



“Año De La Recuperación Y  
Consolidación De La Economía Peruana”



# **UNIVERSIDAD PERUANA LOS ANDES**

“FACULTAD DE INGENIERÍA”

ESCUELA PROFESIONAL “SISTEMAS Y  
COMPUTACIÓN”

**CÁTEDRA:** Base de Datos II

**CATEDRÁTICO:** Ing. Fernandez Bejarano Raul Enrique

**ESTUDIANTE:** Vega Brañez Samuel Max

**CICLO:** V

**SECCIÓN:** B1

**HUANCAYO PERÚ**

**2025**

Manual de implementación de consultas Primera parte de las consultas:

**Funciones de Agregación en SQL: Explicación Detallada**

Las Funciones de Agregación son esenciales en el análisis de datos. Su principal característica es que **operan verticalmente** sobre un conjunto de valores (una columna) y producen un **único valor de salida** que representa el resumen de esa operación.

**1. Las Cinco Funciones Fundamentales**

Función	Propósito	Aplicación Típica	Notas Importantes
<b>COUNT()</b>	Cuenta el número de filas o valores.	Saber el tamaño de un mercado, cuántos pedidos se hicieron.	Se puede usar COUNT(*) (cuenta todas las filas) o COUNT(columna) (cuenta solo valores NO nulos).
<b>SUM()</b>	Devuelve la suma total de valores.	Calcular los ingresos totales, el inventario valorado total.	Solo funciona con <b>columnas numéricas</b> . Ignora automáticamente los valores nulos (NULL).
<b>AVG()</b>	Calcula el valor promedio.	Determinar el salario promedio, el tiempo promedio de respuesta.	Solo funciona con <b>columnas numéricas</b> . El promedio es $\frac{\text{SUM}(X)}{\text{COUNT}(X \text{ no nulos})}$ .

Función	Propósito	Aplicación Típica	Notas Importantes
<b>MIN()</b>	Devuelve el valor más pequeño.	Encontrar el producto más barato, la fecha de registro más antigua.	Funciona con columnas <b>numéricas, de texto</b> (orden alfabético) y <b>de fecha</b> .
<b>MAX()</b>	Devuelve el valor más grande.	Encontrar el producto más caro, la fecha de registro más reciente.	Funciona con columnas <b>numéricas, de texto y de fecha</b> .

## 2. Uso Avanzado con GROUP BY y HAVING

El poder real de las funciones de agregación se manifiesta al agrupar datos.

### Cláusula GROUP BY

- **Función:** Agrupa filas que tienen los mismos valores en una o más columnas. La función de agregación se aplica **independientemente** a cada grupo formado.
- **Regla esencial:** Si usas una función de agregación en la cláusula SELECT, cualquier columna que **no** esté dentro de una función de agregación **debe** estar listada en la cláusula GROUP BY.
  - *Incorrecto:* SELECT Departamento, AVG(Salario) FROM Empleados; (SQL no sabría qué departamento mostrar).

- *Correcto:* SELECT Departamento, AVG(Salario)  
FROM Empleados GROUP BY Departamento;

### Cláusula HAVING

- **Función:** Es el equivalente del WHERE pero se usa para **filtrar grupos** de resultados después de que la agregación ha ocurrido. No puedes usar una función de agregación en la cláusula WHERE.
- **Ejemplo:** Mostrar solo los departamentos donde el salario promedio es superior a \$50,000.

SQL

```
SELECT Departamento, AVG(Salario) AS SalarioPromedio  
FROM Empleados  
GROUP BY Departamento  
HAVING AVG(Salario) > 50000; -- Filtra los grupos
```

### 3. Consideración de Valores Nulos (NULL)

Es crucial entender cómo manejan los valores nulos:

- **SUM(), AVG(), MIN(), MAX():** Estas funciones **ignoran** los valores NULL en sus cálculos. Por ejemplo, al calcular el promedio (AVG), los valores nulos no se cuentan en el denominador.
- **COUNT():**
  - COUNT(\*): Cuenta **todas** las filas, incluyendo aquellas con valores nulos, pues cuenta la existencia de la fila.
  - COUNT(nombre\_columna): Solo cuenta las filas donde nombre\_columna **no es nulo**.

### 4. Uso de DISTINCT

Puedes combinar la palabra clave DISTINCT dentro de las funciones COUNT, SUM y AVG para que operen **solo sobre valores únicos** dentro de la columna.

**Ejemplo:** Contar cuántos títulos de trabajo únicos existen.

## SQL

SELECT COUNT(DISTINCT TituloTrabajo)

FROM Empleados;

Esto evita que se cuenten múltiples empleados con el mismo título de trabajo, dándote la diversidad de roles en lugar del número de empleados.

### FUNCIONES DE AGREGACIÓN

- I. Explica de manera clara y didáctica qué son las **Funciones de Agregación** en SQL y cómo se utilizan.

