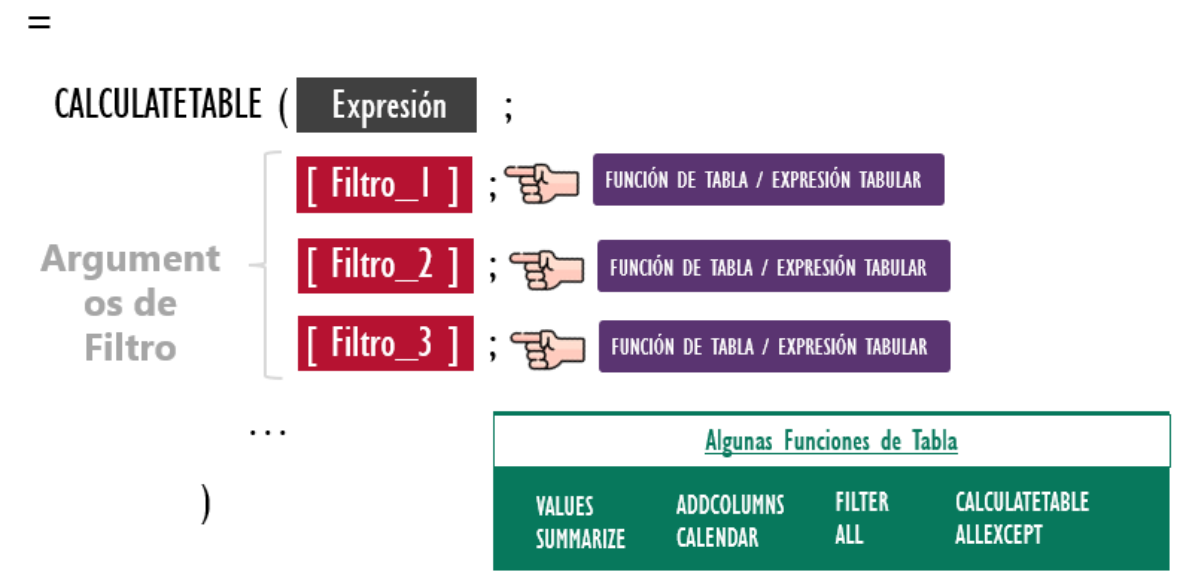
Funciones de tabla : CALCULATETABLE, CROSSJOIN, EXCEPT, UNION y INTERSECT

1. CALCULATETABLE

CALCULATETABLE realiza las mismas operaciones que CALCULATE, con la única diferencia en su resultado. CALCULATETABLE devuelve una tabla, mientras que CALCULATE devuelve un solo valor como un número entero o una cadena.

Como ejemplo, si uno necesita producir una tabla que contenga solo productos rojos, entonces CALCULATETABLE es la función a usar:

```
CALCULATETABLE (
    'Product',
    'Product'[Color] = "Red"
)
```



=CALCULATETABLE (<Tabla> ; <Filtro 1> ; <Filtro 2>...)

PedidosNormal= CALCULATETABLE (Pedidos ; Pedidos[Tipo Compra]="Normal")

2 1

País	ID	Tipo Compra	Ingresos
Colombia	B01	Normal	30
Argentina	B02	Normal	320
Colombia	CB01	Devolución	110
Perú	L01	Normal	250
Colombia	CC01	Normal	110
Perú	L03	Devolución	250

País	ID	Tipo Compra	Ingresos
Colombia	B01	Normal	30
Argentina	B02	Normal	320
Perú	L01	Normal	250
Colombia	CC01	Normal	110

CONTEXTO DE FILTRO (ORIGINAL)

CONTEXTO DE FILTRO (Copia)

Tipo Compra
Normal

CALCULATETABLE desempeña el mismo trabajo que CALCULATE, la única diferencia radica en su resultado, porque la primera retorna una Tabla y la segunda un Valor Escalar. Quiere decir que CALCULATETABLE evalúa una tabla o una expresión que retorna una tabla, en un contexto modificado por los filtros dados.

PedidosNorCol= CALCULATETABLE (Pedidos ; Pedidos[Tipo Compra]="Normal"; Pedidos[País]="Colombia")

País	ID	Tipo Compra	Ingresos
Colombia	B01	Normal	30
Argentina	B02	Normal	320
Colombia	CB01	Devolución	110
Perú	L01	Normal	250
Colombia	CC01	Normal	110
Perú	L03	Devolución	250

País	Sku	Tipo Compra	Ingresos
Colombia	B01	Normal	30
Colombia	CC01	Normal	110

Diferencia entre CALCULATETABLE y FILTER

Una pregunta común es cuál es la diferencia entre CALCULATETABLE y FILTER. De hecho, la expresión anterior también se puede escribir con FILTER:

```
FILTER (
    'Product',
    'Product'[Color] = "Red"
)
```

