ANÁLISIS DE DATOS (AD HOC) AVANZADO

EMPRESA COMERCIAL

HERRAMIENTA DE MANIPULACION Y TRANSFORMACIÓN DE DATOS

HERRAMIENTA DE VISUALIZACIÓN DE DATOS

AUTOR: DANIEL VILLAR RODRÍGUEZ

Este proyecto de análisis de datos pretende dar respuesta a requisitos que se plantean en un negocio comercial dedicado a la compra-venta de productos alimentarios y que inició su andadura comercial en Abril de 2020; los últimos datos a analizar son de principios de Junio 2022; para ello se aplicarán técnicas complejas de transformación y manipulación de datos en MS SQL Server, se crearán vistas para reutizar información y se usará el conector en PowerBI Desktop de MS SQL Server para visualizar hallazgos encontrados a través de instrucciones SQL.

La empresa de momento no se plantea adoptar soluciones de Business Inteligence pero va a ir adaptándose por medio de la visualización de tablas y gráficos en MS PowerBI para empezar a tomar contacto. Para ello, en PowerBI Desktop se crea una tabla de fechas que se relacionará con los hechos analizados cuando se realicen las consultas en SQL Server.

Existen cinco fases en el proceso de análisis de datos:

a) *Fase 1: Toma de requisitos de negocio:*

   En esta primera fase se tratará de concretar la información que se necesite obtener realizando las preguntas oportunas al negocio y resolviendo las que el propio negocio requiera.

b) *Fase 2: Script propuesto en SQL Server:*

   A continuación, realizaremos las transformaciones y consultas oportunas usando el lenguaje T-SQL para interactuar con la base de datos.

c) *Fase 3: Muestras del resultado o bien el resultado general obtenido tras la consulta:*

   Seguidamente, se realizará una captura de pantalla con el resultado obtenido de la consulta a la base de datos.

d) *Fase 4: Interpretación visual usando PowerBI (si procede):*

   En esta fase se usará el conector existente en PowerBI con SQL Server para enlazar con la base de datos y realizaremos el uso de la opción avanzada “Instrucción SQL” en PowerBI con el fin de importar los datos de la consulta.

e) *Fase 5: Conclusiones del análisis:*

   En está última fase se hará una breve conclusión de los hallazgos obtenidos.

Antes de profundizar en los diversos requerimientos de negocio, se va a consolidar la información con el fin de conocer la posibilidad de sesgos o datos atípicos; para ello, se va a realizar una exploración inicial de los datos de:

- *Precios de venta unitarios de los productos.*
- *Fletes cobrados a clientes.*
- *Cantidades vendidas de productos.*

a) EXPLORACION INICIAL DE DATOS

Se usan las medidas de tendencia central “media” y “mediana” para ver la consistencia de los datos y si hay datos atípicos comparando ambas medidas que representaremos en un gráfico de dispersión.

1. PRECIOS DE VENTA UNITARIOS:

1.b) SCRIPT SQL

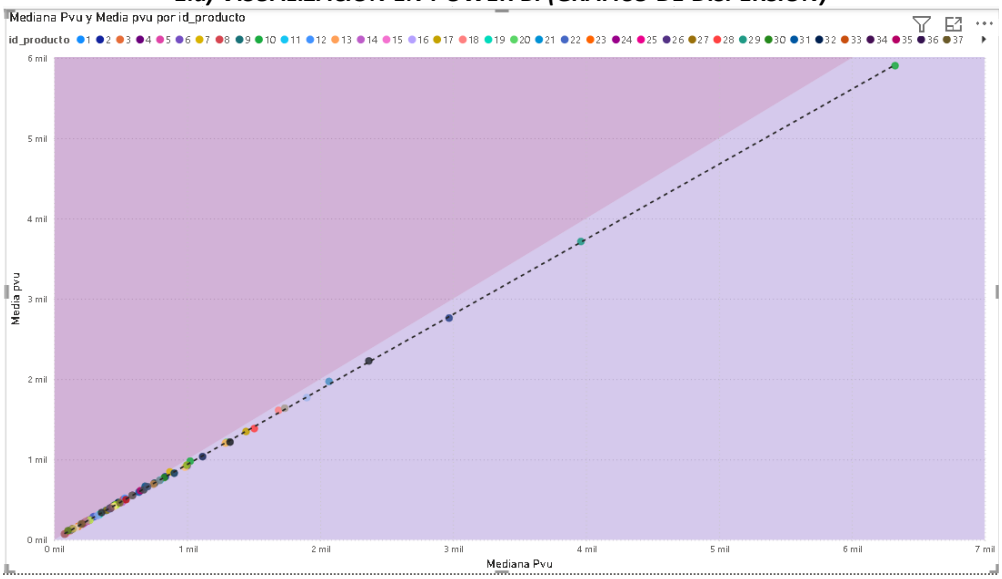
```
WITH T1 AS
 (
            SELECT
              A.id_producto,
              A.preciounitario,
              AVG (A.preciounitario) OVER (PARTITION BY A.id_producto) AS media,
              PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY PrecioUnitario) OVER (PARTITION BY ID_Producto) AS mediana
            FROM
              Ventas.DetalleOrden A
 )
SELECT
  *
FROM T1
```

1.c) MUESTRA DEL RESULTADO

Results | Messages

	id_producto	preciounitario	media	mediana
34	1	18,00	17,1473	18
35	1	18,00	17,1473	18
36	1	18,00	17,1473	18
37	1	18,00	17,1473	18
38	1	18,00	17,1473	18
39	2	15,20	17,8772	19
40	2	15,20	17,8772	19
41	2	15,20	17,8772	19
42	2	15,20	17,8772	19
43	2	15,20	17,8772	19
44	2	15,20	17,8772	19
45	2	15,20	17,8772	19

1.d) VISUALIZACION EN POWER BI (GRÁFICO DE DISPERSION)



1.e) CONCLUSION DEL ANÁLISIS

Se puede observar claramente que *no existen anomalias ni resultados atípicos* extraños en el comportamiento de los datos, es por lo que se pueden realizar los análisis oportunos a los precios de venta unitarios que se requieran sin tener datos sesgados.

2. FLETES COBRADOS A CLIENTES:

2.b) SCRIPT SQL

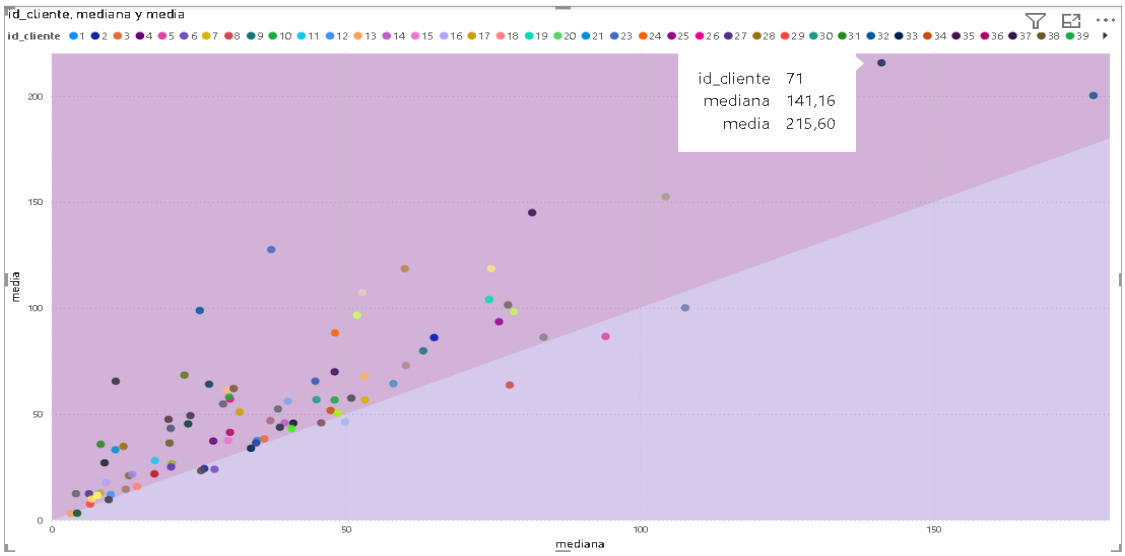
```
SELECT
  T1.id_cliente,
  T1.flete,
  AVG (T1.flete) OVER (PARTITION BY T1.id_cliente) AS media,
  PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY flete) OVER (PARTITION BY id_cliente) AS mediana
FROM
  Ventas.Ordenes T1
GROUP BY
  T1.id_cliente,
  T1.flete
ORDER BY
  id_cliente ASC,
  flete ASC
```

2.c) MUESTRA DEL RESULTADO

Results Messages

	id_cliente	flete	media	mediana
10	2	43,90	24,355	25,955
11	3	4,03	38,36	36,13
12	3	15,64	38,36	36,13
13	3	22,00	38,36	36,13
14	3	36,13	38,36	36,13
15	3	47,45	38,36	36,13
16	3	58,43	38,36	36,13
17	3	84,84	38,36	36,13
18	4	3,04	37,3525	27,485
19	4	4,52	37,3525	27,485
20	4	10,96	37,3525	27,485

2.d) VISUALIZACION EN POWER BI (GRÁFICO DE DISPERSION)



En el punto 2.d podemos observar un posible valor atípico señalado con el id_cliente 71 que procedemos a investigar a que se debe:

SCRIPT SQL

```
WITH Tabla1 AS
 (
            SELECT
              T1.id_orden,
              T1.id_cliente,
              T1.flete,
              AVG (T1.flete) OVER (PARTITION BY T1.id_cliente) AS media,
              PERCENTILE_CONT(0.5)  WITHIN GROUP (ORDER BY flete) OVER (PARTITION BY id_cliente) AS mediana,
              SUM (T2.cantidad * T2.preciounitario * (1-T2.descuento)) AS total_pdo,
              T1.id_empleado
            FROM
              Ventas.Ordenes T1
              INNER JOIN Ventas.DetalleOrden T2
              ON T1.id_orden = T2.id_orden
            WHERE
              T1.id_cliente = 71
            GROUP BY
              T1.id_orden,
              T1.id_cliente,
              T1.flete,
              T1.id_empleado
 )
SELECT
 *,
 CAST (flete / CAST (total_pdo AS decimal (8,2)) AS decimal (8,2)) AS pct_flete
FROM
 Tabla1
ORDER BY
 pct_flete ASC
```

MUESTRA DEL RESULTADO

Results | Messages

	id_orden	id_cliente	flete	media	mediana	total_pdo	id_empleado	pct_flete
21	11002	71	141,16	215,6032	141,16	1811.1000000	4	0.08
22	10627	71	107,46	215,6032	141,16	1185.7500000	8	0.09
23	11031	71	227,22	215,6032	141,16	2393.5000000	6	0.09
24	10612	71	544,08	215,6032	141,16	6375.0000000	1	0.09
25	10555	71	252,49	215,6032	141,16	2944.4000000	6	0.09
26	10941	71	400,81	215,6032	141,16	4011.7500000	7	0.10
27	10847	71	487,57	215,6032	141,16	4931.9200000	4	0.10
28	10748	71	232,55	215,6032	141,16	2196.0000000	3	0.11
29	10984	71	211,22	215,6032	141,16	1809.7500000	1	0.12
30	10815	71	14,62	215,6032	141,16	40.0000000	2	0.37
31	10983	71	657,54	215,6032	141,16	720.9000000	2	0.91

CONCLUSION DEL ANÁLISIS

Se concluye con un resultado que arroja un porcentaje de flete cobrado muy superior al pct_flete común (el cuál se sitúa en torno al 10 %) y se observa que es el **empleado 2** el que factura esos datos que consideramos atípicos; su responsable directo comentará con él para ver el motivo de esos altos fletes que se le cobran de más al cliente.

3. CANTIDADES VENDIDAS DE PRODUCTOS A CLIENTES:

3.b) SCRIPT SQL

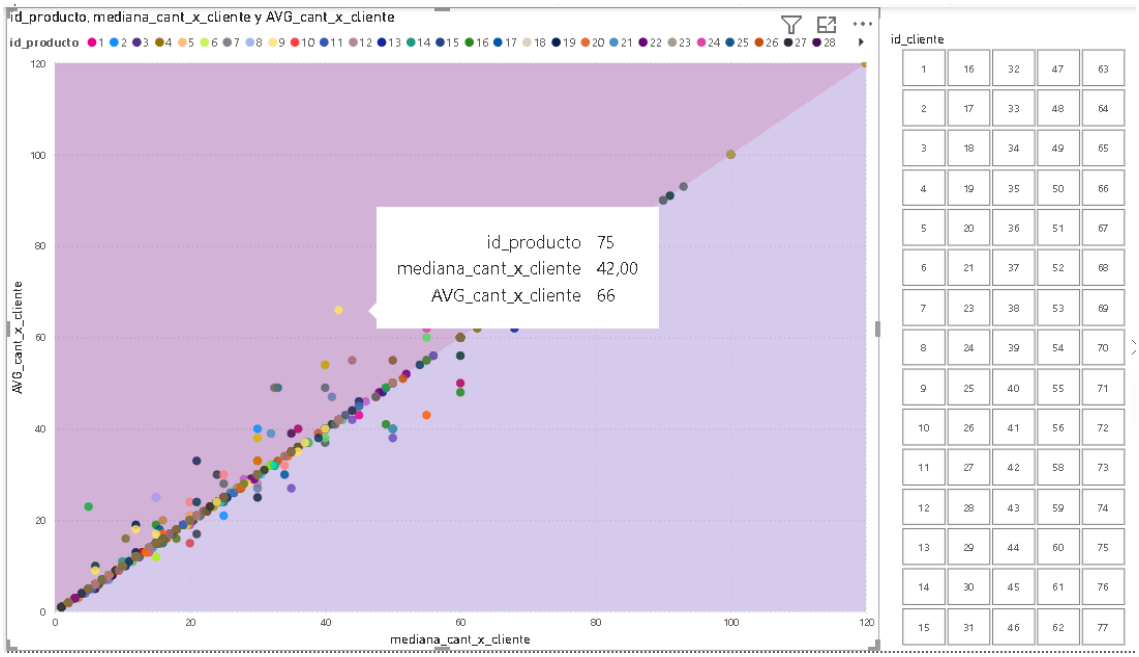
```
WITH tendCant AS
 (
            SELECT
              T1.id_orden,
              T1.id_producto,
              T2.id_cliente,
              T1.cantidad,
              PERCENTILE_CONT  (0.5) WITHIN GROUP (ORDER BY T1.cantidad)
                 OVER (PARTITION  BY T2.id_cliente, T1.id_producto) AS mediana_cant_x_cliente,
              AVG (T1.cantidad) OVER  (PARTITION  BY T2.id_cliente, T1.id_producto) AS AVG_cant_x_cliente
            FROM
              Ventas.DetalleOrden T1
              INNER JOIN Ventas.Ordenes T2
              ON T1.id_orden = T2.id_orden
 )
SELECT
 *,
 mediana_cant_x_cliente - AVG_cant_x_cliente AS dif_ABS_tend_central,
 (mediana_cant_x_cliente - AVG_cant_x_cliente) / AVG_cant_x_cliente AS dif_REL_tend_central
FROM
 tendCant
```

3.c) MUESTRA DEL RESULTADO

Results | Messages

	id_orden	id_producto	id_cliente	cantidad	mediana_cant_x_cliente	AVG_cant_x_cliente	dif_ABS_tend_central	dif_REL_tend_central
43	10768	22	4	4	4	4	0	0
44	10355	24	4	25	25	25	0	0
45	11016	31	4	15	50	38	12	0,315789473684211
46	10768	31	4	50	50	38	12	0,315789473684211
47	10953	31	4	50	50	38	12	0,315789473684211
48	10864	35	4	4	4	4	0	0
49	11016	36	4	16	16	16	0	0
50	10793	41	4	14	14	14	0	0
51	10743	46	4	28	28	28	0	0
52	10558	47	4	25	25	25	0	0
53	10453	48	4	15	15	15	0	0
54	10383	50	4	15	19,5	19	0,5	0,026315789473684

3.d) VISUALIZACION EN POWER BI (GRÁFICO DE DISPERSION)



En el punto 3.d podemos observar un posible valor atípico señalado con el id_producto 75 a nivel general por todos los clientes, el cuál procedemos a investigar a que se debe:

Filtramos el script anterior para obtener solo los datos del id_producto 75:

SCRIPT

```
WITH tendCant AS
 (
            SELECT
              T1.id_orden,
              T1.id_producto,
              T2.id_cliente,
              T1.cantidad,
              PERCENTILE_CONT  (0.5) WITHIN GROUP (ORDER BY T1.cantidad)
                OVER (PARTITION BY T2.id_cliente, T1.id_producto) AS mediana_cant_x_cliente,
              AVG (T1.cantidad) OVER (PARTITION BY T2.id_cliente, T1.id_producto) AS AVG_cant_x_cliente
            FROM
              Ventas.DetalleOrden T1
              INNER JOIN Ventas.Ordenes T2
              ON T1.id_orden = T2.id_orden
 )
```

```
SELECT
 *,
 mediana_cant_x_cliente - AVG_cant_x_cliente AS dif_ABS_tend_central,
 (mediana_cant_x_cliente - AVG_cant_x_cliente) / AVG_cant_x_cliente AS dif_REL_tend_central
FROM tendCant
WHERE
 id_producto = 75
ORDER BY
 dif_REL_tend_central ASC
```

En la muestra del resultado podemos observar un patron que se repite, y es que, si un cliente compra de nuevo el mismo producto, este oscila considerablemente, en términos de cantidades:

MUESTRA DEL RESULTADO

Results Messages

	id_orden	id_producto	id_cliente	cantidad	mediana_cant_x_cliente	AVG_cant_x_cliente	dif_ABS_tend_central	dif_REL_tend_central
1	10510	75	71	36	42	66	-24	-0,363636363636364
2	10722	75	71	42	42	66	-24	-0,363636363636364
3	10894	75	71	120	42	66	-24	-0,363636363636364
4	10460	75	24	4	12	18	-6	-0,333333333333333
5	10955	75	24	12	12	18	-6	-0,333333333333333
6	10980	75	24	40	12	18	-6	-0,333333333333333
7	11077	75	65	4	6	9	-3	-0,333333333333333
8	10294	75	65	6	6	9	-3	-0,333333333333333
9	10761	75	65	18	6	9	-3	-0,333333333333333
10	10924	75	5	6	15	17	-2	-0,117647058823529
11	10572	75	5	15	15	17	-2	-0,117647058823529
12	10280	75	5	30	15	17	-2	-0,117647058823529
13	10436	75	7	24	24	24	0	0
14	10326	75	8	50	50	50	0	0
15	10932	75	9	20	20	20	0	0
16	10975	75	10	10	10	10	0	0
17	10819	75	12	20	20	20	0	0
18	10363	75	17	12	12	12	0	0
19	10670	75	25	25	37	37	0	0
20	10929	75	25	49	37	37	0	0
21	10959	75	31	20	20	20	0	0
22	10785	75	33	10	10	10	0	0

Se realiza una consulta para saber qué tipo de producto es el id_producto 75:

```
SELECT * FROM Produccion.Productos A
INNER JOIN Produccion.Categorias B
ON A.id_categoria = B.id_categoria
WHERE id_producto = 75
```

MUESTRA DEL RESULTADO

Results Messages

	id_producto	nombreproducto	id_proveedor	id_categoria	preciounitario	descontinuado	id_categoria	nombrecategoria	descripcion
1	75	Producto BWRLG	12	1	7,75	0	1	Bebidas	Bebidas suaves, cafés, t

CONCLUSION DEL ANÁLISIS

Se trata de **CAFÉ**, que por distintas razones los clientes no realizan pedidos en cantidades homogéneas constantes, se concluye con que los comerciales de la empresa hablarán con los clientes con más confianza para intentar conocer y tratar el motivo de estos cambios en los pedidos.

RESUMEN REQUERIMIENTOS DE NEGOCIO

---- ANALISIS IMPORTE VENTAS GENERAL----

a) Ventas por productos y por clientes y representación anual con porcentajes ventas.
b) KPIS desempeño de ventas por producto y cliente.

---- ANALISIS PRECIOS DE VENTA UNITARIOS ----

a) Márgenes de ventas de los productos:

1. Márgenes históricos s/compra obtenidos agrupado por producto.
2. TOP 3 de productos con más margen s/compra de cada cliente.
3. Del punto 2, son los productos con más margen s/compra los que más se venden?
4. KPIS márgenes por producto.
5. Categorizar los márgenes de los productos y realizar una comparativa entre márgenes por categorías y márgenes acumulados.

b) Descuentos en productos vendidos:

1. ¿Existe alguna correlación entre que se apliquen descuentos a clientes en los productos con que estos compren más o menos cantidad de productos?

---- ANALISIS FLETES ----

a) Totales y acumulados de fletes cobrados.
b) AVG acumulados vs AVG móviles fletes general para analizar las tendencias de los datos.
c) AVG flete unitario por producto.

---- ANALISIS CANTIDADES VENDIDAS ----

a) Totales mensuales, promedios de cantidades vendidas acumuladas y promedio móvil.
b) Análisis general de cada producto por cliente:
- Producto más vendido por trimestre.
- Cantidades por cliente y producto anual.
- TOP 5 productos con mayor crecimiento en ventas en términos de cantidad comparado con el mismo período del año pasado
- Tendencia entre periodos de las cantidades vendidas para nuestro producto TOP 1 durante el último año
- Cantidades vendidas por país y semestre.
- Cantidad promedio de productos comprados por transacción para nuestros clientes recurrentes.
- Tendencias significativas de cantidades vendidas de productos por categorías
c) Análisis de rotación por producto y categoría.
d) Análisis de frecuencia de rotación y de demanda.

---- ANALISIS TIEMPOS ----

a) Que tiempo transcurre entre la realización de los pedidos y hasta que se carga la mercancía en transporte?
b) Análisis de frecuencia de compras de los clientes.

ANALISIS REQUERIMIENTOS DE NEGOCIO

---- ANALISIS IMPORTE VENTAS GENERAL----

a) Ventas anuales por productos y representación en tabla dinámica:

SCRIPT

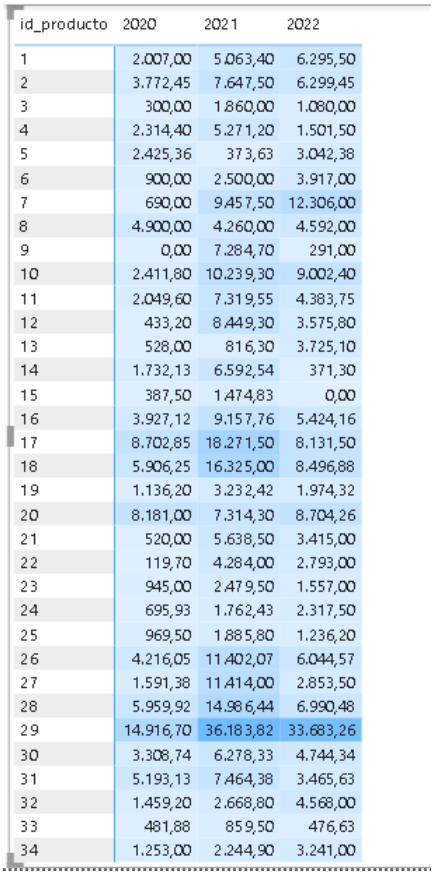
```
WITH tabla1 AS
 (
            SELECT
              YEAR (T3.fecha_orden) AS año,
              T1.id_producto,
              SUM (T2.preciounitario * T1.cantidad  * (1-T1.descuento)) AS importe_vta
            FROM
              Ventas.DetalleOrden T1
              INNER JOIN Produccion.Productos T2
              ON T1.id_producto = T2.id_producto
              INNER JOIN Ventas.Ordenes T3
              ON T1.id_orden = T3.id_orden
            GROUP BY
              YEAR (T3.fecha_orden),
              T1.id_producto
 ), años AS
 (
            SELECT
              id_producto,
              CAST (SUM  (CASE WHEN año = 2020 THEN importe_vta ELSE 0 END) AS decimal (8,2)) AS '2020',
              CAST (SUM  (CASE WHEN año = 2021 THEN importe_vta ELSE 0 END) AS decimal (8,2)) AS '2021',
              CAST (SUM  (CASE WHEN año = 2022 THEN importe_vta ELSE 0 END) AS decimal (8,2)) AS '2022'
            FROM
              tabla1
            GROUP BY
              id_producto
 )
SELECT
 *
FROM
 años
```

MUESTRA DEL RESULTADO

Results | Messages

	id_producto	2020	2021	2022
1	1	2007.00	5063.40	6295.50
2	2	3772.45	7647.50	6299.45
3	3	300.00	1860.00	1080.00
4	4	2314.40	5271.20	1501.50
5	5	2425.36	373.63	3042.38
6	6	900.00	2500.00	3917.00
7	7	690.00	9457.50	12306.00
8	8	4900.00	4260.00	4592.00
9	9	0.00	7284.70	291.00
10	10	2411.80	10239.30	9002.40
11	11	2049.60	7319.55	4383.75

VISUALIZACION EN POWER BI (MAPA DE CALOR)



id_producto	2020	2021	2022
1	2.007,00	5.063,40	6.295,50
2	3.772,45	7.647,50	6.299,45
3	300,00	1.860,00	1.080,00
4	2.314,40	5.271,20	1.501,50
5	2.425,36	373,63	3.042,38
6	900,00	2.500,00	3.917,00
7	690,00	9.457,50	12.306,00
8	4.900,00	4.260,00	4.592,00
9	0,00	7.284,70	291,00
10	2.411,80	10.239,30	9.002,40
11	2.049,60	7.319,55	4.383,75
12	433,20	8.449,30	3.575,80
13	528,00	816,30	3.725,10
14	1.732,13	6.592,54	371,30
15	387,50	1.474,83	0,00
16	3.927,12	9.157,76	5.424,16
17	8.702,85	18.271,50	8.131,50
18	5.906,25	16.325,00	8.496,88
19	1.136,20	3.232,42	1.974,32
20	8.181,00	7.314,30	8.704,26
21	520,00	5.638,50	3.415,00
22	119,70	4.284,00	2.793,00
23	945,00	2.479,50	1.557,00
24	695,93	1.762,43	2.317,50
25	969,50	1.885,80	1.236,20
26	4.216,05	11.402,07	6.044,57
27	1.591,38	11.414,00	2.853,50
28	5.959,92	14.986,44	6.990,48
29	14.916,70	36.183,82	33.683,26
30	3.308,74	6.278,33	4.744,34
31	5.193,13	7.464,38	3.465,63
32	1.459,20	2.668,80	4.568,00
33	481,88	859,50	476,63
34	1.253,00	2.244,90	3.241,00

CONCLUSION DEL ANÁLISIS

En esta muestra del mapa de calor proporcionado por PowerBI podemos observar donde están distribuidos los importes más altos que son los que tienen una intensidad de color azul mayor; en este ejemplo el producto 29 para las ventas tanto de 2021 como 2022 presentan las mayores ventas con respecto al resto de productos en esos años.

b) KPIS desempeño ventas por producto y cliente:

SCRIPT

```
WITH actual AS  -- DATOS INICIALES
 (
            SELECT
             YEAR (T2.fecha_orden) AS año,
             T2.id_cliente,
             CAST (SUM (T1.cantidad * T1.preciounitario * (1-T1.descuento)) AS decimal (8,2)) AS vta_mes_actual
            FROM
             Ventas.DetalleOrden T1
             INNER JOIN Ventas.Ordenes T2
             ON T1.id_orden = T2.id_orden
            GROUP BY
             YEAR (T2.fecha_orden),
             T2.id_cliente
 ),
 anual AS – CALCULO DE LA VENTA ACTUAL DE CADA CLIENTE
 (
            SELECT
             *,
             SUM (vta_mes_actual) OVER (PARTITION BY id_cliente, año) AS vta_actual
            FROM
             actual
 ),
 PY AS – CALCULO DE LA VENTA DEL AÑO ANTERIOR DE CADA CLIENTE
 (
            SELECT
             año,
             id_cliente,
             vta_actual,
             LAG (vta_actual, 1) OVER (PARTITION BY id_cliente ORDER BY año) AS vta_PY
            FROM
             anual
 ),
 KPI AS – CALCULO DE LOS KPI's
 (
            SELECT
             *,
             vta_actual - vta_PY AS KPI_abs,
             CONCAT (CAST ((vta_actual - vta_PY) / vta_PY AS decimal (8,2)) * 100, ' %') AS KPI_rel
            FROM
             PY
 ) – CONSULTA FINAL Y RESULTADO
SELECT
 *
FROM
 KPI
WHERE
 año IN (2021,2022)
 AND vta_PY IS NOT NULL
```

NOTA: Se ha filtrado el resultado por aquellos clientes que tuvieran ventas entre los años 2020 a 2022, ya que de otra forma nos hubieran salido datos nulos y no tendría sentido en términos comparativos este análisis.

MUESTRA DEL RESULTADO

Results | Messages

	año	id_cliente	vta_actual	vta_PY	KPI_abs	KPI_rel
19	2021	11	3179.50	479.40	2700.10	563.00 %
20	2022	11	2431.00	3179.50	-748.50	-24.00 %
21	2022	12	1576.80	238.00	1338.80	563.00 %
22	2021	14	6516.40	1674.22	4842.18	289.00 %
23	2022	14	4158.26	6516.40	-2358.14	-36.00 %
24	2021	15	1128.00	2169.00	-1041.00	-48.00 %
25	2022	15	513.75	1128.00	-614.25	-54.00 %
26	2022	16	931.50	787.60	143.90	18.00 %
27	2021	17	420.00	533.60	-113.60	-21.00 %
28	2022	17	2809.61	420.00	2389.61	569.00 %
29	2021	18	487.00	268.80	218.20	81.00 %

VISUALIZACION EN POWER BI (MAPA DE CALOR)

id_cliente	vta_actual	vta_PY	KPI_rel
44	13.076,13	3.105,38	321,00 %
46	5.175,20	5.394,08	-4,00 %
48	1.837,20	712,00	158,00 %
49	4.695,88	899,84	422,00 %
51	23.332,31	5.539,88	321,00 %
52	3.596,40	1.200,80	200,00 %
55	5.475,38	4.675,80	17,00 %
56	8.254,27	1.504,65	449,00 %
58	2.065,40	680,80	203,00 %
59	9.305,58	10.033,28	-7,00 %
60	1.409,20	1.001,84	41,00 %
61	3.502,41	1.808,80	94,00 %
62	10.132,77	9.210,90	10,00 %
63	61.109,91	11.950,08	411,00 %
65	19.383,75	10.475,78	85,00 %
66	3.000,84	80,10	3.646,00 %
67	4.283,78	1.168,50	267,00 %
68	11.864,42	2.490,50	376,00 %
70	700,00	1.058,40	-34,00 %
71	57.713,58	10.338,27	458,00 %
72	9.021,25	5.564,08	62,00 %
73	16.232,41	352,60	4.504,00 %
75	2.475,00	7.849,63	-68,00 %
76	6.137,48	6.306,70	-3,00 %
77	2.955,40	336,00	780,00 %
79	2.004,34	1.863,40	8,00 %

CONCLUSION DEL ANÁLISIS

En esta ocasión, se ha añadido un segmentador por años para obtener el detalle de los datos; en éste ejemplo, se observan con mayor intensidad de color azul los KPIS mayores del año 2021.

