Principales funciones de inteligencia de tiempo

ACUMULATIVAS — TOTALYTD | TOTALQTD | TOTALMTD

Permiten tratar los importes acumulados en un intervalo de tiempo establecido. TOTAL. Este trio de funciones retorna una tabla con las fechas desde el principio de año, trimestre o mes *permitiendo acumular por año, trimestre o mes.*

COMPARATIVAS — SAMEPERIODLASTYEAR | DATEADD | PARALLELPERIOD…..

Permiten comparar un importe entre un periodo de tiempo de retraso deseado (del mes anterior, del año anterior,…), es decir, estas funciones nos van a permitir seleccionar cuánto tiempo atrás quiero moverme en términos de plazos, por ejemplo, tres días, meses, trimestres o años

Por ejemplo: *Comparando versus el mes previo.* ***Estimar tasa de crecimiento***

De Intervalo y otras — DATESINPERIOD | DATESBETWEEN | FIRSTDATE | ……..

Acumulativas | TOTALYTD | TOTALQTD | TOTALMTD

TOTALYTD

Para importes acumulados

Sintaxis y argumentos: **TOTALQTD (expression,dates,[filter])**

- expression: Expresión que devuelve un escalar.
- dates: Columna conteniendo fechas.
- filter: Argumento opcional. Filtro a aplicar al contexto actual.

Fecha	Importe
martes, 15 de septiembre de 2015	2
miércoles, 30 de septiembre de 2015	3
lunes, 5 de octubre de 2015	1
miércoles, 7 de octubre de 2015	2
miércoles, 11 de noviembre de 2015	3
viernes, 1 de enero de 2016	5
viernes, 15 de enero de 2016	2
miércoles, 3 de febrero de 2016	5
miércoles, 17 de febrero de 2016	1
viernes, 1 de abril de 2016	2
sábado, 21 de mayo de 2016	3
viernes, 15 de julio de 2016	4
miércoles, 28 de septiembre de 2016	2

Ejemplo TOTALYTD Fuente: https://interactivechaos.com/es/dax/function/totalytd

Disponemos de la siguiente tabla de ventas y definimos la medida de ventas totales como: **Ventas = SUM(Ventas[Importe]).**

A continuación, definimos la medida Ventas YTD que calcula el total de ventas desde principio de año del contexto actual hasta la última fecha del contexto actual: **Ventas YTD = TOTALYTD(Ventas[Ventas], Calendario[Fecha]).**

Si llevamos esta última medida a una matriz, obtenemos:

Mes Año	Ventas YTD
2015-09	5
2015-10	8
2015-11	11
2015-12	11
2016-01	7
2016-02	13
2016-03	13
2016-04	15
2016-05	18
2016-06	18
2016-07	22
2016-08	22
2016-09	24
2016-10	24
2016-11	24

Ejemplo TOTALYTD, lo mismo que el anterior pero para trimestres, es decir

Creamos la medida "Ventas QTD" calcula el total de ventas desde principio del trimestre del contexto actual hasta la última fecha del contexto actual:

Ventas QTD = TOTALQTD(Ventas[Ventas], Calendario[Fecha])

Mes Año	Ventas QTD
2015-09	5
2015-10	3
2015-11	6
2015-12	6
2016-01	7
2016-02	13
2016-03	13
2016-04	2
2016-05	5
2016-06	5
2016-07	4
2016-08	4
2016-09	6

Se observa cómo la medida se reinicializa a cero al principio de cada trimestre.

Son una evolución de las tres funciones (DATESYTD – DATESQTD- DATESMTD)

TOTALQTD - TOTALMTD

Ejemplo: https://aprendebi.wordpress.com/2018/09/23/time-intelligence-with-power-bi-part-1/

Cuando necesites hacer acumulados puedes hacer uso de estas funciones: TOTALYTD, TOTALQTD y TOTALYTD, puedes usarlas de la siguiente manera:

Ventas YTD =

TOTALYTD (SUM ('Datos Tipo 1°'[Venta]); 'Calendario Tipo 1°'[Fecha])

Ventas MTD =

TOTALMTD (SUM ('Datos Tipo 1°'[Venta]); 'Calendario Tipo 1°'[Fecha])

Ventas QTD =

TOTALQTD (SUM ('Datos Tipo 1°'[Venta]); 'Calendario Tipo 1°'[Fecha])

Obtendrás un resultado en tabla como este:

Ventas Acumuladas YTD

Mes	Ventas	Ventas YTD
Diciembre	12,457.2	139,229.1
Noviembre	12,156.1	126,771.9
Mayo	12,013.2	56,586.9
Agosto	11,891.8	91,469.2
Octubre	11,812.4	114,615.8
Julio	11,750.5	79,577.4
Marzo	11,692.8	33,215.5
Enero	11,372.0	11,372.0
Abril	11,358.2	44,573.7
Setiembre	11,334.3	102,803.5
Junio	11,240.1	67,827.0
Febrero	10,150.8	21,522.7
Total	139,229.1	139,229.1