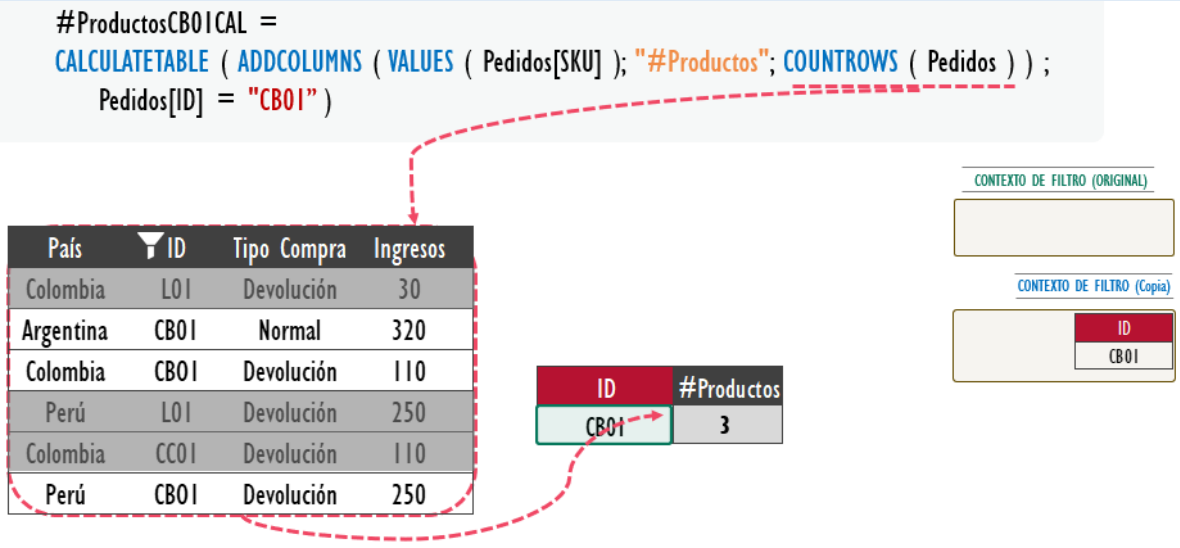
Aunque la única diferencia parece ser el nombre de la función, en realidad la semántica de estas dos funciones es muy diferente. **CALCULATETABLE opera cambiando primero el contexto del filtro y luego evaluando la expresión. FILTER, por otro lado, itera el resultado de su primer argumento, recuperando las filas que satisfacen la condición. En otras palabras, FILTER no cambia el contexto del filtro**.

Puede apreciar la diferencia revisando el siguiente ejemplo:

```
Red Products CALCULATETABLE =
CALCULATETABLE (
    ADDCOLUMNS (
        VALUES ( 'Product'[Color] ),
        "Num of Products", COUNTROWS ( 'Product' )
    ),
    'Product'[Color] = "Red"
)
```

Color	Num of Products
Red	99

Al usar CALCULATETABLE, el contexto de filtro donde se evalúan tanto ADDCOLUMNS como COUNTROWS está filtrando productos rojos. Por lo tanto, el resultado es solo una fila que contiene rojo como color y 99 como número de productos. En otras palabras, COUNTROWS solo contó los productos rojos, **sin requerir una transición de contexto** desde la fila generada por la función VALUES.

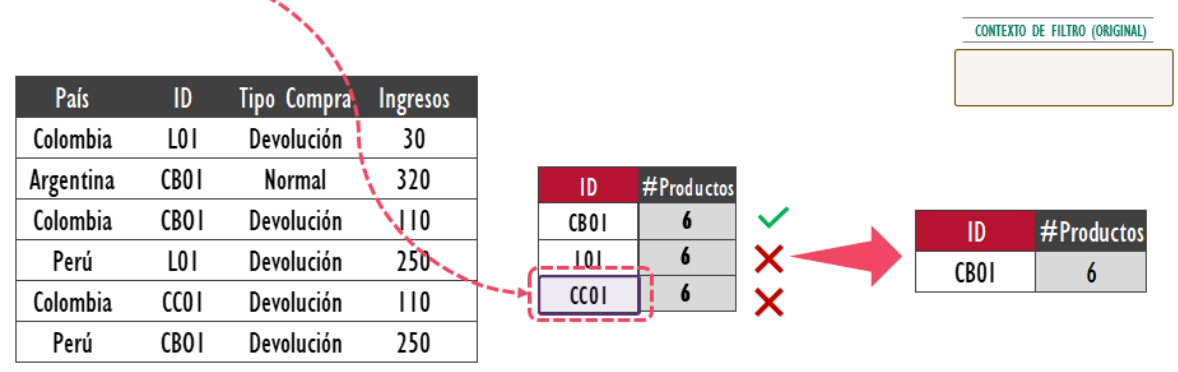


Si se reemplaza CALCULATETABLE con FILTER, el resultado es diferente. Mira la siguiente tabla:

```
Red Products FILTER =
FILTER (
    ADDCOLUMNS (
        VALUES ( 'Product'[Color] ),
        "Num of Products", COUNTROWS ( 'Product' )
    ),
    'Product'[Color] = "Red"
)
```

