

# Inteligencia de tiempo – Funciones para acumulados desde el principio del periodo

## Funciones para Acumulados Desde el Principio del Periodo

DATESYTD, DATESQTD y DATESMTD – TOTALYTD, TOTALQTD y TOTALMTD

Se indica la medida a acumular

Año	YTD Ingresos
2000	
enero	
febrero	
marzo	
abril	
mayo	

2000	QTD Ingresos
julio	
agosto	
septiembre	

Año	MTD Ingresos
2000	
octubre	
1	
2	
3	
4	
5	
6	
7	

YTD Ingresos =  
TOTALYTD (  
[Ingresos Tot] ;  
Calendario[Fecha]  
)

Internamente se  
Convierte en:

La clave primaria de la tabla de calendario

YTD Ingresos =  
CALCULATE (  
[Ingresos Tot];  
DATESYTD (Calendario[Fecha])  
)  
Tabla Desde la Primera Fecha del Año Hasta La Fecha Actual Visible

Fecha
1/01/2000
...
1/04/2000
...
30/04/2000

QTD Ingresos =  
TOTALQTD (  
[Ingresos Tot] ;  
Calendario[Fecha]  
)

QTD Ingresos =  
CALCULATE (  
[Ingresos Tot];  
DATESQTD (Calendario[Fecha])  
)  
Tabla Desde la Primera Fecha del Año Hasta La Fecha Actual Visible

Fecha
1/07/2000
...
1/08/2000
...
30/09/2000

MTD Ingresos =  
TOTALMTD (  
[Ingresos Tot] ;  
Calendario[Fecha]  
)

MTD Ingresos =  
CALCULATE (  
[Ingresos Tot];  
DATESMTD (Calendario[Fecha])  
)  
Tabla Desde la Primera Fecha del Año Hasta La Fecha Actual Visible

Fecha
1/10/2000
...
3/10/2000
...
7/10/2000

### TOTALYTD, TOTALQTD y TOTALMTD

DAX también viene equipado con unas funciones que simplifican las sintaxis de las funciones de inteligencia de tiempo para crear acumulados: TOTALYTD, TOTALQTD y TOTALMTD.

Sin embargo, no son más que sólo *funciones de máscara* para la combinación: CALCULATE y la respectiva función DATES\_TD.

Por lo anterior estas funciones esconden el CALCULATE, por lo que su utilización no es buena practica, dado al mantener visible CALCULATE, nos podemos percatar con mayor facilidad de operaciones como la transición de contextos.



## 1. DATESYTD

### Devuelve una Tabla

La función DATESYTD retorna una columna que contiene las fechas individuales (día a día) desde el principio del año hasta la última fecha en el contexto de filtro actual.

Estos son los dos argumentos de la función DATESYTD:

- **Dates:** Columna de fechas de la tabla de *Calendario*
- **Year End Date:** Si nuestro final de año no es el 31 de diciembre, en este argumento podemos indicar cuál es mediante una cadena de caracteres.

= **CALCULATE**([Ingresos Totales], **DATESYTD**(Calendario[Fechas]; "06-30"))

## 2. DATESYTD – Primitivo

Forma 1

Inteligencia de Tiempo

DATESYTD =  
DATESYTD ( Calendario[Fecha] )

En Primitivo

DATESYTDenPrimitivo =  
VAR FechaActual =  
MAX ( Calendario[Fecha] )  
RETURN  
FILTER (  
ALL ( Calendario[Fecha] );  
Calendario[Fecha] <= FechaActual  
&& YEAR ( Calendario[Fecha] ) = YEAR ( FechaActual )  
)

Forma 2

```
Sales Amount YTD :=  
VAR LastVisibleDate = MAX ( 'Date'[Date] )  
VAR CurrentYear = YEAR ( LastVisibleDate )  
VAR SetOfDatesYtd =  
    FILTER (  
        ALL ( 'Date' ),  
        AND (  
            'Date'[Date] <= LastVisibleDate,  
            YEAR ( 'Date'[Date] ) = CurrentYear  
        )  
    )  
VAR Result =  
    CALCULATE (  
        SUMX ( Sales, Sales[Net Price] * Sales[Quantity] ),  
        SetOfDatesYtd  
    )  
RETURN  
    Result
```

Por ejemplo, si el contexto de filtro es el mes de Marzo, **LastVisibleDate** sería 31/03/2007 y **CurrentYear** sería 2007. Dado que ALL es toda la tabla CALENDARIO e ignora los filtros, la tabla temporal FILTER comenzaría desde el 01/01/2007 hasta el 31/12/2007. Por medio de la función AND filtramos, donde Date[Date] corresponde a la columna de fechas desde el 01/01/2007 hasta el 31/12/2007, donde le decimos que las fechas que contenga deben ser menores o iguales a **LastVisibleDate**, ósea, 31/03/2007, y además que el año de Date[Date] sea igual a **CurrentYear**, que sería el año 2007. Por ende, la tabla FILTER comienza desde el 01/01/2007 hasta el 31/03/2007. Donde finalmente se realizará una suma de las ventas entre estas fechas.

# ¡CUIDADO!

## LA EXPRESIÓN NO ES EQUIVALENTE EN TODOS LOS CASOS

Esto especialmente importante en presencia de un Contexto de Fila

Contar Días =

-- En una C.C

COUNTROWS (

DATESYTD ( Calendario[Fecha] )

Contar Filtro =

-- En una C.C

COUNTROWS (

DATESYTD En Primitivo



1 Contar Días =

2 COUNTROWS( DATESYTD( Calendario[Fecha] ) )

Fecha	Año	Trimestre	Mes Número	Mes Nombre	DS Número	DS Nombre	Día	Semana	Contar Días
1/01/1999	1999	Trim. 1	1	enero	5	viernes	1	1	1
2/01/1999	1999	Trim. 1	1	enero	6	sábado	2	1	2
3/01/1999	1999	Trim. 1	1	enero	7	domingo	3	2	3
4/01/1999	1999	Trim. 1	1	enero	1	lunes	4	2	4
5/01/1999	1999	Trim. 1	1	enero	2	martes	5	2	5
6/01/1999	1999	Trim. 1	1	enero	3	miércoles	6	2	6
7/01/1999	1999	Trim. 1	1	enero	4	jueves	7	2	7
8/01/1999	1999	Trim. 1	1	enero	5	viernes	8	2	8
9/01/1999	1999	Trim. 1	1	enero	6	sábado	9	2	9
10/01/1999	1999	Trim. 1	1	enero	7	domingo	10	3	10
11/01/1999	1999	Trim. 1	1	enero	1	lunes	11	3	11
12/01/1999	1999	Trim. 1	1	enero	2	martes	12	3	12
13/01/1999	1999	Trim. 1	1	enero	3	miércoles	13	3	13
14/01/1999	1999	Trim. 1	1	enero	4	jueves	14	3	14
15/01/1999	1999	Trim. 1	1	enero	5	viernes	15	3	15