Ventas Acumuladas QTD

Mes	Ventas	Ventas QTD
Enero	11,372.0	11,372.0
Febrero	10,150.8	21,522.7
Marzo	11,692.8	33,215.5
Abril	11,358.2	11,358.2
Mayo	12,013.2	23,371.4
Junio	11,240.1	34,611.5
Julio	11,750.5	11,750.5
Agosto	11,891.8	23,642.2
Setiembre	11,334.3	34,976.5
Octubre	11,812.4	11,812.4
Noviembre	12,156.1	23,968.5
Diciembre	12,457.2	36,425.6
Total	139,229.1	36,425.6

Ventas Acumuladas MTD

Fecha	Ventas	Ventas MTD
1/01/2017	487.7	487.7
2/01/2017	506.1	993.7
3/01/2017	389.9	1,383.6
4/01/2017	289.0	1,672.5
5/01/2017	428.5	2,101.0
6/01/2017	304.8	2,405.8
7/01/2017	579.3	2,985.0
8/01/2017	322.3	3,307.3
9/01/2017	299.0	3,606.2
10/01/2017	472.5	4,078.7
11/01/2017	429.8	4,508.5
12/01/2017	244.1	4,752.5
Total	11,372.0	11,372.0

COMPARATIVAS | SAMEPERIODLASTYEAR | DATEADD | PARALLELPERIOD…..

Para comparar

SAMEPERIODLASTYEAR

La función de inteligencia de tiempo SAMEPERIODLASTYEAR significa mismo periodo, pero del año anterior.

Posiblemente sea la función de inteligencia de tiempo más utilizada. Gracias a este tipo de uso, *podemos comparar un importe entre un periodo de tiempo*, con su importe en el mismo periodo de tiempo, pero del año anterior. **Esta función es la más usada de las tres;**
SamePeriodLastYear nos dará exactamente, el mismo período pero del año pasado. OJO el mismo periodo significa que si estás viendo los datos del mes, te devolverá el mismo mes, pero del año pasado y si vez el día, te devolverá el mismo día, pero del año pasado.

Sintaxis y argumentos: **SAMEPERIODLASTYEAR (dates)**

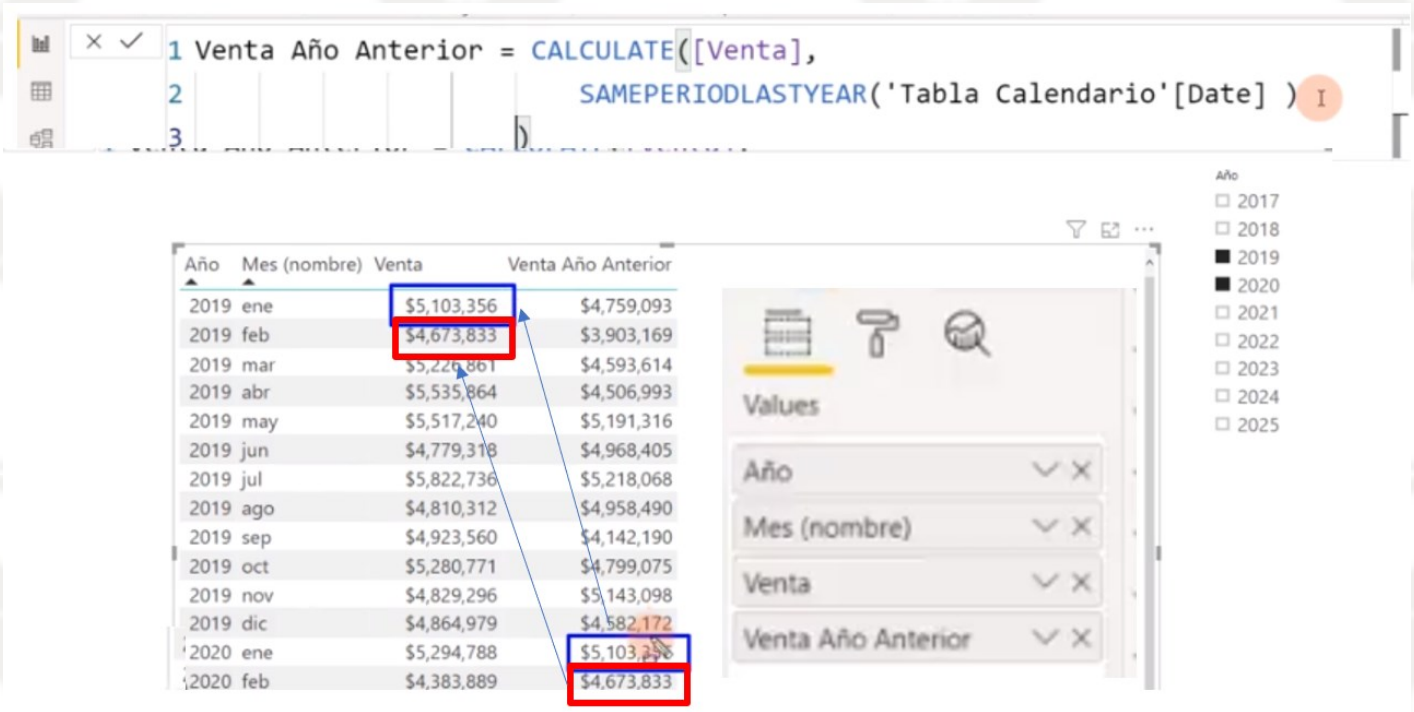
- Date: Lo unico que nos pide esta función es la columna fecha de nuestra tabla calendario

