

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

Primera Parte: Consultas con Funciones de Agregación en SQL

Funciones de Agregación en SQL: Explicación Detallada

Las funciones de agregación cumplen un papel clave en el proceso de análisis dentro de una base de datos. Estas funciones se caracterizan por trabajar de manera vertical sobre un conjunto de registros, es decir, toman los valores de una columna y generan un solo resultado que resume o representa la operación realizada sobre esos datos.

1. Las Cinco Funciones Principales:

Función	Propósito	Aplicación Típica	Notas Importantes
COUNT()	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).
SUM()	Devuelve la suma total de valores.	Calcular los ingresos totales, el inventario valorado total.	Solo funciona con columnas numéricas . Ignora automáticamente los valores nulos (NULL).
AVG()	Calcula el valor promedio.	Determinar el salario promedio, el tiempo promedio de respuesta.	Solo funciona con columnas numéricas . El promedio es $\frac{\text{SUM}(X)}{\text{COUNT}(X \text{ no nulos})}$.

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

2. Uso Avanzado con GROUP BY y HAVING

El verdadero potencial de las funciones de agregación se observa cuando se aplican a datos organizados por grupos.

Cláusula GROUP BY

- **Propósito:** Permite **clasificar y agrupar** las filas que compartan valores iguales en una o más columnas. Una vez formados los grupos, las funciones de agregación se ejecutan **de manera independiente para cada conjunto**.
- **Regla clave:** Siempre que incluyas una función de agregación en el SELECT, cualquier columna que aparezca fuera de dicha función **debe obligatoriamente** colocarse en el GROUP BY.

Incorrecto:

```
SELECT Departamento, AVG(Salario)
FROM Empleados;
```

- (El motor SQL no sabe qué departamento mostrar, ya que no hay agrupación definida.)

Correcto:

```
SELECT Departamento, AVG(Salario)
FROM Empleados
GROUP BY Departamento;
```

○

Cláusula HAVING

- **Propósito:** Funciona como un filtro similar a **WHERE**, pero se aplica **después** de que los grupos ya han sido generados y procesados por las funciones de agregación.
- Es importante recordar que **no se puede usar una función de agregación dentro de WHERE**, por eso HAVING existe.
- **Ejemplo:** Obtener únicamente los departamentos cuyo salario promedio supere los 50,000.

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

3. Manejo de Valores Nulos (NULL)

Comprender cómo las funciones de agregación tratan los valores NULL es fundamental:

- **SUM(), AVG(), MIN(), MAX():**
Estas funciones **no toman en cuenta los valores nulos**. Por ejemplo, si usas AVG(), los NULL no forman parte del cálculo ni del conteo total de valores.
- **COUNT():**
 - **COUNT(*):** Cuenta **todas las filas**, incluso aquellas donde algún campo es NULL, ya que solo verifica la existencia del registro.
 - **COUNT(nombre_columna):** Únicamente contabiliza las filas donde la columna indicada **no tiene NULL**.

4. Uso de DISTINCT en Funciones de Agregación

Es posible agregar la palabra clave **DISTINCT** dentro de funciones como COUNT, SUM o AVG para que trabajen exclusivamente con **valores diferentes**, ignorando duplicados.

Ejemplo: Contar cuántos títulos de trabajo distintos existen.

```
SELECT COUNT(DISTINCT TituloTrabajo)
FROM Empleados;
```

De esta forma, en lugar de contar cuántos empleados hay, se obtiene la **variedad de cargos** presentes en la organización.

FUNCIONES DE AGREGACIÓN

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

1. Mostrar CodArtículo, DescripciónArtículo y ValorInventario.

Salida de cada uno de los casos:

Enunciado: Mostrar CodArtículo, DescripciónArtículo y ValorInventario.

Consulta SQL:

```
SELECT
    A.CodArtículo,
    A.DescripciónArtículo,
    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 artículos descontinuados.
5. Mostrar PrecioMaximo y PrecioMinimo del catálogo.
6. Mostrar el Valor total enviado por guía.
7. Para cada CodArtículo, mostrar TotalSolicitado.
8. Contar órdenes únicas que incluyen cada artículo.
9. Calcular promedio de días por todas las órdenes con FechaIngreso.
10. Sumar CantidadEnviada por CodTransportista.

1. Consulta para Mostrar CodArticulo, DescripcionArticulo y ValorInventario

Enunciado:

Se requiere obtener el código del artículo, la descripción del producto y el valor total del inventario asociado a cada artículo.

Explicación:

Esta consulta permite visualizar información básica y esencial de los artículos almacenados. El campo *CodArticulo* identifica de forma única cada producto, *DescripcionArticulo* proporciona el nombre o detalle del artículo, y *ValorInventario* refleja el valor económico que representa ese artículo dentro del inventario.

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

SELECT

A.CodArticulo,

A.DescripcionArticulo,

-- Calcula el valor del inventario multiplicando StockActual por 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. On the left, the 'Server Explorer' pane displays the database structure for 'QhatsuPERU', including tables like 'dbo.ARTICULO'. The central pane shows a SQL query being executed in the 'QhatsuPERU' database. The query is as follows:

```
USE QhatsuPERU;
GO

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

Below the query editor, the 'Results' pane displays the output of the query as 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 'QhātuPERU'. The 'Tables' folder is expanded, showing various tables like 'dbo.GUÍA_DETALLE', 'dbo.GUÍA_ENVIO', 'dbo.LINEA', 'dbo.ORDEN_COMPRA', 'dbo.ORDEN_DETALLE', 'dbo.PROVEEDOR', 'dbo.TIENDA', 'dbo.TRANSPORTISTA', and 'Dropped Ledger Tables'. The 'dbo.ARTICULO' table is selected. In the center, the 'Query Editor' window shows a SQL query being executed. The query is as follows:

```

1  USE QhātuPERU;
2  GO
3
4  -- 1. Muestran CodArtículo, DescripciónArtículo y ValorInventario. (La primera cons
5  SELECT
6      A.CodArtículo, A.DescripciónArtículo,
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

```

Below the query, the 'Results' pane shows the output of the query. It contains a table with three columns: 'CodArtículo', 'DescripciónArtículo', and 'ValorInventario'. The table has 20 rows of data, representing different inventory items and their values.

CodArtículo	DescripciónArtículo	ValorInventario
191	Interrupción 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 instrucción calcula el valor total del inventario sumando el resultado de multiplicar el StockActual por el PrecioProveedor para cada artículo. El uso de CAST garantiza que el cálculo se realice con el formato decimal adecuado, asegurando la correcta representación del monto final.

2. Calcular el total monetario del inventario

Enunciado:

Obtener el monto total que representa todo el inventario expresado en valor monetario.

Código SQL (sin cambios):

```

-- 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 'QhatuPERU (jean3)', including tables like 'dbo.ARTICULO'. The main window shows a SQL query being executed in the 'Query Editor'. 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 TotalInventarioMonetario
FROM
    ARTICULO A;
GO

```

The 'Results' pane displays the output of the query. It shows a table with three columns: 'CodArticulo', 'DescripcionArticulo', and 'ValorInventario'. The table contains 20 rows of data, followed by a summary row for 'TotalInventarioMonetario'.

CodArticulo	DescripcionArticulo	ValorInventario
191	Interrupcion 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
TotalInventarioMonetario		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

```


File Edit View Help Search

CONNECTIONS: AZURE + ... :connected SQLQuery_1.sql - (75) 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

C: > Users > User > Downloads > actividad 05 > consulta_01.sql

Run Cancel Disconnect Change Database: QhatuPERU Estimated Plan

Enable Actual Plan Parse

```

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

```

Results Messages

	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

File Edit View Help Search

CONNECTIONS: AZURE + ... :connected SQLQuery_1.sql - (75) 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

C: > Users > User > Downloads > actividad 05 > consulta_01.sql

Run Cancel Disconnect Change Database: QhatuPERU Estimated Plan

Enable Actual Plan Parse

```

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

```

Results Messages

	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

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
```

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the database structure for 'QhatuPERU (jean3)', with 'dbo.ARTICULO' selected. The right pane shows the execution of a SQL query. The query text is: -- 5. Mostrar PrecioMaximo y PrecioMinimo del catálogo. SELECT MAX(PrecioProveedor) AS PrecioMaximoCatalogo, MIN(PrecioProveedor) AS PrecioMinimoCatalogo FROM ARTICULO; GO. The results pane shows a single row with the values 52.50 for PrecioMaximoCatalogo and 3.00 for PrecioMinimoCatalogo.

