

PROGRAMBI

TAREA FINAL CURSO SQL BASICO- INTERMEDIO

Alumno: Christian Farnast Contardo

Profesor: Emanuel Berrocal Zapata

Fecha: 21-12-2022

A1) Muestre las utilidades de cada producto por día (utilizando los cruces respectivos)

```
-- 1. mostrar las utilidades de cada producto por día

select
p.Cod_Producto,
familia ,
fecha ,
precio_venta_unit ,
cantidad ,
costo_unitario |
into #tabla_temporal_1
from productos p
full join ventas v on p.cod_producto = v.cod_producto

select*, Utilidades_dia = cantidad*(precio_venta_unit-costo_unitario)
from #tabla_temporal_1
```

	Cod_Producto	familia	fecha	precio_venta_unit	cantidad	costo_unitario	Utilidades_dia
1	AA11	AAAA	2019-01-01	2500	300	1250	375000
2	CC22	RRRR	2019-01-02	3000	200	1950	210000
3	AA12	RRRRTT	2019-01-03	4000	100	2000	200000
4	CC23	GGGGTT	2019-01-04	5000	300	3250	525000
5	AA13	MMM	2019-01-05	6000	500	3000	1500000
6	CC24	RRRR	2019-01-06	7000	700	4550	1715000
7	AA14	RRRRTT	2019-01-07	8000	900	4000	3600000
8	CC25	GGGGTT	2019-01-08	9000	1100	5850	3465000
9	AA15	MMM	2019-01-09	10000	1300	5000	6500000
10	CC26	RRRR	2019-01-10	11000	1500	7150	5775000
11	AA16	RRRRTT	2019-01-11	12000	1700	6000	10200000

Observación: Se hizo a partir de esta consulta, una tabla temporal que permitió el desarrollo de las preguntas 2, 3, 4 y 5.

A2) Muestre un resumen de la suma de utilidades por productos vendidos

```
-- 2. Mostrar un resumen de la suma de utilidades por producto vendido
select cod_producto, sum(cantidad*(precio_venta_unit-costo_unitario)) as utilidades_producto
from #tabla_temporal_1
group by cod_producto;
```

100 %

Results Messages

	cod_producto	utilidades_producto
1	AA11	625000
2	AA12	2388000
3	AA13	8334000
4	AA14	17400000
5	AA15	22560000
6	AA16	38160000
7	AA17	52486000
8	AA18	31719450
9	AA19	52650000
10	AA20	41843900
11	AA21	66716000

A3) Luego Cruce nuevamente la información para obtener un resumen de utilidad por “Familia” de los productos, en base a la suma de utilidad.

SQLQuery2.sql - LL...Administrador (54)* SQLQuery1.sql - LL...Administrador (75)*

```
-- 3. Cruzar información para obtener un resumen de utilidad por "Familia" de los productos en base a la utilidad
select familia, sum(cantidad*(precio_venta_unit-costo_unitario)) as utilidades_producto
from #tabla_temporal_1
group by familia
order by familia desc;
```

100 %

Results Messages

	familia	utilidades_producto
1	RRRRTT	355156000
2	RRRR	256380250
3	MMM	394531000
4	GGGGTT	263172350
5	AAAA	625000

A4) Luego de acuerdo con el último cuadro descubra en que meses fueron las máximas utilidades de las 3 mejores familias que vendieron. Y finalmente cuales fueron los productos que más vendieron (si hay dos o más iguales, nombrellos todos).

```
-- de acuerdo con la pregunta anterior, las familia que mas vendieron son: rrrrtt, rrrr y mmm
-- 4.1 ¿Qué mes las familias tuvieron su maxima utilidad?
select familia, month(fecha) as mes, sum(cantidad*(precio_venta_unit-costo_unitario)) as utilidades_producto
from #tabla_temporal_1
where familia in ('RRRRTT', 'RRRR', 'MMM')
group by familia, month(fecha)
order by sum(cantidad*(precio_venta_unit-costo_unitario)) desc;
```

100 %

Results Messages

	familia	mes	utilidades_producto
1	MMM	6	112456000
2	RRRRTT	5	95860000
3	MMM	4	78223000
4	RRRRTT	6	72234000
5	MMM	5	61068000
6	RRRR	6	60798850
7	RRRR	5	58878400
8	RRRRTT	3	56522000
9	MMM	3	50598000
10	RRRRTT	4	49224000
11	RRRR	4	42087150
12	MMM	2	37639000
13	RRRRTT	1	34130000
14	RRRR	3	33407500
15	MMM	1	30872000
16	RRRR	1	29331050
17	RRRRTT	7	23696000

```
--4.2 ¿Cuáles son los productos que mas vendieron estas 3 familias?
select familia,cod_producto, month(fecha) as mes, sum(cantidad*(precio_venta_unit-costo_unitario)) as utilidades_producto
from #tabla_temporal_1
where familia in ('RRRRTT','RRRR','MMM')
group by familia, cod_producto, month(fecha)
order by sum(cantidad*(precio_venta_unit-costo_unitario)) desc;
```

	familia	cod_producto	mes	utilidades_producto
1	MMM	CC37	6	29631000
2	MMM	AA25	6	26885000
3	RRRRTT	CC35	5	26082000
4	RRRRTT	CC35	6	25488000
5	RRRRTT	AA23	5	23248000
6	RRRRTT	AA23	6	22720000
7	MMM	CC33	4	22305000
8	MMM	CC33	5	21810000
9	MMM	CC33	6	21315000
10	MMM	AA17	1	20272000