Para utilizarla simplemente tenemos que **crear una medida**, y en la parte del filtro poner la función **SAMEPERIODLASTYEAR** con el parámetro de tipo fecha por el que queramos comparar, por defecto será la columna fecha de nuestra dimensión de tiempo.
Estas medidas se usan con la función CALCULATE porque nos permitirá modificar el contexto de filtro en este caso el del año y mes activo por el del periodo similar equivalente (periodo anterior), es decir lo que estamos intentando es modificar el contexto de filtro.

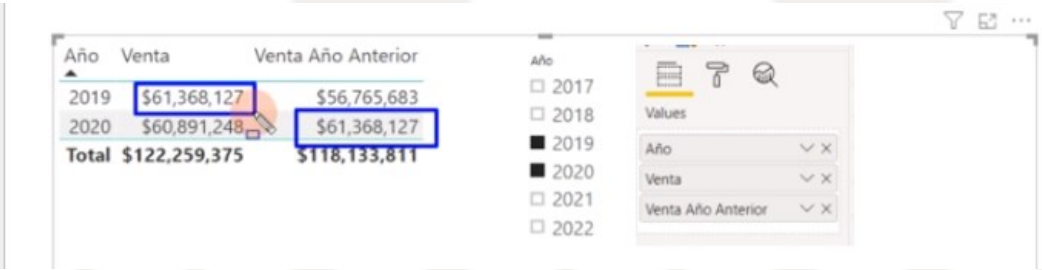
Asi si queremos que la medida nos devuelva las ventas, pero correspondiente al periodo anterior.

```
Venta Año Anterior = CALCULATE([Venta],
                        SAMEPERIODLASTYEAR('Tabla Calendario'[Date] )
                    )
```

Asi creada la nueva medida y añadida a la visualización obtenemos el siguiente resultado



Pero ademas estas funciones de inteligencia de tiempo son dinámicas en el sentido de que si ahora si quitamos el desglose por mes y dejamos el desglose por año se ajusta automáticamente.



Ejemplo – Ventas diarias del año pasado: Fuente: https://inteligencia.dds.pe/dax/funciones-de-inteligencia-de-tiempo-en-dax/

Vamos a crear una medida para comparar las ventas actuales con las del año pasado. Aplicaremos **SAMEPERIODLASTYEAR** combinando la función CALCULATE que recordemos nos permite cambiar el contexto de cualquier cálculo.
SAMEPERIODLASTYEAR devuelve un conjunto de fechas en la selección actual, pero del año anterior. Por último, necesitamos hacer referencia a la columna **Fechas** de la tabla **Tiempo**.

```
Ventas del Año Pasado =
    CALCULATE([Venta Total],SAMEPERIODLASTYEAR(Tiempo[Fecha]))
```

El resultado de la aplicación de esta medida es el de la siguiente tabla donde observamos los datos de ventas actuales en la columna Venta total con los de la columna de la medida Ventas del Año Pasado. La tabla Ventas del Año Pasado calcula las ventas totales, sin embargo, el contexto de la fecha se cambia al año anterior.