1 Contar Días =

2 COUNTROWS( DATESYTD( Calendario[Fecha] ) )

Fecha	Año	Trimestre	Mes Número	Mes Nombre	DS Número	DS Nombre	Día	Semana	Contar Días
28/12/1999	1999	Trim. 4	12	diciembre	2	martes	28	53	362
29/12/1999	1999	Trim. 4	12	diciembre	3	miércoles	29	53	363
30/12/1999	1999	Trim. 4	12	diciembre	4	jueves	30	53	364
31/12/1999	1999	Trim. 4	12	diciembre	5	viernes	31	53	365
1/01/2000	2000	Trim. 1	1	enero	6	sábado	1	1	1
2/01/2000	2000	Trim. 1	1	enero	7	domingo	2	2	2
3/01/2000	2000	Trim. 1	1	enero	1	lunes	3	2	3
4/01/2000	2000	Trim. 1	1	enero	2	martes	4	2	4
5/01/2000	2000	Trim. 1	1	enero	3	miércoles	5	2	5
6/01/2000	2000	Trim. 1	1	enero	4	jueves	6	2	6
7/01/2000	2000	Trim. 1	1	enero	5	viernes	7	2	7
8/01/2000	2000	Trim. 1	1	enero	6	sábado	8	2	8
9/01/2000	2000	Trim. 1	1	enero	7	domingo	9	3	9
10/01/2000	2000	Trim. 1	1	enero	1	lunes	10	3	10
11/01/2000	2000	Trim. 1	1	enero	2	martes	11	3	11

La fórmula anterior tiene un CALCULATETABLE implícito (que no se ve):

SINTAX SUGAR

```
1 Contar Días =  
2 COUNTROWS( DATESYTD( Calendario[Fecha] ) )
```

=

```
1 Contar Días =  
2 COUNTROWS(  
3     DATESYTD(  
4         CALCULATETABLE( DISTINCT( Calendario[Fecha] ) )  
5     )  
6 )
```

## Transición de Contextos con Otras Funciones

*Funciones que Ejecutan Transición de Contextos al tener CALCULATE implícitos o Internos*



Existen **funciones que ejecutan transición de contextos**, claro cuando son llamadas en un contexto de fila, dado que tienen un **CALCULATE** implícito que no vemos pero que está allí por syntax sugar.

La gran mayoría de estas funciones están cobijadas en las funciones de inteligencia de tiempo, dado que cuando se utiliza **una referencia a una columna de fechas en realidad corresponde una syntax sugar que involucra CALCULATETABLE**.

Concretamente:

CLOSINGBALANCEMONTH, CLOSINGBALANCEQUARTER, CLOSINGBALANCEYEAR, DATEADD, DATESMTD, DATESQTD, DATESYTD, ENDOFMONTH, ENDOFMONTH, ENDOFYEAR, FIRSTDATE, FIRSTNONBLANK, FIRSTNONBLANKVALUE, LASTDATE, LASTNONBLANK, LASTNONBLANKVALUE, NEXTDAY, NEXTMONTH, NEXTQUARTER, NEXTYEAR, OPENINGBALANCEMONTH, OPENINGBALANCEQUARTER, OPENINGBALANCEYEAR, PARALLELPERIOD, PREVIOUSDAY, PREVIOUSMONTH, PREVIOUSQUARTER, PREVIOUSYEAR, SAMEPERIODLASTYEAR, STARTOFMONTH, STARTOFQUARTER, STARTOFYEAR.

Pero existe un par adicional que tiene un CALCULATETABLE pero no son de inteligencia de tiempo:

DETAILROWS y RELATEDTABLE

Finalmente, hay unas funciones que tienen un CALCULATE implícito, ellas son:

TOTALMTD, TOTALQTD y TOTALYTD.



Si utilizamos el **DATESYTD Primitivo** en una **Columna calculada** nos retorna “365” para todos los registros. **¿Y esto por qué?**

Fecha	Año	Contar Filtro
01-01-1999	1999	365
02-01-1999	1999	365
...	...	365
31-12-1999	1999	365
...	...	365
01-01-2017	2017	365
02-01-2017	2017	365
...	...	365
31-12-2017	2017	365

```

1 Contar Filtro =
2 VAR FechaActual =
3     MAX( Calendario[Fecha] )
4 VAR Filtro =
5     FILTER(
6         ALL( Calendario[Fecha] ),
7         Calendario[Fecha] <= FechaActual
8         && YEAR(Calendario[Fecha]) = YEAR(FechaActual)
9     )
10 VAR Conteo =
11     COUNTROWS( Filtro )
12 RETURN
13     Conteo

```

1. Como sabemos las funciones de agregación **IGNORAN EL CONTEXTO DE FILA**, la función **MAX** almacenará la fecha máxima, que en este caso

Se debe comprender que **MAX** es el problema, dado que ignora el contexto de fila. Para solucionar este problema tendremos que utilizar en el siguiente ejercicio **TRANSICIÓN DE CONTEXTO**.

Por tanto, **COUNTROWS** devolverá el número de filas, que son en total 365.

