Instituto Politécnico Nacional

Unidad Profesional Interdisciplinaria en

Ingeniería y Tecnologías Avanzadas

Alumnos:

Hernández Arteaga Itzel

García Ortiz Martha Lesly

Profesor: De La Cruz Sosa Carlos

Carrera: Telemática

Materia: Bases de Datos Distribuidas

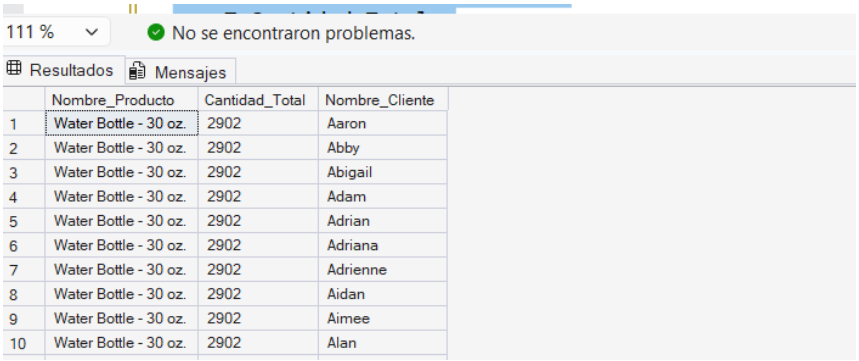
Grupo: 3TM2

Practica 1: Repaso de consultas SQL y uso de servidores vinculados

Fecha: 24/02/2026

Ejercicio 1. Encuentra los 10 productos más vendidos en 2014, mostrando nombre del producto, cantidad total vendida y nombre del cliente.

```
WITH PRODUCTOS AS (
   SELECT TOP 10
    V.ProductID,
   SUM(V.OrderQty) AS Cantidad_Total
   FROM Sales.SalesOrderDetail AS V
   JOIN Production.Product AS P
   ON V.ProductID = P.ProductID
   JOIN Sales.SalesOrderHeader AS H
   ON H.SalesOrderID = V.SalesOrderID
   WHERE YEAR(H.OrderDate) = 2014
   GROUP BY V.ProductID
   ORDER BY SUM(V.OrderQty) DESC
)
SELECT DISTINCT
   P.Name AS Nombre_Producto,
   T.Cantidad_Total,
   E.FirstName AS Nombre_Cliente
FROM PRODUCTOS AS T
JOIN Production.Product AS P
 ON T.ProductID = P.ProductID
JOIN Sales.SalesOrderDetail AS V
 ON T.ProductID = V.ProductID
JOIN Sales.SalesOrderHeader AS H
 ON V.SalesOrderID = H.SalesOrderID
JOIN Sales.Customer AS C
 ON H.CustomerID = C.CustomerID
JOIN Person.Person AS E
 ON C.PersonID = E.BusinessEntityID
ORDER BY T.Cantidad_Total DESC;
```



	Nombre_Producto	Cantidad_Total	Nombre_Cliente
1	Water Bottle - 30 oz.	2902	Aaron
2	Water Bottle - 30 oz.	2902	Abby
3	Water Bottle - 30 oz.	2902	Abigail
4	Water Bottle - 30 oz.	2902	Adam
5	Water Bottle - 30 oz.	2902	Adrian
6	Water Bottle - 30 oz.	2902	Adriana
7	Water Bottle - 30 oz.	2902	Adrienne
8	Water Bottle - 30 oz.	2902	Aidan
9	Water Bottle - 30 oz.	2902	Aimee
10	Water Bottle - 30 oz.	2902	Alan

Fig1. Se realizo una subconsulta no correlacionada ya que se utilizó la sentencia WITH
Lo primero que se realizo fue la subconsulta de los productos más vendidos en el 2014, se utilizaron 2 JOIN'S para poder tener el nombre de todos los productos y después filtrar con el año y después agrupar esos datos con la cantidad total de productos vendidos. Esta es nuestra tabla donde aparece los 10 productos más vendidos del 2014.Después se realizó la segunda subconsulta de los clientes, se utilizaron varios JOIN´S para sacar los nombres de los clientes, se utilizó DISTINCT para eliminar clientes repetidos, ya solo se ocupo la CTE Productos para juntas ambas subconsultas.

