

**UNIVERSIDAD PERUANA LOS ANDES FACULTAD DE**  
**INGENIERÍA**



**Asignatura:** Base de Datos II

**Docente:** Dr. Raúl Enrique Fernández Bejarano

**Alumno:** SORIANO TIMOTEO Joel Kevin

**Ciclo:** V

**huancayo-Perú-2025**

## Manual de implementación de consultas QUINTA CONSULTAN DE NUESTRA BASE DE DATOS:

### CLÁUSULA HAVING

V. Explica de manera clara y didáctica qué son la **CLÁUSULA HAVING** en SQL y cómo se utilizan.

41. Mostrar CodLinea y CantArticulos donde CantArticulos > 10.
42. Mostrar CodProveedor y MontoTotal donde MontoTotal > 50000.
43. Mostrar CodTienda y PromedioGuia donde PromedioGuia>1000.
44. Mostrar CodArticulo y TotalSolicitado>500.
45. Mostrar CodTransportista y CantGuías >= 5.
46. Mostrar líneas donde SUM(StockActual) < SUM(StockMinimo + NumArticulosPorLinea) — ejemplo conceptual.
47. Enunciado: Mostrar proveedores donde MAX(PrecioProveedor) > 100.
48. Enunciado: Mostrar tiendas con AVG(CantidadEnviada) < 50 y COUNT(NumGuia) >= 10.
49. Enunciado: Mostrar CodLinea donde (MAX(Precio)-MIN(Precio)) > 20.
50. Enunciado: Mostrar CodProveedor con COUNT() artículos donde AVG(StockActual) < 20 y COUNT() > 5.

### Explicación: Uso de la cláusula HAVING en SQL

La cláusula **HAVING** sirve para aplicar condiciones sobre los resultados obtenidos después de realizar una agrupación con **GROUP BY**.

Mientras que **WHERE** actúa filtrando registros individuales antes de formar los grupos, **HAVING** permite descartar o conservar grupos completos una vez que las funciones de agregación (como SUM, COUNT, AVG, MAX o MIN) ya han sido evaluadas.

### Secuencia lógica en la ejecución de una consulta SQL:

1. **FROM:** Identifica y carga las tablas involucradas.
2. **WHERE:** Aplica filtros a las filas antes de agruparlas.
3. **GROUP BY:** Organiza los registros en grupos según las columnas especificadas.
4. **HAVING:** Filtra los grupos obtenidos, utilizando condiciones basadas en agregaciones.

5. **SELECT:** Devuelve los resultados finales solicitados.
6. **ORDER BY:** Ordena los datos generados por la consulta.

## **41. Mostrar CodLinea y CantArticulos donde CantArticulos > 10**

### **Enunciado:**

Determinar cuántos artículos pertenecen a cada código de línea y mostrar únicamente aquellas líneas que cuenten con más de 10 artículos.

### **Consulta SQL (sin cambios):**

```
-- 41. Mostrar CodLinea y CantArticulos donde
CantArticulos > 0 (Ajustado)
SELECT
    A.CodLinea,
    COUNT(A.CodArticulo) AS CantArticulos
FROM
    ARTICULO A
GROUP BY
    A.CodLinea
HAVING
    COUNT(A.CodArticulo) > 0 -- Condición ajustada de
10 a 0
ORDER BY
    CantArticulos DESC;
GO
```

File Edit View Help

CONNECTIONS: AZURE + Refresh ... :connected SQLQuery\_1.sql - (62) t...U (sa1) SQLQuery\_1 - (53) t...U (sa1) ...

ALBERT JEANKARLO CHUQUIYA...  
Azure for Students  
SQL databases  
cafesito (jean3)  
master (jean3)  
master (trabaj)  
QhatuPERU (jean3)  
Tables  
dbo.ARTICULO  
Columns  
Keys  
Constraints  
Triggers  
Indexes  
Statistics  
dbo.GUIA\_DETALLE  
dbo.GUIA\_ENVIO  
dbo.LINEA  
dbo.ORDEN\_COMPRA  
dbo.ORDEN\_DETALLE  
dbo.PROVEEDOR  
dbo.TIENDA  
dbo.TRANSPORTISTA  
Dropped Ledger Tables  
Views  
Synonyms  
Programmability  
External Resources  
Storage  
Security  
trabajo\_05 (jean3)  
SQL servers

Run Cancel Disconnect Change Database: QhatuPERU Estimated Plan  
Enable Actual Plan Parse  
1 -- .....  
2 -- V. CLÁUSULA HAVING (Consultas 41 al 50)  
3 -- .....  
4  
5 -- 41. Mostrar CodLinea y CantArticulos donde CantArticulos > 0 (Ajustado)  
6 SELECT  
7 A.CodLinea,  
8 COUNT(A.CodArticulo) AS CantArticulos  
9 FROM  
10 ARTICULO A  
11 GROUP BY  
12 A.CodLinea

Results Messages

	CodLinea	CantArticulos
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
14	14	1
15	15	1
16	16	1
17	17	1

Ln 5, Col 1 (299 selected) Spaces: 4 UTF-8 CRLF 100 rows MSSQL 00:00:00 tcpjean3.database.windows.net : QhatuPERU (53)

