INSTITUTO POLITECNICO NACIONAL
UNIDAD PROFESIONAL INTERDISCIPLINARIA EN INGENIERÍA Y TECNOLOGÍAS AVANZADAS
upiita-ipn

Practica 1:

CONSULTAS Y USO DE SERVIDORES VINCULADOS

Alumno: Martínez Barrueta Mariana
Gamez Gress Isaac Humberto

Profesor: De la Cruz Sosa Carlos

Base De Datos Distrbuidas

consultas

Consulta 1:

```
cantidad total vendida y nombre del cliente. */
SELECT
    P.Name AS Producto,
    T.cant AS TotalVendido,
    PER.FirstName + ' ' + PER.LastName AS NombreCliente
FROM Production.Product P
JOIN (
    -- 1. Subconsulta: Encuentra los 10 IDs más vendidos SOLO en 2014
    SELECT TOP 10
        SOD.ProductID,
        SUM(SOD.OrderQty) AS cant
    FROM Sales.SalesOrderDetail SOD
    JOIN Sales.SalesOrderHeader SOH ON SOD.SalesOrderID = SOH.SalesOrderID
    WHERE YEAR(SOH.OrderDate) = 2014
    GROUP BY SOD.ProductID
    ORDER BY cant DESC
) AS T ON P.ProductID = T.ProductID
-- 2. Unimos con el detalle para ver a TODOS los clientes de esos productos en 2014
JOIN Sales.SalesOrderDetail D ON P.ProductID = D.ProductID
JOIN Sales.SalesOrderHeader H ON D.SalesOrderID = H.SalesOrderID
JOIN Sales.Customer C ON H.CustomerID = C.CustomerID
JOIN Person.Person PER ON C.PersonID = PER.BusinessEntityID
WHERE YEAR(H.OrderDate) = 2014
ORDER BY T.cant DESC, Producto;
```

Ejecución:

	Producto	TotalVendido	NombreCliente
1	Water Bottle - 30 oz.	2902	Isabella Adams
2	Water Bottle - 30 oz.	2902	Julia Adams
3	Water Bottle - 30 oz.	2902	Kaitlyn Adams
4	Water Bottle - 30 oz.	2902	Kaitlyn Adams
5	Water Bottle - 30 oz.	2902	Morgan Adams
6	Water Bottle - 30 oz.	2902	Anna Alexander
7	Water Bottle - 30 oz.	2902	Ian Alexander
8	Water Bottle - 30 oz.	2902	Jasmine Alexander
9	Water Bottle - 30 oz.	2902	Marcus Alexander
10	Water Bottle - 30 oz.	2902	Megan Alexander

Consulta derivada de la anterior:

```sql
SELECT
    P.Name AS Producto,
    T.cant AS TotalVendido2014,
    T.PrecioPromedio,
    PER.FirstName + ' ' + PER.LastName AS NombreCliente
FROM Production.Product P
JOIN (
    -- Subconsulta
    SELECT TOP 10
        SOD.ProductID,
        SUM(SOD.OrderQty) AS cant,
        AVG(SOD.UnitPrice) AS PrecioPromedio
    FROM Sales.SalesOrderDetail SOD
    JOIN Sales.SalesOrderHeader SOH ON SOD.SalesOrderID = SOH.SalesOrderID
    JOIN Production.Product PRO ON SOD.ProductID = PRO.ProductID
    WHERE YEAR(SOH.OrderDate) = 2014     -- Solo año 2014
      AND PRO.ListPrice > 1000           -- Solo productos > 1000
    GROUP BY SOD.ProductID
    ORDER BY cant DESC
) AS T ON P.ProductID = T.ProductID
-- 2. Detalle de los clientes
JOIN Sales.SalesOrderDetail D ON P.ProductID = D.ProductID
JOIN Sales.SalesOrderHeader H ON D.SalesOrderID = H.SalesOrderID
JOIN Sales.Customer C ON H.CustomerID = C.CustomerID
JOIN Person.Person PER ON C.PersonID = PER.BusinessEntityID
WHERE YEAR(H.OrderDate) = 2014
ORDER BY T.cant DESC, Producto;
```

Ejecución:

	Producto	TotalVendido2014	PrecioPromedio	NombreCliente
1	Mountain-200 Black, 38	619	1981.7872	Jay Adams
2	Mountain-200 Black, 38	619	1981.7872	Mary Adams
3	Mountain-200 Black, 38	619	1981.7872	Anna Albright
4	Mountain-200 Black, 38	619	1981.7872	Arianna Alexander
5	Mountain-200 Black, 38	619	1981.7872	Kaitlyn Alexander
6	Mountain-200 Black, 38	619	1981.7872	Eric Allen
7	Mountain-200 Black, 38	619	1981.7872	Kristy Alvarez
8	Mountain-200 Black, 38	619	1981.7872	Peter Anand
9	Mountain-200 Black, 38	619	1981.7872	Katrina Andersen
10	Mountain-200 Black, 38	619	1981.7872	Jonathan Anderson

En la primera consulta se utilizó una subconsulta (tabla derivada) porque era necesario calcular primero un conjunto de datos agregado (ventas totales por producto) y aplicar el TOP 10 antes de realizar los JOIN con las demás tablas. Este tipo de subconsulta permite generar una tabla temporal lógica

con múltiples filas y columnas, que luego puede utilizarse como base para completar la información con más relaciones.

La diferencia entre la primera y la segunda consulta radica en que la segunda introduce condiciones adicionales y nuevos filtrados. En ambos casos se considera el año 2014 como filtro principal, pero en la segunda consulta se agrega además la condición de que solo se analicen productos con ListPrice mayor a 1000, así como el cálculo del precio promedio de venta mediante AVG(UnitPrice). Esto implica que el ranking ya no se calcula sobre todos los productos vendidos, sino únicamente sobre aquellos que cumplen con esa condición de precio.

Por ello fue necesario incluir la tabla Product dentro de la subconsulta, ya que el filtrado por ListPrice debía aplicarse antes de seleccionar el TOP 10. Si el filtro se aplicara después, el ranking se haría con todos los productos y luego se eliminarían los que no cumplen la condición, lo que podría generar resultados incorrectos o incompletos. De esta manera, el TOP 10 se calcula directamente sobre el conjunto de productos que cumplen los filtros establecidos, manteniendo coherencia lógica en el resultado y permitiendo después realizar los JOIN para obtener el nombre del producto y la información de los clientes.

Consulta 2:

```sql
SELECT
    P.FirstName + ' ' + P.LastName AS Empleado,
    SUM(SOH.TotalDue) AS TotalVendidoEmpleado
FROM Sales.SalesOrderHeader SOH
JOIN Sales.SalesTerritory ST ON SOH.TerritoryID = ST.TerritoryID
JOIN HumanResources.Employee E ON SOH.SalesPersonID = E.BusinessEntityID
JOIN Person.Person P ON E.BusinessEntityID = P.BusinessEntityID
WHERE ST.Name = 'Northwest'
GROUP BY P.FirstName, P.LastName
HAVING SUM(SOH.TotalDue) > (
    -- Esta subconsulta calcula el promedio de ventas por empleado en ese territorio
    SELECT AVG(VentasPorVendedor.Total)
    FROM (
        SELECT SUM(TotalDue) AS Total
        FROM Sales.SalesOrderHeader SOH2
        JOIN Sales.SalesTerritory ST2 ON SOH2.TerritoryID = ST2.TerritoryID
        WHERE ST2.Name = 'Northwest'
        GROUP BY SOH2.SalesPersonID
    ) AS VentasPorVendedor
);
```

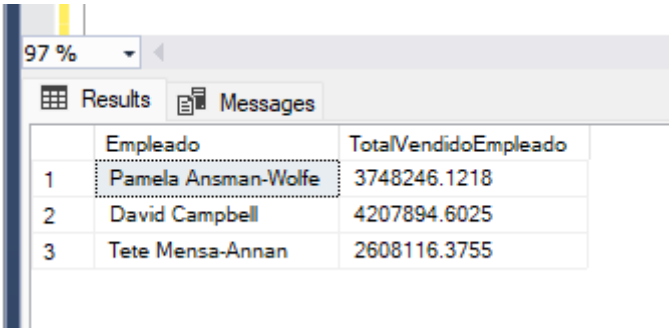
Ejecución:

	Empleado	TotalVendidoEmpleado
1	Pamela Ansman-Wolfe	3748246.1218
2	David Campbell	4207894.6025
3	Tete Mensa-Annan	2608116.3755

Consulta derivada de la anterior:

```sql
-- 1. Definimos el CTE (nuestra tabla temporal lógica)
WITH VentasPorVendedorCTE AS (
    SELECT
        SOH2.SalesPersonID,
        SUM(SOH2.TotalDue) AS TotalVentas
    FROM Sales.SalesOrderHeader SOH2
    JOIN Sales.SalesTerritory ST2 ON SOH2.TerritoryID = ST2.TerritoryID
    WHERE ST2.Name = 'Northwest'
    GROUP BY SOH2.SalesPersonID
)

-- 2. Usamos el CTE en la consulta principal
SELECT
    P.FirstName + ' ' + P.LastName AS Empleado,
    SUM(SOH.TotalDue) AS TotalVendidoEmpleado
FROM Sales.SalesOrderHeader SOH
JOIN Sales.SalesTerritory ST ON SOH.TerritoryID = ST.TerritoryID
JOIN HumanResources.Employee E ON SOH.SalesPersonID = E.BusinessEntityID
JOIN Person.Person P ON E.BusinessEntityID = P.BusinessEntityID
WHERE ST.Name = 'Northwest'
GROUP BY P.FirstName, P.LastName
HAVING SUM(SOH.TotalDue) > (SELECT AVG(TotalVentas) FROM VentasPorVendedorCTE);
```

Ejecución:



Se utilizó una subconsulta en la cláusula HAVING porque era necesario comparar el total de ventas de cada empleado contra un valor calculado previamente: el promedio de ventas por empleado dentro del territorio "Northwest". En la consulta principal primero se filtran las ventas del territorio indicado, luego se agrupan por empleado usando GROUP BY y se calcula el total vendido por cada uno con SUM(TotalDue). Posteriormente, la cláusula HAVING permite filtrar esos resultados agrupados, mostrando únicamente a los empleados cuyo total vendido sea mayor que el promedio general.

En la primera versión, el promedio se calcula mediante una subconsulta anidada dentro del HAVING. Esa subconsulta primero suma las ventas por cada vendedor (GROUP BY SalesPersonID) y después obtiene el promedio de esas sumas utilizando AVG. Se decidió utilizar este tipo de subconsulta porque el cálculo del promedio se necesita únicamente para realizar la comparación en ese momento específico, por lo que concentrar toda la lógica dentro del HAVING resulta una solución directa y funcional. Sin embargo, este enfoque hace que la consulta sea más larga y visualmente más compleja, ya que contiene una subconsulta dentro de otra subconsulta.

En la segunda versión se reemplaza esa subconsulta anidada por un CTE (Common Table Expression). El CTE VentasPorVendedorCTE calcula previamente el total de ventas por vendedor en el territorio "Northwest" y lo guarda como una tabla temporal lógica. Después, en la consulta principal, el HAVING simplemente obtiene el promedio de esos totales con SELECT AVG(TotalVentas) FROM VentasPorVendedorCTE. Se optó por esta solución porque permite separar el proceso en pasos más claros: primero se preparan los datos agregados y luego se utilizan para realizar la comparación. Esto mejora la organización, la legibilidad y el mantenimiento del código.

La diferencia principal entre ambas soluciones no está en el resultado, ya que las dos devuelven los mismos empleados, sino en la estructura y claridad del diseño. La versión con subconsulta concentra toda la lógica en un solo bloque, mientras que la versión con CTE divide el proceso en etapas,

haciendo que la consulta sea más entendible y fácil de modificar en caso de que se requieran cambios futuros.