Color	Num of Products
Red	2517

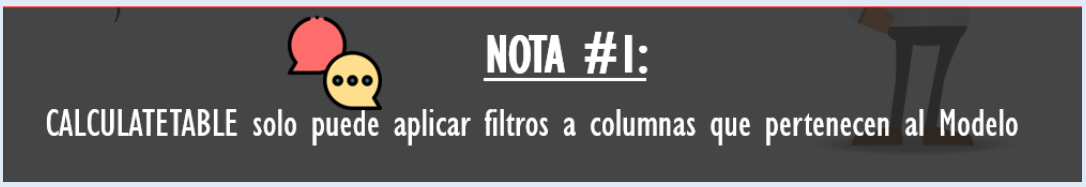
```
#ProductosCB01FIL =
FILTER ( ADDCOLUMNS ( VALUES ( Pedidos[ID] ); "#Productos"; COUNTROWS ( Pedidos ) ) ;
    Pedidos[ID] = "CB01" )
```




```
Red Products FILTER =
FILTER (
    ADDCOLUMNS (
        VALUES ( 'Product'[Color] ),
        "Num of Products", CALCULATE(COUNTROWS ( 'Product' ))
    ),
    'Product'[Color] = "Red"
)
```

Color	Num of Products
Red	99

Por otro lado, **CALCULATETABLE solo puede aplicar filtros a columnas que pertenecen al modelo de datos**. Si solo se necesita al Customer cuyo monto de ventas es mayor que un millón, entonces CALCULATETABLE no es la opción correcta porque el monto de ventas es una medida. Por lo tanto, **CALCULATETABLE no puede aplicar un filtro en una medida, mientras que FILTER sí**.




```
Large Customers =
FILTER (
    Customer,
    [Sales Amount] > 1000000
)
```

Y tampoco puede llamar columnas de tablas temporales:

```
CALCULATETABLE =
CALCULATETABLE(
    ADDCOLUMNS(
        VALUES(Pedidos[SKU]),
        "@Ingresos totales por SKU",
        CALCULATE(COUNTROWS(Pedidos))
    ),
    [@Ingresos totales por SKU]
)
```

No podemos llamar a una columna que no existe.

```
SKUIngresosM30 =
FILTER(
    ADDCOLUMNS(
        VALUES(Pedidos[SKU]),
        "@Ingresos Totales por SKU",
        CALCULATE(SUM(Pedidos[Ingresos]))
    ),
    [@Ingresos Totales por SKU]>30000)
```

Con FILTER si podemos hacerlo.

Ejemplo 1:

Crear una tabla con los datos de los productos: CB01, L01 y CC01

```
PedidosCB01L01CC01 =
CALCULATETABLE(
    Pedidos,
    Pedidos[SKU] in {"CB01","L01","CC01"}
    )
```

```
PedidosCB01L01CC01 =
CALCULATETABLE(
    Pedidos,
    SKUProductos[SKU] in {"CB01","L01","CC01"}
    )
```

Ejemplo 2:

Crear una tabla con los datos de los productos cuyas transacciones de ingresos se encuentran entre 100 y 300 dólares

```
Pedidos100y300 =
    CALCULATETABLE(
        Pedidos,
        Pedidos[Ingresos] >= 100 &&
        Pedidos[Ingresos] <= 300
    )
```

Ejemplo 3:

SKU solo para el país “Colombia”.

```
SKUsColombia =
    CALCULATETABLE(
        VALUES(Pedidos[SKU]),
        Pedidos[País] = "Colombia" -- FILTER +ALL
    )
```

Ejemplo 4: CALCULATETABLE / ADDCOLUMNS

¿Crear dos Tablas Calculadas, que me indiquen el número de transacciones relacionadas a CB01 una con la función CALCULATETABLE y otra con FILTER?

```
SkuTransacciones =
 CALCULATETABLE(
        ADDCOLUMNS(
            VALUES(Pedidos[SKU]),
            "NumeroFilas",
            COUNTROWS(Pedidos)
            ),
         Pedidos[SKU]    = "CB01"
 )
```

SKU	NumeroFilas
CB01	4756

2. CROSSJOIN

=CROSSJOIN (Tabla1; Tabla2 ;...)