---- ANALISIS PRECIOS DE VENTA UNITARIOS ----

a) Márgenes de ventas de los productos:

1. Márgenes históricos s/compra obtenidos agrupado por producto:

SCRIPT

```
WITH tabla1 AS – DATOS INICIALES
 (
            SELECT
              T1.id_orden,
              T3.id_cliente,
              T1.id_producto,
              T1.cantidad,
              T1.preciounitario AS Pcu,
              T2.preciounitario AS Pvu,
              T1.descuento,
              CAST (T2.preciounitario * (1-T1.descuento) AS decimal (8,2)) AS Pvu_descuento
            FROM
              Ventas.DetalleOrden T1
              INNER JOIN Produccion.Productos T2
              ON T1.id_producto = T2.id_producto
              INNER JOIN Ventas.Ordenes T3
              ON T1.id_orden = T3.id_orden
 ),
 margen AS – CALCULO DE LOS MARGENES DEL PRODUCTO POR UNIDAD
 (
            SELECT
              *,
              CAST (Pvu_descuento-Pcu AS decimal (8,2)) * cantidad AS margen_ABS,
              CAST ((Pvu_descuento-Pcu) / Pcu AS decimal (8,2)) * 100 AS margen_REL
            FROM
              tabla1
 ),
 mgprod AS – CALCULO DEL TOTAL DE MARGEN OBTENIDO POR CADA PRODUCTO VENDIDO
 (
            SELECT
              id_producto,
              SUM (margen_ABS) AS mg_x_prod
            FROM
              margen
            GROUP BY
              id_producto
 ),
 acum AS – CALCULO DE LOS MARGENES TOTALES Y ACUMULADOS
 (
            SELECT
              *,
              SUM (mg_x_prod) OVER () AS mg_total,
              SUM (mg_x_prod) OVER (ORDER BY mg_x_prod DESC) AS mg_acum
            FROM
              mgprod
 ),
  grupo AS – CLASIFICACION DE PRODUCTO POR TIPO DE MARGEN
 (
            SELECT
              *,
              (CASE WHEN mg_x_prod < 0 THEN 'Bajo_costo' ELSE 'Con_margen' END) AS grupo
            FROM
              acum
 ),
  filas AS  -- ORDENACION DE LOS GRUPOS POR NUMERO DE FILAS
 (
SELECT
 *,
 ROW_NUMBER () OVER (ORDER BY id_producto) AS nro_filas,
 ROW_NUMBER () OVER (PARTITION BY grupo ORDER BY id_producto) AS nro_filas_x_grupo
```

```
FROM
 grupo
 ),
 clas_filas AS  -- OBTENCION DEL ULTIMO VALOR PARA EL CALCULO DE PORCENTAJE
 (
            SELECT
             *,
             LAST_VALUE (nro_filas) OVER (ORDER BY nro_filas
                                          ROWS BETWEEN CURRENT ROW AND UNBOUNDED FOLLOWING) AS total_filas,
             LAST_VALUE (nro_filas_x_grupo) OVER (PARTITION BY grupo ORDER BY nro_filas_x_grupo
                                          ROWS BETWEEN CURRENT ROW AND UNBOUNDED FOLLOWING) num_filas_x_grupo
            FROM
             filas
 ) – CONSULTA FINAL Y RESULTADO
SELECT
 id_producto,
 mg_x_prod,
 mg_total,
 grupo,
 CAST (num_filas_x_grupo / CAST (total_filas AS decimal (8,2)) AS decimal (8,2)) AS pct_grupo
FROM
 clas_filas
ORDER BY
 mg_x_prod DESC
```

MUESTRA DEL RESULTADO

Results | Messages

	id_producto	mg_x_prod	mg_total	grupo	pct_grupo
34	25	39.90	-735.69	Con_margen	0.48
35	43	29.90	-735.69	Con_margen	0.48
36	48	12.45	-735.69	Con_margen	0.48
37	40	6.74	-735.69	Con_margen	0.48
38	24	-5.36	-735.69	Bajo_costo	0.52
39	8	-8.00	-735.69	Bajo_costo	0.52
40	7	-10.50	-735.69	Bajo_costo	0.52
41	73	-25.80	-735.69	Bajo_costo	0.52
42	6	-28.00	-735.69	Bajo_costo	0.52

CONCLUSION DEL ANÁLISIS

Podemos concluir que la empresa vende un 52 % de los productos **BAJO COSTE**; después de hablar con Gerencia y comentarle el asunto nos comunica que efectivamente es la línea estratégica de negocio a seguir, ya que el beneficio se obtiene del volumen de compras con el proveedor a través de *Rappeles sobre compras.*

2. TOP 3 de productos con más margen s/compra de cada cliente:

a) Creamos una vista del script anterior para combinarla con los datos de los pedidos:

SCRIPT

```
GO
CREATE OR ALTER VIEW dbo.MargenesProductos
AS
WITH tabla1 AS
 (
            SELECT
              T1.id_orden,
              T3.id_cliente,
              T1.id_producto,
              T1.cantidad,
              T1.preciounitario AS Pcu,
              T2.preciounitario AS Pvu,
              T1.descuento,
              CAST (T2.preciounitario * (1-T1.descuento) AS decimal (8,2)) AS Pvu_descuento
            FROM
              Ventas.DetalleOrden T1
              INNER JOIN Produccion.Productos T2
              ON T1.id_producto = T2.id_producto
              INNER JOIN Ventas.Ordenes T3
              ON T1.id_orden = T3.id_orden
 ),
 margen AS
 (
            SELECT
              *,
              CAST (Pvu_descuento-Pcu AS decimal (8,2)) * cantidad AS margen_ABS,
              CAST ((Pvu_descuento-Pcu) / Pcu AS decimal (8,2)) * 100 AS margen_REL
            FROM
              tabla1
 ),
 mgprod AS
 (
            SELECT
              id_producto,
              SUM (margen_ABS) AS mg_x_prod
            FROM
              margen
            GROUP BY
              id_producto
 ),
 acum AS
 (
            SELECT
              *,
              SUM (mg_x_prod) OVER () AS mg_total,
              SUM (mg_x_prod) OVER (ORDER BY mg_x_prod DESC) AS mg_acum
            FROM
              mgprod
 ),
  grupo AS
 (
            SELECT
              *,
              (CASE WHEN mg_x_prod < 0 THEN 'Bajo_costo' ELSE 'Con_margen' END) AS grupo
            FROM
              acum
 ),
   filas AS
 (
SELECT
 *,
 ROW_NUMBER () OVER (ORDER BY id_producto) AS nro_filas,
 ROW_NUMBER () OVER (PARTITION BY grupo ORDER BY id_producto) AS nro_filas_x_grupo
```

```
FROM
 grupo
 ),
  clas_filas AS
 (
              SELECT
               *,
               LAST_VALUE (nro_filas) OVER (ORDER BY nro_filas
                                            ROWS BETWEEN  CURRENT  ROW AND UNBOUNDED  FOLLOWING) AS total_filas,
               LAST_VALUE (nro_filas_x_grupo) OVER (PARTITION  BY grupo ORDER BY nro_filas_x_grupo
                                                    ROWS BETWEEN  CURRENT  ROW AND UNBOUNDED  FOLLOWING) num_filas_x_grupo
              FROM
               filas
 )
SELECT
 id_producto,
 mg_x_prod,
 mg_total,
 grupo,
 CAST (num_filas_x_grupo / CAST (total_filas  AS decimal (8,2)) AS decimal (8,2)) AS pct_grupo
FROM
 clas_filas
GO
```

b) Combinamos la vista creada con las otras tablas para obtener el TOP 3:

SCRIPT

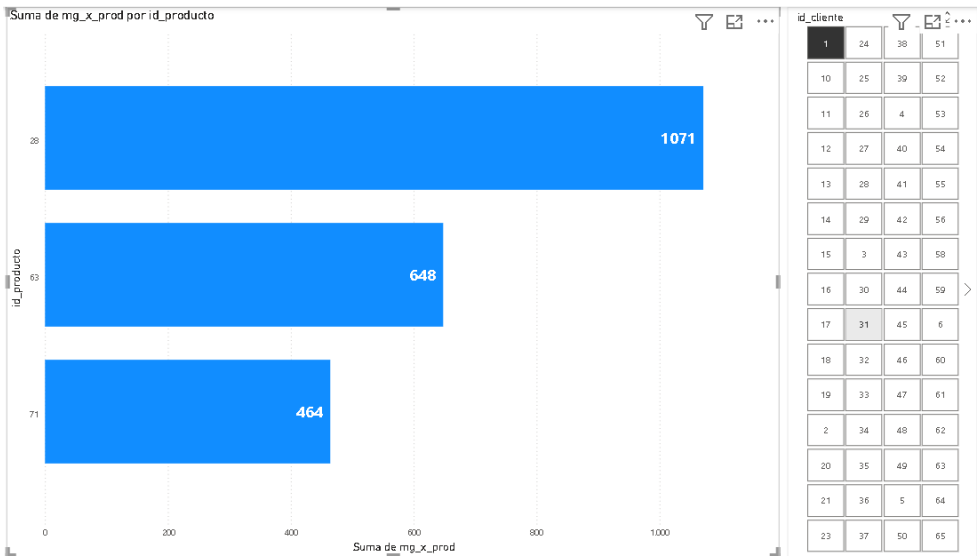
```
WITH tabla1 AS
 (
              SELECT
               id_cliente,
               a.id_producto,
               mg_x_prod
              FROM
               dbo.MargenesProductos A
               INNER JOIN Ventas.DetalleOrden  B
               ON A.id_producto = B.id_producto
               INNER JOIN Ventas.Ordenes C
               ON C.id_orden = B.id_orden
 ),
 ranking AS
 (
              SELECT
               *,
               DENSE_RANK () OVER (PARTITION  BY id_cliente ORDER BY mg_x_prod DESC) AS Ranking
              FROM
               tabla1
 )
SELECT
 id_cliente,
 id_producto,
 mg_x_prod,
 Ranking
FROM
 ranking
WHERE
 Ranking BETWEEN  1 AND 3
GROUP BY
 id_cliente,
 id_producto,
 mg_x_prod,
 Ranking
ORDER BY
 id_cliente ASC,
 Ranking ASC,
 id_producto ASC
```

MUESTRA DEL RESULTADO

Results | Messages

	id_cliente	id_producto	mg_x_prod	Ranking
1	1	28	1071.24	1
2	1	63	647.89	2
3	1	71	464.20	3
4	2	72	1347.78	1
5	2	19	183.44	2
6	2	14	65.83	3
7	3	53	1113.60	1
8	3	57	494.52	2
9	3	59	387.75	3
10	4	53	1113.60	1

VISUALIZACION EN POWER BI (GRAFICO DE BARRAS / RANKING)



CONCLUSION DEL ANÁLISIS

En este gráfico se añade un segmentador por cliente, el cuál seleccionando un cliente se puede visualizar el TOP 3 de productos con mayor margen.

3. Del punto 2, ¿son los productos con más margen s/compra los que más se venden?:

Esta respuesta se concreta en varias fases de consulta:

a) Para obtener el resultado primero guardamos la consulta anterior como una vista, QUITANDO el filtro y la columna de ranking para obtener todos los productos SIN AGRUPAR:

SCRIPT

```
GO
CREATE OR ALTER VIEW rankMgProdyCliente
AS
WITH tabla1 AS
 (
            SELECT
             id_cliente,
             a.id_producto,
             mg_x_prod
            FROM
             dbo.MargenesProductos A
             INNER JOIN Ventas.DetalleOrden B
             ON A.id_producto = B.id_producto
             INNER JOIN Ventas.Ordenes C
             ON C.id_orden = B.id_orden
 ),
  tabla2 AS
 (
            SELECT
             *
            FROM
             tabla1
 )
SELECT
 id_cliente,
 id_producto,
 mg_x_prod
FROM
 tabla2
GO
```

b) Luego combinamos otras tablas de cantidades vendidas de producto y de cliente y generamos una nueva vista SIN AGRUPAR:

SCRIPT

```
GO
CREATE OR ALTER VIEW rankCantProdyCliente
AS
WITH cantidades AS
 (
            SELECT
             T2.id_cliente,
             T1.id_producto,
             SUM (T1.cantidad) OVER (PARTITION BY T2.id_cliente, T1.id_producto) AS cantidades
            FROM
             Ventas.DetalleOrden T1
             INNER JOIN Ventas.Ordenes T2
             ON T1.id_orden = T2.id_orden
 )
SELECT
 *
FROM
 cantidades
GO
```

c) Combinamos las 2 vistas creadas para obtener el RANKING de cada columna a analizar (margen y cantidades) Y AGRUPAMOS para obtener el resultado:

SCRIPT

```
WITH ranking AS
 (
            SELECT
             T1.id_cliente,
             T1.id_producto,
             T1.cantidades,
             DENSE_RANK () OVER (PARTITION BY T1.id_cliente ORDER BY cantidades DESC) AS rank_más_vendidos,
             T2.mg_x_prod,
             DENSE_RANK () OVER (PARTITION BY T1.id_cliente ORDER BY mg_x_prod DESC) AS rank_mayor_margen
            FROM
             rankCantProdyCliente T1
             INNER JOIN rankMgProdyCliente T2
             ON T1.id_cliente = T2.id_cliente
             AND T1.id_producto = T2.id_producto
            GROUP BY
             T1.id_cliente,
             T1.id_producto,
             T1.cantidades,
             T2.mg_x_prod
 ),
  rankmargen AS
 (
            SELECT
             *
            FROM
             ranking
            WHERE
             rank_mayor_margen BETWEEN 1 AND 3
 ),
  coincidencia AS
 (
            SELECT
             *,
             (CASE WHEN rank_mayor_margen IN (1,2,3) AND rank_más_vendidos IN (1,2,3) THEN 'SI' ELSE 'NO' END) AS
             coincidencia
            FROM
             rankmargen
 ),
  filas AS
 (
            SELECT
             *,
             ROW_NUMBER () OVER (ORDER BY coincidencia) AS Num_filas_totales,
             ROW_NUMBER () OVER (PARTITION BY coincidencia ORDER BY coincidencia) AS Nro_filas_particion
            FROM
             coincidencia
 ),
  máximos AS
 (
            SELECT
             *,
             LAST_VALUE (Num_filas_totales) OVER (ORDER BY Num_filas_totales
                                                  ROWS BETWEEN CURRENT ROW AND UNBOUNDED FOLLOWING) AS total_filas,
             MAX (Nro_filas_particion) OVER (PARTITION BY coincidencia) AS MAX_coincidencia
            FROM
             filas
 ),
  pct AS
 (
SELECT
 *,
 CAST (MAX_coincidencia / CAST (total_filas AS decimal (8,2)) AS decimal (8,2)) AS pct_coincidencia
FROM
 máximos
```

```
 )
SELECT
  id_cliente,
  id_producto,
  cantidades,
  rank_más_vendidos,
  mg_x_prod,
  rank_mayor_margen,
  coincidencia AS más_margen_y_más_vendido,
  pct_coincidencia
FROM
  pct
ORDER BY
  id_cliente ASC,
  rank_mayor_margen ASC
```

MUESTRA DEL RESULTADO

Results Messages

	id_cliente	id_producto	cantidades	rank_más_vendidos	mg_x_prod	rank_mayor_margen	más_margen_y_más_vendido	pct_coincidencia
1	1	28	17	4	1071.24	1	NO	0.66
2	1	63	20	3	647.89	2	SI	0.34
3	1	71	20	3	464.20	3	SI	0.34
4	2	72	10	1	1347.78	1	SI	0.34
5	2	19	7	2	183.44	2	SI	0.34
6	2	14	3	4	65.83	3	NO	0.66
7	3	53	25	5	1113.60	1	NO	0.66
8	3	57	5	10	494.52	2	NO	0.66
9	3	59	15	8	387.75	3	NO	0.66
10	4	53	18	11	1113.60	1	NO	0.66
11	4	20	50	4	563.76	2	NO	0.66

CONCLUSION DEL ANÁLISIS

Podemos afirmar que realizando una comparativa de productos vendidos a los distintos clientes, en el 66 % de los casos, los productos vendidos que tienen mejor margen **NO son los más vendidos a los clientes**; del TOP 3 de los productos que tienen mejor margen sólo el 34 % también coinciden que son los más vendidos a esos clientes, es decir, *los productos MENOS vendidos tienen MAS MARGEN y viceversa*.

4. KPIS márgenes por producto:

SCRIPT

```
WITH unitarios AS
 (
            SELECT
             T2.fecha_orden,
             T1.id_producto,
             T1.cantidad,
             T1.preciounitario AS Pcu,
             T3.preciounitario * (1- T1.descuento) AS Pvu_neto,
             (T3.preciounitario * (1- T1.descuento)) - T1.preciounitario AS mg_unit_pdo
            FROM
             Ventas.DetalleOrden T1
             INNER JOIN Ventas.Ordenes T2
             ON T1.id_orden = T2.id_orden
             INNER JOIN Produccion.Productos T3
             ON T3.id_producto = T1.id_producto
 ),
  pdo AS  -- CALCULO DE MARGENES  POR AÑO Y PRODUCTO
 (
            SELECT
              YEAR (fecha_orden) AS año,
              id_producto,
              CAST (SUM (cantidad * mg_unit_pdo) AS decimal (8,2)) AS mg_prod
             FROM
               unitarios
             GROUP BY
              YEAR (fecha_orden),
              id_producto
 ),
  anterior AS -- CALCULO DE MARGENES  DEL AÑO ANTERIOR  POR AÑO Y PRODUCTO
 (
            SELECT
             *,
             LAG (mg_prod, 1, 0) OVER (PARTITION  BY id_producto ORDER BY año) AS mg_año_anterior
            FROM
             pdo
 ),
  KPIabs AS  -- CALCULO DE KPI ABSOLUTO SIN TOMAR EN  CUENTA  LOS MARGENES  "0" DEL AÑO ANTERIOR
 (
            SELECT
             *,
             CAST (mg_prod  - mg_año_anterior AS decimal (8,2)) AS KPI_mg_abs
            FROM
             anterior
            WHERE
             mg_año_anterior IS DISTINCT  FROM 0
 ) -- CONSULTA  FINAL  Y RESULTADO
SELECT
 *,
 CONCAT (CAST (KPI_mg_abs / mg_año_anterior AS decimal (8,2))* 100, ' %') AS KPI_mg_pct
FROM
 KPIabs
WHERE
 mg_prod > 1
 AND mg_año_anterior > 1
```

NOTA: en esta consulta se ha omitido tanto el primer año para el cálculo del KPI como los valores negativos.