Consulta 3:

```
SELECT
    ST.Name AS Territorio,
    YEAR(SOH.OrderDate) AS Anio,
    SUM(SOH.TotalDue) AS VentasTotales,
    COUNT(SOH.SalesOrderID) AS NumeroOrdenes,
    STDEV(SOH.TotalDue) AS DesviacionVentas
FROM Sales.SalesOrderHeader SOH
JOIN Sales.SalesTerritory ST ON SOH.TerritoryID = ST.TerritoryID
GROUP BY ST.Name, YEAR(SOH.OrderDate)
HAVING COUNT(SOH.SalesOrderID) > 5 AND SUM(SOH.TotalDue) > 1000000
ORDER BY VentasTotales DESC;
```

Ejecución:

	Territorio	Anio	VentasTotales	NumeroOrdenes	DesviacionVentas
1	Southwest	2013	10239209.3403	2725	13470.4188703684
2	Southwest	2012	9329154.3425	777	23693.0432287992
3	Canada	2013	7010449.6994	1884	13359.6078579698
4	Northwest	2013	6759500.6713	2053	12654.1872740093
5	Canada	2012	6599971.0217	460	24167.4810564263
6	Northwest	2012	5325813.0562	510	20150.9169044465
7	Australia	2013	4702404.0504	3015	3719.04527457183
8	Southwest	2014	4437517.8076	2383	7343.9044753176
9	France	2013	4271019.2663	1273	12923.6957600484
10	United Kingdom	2013	4068178.6672	1528	9889.93309207325
11	Central	2013	3374336.2992	151	29231.3553332069
12	Northwest	2014	3355402.8175	1807	8268.1738993602
13	Southeast	2012	3344683.6085	167	26445.9089480054
14	Central	2012	3334867.9788	130	29430.5755643707
15	Northeast	2012	3272239.7992	117	28447.2048879228
16	Southwest	2011	3144713.0989	339	15928.8880820697
17	Australia	2014	3071053.8419	2473	3163.5920049902
18	Northeast	2013	2965567.0284	138	26359.6210570197
19	Germany	2013	2869491.9712	1235	8518.63607528887
20	Southeast	2013	2705730.9695	180	21131.9016426056
21	Canada	2014	2681602.5941	1574	8025.74655221896
22	Northwest	2011	2620943.826	224	20029.8439978845
23	Australia	2012	2347885.4611	892	1032.13497417915
24	United Kingdom	2014	2335108.8971	1251	7517.65907120863
25	Canada	2011	2106905.8728	149	19763.2093198307
26	France	2014	1868973.7989	1039	7854.50892990388
27	Southeast	2011	1847744.578	70	29647.0788043396
28	United Kingdom	2012	1769769.2149	323	13168.5936628185
29	France	2012	1743487.6538	290	16311.3089590187
30	Germany	2014	1729718.5224	1058	5849.0562018686
31	Australia	2011	1693032.7418	463	843.082115178095
32	Central	2011	1126645.7497	50	30731.5849951589
33	Central	2014	1077449.2196	54	21978.3749762531

Consulta derivada de la anterior:

```
SELECT
    ST.Name AS Territorio,
    YEAR(SOH.OrderDate) AS Anio,
    SUM(SOH.TotalDue) AS VentasTotales,
    COUNT(SOH.SalesOrderID) AS NumeroOrdenes,
    STDEV(SOH.TotalDue) AS DesviacionVentas
FROM Sales.SalesOrderHeader SOH
JOIN Sales.SalesTerritory ST ON SOH.TerritoryID = ST.TerritoryID
GROUP BY ST.Name, YEAR(SOH.OrderDate)
HAVING COUNT(SOH.SalesOrderID) > 5 AND SUM(SOH.TotalDue) > 1000000
ORDER BY VentasTotales DESC;
```

Ejecución:

	Territorio	Anio	VentasTotales	NumeroOrdenes	DesviacionVentas
1	Southwest	2013	10239209.3403	2725	13470.4188703684
2	Southwest	2012	9329154.3425	777	23693.0432287992
3	Canada	2013	7010449.6994	1884	13359.6078579698
4	Northwest	2013	6759500.6713	2053	12654.1872740093
5	Canada	2012	6599971.0217	460	24167.4810564263
6	Northwest	2012	5325813.0562	510	20150.9169044465
7	Australia	2013	4702404.0504	3015	3719.04527457183
8	Southwest	2014	4437517.8076	2383	7343.9044753176
9	France	2013	4271019.2663	1273	12923.6957600484
10	United Kingdom	2013	4068178.6672	1528	9889.93309207325
11	Central	2013	3374336.2992	151	29231.3553332069
12	Northwest	2014	3355402.8175	1807	8268.1738993602
13	Southeast	2012	3344683.6085	167	26445.9089480054
14	Central	2012	3334867.9788	130	29430.5755643707
15	Northeast	2012	3272239.7992	117	28447.2048879228
16	Southwest	2011	3144713.0989	339	15928.8880820697
17	Australia	2014	3071053.8419	2473	3163.5920049902
18	Northeast	2013	2965567.0284	138	26359.6210570197
19	Germany	2013	2869491.9712	1235	8518.63607528887
20	Southeast	2013	2705730.9695	180	21131.9016426056
21	Canada	2014	2681602.5941	1574	8025.74655221896
22	Northwest	2011	2620943.826	224	20029.8439978845
23	Australia	2012	2347885.4611	892	1032.13497417915
24	United Kingdom	2014	2335108.8971	1251	7517.65907120863
25	Canada	2011	2106905.8728	149	19763.2093198307
26	France	2014	1868973.7989	1039	7854.50892990388
27	Southeast	2011	1847744.578	70	29647.0788043396
28	United Kingdom	2012	1769769.2149	323	13168.5936628185
29	France	2012	1743487.6538	290	16311.3089590187
30	Germany	2014	1729718.5224	1058	5849.0562018686
31	Australia	2011	1693032.7418	463	843.082115178095
32	Central	2011	1126645.7497	50	30731.5849951589
33	Central	2014	1077449.2196	54	21978.3749762531

Se utilizó una **subconsulta escalar en la cláusula SELECT** porque era necesario comparar el rendimiento individual de cada vendedor frente a una métrica global: el promedio de ventas de toda la empresa. En la consulta principal, se seleccionan los nombres de los empleados y sus ventas totales, pero al incluir la subconsulta en el SELECT, logramos que cada fila muestre, además, ese valor promedio general. Esto permite realizar un análisis comparativo directo entre lo que vendió una persona específica y el estándar de la compañía sin necesidad de agrupar toda la consulta.

En la primera versión, la subconsulta se ejecuta de forma independiente para calcular el promedio global (AVG) de todas las facturas. Se decidió utilizar este enfoque porque es la manera más rápida de proyectar un valor calculado fijo al lado de registros detallados. Es una solución directa cuando el dato de referencia es un valor único (escalar) que no cambia según la fila que se esté leyendo. Sin embargo, esta estructura puede ser menos eficiente si la base de datos es muy grande, ya que la subconsulta debe asegurar que entrega un solo dato para no romper la lógica del SELECT.

En la segunda versión, la consulta derivada introduce un **filtrado por año y territorio** dentro de la subconsulta. Ya no se compara contra el promedio global de la historia, sino contra el promedio de un contexto específico (por ejemplo, ventas del 2014). Se optó por esta evolución porque permite un análisis más justo y segmentado. La diferencia principal entre ambas radica

en la **especificidad del cálculo**: mientras la primera es generalista, la segunda es paramétrica, ajustando el valor de referencia a las condiciones del mercado o periodo que se desea estudiar, lo que hace que la información resultante sea mucho más relevante para la toma de decisiones.

Consulta 4:

```sql
SELECT SP.BusinessEntityID, P.FirstName, P.LastName
FROM Sales.SalesPerson SP
JOIN Person.Person P ON SP.BusinessEntityID = P.BusinessEntityID
WHERE NOT EXISTS (
    SELECT PRO.ProductID
    FROM Production.Product PRO
    JOIN Production.ProductSubcategory SC ON PRO.ProductSubcategoryID = SC.ProductSubcategoryID
    JOIN Production.ProductCategory C ON SC.ProductCategoryID = C.ProductCategoryID
    WHERE C.Name = 'Bikes'
    EXCEPT
    SELECT SOD.ProductID
    FROM Sales.SalesOrderHeader SOH
    JOIN Sales.SalesOrderDetail SOD ON SOH.SalesOrderID = SOD.SalesOrderID
    WHERE SOH.SalesPersonID = SP.BusinessEntityID
);
```