Fecha

01/01/2017 31/12/2018

Fecha	Venta Total	Ventas del Año Pasado
domingo, 1 de enero de 2017	$280,650.09	$306,288.84
lunes, 2 de enero de 2017	$563,182.46	$667,560.05
martes, 3 de enero de 2017	$321,628.96	$800,599.29
miércoles, 4 de enero de 2017	$80,320.91	$205,994.60
jueves, 5 de enero de 2017	$682,575.17	$106,780.43
viernes, 6 de enero de 2017	$985,562.55	$451,197.13
sábado, 7 de enero de 2017	$676,083.75	$862,581.55

Ejemplo – Ventas mensuales año pasado:

Fuente: https://interactivechaos.com/es/dax/function/sameperiodlastyear

Disponemos de una tabla con ventas y definimos la medida Ventas (como total ventas) de la siguiente forma:

Ventas = SUM(FactSales[SalesAmount])

Creamos una medida para calcular las ventas del mismo período que el implicado en el contexto un año antes:

Ventas año anterior = CALCULATE([Ventas], SAMEPERIODLASTYEAR(DimDate[Datekey]))

Creamos una tabla con estas dos medidas y vemos el resultado

Year	Month	Ventas	Ventas año anterior
2007	January	193.305.554,64 €	
	February	209.439.067,93 €	
	March	203.991.979,69 €	
	April	276.891.048,16 €	
	May	288.749.508,61 €	
	June	283.186.644,54 €	
	July	272.818.635,11 €	
	August	263.780.279,28 €	
	September	257.282.781,95 €	
	October	288.853.903,92 €	
	November	308.752.784,65 €	
	December	297.341.103,65 €	
	Total	**3.144.393.292,13 €**	
2008	January	183.970.020,28 €	193.305.554,64 €
	February	191.106.948,30 €	209.439.067,93 €
	March	183.393.312,89 €	203.991.979,69 €
	April	223.849.292,33 €	276.891.048,16 €
	May	220.502.302,24 €	288.749.508,61 €
	June	214.455.381,54 €	283.186.644,54 €
	July	246.239.251,91 €	272.818.635,11 €
	August	231.189.642,07 €	263.780.279,28 €
	September	227.942.617,84 €	257.282.781,95 €
	October	211.203.579,42 €	288.853.903,92 €
	November	247.706.608,64 €	308.752.784,65 €
	December	260.854.259,59 €	297.341.103,65 €
	Total	**2.642.413.217,03 €**	**3.144.393.292,13 €**
2009	January	183.941.322,57 €	183.970.020,28 €
	February	180.954.406,26 €	191.106.948,30 €
	March	180.981.062,98 €	183.393.312,89 €
	April	211.894.484,93 €	223.849.292,33 €
	May	234.303.813,59 €	220.502.302,24 €
	June	227.822.106,40 €	214.455.381,54 €

DATEADD

Devuelve una tabla que contiene una única columna de valores de fechas que se han desplazado hacia delante o atrás en el tiempo según el número especificado de intervalos desde las fechas del contexto actual. *Esa tabla devuelta la vamos a utilizar con CALCULATE*

Sintaxis y argumentos: **DATEADD(<dates>,<number_of_intervals>,<interval>)**

- Date: Una columna que contiene fechas.
- number of intervals: Número de intervalos que se van a sumar o restar a las fechas
- Interval : Intervalo por el que se van a desplazar las fechas. El valor del intervalo puede ser uno de los siguientes: year, quarter, month o day

Notas : Si el número de intervalos especificado para number_of_intervals es positivo, las fechas de dates se mueven hacia delante en el tiempo; si el número es negativo, las fechas de dates se desplazan hacia atrás en el tiempo.

- DateAdd suma o resta una cantidad ya sea de días, meses, trimestres o años desde o hacia un campo de fecha.
- El parámetro interval es una enumeración, no un conjunto de cadenas, por lo tanto, los valores no deben ir entre comillas. Además, los valores year, quarter, month y day deben escribirse completos cuando se usan.
- La tabla de resultados solo incluye fechas que existen en la columna dates

DateAdd comparte la característica de ser dinámica en el periodo como SamePeriodLastYear, pero SamePeriodLastYear solo dura un año en cambio DateAdd puede durar dos años o incluso más; es por eso que DateAdd es una versión personalizada de SamePeriodLastYear.

Ejemplo: Curso Completo Power BI Desktop - Última versión de Power BI | Udemy - De **Diego Lopez**

Hemos calculado la suma de ventas del mes previo al seleccionado usando la función DATEADD, que lo que viene hacer es devolver una tabla con las fechas correspondientes al mes anterior del seleccionado y aplica la medida ventas (suma de ventas), es decir crear una tabla virtual con el conjunto de fechas correspondiente al mes anterior del que esta seleccionado o filtrado en este momento y sobre esta selección aplica la medida ventas.