Fecha	Año	Trimestre	Mes Número	Mes Nombre	DS Número	DS Nombre	Día	Semana	Contar Días	Contar Filtro
1/01/1999	1999	Trim. 1	1	enero	5	viernes	1	1	1	365
2/01/1999	1999	Trim. 1	1	enero	6	sábado	2	1	2	365
3/01/1999	1999	Trim. 1	1	enero	7	domingo	3	2	3	365
4/01/1999	1999	Trim. 1	1	enero	1	lunes	4	2	4	365
5/01/1999	1999	Trim. 1	1	enero	2	martes	5	2	5	365
6/01/1999	1999	Trim. 1	1	enero	3	miércoles	6	2	6	365
7/01/1999	1999	Trim. 1	1	enero	4	jueves	7	2	7	365

**Calendario[Fecha] <= FechaActual**

Fecha	Año	Contar Filtro
01-01-1999	1999	365
02-01-1999	1999	365
...	...	365
31-12-1999	1999	365
...	...	365
01-01-2017	2017	365
02-01-2017	2017	365
...	...	365
31-12-2017	2017	365

2. Recordar que aunque dentro del **ALL** se utilice la columna de la fecha se creará de manera automática e implícita un **ALL (Calendario)**. Sabiendo esto, se iterará la tabla y para cada valor de la columna **Fecha** donde la fecha sea menor o igual a 31-12-2017 se cumplirá la condición, es decir, será **VERDADERO (TRUE)**. Por tanto, **TODAS** las fechas son menores al 31-12-2017, todas las fechas cumplen esta condición.

**&& YEAR(Calendario[Fecha]) = YEAR(FechaActual)**

3. Esta condición no se cumplirá para todas las fechas, sino, solo para las fechas que pertenecen al año 2017. Es decir 365 fechas.

Fecha	Año	Contar Filtro
01-01-1999	1999	365
02-01-1999	1999	365
...	...	365
31-12-1999	1999	365
...	...	365
01-01-2017	2017	365
02-01-2017	2017	365
...	...	365
31-12-2017	2017	365

Encerramos todo dentro de un CALCULATETABLE generando TRANSICIÓN DE CONTEXTO

1	Contar Filtro =
2	COUNTROWS(
3	CALCULATETABLE(
4	VAR FechaActual =
5	MAX( Calendario[Fecha] )
6	VAR Filtro =
7	FILTER(
8	ALL( Calendario[Fecha] ),
9	Calendario[Fecha] <= FechaActual
10	&& YEAR(Calendario[Fecha]) = YEAR(FechaActual)
11	)
12	RETURN
13	Filtro
14	)
15	)

Fecha	Año	Trimestre	Mes Número	Mes Nombre	DS Número	DS Nombre	Día	Semana	Contar Días	Contar Filtro
1/01/1999	1999	Trim. 1	1	enero	5	viernes	1	1	1	1
2/01/1999	1999	Trim. 1	1	enero	6	sábado	2	1	2	2
3/01/1999	1999	Trim. 1	1	enero	7	domingo	3	2	3	3
4/01/1999	1999	Trim. 1	1	enero	1	lunes	4	2	4	4
5/01/1999	1999	Trim. 1	1	enero	2	martes	5	2	5	5
6/01/1999	1999	Trim. 1	1	enero	3	miércoles	6	2	6	6

Calendario (7,305 filas) Columna: Contar Filtro (366 valores distintos)

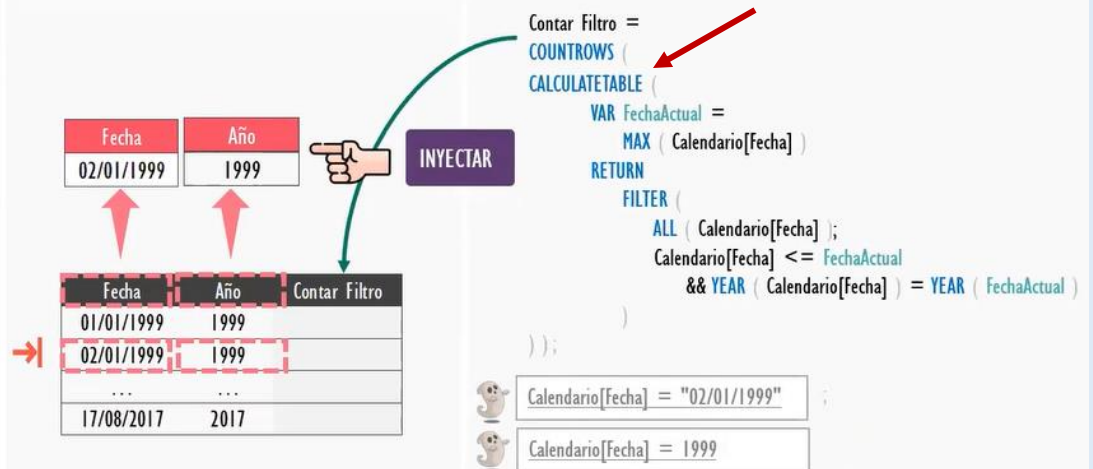
CALCULATE/CALCULATETABLE en Contextos de Fila: Activa la Transición de Contexto.

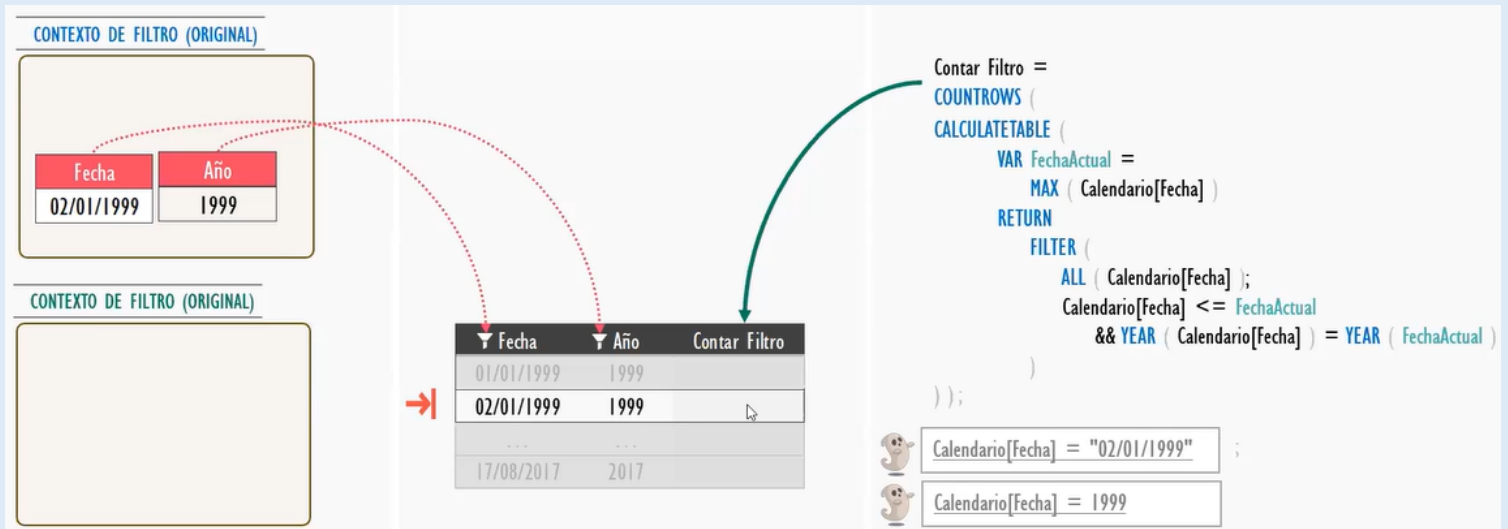
Operación 1: Inválida cualquier Contexto de Fila existente.