CROSSJOIN retorna una tabla que es resultado de realizar el producto cartesiano de las tablas proporcionadas en sus parámetros, es decir, que crea todas las posibilidades entre cruces de registros en las distintas tablas

TablaCROSSJOIN= CROSSJOIN(Descuentos ; CategoríaDeProducto)

Descuento	Cat.Descuento
0	Non
0,25	Black Friday
0,35	Special Day

ID	Categoría
B01	Blue Ray
B02	Blue Ray
CB01	Combo

Descuento	Cat.Descuento	ID	Categoría
0	Non	B01	Blue Ray
0,25	Black Friday	B01	Blue Ray
0,35	Special Day	B01	Blue Ray
0	Non	B02	Blue Ray
0,25	Black Friday	B02	Blue Ray
0,35	Special Day	B02	Blue Ray
0	Non	CB01	Combo
0,25	Black Friday	CB01	Combo
0,35	Special Day	CB01	Combo

3. EXCEPT

Desde el punto de vista del linaje, EXCEPT retiene el linaje de datos de la primera tabla, como fue el caso de INTERSECT.

=EXCEPT (<Tablalzquierda>; <Tabla2/ExpresiónTabla>)

EXCEPT retorna todas las filas que están en la tabla asignada como primer argumento, pero que NO están en la segunda tabla

TablaEXCEPT= EXCEPT(Pedidos1 ; Pedidos2)

País	ID	Tipo Compra	Ingresos
Colombia	B01	Normal	30
Colombia	CB01	Devolución	110
Argentina	B02	Normal	320
Colombia	CC01	Normal	110

País	ID	Tipo Compra	Ingresos
Argentina	B02	Normal	320
Perú	L01	Normal	250
Perú	L03	Devolución	250

País	ID	Tipo Compra	Ingresos
Colombia	B01	Normal	30
Colombia	CB01	Devolución	110
Colombia	CC01	Normal	110

Ejemplo 1:

Clientes que compraron un producto en 2007 pero no en 2008

```
CustomersBuyingIn2007butNotIn2008 =
VAR Customers2007 =
    CALCULATETABLE (
        SUMMARIZE ( Sales, Customer[Customer Code] ),
        'Date'[Calendar Year] = "CY 2007"
    )
VAR Customers2008 =
    CALCULATETABLE (
        SUMMARIZE ( Sales, Customer[Customer Code] ),
        'Date'[Calendar Year] = "CY 2008"
    )
VAR Result =
    EXCEPT ( Customers2007, Customers2008 )
RETURN Result
```

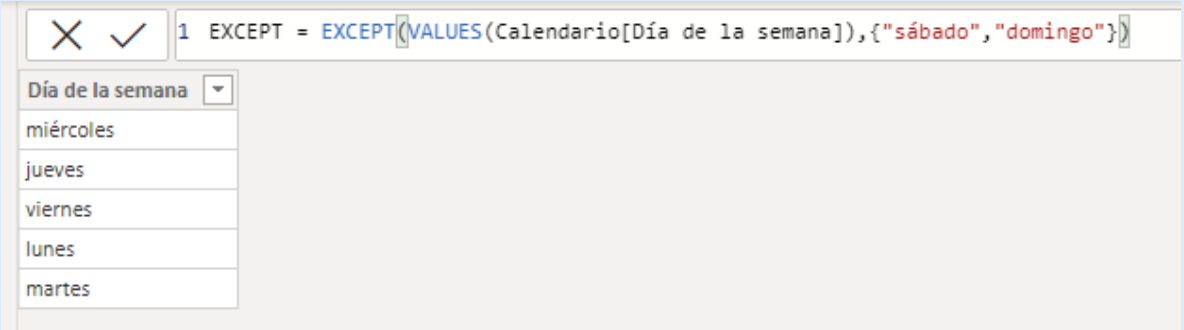
Ejemplo 2:

El número de Customers que no compraron nada el año pasado pero sí compraron algo este año

```
SalesOfNewCustomers :=
VAR CurrentCustomers =
    VALUES ( Sales[CustomerKey] )
VAR CustomersLastYear =
    CALCULATETABLE (
        VALUES ( Sales[CustomerKey] ),
        DATESINPERIOD ( 'Date'[Date], MIN ( 'Date'[Date] ) - 1, -1, YEAR
)
    )
VAR CustomersNotInLastYear =
    EXCEPT ( CurrentCustomers, CustomersLastYear )
VAR Result =
    CALCULATE ( [Sales Amount], CustomersNotInLastYear )
RETURN Result
```

Ejemplo 3:

Los valores del segundo parámetro son removidos de los valores del primer parámetro:



Otra forma de realizarlo:

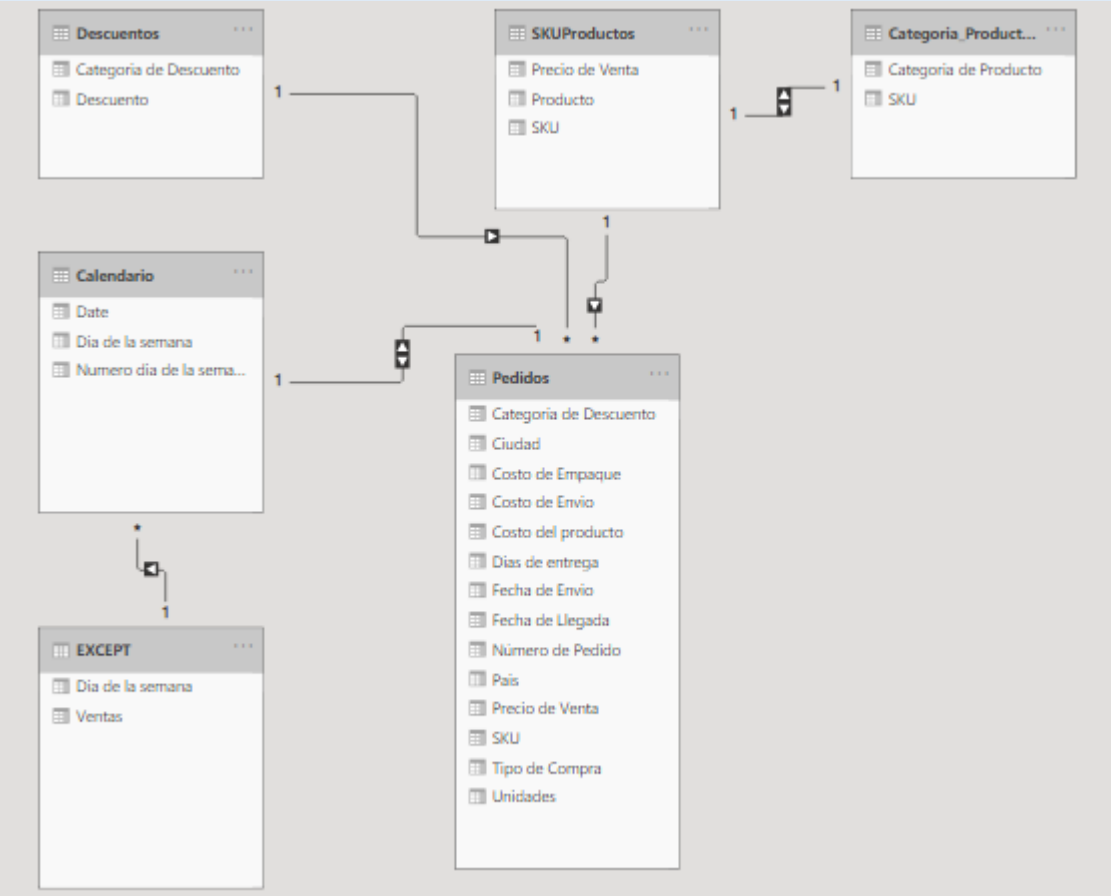
```
VAR Dias = VALUES(Calendario[Día de la semana])
VAR FindeSemana = {"sábado","domingo"}
VAR DiasLaborales = EXCEPT(Dias,FindeSemana)
RETURN
  DiasLaborales
```

```
EXCEPT =
VAR Dias = VALUES(Calendario[Día de la semana])
VAR FindeSemana = {"sábado","domingo"}
VAR DiasLaborales = EXCEPT(Dias,FindeSemana)
RETURN
DiasLaborales
```

Día de la semana
miércoles
jueves
viernes
lunes
martes

Ejemplo 4:

Vamos a extraer información de la Tabla Pedidos. La Tabla EXCEPT la relacione con la Tabla Calendario entre columnas “Dia de la semana”.



```
VAR Dias = VALUES(Calendario[Día de la semana])
VAR FindeSemana = {"sábado","domingo"}
VAR DiasLaborales = EXCEPT(Dias,FindeSemana)
RETURN
    ADDCOLUMNS(DiasLaborales,"Ventas",[Ingresos])
```

```
EXCEPT =
VAR Dias = VALUES(Calendario[Dia de la semana])
VAR FindeSemana = {"sábado","domingo"}
VAR DiasLaborales = EXCEPT(Dias,FindeSemana)
RETURN
ADDCOLUMNS(DiasLaborales,"Ventas",[Ingresos])
```

Dia de la semana	Ventas
miércoles	$4.038.229
jueves	$4.445.181
viernes	$4.816.004
lunes	$4.438.708
martes	$4.110.544

Ejemplo 5:

```
IngresosSinFestivos =
VAR DiasSinFestivos =
    EXCEPT(
        VALUES(Calendario[Fecha]),
        VALUES(FechasFestivas[Fecha de Festividad])
    )
VAR Resultado =
    CALCULATE(
        [Ingresos Tot],
        DiasSinFestivos
    )
RETURN
    Resultado
```