Ventas mes previo

```
1 Venta Mes Previo =
2 CALCULATE([Venta],
3     DATEADD('Tabla Calendario'[Date], -1, MONTH)
4 )
```

Año	Mes (nombre)	Venta	Venta Año Anterior	Venta Mes Previo
2019	ene	$5,103,356	$4,759,093	$4,582,172
2019	feb	$4,673,833	$3,903,169	$5,103,356
2019	mar	$5,226,861	$4,593,614	$4,673,833
2019	abr	$5,535,864	$4,506,993	$5,226,861
2019	may	$5,517,240	$5,191,316	$5,535,864
2019	jun	$4,779,318	$4,968,405	$5,517,240
2019	jul	$5,822,736	$5,218,068	$4,779,318
2019	ago	$4,810,312	$4,958,490	$5,822,736
2019	sep	$4,923,560	$4,142,190	$4,810,312
2019	oct	$5,280,771	$4,799,075	$4,923,560
2019	nov	$4,829,296	$5,143,098	$5,280,771
2019	dic	$4,864,979	$4,582,172	$4,829,296

Ventas trimestre previo

```
1  Venta Trimestre previo =
2 CALCULATE([Venta],
3     DATEADD('Tabla Calendario'[Date], -1, QUARTER)
4 )
```

Año Trimestre	Vtas Trimestre	Vtas Trimestre Año anterior	Vtas Trimestre anterior
2019 Trim. 1	$15,004,051	$13,255,876	$14,524,345
2019 Trim. 2	$15,832,422	$14,666,715	$15,004,051
2019 Trim. 3	$15,556,608	$14,318,748	$15,832,422
2019 Trim. 4	$14,975,046	$14,524,345	$15,556,608
2020 Trim. 1	$14,722,178	$15,004,051	$14,975,046
2020 Trim. 2	$16,045,740	$15,832,422	$14,722,178
2020 Trim. 3	$15,110,860	$15,556,608	$16,045,740
2020 Trim. 4	$15,012,469	$14,975,046	$15,110,860
Total	**$122,259,375**	**$118,133,811**	**$121,771,251**

Ejemplo con DATEADD: Crecimiento respecto a un periodo anterior

Fuente: https://interactivechaos.com/es/dax/scenario/crecimiento-respecto-un-periodo-anterior

Hemos calculado la suma de ventas del mes previo al seleccionado usando la función DATEADD, que lo que viene hacer es devolver una tabla con las fechas correspondientes al mes anterior del seleccionado y aplica la medida ventas (suma de ventas), es decir crear una tabla virtual con el conjunto de fechas correspondiente al mes anterior del que esta seleccionado o filtrado en este momento y sobre esta selección aplica la medida ventas.

Un escenario frecuente es el que supone el cálculo de crecimiento de una métrica expresado como porcentaje (ventas, número de clientes, número de unidades vendidas, etc.) con respecto a un cierto período anterior, por ejemplo un año o un trimestre, por poner un par de ejemplos. El cálculo básico a realizar es el siguiente:

[% crecimiento] = (Valor actual - Valor hace un año) / Valor hace un año

Para el cálculo del "Valor hace un año" vamos a utilizar la función DATEADD que devuelve una tabla conteniendo una columna de fechas que coinciden con las implicadas en el contexto actual desplazadas hacia adelante o hacia atrás un número de intervalos determinado. De esta forma, si la métrica es [Valor actual], podríamos definir:

[Valor hace un año] = CALCULATE([Valor actual];DATEADD(DimDate[Datekey];-1;YEAR))

Por último, la definición de la métrica buscada sería:

```
% crecimiento = DIVIDE(
        [Valor actual] - [Valor hace un año];
        [Valor hace un año];)
)
```

La función DATEADD nos permite referir el crecimiento a cualquier número de días, meses, trimestres o años (atrás o adelante en el tiempo).

PARALLELPERIOD

Es una función que tiene la capacidad de obtener el período paralelo al período actual, es decir puede navegar a períodos en el pasado o en el futuro, pero que está determinado estéticamente en el parámetro del Intervalo y puede ser solo consultado por Mes, Trimestre o Año. Su estructura es muy parecida a la de DateAdd.

¿Qué lo hace diferente al ser estático?; veamos este ejemplo para que se entienda mejor; al usar ParallelPeriod para consultar las ventas del mes pasado usando esta medida:

```
Ventas Last Month =
CALCULATE (
   SUM ( 'Datos Tipo 1°'[Venta] );
   PARALLELPERIOD ( 'Calendario Tipo 1°'[Fecha]; -1; MONTH )
```

- Primero, **DateAdd** y **SamePeriodLastYear** son funciones que navegan en un periodo dinámico a través del contexto del filtro (Filter Context); con la diferencia que **DateAdd** puede retornar más de un año atrás.
- Segundo, **DateAdd** y **ParallelPeriod** pueden ir atrás y adelante según sea el parámetro y en cuanto a intervalo de tiempo **DateAdd** sirve para día, mes, trimestre y año, mientras que **ParallelPeriod** solo por mes, trimestre y año.
- Tercero, **ParallelPeriod** a pesar de su gran parecido en estructura con **DateAdd**,el solo funciona de una manera estática en función de su intervalo