Una vez resuelta la consulta: agrega el precio unitario promedio (AVG(UnitPrice)) y filtra solo productos con ListPrice > 1000.

```
WITH TopProductos AS (
SELECT TOP 10 V.ProductID, SUM(V.OrderQty) AS Cantidad_Total,
AVG(V.UnitPrice) AS Precio_Unitario FROM Sales.SalesOrderDetail AS V
JOIN Production.Product AS P ON V.ProductID = P.ProductID
JOIN Sales.SalesOrderHeader AS H ON H.SalesOrderID = V.SalesOrderID
WHERE YEAR(H.OrderDate) = 2014 AND P.ListPrice > 1000
GROUP BY V.ProductID ORDER BY SUM(V.OrderQty) DESC)
SELECT DISTINCT P.Name AS Nombre_Producto, T.Cantidad_Total,
T.Precio_Unitario, E.FirstName AS Nombre_Cliente FROM TopProductos AS T
JOIN Production.Product AS P ON T.ProductID = P.ProductID
JOIN Sales.SalesOrderDetail AS V ON T.ProductID = V.ProductID
JOIN Sales.SalesOrderHeader AS H
ON V.SalesOrderID = H.SalesOrderID
JOIN Sales.Customer AS C
ON H.CustomerID = C.CustomerID
JOIN Person.Person AS E
ON C.PersonID = E.BusinessEntityID
ORDER BY T.Cantidad_Total DESC;
```

111 % No se encontraron problemas.

Resultados Mensajes

	Nombre_Producto	Cantidad_Total	Precio_Unitario	Nombre_Cliente
1	Mountain-200 Black, 38	619	1981.7872	Aaron
2	Mountain-200 Black, 38	619	1981.7872	Abigail
3	Mountain-200 Black, 38	619	1981.7872	Adam
4	Mountain-200 Black, 38	619	1981.7872	Aidan
5	Mountain-200 Black, 38	619	1981.7872	Alan
6	Mountain-200 Black, 38	619	1981.7872	Alberto
7	Mountain-200 Black, 38	619	1981.7872	Alejandro
8	Mountain-200 Black, 38	619	1981.7872	Alexa
9	Mountain-200 Black, 38	619	1981.7872	Alexandra
10	Mountain-200 Black, 38	619	1981.7872	Alexandria

Fig2. Solo se agregó la línea que se pido en la primera subconsulta dentro de SELECT para realizar la operación correspondiente y la condición dentro de la sentencia WHERE, también se agrego la columna de Precio unitario en la segunda subconsulta para que se pueda visualizar la columna

Ejercicio 2: Lista los empleados que han vendido más que el promedio de ventas por empleado en el territorio 'Northwest'. 1. Requisito adicional: aplicar subconsultas.

```
selec p.FirstName, p.LastName,  e.SalesYTD
from Sales.SalesPerson AS e inner join
   Person.Person AS p ON e.BusinessEntityID = p.BusinessEntityID
   where  e.SalesYTD> (select  AVG(sp.SalesYTD) from Sales.SalesPerson as sp
   inner join Sales.SalesTerritory as st ON sp.TerritoryID=st.TerritoryID
   where  st.Name='Northwest')
   and e.TerritoryID = (SELECT TerritoryID FROM Sales.SalesTerritory WHERE
Name = 'Northwest');
```

2. Una vez resuelta la consulta convierte la subconsulta en un CTE (Common Table Expresión).

```
   WITH Promedio as (select  AVG(sp.SalesYTD) ValorPromedio from
Sales.SalesPerson as sp
   INNER JOIN Sales.SalesTerritory as st ON sp.TerritoryID=st.TerritoryID
   where  st.Name='Northwest'  )
   SELECT   p.FirstName, p.LastName,  e.SalesYTD
FROM Sales.SalesPerson AS e INNER JOIN
   Person.Person AS p ON e.BusinessEntityID = p.BusinessEntityID
   where  e.SalesYTD> (select  ValorPromedio from Promedio)/*CTE*/
     AND e.TerritoryID = (SELECT TerritoryID FROM Sales.SalesTerritory WHERE
Name = 'Northwest');
```