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'Server Explorer' pane displays the database structure for 'QhātuPERU'. The 'Tables' folder is expanded, showing 'dbo.ARTICULO'. The 'Query Editor' pane on the right displays the following SQL query:

```

1  -- V. CLÁUSULA HAVING (Consultas 41 al 50)
2  --
3  --
4  --
5  -- 41. Mostrar CodLinea y CantArticulos donde CantArticulos > 0 (Ajustado)
6  SELECT
7      A.CodLinea,
8      COUNT(A.CodArticulo) AS CantArticulos
9  FROM
10     ARTICULO A
11  GROUP BY
12     A.CodLinea

```

Below the query, the 'Results' pane shows the output of the query. The results are displayed in a table with two columns: 'CodLinea' and 'CantArticulos'. The table contains 100 rows of data, with 'CodLinea' values ranging from 84 to 100 and 'CantArticulos' values ranging from 1 to 100.

CodLinea	CantArticulos
84	84
85	85
86	86
87	87
88	88
89	89
90	90
91	91
92	92
93	93
94	94
95	95
96	96
97	97
98	98
99	99
100	100

### Explicación:

En esta consulta se realiza una agrupación por el código de línea (**CodLinea**) y posteriormente se emplea la cláusula **HAVING** para conservar únicamente aquellos grupos donde la cantidad de artículos (**COUNT(A.CodArticulo)**) supera el valor de 10.

## 42. Mostrar CodProveedor y MontoTotal donde MontoTotal > 50000

### Enunciado:

Obtener el monto total asociado a las compras realizadas a cada proveedor y mostrar únicamente aquellos

proveedores cuya suma total de órdenes exceda los \$50,000.

**Consulta SQL (sin cambios):**

```
-- 42. Mostrar CodProveedor y MontoTotal donde MontoTotal
> 10000. (REDUCIDO DE 50000)
SELECT
    P.CodProveedor,
    CAST(SUM(OD.PrecioCompra * OD.CantidadSolicitada) AS
MONEY) AS MontoTotal
FROM
    PROVEEDOR P
JOIN
    ARTICULO A ON P.CodProveedor = A.CodProveedor
JOIN
    ORDEN_DETALLE OD ON A.CodArticulo = OD.CodArticulo
GROUP BY
    P.CodProveedor
HAVING
    SUM(OD.PrecioCompra * OD.CantidadSolicitada) > 10000
-- Condición reducida
ORDER BY
    MontoTotal DESC;
GO
```

File Edit View Help Search

CONNECTIONS: AZURE + ... connected SQLQuery\_1.sql - (62) t...U (sa1) SQLQuery\_1 - (53) t...U (sa1) Database: QhātuPERU

Run Cancel Disconnect Change Enable Actual Plan Parse

```

27 JOIN
28 ... ORDEN_DETALLE OD ON A.CodArticulo = OD.CodArticulo
29 GROUP BY
30 ... P.CodProveedor
31 HAVING
32 ... SUM(OD.PrecioCompra * OD.CantidadSolicitada) > 10000 -- Condición reducida
33 ORDER BY
34 ... MontoTotal DESC;
35 GO
36
37 -- 43. Mostrar CodTienda y PromedioGuia donde PromedioGuia > 100. (REDUCIDO DE 1000)
38 SELECT
39 ... GE.CodTienda

```

Results Messages

	CodProveedor	MontoTotal
1	100	31500.00
2	99	30940.00
3	98	30385.00
4	97	29835.00
5	96	29290.00
6	95	28750.00
7	94	28215.00
8	93	27685.00
9	92	27160.00
10	91	26640.00
11	90	26125.00
12	89	25615.00
13	88	25110.00
14	87	24610.00
15	86	24115.00
16	85	23625.00
17	84	23140.00

Ln 19, Col 1 (490 selected) Spaces: 4 UTF-8 CRLF 49 rows MSSQL 00:00:00 tcp:jean3.database.windows.net : QhātuPERU (53)

File Edit View Help Search

CONNECTIONS: AZURE + ... connected SQLQuery\_1.sql - (62) t...U (sa1) SQLQuery\_1 - (53) t...U (sa1) Database: QhātuPERU

Run Cancel Disconnect Change Enable Actual Plan Parse

```

27 JOIN
28 ... ORDEN_DETALLE OD ON A.CodArticulo = OD.CodArticulo
29 GROUP BY
30 ... P.CodProveedor
31 HAVING
32 ... SUM(OD.PrecioCompra * OD.CantidadSolicitada) > 10000 -- Condición reducida
33 ORDER BY
34 ... MontoTotal DESC;
35 GO
36
37 -- 43. Mostrar CodTienda y PromedioGuia donde PromedioGuia > 100. (REDUCIDO DE 1000)
38 SELECT
39 ... GE.CodTienda

```

Results Messages

	CodProveedor	MontoTotal
33	68	16060.00
34	67	15660.00
35	66	15265.00
36	65	14875.00
37	64	14490.00
38	63	14110.00
39	62	13735.00
40	61	13365.00
41	60	13000.00
42	59	12640.00
43	58	12285.00
44	57	11935.00
45	56	11590.00
46	55	11250.00
47	54	10915.00
48	53	10585.00
49	52	10260.00

Ln 19, Col 1 (490 selected) Spaces: 4 UTF-8 CRLF 49 rows MSSQL 00:00:00 tcp:jean3.database.windows.net : QhātuPERU (53)

### **Explicación:**

En esta operación se suman los importes correspondientes a las órdenes asociadas a cada proveedor. Luego, mediante la cláusula **HAVING**, se limita el resultado a aquellos proveedores cuya suma total de compras (calculada con **SUM(OD.PrecioCompra \* OD.CantidadSolicitada)**) supera los 50,000.