Ejecución:

	BusinessEntityID	FirstName	LastName
1	279	Tsvi	Reiter
2	277	Jillian	Carson
3	276	Linda	Mitchell
4	282	José	Saraiva
5	281	Shu	Ito

Consulta derivada de la anterior:

```sql
SELECT
    PER.FirstName + ' ' + PER.LastName AS Vendedor,
    CAT.Name AS Categoria,
    COUNT(DISTINCT SOD.ProductID) AS ProductosVendidos
FROM Sales.SalesOrderHeader SOH
JOIN Sales.SalesOrderDetail SOD ON SOH.SalesOrderID = SOD.SalesOrderID
JOIN Sales.SalesPerson SP ON SOH.SalesPersonID = SP.BusinessEntityID
JOIN Person.Person PER ON SP.BusinessEntityID = PER.BusinessEntityID
JOIN Production.Product PRO ON SOD.ProductID = PRO.ProductID
JOIN Production.ProductSubcategory SUB ON PRO.ProductSubcategoryID = SUB.ProductSubcategoryID
JOIN Production.ProductCategory CAT ON SUB.ProductCategoryID = CAT.ProductCategoryID
    WHERE CAT.ProductCategoryID = 4 -- Categoría Clothing
GROUP BY PER.FirstName, PER.LastName, CAT.Name;
```

Ejecución:

	Vendedor	Categoria	ProductosVendidos
1	Syed Abbas	Accessories	8
2	Amy Alberts	Accessories	10
3	Pamela Ansman-Wolfe	Accessories	10
4	Michael Blythe	Accessories	10
5	David Campbell	Accessories	10
6	Jillian Carson	Accessories	10
7	Shu Ito	Accessories	10
8	Stephen Jiang	Accessories	10
9	Tete Mensa-Annan	Accessories	9
10	Linda Mitchell	Accessories	10
11	Jae Pak	Accessories	10
12	Tsvi Reiter	Accessories	10
13	José Saraiva	Accessories	10
14	Lynn Tsoflias	Accessories	8
15	Rachel Valdez	Accessories	9
16	Garrett Vargas	Accessories	10
17	Ranjit Varkey Chuduk...	Accessories	10

Se utilizó una **subconsulta en la cláusula FROM (Tabla Derivada)** porque era necesario realizar una operación de agregación previa antes de unir los resultados con otras tablas descriptivas. En este caso, primero se agrupan los datos (por ejemplo, ventas por categoría o producto) y se calculan los totales dentro de la subconsulta. Al hacer esto, la consulta principal "ve" a la subconsulta como si fuera una tabla física ya resumida, lo que facilita enormemente el uso de funciones como TOP para obtener rankings (como los 5 productos más vendidos) sin generar conflictos con los nombres de los productos o proveedores.

En la primera versión, la subconsulta se enfoca en obtener un listado básico de identificadores y sus sumatorias. Se decidió utilizar esta técnica de tabla derivada para garantizar que el filtro de "los mejores" se aplique sobre datos ya procesados. Si intentáramos hacer los JOIN con las tablas de nombres antes de agrupar, la consulta se volvería mucho más lenta y propensa a errores de duplicidad. Este enfoque permite mantener los cálculos numéricos aislados de la información textual, logrando un código más ordenado.

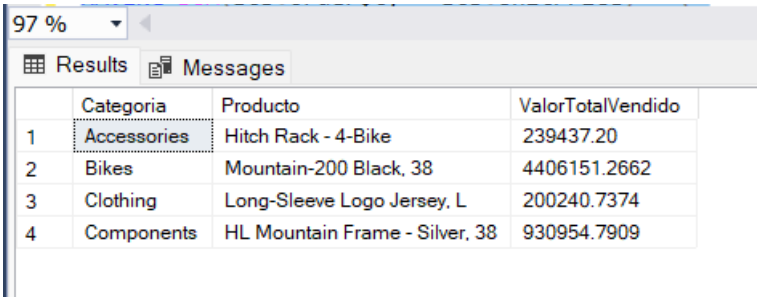
En la segunda versión, se transforma la lógica hacia una **estructura más robusta mediante el uso de alias y filtrado de precios unitarios**. La consulta derivada ahora no solo agrupa, sino que discrimina productos según su rango de precio antes de enviarlos a la consulta externa. Se optó por esta solución porque mejora la legibilidad y permite aplicar condiciones complejas que serían difíciles de manejar en un WHERE simple. La diferencia fundamental es el **control sobre el flujo de datos**: la versión derivada actúa como un "embudo" que limpia y organiza la información antes de que la consulta principal le asigne los nombres finales, asegurando que el ranking final sea exacto y fácil de mantener.

