



UNIVERSIDAD PRIVADA DE TACNA

FACULTAD DE INGENIERIA

Escuela Profesional de Ingeniería de Sistemas

INFORME DE SEMANA N°01

**“LABORATORIO N° 01
UTILIZANDO EXPRESIONES DE TABLA”**

Curso: BASE DE DATOS II

Docente: Mag. Ing. Patrick Cuadros Quiroga

Robles Flores, Anthony Richard (2016056192)

**Tacna – Perú
2019**



INDICE

I. INFORMACIÓN GENERAL	3
- Objetivos:	3
- Equipos, materiales, programas y recursos utilizados:	3
II. MARCO TEORICO	3
III. PROCEDIMIENTO	4
IV. CUESTIONARIO	13
CONCLUSIONES	13



LABORATORIO N° 02

UTILIZANDO OPERADORES DE CONJUNTOS DE DATOS

I. INFORMACIÓN GENERAL

- Objetivos:

Utilizando Aure Data Studio para poder realizar las consultas que se verán en este informe:

- Utilización de la base de datos TSQL.
- Ejecución de consultas.
- Analizar y realizar consultas bajas medias y avanzadas.

- Equipos, materiales, programas y recursos utilizados:

Para poder realizar las consultas que se observaran se utilizara lo siguiente:

- Computadora con sistema operativo Windows que soporte SQL SERVER 2016.
- Software “Azure Data Studio”.
- Explicación de docente del curso

II. MARCO TEORICO

EXPRESIONES DE TABLA COMUNES

SQL Server nos ofrece una funcionalidad bastante interesante llamada Common Table Expressions o CTEs. Se trata de una manera de definir un conjunto de datos temporal (como una tabla temporal) que sólo pervive mientras se ejecute nuestra consulta y no se almacena en ningún sitio. Sin embargo, nos permite consultarla y trabajar con ella como si fuese una tabla real de la base de datos. Estas CTE pueden hacer referencia a sí mismas (algo súper-potente) y se puede usar varias veces en la misma consulta.

Además de emplear en consultas recursivas se pueden utilizar, entre otras cosas, para:

- ✓ Filtrar de manera sencilla por campos que antes no existían
- ✓ Sustituir el uso de vistas, que muchas veces no son necesarias
- ✓ Referenciar a la misma tabla varias veces en la misma expresión
- ✓ Agrupar o filtrar por campos que se derivan de subconsultas
- ✓ Agrupar o filtrar por campos que resultan de funciones que no son deterministas (es decir, que devuelven un valor diferente cada vez que se llaman, aún pasándoles los mismos argumentos).

Una expresión de tabla común o Common Table Expression (CTE) es un conjunto de resultados que se definen en tiempo de ejecución, tiene un concepto similar a una Tabla Derivada:

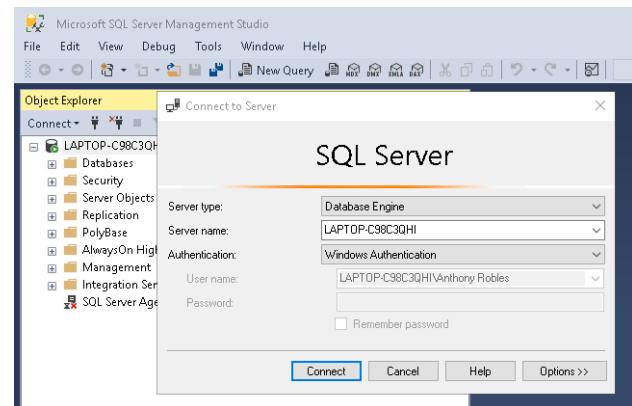
```
WITH <nombreCTE>
AS
(
    <consulta_interna>
)
<consulta_externa>
```

III. PROCEDIMIENTO

Comencemos asegurando que nuestro SQL pueda ingresar correctamente.

Paso 1:

- Abrir SQL Server tomando en cuenta que en los servicios de Windows se encuentre iniciado.