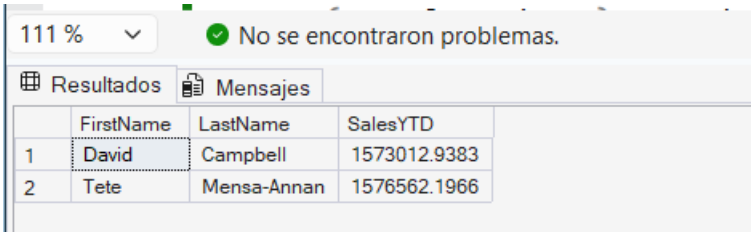


Fig3. El objetivo fue identificar a los vendedores que superaron el rendimiento promedio del territorio **'Northwest'**. El mayor reto aquí es que el "promedio" no es un número fijo, sino que SQL debe calcularlo primero para poder comparar a cada empleado contra él.

Ejercicio 3: Calcula ventas totales por territorio y año, mostrando solo aquellos con más de 5 órdenes y ventas > $1,000,000, ordenado por ventas descendente.

```
select
year(ve.OrderDate) as Fecha,
te.Name Territorio,
COUNT(ve.SalesOrderID) as OrdenesTotales,
SUM(ve.TotalDue) as TotalVentas
from Sales.SalesOrderHeader as ve
inner join Sales.SalesTerritory as te on ve.TerritoryID=te.TerritoryID
GROUP BY YEAR(ve.OrderDate), te.Name
HAVING COUNT(ve.SalesOrderID)>'5' AND SUM(ve.TotalDue)>'1000000'
order by totalVentas DESC;
```

Resultados | Mensajes

	Fecha	Territorio	OrdenesTotales	TotalVentas
1	2013	Southwest	2725	10239209.3403
2	2012	Southwest	777	9329154.3425
3	2013	Canada	1884	7010449.6994
4	2013	Northwest	2053	6759500.6713
5	2012	Canada	460	6599971.0217
6	2012	Northwest	510	5325813.0562
7	2013	Australia	3015	4702404.0504
8	2014	Southwest	2383	4437517.8076
9	2013	France	1273	4271019.2663
10	2013	United Kingdom	1528	4068178.6672

Fig4. Aquí el objetivo fue agrupar miles de órdenes individuales en un reporte resumido por región y año, aplicando filtros de "calidad" a esos resultados finales.

Agrupamiento: Usamos GROUP BY sobre el **Año** (extraído con YEAR) y el **Nombre del Territorio**. Esto "colapsa" todas las filas en totales resumidos.

1. Una vez resuelta la consulta agrega desviación estándar de ventas

```
select
year(ve.OrderDate) as Fecha,
te.Name Territorio,
COUNT(ve.SalesOrderID) as OrdenesTotales,
SUM(ve.TotalDue) as TotalVentas,
STDEV(ve.TotalDue) AS DesviacionVentas
from Sales.SalesOrderHeader as ve
inner join Sales.SalesTerritory as te on ve.TerritoryID=te.TerritoryID
GROUP BY YEAR(ve.OrderDate), te.Name
```

Resultados | Mensajes

	Fecha	Territorio	OrdenesTotales	TotalVentas	DesviacionVentas
1	2013	Southwest	2725	10239209.3403	13470.4188703684
2	2012	Southwest	777	9329154.3425	23693.0432287992
3	2013	Canada	1884	7010449.6994	13359.6078579698
4	2013	Northwest	2053	6759500.6713	12654.1872740093
5	2012	Canada	460	6599971.0217	24167.4810564263
6	2012	Northwest	510	5325813.0562	20150.9169044465
7	2013	Australia	3015	4702404.0504	3719.04527457183
8	2014	Southwest	2383	4437517.8076	7343.9044753176
9	2013	France	1273	4271019.2663	12923.6957600484
10	2013	United Kingdom	1528	4068178.6672	9889.93309207325

Fig5. **Desviación Estándar (Variante):** Añadimos STDEV(TotalDue) para medir la consistencia. Una desviación alta indica que hubo ventas muy variadas (unas muy pequeñas y otras gigantes), mientras que una baja indica ventas muy similares entre sí.