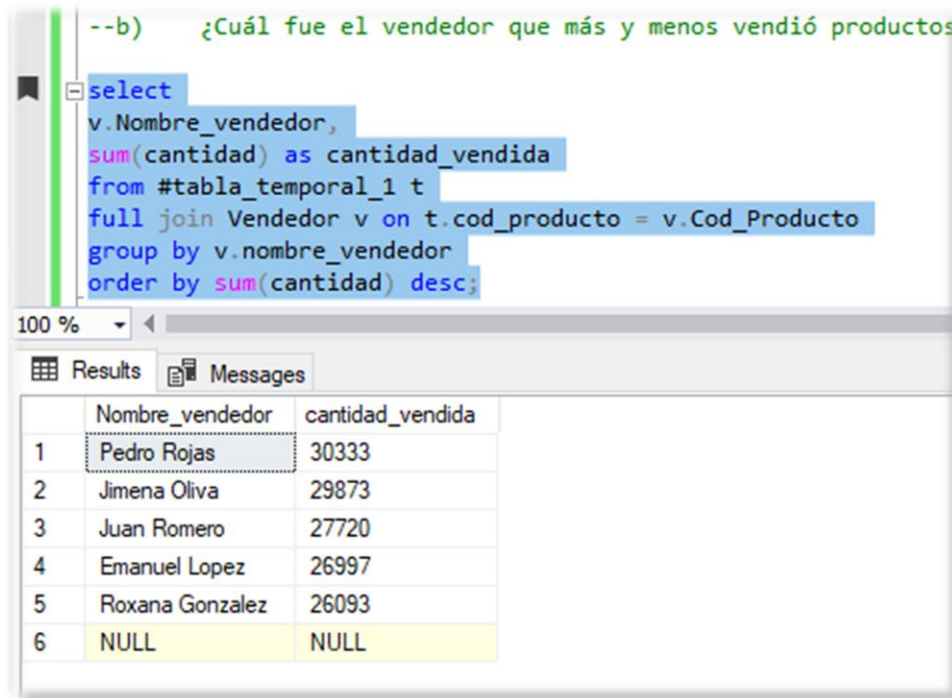
A5) Deberá hacer nuevos cruces para responder las siguientes preguntas:

a) ¿En qué país se vendió más productos y cuanto fue el monto?

```
-- 5. Deberá hacer nuevos cruces para responder las siguientes preguntas:
--a) ¿En qué país se vendió más productos y cuanto fue el monto?
select
--t.cod_producto,
p.pais_origen,
sum(cantidad) as cantidad_vendida,
sum(cantidad*(precio_venta_unit-costo_unitario)) as venta_total
from #tabla_temporal_1 t
full join pais p on t.cod_producto = p.cod_bueno
group by p.pais_origen--, t.cod_producto
order by sum(cantidad)desc;
```

	pais_origen	cantidad_vendida	venta_total
1	Chile	30586	328029000
2	Alemania	29693	326415000
3	China	29209	229316850
4	Japon	27570	212499000
5	Brasil	13467	77736750
6	EEUU	10491	95868000
7	NULL	NULL	NULL

b) ¿Cuál fue el vendedor que más y menos vendió productos?



The screenshot shows a SQL query in a database tool's editor. The query is as follows:

```
--b) ¿Cuál fue el vendedor que más y menos vendió productos

select
v.Nombre_vendedor,
sum(cantidad) as cantidad_vendida
from #tabla_temporal_1 t
full join Vendedor v on t.cod_producto = v.Cod_Producto
group by v.nombre_vendedor
order by sum(cantidad) desc;
```

Below the query editor, the 'Results' tab is active, displaying the following data:

	Nombre_vendedor	cantidad_vendida
1	Pedro Rojas	30333
2	Jimena Oliva	29873
3	Juan Romero	27720
4	Emanuel Lopez	26997
5	Roxana Gonzalez	26093
6	NULL	NULL

A6) Teniendo en cuenta el siguiente cálculo para el impuesto:

- $\text{Impuesto_producto} = \text{Utilidad} * (\text{Tasa_impuesto del País})$

Y esto da origen a la utilidad Final:

- $\text{Utilidad_Final} = \text{Utilidad} - \text{Impuesto_producto} = \text{Utilidad} * (1 - \text{impuesto_pais})$

Calcule la utilidad final por familia y por País (en suma) visto como pivot, utilizando SQL dinámico, viendo los países como las columnas pivoteadas y familia que quede como columna no pivoteada. Finalmente, esta query debe quedar como un proceso de almacenamiento.

