSMS interface with a query window containing T-SQL code and a results grid below it.

**Query Window (SQLQuery\_1 - (76) z...min123) :**

```
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 GE.CodTienda, GE.NumGuia,  
       SUM(GD.CantidadEnviada * GD.PrecioVenta) AS TotalGuia,  
       SUM(SUM(GD.CantidadEnviada * GD.PrecioVenta)) OVER (PARTITION BY GE.CodTienda  
FROM GUIA_ENVIO GE  
JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia  
GROUP BY GE.CodTienda, GE.NumGuia;
```

**Results Grid:**

	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 Management Studio (SSMS) interface. In the top-left corner, there's a 'Welcome' button and a tab labeled 'SQLQuery\_1 - (76 z..min123)'. Below the tabs are buttons for 'Run', 'Cancel', 'Disconnect', 'Change', and dropdowns for 'Database' (set to 'QhatuPERU') and 'Estimated Plan'. There are also checkboxes for 'Enable Actual Plan', 'Parse', 'Enable SQLCMD', and 'To Notebook'. The main area contains a multi-line code editor with numbered lines 179 through 194. The code is a PIVOT query designed to show shipping dates and three store IDs as columns. The results grid below shows 10 rows of data with columns: Fecha, Tienda\_1, Tienda\_2, and Tienda\_3. All values in the grid are currently 0.

	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.CodArticulo, 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 PrecioVenta
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**

CodArticulo	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;

```

The screenshot shows a SQL query being run in SSMS. The code is as follows:

```

205 -- Consulta 33: Mostrar AñoMes y tiendas como columnas con suma de PrecioVen
206 SELECT *
207 FROM (
208     SELECT FORMAT(GE.FechaSalida, 'yyyy-MM') AS AñoMes, GE.CodTienda,
209     GD.PrecioVenta * GD.CantidadEnviada AS Monto
210     FROM GUIA_ENVIO GE
211     JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
212 ) SRC
213 PIVOT (SUM(Monto) FOR CodTienda IN ([1],[2],[3])) AS PVT;
214
215 -- Consulta 34: Mostrar CodArtículo con columnas para cada Estado
216 SELECT *
217 FROM (
218     SELECT OD.CodArticulo, OD.Estado, OD.CantidadSolicitada

```

The results pane shows the output of the query:

	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.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],[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.CodArticulo, GD.CantidadEnviada
    FROM CITA_ENVIO GE
    JOIN DETALLE_ENVIO GD ON GE.CodTienda = GD.CodTienda
) T
PIVOT SUM(CantidadEnviada) FOR CodTienda IN (' + @cols + ')
'
PRINT @sql
:Messages
```

**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.CantidadEnviada

```

**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:**

```

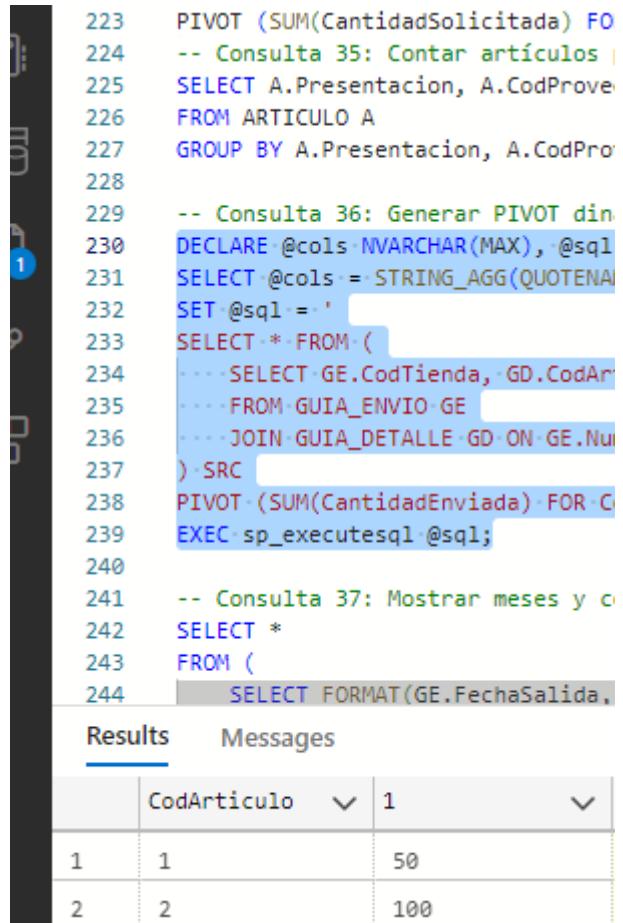
DECLARE @cols NVARCHAR(MAX), @sql NVARCHAR(MAX);
SELECT @cols = STRING_AGG(QUOTENAME(CodTienda), ',') FROM TIENDA;
SET @sql = '
SELECT * FROM (
    SELECT GE.CodTienda, GD.CodArticulo, GD.CantidadEnviada
    FROM GUIA_ENVIO GE
    JOIN GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
)
PIVOT (SUM(CantidadEnviada) FOR CodTienda IN (' + @cols + ')) AS P;

```