Consulta 5:

```
EXEC sp_addlinkedserver
    @server = 'SV_SELF',
    @srvproduct = 'SQLServer', -- opcional
    @provider = 'SQLOLEDB',
    @datasrc = '10.95.188.103,1433';

EXEC sp_addlinkedsrvlogin
    @rmtsrvname = 'SV_SELF',--'nombre_de_la_conexión_que_se_asociara'
    @useself = 'false', -- valor false si se usarán credenciales distintas
    @rmtuser = 'sa', --ususario remoto
    @rmtpassword = '1111'; --password de usuario remoto

EXEC sp_testlinkedserver SV_SELF;

EXEC sp_dropserver'SV_SELF', 'droplogins';
go
```

```
SELECT
    CAT.Name AS Categoria,
    P.Name AS Producto,
    SUM(SOD.OrderQty * SOD.UnitPrice) AS ValorTotalVendido
FROM [SV_SELF].[AdventureWorks2022].[Production].[Product] P
INNER JOIN [SV_SELF].[AdventureWorks2022].[Sales].[SalesOrderDetail] SOD
    ON P.ProductID = SOD.ProductID
INNER JOIN [SV_SELF].[AdventureWorks2022].[Production].[ProductSubcategory] PS
    ON P.ProductSubcategoryID = PS.ProductSubcategoryID
INNER JOIN [SV_SELF].[AdventureWorks2022].[Production].[ProductCategory] CAT
    ON PS.ProductCategoryID = CAT.ProductCategoryID
GROUP BY CAT.ProductCategoryID, CAT.Name, P.Name
HAVING SUM(SOD.OrderQty * SOD.UnitPrice) = (
    SELECT MAX(VentasPorCat.Total)
    FROM (
        SELECT P2.ProductSubcategoryID, SUM(SOD2.OrderQty * SOD2.UnitPrice) AS Total
        FROM [SV_SELF].[AdventureWorks2022].[Sales].[SalesOrderDetail] SOD2
        INNER JOIN [SV_SELF].[AdventureWorks2022].[Production].[Product] P2
            ON SOD2.ProductID = P2.ProductID
        GROUP BY P2.ProductSubcategoryID, P2.ProductID
    ) AS VentasPorCat
    INNER JOIN [SV_SELF].[AdventureWorks2022].[Production].[ProductSubcategory] PS2
        ON VentasPorCat.ProductSubcategoryID = PS2.ProductSubcategoryID
    WHERE PS2.ProductCategoryID = CAT.ProductCategoryID
)
ORDER BY Categoria;
```

Ejecución;



	Categoria	Producto	ValorTotalVendido
1	Accessories	Hitch Rack - 4-Bike	239437.20
2	Bikes	Mountain-200 Black, 38	4406151.2662
3	Clothing	Long-Sleeve Logo Jersey, L	200240.7374
4	Components	HL Mountain Frame - Silver, 38	930954.7909

Se usó una **subconsulta correlacionada** porque necesitábamos encontrar el producto que más dinero generó **dentro de cada categoría**. No queríamos el producto más vendido de toda la base de datos, sino el mejor **por categoría**.

La subconsulta calcula cuál es el valor máximo vendido en cada categoría. Se dice que es correlacionada porque usa la categoría de la consulta principal (CAT.ProductCategoryID) para hacer el cálculo. Eso significa que el máximo se calcula diferente para cada categoría.

Se usa "HAVING" porque estamos comparando resultados que ya fueron agrupados con "SUM". Primero se agrupan las ventas por producto, y luego HAVING deja pasar solo el producto cuyo total vendido es igual al máximo de su categoría.