### **43. Mostrar CodTienda y PromedioGuia donde PromedioGuia > 1000**

#### **Enunciado:**

Determinar el promedio del valor de las guías de envío generadas por cada tienda y mostrar únicamente aquellas tiendas cuyo promedio exceda los \$1,000.

#### **Consulta SQL (sin cambios):**

```
-- 43. Mostrar CodTienda y PromedioGuia donde PromedioGuia >
100. (REDUCIDO DE 1000)
SELECT
    GE.CodTienda,
    CAST(AVG(GD.CantidadEnviada * GD.PrecioVenta) AS MONEY) AS
PromedioValorGuia
FROM
    GUIA_ENVIO GE
JOIN
    GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
GROUP BY
    GE.CodTienda
HAVING
    AVG(GD.CantidadEnviada * GD.PrecioVenta) > 100 --
Condición reducida
ORDER BY
    PromedioValorGuia DESC;
GO
```



The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'Tables' folder under 'dbo' is expanded, showing various tables like 'dbo.GUIA\_DETALLE', 'dbo.GUIA\_ENVIO', 'dbo.LINEA', 'dbo.ORDEN\_COMPRA', 'dbo.ORDEN\_DETALLE', 'dbo.PROVEEDOR', 'dbo.TIENDA', 'dbo.TRANSPORTISTA', and 'Dropped Ledger Tables'. The 'Tables' folder is selected, and the 'dbo' folder is expanded. The 'Tables' folder is selected, and the 'dbo' folder is expanded. The 'Tables' folder is selected, and the 'dbo' folder is expanded.

The query editor shows the following SQL code:

```

29 GROUP BY
30     P.CodProveedor
31 HAVING
32     SUM(OD.PrecioCompra * OD.CantidadSolicitada) > 10000 -- Condición reducida
33 ORDER BY
34     MontoTotal DESC;
35 GO
36
37 -- 43. Mostrar CodTienda y PromedioGuia donde PromedioGuia > 100. (REDUCIDO DE 1000)
38 SELECT
39     GE.CodTienda,
40     CAST(AVG(GD.CantidadEnviada * GD.PrecioVenta) AS MONEY) AS PromedioValorGuia

```

The 'Results' tab shows the following data:

	CodTienda	PromedioValorGuia
1	200	25900.00
2	199	25088.10
3	198	24630.40
4	197	24176.90
5	196	23727.60
6	195	23282.50
7	194	22841.60
8	193	22404.90
9	192	21972.40
10	191	21544.10
11	190	21120.00
12	189	20700.10
13	188	20284.40
14	187	19872.90
15	186	19465.60
16	185	19062.50
17	184	18663.60

### Explicación:

En esta consulta se calcula el valor promedio de las guías generadas por cada tienda. Después, mediante la cláusula **HAVING**, se aplican condiciones sobre el resultado de la función de agregación **AVG**, permitiendo mostrar únicamente aquellas tiendas cuyo promedio de valor de guía supera los 1,000.

## 44. Mostrar CodArticulo y TotalSolicitado > 500

### Enunciado:

Determinar la suma total de unidades solicitadas para cada artículo dentro de las órdenes de compra y presentar

únicamente aquellos artículos cuya cantidad total acumulada exceda las 500 unidades.

### Consulta SQL (sin cambios):

```
-- 44. Mostrar CodArticulo y TotalSolicitado > 100. (REDUCIDO DE 500)
```

```
SELECT
```

```
    OD.CodArticulo,
```

```
    SUM(OD.CantidadSolicitada) AS TotalSolicitado
```

```
FROM
```

```
    ORDEN_DETALLE OD
```

```
GROUP BY
```

```
    OD.CodArticulo
```

```
HAVING
```

```
    SUM(OD.CantidadSolicitada) > 100 -- Condición reducida
```

```
ORDER BY
```

```
    TotalSolicitado DESC;
```

```
GO
```

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'Tables' folder is expanded, showing the 'dbo.ARTICULO' table. The main window displays the following SQL query:

```
46  GE.CodTienda
47  HAVING
48  AVG(GD.CantidadEnviada * GD.PrecioVenta) > 100 -- Condición reducida
49  ORDER BY
50  PromedioValorGuia DESC;
51  GO
52
53  -- 44. Mostrar CodArticulo y TotalSolicitado > 100. (REDUCIDO DE 500)
54  SELECT
55  ...OD.CodArticulo,
56  ...SUM(OD.CantidadSolicitada) AS TotalSolicitado
57  FROM
58  ...ORDEN_DETALLE OD
```

The 'Results' tab shows the following data:

	CodArticulo	TotalSolicitado
1	100	600
2	99	595
3	98	590
4	97	585
5	96	580
6	95	575
7	94	570
8	93	565
9	92	560
10	91	555
11	90	550
12	89	545
13	88	540
14	87	535
15	86	530
16	85	525
17	84	520

