Empezando en el editor de consultas Power Query

Trabajando con el editor de consultas (Power Query), ventajas y consejos relacionados

Jose Ignacio González Gómez
Departamento de Economía Contabilidad y Finanzas - Universidad de La Laguna
www.jggomez.eu

v.2.5

INDICE

1 Introducción, ventajas de trabajar en consultas

Un aspecto relevante a considerar de carácter técnico conceptual es que es mejor hacer todas las transformaciones posibles en el editor de consultas (Power Query) que con formulas DAX en el modelo de datos por varias razones clave:

Separación de responsabilidades

- **Power Query** se encarga de la **preparación de datos**: limpieza, filtrado, transformación, combinación, etc.
- **DAX** se usa para **modelado y análisis**: cálculos, medidas, KPIs, etc.

Mantener esta separación hace que el modelo sea más claro y fácil de mantener

Rendimiento

- Power Query transforma los datos **antes de cargarlos al modelo**, lo que reduce el tamaño del modelo y mejora el rendimiento.
- DAX trabaja **sobre los datos ya cargados**, lo que puede ser más costoso en tiempo de cálculo, especialmente con grandes volúmenes de datos.

Reutilización de datos

- Las transformaciones en Power Query pueden ser reutilizadas en múltiples tablas o informes.
- Las fórmulas DAX suelen ser específicas para un visual o medida.

Limpieza y calidad de datos

- Power Query tiene herramientas más robustas para detectar y corregir errores, tipos de datos inconsistentes, valores nulos, etc.
- DAX no está diseñado para limpiar datos, sino para analizarlos.

Facilidad de uso

- Power Query tiene una interfaz más intuitiva, con pasos visuales y editor de fórmulas M.
- DAX requiere más conocimiento técnico y puede ser más difícil de depurar.

2 Consejos relacionados con el uso del editor de consultas

Realiza todas las transformaciones posibles en Power Query

Como mencionamos antes, es mejor transformar los datos antes de cargarlos al modelo para mejorar el rendimiento y la claridad.

Usa nombres descriptivos para las consultas y pasos

Evita nombres genéricos como Consulta1 o Paso1. Usa nombres como VentasFiltradas2025 o AgrupadoPorRegión para que tú (y otros) entiendan fácilmente qué hace cada parte

Limpia los datos al principio

Haz limpieza básica al inicio: Elimina columnas innecesarias, corrige tipos de datos, quita filas vacías o errores. Esto reduce el tamaño del modelo y evita errores posteriores

Divide transformaciones complejas en pasos simples

En lugar de hacer todo en un solo paso, divide las transformaciones en pasos intermedios. Esto facilita la depuración y mejora la legibilidad.

Usa referencias en lugar de duplicados cuando sea posible

Como vimos antes, **referenciar** una consulta permite reutilizarla sin duplicar lógica, lo que mejora la eficiencia y el mantenimiento.

Evita columnas calculadas en DAX si puedes hacerlo en Power Query

Las columnas calculadas en DAX consumen más memoria. Si puedes crear esa columna en Power Query, hazlo allí.

Reduce el número de filas y columnas antes de cargar

Filtra los datos innecesarios antes de cargarlos al modelo. Esto mejora el rendimiento y reduce el tamaño del archivo.

Documenta tus pasos

Puedes agregar comentarios en los pasos del Editor Avanzado o usar nombres de pasos descriptivos para explicar qué hace cada transformación

Usa funciones personalizadas cuando repitas lógica

Si haces la misma transformación en varias consultas, considera crear una función personalizada para evitar duplicación de código.

Revisa el orden de los pasos

Algunos pasos (como cambiar tipos de datos) es mejor hacerlos después de filtrar o transformar, para evitar errores o cálculos innecesarios

3 Ejemplo relacionado

Vamos a ver un **ejemplo concreto de buenas prácticas en Power Query**, usando un escenario típico: una tabla de ventas con columnas como las siguientes.

Fecha	Producto	Cantidad	PrecioUnitario	Región
2025-01-01	A	10	5.00	Norte
2025-01-02	B	20	7.50	Sur
...	...	...	...	

Transformaciones aplicadas en Power Query siguiendo buenas prácticas:

1. Renombrar la consulta

- De Tabla1 a Ventas2025.

2. Eliminar columnas innecesarias

- Si hay columnas como Notas o IDInterno que no se usan, se eliminan.

3. Cambiar tipos de datos

- Fecha → tipo fecha.
- Cantidad y PrecioUnitario → tipo número decimal.

4. Agregar columna calculada

- Crear una columna ImporteTotal `= [Cantidad] * [PrecioUnitario]`

5. Filtrar datos

- Filtrar para incluir solo ventas del año 2025
  `= Table.SelectRows(Ventas2025, each Date.Year([Fecha]) = 2025)`

6. Agrupar datos

- Agrupar por Región y sumar ImporteTotal
  `= Table.Group(Ventas2025, {"Región"}, {{"TotalVentas", each List.Sum([ImporteTotal]), type number}})`

7. Usar referencia

- Crear una nueva consulta VentasPorProducto como referencia de Ventas2025, para agrupar por Producto.

8. Renombrar pasos

- En lugar de Paso1, Paso2, usar nombres como FiltrarAño2025, AgregarImporteTotal, etc.

Resultado: modelo limpio, eficiente y fácil de mantener

- Las transformaciones están bien organizadas.
- Se evita duplicar lógica.
- El modelo es más rápido y más claro.