Parte 1: creación de tablas temporales y stored procedure

```
-- Creacion de tabla temporal
select p.pais_origen, t.familia,
sum(cantidad) as cantidad_vendida,
sum(cantidad*(precio_venta_unit-costo_unitario)) as venta_total
into #tabla
from #tabla_temporal_1 t
full join pais p on t.cod_producto = p.cod_bueno
group by p.pais_origen, t.familia
order by sum(cantidad)desc;

-- Creacion de una nueva tabla temporal, la que cruzara la #tabla con impuestos
select
t.pais_origen,
t.familia,
sum(venta_total) - sum(venta_total*(1-tasa_impuesto)) as Utilidad_Impuesto
into #tabla2
from #tabla t
full join impuestos i on t.pais_origen = i.Pais
group by t.pais_origen, t.familia

-- Creacion del stored procedure
CREATE PROCEDURE tabla_pivot
AS
select
pais_origen, [AAAA], [GGGGTT], [MMM],[RRRR],[RRRRTT]
from #tabla2
pivot
(
    AVG(UTILIDAD_IMPUESTO)
    FOR familia in ([AAAA], [GGGGTT], [MMM],[RRRR],[RRRRTT]))
AS P
GO
```


Parte 2: resultados de las tablas temporales y stored procedure

```
-- Creacion de tabla temporal
select p.pais_origen, t.familia,
sum(cantidad) as cantidad_vendida,
sum(cantidad*(precio_venta_unit-costo_unitario)) as venta_total
into #tabla
from #tabla_temporal_1 t
full join pais p on t.cod_producto = p.cod_bueno
group by p.pais_origen, t.familia
order by sum(cantidad) desc;

select*from #tabla
```

100 %

Results Messages

	pais_origen	familia	cantidad_vendida	venta_total
1	Chile	AAAA	500	625000
2	Brasil	GGGGTT	8050	45959900
3	China	GGGGTT	13954	111033300
4	Japon	GGGGTT	13401	106179150
5	NULL	MMM	NULL	NULL
6	Alemania	MMM	14577	161817000
7	Chile	MMM	15460	172972000
8	EEUU	MMM	6726	59742000
9	Brasil	RRRR	5417	31776850
10	China	RRRR	15255	118283550
11	Japon	RRRR	14169	106319850
12	Alemania	RRRRTT	15116	164598000
13	Chile	RRRRTT	14626	154432000
14	EEUU	RRRRTT	3765	36126000


```
-- Creacion de una nueva tabla temporal, la que cruzara la #tabla con impuestos
select
t.pais_origen,
t.familia,
sum(venta_total) - sum(venta_total*(1-tasa_impuesto)) as Utilidad_Impuesto
into #tabla2
from #tabla t
full join impuestos i on t.pais_origen = i.Pais
group by t.pais_origen, t.familia

select*from #tabla2
```

100 %

Results Messages

	pais_origen	familia	Utilidad_Impuesto
1	Chile	AAAA	16875000
2	Brasil	GGGGTT	2297995000
3	China	GGGGTT	2775832500
4	Japon	GGGGTT	3185374500
5	NULL	MMM	NULL
6	Alemania	MMM	6472680000
7	Chile	MMM	4670244000
8	EEUU	MMM	2270196000
9	Brasil	RRRR	1588842500
10	China	RRRR	2957088750
11	Japon	RRRR	3189595500
12	Alemania	RRRRTT	6583920000
13	Chile	RRRRTT	4169664000
14	EEUU	RRRRTT	1372788000

```
-- Creacion del stored procedure
CREATE PROCEDURE tabla_pivot
AS
select
pais_origen, [AAAA], [GGGGTT], [MMM],[RRRR],[RRRRTT]
from #tabla2
pivot
(
    AVG(UTILIDAD_IMPUESTO)
    FOR familia in ([AAAA], [GGGGTT], [MMM],[RRRR],[RRRRTT]))
AS P
GO

EXEC tabla_pivot
```

00 %

Results Messages

	pais_origen	AAAA	GGGGTT	MMM	RRRR	RRRRTT
1	NULL	NULL	NULL	NULL	NULL	NULL
2	Alemania	NULL	NULL	6472680000	NULL	6583920000
3	Brasil	NULL	2297995000	NULL	1588842500	NULL
4	Chile	16875000	NULL	4670244000	NULL	4169664000
5	China	NULL	2775832500	NULL	2957088750	NULL
6	EEUU	NULL	NULL	2270196000	NULL	1372788000
7	Japon	NULL	3185374500	NULL	3189595500	NULL