Operación 2: Añade como Argumentos de Filtros Todas las Columnas con los Valores de la Iteración Actual.

CONTEXTO DE FILTRO (ORIGINAL)

CONTEXTO DE FILTRO (ORIGINAL)





1. Dijimos que al encerrar todo con CALCULATETABLE se generaba transición de contexto. Ahora si nos centramos en la fila de la fecha 02-01-1999, la variable **FechaActual** almacenará la fecha de la fila, es decir, 02-01-1999.

Fecha	Año	Contar Filtro
01-01-1999	1999	1
02-01-1999	1999	2
...	...	3
31-12-1999	1999	4
...	...	5
01-01-2017	2017	6
02-01-2017	2017	7
...	...	8
31-12-2017	2017	9

```

1 Contar Filtro =
2 COUNTROWS(
3     CALCULATETABLE(
4         VAR FechaActual =
5             MAX( Calendario[Fecha] )
6         VAR Filtro =
7             FILTER(
8                 ALL( Calendario[Fecha] ),
9                 Calendario[Fecha] <= FechaActual
10                && YEAR(Calendario[Fecha]) = YEAR(FechaActual)
11            )
12         RETURN
13             Filtro
14     )
15 )

```

Fecha	Año	Trimestre	Mes Número	Mes Nombre	DS Número	DS Nombre	Día	Semana	Contar Días	Contar Filtro
1/01/1999	1999	Trim. 1	1	enero	5	viernes	1	1	1	1
2/01/1999	1999	Trim. 1	1	enero	6	sábado	2	1	2	2
3/01/1999	1999	Trim. 1	1	enero	7	domingo	3	2	3	3
4/01/1999	1999	Trim. 1	1	enero	1	lunes	4	2	4	4
5/01/1999	1999	Trim. 1	1	enero	2	martes	5	2	5	5
6/01/1999	1999	Trim. 1	1	enero	3	miércoles	6	2	6	6

Calendario (7,305 filas) Columna: Contar Filtro (366 valores distintos)

```

Calendario[Fecha] <= FechaActual
&& YEAR(Calendario[Fecha]) = YEAR(FechaActual)

```

2. Para estas condiciones la fecha 01-01-1999 y la misma 02-01-1999 son las que cumplen las condiciones. Es decir, 2 fechas. Todas las demás fechas no cumplen las condiciones.

3. Finalmente, COUNTROWS contará y encontrará solo 2 filas y ese será el valor que retorne.



### 3. DATESQTD

#### Devuelve una Tabla

La primera variante de acumulado se presenta si queremos desde el principio del trimestre hasta la fecha actual. Como se observa en la matriz ya no es el agregado desde el primer mes del año, en su lugar es desde el primer mes del trimestre.

En suma, la función DATESQTD hace la tarea de la lista de fechas desde el principio del trimestre hasta la fecha actual para ser utilizada en CALCULATE.

**QTD Ingresos** = **CALCULATE**([Ingresos Totales],**DATESQTD**(Calendario[Fechas]))

Año	Ingresos Totales	YTD Ingresos	QTD Ingresos
1999	8.685,83	8.685,83	8.685,83
Tr. 4	8.685,83	8.685,83	8.685,83
Diciembre	8.685,83	8.685,83	8.685,83
2000	180.680,24	180.680,24	43.557,78
Tr. 1	49.429,52	49.429,52	49.429,52
Enero	13.508,96	13.508,96	13.508,96
Febrero	17.592,04	31.101,00	31.101,00
Marzo	18.328,51	49.429,52	49.429,52
Tr. 2	45.700,64	95.130,16	45.700,64
Abril	14.726,52	64.156,03	14.726,52
Mayo	13.507,17	77.663,20	28.233,68
Junio	17.466,96	95.130,16	45.700,64
Tr. 3	41.992,30	137.122,46	41.992,30
Julio	15.729,11	110.859,27	15.729,11
Agosto	15.056,15	125.915,42	30.785,26
Septiembre	11.207,04	137.122,46	41.992,30
Tr. 4	43.557,78	180.680,24	43.557,78
Octubre	14.678,72	151.801,18	14.678,72
Noviembre	14.862,94	166.664,13	29.541,67
Diciembre	14.016,11	180.680,24	43.557,78
Total	3.320.885,54		

La función QTD solo tiene un argumento y nada más, no existen argumentos opcionales ni de ninguna otra clase.

#### 4. DATESMTD

##### Devuelve una Tabla

La segunda variante de acumulado se presenta si queremos desde el principio del mes hasta la fecha actual. Para el MTD es obligatorio desplegar toda la jerarquía si queremos observar los efectos de la medida MTD Ingresos ¿Por qué? porque como el acumulado es desde el principio del mes su efecto se ve a nivel de días.