MUESTRA DEL RESULTADO

Results | Messages

	año	id_producto	mg_prod	mg_año_anterior	KPI_mg_abs	KPI_mg_pct
1	2021	2	47.50	337.25	-289.75	-86.00 %
2	2021	3	100.00	60.00	40.00	67.00 %
3	2021	7	13.50	90.00	-76.50	-85.00 %
4	2021	14	31.39	151.13	-119.74	-79.00 %
5	2021	19	20.52	231.00	-210.48	-91.00 %
6	2021	23	87.30	189.00	-101.70	-54.00 %
7	2021	24	5.63	127.13	-121.50	-96.00 %
8	2021	26	440.70	232.05	208.65	90.00 %
9	2021	27	440.00	187.38	252.62	135.00 %
10	2021	28	376.44	1045.92	-669.48	-64.00 %

5. Categorizar los márgenes de los productos y realizar una comparativa entre márgenes por categorías y márgenes acumulados:

Para ello usamos la vista rankMgProdyCliente que consolidamos con la tabla Categorias y la tabla Productos y obtenemos los márgenes obtenidos por cada producto:

SCRIPT

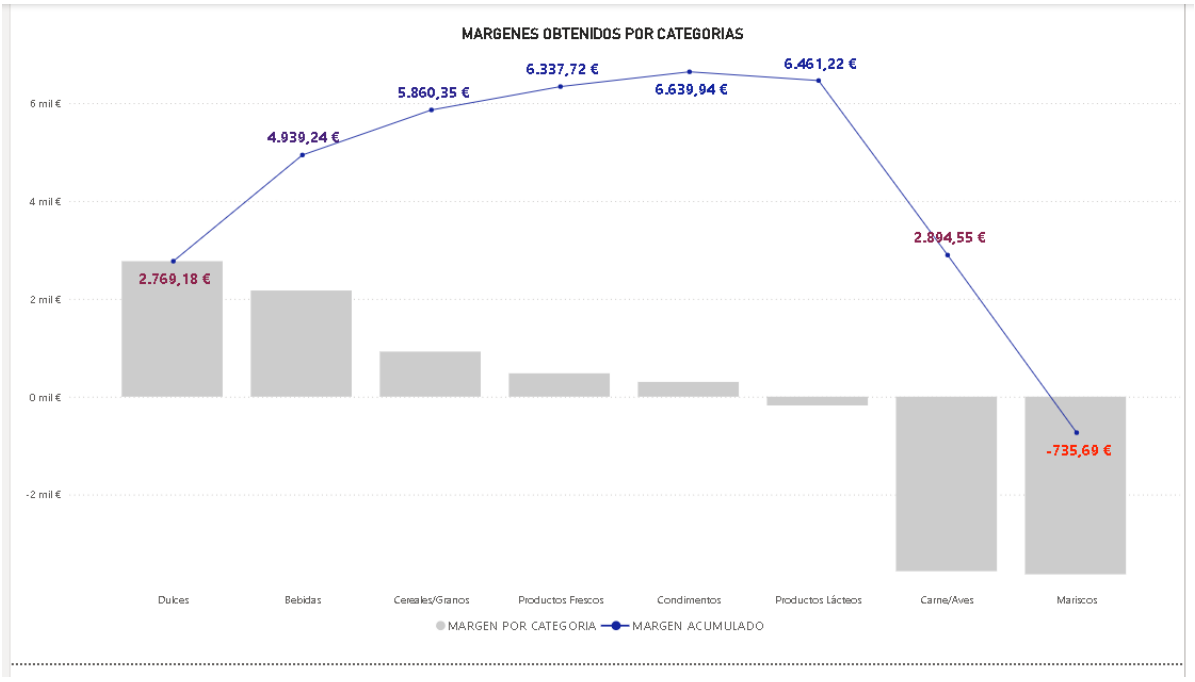
```
WITH promedios AS
 (
            SELECT
              A.nombrecategoria,
              B.id_producto,
              AVG (C.mg_x_prod) AS mg_prod
            FROM
              Produccion.Categorias A
              INNER JOIN Produccion.Productos B
              ON A.id_categoria = B.id_categoria
              LEFT JOIN rankMgProdyCliente C
              ON B.id_producto = C.id_producto
            GROUP BY
              A.nombrecategoria,
              B.id_producto
 ) – CALCULO DEL TOTAL POR CATEGORIA DE PRODUCTO Y TOTAL ACUMULADO Y RESULTADO
SELECT
 nombrecategoria,
 CAST (SUM (mg_prod) AS decimal (8,2)) AS total_x_cat,
 CAST (SUM (SUM (mg_prod)) OVER (ORDER BY (SUM (mg_prod)) DESC ) AS decimal (8,2)) AS total_acum
FROM
 promedios
GROUP BY
 nombrecategoria
ORDER BY
 total_x_cat DESC
```

MUESTRA DEL RESULTADO

Results | Messages

	nombrecategoria	total_x_cat	total_acum
1	Dulces	2769.18	2769.18
2	Bebidas	2170.06	4939.24
3	Cereales/Granos	921.11	5860.35
4	Productos Frescos	477.37	6337.72
5	Condimentos	302.22	6639.94
6	Productos Lácteos	-178.72	6461.22
7	Carne/Aves	-3566.67	2894.55
8	Mariscos	-3630.24	-735.69

VISUALIZACION EN POWER BI (GRAFICO DE COLUMNAS Y LINEAS)



NOTA: En este gráfico combinado se ha aplicado un degradado de color al margen acumulado, de tal forma que se observa un azul oscuro en su pico más alto y un color rojo en su posición más baja.

b) Descuentos en productos vendidos:

1. ¿Existe alguna correlación entre que se apliquen descuentos a clientes en los productos con que estos compren más o menos cantidad de productos?:

SCRIPT

```
WITH descuento AS  -- CLASIFICACION DE LOS DESCUENTOS
 (
            SELECT
             id_producto,
             SUM (cantidad) AS cantidad,
             descuento,
             (CASE WHEN descuento > 0.01 THEN 'SI' ELSE 'NO' END) AS aplica_dcto
            FROM
             Ventas.DetalleOrden T1
            GROUP BY
             id_producto,
             descuento
 ),
  agrup AS – CALCULO DEL TOTAL DE CANTIDADES  POR PRODUCTO
 (
            SELECT
             id_producto,
             cantidad,
             aplica_dcto,
             SUM (cantidad) OVER (PARTITION BY id_producto) total_cant_prod
            FROM
             descuento
            GROUP BY
             id_producto,
             cantidad,
             aplica_dcto
 ),
  subtotales AS  -- CALCULO DEL SUBTOTAL DEL PRODUCTO CLASIFICADO POR DESCUENTOS
 (
            SELECT
             *,
             SUM (cantidad) OVER (PARTITION BY id_producto, aplica_dcto) AS subtotal_prod
            FROM
             agrup
 ),
  agrup1 AS  -- AGRUPACION DE LA CONSULTA
 (
            SELECT
             id_producto,
             aplica_dcto,
             subtotal_prod,
             total_cant_prod
            FROM
             subtotales
            GROUP BY
             id_producto,
             aplica_dcto,
             subtotal_prod,
             total_cant_prod
 ),
  pct AS  -- OBTENCION  DEL PORCENTAJE
 (
            SELECT
             *,
             CAST (subtotal_prod / CAST (total_cant_prod AS decimal (8,2)) AS decimal (8,2)) AS pct
            FROM
             agrup1
 ),
  clasif AS – CLASIFICACION DE PRODUCTOS SI APLICA O NO DESCUENTOS
 (
            SELECT
             id_producto,
```

```
            SUM (CASE WHEN aplica_dcto = 'SI' THEN pct ELSE 0 END) AS Cant_CON_dctos,
            SUM (CASE WHEN aplica_dcto = 'NO' THEN pct ELSE 0 END) AS Cant_SIN_dctos
          FROM
            pct
          GROUP BY
            id_producto
 ),
  respuesta AS  -- CLASIFICACION DE LA RESPUESTA  A OBTENER
 (
          SELECT
            *,
            (CASE WHEN Cant_CON_dctos > Cant_SIN_dctos THEN 'SI' ELSE 'NO' END) AS vendo_más_aplic_dctos
          FROM
            clasif
 ),
  particion AS  -- RECLASIFICACION DE PRODUCTOS POR RESPUESTA
 (
          SELECT
            vendo_más_aplic_dctos,
            COUNT (*) AS total_num,
            SUM (COUNT (*)) OVER () AS total_denom
          FROM
            respuesta
          GROUP BY
            vendo_más_aplic_dctos
 ) – CONSULTA FINAL Y RESULTADO
SELECT
 vendo_más_aplic_dctos,
 CONCAT (CAST (CAST (total_num AS decimal (8,2)) / total_denom AS decimal (8,2)) * 100, ' %') AS respuesta
FROM
 particion
```

MUESTRA DEL RESULTADO Y CONCLUSION DEL ANALISIS

Results | Messages

	vendo_más_aplic_dctos	respuesta
1	NO	70.00 %
2	SI	30.00 %

---- ANALISIS FLETES ----

a) Totales y acumulados de fletes cobrados:

SCRIPT

```
WITH T1 AS  -- DATOS INICIALES
 (
            SELECT
              YEAR (A.fecha_orden) AS Año,
              A.paistransporte AS pais,
              SUM (A.flete) AS flete_anual_pais
            FROM
              Ventas.Ordenes A
            GROUP BY
              YEAR (A.fecha_orden),
              A.paistransporte
 ),
  T2 AS – CALCULO DEL FLETE TOTAL POR AÑO
 (
            SELECT
              *,
              SUM (flete_anual_pais) OVER (PARTITION BY año) AS flete_total_anual
            FROM
              T1
 ),
  T3 AS  -- CALCULO DEL FLETE ACUMULADO POR PAIS Y AÑO
 (
            SELECT
              *,
              SUM (flete_anual_pais) OVER (PARTITION BY año  ORDER BY flete_anual_pais DESC
                                    ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS flete_anual_acum
            FROM
              T2
 ),
  T4 AS – CALCULO DEL PORCENTAJE DE PARETO
 (
            SELECT
              *,
              flete_anual_acum / flete_total_anual AS pct_pareto
            FROM
              T3
 ) -- CONSULTA FINAL Y RESULTADO
SELECT
 Año,
 pais,
 flete_anual_pais,
 flete_anual_acum,
 flete_total_anual,
 pct_pareto
FROM
 T4
```

MUESTRA DEL RESULTADO

Results | Messages

	Año	pais	flete_anual_pais	flete_anual_acum	flete_total_anual	pct_pareto
1	2020	USA	1972,35	1972,35	10279,87	0,1918
2	2020	Alemania	1957,14	3929,49	10279,87	0,3822
3	2020	Austria	1255,09	5184,58	10279,87	0,5043
4	2020	Brasil	1223,89	6408,47	10279,87	0,6233
5	2020	Francia	663,67	7072,14	10279,87	0,6879
6	2020	Reino Unido	566,35	7638,49	10279,87	0,743
7	2020	Venezuela	473,16	8111,65	10279,87	0,789
8	2020	Irlanda	416,78	8528,43	10279,87	0,8296
9	2020	Suecia	342,78	8871,21	10279,87	0,8629
10	2020	Mexico	244,37	9115,58	10279,87	0,8867
11	2020	España	219,08	9334,66	10279,87	0,908
12	2020	Finland	197,43	9532,09	10279,87	0,9272

VISUALIZACION EN POWER BI (GRAFICO DE CASCADA / PARETO)

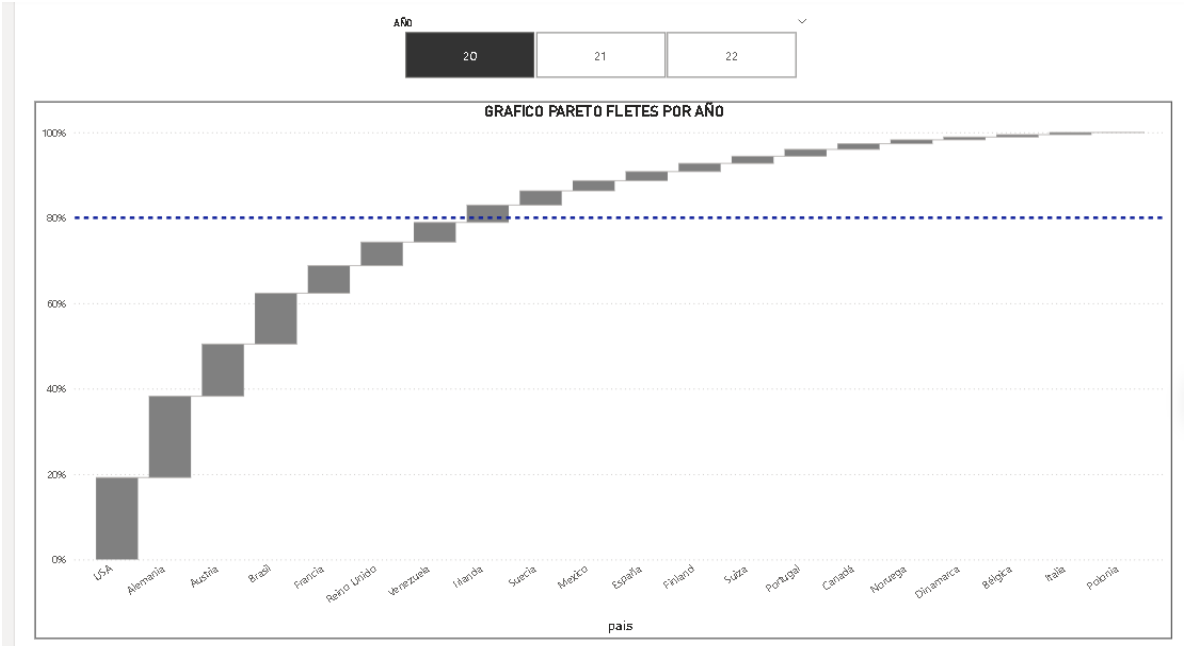


Gráfico con la representación del año 2020

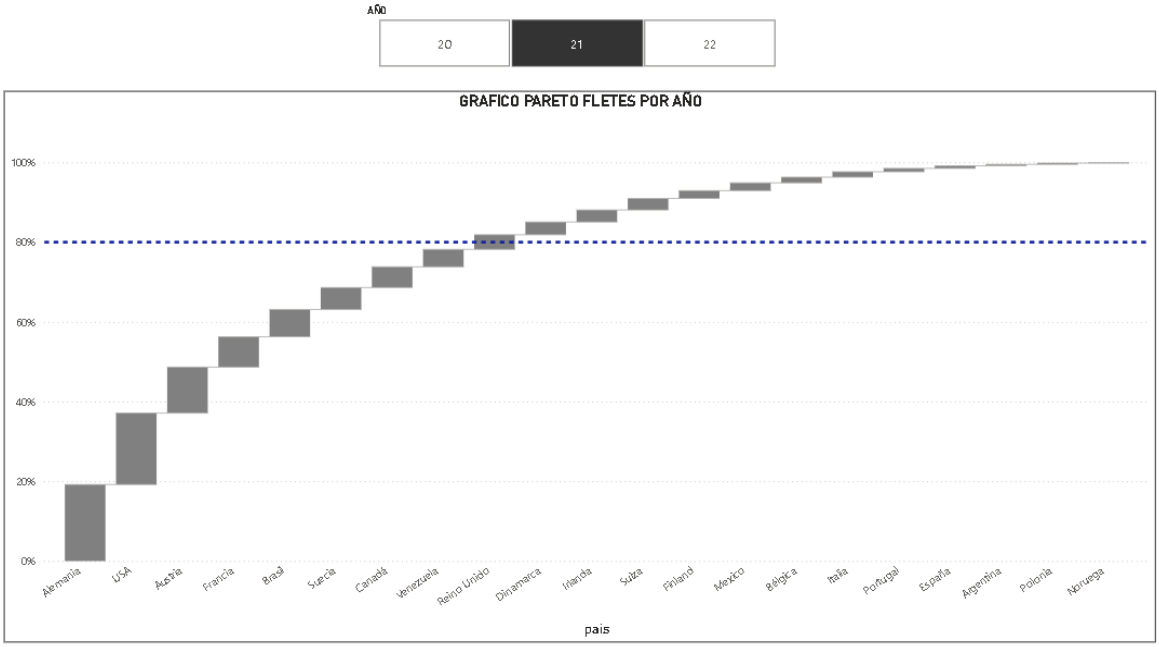


Gráfico con la representación del año 2021

Aunque esta opción es válida, se requiere un gráfico de columnas y lineas combinado para la visualización del porcentaje de Pareto.