Ejercicio 4: Encuentra vendedores que han vendido TODOS los productos de la categoría "Bikes". 1. Cambia a categoría "Clothing" (ID=4). 2. Cuenta cuántos productos por categoría maneja cada vendedor.

```
SELECT DISTINCT Per.FirstName AS Vendedor
FROM Sales.SalesPerson SP
JOIN Person.Person Per
ON SP.BusinessEntityID = Per.BusinessEntityID
WHERE NOT EXISTS
(SELECT P.ProductID
FROM Production.Product P
JOIN Production.ProductSubcategory PS
ON P.ProductSubcategoryID = PS.ProductSubcategoryID
JOIN Production.ProductCategory PC
ON PS.ProductCategoryID = PC.ProductCategoryID
WHERE PC.Name = 'Bikes'
AND NOT EXISTS
(SELECT *
FROM Sales.SalesOrderHeader H
JOIN Sales.SalesOrderDetail D
ON H.SalesOrderID = D.SalesOrderID
WHERE H.SalesPersonID = SP.BusinessEntityID
AND D.ProductID = P.ProductID)
);
```

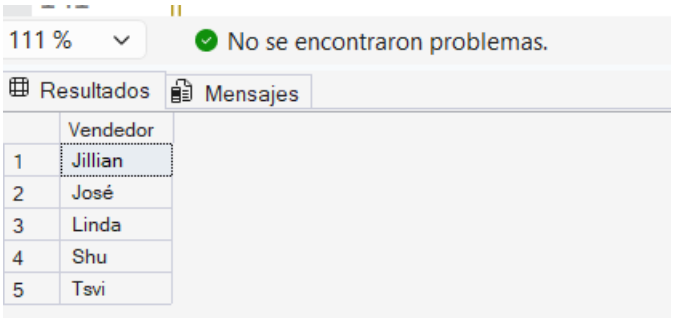


Fig6. De acuerdo a que se pide identificar que los valores se emparejen con TODOS se realizó una división relacional

En la subconsulta externa se utiliza un join para identificar a todos los vendedores que cumplan con la condición de nuestra segunda subconsulta

En la segunda subconsulta también hacemos uso de JOIN ´S para la búsqueda de los productos que sean de la categoría “Bikes”.

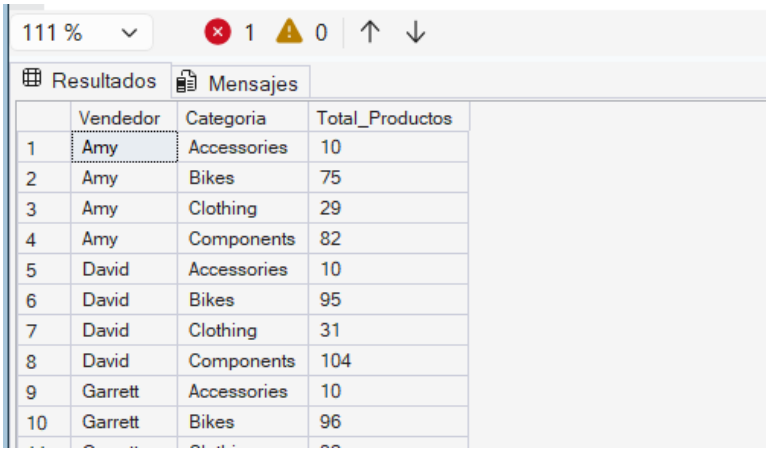
En la tercera subconsulta también ocupamos los JOIN ´S para verificar si el vendedor vendió ese producto. Con el primer NOT EXISTS podemos confirmar que productos de esa categoría no se han vendido y con el segundo NOT EXISTS si vendió el producto dela categoría “Bikes”

Cambia a categoría "Clothing" (ID=4).

WHERE PC.Name = 'Accsessories'

Cuenta cuántos productos por categoría maneja cada vendedor.

```
SELECT
  Per.FirstName AS Vendedor,
  PC.Name AS Categoria,
  COUNT(DISTINCT P.ProductID) AS Total_Productos
FROM Sales.SalesPerson SP
JOIN Person.Person Per
  ON SP.BusinessEntityID = Per.BusinessEntityID
JOIN Sales.SalesOrderHeader H
  ON H.SalesPersonID = SP.BusinessEntityID
JOIN Sales.SalesOrderDetail D
  ON D.SalesOrderID = H.SalesOrderID
JOIN Production.Product P
  ON P.ProductID = D.ProductID
JOIN Production.ProductSubcategory PS
  ON PS.ProductSubcategoryID = P.ProductSubcategoryID
JOIN Production.ProductCategory PC
  ON PC.ProductCategoryID = PS.ProductCategoryID
GROUP BY
  Per.FirstName,
  PC.Name
ORDER BY
  Per.FirstName,
  PC.Name;
```