1. Mostrar CodArticulo, DescripcionArticulo y ValorInventario.

**Salida de cada uno de los casos:**

**Enunciado:** Mostrar CodArticulo, DescripcionArticulo y ValorInventario.

**Consulta SQL:**

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

**Explicación:** Multiplica stock por precio por fila; castea para precisión.

2. Calcular el total monetario del inventario.
3. Obtener CodLinea y PrecioProveedor promedio.
4. Contar articulos descontinuados.
5. Mostrar PrecioMaximo y PrecioMinimo del catálogo.
6. Mostrar el Valor total enviado por guía.
7. Para cada CodArticulo, mostrar TotalSolicitado.
8. Contar órdenes únicas que incluyen cada articulo.
9. Calcular promedio de días por todas las órdenes con FechaIngreso.
10. Sumar CantidadEnviada por CodTransportista.

### Consultas SQL y Explicaciones (Texto Continuo)

1. **Mostrar CodArticulo, DescripcionArticulo y ValorInventario.**

**Enunciado:** Mostrar CodArticulo, DescripcionArticulo y ValorInventario.

## Código SQL:

-- 1. Mostrar CodArticulo, DescripcionArticulo y ValorInventario. (La primera consulta solicitada)

SELECT

A.CodArticulo, A.DescripcionArticulo,

-- Multiplica StockActual \* PrecioProveedor

CAST(A.StockActual \* CAST(A.PrecioProveedor AS  
DECIMAL(18,2)) AS DECIMAL(18,2)) AS ValorInventario

FROM

ARTICULO A;

GO

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the database structure for 'QhatuPERU (jean3)', with the 'dbo.ARTICULO' table selected. The right pane shows the SQL query editor with the following code:

```
1 USE QhatuPERU;  
2 GO  
3  
4 -- 1. Mostrar CodArticulo, DescripcionArticulo y ValorInventario. (La primera consulta)  
5 SELECT  
6     A.CodArticulo, A.DescripcionArticulo,  
7     -- Multiplica StockActual * PrecioProveedor  
8     CAST(A.StockActual * CAST(A.PrecioProveedor AS DECIMAL(18,2)) AS DECIMAL(18,2)) AS  
9 FROM  
10     ARTICULO A;  
11  
12 GO
```

Below the query editor, the 'Results' tab is active, displaying a table with 10 rows and 3 columns: 'CodArticulo', 'DescripcionArticulo', and 'ValorInventario'.

	CodArticulo	DescripcionArticulo	ValorInventario
1	1	Filtro Hepa Mod-2 Sku-1	156.00
2	2	Filtro Hepa Mod-3 Sku-2	189.00
3	3	Adhesivo Epóxico Mod-4 Sku-3	224.00
4	4	Filtro Hepa Mod-5 Sku-4	261.00
5	5	Adhesivo Epóxico Mod-6 Sku-5	300.00
6	6	Válvula Flujo Mod-7 Sku-6	341.00
7	7	Sensor Óptico Mod-8 Sku-7	384.00
8	8	Filtro Hepa Mod-9 Sku-8	429.00
9	9	Válvula Flujo Mod-10 Sku-9	476.00
10	10	Malla Industrial Mod-1 Sku-10	525.00

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'Server Explorer' pane displays the database structure for 'QhatuPERU (jean3)', including tables like 'dbo.ARTICULO'. The main window shows a SQL query in the 'Query Editor' pane, which is executed against the 'QhatuPERU' database. The query is as follows:

```

1  USE QhatuPERU;
2  GO
3
4  -- 1. Mostrar CodArticulo, DescripcionArticulo y ValorInventario. (La primera cons
5  SELECT
6      A.CodArticulo, A.DescripcionArticulo,
7      -- Multiplica StockActual * PrecioProveedor
8      CAST(A.StockActual * CAST(A.PrecioProveedor AS DECIMAL(18,2)) AS DECIMAL(18,2))
9  FROM
10     ARTICULO A;
11  GO
12

```

Below the query, the 'Results' pane displays the output of the query, showing a table with three columns: 'CodArticulo', 'DescripcionArticulo', and 'ValorInventario'. The table contains 11 rows of data.

CodArticulo	DescripcionArticulo	ValorInventario
191	Interrupor Termico Mod-2 Sk...	42336.00
192	Malla Industrial Mod-3 Sku-1...	42749.00
193	Cable Cat 6 Mod-4 Sku-193	43164.00
194	Válvula Flujo Mod-5 Sku-194	43581.00
195	Rodamiento Z-20 Mod-6 Sku-195	44000.00
196	Válvula Flujo Mod-7 Sku-196	44421.00
197	Rodamiento Z-20 Mod-8 Sku-197	44844.00
198	Rodamiento Z-20 Mod-9 Sku-198	45269.00
199	Aceite Sintético Mod-10 Sku-...	45696.00
200	Sensor Óptico Mod-1 Sku-200	46125.00

**Explicación:** Esta consulta multiplica el StockActual por el PrecioProveedor para obtener el valor de inventario de cada artículo. Se utiliza CAST para asegurar la precisión decimal adecuada en el resultado monetario.

## 2. Calcular el total monetario del inventario.

**Enunciado:** Calcular el total monetario del inventario.

**Código SQL:**

-- 2. Calcular el total monetario del inventario.