	PrecioMaximoCatalogo	PrecioMinimoCatalogo
1	52.50	3.00

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
    GUIA_DETALLE GD
GROUP BY
    GD.NumGuia;
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.GUÍA_DETALLE'. The central pane shows a SQL query being executed. 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
    GUIA_DETALLE GD
GROUP BY
    GD.NumGuia;
GO
```

Below the query editor, the 'Results' pane displays the output of the query. It shows a table with two columns: 'NumGuia' and 'ValorTotalEnviado'. The table contains 15 rows of data, representing the total value sent for each guide number.

	NumGuia	ValorTotalEnviado
1	4001	196.10
2	4002	246.40
3	4003	300.90
4	4004	359.60
5	4005	422.50
6	4006	489.60
7	4007	560.90
8	4008	636.40
9	4009	716.10
10	4010	800.00
11	4011	888.10
12	4012	980.40
13	4013	1076.90
14	4014	1177.60
15	4015	1282.50

CONNECTIONS: AZURE + ... connected | SQLQuery_1.sql - (66) t...U (sa1) | consulta_01.sql - (54) t...U (sa1)

File Edit View Help | Search

C: > Users > User > Downloads > actividad 05 > consulta_01.sql

Run Cancel Disconnect Change | Database: QhaturPERU | 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

CONNECTIONS: AZURE + ... :connected SQLQuery_1.sql - (66) t...U (sa1) consulta_01.sql - (54) t...U (sa1)

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

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

File Edit View Help

CONNECTIONS: AZURE + ... :connected SQLQuery_1.sql - (66) t...U (sa1) consulta_01.sql - (54) t...U (sa1)

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

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

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.

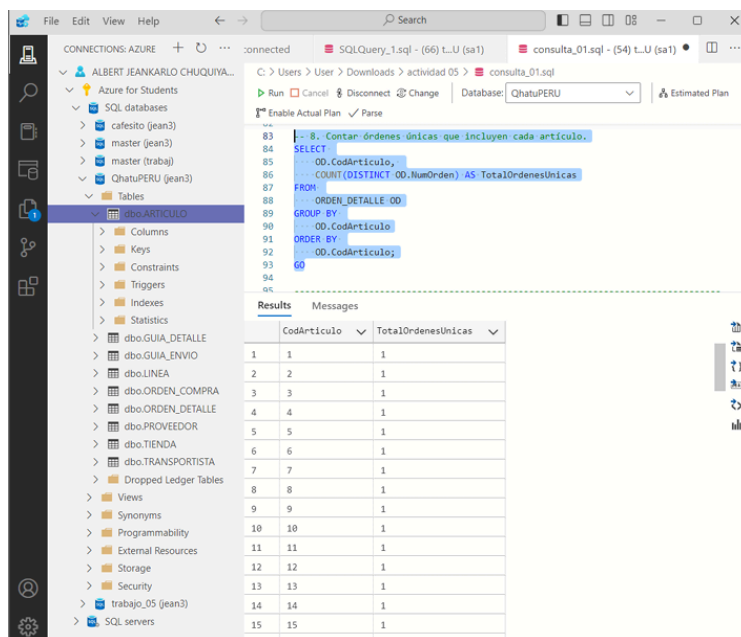
8. Contar órdenes únicas que incluyen cada artículo.

Enunciado: Contar órdenes únicas que incluyen cada artículo.

Código SQL:

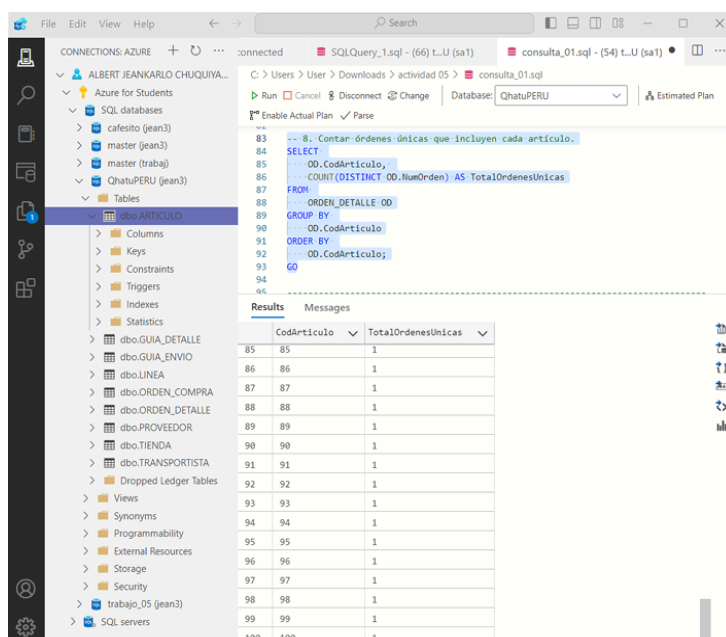
-- 8. Contar órdenes únicas que incluyen cada artículo.

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



Query Results:

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



Query Results:

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

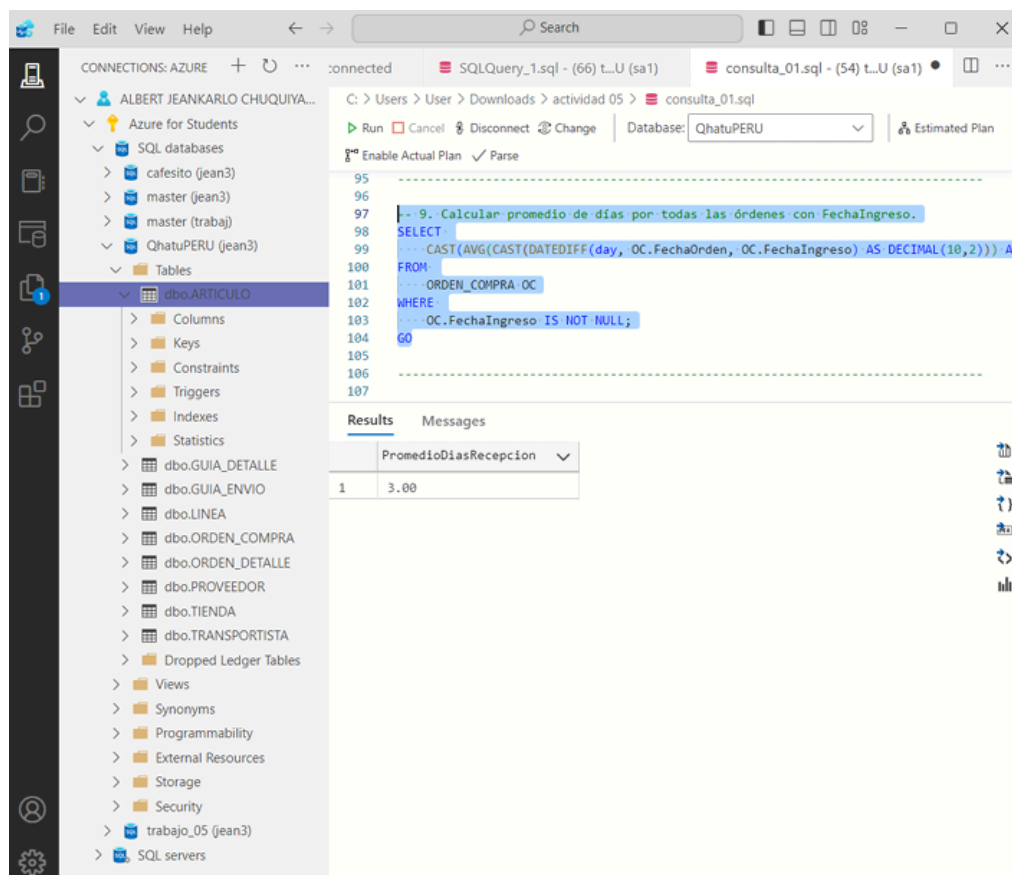
FROM

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 the execution of a SQL query in 'consulta_01.sql'. The query is as follows:

```
-- 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
```

Below the query, the 'Results' pane displays the output of the query. The results are as follows:

	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

The screenshot shows the Microsoft SQL Server Enterprise Manager interface. On the left, the 'Server Enterprise Explorer' tree is expanded to show the 'dbo' database, specifically the 'ARTICULO' table. The central pane displays the SQL query being executed in the 'consulta_01.sql' file. The query is as follows:

```

102 WHERE
103     OC.FechaIngreso IS NOT NULL;
104 GO
105
106 -----
107
108 -- 10. Sumar CantidadEnviada por CodTransportista.
109 SELECT
110     T.CodTransportista,
111     T.NomTransportista,
112     SUM(GD.CantidadEnviada) AS TotalCantidadEnviada
113 FROM
114     ... GUIA ENVIO GE
  
```

Below the query, the 'Results' tab is active, displaying the output of the query in a table format. The table has three columns: 'CodTransportista', 'NomTransportista', and 'TotalCantidadEnviada'. The results are sorted by 'CodTransportista' in descending order.

	CodTransportista	NomTransportista	TotalCantidadEnviada
85	2016	Envíos Urgentes F-16	98
86	2015	Fletes Seguros F-15	95
87	2014	Carga Pesada F-14	92
88	2013	Rutas Andinas F-13	89
89	2012	Courier Veloz F-12	86
90	2011	Delivery Perú F-11	83
91	2010	Delivery Perú F-10	80
92	2009	LogiExpress F-9	77
93	2008	Delivery Perú F-8	74
94	2007	Fletes Seguros F-7	71
95	2006	LogiExpress F-6	68
96	2005	Rutas Andinas F-5	65
97	2004	Envíos Urgentes F-4	62
98	2003	Carga Pesada F-3	59
99	2002	Transp. Rápido F-2	56
100	2001	LogiExpress F-1	53

Explicación: Une las tres tablas (GUIA_ENVIO, GUIA_DETALLE, TRANSPORTISTA). Agrupa el resultado por transportista y suma la CantidadEnviada para determinar el volumen total transportado por cada compañía.