VISUALIZACION EN POWER BI (GRAFICO DE COLUMNAS Y LINEAS / PARETO)

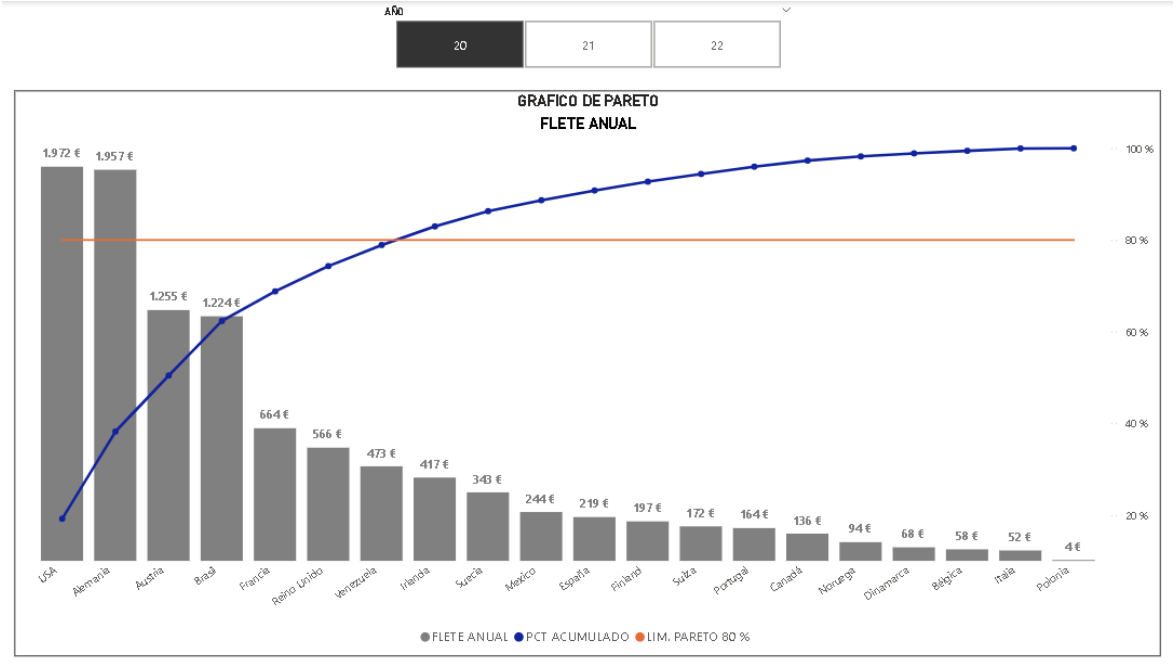


Gráfico con la representación del año 2020 (versión gráfico combinado)

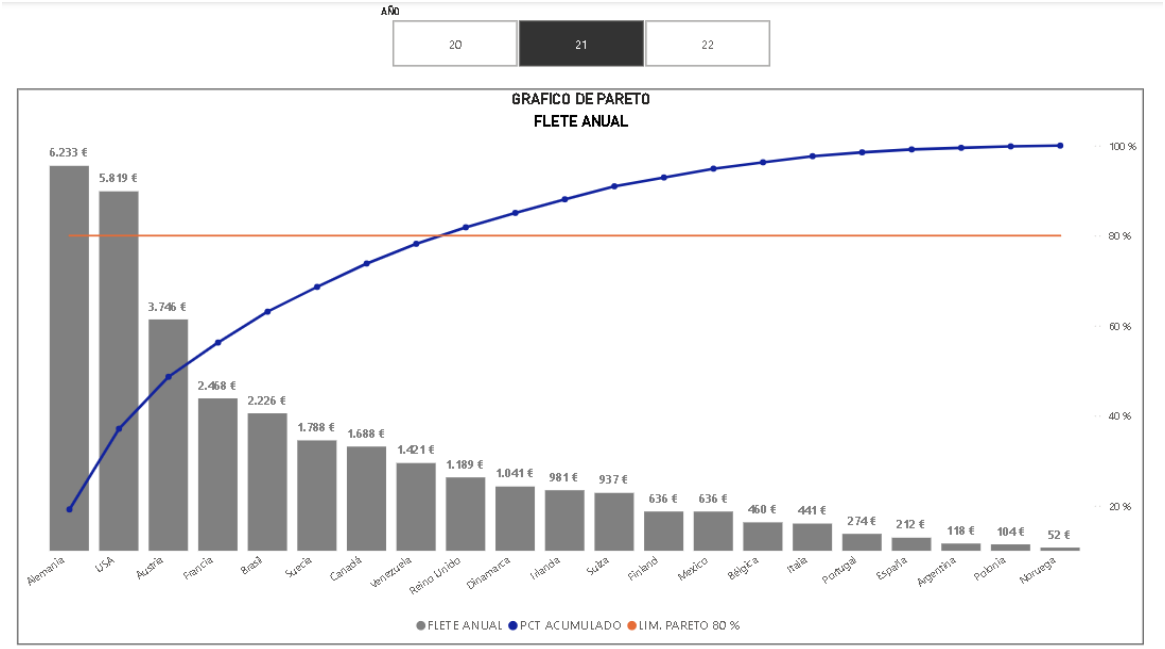


Gráfico con la representación del año 2021 (versión gráfico combinado)

b) AVG acumulados vs AVG móviles fletes general para analizar las tendencias de los datos:

SCRIPT

```
WITH inicial AS  -- DATOS INICIALES
 (
            SELECT
              YEAR (T1.fecha_orden) AS año,
              MONTH (T1.fecha_orden) AS mes,
              SUM (T1.flete) AS flete
            FROM
              Ventas.Ordenes T1
            GROUP BY
              YEAR (T1.fecha_orden),
              MONTH (T1.fecha_orden)
 ),
  promedios AS  -- CALCULO DE PROMEDIOS Y ACUMULADOS
 (
            SELECT
              '01' AS día,
              mes,
              año,
              flete,
              CAST (AVG (flete) OVER (ORDER BY año, mes
                                      ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS decimal (8,2))
                                      AS AVG_flete_acum,
             LAG (flete, 1, 0) OVER (ORDER BY año, mes) AS flete_ant,
              CAST (AVG (flete) OVER (ORDER BY año, mes
                                       ROWS BETWEEN 2 PRECEDING AND CURRENT ROW) AS decimal (8,2)) AS AVG_movil_3M_flete
            FROM
              inicial
 ),
  anterior AS  -- REPETICION DE CONSULTA
 (
            SELECT
              *
            FROM
            promedios
 ),
```

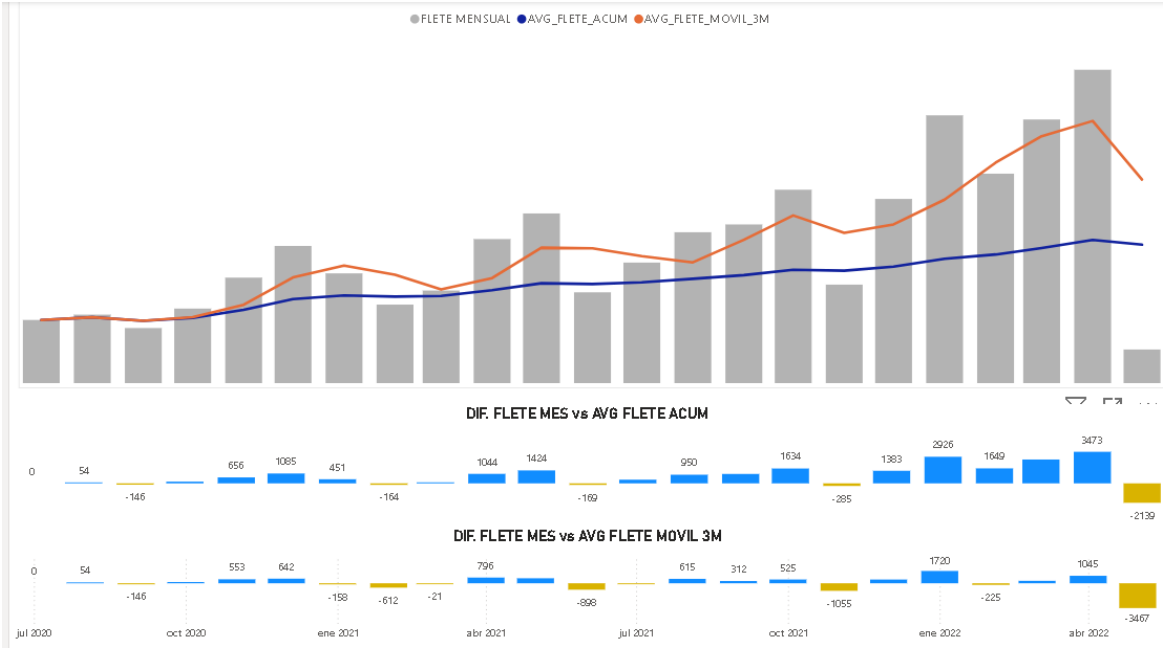
```
  diferencias AS  -- CALCULO DE PROMEDIO MOVIL AÑO ANTERIOR Y CLASIFICACION DE LA TENDENCIA
(
            SELECT
              *,
              LAG (AVG_movil_3M_flete, 1, 0) OVER (ORDER BY año, mes) AS AVG_movil_3M_flete_ant,
              (CASE WHEN AVG_flete_acum > flete_ant THEN 'creciente' ELSE 'decreciente' END) AS tend_AVG_acum
            FROM
              anterior
),
  tendmovil AS  -- CLASIFICACION DE LA TENDENCIA  MOVIL
(
            SELECT
              *,
              (CASE WHEN AVG_movil_3M_flete > AVG_movil_3M_flete_ant THEN 'creciente' ELSE 'decreciente'  END)
               AS tend_AVG_movil_3M_flete
            FROM
              diferencias
),
  difmensual AS  --  CALCULO DE DIFERENCIAS  ENTRE  PERIODOS
(
            SELECT
              *,
              AVG_flete_acum - flete_ant AS dif_mensual_tend_AVG_acum,
              AVG_movil_3M_flete - AVG_movil_3M_flete_ant AS di_mens_tend_AVG_movil_3M
            FROM
              tendmovil
 ) – CONSULTA  FINAL Y RESULTADO
SELECT
  día,
  mes,
  año,
  flete,
  AVG_flete_acum,
  AVG_movil_3M_flete,
  (CASE WHEN flete > AVG_flete_acum THEN 'creciente' ELSE 'decreciente' END) AS tend_AVG_flete_acum,
  (CASE WHEN flete > AVG_movil_3M_flete THEN 'creciente' ELSE 'decreciente' END) AS tend_AVG_movil_3M_flete
FROM
  difmensual
```

MUESTRA DEL RESULTADO

Results | Messages

	día	mes	año	flete	AVG_flete_acum	AVG_movil_3M_flete	tend_AVG_flete_acum	tend_AVG_movil_3M_flete
4	01	10	2020	1520,59	1332.36	1347.08	creciente	creciente
5	01	11	2020	2151,86	1496.26	1598.64	creciente	creciente
6	01	12	2020	2798,59	1713.31	2157.01	creciente	creciente
7	01	1	2021	2238,98	1788.41	2396.48	creciente	decreciente
8	01	2	2021	1601,45	1765.04	2213.01	decreciente	decreciente
9	01	3	2021	1888,81	1778.79	1909.75	creciente	decreciente
10	01	4	2021	2939,10	1894.82	2143.12	creciente	creciente
11	01	5	2021	3461,40	2037.24	2763.10	creciente	creciente
12	01	6	2021	1852,65	2021.86	2751.05	decreciente	decreciente
13	01	7	2021	2458,72	2055.46	2590.92	creciente	decreciente
14	01	8	2021	3078,27	2128.52	2463.21	creciente	creciente
15	01	9	2021	3237,05	2202.42	2924.68	creciente	creciente
16	01	10	2021	3945,53	2311.36	3420.28	creciente	creciente
17	01	11	2021	2008,85	2293.57	2963.81	decreciente	decreciente

VISUALIZACION EN POWER BI (GRAFICO DE COLUMNAS Y LINEAS + GRAFICOS DE COLUMNAS PARA MOSTRAR LAS DIFERENCIAS)



El gráfico inicial muestra los fletes mensuales en columnas grises; si el flete mensual bate al promedio acumulado hasta la fecha (línea azul) como cuando bate al promedio del flete móvil a 3 meses (línea naranja), se ve claramente como la columna rompe las líneas por arriba. En los dos gráficos siguientes se representan las diferencias absolutas de cada mes, representado por colores para un entendimiento de la visualización más sencillo y comprensible; en esta ocasión, el requerimiento pedía mostrar las diferencias positivas en color azul y las diferencias negativas en amarillo.

CONCLUSION DEL ANALISIS

Se puede observar en líneas generales una tendencia creciente de los fletes cobrados, aunque en el último mes se observa una fuerte caída de los fletes.

c) AVG flete unitario por producto:

SCRIPT

```
WITH inicial AS --DATOS INICIALES
 (
            SELECT
              T3.fecha_orden,
              T1.id_orden,
              T3.id_cliente,
              T1.id_producto,
              T1.cantidad,
              T1.preciounitario AS Pcu,
              T2.preciounitario AS Pvu,
              T1.descuento,
              CAST (T2.preciounitario * (1-T1.descuento) AS decimal (8,2)) AS Pvu_descuento,
              T3.flete
            FROM
              Ventas.DetalleOrden T1
              INNER JOIN Produccion.Productos T2
              ON T1.id_producto = T2.id_producto
```

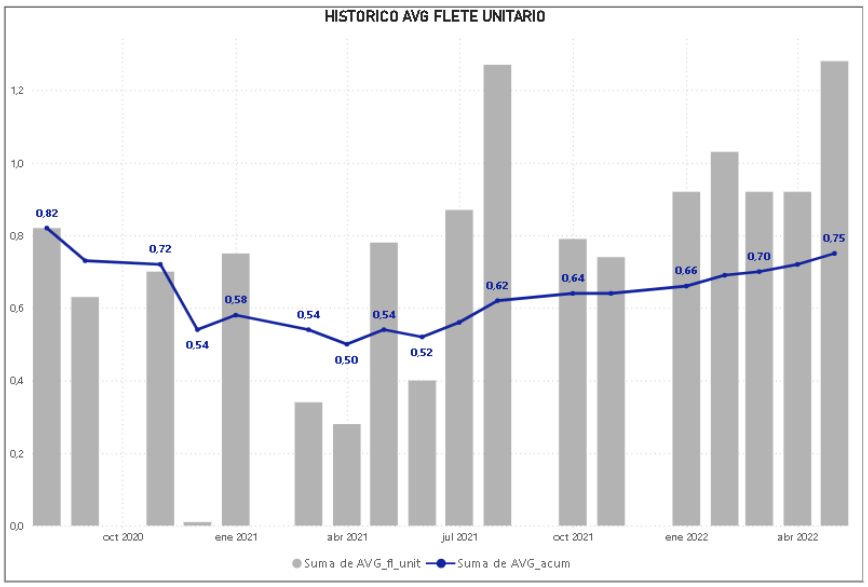
```
            INNER JOIN Ventas.Ordenes T3
            ON T1.id_orden = T3.id_orden
),
  pedidos AS  -- OBTENEMOS  EL SUBTOTAL Y TOTAL POR PEDIDO
(
            SELECT
             *,
             SUM (cantidad * Pvu_descuento) OVER (PARTITION BY id_orden, id_producto) AS subtotal_pdo,
             SUM (cantidad * Pvu_descuento) OVER (PARTITION BY id_orden) AS Total_pdo
            FROM
             inicial
),
  proporcion AS -- CALCULAMOS LA PROPORCION DEL SUBTOTAL DEL PEDIDO SOBRE EL TOTAL DEL PEDIDO
(
            SELECT
             *,
             subtotal_pdo / Total_pdo AS proporc_pdo
            FROM
             pedidos
),
  reparto AS -- REPARTIMOS EL FLETE TOTAL ENTRE  LA PROPORCION DEL PEDIDO PARA OBTENER EL FLETE POR SUBTOTAL DE
PEDIDO
(
            SELECT
             *,
             flete * proporc_pdo AS rep_flete
            FROM
             proporcion
),
  fleteunit AS -- OBTENEMOS  EL FLETE UNITARIO  POR PRODUCTO Y PEDIDO
(
            SELECT
             *,
             CAST (rep_flete / cantidad AS decimal (8,2)) AS flete_unit_pdo
            FROM
             reparto
),
  promediomes AS  --OBTENEMOS EL PROMEDIO  DE FLETE  UNITARIO  DE CADA PRODUCTO AGRUPADO POR MES
(
            SELECT
             YEAR (fecha_orden) AS año,
             MONTH (fecha_orden) AS mes,
             id_producto,
             CAST (AVG (flete_unit_pdo) AS decimal (8,2)) AS AVG_fl_unit
            FROM
             fleteunit
            GROUP BY
             YEAR (fecha_orden),
             MONTH (fecha_orden),
             id_producto
) -- EN LA CONSULTA FINAL, OBTENEMOS  EL PROMEDIO FLETE  UNITARIO  HASTA LA FECHA
SELECT
 CONCAT ('01','-', mes,'-', año) AS fecha,
 id_producto,
 AVG_fl_unit,
 CAST (AVG (AVG_fl_unit) OVER (PARTITION BY id_producto
                ORDER BY año, mes) AS decimal (8,2)) AS AVG_acum
FROM
 promediomes
```

MUESTRA DEL RESULTADO

Results | Messages

	fecha	id_producto	AVG_fl_unit	AVG_acum
13	01-11-2021	1	0.74	0.64
14	01-1-2022	1	0.92	0.66
15	01-2-2022	1	1.03	0.69
16	01-3-2022	1	0.92	0.70
17	01-4-2022	1	0.92	0.72
18	01-5-2022	1	1.28	0.75
19	01-7-2020	2	0.68	0.68
20	01-9-2020	2	0.97	0.83
21	01-10-2020	2	0.35	0.67
22	01-12-2020	2	0.56	0.64
23	01-1-2021	2	0.15	0.54
24	01-2-2021	2	0.23	0.49

VISUALIZACION EN POWER BI (GRAFICO DE COLUMNAS Y LINEAS)



En este gráfico podemos interactuar con el id_producto, ya que al seleccionarlo nos da el resultado de ese producto.