SELECT

CAST(SUM(A.StockActual \* CAST(A.PrecioProveedor AS DECIMAL(18,2))) AS MONEY) AS TotalInventarioMonetario

FROM

ARTICULO A;

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 'QhātuPERU' database. The query is as follows:

```

-- 2. Calcular el total monetario del inventario.
SELECT
    CAST(SUM(A.StockActual * CAST(A.PrecioProveedor AS DECIMAL(18,2))) AS MONEY) AS To
FROM
    ARTICULO A;
GO

```

Below the query, the 'Results' pane displays a table with the following data:

	CodArticulo	DescripcionArticulo	ValorInventario
191	191	Interruptor Termico Mod-2 Sk...	42336.00
192	192	Malla Industrial Mod-3 Sku-1...	42749.00
193	193	Cable Cat 6 Mod-4 Sku-193	43164.00
194	194	Válvula Flujo Mod-5 Sku-194	43581.00
195	195	Rodamiento Z-20 Mod-6 Sku-195	44000.00
196	196	Válvula Flujo Mod-7 Sku-196	44421.00
197	197	Rodamiento Z-20 Mod-8 Sku-197	44844.00
198	198	Rodamiento Z-20 Mod-9 Sku-198	45269.00
199	199	Aceite Sintético Mod-10 Sku-...	45696.00
200	200	Sensor Óptico Mod-1 Sku-200	46125.00
	TotalInventarioMonetario		
1	3314700.00		

**Explicación:** Se utiliza la función de agregación SUM() para sumar el valor de inventario calculado de *todos* los artículos, obteniendo el valor total del inventario en todo el catálogo.

### 3. Obtener CodLinea y PrecioProveedor promedio.

**Enunciado:** Obtener CodLinea y PrecioProveedor promedio.

**Código SQL:**

```

-- 3. Obtener CodLinea y PrecioProveedor promedio.
SELECT
    A.CodLinea,
    CAST(AVG(A.PrecioProveedor) AS DECIMAL(18,2)) AS
PrecioProveedorPromedio
FROM
    ARTICULO A
GROUP BY
    A.CodLinea
;
GO

```

SQLQuery\_1.sql - (75) T.U (sa1) | consultas\_01.sql - (54) T.U (sa1)

Database: QhataPERU

```

23
24 -- 3. Obtener CodLinea y PrecioProveedor promedio.
25 SELECT
26     A.CodLinea,
27     CAST(AVG(A.PrecioProveedor) AS DECIMAL(18,2)) AS PrecioProveedorPromedio
28 FROM
29     ARTICULO A
30 GROUP BY
31     A.CodLinea;
32 GO
33
34

```

	CodLinea	PrecioProveedorPromedio
1	1	3.00
2	2	3.50
3	3	4.00
4	4	4.50
5	5	5.00
6	6	5.50
7	7	6.00
8	8	6.50
9	9	7.00
10	10	7.50

SQLQuery\_1.sql - (75) T.U (sa1) | consultas\_01.sql - (54) T.U (sa1)

Database: QhataPERU

```

23
24 -- 3. Obtener CodLinea y PrecioProveedor promedio.
25 SELECT
26     A.CodLinea,
27     CAST(AVG(A.PrecioProveedor) AS DECIMAL(18,2)) AS PrecioProveedorPromedio
28 FROM
29     ARTICULO A
30 GROUP BY
31     A.CodLinea;
32 GO
33
34

```

	CodLinea	PrecioProveedorPromedio
191	191	98.00
192	192	98.50
193	193	99.00
194	194	99.50
195	195	100.00
196	196	100.50
197	197	101.00
198	198	101.50
199	199	102.00
200	200	102.50

**Explicación:** Esta consulta agrupa los artículos por CodLinea y calcula el precio promedio del proveedor (AVG()) para todos los productos que pertenecen a esa misma línea.