```

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

```



```

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.CodArt
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,

```

Results    Messages

	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 CodTransportista IN ([1],[2],[3]))
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) BETWEEN 5 AND 10
257          ...
258  )

```

Results    Messages

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

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;
```

```
Run Cancel Disconnect Change Database: QchatuPERU Estimated Plan  
Enable Actual Plan Parse Enable SQLCMD To Notebook  
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 CodLinea y CantArticulos donde CantArticulos > 10  
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
```

Results Messages

CodLinea	CantArticulos
----------	---------------

**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 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;
```

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 43: 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 CodArticulo y TotalSolicitado > 500
300  SELECT OD.CodArticulo, 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.CodArticulo,
    SUM(OD.CantidadSolicitada) AS TotalSolicitado
FROM ORDEN_DETALLE OD
GROUP BY OD.CodArticulo

```

```

HAVING SUM(OD.CantidadSolicitada) > 500;

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

```

Results    Messages

CodArticulo	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 CantGuias
FROM GUIA_ENVIO
GROUP BY CodTransportista
HAVING COUNT(NumGuia) >= 5;

```

The screenshot shows a SQL query editor interface. On the left, there are several icons: a file, a database, a table, a search, a refresh, and a help. A vertical toolbar on the far left has icons for back, forward, and search. The main area contains the following SQL code:

```
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 CantGuias
307 FROM GUIA_ENVIO
308 GROUP BY CodTransportista
309 HAVING COUNT(NumGuia) >= 5;
310
311 -- Consulta 46: Mostrar líneas donde SUM(StockActual) < SUM(StockMínimo)
312 SELECT CodLinea, SUM(StockActual) AS StockTotal, SUM(StockMinimo) AS StockMínimo
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
```

Below the code, there are tabs for "Results" and "Messages". Under "Results", there is a table header row with columns "CodTransportista" and "CantGuias".

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

## Consulta 46

**Enunciado:** Mostrar líneas donde  $\text{SUM}(\text{StockActual}) < \text{SUM}(\text{StockMínimo} * \text{NumArtículosPorLínea})$ .

**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
331 -- Consulta 49: Mostrar CodLinea donde MAX(Precio) - MIN(Precio) > 20
332 SELECT CodLinea, MAX(Precioproveedor) - MIN(Precioproveedor) AS RangoPrecio
333 FROM ARTICULO
334 GROUP BY CodLinea
335 HAVING MAX(Precioproveedor) - MIN(Precioproveedor) > 20;
336
337 -- Consulta 50: Mostrar CodProveedor con AVG(StockActual) < 20 y COUNT() > 5
338 SELECT CodProveedor, COUNT(CodArticulo) AS CantArticulos, AVG(StockActual) AS Promedios
339

```

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;

```

```
327 GROUP BY CodLinea
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 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 PromedioStock
338 FROM ARTICULO
339 GROUP BY CodProveedor
340 HAVING AVG(StockActual) < 2 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.