---- ANALISIS CANTIDADES VENDIDAS ----

a) Totales mensuales, promedios de cantidades vendidas acumuladas y promedio móvil a 3 meses:

SCRIPT

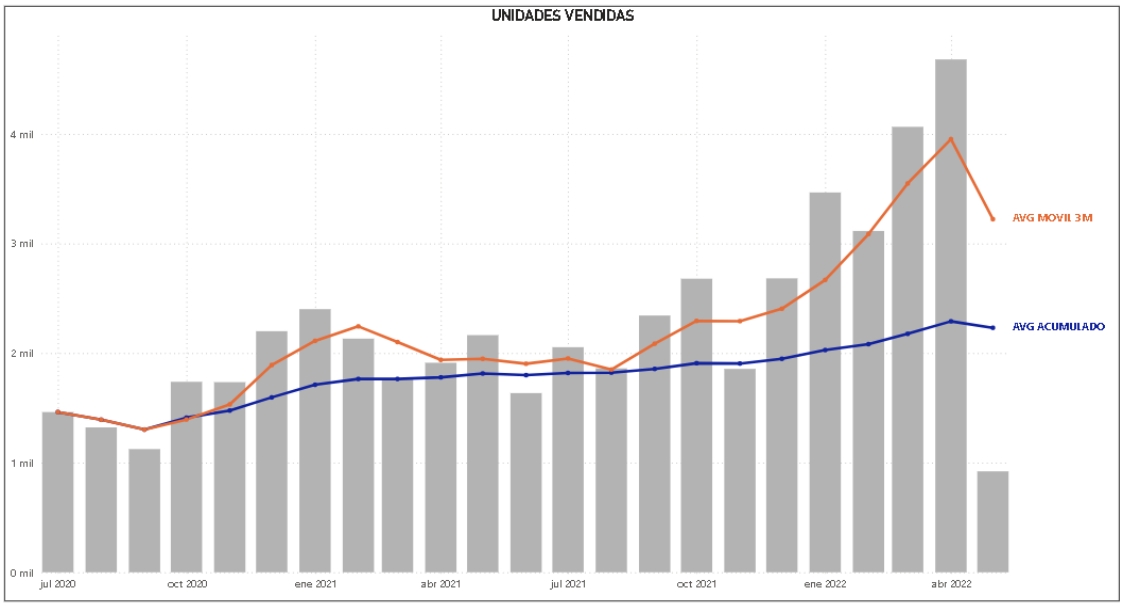
```
WITH totales AS --DATOS INICIALES
 (
            SELECT
              YEAR (T2.fecha_orden) AS Año,
              MONTH (T2.fecha_orden) AS Mes,
              SUM (T1.cantidad) AS total_cantidad
            FROM
              Ventas.DetalleOrden T1
              INNER JOIN
              Ventas.Ordenes T2
              ON T2.id_orden = T1.id_orden
            GROUP BY
              YEAR (T2.fecha_orden),
              MONTH (T2.fecha_orden)
 ),
 promedios AS --OBTENCION DEL PROMEDIO ACUMULADO Y EL PROMEDIO MOVIL 3 MESES
 (
            SELECT
            Año,
            Mes,
            total_cantidad,
            AVG (total_cantidad) OVER (ORDER BY Año, Mes
                                       ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS AVG_acum,
            AVG (total_cantidad) OVER (ORDER BY Año, Mes
                                       ROWS BETWEEN 2 PRECEDING AND CURRENT ROW) AS AVG_movil_3M
            FROM totales
 )
SELECT
 *
FROM
 Promedios
```

MUESTRA DEL RESULTADO

Results | Messages

	Año	Mes	total_cantidad	AVG_acum	AVG_movil_3M
1	2020	7	1462	1462	1462
2	2020	8	1322	1392	1392
3	2020	9	1124	1302	1302
4	2020	10	1738	1411	1394
5	2020	11	1735	1476	1532
6	2020	12	2200	1596	1891
7	2021	1	2401	1711	2112
8	2021	2	2132	1764	2244
9	2021	3	1770	1764	2101
10	2021	4	1912	1779	1938
11	2021	5	2164	1814	1948
12	2021	6	1635	1799	1903

VISUALIZACION EN POWER BI (GRAFICO DE COLUMNAS Y LINEAS)



CONCLUSION DEL ANALISIS

Se puede observar en líneas generales una tendencia creciente de las cantidades vendidas, aunque en el último mes se observa una fuerte caída en ventas.

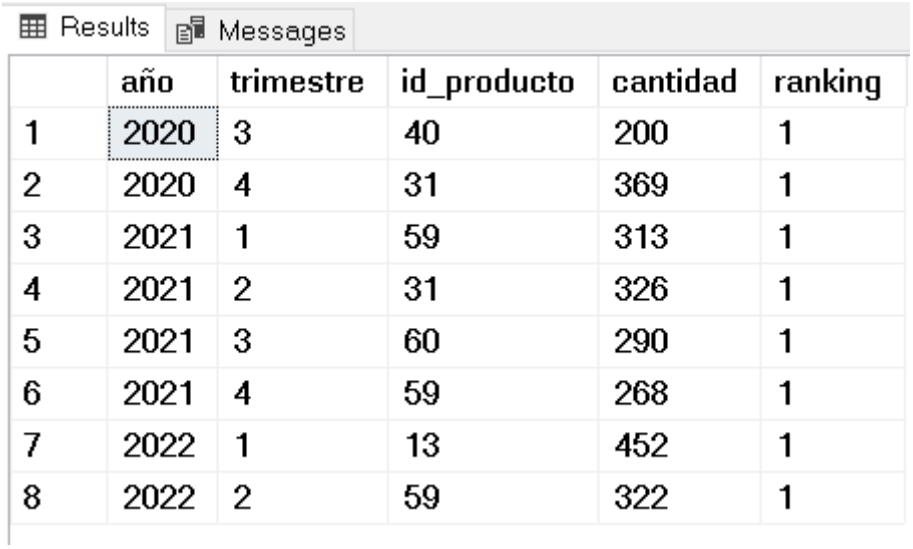
b) Análisis general de cada producto por cliente; *estos análisis no conllevan gráficas*

- Producto más vendido por trimestre:

SCRIPT

```
WITH inicial AS  -- DATOS INICIALES
 (
            SELECT
              YEAR (T2.fecha_orden) AS año,
              DATEPART (QUARTER, T2.fecha_orden) AS trimestre,
              T1.id_producto,
              SUM (T1.cantidad)  AS cantidad
            FROM
              Ventas.DetalleOrden T1
              INNER JOIN Ventas.Ordenes T2
              ON T1.id_orden = T2.id_orden
            GROUP BY
              YEAR (T2.fecha_orden),
              DATEPART (QUARTER, T2.fecha_orden),
              T1.id_producto
 ),
  ranking AS  -- CLASIFICACION POR RANKING DE LAS CANTIDADES VENDIDAS POR AÑO Y TRIMESTRE
 (
            SELECT
              *,
              DENSE_RANK () OVER (PARTITION BY año, trimestre ORDER BY cantidad DESC) AS ranking
            FROM
              inicial
 ) -- CONSULTA FINAL Y RESULTADO
SELECT
 *
FROM
 ranking
WHERE
 ranking = 1
```

MUESTRA DEL RESULTADO



Results | Messages

	año	trimestre	id_producto	cantidad	ranking
1	2020	3	40	200	1
2	2020	4	31	369	1
3	2021	1	59	313	1
4	2021	2	31	326	1
5	2021	3	60	290	1
6	2021	4	59	268	1
7	2022	1	13	452	1
8	2022	2	59	322	1

- Cantidades por cliente y producto anual:

Para este requerimiento en concreto se pide tener una consulta siempre disponible para ser seleccionada en caso de demanda de información, es por lo que se procede a crear una TVF para reutilizar el código y obtener el resultado una vez seleccionado el cliente y el producto.

SCRIPT

```
GO
CREATE OR ALTER FUNCTION dbo.cantClientes_Prod (@cliente INT, @producto INT)
RETURNS TABLE
AS
RETURN
WITH inicial AS
 (
            SELECT
                        YEAR (T2.fecha_orden) AS año,
                        T1.id_producto,
                        T2.id_cliente,
                        SUM (T1.cantidad) AS cantidad
                       FROM
                        Ventas.DetalleOrden T1
                        INNER JOIN Ventas.Ordenes T2
                        ON T1.id_orden = T2.id_orden
                       GROUP BY
                        YEAR (T2.fecha_orden),
                        T1.id_producto,
                        T2.id_cliente
 )
SELECT
 id_cliente,
 id_producto,
 SUM (CASE WHEN año = 2020 THEN cantidad END) AS '2020',
 SUM (CASE WHEN año = 2021 THEN cantidad END) AS '2021',
 SUM (CASE WHEN año = 2022 THEN cantidad END) AS '2022'
FROM
 inicial
WHERE
 id_cliente = @cliente
 AND
 id_producto = @producto
GROUP BY
 id_cliente,
```

```
 id_producto
GO
```

Una vez creada la expresión de tabla, probamos que funcione; para ello seleccionamos el cliente 11 y el producto 13:

SCRIPT

```
SELECT * FROM dbo.cantClientes_Prod (11, 13)
```

RESULTADO

Results | Messages

	id_cliente	id_producto	2020	2021	2022
1	11	13	NULL	8	15

CONCLUSION DEL ANALISIS

El cliente 11 compra del producto 13 en el año 2021 un total de 8 unidades y el año 2022 un total de 15 unidades.

- TOP 5 productos con mayor crecimiento en ventas en términos de cantidad comparado con el mismo período del año pasado:

SCRIPT

```
WITH inicial AS  -- DATOS INICIALES
 (
            SELECT
              YEAR (T2.fecha_orden) AS año,
              DATEPART (QUARTER, T2.fecha_orden) AS trimestre,
              T1.id_producto,
             SUM (T1.cantidad)  AS cantidad
            FROM
              Ventas.DetalleOrden T1
              INNER JOIN Ventas.Ordenes T2
              ON T1.id_orden = T2.id_orden
            GROUP BY
              YEAR (T2.fecha_orden),
              DATEPART (QUARTER, T2.fecha_orden),
              T1.id_producto
 ),
  anterior AS  -- CALCULO DE LAS CANTIDADES  DEL TRIMESTRE  Y AÑO ANTERIOR
 (
            SELECT
              *,
              LAG (cantidad, 4) OVER (PARTITION  BY id_producto ORDER BY año, trimestre) AS cant_trim_año_ant
            FROM
              inicial
 ),
  pct AS – CALCULO DE PORCENTAJE
 (
            SELECT
              *,
              CAST ((CAST (cantidad AS decimal (8,2)) / cant_trim_año_ant)  -1 AS decimal (8,2)) * 100 AS pct
            FROM
              anterior
 ),
  ranking AS  -- CALCULO DEL RANKING DEL  PORCENTAJE  POR AÑO Y TRIMESTRE
 (
```

```
        SELECT
          *,
          DENSE_RANK () OVER (PARTITION BY año, trimestre ORDER BY pct DESC) AS ranking_prod
        FROM
          pct
 ) -- CONSULTA FINAL Y RESULTADO
SELECT
 *
FROM
 ranking
WHERE
 pct IS NOT NULL
 AND ranking_prod BETWEEN 1 AND 5
```

RESULTADO

Results | Messages

	año	trimestre	id_producto	cantidad	cant_trim_año_ant	pct	ranking_prod
1	2021	3	42	201	18	1017.00	1
2	2021	3	46	128	15	753.00	2
3	2021	3	54	120	15	700.00	3
4	2021	3	56	184	26	608.00	4
5	2021	3	69	150	23	552.00	5
6	2021	4	40	100	4	2400.00	1
7	2021	4	45	221	15	1373.00	2
8	2021	4	7	128	10	1180.00	3
9	2021	4	47	127	16	694.00	4
10	2021	4	21	147	25	488.00	5
11	2022	1	13	452	3	14967.00	1
12	2022	1	37	68	1	6700.00	2
13	2022	1	36	165	5	3200.00	3
14	2022	1	67	94	5	1780.00	4
15	2022	1	22	108	12	800.00	5
16	2022	2	2	263	12	2092.00	1
17	2022	2	71	197	9	2089.00	2
18	2022	2	32	86	6	1333.00	3
19	2022	2	46	81	9	800.00	4
20	2022	2	13	207	28	639.00	5

- Tendencia entre periodos de las cantidades vendidas para nuestro producto TOP 1 durante el último año:

SCRIPT

```
WITH inicial AS – CALCULO DEL RANKING DE CANTIDADES POR FECHA
 (
        SELECT
          YEAR (T2.fecha_orden) AS año,
          T1.id_producto,
          SUM (T1.cantidad) AS cantidad,
          DENSE_RANK () OVER (PARTITION BY YEAR (T2.fecha_orden) ORDER BY SUM (T1.cantidad) DESC) AS ranking_prod
        FROM
          Ventas.DetalleOrden T1
          INNER JOIN Ventas.Ordenes T2
          ON T1.id_orden = T2.id_orden
```

```
            GROUP BY
              YEAR (T2.fecha_orden),
              T1.id_producto
),
  TOP1 AS -- OBTENCION  DEL PRODUCTO  MAS VENDIDO  POR AÑO
(
            SELECT
              *
            FROM
              inicial
            WHERE
              ranking_prod = 1
),
cantidades AS -- OBTENCION  DE LAS CANTIDADES  VENDIDAS  POR MES Y AÑO  DE LOS PRODUCTOS  TOP 1
(
            SELECT
              a.año,
              MONTH  (C.fecha_orden) AS mes,
              SUM  (B.cantidad)  AS cantidad,
              a.id_producto
            FROM
              TOP1 A
              LEFT JOIN Ventas.DetalleOrden B
              ON A.id_producto = B.id_producto
              LEFT JOIN Ventas.Ordenes C
              ON B.id_orden = C.id_orden
            WHERE
              año = YEAR (fecha_orden)
            GROUP  BY
              a.año,
              MONTH  (C.fecha_orden),
              a.id_producto
),
  anterior AS --OBTENCION  DE LA COLUMNA  PERIODO  ANTERIOR  PARA VER LA TENDENCIA
(
            SELECT
              *,
              LAG (cantidad, 1, 0) OVER (PARTITION  BY id_producto ORDER BY año, mes) AS cant_anterior
            FROM
              cantidades
 )-- CONSULTA  FINAL, CLASIFICACION  DE LA TENDENCIA  DEL  PRODUCTO  Y RESULTADO
SELECT
 *,
 CASE WHEN  cantidad > cant_anterior THEN 'creciente' ELSE 'decreciente' END AS tendencia
FROM
 anterior
ORDER BY
 año,
 mes
```