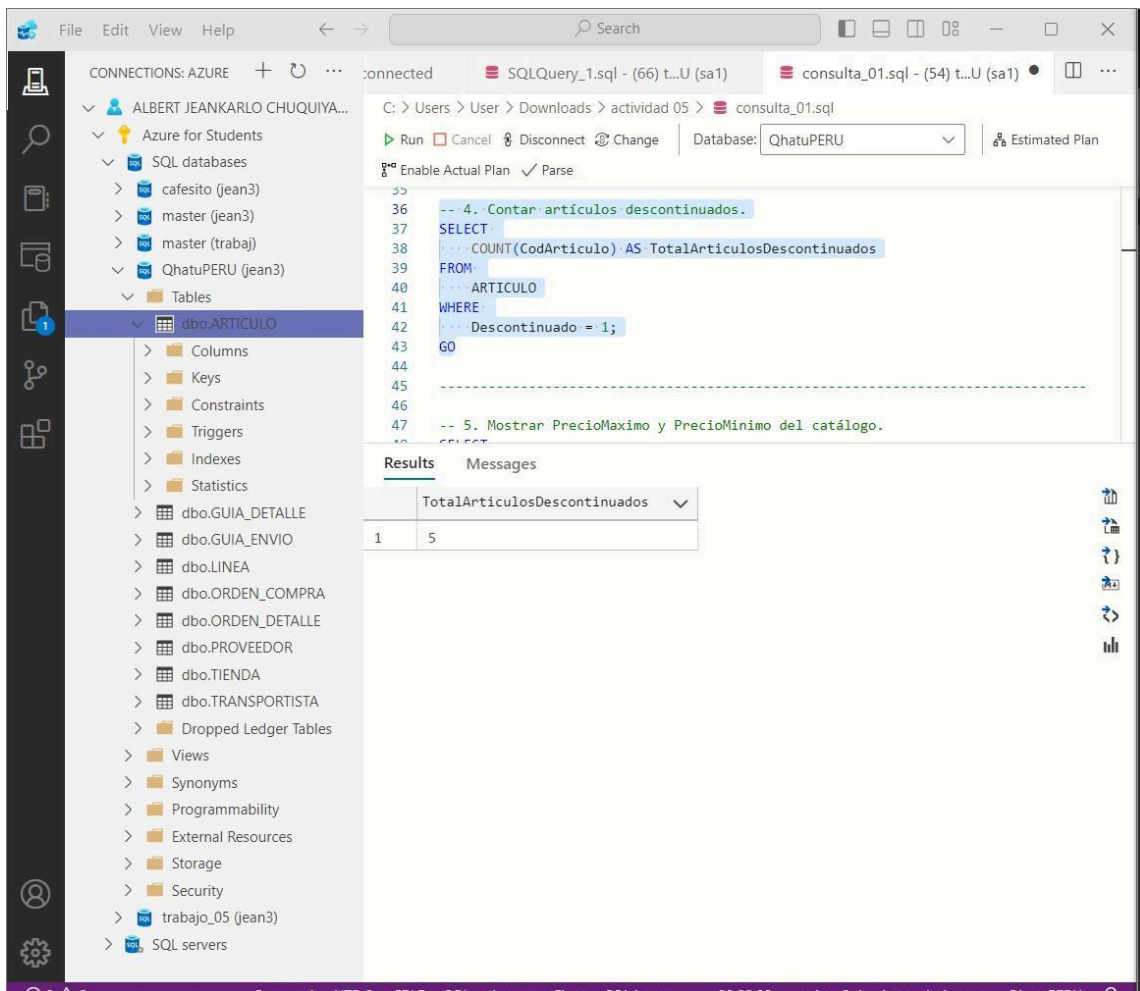
#### 4. Contar artículos descontinuados.

**Enunciado:** Contar artículos descontinuados.

**Código SQL:**

-- 4. Contar artículos descontinuados.

```
SELECT  
    COUNT(CodArticulo) AS TotalArticulosDescontinuados  
FROM  
    ARTICULO  
WHERE  
    Descontinuado = 1;  
GO
```



**Explicación:** Se filtra la tabla ARTICULO usando WHERE Descontinuado = 1 y luego se usa COUNT() para contar el número total de artículos marcados como descontinuados.

## 5. Mostrar PrecioMaximo y PrecioMinimo del catálogo.

**Enunciado:** Mostrar PrecioMaximo y PrecioMinimo del catálogo.

**Código SQL:**

-- 5. Mostrar PrecioMaximo y PrecioMinimo del catálogo.