**QTD Ingresos** = **CALCULATE**([Ingresos Totales], **DATESMTD**(Calendario[Fechas]))

Año	MTD Ingresos	QTD Ingresos	YTD Ingresos	Ingresos Totales
lunes, 31 de enero de 2000	13.500,50	13.500,50	13.500,50	311,09
Febrero	17.592,04	31.101,00	31.101,00	17.592,04
martes, 1 de febrero de 2000	1.443,84	14.952,80	14.952,80	1.443,84
miércoles, 2 de febrero de 2000	1.571,49	15.080,45	15.080,45	127,65
jueves, 3 de febrero de 2000	2.030,14	15.539,10	15.539,10	458,65
viernes, 4 de febrero de 2000	3.459,74	16.968,70	16.968,70	1.429,60
sábado, 5 de febrero de 2000	4.166,11	17.675,07	17.675,07	706,37
domingo, 6 de febrero de 2000	4.847,55	18.356,51	18.356,51	681,45
lunes, 7 de febrero de 2000	5.144,98	18.653,94	18.653,94	297,43
martes, 8 de febrero de 2000	6.140,98	19.649,94	19.649,94	996,00
miércoles, 9 de febrero de 2000	6.140,98	19.649,94	19.649,94	
jueves, 10 de febrero de 2000	7.226,43	20.735,39	20.735,39	1.085,45
viernes, 11 de febrero de 2000	7.697,21	21.206,17	21.206,17	470,78
sábado, 12 de febrero de 2000	8.853,41	22.362,37	22.362,37	1.156,20
domingo, 13 de febrero de 2000	9.096,25	22.605,21	22.605,21	242,84
lunes, 14 de febrero de 2000	9.096,25	22.605,21	22.605,21	
martes, 15 de febrero de 2000	10.810,75	24.319,71	24.319,71	1.714,49
miércoles, 16 de febrero de 2000	11.056,39	24.565,35	24.565,35	245,64
Total				3.320.885,54

Figura 19. 22 - Matriz con la Medida MTD Ingresos

## 5. TOTALYTD

La función también puede tomar un tercer argumento opcional que represente la última fecha de un año. La ausencia de esta fecha significa que el 31 de diciembre es la última fecha del año.

Corresponde a un atajo:

Acumulado YTD

**CALCULATE** ( [Ingresos Totales] ; **DATESYTD** ( Calendario[Fechas] ) )  
→ **TOTALYTD** ( [Ingresos Totales] ; Calendario[Fechas] )

La función TOTALYTD tiene un tercer argumento opcional para señalar el final del año, ejemplo con el año fiscal

**CALCULATE** ( [Ingresos Totales] ; **DATESYTD** ( Calendario[Fechas]; "06-30" ) )  
→ **TOTALYTD** ( [Ingresos Totales] ; Calendario[Fechas] ; "06-30" )

## 6. TOTALQTD

Corresponde a un atajo:

Acumulado QTD

**CALCULATE** ( [Ingresos Totales] ; **DATESQTD** ( Calendario[Fechas] ) )  
→ **TOTALQTD** ( [Ingresos Totales] ; Calendario[Fechas] )

## 7. TOTALMTD


Corresponde a un atajo:

Acumulado MTD

**CALCULATE** ( [Ingresos Totales] ; **DATESMTD** ( Calendario[Fechas] ) )  
→ **TOTALMTD** ( [Ingresos Totales] ; Calendario[Fechas] )

## 8. HTD (Acumulado Histórico a la Fecha)

Es posible que queramos el acumulado, pero desde la primera fecha de la tabla de datos, TOTALYTD no funciona porque toma la fecha desde el principio del año, nosotros queremos el acumulado histórico.



Año	Ingresos Totales
1999	8.685,83
2000	180.680,24
2001	166.860,36
2002	184.876,83
2003	177.061,95
2004	168.059,37
2005	187.906,44
2006	198.797,54
2007	170.712,79
2008	175.936,37
2009	198.222,01
2010	240.873,82
2011	216.088,04
<b>Total</b>	<b>3.320.885,54</b>

Figura 19. 23 – Objetivo del Acumulado Histórico

HTD Ingresos

```
= CALCULATE([Ingresos], FILTER(ALL(Calendario[Fechas]),
    Calendario[Fechas] <=
    MAX(Calendario[Fechas])) )
```



Año	Ingresos Totales	HTD Ingresos
1999	8.685,83	8.685,83
2000	180.680,24	189.366,07
2001	166.860,36	356.226,43
2002	184.876,83	541.103,26
2003	177.061,95	718.165,21
2004	168.059,37	886.224,58
2005	187.906,44	1.074.131,01
2006	198.797,54	1.272.928,55
2007	170.712,79	1.443.641,34
2008	175.936,37	1.619.577,71
2009	198.222,01	1.817.799,71
2010	240.873,82	2.058.673,53
2011	216.088,04	2.274.761,57
2012	229.644,13	2.504.405,71
2013	232.463,65	2.736.869,36
2014	256.692,19	2.993.561,55
2015	327.323,99	3.320.885,54
2016		3.320.885,54
2017		3.320.885,54
2018		3.320.885,54
<b>Total</b>	<b>3.320.885,54</b>	<b>3.320.885,54</b>

Figura 19. 24 – Medida HTD Ingresos

## 9. Acumulado hasta la fecha (DAX Primitivo – Estático)

### Ejemplo 1

**La Pregunta Sería:**

Cómo hacemos para crear una expresión tabular que genere una lista de fechas desde el 1 de Enero del 2000 hasta el 30 de abril del 2000.

Ejecutar Expresión DAX

## Acumulado Hasta la Fecha (YTD)

En DAX Plano: Primitivo - Estático.



### CONTEXTO DE FILTRO (COPIA)

Fecha	Año	Mes
1/01/2000	2000	Enero
...	...	...
1/04/2000	2000	Abril
...	...	...
30/04/2000	2000	Abril

### CONTEXTO DE FILTRO (ORIGINAL)

Año	Mes
2000	Abril

Año	YTD Ingresos Abril
2000	52
enero	
febrero	
marzo	
abril	
mayo	

1 Identificar Filtros

Fecha	Año	Mes
1/01/1999	1999	Diciembre
...	...	...
1/01/2000	2000	Enero
...	...	...
1/04/2000	2000	Abril
...	...	...
30/04/2000	2000	Abril
...	...	...
1/01/2001	2001	Enero
...	...	...
31/12/2015	2015	Diciembre