PARALLELPERIOD

Es una función que tiene la capacidad de obtener el período paralelo al período actual, es decir puede navegar a períodos en el pasado o en el futuro, pero que está determinado estaticamente en el parámetro del Intervalo y puede ser solo consultado por Mes, Trimestre o Año. Su estructura es muy parecida a la de DateAdd. **¿Qué lo hace diferente al ser estático?;** veamos este ejemplo al usar ParallelPeriod para consultar las ventas del mes pasado usando esta medida:

```
Ventas Last Month =
CALCULATE (
    SUM ( 'Datos Tipo 1°'[Venta] );
    PARALLELPERIOD ( 'Calendario Tipo 1°'[Fecha]; -1; MONTH )
```

El resultado en Tabla seria este

Cuando hacemos un Drilldown en Febrero podemos observar que en la vista a nivel diario ParallelPeriod devolverá las ventas de todo el último mes sin importar que nosotros estemos viendo el día; manera estática como se determinó en el parámetro del intervalo.

Año	Ventas	Ventas Last Month
2017	139,229.1	138,516.7
enero	11,372.0	11,744.8
febrero	10,150.8	11,372.0
marzo	11,692.8	10,150.8
abril	11,358.2	11,692.8
mayo	12,013.2	11,358.2
junio	11,240.1	12,013.2
julio	11,750.5	11,240.1
agosto	11,891.8	11,750.5
septiembre	11,334.3	11,891.8
octubre	11,812.4	11,334.3
noviembre	12,156.1	11,812.4
diciembre	12,457.2	12,156.1
Total	139,229.1	138,516.7

Año	Ventas	Ventas Last Month
febrero	10,150.8	11,372.0
1	261.3	11,372.0
2	313.8	11,372.0
3	248.2	11,372.0
4	554.2	11,372.0
5	279.2	11,372.0
6	306.0	11,372.0
7	210.8	11,372.0
8	248.9	11,372.0
9	386.3	11,372.0
10	267.5	11,372.0
11	495.6	11,372.0
12	497.2	11,372.0

Conclusiones: DATEADD –SAMEPERIODLASTYEAR-PARALLELPERIOD

1. DateAdd y SamePeriodLastYear son funciones que navegan en un periodo dinámico a través del contexto del filtro (Filter Context); con la diferencia que DateAdd puede retornar más de un año atrás, es decir, SAMEPERIODLASTYEAR y DATEADD (con parámetro "-1 year") son equivalentes. Y devuelven el mismo resultado. SAMEPERIODLASTYEAR sólo sirve para año paralelo mientras que DATEADD permite agregar parámetros para calcular otros periodos paralelos: Meses, dias…
2. DateAdd y ParallelPeriod pueden ir atrás y adelante según sea el parámetro y en cuanto a intervalo de tiempo DateAdd sirve para día, mes, trimestre y año, mientras que ParallelPeriod solo por mes, trimestre y año.
3. ParallelPeriod a pesar de su gran parecido en estructura con DateAdd,el solo funciona de una manera estática en función de su intervalo.
4. PARALLELPERIOD es similar a DATEADD, pero la segunda permite especificar periodos paralelos hasta nivel día ("day"). Mientras que la primera sólo admite "year", "quarter", "month" como tercer parámetro. Por eso, cuando has seleccionado del día 2/3/2018 al 15/5/2018, PARALLELPERIOD devuelve en este caso el importe del año anterior completo

De Intervalo y otras | DATESINPERIOD | DATESBETWEEN | FIRSTDATE | ……

Hay dos funciones que funcionan de manera muy similar entre sí, pero tienen un uso un poco diferente; DatesInPeriod y DatesBetween y como primera aproximación tenemos:

- **DatesInPeriod** es una función perfecta para calcular períodos estándar que siguen intervalos de día, mes, trimestre y año pero siempre en relación con el período actual.
- **DatesBetween** es una versión más genérica de DatesInPeriod en el sentido de que ofrece mas flexibilidad. Con esta función, no necesita preocuparse por el intervalo o el número de intervalos. Esta función le dará todas las fechas entre una fecha de inicio y una fecha de finalización. Es útil cuando necesita analizar datos dentro de un intervalo de fechas específico

DATESINPERIOD

Devuelve una tabla que contiene una columna de fechas que empieza por una fecha de inicio específica y sigue hasta el número y tipo de intervalos de fechas especificados, hacia atrás o hacia adelante en el tiempo. Si el número de intervalos indicado es positivo, el intervalo indicado se añadirá a la fecha de inicio dicho número de veces. Por el contrario, si es negativo se restará. Pero siempre en relación con el período actual Esta función es adecuada para pasar como filtro en la función CALCULATE, permite un control total sobre el intervalo de fechas de una medida.