SELECT

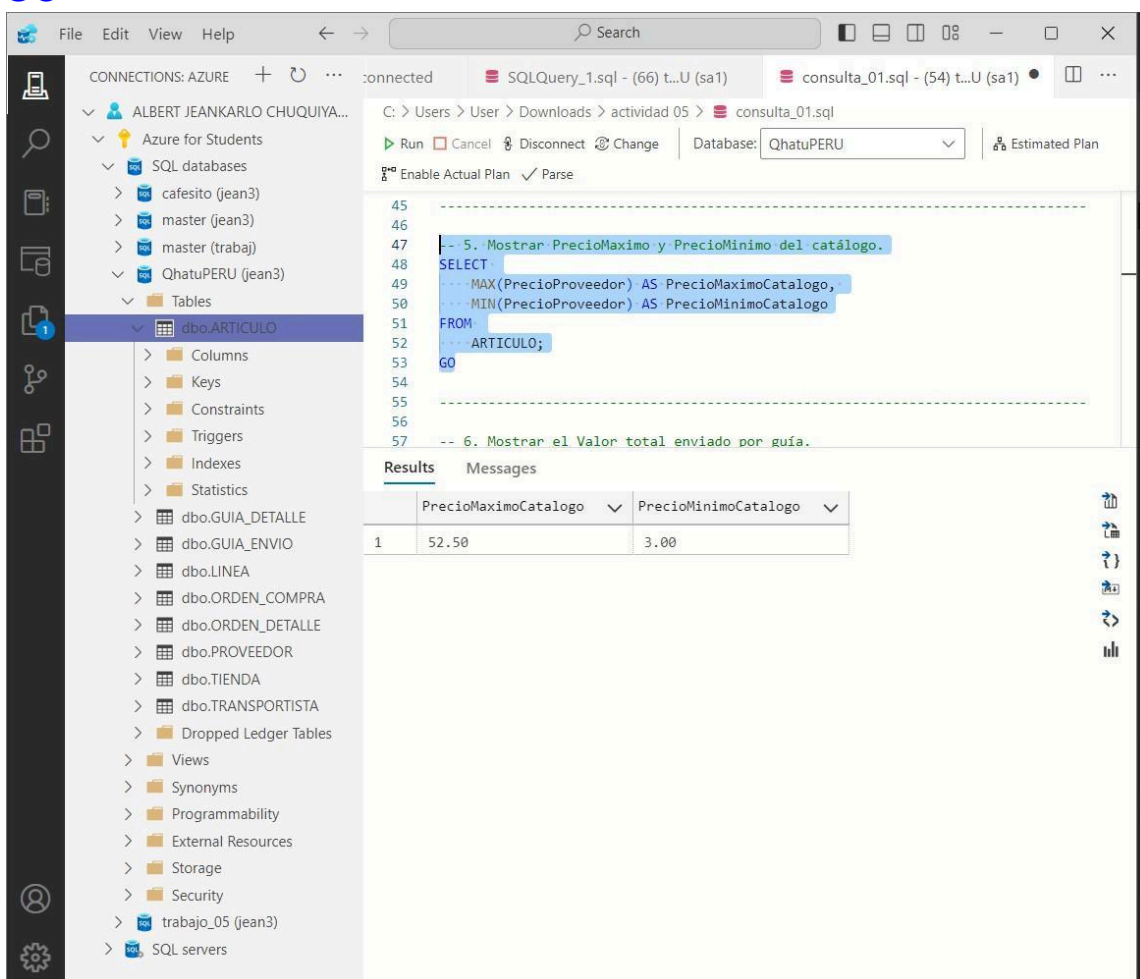
MAX(PrecioProveedor) AS PrecioMaximoCatalogo,

MIN(PrecioProveedor) AS PrecioMinimoCatalogo

FROM

ARTICULO;

GO



**Explicación:** Se utilizan las funciones de agregación MAX() y MIN() sobre la columna PrecioProveedor para identificar los precios más alto y más bajo de todos los artículos en el catálogo.

## 6. Mostrar el Valor total enviado por guía.

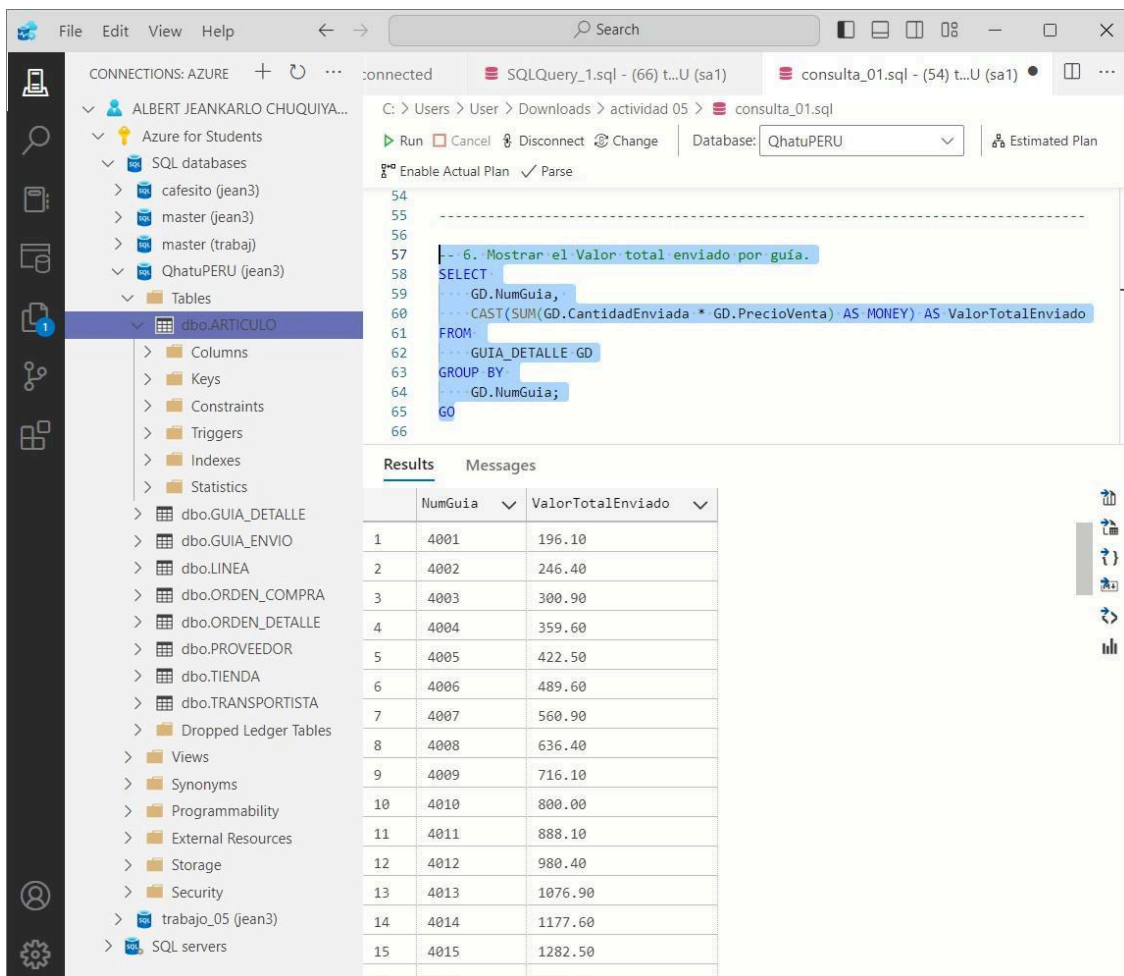
**Enunciado:** Mostrar el Valor total enviado por guía.

