

DAX Time Intelligence, Ranking, Parameters & Dynamic Measures

Make dashboards answer "compared with when?". Build year-to-date, month-to-date, same-period-last-year, month-on-month and year-on-year measures, rolling averages, rankings with RANKX and TOPN, what-if parameters, field parameters and dynamic measure switching, and know visual calculations and calculation groups.

● Power BI classes 3–4

✦ Power BI Desktop 2026 (field parameters, calculation groups, visual calculations)

Advanced

🕒 150 min

By the end of this lesson you can

- Explain what time intelligence needs (a marked date table) and use TOTALYTD, DATESYTD, SAMEPERIODLASTYEAR, DATEADD and PARALLELPERIOD
- Build YoY %, MoM %, YTD, rolling 3-month and running-total measures
- Rank items with RANKX and show the Top N with TOPN, including an "Others" bucket
- Create what-if parameters and dynamic measure switching with field parameters
- Describe calculation groups, visual calculations and DAX query view, and when to use them

Datasets used

adventure_cycles/fact_sales.csv

adventure_cycles/dim_product.csv

adventure_cycles/dim_date.csv

What time intelligence needs

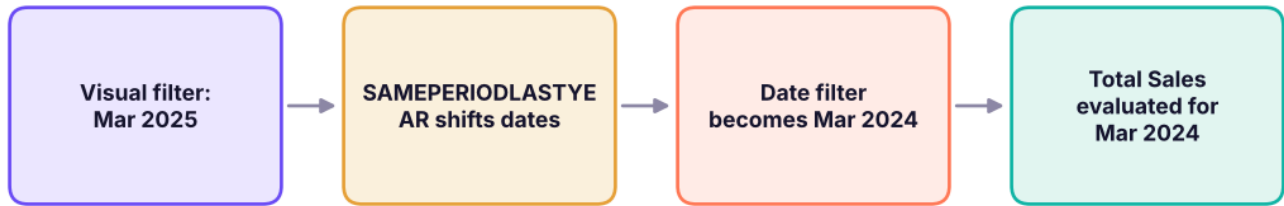
Time-intelligence functions move or expand the **date filter**: "this year to date", "the same period last year". They only work when:

1. You have a **date table** with every date (Lesson 2),
2. it is **marked as a date table**, and

3. it is **related** to the fact table's date column.

Always use the **date table's** `Date` column inside time functions, never the fact table's date.

How CALCULATE + time intelligence works



Time intelligence changes the date filter, then CALCULATE re-evaluates the measure.
SAMEPERIODLASTYEAR shifts the selected dates back exactly one year.

Core measures

DAX

```
Total Sales = SUM ( fact_sales[sales_amount] )

-- Cumulative totals
Sales YTD = TOTALYTD ( [Total Sales], dim_date[date] )
Sales QTD = TOTALQTD ( [Total Sales], dim_date[date] )
Sales MTD = TOTALMTD ( [Total Sales], dim_date[date] )
-- Indian financial year ending 31 March:
Sales FYTD = TOTALYTD ( [Total Sales], dim_date[date], "31-3" )

-- Previous periods
Sales LY   = CALCULATE ( [Total Sales], SAMEPERIODLASTYEAR ( dim_date[date] ) )
Sales PM   = CALCULATE ( [Total Sales], DATEADD ( dim_date[date], -1, MONTH ) )
Sales PQ   = CALCULATE ( [Total Sales], DATEADD ( dim_date[date], -1, QUARTER ) )

-- Growth
YoY %     = DIVIDE ( [Total Sales] - [Sales LY], [Sales LY] )
MoM %     = DIVIDE ( [Total Sales] - [Sales PM], [Sales PM] )
YoY Δ     = [Total Sales] - [Sales LY]
```

Function	Returns	Notes
TOTALYTD (expr, dates, [year_end])	Year-to-date total	Shortcut for CALCULATE (expr, DATESYTD (dates))
DATESYTD / DATESQTD / DATESMTD	The date set from the period start to the last visible date	Use inside CALCULATE
SAMEPERIODLASTYEAR (dates)	The selected dates shifted back one year	Needs continuous dates
DATEADD (dates, n, interval)	Shift by n days, months, quarters or years	Month/quarter shifts handle month-end well
PARALLELPERIOD (dates, n, interval)	The full period shifted (full month, quarter, year)	
DATESBETWEEN (dates, start, end)	Custom ranges	
DATESINPERIOD (dates, end, -n, interval)	Rolling window ending at end	Rolling 3/12 months
PREVIOUSMONTH / PREVIOUSQUARTER / PREVIOUSYEAR	Whole previous period	
EOMONTH , EDATE , DATEDIFF	Date arithmetic	