Sintaxis y argumentos: **DATESINPERIOD(<dates>, <start_date>, <number_of_intervals>, <interval>)**

- Date: Una columna que contiene fechas.
- start date. Una expresión de fecha.
- number of intervals: Número de intervalos que se van a sumar o restar a las fechas
- Interval : Intervalo por el que se van a desplazar las fechas. El valor del intervalo puede ser uno de los siguientes: year, quarter, month o day

Ejemplo 1 con DATESINPERIOD: cálculo del total de ventas anual móvil (Revenue MAT - Moving Annual Total) o ingresos totales anuales móviles

Fuente: - La función DAX DATESINPERIOD | Soluciones G Com (gcomsolutions.co.uk)

Disponemos de una tabla con los ingresos totales según mes y año y vamos a calcular Total Anual Móvil (MAT) de los ingresos y que llamaremos "**Revenue MAT**". Usaremos la función CALCULATE para la medida Ingresos Totales [Total Revenue] y como argumento de filtro usamos la función DATESINPERIOD, de esta forma la medida será:

Year	Month	Total Revenue
2014	January	£1,729,615
2014	February	£1,581,659
2014	March	£1,742,155
2014	April	£1,513,329
2014	May	£1,762,127

```
1  Revenue MAT =
2  CALCULATE(
3      [Total Revenue],
4      DATESINPERIOD(
5          'Date Table'[Date],
6          MAX('Date Table'[Date]),
7          -1,
8          YEAR
9      )
10 )
```

Year	Month	Total Revenue	Revenue MAT
2014	January	£1,729,615	£1,729,615
2014	February	£1,581,659	£3,311,274
2014	March	£1,742,155	£5,053,429
2014	April	£1,513,329	£6,566,758
2014	May	£1,762,127	£8,328,885
2014	June	£1,639,773	£9,968,658
2014	July	£1,609,794	£11,578,452
2014	August	£1,719,731	£13,298,183
2014	September	£1,635,975	£14,934,158
2014	October	£1,760,912	£16,695,070
2014	November	£1,796,654	£18,491,724
2014	December	£1,627,926	£20,119,650
2015	January	£1,837,315	£20,227,350
2015	February	£1,519,182	£20,164,872
2015	March	£1,810,628	£20,233,345
Total		**£133,833,761**	**£26,884,906**

Para nuestro argumento Start Date, queremos la última fecha en el período actual que se puede obtener con MAX('Date Table'[Date]).

Estamos trabajando hacia atrás a partir de ahí, por tanto, el número de intervalos es menos uno, y el intervalo, o unidades de tiempo, es el año.

Ejemplo 2 DATESINPERIOD: cálculo promedio de ingresos de los tres últimos meses

Fuente: - La función DAX DATESINPERIOD | Soluciones G Com (gcomsolutions.co.uk)

Retomamos el caso anterior donde disponemos de la tabla con los ingresos según mes y año y queremos calcular el promedio mensual de los tres últimos meses (la media móvil tres meses), lo que implica que tendremos que dividir el acumulado trimestral por tres y llamaremos a esta medida Revenue 3MMA

Year	Month	Total Revenue
2014	January	£1,729,615
2014	February	£1,581,659
2014	March	£1,742,155
2014	April	£1,513,329
2014	May	£1,762,127

Por tanto, usamos la función DIVIDIR y dentro de eso, usamos CALCULATE. La expresión que se va a calcular es [Total Revenue] y para nuestra expresión de filtro, usamos DATESINPERIOD. El argumento número de intervalos es menos tres; es decir, los últimos tres meses. Y el intervalo, o unidades, es Mes.

```
1  Revenue 3MMA =
2  DIVIDE(
3      CALCULATE(
4          [Total Revenue],
5          DATESINPERIOD(
6              'Date Table'[Date],
7              MAX('Date Table'[Date]),
8              -3,
9              MONTH)
10     ),
11     3
12 )
```

Year	Month	Total Revenue	Revenue MAT	Revenue 3MMA
2014	January	£1,729,615	£1,729,615	£576,538
2014	February	£1,581,659	£3,311,274	£1,103,758
2014	March	£1,742,155	£5,053,429	£1,684,476
2014	April	£1,513,329	£6,566,758	£1,612,381
2014	May	£1,762,127	£8,328,885	£1,672,537
2014	June	£1,639,773	£9,968,658	£1,638,410
2014	July	£1,609,794	£11,578,452	£1,670,565
2014	August	£1,719,731	£13,298,183	£1,656,432
2014	September	£1,635,975	£14,934,158	£1,655,167
2014	October	£1,760,912	£16,695,070	£1,705,539
2014	November	£1,796,654	£18,491,724	£1,731,180
2014	December	£1,627,926	£20,119,650	£1,728,497
2015	January	£1,837,315	£20,227,350	£1,753,965
2015	February	£1,519,182	£20,164,872	£1,661,474
2015	March	£1,810,628	£20,233,345	£1,722,375
Total		**£133,833,761**	**£26,884,906**	**£2,318,743**

Cuando agregamos nuestra función Revenue 3MMA a nuestra tabla visual, podemos ver que, a partir del tercer mes, tenemos una cifra que podemos comparar con el rendimiento del mes actual para ver si el mes actual está por encima o por debajo de la media móvil de tres meses.