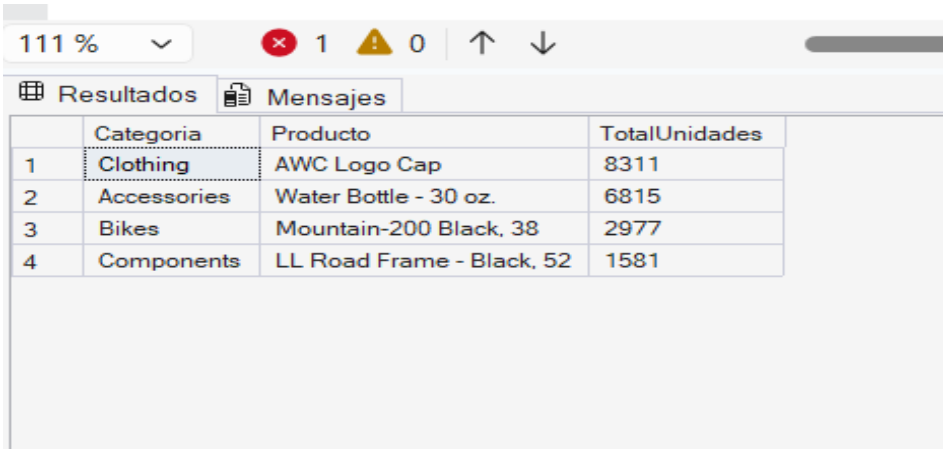


	Vendedor	Categoria	Total_Productos
1	Amy	Accessories	10
2	Amy	Bikes	75
3	Amy	Clothing	29
4	Amy	Components	82
5	David	Accessories	10
6	David	Bikes	95
7	David	Clothing	31
8	David	Components	104
9	Garrett	Accessories	10
10	Garrett	Bikes	96

Fig7. En esta consulta primero se contó los productos de cada categoría sin repetir datos después se utilizó un GROUP BY donde primero con JOIN'S se fue buscando cada vendedor y cada categoría, después estos datos se agruparon con el nombre del vendedor.

Ejercicio 5: Determinar el producto más vendido de cada categoría de producto, considerando el escenario de que el esquema SALES se encuentra en una instancia (servidor) A y el esquema PRODUCTION en otra instancia (servidor) B.

```
  SELECT * FROM SV_LES.master.sys.databases;
  WITH VentasRanking AS (
   SELECT   cat.Name AS Categoria, prod.Name AS Producto,
     SUM(det.OrderQty) AS TotalUnidades,   ROW_NUMBER() OVER(
        PARTITION BY cat.Name
        ORDER BY SUM(det.OrderQty) DESC
     ) AS TopRanking
   FROM Sales.SalesOrderDetail AS det
   -- Tablas en el "Servidor B" (usando SV_SELF)
   INNER JOIN SV_LES.AdventureWorks2022.Production.Product AS prod
     ON det.ProductID = prod.ProductID
   INNER JOIN SV_LES.AdventureWorks2022.Production.ProductSubcategory AS sub
     ON prod.ProductSubcategoryID = sub.ProductSubcategoryID
   INNER JOIN SV_LES.AdventureWorks2022.Production.ProductCategory AS cat
     ON sub.ProductCategoryID = cat.ProductCategoryID
   GROUP BY cat.Name, prod.Name
)
SELECT
   Categoria,
   Producto,
   TotalUnidades
FROM VentasRanking
WHERE TopRanking = 1
ORDER BY TotalUnidades DESC;
```



	Categoria	Producto	TotalUnidades
1	Clothing	AWC Logo Cap	8311
2	Accessories	Water Bottle - 30 oz.	6815
3	Bikes	Mountain-200 Black, 38	2977
4	Components	LL Road Frame - Black, 52	1581

Fig8.Creamos una conexión entre dos instancias de SQL Server (local y remota) para que pudieran "hablarse". Implementamos la sintaxis Servidor.Base.Esquema.Tabla para consultar datos que no están en nuestra base de datos local