Let's compute what these measures return for our data (monthly sales, year-to-date, last year and YoY):

Python

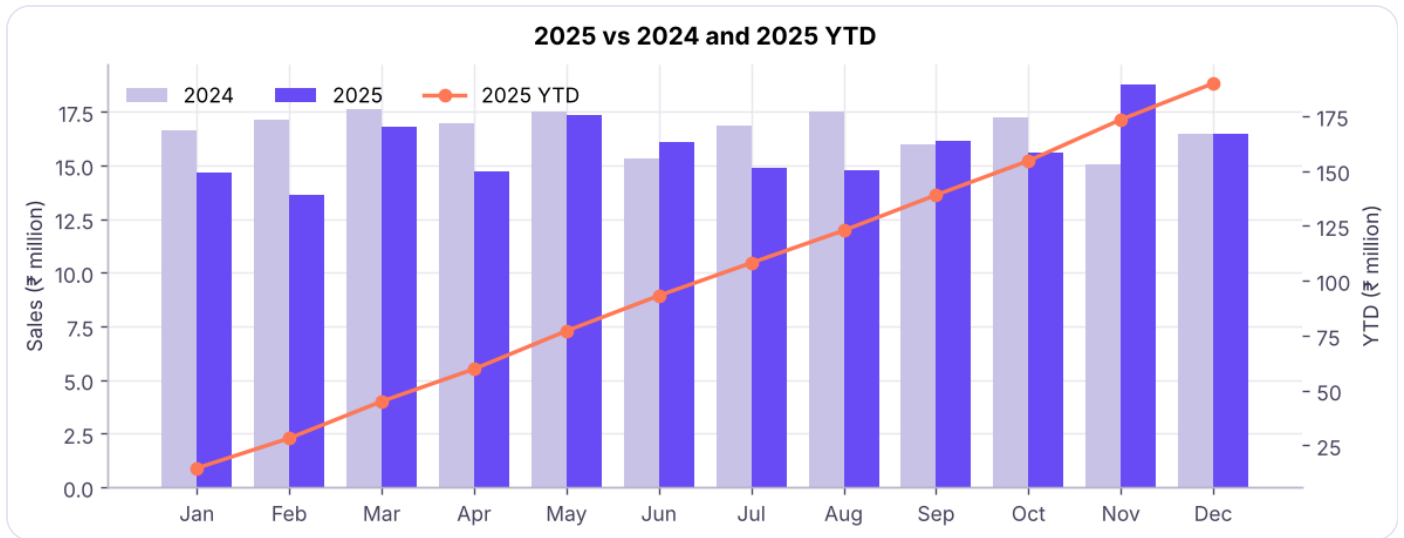
```
import pandas as pd
D = "adventure_cycles/"
f = pd.read_csv(D + "fact_sales.csv", parse_dates=["order_date"]); p = pd.read_csv(D + "dim_product.csv")
f["ym"] = f.order_date.dt.to_period("M")
mth = f.groupby("ym").sales_amount.sum().to_frame("Total Sales")
mth["Sales PM"] = mth["Total Sales"].shift(1)
mth["MoM %"] = (mth["Total Sales"] / mth["Sales PM"] - 1) * 100
mth["Sales LY"] = mth["Total Sales"].shift(12)
mth["YoY %"] = (mth["Total Sales"] / mth["Sales LY"] - 1) * 100
mth["Sales YTD"] = mth.groupby(mth.index.year)["Total Sales"].cumsum()
out = mth[mth.index.year == 2025].copy()
out["Total Sales"] = (out["Total Sales"] / 1e6).round(1); out["Sales LY"] = (out["Sales LY"] / 1e6).round(1); out["Sales YTD"] = (out["Sales YTD"] / 1e6).round(1)
out[["Total Sales", "Sales LY", "YoY %", "MoM %", "Sales YTD"]].round(1).rename_axis("2025 month")
```

OUTPUT

	Total Sales	Sales LY	YoY %	MoM %	Sales YTD
2025 month					
2025-01	14.6	16.7	-12.1	-11.2	14.6
2025-02	13.6	17.1	-20.5	-6.9	28.3
2025-03	16.8	17.6	-4.8	23.1	45.1
2025-04	14.7	17.0	-13.2	-12.1	59.8
2025-05	17.4	17.5	-1.0	17.7	77.2
2025-06	16.1	15.3	5.0	-7.4	93.2
2025-07	14.9	16.9	-11.5	-7.2	108.1
2025-08	14.8	17.5	-15.6	-1.0	122.9
2025-09	16.2	16.0	1.0	9.4	139.1
2025-10	15.6	17.3	-9.7	-3.5	154.7
2025-11	18.8	15.1	24.6	20.5	173.4
2025-12	16.5	16.5	0.0	-12.2	189.9

12 rows × 5 columns

(Values in ₹ million.) In a Power BI matrix with `dim_date[month]` on rows and these measures in the Values area you would see exactly this.