Ingresos Tot	IngresosSinFestivos
3.320.219,45	3.147.493,22

Fecha de Festividad	Descripción
viernes, 1 de enero de 1999	Año Nuevo
lunes, 11 de enero de 1999	Día de los Reyes Magos
lunes, 22 de marzo de 1999	Día de San José
domingo, 28 de marzo de 1999	Domingo de Ramos
jueves, 1 de abril de 1999	Jueves Santo
viernes, 2 de abril de 1999	Viernes Santo
domingo, 4 de abril de 1999	Domingo de Resurrección
sábado, 1 de mayo de 1999	Día del Trabajo
lunes, 17 de mayo de 1999	Día de la Ascensión
lunes, 7 de junio de 1999	Corpus Christi
lunes, 14 de junio de 1999	Sagrado Corazón
lunes, 5 de julio de 1999	San Pedro y San Pablo
martes, 20 de julio de 1999	Día de la Independencia
sábado, 7 de agosto de 1999	Batalla de Boyacá
lunes, 16 de agosto de 1999	La asunción de la Virgen
lunes, 18 de octubre de 1999	Día de la Raza
lunes, 1 de noviembre de 1999	Todos los Santos
lunes, 15 de noviembre de 1999	Independencia de Cartagena
miércoles, 8 de diciembre de 1999	Día de la Inmaculada Concepción
sábado, 25 de diciembre de 1999	Día de Navidad
sábado, 1 de enero de 2000	Año Nuevo
lunes, 10 de enero de 2000	Día de los Reyes Magos
lunes, 20 de marzo de 2000	Día de San José
domingo, 16 de abril de 2000	Domingo de Ramos
jueves, 20 de abril de 2000	Jueves Santo
viernes, 21 de abril de 2000	Viernes Santo

Ejemplo 6:

Ingresos sin días festivos ni sábados y domingos.

```
IngresosSinFestivosNiFinesdesemana =
VAR FechaFinesSemana =
    FILTER(VALUES(Calendario[Fecha]),WEEKDAY(Calendario[Fecha],2)<6)
VAR DiasSinFestivos = EXCEPT(
    VALUES(Calendario[Fecha]),
    VALUES(FechasFestivas[Fecha de Festividad]))
Return
CALCULATE([Ingresos Tot],DiasSinFestivos,FechaFinesSemana)
```

Ingresos Tot	IngresosSinFestivos	IngresosSinFestivosNiFinesdesemana
3.320.219,45	3.147.493,22	2.215.817,00

4. UNION

UNION es una función de conjunto que realiza la unión de dos tablas. La capacidad de combinar diferentes tablas en una sola tabla puede ser importante en ciertas circunstancias. Se utiliza principalmente en tablas calculadas, y con mucha menos frecuencia en medidas.

UNION no elimina los duplicados antes de devolver un resultado.

Para unir dos tablas deben tener la misma estructura y cada columna debe tener el mismo tipo de dato. Es similar a ANEXAR (Append).

Ejemplo 1:

```
AllCountryRegions =
UNION (
    ALL ( Customer[CountryRegion] ),
    ALL ( Store[CountryRegion] )
)
```

CountryRegion
Australia
Australia
United States
United States
Canada
Canada
Germany
Germany
United Kingdom
United Kingdom

Ejemplo 2:

Aprovecharemos la función DISTINCT para devolver los valores únicos:

```
DistinctCountryRegions =
VAR CountryRegions =
    UNION (
        ALL ( Customer[CountryRegion] ),
        ALL ( Store[CountryRegion] )
    )
VAR UniqueCountryRegions =
    DISTINCT ( CountryRegions )
RETURN UniqueCountryRegions
```

CountryRegion
Australia
United States
Canada
Germany
United Kingdom
France
the Netherlands
Greece
Switzerland

UNION mantiene el linaje de datos de las tablas de entrada si el linaje de ambas tablas es el mismo. **En la fórmula anterior, el resultado de DISTINCT no tiene linaje porque la primera tabla contiene *Customer[CountryRegion]* y la segunda tabla contiene *Store[CountryRegion].* Debido a que el linaje de datos de las tablas de entrada es diferente, el resultado tiene un nuevo linaje que no corresponde a ninguna de las columnas existentes**.