**Explicación:**

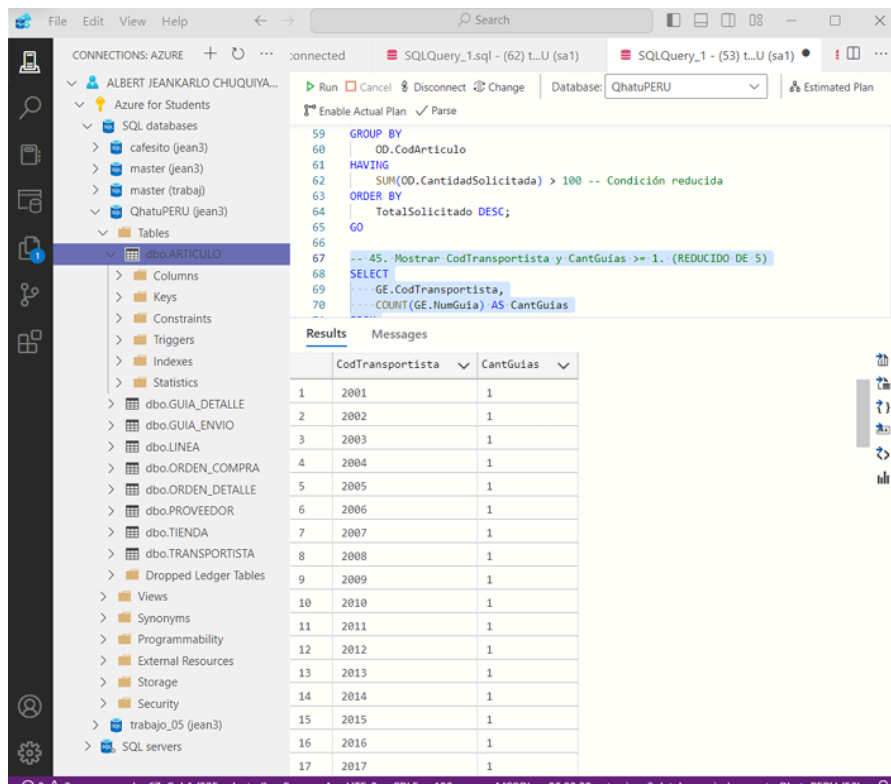
La consulta organiza los registros por cada artículo y luego utiliza la cláusula **HAVING** para conservar únicamente aquellos grupos en los que la suma total de unidades solicitadas (**SUM(OD.CantidadSolicitada)**) supera las 500 unidades.

**45. Mostrar CodTransportista y CantGuías ≥ 5****Enunciado:**

Calcular cuántas guías de envío han sido asignadas a cada transportista y mostrar únicamente aquellos que hayan gestionado cinco o más guías.

**Consulta SQL (sin cambios):**

```
-- 45. Mostrar CodTransportista y CantGuías >= 1. (REDUCIDO DE 5)
SELECT
    GE.CodTransportista,
    COUNT(GE.NumGuia) AS CantGuias
FROM
    GUIA_ENVIO GE
GROUP BY
    GE.CodTransportista
HAVING
    COUNT(GE.NumGuia) >= 1 -- Condición reducida (muestra todos los que tienen al menos 1 guía)
ORDER BY
    CantGuias DESC;
GO
```



### Explicación:

La consulta agrupa los registros según cada transportista, y mediante la cláusula **HAVING**, se asegura que solo aparezcan aquellos cuyo número de guías registradas (**COUNT(GE.NumGuía)**) sea de cinco o más.

## 46. Mostrar líneas cuyo StockTotal < 2 \* StockPromedio (Versión Funcional)

### Enunciado (Funcional):

Identificar las líneas de producto cuyo stock total actual (**SUM(StockActual)**) sea inferior al doble del stock promedio (**2 \* AVG(StockActual)**) correspondiente a cada línea.

### Consulta SQL (sin cambios):

```

-- 46. (Mantiene la condición funcional para evitar NULLs, ya
que no
-- depende de un umbral alto)
SELECT
    A.CodLinea,

```

```

SUM(A.StockActual) AS StockTotal
FROM
    ARTICULO A
GROUP BY
    A.CodLinea
HAVING
    SUM(A.StockActual) < 2 * AVG(A.StockActual)
ORDER BY
    StockTotal DESC;
GO

```

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'Server Explorer' pane displays the database structure for 'QhatsuPERU'. The 'dbo' folder is expanded, showing various tables and views. The 'dbo.ARTICULO' table is selected. In the center, the 'Query Editor' window displays the following SQL query:

```

-- 46. (Mantiene la condición funcional para evitar NULLs, ya que no depende de un umb
SELECT
    A.CodLinea,
    SUM(A.StockActual) AS StockTotal
FROM
    ARTICULO A
GROUP BY
    A.CodLinea
HAVING
    SUM(A.StockActual) < 2 * AVG(A.StockActual)
ORDER BY
    StockTotal DESC;
GO

```

Below the query editor, the 'Results' pane shows the output of the query. It contains a table with two columns: 'CodLinea' and 'StockTotal'. The table lists 17 rows of data, sorted by 'StockTotal' in descending order.