RESULTADO

Results | Messages

	año	mes	cantidad	id_producto	cant_anterior	tendencia
1	2020	7	20	31	0	creciente
2	2020	8	55	31	20	creciente
3	2020	10	85	31	55	creciente
4	2020	11	120	31	85	creciente
5	2020	12	164	31	120	creciente
6	2021	1	105	56	0	creciente
7	2021	2	83	56	105	decreciente
8	2021	3	50	56	83	decreciente
9	2021	4	154	56	50	creciente
10	2021	5	30	56	154	decreciente
11	2021	6	100	56	30	creciente
12	2021	7	47	56	100	decreciente
13	2021	8	80	56	47	creciente
14	2021	9	57	56	80	decreciente
15	2021	10	68	56	57	creciente
16	2021	11	97	56	68	creciente
17	2021	12	100	56	97	creciente
18	2022	1	91	13	0	creciente
19	2022	2	73	13	91	decreciente
20	2022	3	288	13	73	creciente
21	2022	4	193	13	288	decreciente
22	2022	5	14	13	193	decreciente

- Cantidades vendidas por país y semestre:

SCRIPT

```
WITH inicial AS – DATOS INICIALES
 (
            SELECT
              YEAR (T2.fecha_orden) AS año,
              MONTH (T2.fecha_orden) AS mes,
              SUM (T1.cantidad) AS cantidad,
              T2.paistransporte AS pais
            FROM
              Ventas.DetalleOrden T1
              INNER JOIN Ventas.Ordenes T2
              ON T1.id_orden = T2.id_orden
            GROUP BY
              YEAR (T2.fecha_orden),
              MONTH (T2.fecha_orden),
              T2.paistransporte
 ) – CONSULTA FINAL, CLASIFICACION DINAMICA POR SEMESTRES Y RESULTADO
SELECT
 pais,
 SUM (CASE WHEN año = 2020 AND mes BETWEEN 7 AND 12 THEN cantidad ELSE 0 END) AS 'semestre2-2020',
 SUM (CASE WHEN año = 2021 AND mes BETWEEN 1 AND 6 THEN cantidad ELSE 0 END) AS 'semestre1-2021',
 SUM (CASE WHEN año = 2021 AND mes BETWEEN 7 AND 12 THEN cantidad ELSE 0 END) AS 'semestre2-2021',
 SUM (CASE WHEN año = 2022 AND mes BETWEEN 1 AND 6 THEN cantidad ELSE 0 END) AS 'semestre1-2022'
FROM
 inicial
GROUP BY
 pais
```

RESULTADO (TABLA PIVOTADA/DINAMICA)

Results | Messages

	pais	semestre2-2020	semestre1-2021	semestre2-2021	semestre1-2022
1	Alemania	1910	2289	2467	2547
2	Argentina	0	71	23	245
3	Austria	949	1056	1291	1871
4	Bélgica	185	397	119	691
5	Brasil	768	831	1226	1422
6	Canadá	297	711	538	438
7	Dinamarca	232	265	501	172
8	España	229	51	89	349
9	Finland	157	391	213	124
10	Francia	658	1027	780	789
11	Irlanda	490	393	406	395
12	Italia	81	210	214	317
13	Mexico	229	391	160	245
14	Noruega	48	13	8	92
15	Polonia	45	0	59	101
16	Portugal	135	275	67	56
17	Reino Un...	580	756	515	891
18	Suecia	375	465	484	911
19	Suiza	232	232	396	415
20	USA	1539	1454	3185	3152
21	Venezuela	442	736	734	1024

- Cantidad promedio de productos comprados por transacción para nuestros clientes recurrentes *(los clientes que realizan 12 pedidos o más al año)*:

SCRIPT

```
WITH AVGcant AS --OBTENCION DEL PROMEDIO DE CANTIDADES POR PEDIDO
 (
            SELECT
             YEAR (T2.fecha_orden) AS año,
             T2.id_cliente,
             T1.id_orden,
             T1.id_producto,
             SUM (T1.cantidad) AS cantidad_x_pdo,
             AVG (SUM (T1.cantidad)) OVER (PARTITION BY T1.id_orden) AS AVG_cant_x_pdo
            FROM
             Ventas.DetalleOrden T1
             INNER JOIN Ventas.Ordenes T2
             ON T1.id_orden = T2.id_orden
            GROUP BY
             YEAR (T2.fecha_orden),
             T2.id_cliente,
             T1.id_orden,
             T1.id_producto
 ),
  pedidos AS --OBTENCION DEL NUMERO DE PEDIDOS POR AÑO
 (
            SELECT
             año,
             id_cliente,
             id_orden,
             AVG_cant_x_pdo,
             DENSE_RANK () OVER (PARTITION BY año, id_cliente ORDER BY id_orden) AS nro_pedidos
            FROM
```

```
            AVGcant
           GROUP BY
            año,
            id_cliente,
            id_orden,
    AVG_cant_x_pdo
 ),
  AVGcliente AS -- PROMEDIO CANTIDADES  POR CLIENTES CON 12 O MAS PEDIDOS AL AÑO (RECURRENTES)
 (
           SELECT
            id_cliente,
            AVG (AVG_cant_x_pdo) OVER (PARTITION BY id_cliente) AS AVG_cant_x_cliente
           FROM
            pedidos
           WHERE
            nro_pedidos >= 12
 )--CONSULTA FINAL, OBTENCION  DEL RESULTADO
SELECT
 id_cliente,
 AVG_cant_x_cliente AS prom_cant_x_pdos
FROM
 AVGcliente
GROUP BY
 id_cliente,
 AVG_cant_x_cliente
```

RESULTADO

Results | Messages

	id_cliente	prom_cant_x_pdos
1	20	36
2	63	49
3	71	37

- Tendencias significativas de cantidades vendidas de productos por categorías:

SCRIPT

```
WITH categorias AS  -- DATOS INICIALES
 (
           SELECT
            YEAR (T2.fecha_orden) AS año,
            MONTH (T2.fecha_orden) AS mes,
            T4.nombrecategoria,
            SUM (T1.cantidad)  AS cantidad
           FROM
            Ventas.DetalleOrden T1
            INNER JOIN Ventas.Ordenes T2
            ON T1.id_orden = T2.id_orden
            INNER JOIN Produccion.Productos T3
            ON T3.id_producto = T1.id_producto
            INNER JOIN Produccion.Categorias T4
            ON T4.id_categoria = T3.id_categoria
           GROUP BY
            YEAR (T2.fecha_orden),
            MONTH (T2.fecha_orden),
            T4.nombrecategoria
 ),
  anterior AS --OBTENCION  DE LA CANTIDADES VENDIDAS EN EL ANTERIOR PERIODO
 (
           SELECT
            *,
```

```
            LAG (cantidad, 1) OVER (PARTITION BY nombrecategoria ORDER BY año, mes) AS cant_ant
          FROM
            categorias
),
 variacion AS  -- CALCULO DEL PORCENTAJE (SOLO CANTIDADES NO NULAS)
(
            SELECT
             *,
             cantidad - cant_ant AS variacion,
             CAST ((cantidad / CAST (cant_ant AS decimal (8,2))) -1 AS decimal (8,2)) * 100 AS pct_var
            FROM
             anterior
            WHERE
             cant_ant IS NOT NULL
),
  tendencia AS --CLASIFICACION DE LAS TENDENCIAS SIGNIFICATIVAS  (muy bajistas y muy alcistas)
(
            SELECT
             *,
             CASE WHEN pct_var < -30 THEN 'Muy bajista'
                  WHEN pct_var > 30 THEN 'Muy alcista'  ELSE NULL END  AS tendencia
            FROM
             variacion
 ) -- CONSULTA FINAL Y RESULTADO
SELECT
 año,
 mes,
 nombrecategoria,
 cantidad,
 cant_ant,
 CONCAT (pct_var, ' %') AS pct_var,
 tendencia
FROM
 tendencia
WHERE
 tendencia IS NOT NULL
ORDER BY
 nombrecategoria,
 año,
 mes
```

MUESTRA DEL RESULTADO

Results Messages

	año	mes	nombrecategoria	cantidad	cant_ant	pct_var	tendencia
1	2021	2	Bebidas	220	330	-33.00 %	Muy bajista
2	2021	3	Bebidas	471	220	114.00 %	Muy alcista
3	2021	4	Bebidas	268	471	-43.00 %	Muy bajista
4	2021	9	Bebidas	174	334	-48.00 %	Muy bajista
5	2021	10	Bebidas	491	174	182.00 %	Muy alcista
6	2021	11	Bebidas	248	491	-49.00 %	Muy bajista
7	2021	12	Bebidas	498	248	101.00 %	Muy alcista
8	2022	2	Bebidas	834	622	34.00 %	Muy alcista
9	2022	5	Bebidas	221	1092	-80.00 %	Muy bajista
10	2020	8	Carne/Aves	111	76	46.00 %	Muy alcista
11	2020	10	Carne/Aves	153	100	53.00 %	Muy alcista
12	2020	11	Carne/Aves	201	153	31.00 %	Muy alcista

c) Análisis de rotación por producto y categoría:

SCRIPT

```
WITH totales AS --DATOS INICIALES
 (
              SELECT
               T4.fecha_orden,
               T3.nombrecategoria,
               T1.id_producto,
               SUM (T1.cantidad) AS total_cantidad
              FROM
               Ventas.DetalleOrden T1
               INNER JOIN Produccion.Productos T2
               ON T1.id_producto = T2.id_producto
               INNER JOIN Produccion.Categorias T3
               ON T3.id_categoria = T2.id_categoria
               INNER JOIN Ventas.Ordenes T4
               ON T4.id_orden = T1.id_orden
              GROUP BY
               T4.fecha_orden,
               T3.nombrecategoria,
               T1.id_producto
 ),
  total_cat AS  --OBTENCION DE TOTAL POR CATEGORIAS
 (
              SELECT
               *,
               SUM (total_cantidad) OVER (PARTITION BY nombrecategoria) AS totales_x_cat
              FROM
               totales
 ),
  fecha_ant AS -- CALCULO DE LA ULTIMA FECHA DE VENTA
 (
              SELECT
               *,
               MIN (fecha_orden) OVER (PARTITION BY id_producto
                                       ORDER BY fecha_orden ROWS BETWEEN 1 PRECEDING AND CURRENT ROW) AS fecha_ant
              FROM
               total_cat
 ),
  dias AS --CALCULO DE LA DIFERENCIA DE DIAS ENTRE PEDIDOS
 (
              SELECT
               *,
               DATEDIFF (DAY, fecha_ant, fecha_orden) AS dias_dsd_ult_vta
              FROM
               fecha_ant
 ),
  promedios AS --CALCULO DE PROMEDIOS
 (
              SELECT
               *,
               AVG (total_cantidad) OVER (PARTITION BY id_producto) AS AVG_cant_vendida_x_prod,
               AVG (dias_dsd_ult_vta) OVER (PARTITION BY id_producto) AS AVG_dias_rotacion_x_prod
              FROM
               dias
 ),
  categorias AS -- CLASIFICACION POR GRUPOS DE ROTACION DE LOS PRODUCTOS
 (
              SELECT
               *,
               (CASE WHEN AVG_dias_rotacion_x_prod <= 21 THEN 'd) Muy_alta'
                     WHEN AVG_dias_rotacion_x_prod >= 22 AND AVG_dias_rotacion_x_prod <= 31 THEN 'c) Alta'
                     WHEN AVG_dias_rotacion_x_prod >= 32 AND AVG_dias_rotacion_x_prod <= 45 THEN 'b) Media'
                     WHEN AVG_dias_rotacion_x_prod >= 46 THEN 'a) Baja' END) AS Rotación
              FROM
               promedios
 ),
```

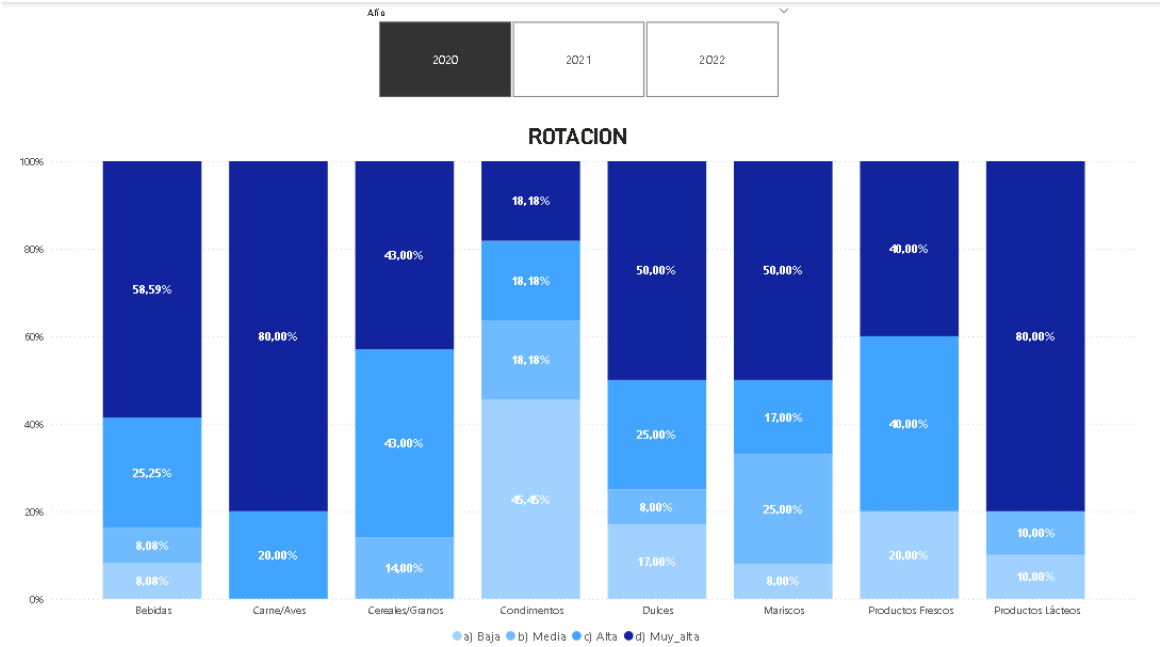
```
  conteo_año AS --OBTENCION  DE CANTIDADES  VENDIDAS
(
            SELECT
              YEAR (fecha_orden) AS año,
              nombrecategoria,
              id_producto,
              SUM (total_cantidad)  AS total_cantidad,
              SUM (SUM (total_cantidad)) OVER (PARTITION  BY
                                                  YEAR (fecha_orden), nombrecategoria) AS total_anual_cant_vendidas,
              Rotación,
              COUNT  (id_producto) OVER (PARTITION  BY YEAR (fecha_orden), nombrecategoria) AS total_prod_x_cat_y_año
            FROM
              categorias
            GROUP  BY
              YEAR (fecha_orden),
              nombrecategoria,
              id_producto,
              Rotación
),
  conteo_rev AS --REPARTO DEL CONTEO  DE CANTIDADES  POR CATEGORIAS
(
            SELECT
             año,
             nombrecategoria,
             Rotación,
             COUNT  (*) AS parciales,
             SUM (COUNT  (*)) OVER (PARTITION  BY año, nombrecategoria) AS total_año
            FROM
              conteo_año
            GROUP  BY
             año,
             nombrecategoria,
             Rotación
 ) -- CONSULTA  FINAL  Y RESULTADO
SELECT
 año,
 nombrecategoria,
 Rotación,
 CAST (parciales / CAST (total_año  AS decimal (8,2)) AS decimal (8,2)) * 100 AS pct_rotación
FROM
 conteo_rev
```

MUESTRA DEL RESULTADO

Results | Messages

	año	nombrecategoria	Rotación	pct_rotación
1	2020	Bebidas	a) Baja	8.00
2	2020	Bebidas	b) Media	8.00
3	2020	Bebidas	c) Alta	25.00
4	2020	Bebidas	d) Muy_alta	58.00
5	2020	Carne/Aves	c) Alta	20.00
6	2020	Carne/Aves	d) Muy_alta	80.00
7	2020	Cereales/Granos	b) Media	14.00
8	2020	Cereales/Granos	c) Alta	43.00
9	2020	Cereales/Granos	d) Muy_alta	43.00
10	2020	Condimentos	a) Baja	45.00

VISUALIZACION EN POWER BI (GRAFICO DE COLUMNAS APILADAS 100%)



CONCLUSION DEL ANÁLISIS

Se puede observar claramente que para el año seleccionado (2020) las categorías de productos que más rotación tienen son las **Carnes/Aves y los Productos Lácteos**, ambos con un 80 % de rotación y la categoría de producto con menos rotación son claramente los **Condimentos**, con un 45,45 % de baja rotación.