Pais	SKU	F. Envio	F. Llegada	Ingreso
...	...	...	...	...
Colombia	CB01	1/01/2000	7/01/2000	5
Colombia	CB01	1/01/2000	8/01/2000	4
Colombia	L07	1/02/2000	1/03/2000	4
Perú	L02	1/02/2000	2/03/1900	4
Colombia	L07	1/03/2000	29/04/2000	7
Colombia	CB01	1/03/2000	29/04/2000	8
Colombia	CB01	2/04/2000	30/04/2000	9
Colombia	L01	2/04/2000	30/04/2000	1
Argentina	CB01	3/04/2000	1/05/2000	7
Argentina	L07	3/04/2000	1/05/2000	3
				<b>Σ=52</b>

```
YTD Ingresos Abril =  
CALCULATE (  
    [Ingresos Tot];  
    FILTER (  
        ALL ( Calendario ) ;  
        AND (  
            Calendario[Fecha] >= DATE ( 2000; 1; 1 );  
            Calendario[Fecha] <= DATE ( 2000; 4; 30 )  
        )  
    )  
)
```

3 Ejecutar Expresión DAX

Año	Ingresos Tot	IngresoEneroAbril2000
octubre		64.156,03
noviembre		64.156,03
diciembre	8.019,74	64.156,03
<b>2000</b>	<b>180.680,24</b>	<b>64.156,03</b>
enero	13.508,96	64.156,03
febrero	17.592,04	64.156,03
marzo	18.328,51	64.156,03
abril	14.726,52	64.156,03
mayo	13.507,17	64.156,03
junio	17.466,96	64.156,03
julio	15.729,11	64.156,03
agosto	15.056,15	64.156,03
septiembre	11.207,04	64.156,03
octubre	14.678,72	64.156,03
noviembre	14.862,94	64.156,03
<b>Total</b>	<b>3.320.219,45</b>	<b>64.156,03</b>



## Ejemplo 2

Acumulado hasta la fecha para el mismo año.

### Acumulado Hasta la Fecha (YTD)

En DAX Plano: Primitivo - Dinámico.



**CONTEXTO DE FILTRO (COPIA)**

Año	Mes
2000	Abril

**CONTEXTO DE FILTRO (ORIGINAL)**

Año	Mes
2000	Abril

Año: 2000 YTD Ingresos Abril

**Tabla de Calendario**

Fecha	Año	Mes
1/01/1999	1999	Diciembre
...	...	...
1/01/2000	2000	Enero
...	...	...
1/04/2000	2000	Abril
...	...	...
30/04/2000	2000	Abril
...	...	...
1/01/2001	2001	Enero
...	...	...
31/12/2015	2015	Diciembre

**Tabla de Ingresos**

Pais	SKU	F. Envio	F. Llegada	Ingreso
Colombia	CB01	1/01/2000	7/01/2000	5
Colombia	CB01	1/01/2000	8/01/2000	4
Colombia	L07	1/02/2000	1/03/2000	4
Perú	L02	1/02/2000	2/03/1900	4
Colombia	L07	1/03/2000	29/04/2000	7
Colombia	CB01	1/03/2000	29/04/2000	8
Colombia	CB01	2/04/2000	30/04/2000	9
Colombia	L01	2/04/2000	30/04/2000	1
Argentina	CB01	3/04/2000	1/05/2000	7
Argentina	L07	3/04/2000	1/05/2000	3

**DAX Expression:**

```

YTD Ingresos Abril =
VAR UltimaFechaVisible = MAX(Calendario[Fecha])
VAR AnActual = YEAR(UltimaFechaVisible)
RETURN
CALCULATE (
    [Ingresos Tot];
    FILTER (
        ALL (Calendario)
        AND (
            Calendario[Fecha] <= UltimaFechaVisible;
            YEAR(Calendario[Fecha]) = AnActual
        )
    )
    
```

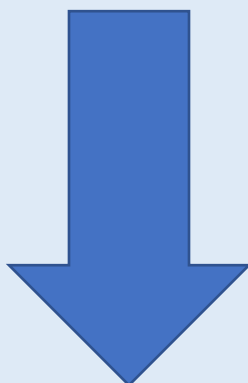
1 Identificar Filtros

2 Aplicar Filtros

3 Ejecutar Expresión DAX

```

1 YTDPriIngresos =
2 VAR UltimaFechaVisible = MAX(Calendario[Fecha])
3 VAR AUltimaFechaVisible = YEAR(UltimaFechaVisible)
4 Return
5 CALCULATE(
6     [Ingresos Tot];
7     FILTER(
8         ALL(Calendario);
9         AND( Calendario[Fecha]<=UltimaFechaVisible;
10         YEAR(Calendario[Fecha]) = AUltimaFechaVisible)
11     )
12 )
    
```