Ejemplo 3:

```
= UNION(Tabla1, Tabla2, Tabla3,...)
```

Cantidades

Mes	Producto	Cantidad
miércoles, 01 de enero de 2020	A	500
jueves, 02 de enero de 2020	B	600
viernes, 03 de enero de 2020	C	700
sábado, 04 de enero de 2020	D	400
domingo, 05 de enero de 2020	A	520
lunes, 06 de enero de 2020	B	575
martes, 07 de enero de 2020	C	710
miércoles, 08 de enero de 2020	D	412
jueves, 09 de enero de 2020	A	495
viernes, 10 de enero de 2020	B	635
sábado, 11 de enero de 2020	C	695
domingo, 12 de enero de 2020	D	460
lunes, 13 de enero de 2020	A	505
martes, 14 de enero de 2020	B	620
miércoles, 15 de enero de 2020	C	735
jueves, 16 de enero de 2020	D	480

Precios

Mes	Producto	Precio
viernes, 17 de enero de 2020	A	125
sábado, 18 de enero de 2020	B	400
domingo, 19 de enero de 2020	C	250
lunes, 20 de enero de 2020	D	1200
martes, 21 de enero de 2020	A	150
miércoles, 22 de enero de 2020	B	380
jueves, 23 de enero de 2020	C	225
viernes, 24 de enero de 2020	D	1100
sábado, 25 de enero de 2020	A	175
domingo, 26 de enero de 2020	B	300
lunes, 27 de enero de 2020	C	275
martes, 28 de enero de 2020	D	400
miércoles, 29 de enero de 2020	A	95
jueves, 30 de enero de 2020	B	425
viernes, 31 de enero de 2020	C	300
sábado, 01 de febrero de 2020	D	200

```
1 UNION = UNION(Cantidades,Precios)
```

Mes	Producto	Cantidad
01-01-2020 0:00:00	A	500
02-01-2020 0:00:00	B	600
03-01-2020 0:00:00	C	700
04-01-2020 0:00:00	D	400
05-01-2020 0:00:00	A	520
06-01-2020 0:00:00	B	575
07-01-2020 0:00:00	C	710
08-01-2020 0:00:00	D	412
09-01-2020 0:00:00	A	495
10-01-2020 0:00:00	B	635
11-01-2020 0:00:00	C	695
12-01-2020 0:00:00	D	460
13-01-2020 0:00:00	A	505
14-01-2020 0:00:00	B	620
15-01-2020 0:00:00	C	735
16-01-2020 0:00:00	D	480
17-01-2020 0:00:00	A	125
18-01-2020 0:00:00	B	400
19-01-2020 0:00:00	C	250
20-01-2020 0:00:00	D	1200
21-01-2020 0:00:00	A	150
22-01-2020 0:00:00	B	380
23-01-2020 0:00:00	C	225

Tabla: UNION (32 filas)

Ejemplo 4: UNION \ ROW

Podemos crear una tabla:

```
UNION_ROW =
UNION(
        ROW("Nombre","Alfonso","Apellido","Perez","Edad",35),
        ROW("Nombre","Gonzalo","Apellido","Lino","Edad",37),
        ROW("Nombre","Tomas","Apellido","Torrico","Edad",70))
```

Nombre	Apellido	Edad
Alfonso	Perez	35
Gonzalo	Lino	37
Tomas	Torrico	70

Ejemplo 5: UNION \ SELECTCOLUMNS

En caso de que necesitemos unir dos tablas con distinto orden de columnas y encabezado distinto, usaremos SELECTCOLUMNS.

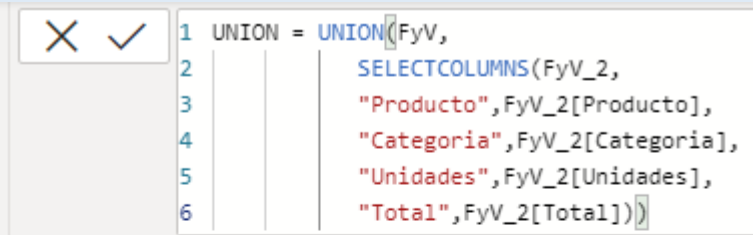
Tabla FyV

Producto	Categoria	Cantidad	Total
Manzana	Frutas	20	800
Espinaca	Verduras	15	1000
Zapallo	Verduras	10	2000
Platano	Frutas	5	600

Tabla FyV_2

Producto	Categoria	Total	Unidades
Manzana	Frutas	800	20
Espinaca	Verduras	1000	15
Zapallo	Verduras	2000	10
Platano	Frutas	600	5

```
UNION = UNION(FyV,
            SELECTCOLUMNS(FyV_2,
            "Producto",FyV_2[Producto],
            "Categoria",FyV_2[Categoria],
            "Unidades",FyV_2[Unidades],
            "Total",FyV_2[Total]))
```



Producto	Categoria	Cantidad	Total
Manzana	Frutas	20	800
Espinaca	Verduras	15	1000
Zapallo	Verduras	10	2000
Platano	Frutas	5	600
Manzana	Frutas	20	800
Espinaca	Verduras	15	1000
Zapallo	Verduras	10	2000
Platano	Frutas	5	600

5. INTERSECT

INTERSECT es una función establecida muy similar a UNION. Sin embargo, en lugar de agregar una tabla a otra, devuelve la intersección de las dos tablas, es decir, solo las filas que aparecen en ambas tablas.