## Código SQL:

**Explicación:** Agrupa los detalles de envío por NumGuia y suma el valor monetario de los productos enviados (CantidadEnviada \* PrecioVenta) para determinar el valor total de cada guía.

-- 6. Mostrar el Valor total enviado por guía.

```
SELECT
    GD.NumGuia,
    CAST(SUM(GD.CantidadEnviada * GD.PrecioVenta) AS
MONEY) AS ValorTotalEnviado
FROM
    GUIDA_DETALLE GD
GROUP BY
    GD.NumGuia;
GO
```



The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the database structure, including the 'dbo' schema and various tables. The central pane shows the execution of a SQL query. The query is as follows:

```
-- 6. Mostrar el Valor total enviado por guía.
SELECT
    GD.NumGuia,
    CAST(SUM(GD.CantidadEnviada * GD.PrecioVenta) AS MONEY) AS ValorTotalEnviado
FROM
    GUIDA_DETALLE GD
GROUP BY
    GD.NumGuia;
GO
```

The results are displayed in a table with the following columns: NumGuia and ValorTotalEnviado. The table contains 16 rows of data.

NumGuia	ValorTotalEnviado
4001	196.10
4002	246.40
4003	300.90
4004	359.60
4005	422.50
4006	489.60
4007	560.90
4008	636.40
4009	716.10
4010	800.00
4011	888.10
4012	980.40
4013	1076.90
4014	1177.60
4015	1282.50
4016	1391.60

CONNECTIONS: AZURE + ... :connected SQLQuery\_1.sql - (66) t...U (sa1) consulta\_01.sql - (54) t...U (sa1)

Albert JEANKARLO CHUQUIYA...  
 Azure for Students  
 SQL databases  
 cafesito (jean3)  
 master (jean3)  
 master (trabaj)  
 QhātuPERU (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

C: > Users > User > Downloads > actividad 05 > consulta\_01.sql  
 Run Cancel Disconnect Change Database: QhātuPERU Estimated Plan  
 Enable Actual Plan Parse

```

54
55
56
57 -- 6. Mostrar el Valor total enviado por guía.
58 SELECT
59     GD.NumGuia,
60     CAST(SUM(GD.CantidadEnviada * GD.PrecioVenta) AS MONEY) AS ValorTotalEnviado
61 FROM
62     GUIA_DETALLE GD
63 GROUP BY
64     GD.NumGuia;
65 GO
66

```

Results Messages

	NumGuia	ValorTotalEnviado
85	4085	19062.50
86	4086	19465.60
87	4087	19872.90
88	4088	20284.40
89	4089	20700.10
90	4090	21120.00
91	4091	21544.10
92	4092	21972.40
93	4093	22404.90
94	4094	22841.60
95	4095	23282.50
96	4096	23727.60
97	4097	24176.90
98	4098	24630.40
99	4099	25088.10
100	4100	25900.00

## 7. Para cada CodArticulo, mostrar TotalSolicitado.

**Enunciado:** Para cada CodArticulo, mostrar TotalSolicitado.

**Código SQL:**

-- 7. Para cada CodArticulo, mostrar TotalSolicitado.

```

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

```

File Edit View Help Search

CONNECTIONS: AZURE + ... :connected SQLQuery\_1.sql - (66) t...U (sa1) consulta\_01.sql - (54) 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

C:\Users\User\Downloads\actividad 05\consulta\_01.sql  
Run Cancel Disconnect Change Database: QhatuPERU Estimated Plan  
Enable Actual Plan Parse

```
67  
68  
69 -- 7. Para cada CodArticulo, mostrar TotalSolicitado.  
70 SELECT  
71     OD.CodArticulo,  
72     SUM(OD.CantidadSolicitada) AS TotalSolicitado  
73 FROM  
74     ORDEN_DETALLE OD  
75 GROUP BY  
76     OD.CodArticulo  
77 ORDER BY  
78     OD.CodArticulo;  
79 GO
```

Results Messages

	CodArticulo	TotalSolicitado
1	1	105
2	2	110
3	3	115
4	4	120
5	5	125
6	6	130
7	7	135
8	8	140
9	9	145
10	10	150
11	11	155
12	12	160
13	13	165
14	14	170
15	15	175
16	16	180