Monthly sales for 2025 against the same month in 2024, with year-to-date sales on the right axis. This is the classic "this year vs last year" chart.

COMMON MISTAKES

Common time-intelligence bugs

- **Date table not marked / has gaps:** SAMEPERIODLASTYEAR returns blank or wrong values.
- **Using the fact table's date column** in the function: use `dim_date[date]`.
- **Comparing partial periods:** the current month has data for 10 days, last year's month has 31, so YoY looks terrible. Compare "to date" with `DATESYTD` and `SAMEPERIODLASTYEAR`, or restrict to complete months.
- **Fiscal year:** pass "31-3" as the year-end argument to `TOTALYTD`, and create Fiscal Year and Fiscal Quarter columns in the date table.
- **Showing a "last year" value in months with no data** (future months): wrap with `IF ([Total Sales] <> BLANK (), ... )`.

Rolling windows and running totals

DAX

```
-- Rolling 3-month sales ending at the last visible date
Sales Rolling 3M =
CALCULATE ( [Total Sales], DATESINPERIOD ( dim_date[date], MAX ( dim_date[date] ), -3,
MONTH ) )

-- Rolling 3-month average of monthly sales (average over the months in the window)
Avg Monthly Sales R3M =
DIVIDE ( [Sales Rolling 3M], 3 )      -- valid once three full months exist in the window

-- Running total across all dates up to the current one
Sales Running Total =
CALCULATE ( [Total Sales], FILTER ( ALL ( dim_date[date] ), dim_date[date] ≤ MAX (
dim_date[date] ) ) )
```

In SQL this was `SUM() OVER (ORDER BY ... ROWS BETWEEN ...)` (SQL Lesson 12). DAX does the same thing by redefining the date filter.

Python

```
mth["Rolling 3M"] = mth["Total Sales"].rolling(3).sum()
mth["Rolling 3M avg"] = mth["Total Sales"].rolling(3).mean()
(mth[["Total Sales", "Rolling 3M", "Rolling 3M avg"]].tail(6) /
1e6).round(1).rename_axis("month (₹ million)")
```

OUTPUT

	Total Sales	Rolling 3M	Rolling 3M avg
month (₹ million)			
2025-07	14.9	48.3	16.1
2025-08	14.8	45.8	15.3
2025-09	16.2	45.8	15.3
2025-10	15.6	46.5	15.5
2025-11	18.8	50.5	16.8
2025-12	16.5	50.9	17.0

6 rows × 3 columns

Ranking: RANKX and TOPN

DAX

```
-- Rank of each product category by sales within the current context
Category Rank = RANKX ( ALL ( dim_product[category] ), [Total Sales], , DESC, DENSE )

-- Show only the top 5 products: sales of a product if it is in the top 5, else blank
Top 5 Product Sales =
VAR top5 = TOPN ( 5, ALL ( dim_product[product] ), [Total Sales], DESC )
RETURN CALCULATE ( [Total Sales], KEEPFILTERS ( top5 ) )

-- Everything outside the top 5, as one "Others" value
Others Sales = [Total Sales] - [Top 5 Product Sales]
```

For simple Top N visuals, you do not need DAX: in the **Filters pane**, choose the visual's field → Filter type **Top N** → Show items Top 5 → By value Total Sales → Apply filter. Use DAX when the Top N must respond to slicers in a custom way or you need an "Others" bar.