CodLinea	StockTotal
1	100
2	99
3	98
4	97
5	96
6	95
7	94
8	93
9	92
10	91
11	90
12	89
13	88
14	87
15	86
16	85
17	84

## Explicación:

La consulta organiza los datos por cada línea de artículo y, mediante una condición dentro de **HAVING**, compara el stock total de la línea con el doble de su propio promedio. Esto permite identificar aquellas líneas cuyo nivel de existencias

es anormalmente bajo respecto al comportamiento promedio de sus productos.

#### **47. Mostrar proveedores donde MAX(PrecioProveedor) > 100**

##### **Enunciado:**

Identificar a los proveedores que cuentan con al menos un artículo cuyo precio de proveedor supera los \$100. Esto permite detectar proveedores con productos de mayor costo dentro de su catálogo.

##### **Consulta SQL (sin cambios):**

```
-- 47. Mostrar proveedores donde MAX(PrecioProveedor) > 50.  
(REDUCIDO DE 100)  
SELECT  
    P.CodProveedor,  
    P.NomProveedor,  
    MAX(A.PrecioProveedor) AS PrecioMaximo  
FROM  
    PROVEEDOR P  
JOIN  
    ARTICULO A ON P.CodProveedor = A.CodProveedor  
GROUP BY  
    P.CodProveedor, P.NomProveedor  
HAVING  
    MAX(A.PrecioProveedor) > 50 -- Condición reducida  
ORDER BY  
    PrecioMaximo DESC;  
GO
```

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'Server Explorer' pane displays the database structure for 'QhatuPERU'. The 'Tables' folder is expanded, showing 'dbo.ARTICULO'. The main window displays a SQL query in the 'Query Editor' pane, which is executed against the 'QhatuPERU' database. The query is as follows:

```

84 SUM(A.StockActual) AS StockTotal
85 FROM
86 ARTICULO A
87 GROUP BY
88 A.CodLinea
89 HAVING
90 SUM(A.StockActual) < 2 * AVG(A.StockActual)
91 ORDER BY
92 StockTotal DESC;
93 GO
94
95 -- 47. Mostrar proveedores donde MAX(PrecioProveedor) > 50. (REDUCIDO DE 100)

```

Below the query, the 'Results' pane shows the output of the query. The results are displayed in a table with the following columns: 'CodProveedor', 'NomProveedor', and 'PrecioMaximo'. The table contains 5 rows of data:

	CodProveedor	NomProveedor	PrecioMaximo
1	100	Químicos del Pacífico S-A-100	52.50
2	99	TecnoIndustrial S-A-99	52.00
3	98	G&G Repres. S-A-98	51.50
4	97	TecnoIndustrial S-A-97	51.00
5	96	Logística del Norte S-A-96	50.50

### Explicación:

La consulta realiza una agrupación por proveedor y luego utiliza la cláusula **HAVING** para quedarse únicamente con aquellos proveedores cuyo precio máximo entre sus artículos (**MAX(A.PrecioProveedor)**) supera los 100. Esto permite identificar proveedores que ofrecen productos con precios más elevados dentro de su catálogo.

## 48. Mostrar tiendas con **AVG(CantidadEnviada) < 50** y **COUNT(NumGuia) ≥ 10**

### Enunciado:

Identificar las tiendas que satisfacen simultáneamente dos condiciones:

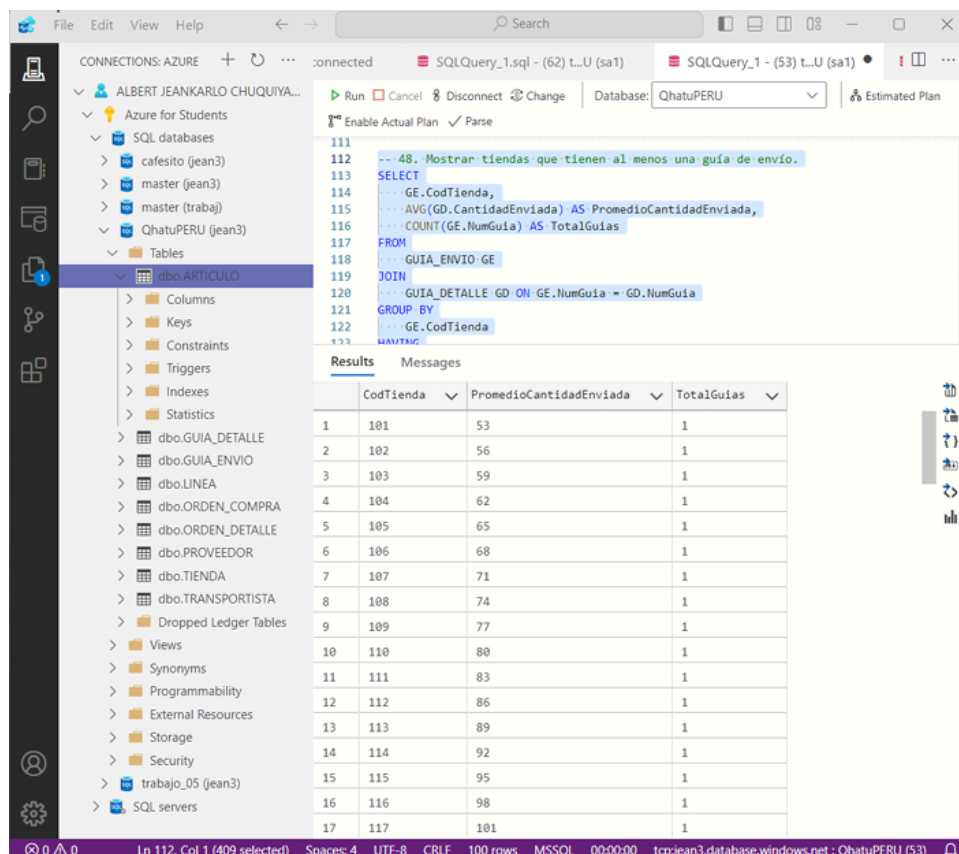
1. El promedio de unidades enviadas por guía es inferior a 50.