Paso 2:

- Abriremos Azure DataStudio tomando en cuenta que el usuario de ingreso es el mismo con el que se ingresa a SQL SERVER.
- Conectar con la BD TSQL

```
[2] 1 USE TSQL;
     2 GO
```



Paso 3:

- Escriba una instrucción SELECT para devolver las columnas productid, productname, supplierid, unitprice y discontinued de la tabla Production.Products. Filtre los resultados para incluir solo los productos que pertenecen a la categoría Bebidas (categoryid es igual a 1)

```
1 select productid , productname , supplierid ,  
2 unitprice , discontinued from Production.Products  
3 where categoryid=1
```

	productid	productname	supplierid	unitprice	discontinued
1	1	Product HHYDP	1	18,0000	0
2	2	Product RECZE	1	19,0000	0
3	24	Product QOGNU	10	4,5000	1
4	34	Product SWNJY	16	14,0000	0
5	35	Product NEVTJ	16	18,0000	0
6	38	Product QDOMO	18	263,5000	0

Paso 4:

- Modifique el código T-SQL para incluir la siguiente instrucción T-SQL suministrada. Ponga esta declaración antes de la cláusula SELECT:

```
1 CREATE VIEW Production.ProductsBeverages  
2 AS select productid , productname , supplierid ,  
3 unitprice , discontinued from Production.Products
```

Paso 5:

- Escriba una instrucción SELECT para devolver las columnas productid y productname de la vista Production.ProductsBeverages. Filtre los resultados para incluir solo productos en los que supplierid sea igual a 1

```
1 select productid , productname  
2 from Production.ProductsBeverages where supplierid = 1
```



	productid	productname
1	1	Product HHYDP
2	2	Product RECZE
3	3	Product IMEHJ

Paso 6:

- El departamento de TI ha escrito una declaración T-SQL que agrega una columna calculada adicional al

```
1 ALTER VIEW Production.ProductsBeverages AS
2 SELECT
3 productid, productname, supplierid, unitprice, discontinued,
4 CASE WHEN unitprice > 100. THEN N'high' ELSE N'normal' END
5 as Resultado
6 FROM Production.Products
7 WHERE categoryid = 1;
```

Paso 7:

- Aplique los cambios necesarios para que la instrucción T-SQL se ejecute correctamente

```
1 select * from Production.ProductsBeverages
```

	productid	productname	supplierid	unitprice	discontinued	R
1	1	Product HHYDP	1	18,0000	0	
2	2	Product RECZE	1	19,0000	0	
3	24	Product QOGNU	10	4,5000	1	
4	34	Product SWNJY	16	14,0000	0	
5	35	Product NEVTJ	16	18,0000	0	
6	38	Product QDOMO	18	263,5000	0	
7	39	Product LSOFL	18	18,0000	0	
8	43	Product ZZZHR	20	46,0000	0	
9	67	Product XLXQF	16	14,0000	0	
10	70	Product TOONT	7	15,0000	0	
11	75	Product BWRLG	12	7,7500	0	



Paso 8 Utilizando utilizando Tablas Derivadas

- Escriba una instrucción SELECT contra una tabla derivada y recupere las columnas productid y productname. Filtre los resultados para incluir solo las filas en las que el valor de la columna pricetype sea igual a high. Use la instrucción SELECT del ejercicio 1, tarea 4, como la consulta interna que define la tabla derivada. No olvide usar un alias para la tabla derivada. (Puede usar el alias "p").

```
1 select
2 p.productid , p.productname
3 FROM
4 (
5     SELECT
6         productid , productname,supplierid , unitprice, discontinued,
7         CASE WHEN unitprice > 100. THEN N'high' ELSE N'normal' END as pricetype
8         FROM Production.Products
9         WHERE categoryid = 1
10 )
11 as p
12 where p.pricetype = N'high';
```

	productid	productname
1	38	Product QDOMO


Paso 10:

- Escriba una instrucción SELECT para recuperar la columna custid y dos columnas calculadas: totalsalesamount, que devuelve la cantidad total de ventas por cliente, y avgsalesamount, que devuelve la cantidad promedio de ventas de pedidos por cliente. Para calcular correctamente el monto promedio de ventas de pedidos por cliente, primero debe calcular el monto total de ventas por pedido. Puede hacerlo definiendo una tabla derivada basada en una consulta que une las tablas Sales.Orders y Sales.OrderDetails. Puede usar las columnas custid y orderid de la tabla Sales.Orders y las columnas qty y unitprice de la tabla Sales.OrderDetails



```
1 select
2     c.custid,
3     SUM(c.totalsalesamountperorder) as totalsalesamount,
4     AVG(c.totalsalesamountperorder) as avgsalesamount
5 from
6     (
7         select
8             o.custid , o.orderid , SUM (d.unitprice * d.qty) as totalsalesamountperorder
9         from Sales.Orders as o
10        inner join Sales.OrderDetails d ON d.orderid = d.orderid
11        group by o.custid , o.orderid
12    ) as c
13    group by c.custid;
```

	custid	totalsalesamount	avgsalesamount
1	1	8126751,5400	1354458,5900
2	2	5417834,3600	1354458,5900
3	3	9481210,1300	1354458,5900
4	4	17607961,6700	1354458,5900
5	5	24380254,6200	1354458,5900
6	6	9481210,1300	1354458,5900
7	7	14899044,4900	1354458,5900
8	8	4063375,7700	1354458,5900
9	9	23025796,0300	1354458,5900
10	10	18962420,2600	1354458,5900
11	11	13544585,9000	1354458,5900



Paso 11:

- Escriba una instrucción SELECT como la del ejercicio 2.1, pero use un CTE en lugar de una tabla derivada. Utilice el alias de columna en línea en la consulta CTE y nombre las Bebidas de producto CTE.



```
1 with ProductsBeverages as
2 (
3     SELECT
4         productid , productname,supplierid , unitprice, discontinued,
5         CASE WHEN unitprice > 100. THEN N'high' ELSE N'normal' END as pricetype
6     FROM Production.Products
7     WHERE categoryid = 1
8 )
9 select
10 productid , productname
11 from ProductsBeverages
12 where pricetype = N'high';
```

	productid	productname
1	38	Product QDOMO

Paso 12:

- Escriba una declaración SELECT contra Sales.OrderValues para recuperar el ID de cada cliente y el monto total de ventas para el año 2008. Defina un CTE llamado c2008 basado en esta consulta, utilizando el formulario de alias externo para nombrar las columnas CTE custid y salesamt2008. Únase a la tabla Sales.Customers y al c2008 CTE, devolviendo las columnas custid y contactname de la tabla Sales.Customers y la columna salesamt2008 del c2008 CTE.

```
1 with c2008 (custid , salesamt2008) as
2 (
3     select
4         custid , SUM (val)
5         from Sales.OrderValues
6         where YEAR(orderdate) = 2008
7         group by custid
8 )
9 select
10 c.custid , c.contactname , c2008.salesamt2008
11 from Sales.Customers as c
12 left outer join c2008 on c.custid = c2008.custid;
```



	custid	contactname	salesamt2008
1	1	Allen, Michael	2250,50
2	2	Hassall, Mark	514,40
3	3	Peoples, John	660,00
4	4	Arndt, Torsten	5604,75
5	5	Higginbotham, Tom	6754,16
6	6	Poland, Carole	2160,00
7	7	Bansal, Dushyant	730,00
8	8	Ilyina, Julia	224,00
9	9	Raghav, Amritansh	6680,61
10	10	Bassols, Pilar Colome	11338,56



Paso 13: Utilizando Funciones de Tabla en línea

- Escriba una instrucción SELECT en la vista Sales.OrderValues y recupere las columnas custid y totalsalesamount como un total de la columna val. Filtre los resultados para incluir pedidos solo para el año 2007.

```
1 select
2 custid , sum(val) as totalsalesamount
3 from sales.ordervalues
4 where year(orderdate) = 2007
5 group by custid
6 go
```

	custid	totalsalesamount
1	1	2022,50
2	2	799,75
3	3	5960,78
4	4	6406,90
5	5	13849,02
6	6	1079,80
7	7	7817,88
8	8	3026,85
9	9	11208,36





Paso 14:

- Defina un TVF en línea utilizando el siguiente encabezado de función y agregue su consulta anterior después de la cláusula RETURN: modifique la consulta reemplazando el valor del año constante 2007 en la cláusula WHERE con el parámetro @orderyear

```
1 create FUNCTION dbo.fnGetSalesByCustomer
2 (@orderyear AS INT) RETURNS TABLE
3 AS
4 RETURN
5 SELECT
6 custid, SUM(val) AS totalsalesamount
7 FROM Sales.OrderValues
8 WHERE YEAR(orderdate) = @orderyear
9 GROUP BY custid;
10 GO
```

Paso 15:

- Escriba una declaración SELECT que recupere los tres productos más vendidos en función del valor de venta total para el cliente con ID 1. Devuelva las columnas productid y productname de la tabla Production.Products. Use las columnas qty y unitprice de la tabla Sales.OrderDetails para calcular el valor de cada línea de pedido y devuelva la suma de todos los valores por producto, nombrando la cantidad total de la columna resultante. Filtre los resultados para incluir solo las filas donde el valor custid es igual a 1.

```
1 SELECT TOP(3)
2     d.productid,
3     MAX(p.productname) AS productname,
4     SUM(d.qty * d.unitprice) AS totalsalesamount
5 FROM Sales.Orders AS o
6 INNER JOIN Sales.OrderDetails AS d ON d.orderid = o.orderid
7 INNER JOIN Production.Products AS p ON p.productid = d.productid
8 WHERE custid = 1
9 GROUP BY d.productid
10 ORDER BY totalsalesamount DESC;
11
12 GO
```



	productid	productname	totalsalesamount
1	63	Product ICKNK	878,0000
2	59	Product UKXRI	825,0000
3	28	Product OFBNT	775,2000

Paso 16:

- Cree un TVF en línea basado en el siguiente encabezado de función, utilizando la instrucción SELECT anterior. Reemplace el valor custid constante 1 en la consulta con el parámetro de entrada de la función @custid:

```
1 CREATE FUNCTION dbo.fnGetTop3ProductsForCustomer
2 (@custid AS INT) RETURNS TABLE
3 AS
4 RETURN
5 SELECT TOP(3)
6     d.productid,
7     MAX(p.productname) AS productname,
8     SUM(d.qty * d.unitprice) AS totalsalesamount
9 FROM Sales.Orders AS o
10 INNER JOIN Sales.OrderDetails AS d ON d.orderid = o.orderid
11 INNER JOIN Production.Products AS p ON p.productid = d.productid
12 WHERE custid = @custid
13 GROUP BY d.productid
14 ORDER BY totalsalesamount DESC;
15
16 GO
```

Paso 17:

- Pruebe el TVF en línea creado escribiendo una instrucción SELECT contra él y use el valor 1 para el parámetro ID de cliente. Recupere las columnas productid, productname y totalsalesamount, y use el alias "p" para el TVF en línea

```
1 SELECT
2 p.productid,
3 p.productname,
4 p.totalsalesamount
5 from dbo.fnGetTop3ProductsForCustomer(5) as p
```



	productid	productname	totalsalesamount
1	38	Product QDOMO	3952,5000
2	43	Product ZZZHR	2760,0000
3	28	Product OFBNT	2280,0000

Paso 18:

- Escribir el código necesario para limpiar todos los cambios realizados.

```
1
2 DROP VIEW Production.ProductsBeverages;
3 GO
4 DROP FUNCTION dbo.fnGetSalesByCustomer;
5 GO
6 DROP FUNCTION dbo.fnGetTop3ProductsForCustomer;
```

IV. CUESTIONARIO

CONCLUSIONES

- ✓ Las consultas son uno de los análisis fundamentales dentro de un SIG. Básicamente, una consulta efectúa una pregunta acerca de la información contenida en una capa, y obtiene como resultado los elementos de la capa que dan respuesta a dicha pregunta. Las consultas son en general un elemento aplicado sobre capas vectoriales, y el resultado de la consulta se expresa mediante una selección de entidades dentro de aquellas que componen dicha capa.

WEBGRAFIA

- ✓ Expresiones de tablas comunes
Recuperado de: <https://teamgeekrd.wordpress.com/2017/10/23/expresiones-de-tabla-comunes-cte-en-sql-server/>
- ✓ CTE
Recuperado de: <http://bitsmi.com/2016/08/sql-expresiones-de-tablas-comunes-cte/>