Python

```
prod_sales = f.merge(p, on="product_key").groupby(["category",
"product"]).sales_amount.sum().sort_values(ascending=False)
top5 = prod_sales.head(5)
print((top5 / 1e6).round(1).to_string())
print(f"\nTop 5 total : {top5.sum() / 1e6:,.1f} M    |    Others: {(prod_sales.sum() -
top5.sum()) / 1e6:,.1f} M    |    Top 5 share: {top5.sum() / prod_sales.sum():.1%}")
```

OUTPUT

category	product	
Bikes	Road Bike E-331	51.3
	Mountain Bike C-245	42.4
	Touring Bike A-507	39.7
	Mountain Bike D-662	36.2
	Mountain Bike B-168	35.9

Top 5 total : 205.4 M | Others: 389.2 M | Top 5 share: 34.5%

Parameters

What-if parameter

Modeling → New parameter → Numeric range: name Price Change %, from -20 to +20, increment 1, default 0, and tick **Add slicer**. Power BI creates a table and a measure Price Change % Value. Use it in your own measure to simulate:

DAX

```
Simulated Sales = [Total Sales] * ( 1 + [Price Change % Value] / 100 )  
Simulated Profit = [Simulated Sales] - [Total Cost]
```

Dragging the slicer instantly shows what profit would be with a 5% price increase (ignoring demand changes: always say so on the page).

Python

```
base_sales, base_cost = f.sales_amount.sum(), f.total_cost.sum()  
rows = [(chg, base_sales * (1 + chg / 100) / 1e6, (base_sales * (1 + chg / 100) -  
base_cost) / 1e6) for chg in (-10, -5, 0, 5, 10)]  
pd.DataFrame(rows, columns=["Price change %", "Simulated sales (₹ M)", "Simulated profit (₹  
M)"]).round(1)
```

OUTPUT

	Price change %	Simulated sales (₹ M)	Simulated profit (₹ M)
0	-10	535.1	150.7
1	-5	564.8	180.4
2	0	594.6	210.2
3	5	624.3	239.9
4	10	654.0	269.6

5 rows × 3 columns

Power Query parameters

Home → Manage parameters creates parameters used inside queries, for example StartDate or a ServerName. Use them to switch between a test and a production database, or to filter large tables at load time. In the Service, parameters can be changed in dataset settings.

Dynamic measure switching

A very popular pattern: a slicer that lets users choose which measure the chart shows (Sales, Profit, Orders).

The modern way: Field parameters. Modeling → **New parameter** → **Fields**: add Total Sales, Total Profit, Orders; tick **Add slicer**. Power BI creates a table with the three measures; use that

table's field on a chart's axis or values and the slicer swaps the measure. A field parameter can also switch **axis** fields (Category, Country, Year).

The classic way: a disconnected table + SWITCH.

DAX

```
-- Table "Measure Selector" (Enter data): columns Measure Name, Sort
Selected Measure =
SWITCH (
    SELECTEDVALUE ( 'Measure Selector'[Measure Name], "Sales" ),
    "Sales", [Total Sales],
    "Profit", [Total Profit],
    "Orders", [Orders]
)
```

Format strings can be dynamic too (Power BI 2024+ "dynamic format string"): thousands for sales, percentage for margin.

KPI visuals and forecasting

- **KPI visual:** needs a value (indicator), a trend axis (date) and a target (for example last year's sales). Colours show good/bad.
- **Gauge:** current vs target; min/max/target set in the format pane.
- **Forecasting:** in a **line chart** with a date axis, Analytics pane → **Forecast** → set forecast length (for example 6 months), confidence interval (95%) and seasonality (12 points). Power BI uses exponential smoothing (ETS). It is a quick what-comes-next view, not a replacement for proper forecasting models (ML track).
- **Analytics pane** also offers constant lines, average, min/max, percentile lines, trend lines, and **anomaly detection**.

Newer DAX features worth knowing

Feature	What it is
Calculation groups	A reusable "time-intelligence switcher": define Current, YTD, LY, YoY % once and apply them to every measure (created in Tabular Editor or in the model view in recent versions)
Visual calculations	DAX written inside a visual that can refer to other fields in the visual (running sum, moving average, % of parent) with simple functions such as RUNNINGSUM, MOVINGAVERAGE, PREVIOUS, FIRST
DAX query view	EVALUATE queries to test and debug measures; Copilot can write DAX here
User-defined functions (DAX UDFs)	Reusable functions defined with the FUNCTION keyword (preview)
INFO.* functions	Query model metadata (INFO.TABLES(), INFO.MEASURES())

DAX

```
-- Visual calculation (typed in the visual's formula bar)
Running Sales = RUNNINGSUM ( [Total Sales] )
Moving Avg 3 = MOVINGAVERAGE ( [Total Sales], 3 )
```

DO IT WITH AI Do it with AI

Copilot can write time-intelligence measures from a sentence ("Show sales year to date versus last year"). It works best if the date table is marked, columns and tables have clear names, and you add **descriptions** to measures (Model view → Properties). Check the generated DAX uses your `dim_date[date]` (not the fact table's date), and confirm one value by hand.