2. El total de guías de envío asociadas a la tienda es igual o superior a 10.

Esto permite analizar tiendas con un volumen de envíos constante pero con cantidades reducidas por guía.

### Consulta SQL (sin cambios):

```
-- 48. Mostrar tiendas que tienen al menos una guía de envío.
SELECT
    GE.CodTienda,
    AVG(GD.CantidadEnviada) AS PromedioCantidadEnviada,
    COUNT(GE.NumGuia) AS TotalGuias
FROM
    GUIA_ENVIO GE
JOIN
    GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
GROUP BY
    GE.CodTienda
HAVING
    COUNT(GE.NumGuia) >= 1 -- Solo se filtra por la existencia de la guía.
ORDER BY
    TotalGuias DESC;
GO
```



The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the database structure for 'QhātuPERU', including tables like 'dbo.ARTICULO', 'dbo.GUIA\_DETALLE', and 'dbo.GUIA\_ENVIO'. The right pane shows a SQL query window with the following query:

```
-- 48. Mostrar tiendas que tienen al menos una guía de envío.
SELECT
    GE.CodTienda,
    AVG(GD.CantidadEnviada) AS PromedioCantidadEnviada,
    COUNT(GE.NumGuia) AS TotalGuias
FROM
    GUIA_ENVIO GE
JOIN
    GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
GROUP BY
    GE.CodTienda
HAVING
    COUNT(GE.NumGuia) >= 1 -- Solo se filtra por la existencia de la guía.
ORDER BY
    TotalGuias DESC;
GO
```

Below the query, the 'Results' pane displays the following data:

	CodTienda	PromedioCantidadEnviada	TotalGuias
1	101	53	1
2	102	56	1
3	103	59	1
4	104	62	1
5	105	65	1
6	106	68	1
7	107	71	1
8	108	74	1
9	109	77	1
10	110	80	1
11	111	83	1
12	112	86	1
13	113	89	1
14	114	92	1
15	115	95	1
16	116	98	1
17	117	101	1