Spaces: 4 UTF-8 GBK SQL 100 rows Choose SQL Language 00:00:00 tcp:jean3.database.windows.net: QhatuPERU

SQL Query Editor (QhātuPERU) showing the execution of a query to calculate the total quantity requested for each article code.

```

67
68
69 -- 7. Para cada CodArticulo, mostrar TotalSolicitado.
70 SELECT
71     OD.CodArticulo,
72     SUM(OD.CantidadSolicitada) AS TotalSolicitado
73 FROM
74     ORDEN_DETALLE OD
75 GROUP BY
76     OD.CodArticulo
77 ORDER BY
78     OD.CodArticulo;
79 GO

```

CodArticulo	TotalSolicitado
85	525
86	530
87	535
88	540
89	545
90	550
91	555
92	560
93	565
94	570
95	575
96	580
97	585
98	590
99	595
100	600

**Explicación:** Agrupa los registros de órdenes por CodArticulo y utiliza SUM() sobre la columna CantidadSolicitada para obtener la cantidad total de cada artículo que ha sido solicitada.

## 8. Contar órdenes únicas que incluyen cada artículo.

**Enunciado:** Contar órdenes únicas que incluyen cada artículo.

**Código SQL:**

-- 8. Contar órdenes únicas que incluyen cada artículo.

```

SELECT
    OD.CodArticulo,
    COUNT(DISTINCT OD.NumOrden) AS TotalOrdenesUnicas
FROM
    ORDEN_DETALLE OD
GROUP BY
    OD.CodArticulo
ORDER BY

```

OD.CodArticulo;  
GO

The screenshot displays the SQL Server Enterprise Manager interface. On the left, the 'Server Explorer' pane shows the database structure for 'QhātuPERU (jean3)', with 'dbo.ARTICULO' selected. The central pane shows a SQL query being executed in the 'consulta\_01.sql' file. The query is as follows:

```
-- 8. Contar órdenes únicas que incluyen cada artículo.
SELECT
    OD.CodArticulo,
    COUNT(DISTINCT OD.NumOrden) AS TotalOrdenesUnicas
FROM
    ORDEN_DETALLE OD
GROUP BY
    OD.CodArticulo
ORDER BY
    OD.CodArticulo;
GO
```

Below the query editor, the 'Results' pane shows the output of the query. It contains a table with two columns: 'CodArticulo' and 'TotalOrdenesUnicas'. The table lists 15 unique article codes, each with a count of 1.

CodArticulo	TotalOrdenesUnicas
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	1
9	1
10	1
11	1
12	1
13	1
14	1
15	1

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'Server Explorer' pane displays the database structure for 'QhātuPERU (jean3)', with 'dbo.ARTICULO' selected. The 'Query Editor' pane shows a SQL query:
 

```
-- 8. Contar órdenes únicas que incluyen cada artículo.
SELECT
  OD.CodArticulo,
  COUNT(DISTINCT OD.NumOrden) AS TotalOrdenesUnicas
FROM
  ORDEN_DETALLE OD
GROUP BY
  OD.CodArticulo
ORDER BY
  OD.CodArticulo;
GO
```

 The 'Results' pane at the bottom displays the output of the query as a table with two columns: 'CodArticulo' and 'TotalOrdenesUnicas'. The table contains 10 rows of data, showing that each article has exactly 1 unique order.
 

CodArticulo	TotalOrdenesUnicas
85	1
86	1
87	1
88	1
89	1
90	1
91	1
92	1
93	1
94	1
95	1
96	1
97	1
98	1
99	1
100	1

**Explicación:** Agrupa por CodArticulo y utiliza COUNT(DISTINCT NumOrden) para contar cuántos números de orden *diferentes* (únicos) compraron ese artículo.

## 9. Calcular promedio de días por todas las órdenes con FechaIngreso.

**Enunciado:** Calcular promedio de días por todas las órdenes con FechaIngreso.

**Código SQL:**

-- 9. Calcular promedio de días por todas las órdenes con FechaIngreso.