DATESBETWEEN

Devuelve una tabla conteniendo una única columna de fechas que comienza y termina con las fechas incluidas como parámetros. Es decir, permite un control total sobre el intervalo de fechas de una medida.

Sintaxis y argumentos: **DATESBETWEEN(dates, start_date, end_date)**

- Date: Referencia a una columna que contiene fechas.
- start_date. Fecha inicial a considerar.
- end_date: Fecha final a considerar.

Ejemplo 1 con DATESBETWEEN: Fuente: https://interactivechaos.com/es/dax/function/datesbetween

Definimos la siguiente medida en la que calculamos las ventas entre el 2 de enero y el 4 de enero de 2007:

```
Ventas período = CALCULATE( [Ventas], DATESBETWEEN( FactSales[DateKey],  DATE(2007,1,2), DATE(2007,1,4)))
```

Esto nos serviría para comparar las ventas en campañas de distintos añosque no son coincidente en fechas ni duración, por ejemplo etiquetas con las ventas para los periodos (X-Y-Z)

- Semana Santa
- Puente de mayo
- Puente de diciembre
- Navidad…..

DateKey	SalesAmount
lunes, 1 de enero de 2007	6.085.839,18 €
martes, 2 de enero de 2007	6.270.657,17 €
miércoles, 3 de enero de 2007	6.096.024,11 €
jueves, 4 de enero de 2007	5.979.164,13 €
viernes, 5 de enero de 2007	5.926.584,02 €
sábado, 6 de enero de 2007	6.150.610,75 €
domingo, 7 de enero de 2007	6.517.040,34 €
lunes, 8 de enero de 2007	5.856.724,52 €
martes, 9 de enero de 2007	6.184.820,44 €
miércoles, 10 de enero de 2007	6.612.222,96 €
jueves, 11 de enero de 2007	6.524.046,81 €
viernes, 12 de enero de 2007	5.889.201,15 €
sábado, 13 de enero de 2007	6.330.346,71 €
Total	**8.341.224.364,83 €**

18.345.845,41 €

Ventas período

Ejemplo 2 con DATESBETWEEN: Calculo de la cantidad de ventas del día actual y del anterior

Fuente: DAX Power BI: DATESBETWEEN & DATESINPERIOD | por Andrei Khaidarov, MVP de Microsoft, PhD | Plataforma de energía | Medio (medium.com)

Vamos a contar la cantidad de ventas del día actual y del día anterior. Para obtener el valor anterior, podemos tomar la fecha y restar 1 de ella, así obtenemos la fecha anterior con la función FIRSTDATE y con la función LASTDATE

```
1 TotalCurrentAndPreviosDays =
2     CALCULATE(
3         SUM('Orders'[Total]),
4         DATESBETWEEN(
5             'Orders'[Date],
6             FIRSTDATE('Orders'[Date])-1,
7             LASTDATE('Orders'[Date])
8         )
9     )
```

Date	Total	TotalCurrentAndPreviosDays
31 октября 2021 г.	17 400	17 400
1 ноября 2021 г.	21 100	38 500
2 ноября 2021 г.	16 600	37 700
3 ноября 2021 г.	16 200	32 800
4 ноября 2021 г.	15 900	32 100

Código de tabla calendario desde la fecha indicada hasta hoy

25 formatos de calendario disponible

Formatos de fecha y hora definidos por el usuario

Todo ello disponible en el documento enlace:

Referencias

https://aleson-itc.com/%F0%9F%95%92-trabajando-con-funciones-de-time-intelligence-serie-dimdate-2-3/

Construcción de tablas de calendario y porqué son la bomba
https://www.excelfreeblog.com/tablas-de-calendario-porque-son-la-bomba/

Funciones DAX de inteligencia de tiempo

<https://interactivechaos.com/es/recursos-educativos/funciones-dax?title=&field_funcion_dax_categoria_value%5B%5D=Funciones+de+inteligencia+de+tiempo>

Inteligencia de tiempo en Power Pivot en Excel
https://support.office.com/es-es/article/inteligencia-de-tiempo-en-power-pivot-en-excel-016acf7b-9ded-411e-ba6c-ed8b8c368011?ui=es-ES&rs=es-ES&ad=ES