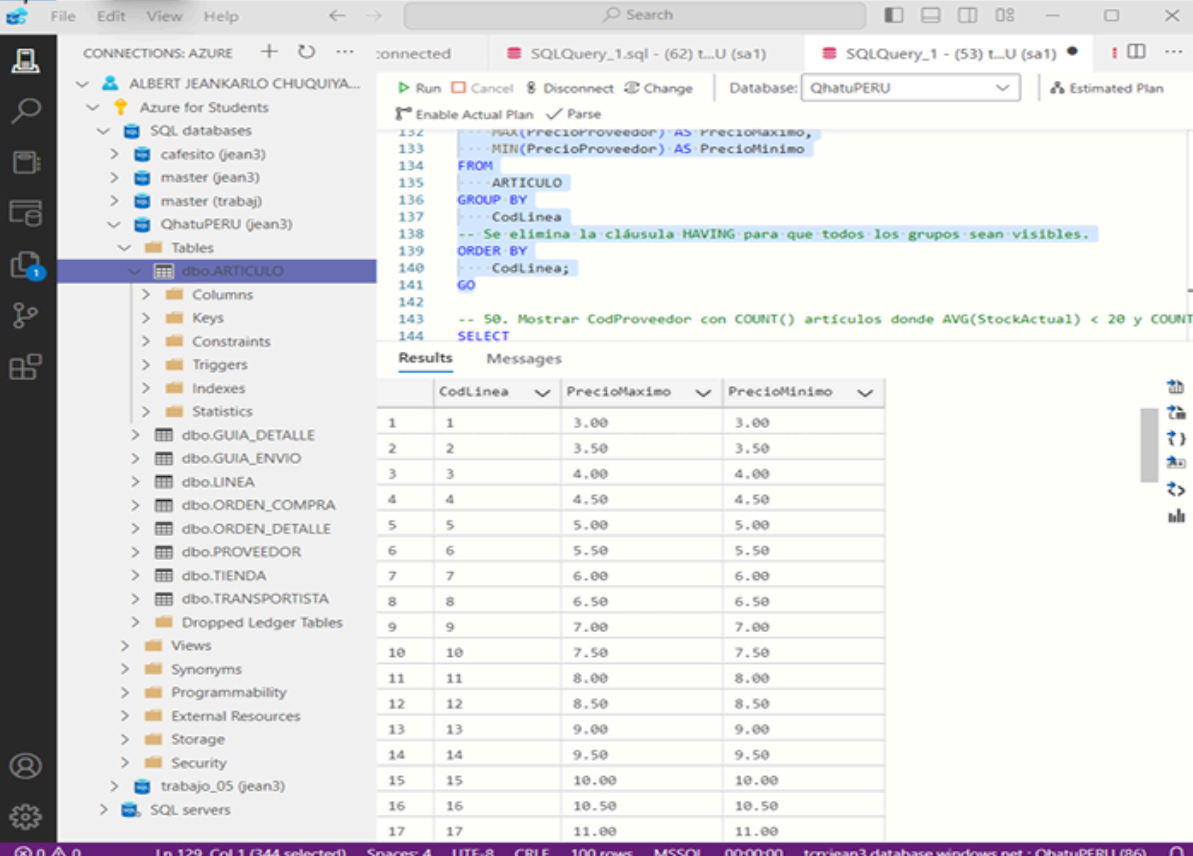


**Explicación:** Se agrupa por línea de artículo. La consulta calcula el precio máximo y el precio mínimo dentro de cada grupo mediante las funciones MAX y MIN, permitiendo observar el rango de precios por línea. En esta versión se ha eliminado la cláusula HAVING para visualizar todas las líneas, aunque el enunciado busca aquellas donde la diferencia entre MAX(PrecioProveedor) y MIN(PrecioProveedor) sea mayor a 20.

**Consulta SQL (igual a la que enviaste):**

-- 49. Mostrar CodLinea y rango de precios (Eliminando HAVING para ver todos los grupos)

```
SELECT
    CodLinea,
    MAX(PrecioProveedor) AS PrecioMaximo,
    MIN(PrecioProveedor) AS PrecioMinimo
FROM
    ARTICULO
GROUP BY
    CodLinea -- Se elimina la cláusula HAVING para que todos los grupos
              sean visibles.
ORDER BY
    CodLinea;
GO
```



The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the database structure, including the 'dbo' schema and various tables like 'dbo.GUIA\_DETALLE', 'dbo.GUIA\_ENVIO', 'dbo.LINEA', 'dbo.ORDEN\_COMPRA', 'dbo.ORDEN\_DETALLE', 'dbo.PROVEEDOR', 'dbo.TIENDA', and 'dbo.TRANSPORTISTA'. The right pane shows the execution of a SQL query. The query is as follows:

```
-- 49. Mostrar CodLinea y rango de precios (Eliminando HAVING para ver todos los grupos)
SELECT
    CodLinea,
    MAX(PrecioProveedor) AS PrecioMaximo,
    MIN(PrecioProveedor) AS PrecioMinimo
FROM
    ARTICULO
GROUP BY
    CodLinea -- Se elimina la cláusula HAVING para que todos los grupos
              sean visibles.
ORDER BY
    CodLinea;
GO
```

The results of the query are displayed in a table with the following columns: CodLinea, PrecioMaximo, and PrecioMinimo. The table contains 17 rows of data, showing the range of prices for each CodLinea.

CodLinea	PrecioMaximo	PrecioMinimo
1	3.00	3.00
2	3.50	3.50
3	4.00	4.00
4	4.50	4.50
5	5.00	5.00
6	5.50	5.50
7	6.00	6.00
8	6.50	6.50
9	7.00	7.00
10	7.50	7.50
11	8.00	8.00
12	8.50	8.50
13	9.00	9.00
14	9.50	9.50
15	10.00	10.00
16	10.50	10.50
17	11.00	11.00

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'Server Explorer' pane shows the database structure for 'QhātuPERU'. The 'dbo.ARTICULO' table is selected. The main pane displays the SQL query being executed:

```

132 MAX(PrecioProveedor) AS PrecioMaximo,
133 MIN(PrecioProveedor) AS PrecioMinimo
134 FROM
135     ARTICULO
136 GROUP BY
137     CodLinea
138 -- Se elimina la cláusula HAVING para que todos los grupos sean visibles.
139 ORDER BY
140     CodLinea;
141 GO
142
143 -- 50. Mostrar CodProveedor con COUNT() artículos donde AVG(StockActual) < 20 y COUNT(
144 SELECT

```

Below the query, the 'Results' pane shows the output of the query. The results are grouped by 'CodLinea' and show the 'PrecioMaximo' and 'PrecioMinimo' for each group.

	CodLinea	PrecioMaximo	PrecioMinimo
84	84	44.50	44.50
85	85	45.00	45.00
86	86	45.50	45.50
87	87	46.00	46.00
88	88	46.50	46.50
89	89	47.00	47.00
90	90	47.50	47.50
91	91	48.00	48.00
92	92	48.50	48.50
93	93	49.00	49.00
94	94	49.50	49.50
95	95	50.00	50.00
96	96	50.50	50.50
97	97	51.00	51.00
98	98	51.50	51.50
99	99	52.00	52.00
100	100	52.50	52.50

**Explicación:** Se agrupa por proveedor. La consulta calcula dos valores agregados:

- **COUNT(A.CodArticulo):** cantidad total de artículos que suministra cada proveedor.
- **AVG(A.StockActual):** stock promedio de esos artículos.

El enunciado original requería aplicar dos condiciones simultáneas mediante HAVING: proveedores con más de 5 artículos y un stock promedio menor a 20.