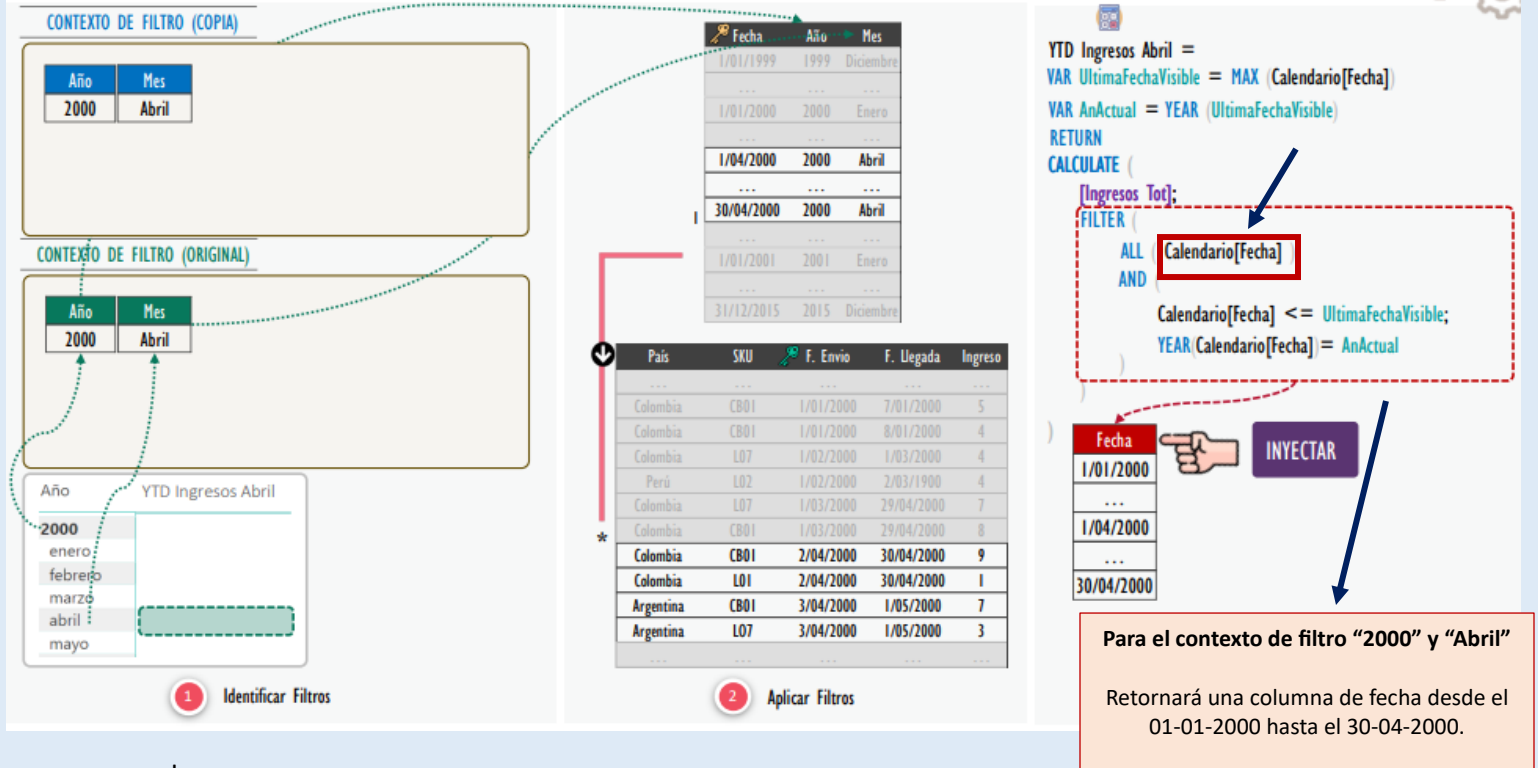
Año	Ingresos Tot	YTDPriIngresos
1999	8.019,74	8.019,74
diciembre	8.019,74	8.019,74
2000	180.680,24	180.680,24
enero	13.508,96	13.508,96
febrero	17.592,04	31.101,00
marzo	18.328,51	49.429,52
abril	14.726,52	64.156,03
mayo	13.507,17	77.663,20
junio	17.466,96	95.130,16
julio	15.729,11	110.859,27
agosto	15.056,15	125.915,42
septiembre	11.207,04	137.122,46
octubre	14.678,72	151.801,18
noviembre	14.862,94	166.664,13
diciembre	14.016,11	180.680,24
<b>Total</b>	<b>3.320.219,45</b>	

Qué pasa si utilizásemos **ALL ( Calendario [ Fecha ] )**

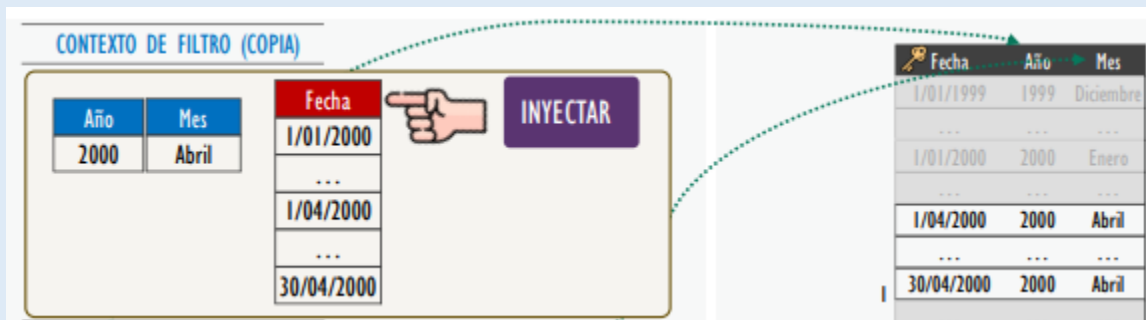
## Consideraciones de Inteligencia de Tiempo

Código DAX interno en relaciones con columnas de fecha.

6



Luego:



Tendríamos 3 contextos de filtro. Al ser 3 columnas distintas TODOS ELLOS SE APLICAN AL MODELO DE DATOS, donde por lógica, debiese retornarnos valores solo para el rango de fechas comprendido entre 01-04-2000 y 30-04-2000, dado que el filtro "Mes: Abril" reduce el rango de fechas del filtro "Fechas".

### NOTA #1:

Esta expresión DAX no funcionaría, dado que no devolvería el acumulado sino únicamente los ingresos para el mes de abril en el año 2000.

Ejecutar Expresión DAX



## REPASA TU FICHA TÉCNICA DE CALCULATE

Cuando un filtro inyectado por un argumento de filtro en CALCULATE afecta a la misma columna en el contexto de filtro original, entonces, el filtro de CALCULATE se impone sobre el filtro nativo del contexto de filtro, sobrescribiéndolo. La sobrescritura también ocurre si el filtro viene en una tabla acompañado de otras columnas. Los filtros finales, se aplican como si se tratará de una conjunción lógica. De lo anterior, si los filtros son de diferentes de columnas, entonces, todo ellos se aplican al modelo de datos.

Argentina

L07

3/04/2000

1/05/2000

3

101011  
100111  
1010

### NOTA #2:

No obstante, la expresión DAX si funciona, puesto que: DAX automáticamente añade un argumento de filtro adicional: ALL ( TablaDeBusqueda )

Ejecutar Expresión DAX

Paradójicamente, DE IGUAL MANERA obtenemos un resultado correcto. La razón es un comportamiento especial de DAX cuando la relación entre dos tablas se basa en una columna de fecha, como sucede con la relación con Date en el modelo de demostración que estamos usando aquí. **Siempre que se aplica un filtro a una columna de tipo Date o DateTime que se utiliza en una relación entre dos tablas**, DAX agrega automáticamente ALL a toda la tabla de fecha como un argumento de filtro adicional para CALCULATE.