SELECT

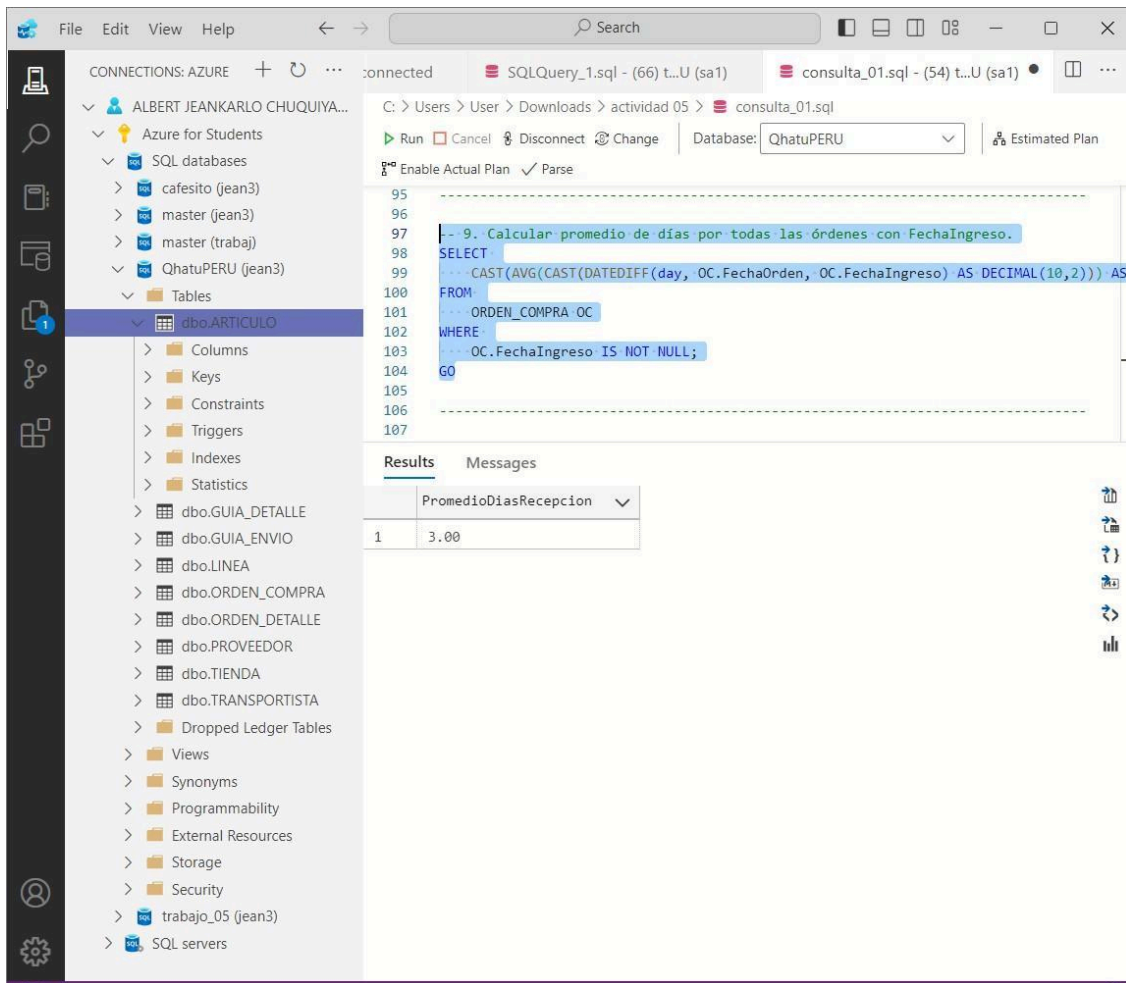
CAST(AVG(CAST(DATEDIFF(day, OC.FechaOrden, OC.FechaIngreso) AS DECIMAL(10,2))) AS DECIMAL(10,2)) AS PromedioDiasRecepcion

FROM

```

ORDEN_COMPRA OC
WHERE
OC.FechaIngreso IS NOT NULL;
GO

```



**Explicación:** DATEDIFF(day, ...) calcula el tiempo en días entre la orden y el ingreso. Luego, AVG() calcula el promedio de ese tiempo de recepción para todas las órdenes que ya tienen una FechaIngreso.

## 10. Sumar CantidadEnviada por CodTransportista.

**Enunciado:** Sumar CantidadEnviada por CodTransportista.

**Código SQL:**

```
-- 10. Sumar CantidadEnviada por CodTransportista.
```

```

SELECT
    T.CodTransportista,
    T.NomTransportista,

```

```

SUM(GD.CantidadEnviada) AS TotalCantidadEnviada
FROM
    GUIA_ENVIO GE
JOIN
    GUIA_DETALLE GD ON GE.NumGuia = GD.NumGuia
JOIN
    TRANSPORTISTA T ON GE.CodTransportista =
T.CodTransportista
GROUP BY
    T.CodTransportista, T.NomTransportista
ORDER BY
    TotalCantidadEnviada 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.GUIA\_DETALLE', 'dbo.GUIA\_ENVIO', and 'dbo.TRANSPORTISTA'. The central pane shows a SQL query being executed in 'consulta\_01.sql'. The query is a SELECT statement that joins 'GUIA\_ENVIO' and 'GUIA\_DETALLE' tables, grouping by transportista code and name, and ordering by total quantity sent in descending order. The 'Results' pane at the bottom displays the output of the query as a table with 15 rows.

	CodTransportista	NomTransportista	TotalCantidadEnviada
1	2100	Rutas Andinas F-100	350
2	2099	Delivery Perú F-99	347
3	2098	Delivery Perú F-98	344
4	2097	Envíos Urgentes F-97	341
5	2096	Carga Pesada F-96	338
6	2095	Envíos Urgentes F-95	335
7	2094	Rutas Andinas F-94	332
8	2093	Carga Pesada F-93	329
9	2092	Carga Pesada F-92	326
10	2091	Rutas Andinas F-91	323
11	2090	Transportes del Sur F-90	320
12	2089	LogiExpress F-89	317
13	2088	Transportes del Sur F-88	314
14	2087	Carga Pesada F-87	311
15	2086	Envíos Urgentes F-86	308