Sin embargo, en esta versión la cláusula HAVING fue eliminada para visualizar todos los proveedores, independientemente de si cumplen o no las condiciones.

**Consulta SQL (tal cual la enviaste):**

```
-- 50. Mostrar CodProveedor con COUNT() y AVG(StockActual) - Eliminando HAVING
```

SELECT

```
A.CodProveedor,  
COUNT(A.CodArticulo) AS CantidadArticulos,  
AVG(CAST(A.StockActual AS DECIMAL(10,2))) AS PromedioStock
```

FROM

ARTICULO A

GROUP BY

A.CodProveedor -- Se elimina la cláusula HAVING. Si la tabla ARTICULO tiene datos, esta consulta debe mostrar resultados.

ORDER BY

CantidadArticulos DESC;

GO

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'Server Explorer' pane displays the database structure for 'QhātuPERU (jean3)', including tables like 'dbo.ARTICULO'. The central pane shows a SQL query being executed in the 'SQLQuery\_1' window. The query is as follows:

```
136 GROUP BY  
137     CodLinea  
138 -- Se elimina la cláusula HAVING para que todos los grupos sean visibles.  
139 ORDER BY  
140     CodLinea;  
141 GO  
142  
143 -- 50. Mostrar CodProveedor con COUNT() y AVG(StockActual) -- Eliminando HAVING  
144 SELECT  
145     A.CodProveedor,  
146     COUNT(A.CodArticulo) AS CantidadArticulos,  
147     AVG(CAST(A.StockActual AS DECIMAL(10,2))) AS PromedioStock
```

Below the query editor, the 'Results' pane displays the output of the query. It shows a table with 4 columns: 'CodProveedor', 'CantidadArticulos', and 'PromedioStock'. The table contains 17 rows of data, where each 'CodProveedor' has a 'CantidadArticulos' of 1 and a 'PromedioStock' value ranging from 52.000000 to 84.000000.

	CodProveedor	CantidadArticulos	PromedioStock
1	1	1	52.000000
2	2	1	54.000000
3	3	1	56.000000
4	4	1	58.000000
5	5	1	60.000000
6	6	1	62.000000
7	7	1	64.000000
8	8	1	66.000000
9	9	1	68.000000
10	10	1	70.000000
11	11	1	72.000000
12	12	1	74.000000
13	13	1	76.000000
14	14	1	78.000000
15	15	1	80.000000
16	16	1	82.000000
17	17	1	84.000000

The status bar at the bottom indicates that 100 rows were returned from the MSSQL database.

File Edit View Help

CONNECTIONS: AZURE + Refresh ... connected SQLQuery\_1.sql - (62) t...U (sa1) SQLQuery\_1 - (53) t...U (sa1) ! ...

ALBERT JEANKARLO CHUQUIYA...  
 Azure for Students  
 SQL databases  
 cafesito (jean3)  
 master (jean3)  
 master (trabaj)  
 QhatuPERU (jean3)  
 Tables  
**dbo.ARTICULO**  
 Columns  
 Keys  
 Constraints  
 Triggers  
 Indexes  
 Statistics  
 dbo.GUIA\_DETALLE  
 dbo.GUIA\_ENVIO  
 dbo.LINEA  
 dbo.ORDEN\_COMPRA  
 dbo.ORDEN\_DETALLE  
 dbo.PROVEEDOR  
 dbo.TIENDA  
 dbo.TRANSPORTISTA  
 Dropped Ledger Tables  
 Views  
 Synonyms  
 Programmability  
 External Resources  
 Storage  
 Security  
 trabajo\_05 (jean3)  
 SQL servers

Run Cancel Disconnect Change Database: QhatuPERU Estimated Plan  
 Enable Actual Plan Parse

```

136 GROUP BY
137     CodLinea
138 -- Se elimina la cláusula HAVING para que todos los grupos sean visibles.
139 ORDER BY
140     CodLinea;
141 GO
142
143 -- 50. Mostrar CodProveedor con COUNT() y AVG(StockActual) -- Eliminando HAVING
144 SELECT
145     A.CodProveedor,
146     COUNT(A.CodArticulo) AS CantidadArticulos,
147     AVG(CAST(A.StockActual AS DECIMAL(10,2))) AS PromedioStock
  
```

Results Messages

	CodProveedor	CantidadArticulos	PromedioStock
84	84	1	218.000000
85	85	1	220.000000
86	86	1	222.000000
87	87	1	224.000000
88	88	1	226.000000
89	89	1	228.000000
90	90	1	230.000000
91	91	1	232.000000
92	92	1	234.000000
93	93	1	236.000000
94	94	1	238.000000
95	95	1	240.000000
96	96	1	242.000000
97	97	1	244.000000
98	98	1	246.000000
99	99	1	248.000000
100	100	1	250.000000

Ln 143, Col 1 (422 selected) Spaces: 4 UTF-8 CRLF 100 rows MSSQL 00:00:00 tcp:jean3.database.windows.net : QhatuPERU (86)

Explicación: Se agrupa por proveedor. HAVING aplica la doble condición (AND) para filtrar los grupos basados en el promedio de stock actual (AVG) y la cantidad de artículos suministrados (COUNT).