1 Por orden de presidencia: identificado como argumento de filtro en la función CALCULATE.

2 La columna afecta debe estar previamente con algunos de los dos formatos.

El motor DAX automáticamente añade como argumento de filtro un ALL que remueve todos los filtros de la tabla de búsqueda donde reside una columna de fecha que ha sido afectada por un filtro del contexto de filtro, siempre y cuando tenga asignado el formato de tipo fecha o fecha & hora, y además sirva como clave primaria en una relación uno a muchos.

3 La columna afecta está en el lado de los unos y es la PK.

En otras palabras, el código anterior debería leerse de esta manera:

**YTD Ingresos Abril =**  
**VAR** UltimaFechaVisible = **MAX** (Calendario[Fecha])  
**VAR** AnActual = **YEAR** (UltimaFechaVisible)  
**RETURN**  
**CALCULATE** (  
 [Ingresos Tot];  
**FILTER** (  
**ALL** (Calendario[Fecha])  
**AND** (  
 Calendario[Fecha] <= UltimaFechaVisible;  
**YEAR**(Calendario[Fecha]) = AnActual  
 )  
 )  
**ALL** (Calendario) -- Insertado por Motor DAX  
 )

En este caso el tercer argumento de **CALCULATE** deja de actuar como un argumento de filtro que inyecta filtros al contexto de filtro, ahora con la presencia de **ALL** actúa como modificador de **CALCULATE**, es decir, cambia su comportamiento cambiar la estructura del contexto de filtro.

## Consideraciones de Inteligencia de Tiempo

Código DAX interno en relaciones con columnas de fecha.

**CONTEXTO DE FILTRO (COPIA)**

Fecha
1/01/2000
...
1/04/2000
...
30/04/2000

**CONTEXTO DE FILTRO (ORIGINAL)**

Año	Mes
2000	Abril

Año: 2000  
 YTD Ingresos Abril: 52

Fecha	Año	Mes
1/01/1999	1999	Diciembre
...	...	...
1/01/2000	2000	Enero
...	...	...
1/04/2000	2000	Abril
...	...	...
30/04/2000	2000	Abril
...	...	...
1/01/2001	2001	Enero
...	...	...
31/12/2015	2015	Diciembre

País	SKU	F. Envío	F. Llegada	Ingreso
Colombia	CB01	1/01/2000	7/01/2000	5
Colombia	CB01	1/01/2000	8/01/2000	4
Colombia	L07	1/02/2000	1/03/2000	4
Perú	L02	1/02/2000	2/03/1900	4
Colombia	L07	1/03/2000	29/04/2000	7
Colombia	CB01	1/03/2000	29/04/2000	8
Colombia	CB01	2/04/2000	30/04/2000	9
Colombia	L01	2/04/2000	30/04/2000	1
Argentina	CB01	3/04/2000	1/05/2000	7
Argentina	L07	3/04/2000	1/05/2000	3
...	...	...	...	...

**Σ=52**

**YTD Ingresos Abril =**  
**VAR** UltimaFechaVisible = **MAX** (Calendario[Fecha])  
**VAR** AnActual = **YEAR** (UltimaFechaVisible)  
**RETURN**  
**CALCULATE** (  
 [Ingresos Tot];  
**FILTER** (  
**ALL** (Calendario[Fecha])  
**AND** (  
 Calendario[Fecha] <= UltimaFechaVisible;  
**YEAR**(Calendario[Fecha]) = AnActual  
 )  
 )  
**ALL** (Calendario) -- Insertado por Motor DAX  
 )

1

Identificar Filtros

2

Aplicar Filtros

3

Ejecutar Expresión DAX



```

YTD Ingresos =
VAR UltimaFechaVisible =
    MAX ( Calendario[Fecha] )
VAR AnActual =
    YEAR ( UltimaFechaVisible )
RETURN
    CALCULATE (
        [Ingresos Tot];
        FILTER (
            ALL ( Calendario[Fecha] );
            AND (
                Calendario[Fecha] <= UltimaFechaVisible;
                YEAR ( Calendario[Fecha] ) = AnActual
            )
        )
    )

```

```

YTD Ingresos =
VAR UltimaFechaVisible =
    MAX ( Calendario[Fecha] )
VAR AnActual =
    YEAR ( UltimaFechaVisible )
RETURN
    CALCULATE (
        [Ingresos Tot];
        AND (
            Calendario[Fecha] <= UltimaFechaVisible;
            YEAR ( Calendario[Fecha] ) = AnActual
        )
    )

```

```

YTD Ingresos =
VAR UltimaFechaVisible =
    MAX ( Calendario[Fecha] )
VAR AnActual =
    YEAR ( UltimaFechaVisible )
RETURN
    CALCULATE (
        [Ingresos Tot];
        Calendario[Fecha] <= UltimaFechaVisible
        && YEAR ( Calendario[Fecha] ) = AnActual
    )

```



A Sweet: Syntax Sugar!



Otra forma un poquito más dulce

Tres condiciones para la agregación automática de ALL como argumento de filtro, el cual facilita la escritura de expresiones DAX de inteligencia de tiempo:

1

Filtro que afecta a una columna de Fechas

Un filtro generado por un argumento de CALCULATE



2

Columna de tipo Fecha o Fecha y Hora

Sólo admisible para esos dos tipos de datos



3

La Columna debe ser Clave primaria

Debe ser en una Relación Uno a Muchos



Si algunas de las tres condiciones no se cumple, el ALL automático que facilite las expresiones DAX no se agrega, por lo tanto, puede llevar a cálculos no correctos y conclusiones catastróficas.

Muchos modelos de datos utilizan un número entero en la tabla de Calendario y en la tabla transaccional para crear la relación, generalmente en un formato AAAAMMDD (YYYYMMDD) algo del estilo: 20191019.

No obstante, al ser creados de esta manera **se incumple dos de las tres condiciones** para la agregación del ALL automática, específicamente:

**Primero** Filtro que afecta columna de fechas y **Segundo** el formato de la columna no es de tipo fecha o fecha y hora

## Marcar Tabla Como Tabla de Fechas

2



**Marcar Como Tabla de Fechas:** En ocasiones, y sobre todo, en modelo de daos antiguos no siempre se utilizaba una columna de fechas para relacionar la tabla de calendario con una tabla transaccional, puesto que en lugar de ellos se utiliza un número entero en formato YYYYMMDD