Summary

KEY TAKEAWAYS

- Time intelligence needs a **continuous, marked date table**, related to the fact; always reference the date table's column.
- `TOTALYTD`, `SAMEPERIODLASTYEAR`, `DATEADD`, `PARALLELPERIOD`, `DATESINPERIOD` build YTD, last-year, MoM, YoY and rolling windows; `DIVIDE ( current - previous, previous )` gives growth.
- `RANKX` ranks, `TOPN` selects the top items (add an "Others" measure); the Filters-pane **Top N** is the no-code route.
- **What-if parameters** simulate scenarios; **field parameters** let users switch measures or axes; the disconnected table + `SWITCH` pattern does the same in older models.
- Know calculation groups, visual calculations and the DAX query view for advanced work.

Quick quiz

6 questions · tap an answer to check it

1. Which measure shows year-to-date sales?

A `SUM ( fact_sales[sales_amount] )`

B `TOTALYTD ( [Total Sales], dim_date[date] )`

C `SAMEPERIODLASTYEAR ( [Total Sales] )`

D `RANKX ( dim_date, [Total Sales] )`

Answer: B. `TOTALYTD` accumulates from the start of the year to the current date.

2. SAMEPERIODLASTYEAR returns blank for every month. What should you check first?

A The theme

B That the date table is continuous, marked as a date table and related to the fact

C The colour of the visual

D The number of slicers

Answer: B. Time intelligence relies on a valid date table.

3. Which formula gives month-on-month growth?

A `[Total Sales] - [Sales PM]`

B `DIVIDE ( [Total Sales] - [Sales PM], [Sales PM] )`

C `SUM ( [Sales PM] ) / [Total Sales]`

D `DATEADD ( [Total Sales], -1, MONTH )`

Answer: B. Growth rate = (current – previous) ÷ previous, using DIVIDE.

4. What does a field parameter let users do?

A Change the report theme

B Switch which measures or fields a visual shows with a slicer

C Refresh the data

D Create relationships

Answer: B. Field parameters create a slicer that swaps fields in visuals.

5. In the Indian financial year (April–March), how do you build YTD sales?

A TOTALYTD ([Total Sales], dim_date[date]) always works

B TOTALYTD ([Total Sales], dim_date[date], "31-3")

C DATESMTD

D PARALLELPERIOD (...)

Answer: B. The third argument sets the year-end date.

6. What is the correct way to compute "Others" after a Top 5?

A Add a second visual

B Total sales minus the Top 5 measure

C Use FILTER ALL

D Use PREVIOUSMONTH

Answer: B. Others = overall total – Top N total.

EXERCISE Time-intelligence measures

For Adventure Cycles write measures for: (1) sales for the last 12 months ending at the selected date; (2) YoY % by category; (3) the best month of each year; (4) a what-if parameter "Discount %" from 0 to 30 that reduces Total Sales and shows simulated profit margin.

Show solution (worked answer)

DAX

```
-- (1)
Sales L12M = CALCULATE ( [Total Sales], DATESINPERIOD ( dim_date[date], MAX (
dim_date[date] ), -12, MONTH ) )
-- (2) put dim_product[category] on rows and dim_date[year] on columns
YoY % = DIVIDE ( [Total Sales] - [Sales LY], [Sales LY] )
-- (3)
Best Month =
VAR t = ADDCOLUMNS ( VALUES ( dim_date[month] ), "@s", [Total Sales] )
RETURN MAXX ( TOPN ( 1, t, [@s], DESC ), dim_date[month] )
-- (4) parameter table 'Discount %' with measure [Discount % Value]
Simulated Margin = DIVIDE ( [Total Sales] * ( 1 - [Discount % Value] / 100 ) - [Total
Cost], [Total Sales] * ( 1 - [Discount % Value] / 100 ) )
```

Validate (2) and (3) against pandas group-bys.

INTERVIEW CORNER Questions you may be asked

Q: What do you need for time-intelligence functions to work? A date table with continuous dates covering the data, marked as a date table, related to the fact table by date.

Q: How would you calculate YoY growth? Sales LY = CALCULATE ([Total Sales], SAMEPERIODLASTYEAR (dim_date[date])) then YoY % = DIVIDE ([Total Sales] - [Sales LY], [Sales LY]).

Q: How can users choose which metric a chart shows? Field parameters (modern) or a disconnected table with a SWITCH (SELECTEDVALUE (...), ...) measure.

Q: How do you show Top N plus an "Others" bar? A Top N measure using TOPN/KEEPFILTERS and an Others measure = total – Top N.

Practise more online

Free, trusted places to test yourself on this topic. Links open the original site in a new tab.