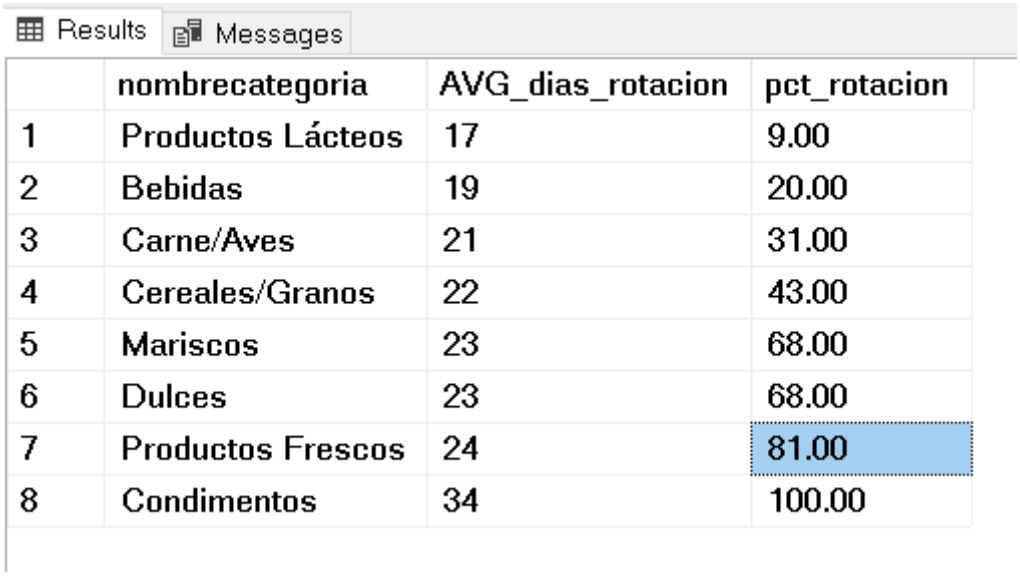
d) Análisis de frecuencia de rotación y de demanda.

SCRIPT 1 (ROTACION)

```
WITH ultvta AS
 (
            SELECT
              T4.fecha_orden,
              T3.nombrecategoria,
              T1.id_producto,
              T1.cantidad,
              LAG (T4.fecha_orden, 1) OVER (PARTITION BY T1.id_producto ORDER BY T4.fecha_orden) AS fecha_ult_vta
            FROM
              Ventas.DetalleOrden T1
              INNER JOIN Produccion.Productos T2
              ON T1.id_producto = T2.id_producto
              INNER JOIN Produccion.Categorias T3
              ON T3.id_categoria = T2.id_categoria
              INNER JOIN Ventas.Ordenes T4
              ON T4.id_orden = T1.id_orden
            GROUP BY
              T4.fecha_orden,
              T3.nombrecategoria,
              T1.id_producto,
              T1.cantidad
 ),
  dias AS --PERIODO EN DIAS ENTRE PEDIDOS DE MISMO PRODUCTO
 (
            SELECT
              *,
              DATEDIFF (DAY, fecha_ult_vta, fecha_orden) AS dias_transc_ult_vta
            FROM
```

```
            ultvta
),
 AVGgral AS -- CALCULO DE PROMEDIOS
(
            SELECT
             *,
             AVG (dias_transc_ult_vta) OVER (PARTITION BY id_producto) AS AVG_rotacion,
             AVG (cantidad) OVER (PARTITION BY id_producto) AS AVG_cant_vendidas
            FROM
             dias
),
 AVGcateg AS --PROMEDIOS POR CATEGORIAS
(
            SELECT
             nombrecategoria,
             AVG (AVG_rotacion) AS AVG_dias_rotacion,
             AVG (AVG_cant_vendidas) AS AVG_cant_vendida
            FROM
             AVGgral
            GROUP BY
             nombrecategoria
),
 acum AS --OBTENCION DE ACUMULADOS
(
            SELECT
             *,
             SUM (AVG_dias_rotacion) OVER (ORDER BY AVG_dias_rotacion ASC) AS dias_rot_acum,
             SUM (AVG_cant_vendida) OVER (ORDER BY AVG_cant_vendida ASC) AS cant_vendida_acum
            FROM
             AVGcateg
),
 ultvalor AS --CALCULO DEL ULTIMO VALOR DISPONIBLE
(
SELECT
 *,
 LAST_VALUE (dias_rot_acum) OVER (ORDER BY dias_rot_acum
                    ROWS BETWEEN CURRENT ROW AND UNBOUNDED FOLLOWING) AS ult_valor_dia_rot_acum,
 LAST_VALUE (cant_vendida_acum) OVER (ORDER BY cant_vendida_acum
                      ROWS BETWEEN CURRENT ROW AND UNBOUNDED FOLLOWING) AS ult_valor_cant_vendida_acum
FROM
 acum
 ),
  pct AS --CALCULO DE PORCENTAJES ACUMULADOS
 (
SELECT
 *,
 CAST (dias_rot_acum / CAST (ult_valor_dia_rot_acum AS decimal (8,2)) AS decimal (8,2)) * 100 AS pct_rotacion,
 CAST (cant_vendida_acum / CAST (ult_valor_cant_vendida_acum AS decimal (8,2)) AS decimal (8,2)) * 100 AS pct_cant_vendidas
FROM
 ultvalor
 )
SELECT
 nombrecategoria,
 AVG_dias_rotacion,
 pct_rotacion
FROM
 pct
ORDER BY
 pct_rotacion ASC
```

RESULTADO



Results | Messages

	nombrecategoria	AVG_dias_rotacion	pct_rotacion
1	Productos Lácteos	17	9.00
2	Bebidas	19	20.00
3	Carne/Aves	21	31.00
4	Cereales/Granos	22	43.00
5	Mariscos	23	68.00
6	Dulces	23	68.00
7	Productos Frescos	24	81.00
8	Condimentos	34	100.00

SCRIPT 2 (DEMANDA)

```
WITH ultvta AS
 (
		SELECT
		  T4.fecha_orden,
		  T3.nombrecategoria,
		  T1.id_producto,
		  T1.cantidad,
		  LAG (T4.fecha_orden, 1) OVER (PARTITION BY T1.id_producto ORDER BY T4.fecha_orden) AS fecha_ult_vta
		FROM
		  Ventas.DetalleOrden T1
		  INNER JOIN Produccion.Productos T2
		  ON T1.id_producto = T2.id_producto
		  INNER JOIN Produccion.Categorias T3
		  ON T3.id_categoria = T2.id_categoria
		  INNER JOIN Ventas.Ordenes T4
		  ON T4.id_orden = T1.id_orden
		GROUP BY
		  T4.fecha_orden,
		  T3.nombrecategoria,
		  T1.id_producto,
		  T1.cantidad
 ),
  dias AS --PERIODO EN DIAS ENTRE PEDIDOS DE MISMO PRODUCTO
 (
		SELECT
		  *,
		  DATEDIFF (DAY, fecha_ult_vta, fecha_orden) AS dias_transc_ult_vta
		FROM
		  ultvta
 ),
  AVGgral AS -- CALCULO DE PROMEDIOS
 (
		SELECT
		  *,
		  AVG (dias_transc_ult_vta) OVER (PARTITION BY id_producto) AS AVG_rotacion,
		  AVG (cantidad) OVER (PARTITION BY id_producto) AS AVG_cant_vendidas
		FROM
		  dias
 ),
  AVGcateg AS --PROMEDIOS POR CATEGORIAS
 (
		SELECT
		  nombrecategoria,
```

```
            AVG (AVG_rotacion) AS AVG_dias_rotacion,
            AVG (AVG_cant_vendidas) AS AVG_cant_vendida
           FROM
            AVGgral
           GROUP BY
            nombrecategoria
 ),
  acum AS --OBTENCION DE ACUMULADOS
 (
           SELECT
            *,
            SUM (AVG_dias_rotacion) OVER (ORDER BY AVG_dias_rotacion ASC) AS dias_rot_acum,
            SUM (AVG_cant_vendida) OVER (ORDER BY AVG_cant_vendida ASC, nombrecategoria) AS cant_vendida_acum
           FROM
            AVGcateg
 ),
  ultvalor AS --CALCULO DEL ULTIMO VALOR DISPONIBLE
 (
SELECT
 *,
 LAST_VALUE (dias_rot_acum) OVER (ORDER BY dias_rot_acum
                     ROWS BETWEEN CURRENT ROW AND UNBOUNDED FOLLOWING) AS ult_valor_dia_rot_acum,
 LAST_VALUE (cant_vendida_acum) OVER (ORDER BY cant_vendida_acum
                       ROWS BETWEEN CURRENT ROW AND UNBOUNDED FOLLOWING) AS ult_valor_cant_vendida_acum
FROM
 acum
 ),
  pct AS --CALCULO DE PORCENTAJES ACUMULADOS
 (
SELECT
 *,
 CAST (dias_rot_acum / CAST (ult_valor_dia_rot_acum AS decimal (8,2)) AS decimal (8,2)) * 100 AS pct_rotacion,
 CAST (cant_vendida_acum / CAST (ult_valor_cant_vendida_acum AS decimal (8,2)) AS decimal (8,2)) * 100 AS pct_cant_vendidas
FROM
 ultvalor
 )
SELECT
 nombrecategoria,
 AVG_cant_vendida,
 pct_cant_vendidas
FROM
 pct
ORDER BY
 pct_cant_vendidas ASC
```

RESULTADO

Results | Messages

	nombrecategoria	AVG_cant_vendida	pct_cant_vendidas
1	Productos Frescos	21	12.00
2	Cereales/Granos	22	24.00
3	Mariscos	22	36.00
4	Bebidas	23	48.00
5	Carne/Aves	23	61.00
6	Dulces	23	74.00
7	Condimentos	24	87.00
8	Productos Lácteos	24	100.00

CONCLUSION DEL ANÁLISIS

Podemos afirmar que en el 81 % de las ventas totales, la rotación de los productos es de **24 DÍAS O MENOS**.

También podemos afirmar que en el 74 % de las ventas totales, la demanda de los productos es de **23 UNIDADES O MENOS.**

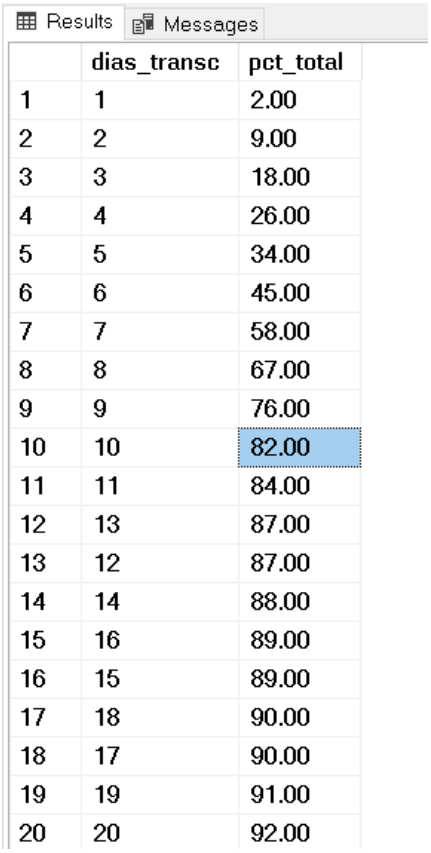
---- ANALISIS TIEMPOS ----

a) Que tiempo transcurre entre la realización de los pedidos hasta que se carga el pedido en transporte?

SCRIPT

```
WITH dias AS
 (
            SELECT
             T1.id_orden,
             T1.fecha_orden,
             T1.fecha_transporte,
             DATEDIFF (DAY, T1.fecha_orden, T1.fecha_transporte) AS dias_transc
            FROM
             Ventas.Ordenes T1
 ),
  totalreg AS  --CALCULO DE LOS REGISTROS  PARA SU VALORACION
 (
            SELECT
             dias_transc,
             COUNT (*) AS total,
             SUM (COUNT (*)) OVER (ORDER BY dias_transc ASC) AS total_acum
            FROM
             dias
            WHERE
             dias_transc IS NOT NULL
            GROUP BY
             dias_transc
 ),
  ultvalor AS --CALCULO DEL ULTIMO VALOR DISPONIBLE
 (
            SELECT
             *,
             LAST_VALUE (total_acum) OVER (ORDER BY total_acum
                                           ROWS BETWEEN  CURRENT  ROW AND UNBOUNDED  FOLLOWING) AS ult_valor
            FROM
             totalreg
 ),
  pct AS --CALCULO DEL PORCENTAJE  SOBRE EL TOTAL
 (
            SELECT
             *,
             CAST (total_acum  / CAST (ult_valor AS decimal (8,2)) AS decimal (8,2)) * 100 AS pct_total
            FROM
             ultvalor
 )-- CONSULTA  FINAL Y RESULTADO
SELECT
 dias_transc,
 pct_total
FROM
 pct
ORDER BY
 pct_total ASC
```

MUESTRA DEL RESULTADO



Results | Messages

	dias_transc	pct_total
1	1	2.00
2	2	9.00
3	3	18.00
4	4	26.00
5	5	34.00
6	6	45.00
7	7	58.00
8	8	67.00
9	9	76.00
10	10	82.00
11	11	84.00
12	13	87.00
13	12	87.00
14	14	88.00
15	16	89.00
16	15	89.00
17	18	90.00
18	17	90.00
19	19	91.00
20	20	92.00

CONCLUSION DEL ANÁLISIS

El 82 % de los pedidos una vez realizado por el cliente, se cargan en el transporte en **10 DÍAS O MENOS.**

b) Análisis de frecuencia de compras de los clientes.

SCRIPT

```
WITH num AS --DATOS INICIALES
 (
            SELECT
              T1.id_cliente,
              SUM (COUNT (*)) OVER (PARTITION BY id_cliente) AS Nro_pdos_cliente
            FROM
              Ventas.Ordenes T1
            GROUP BY
              T1.id_cliente
 ),
  pdos AS -- CLASIFICACION DE CLIENTES SEGUN NUMERO DE PEDIDOS REALIZADOS
 (
            SELECT
              CASE WHEN Nro_pdos_cliente BETWEEN 0 AND 5 THEN 'a) 5=>'
                        WHEN Nro_pdos_cliente BETWEEN 6 AND 10 THEN 'b) 6<10'
                        WHEN Nro_pdos_cliente BETWEEN 11 AND 15 THEN 'c) 11<15'
                        WHEN Nro_pdos_cliente BETWEEN 16 AND 20 THEN 'd) 16<20'
                        WHEN Nro_pdos_cliente BETWEEN 21 AND 25 THEN 'e) 21<25'
                        WHEN Nro_pdos_cliente BETWEEN 26 AND 30 THEN 'f) 26<30'
                        WHEN Nro_pdos_cliente >=31 THEN 'g) 31=<' END AS num_pdos,
              COUNT (*) AS Nro_clientes
            FROM
```

```
            num
           GROUP BY
            CASE WHEN Nro_pdos_cliente BETWEEN 0 AND 5 THEN 'a) 5=>'
                      WHEN Nro_pdos_cliente BETWEEN 6 AND 10 THEN 'b) 6<10'
                      WHEN Nro_pdos_cliente BETWEEN 11 AND 15 THEN 'c) 11<15'
                      WHEN Nro_pdos_cliente BETWEEN 16 AND 20 THEN 'd) 16<20'
                      WHEN Nro_pdos_cliente BETWEEN 21 AND 25 THEN 'e) 21<25'
                      WHEN Nro_pdos_cliente BETWEEN 26 AND 30 THEN 'f) 26<30'
                      WHEN Nro_pdos_cliente >=31 THEN 'g) 31=<' END
),
 acum AS -- ACUMULADO DE CLIENTES
(
           SELECT
            *,
            SUM (Nro_clientes) OVER (ORDER BY num_pdos) AS Nro_clientes_acum
           FROM
            pdos
),
 ultvalor AS -- OBTENCION DEL ULTIMO VALOR DEL PORCENTAJE
(
           SELECT
            *,
            LAST_VALUE (Nro_clientes_acum) OVER (ORDER BY Nro_clientes_acum
                                                 ROWS BETWEEN CURRENT ROW AND UNBOUNDED FOLLOWING) AS ult_valor
           FROM
            acum
),
 pct AS -- CALCULO DEL PORCENTAJE SOBRE EL NUMERO DE PEDIDOS REALIZADOS
(
           SELECT
            *,
            CAST (Nro_clientes_acum / CAST (ult_valor AS decimal (8,2)) AS decimal (8,2)) * 100 AS pct
           FROM
            ultvalor
) -- CONSULTA FINAL Y RESULTADO
SELECT
 num_pdos,
 pct
FROM
 pct
ORDER BY
 pct ASC
```

RESULTADO

Results | Messages

	num_pdos	pct
1	a) 5=>	29.00
2	b) 6<10	69.00
3	c) 11<15	90.00
4	d) 16<20	97.00
5	f) 26<30	99.00
6	g) 31=<	100.00

CONCLUSION DEL ANÁLISIS

Después de realizar este análisis de frecuencias acumuladas del número de pedidos sobre los clientes, podemos afirmar que el **90 % de los clientes han realizado un MAXIMO DE 15 PEDIDOS.**