Desde el punto de vista del linaje, INTERSECT conserva el linaje de datos de la primera tabla. Si se construye una tabla, donde las tablas tienen el mismo linaje de datos, se mantendrá el mismo linaje. Si se construye una tabla con diferentes linajes de datos, solo se mantiene el linaje de la primera.

=INTERSECT (<TablaIzquierda>; <TablaDerecha>)

INTERSECT retorna todas las filas que son comunes a ambas tablas que han sido definidas como parámetros de la función

TablaINTERSECT= INTERSECT(Pedidos1 ; Pedidos2)

País	ID	Tipo Compra	Ingresos
Colombia	B01	Normal	30
Colombia	CB01	Devolución	110
Argentina	B02	Normal	320
Colombia	CC01	Normal	110

País	ID	Tipo Compra	Ingresos
Argentina	B02	Normal	320
Perú	L01	Normal	250
Perú	L03	Devolución	250

País	ID	Tipo Compra	Ingresos
Argentina	B02	Normal	320

Ejemplo 1:

Recuperar los Customers que compraron tanto en 2007 como en 2008.

```
CustomersBuyingInTwoYears =
VAR Customers2007 =
    CALCULATETABLE(
        SUMMARIZE(
            Sales,
            Customer[Customer Code]
        ),
        'Date'[Calendar Year] = "CY 2007"
    )
VAR Customers2008 =
    CALCULATETABLE(
        SUMMARIZE(
            Sales,
            Customer[Customer Code]
        ),
        'Date'[Calendar Year] = "CY 2008"
    )
VAR Result =
    INTERSECT(Customers2007,Customers2008)
RETURN
    Result
```

Customer Code
11009
11010
11011
11015
11040
11048
11050
11052
11059
11074
11076
11077
11081
11082
11083
11092
11093
11094
11095
11130
11131
11132
11133
11140
11173
11174

Tabla: CustomersBuyingInTwoYears (1,732 filas)

Si se construye una tabla, a partir de tablas que tienen el mismo linaje de datos, se mantendrá el mismo linaje. Por tanto, podemos crear una columna calculada y traer sus ingresos totales.

```
VAR Customers2007 =
    CALCULATETABLE(
        SUMMARIZE(
            Sales,
            Customer[Customer Code]
        ),
        'Date'[Calendar Year] = "CY 2007"
    )
VAR Customers2008 =
    CALCULATETABLE(
        SUMMARIZE(
            Sales,
            Customer[Customer Code]
        ),
        'Date'[Calendar Year] = "CY 2008"
    )
VAR Interseccion =
    INTERSECT(Customers2007,Customers2008)

VAR Resultado =
    ADDCOLUMNS(
        Interseccion,
        "Ventas",
        [Sales Amount]
    )
RETURN
    Resultado
```

Customer Code	Ventas
11009	$1.527
11010	$1.448
11011	$4.255
11015	$272
11040	$1.028
11048	$533
11050	$593
11052	$1.463
11059	$1.988
11074	$1.561
11076	$1.294
11077	$1.294
11081	$103
11082	$1.177
11083	$80
11092	$5.131
11093	$14
11094	$14
11095	$50
11130	$1.034
11131	$2.151
11132	$1.121
11133	$1.028
11140	$395
11173	$31
11174	$224

Tabla: CustomersBuyingInTwoYears (1,732 filas)

Customer Code	Sales Amount
11005	$1.460
11006	$2.718
11007	$1.359
11008	$1.359
11009	$1.527
11010	$1.448
11011	$4.255
11015	$272
11016	$79
11017	$37
11018	$48
11025	$48
Total	**$30.591.344**

Filas

Customer Code

Columnas

Agregar campos de datos a...

Valores

Sales Amount

Obtener detalles

Entre varios informes

Desactivar

Mantener todos los filtros

Ejemplo 2:

Si se construye una tabla con diferentes linajes de datos, solo se mantiene el linaje de la primera tabla. En este último ejemplo, el linaje es el de *Store[CountryRegion]*. En consecuencia, una expresión más compleja como la siguiente devuelve las ventas filtradas por *Store[CountryRegion],* no las de *Customer[CountryRegion]:*

```
SalesStoresInCustomersCountries =
VAR CountriesWithStoresAndCustomers =
    INTERSECT (
        ALL ( Store[CountryRegion] ),
        ALL ( Customer[CountryRegion] )
    )
VAR Result =
    ADDCOLUMNS (
        CountriesWithStoresAndCustomers,
        "StoresSales", [Sales Amount]
    )
RETURN Result
```

La columna *StoresSales* contiene las ventas relacionadas con el país de la *Store*:

CountryRegion	StoresSales
United States	11,195,063.06
United Kingdom	
France	
Australia	
Canada	
Germany	8,670,581.01
Turkmenistan	
Thailand	
China	10,725,699.91
Kyrgyzstan	