



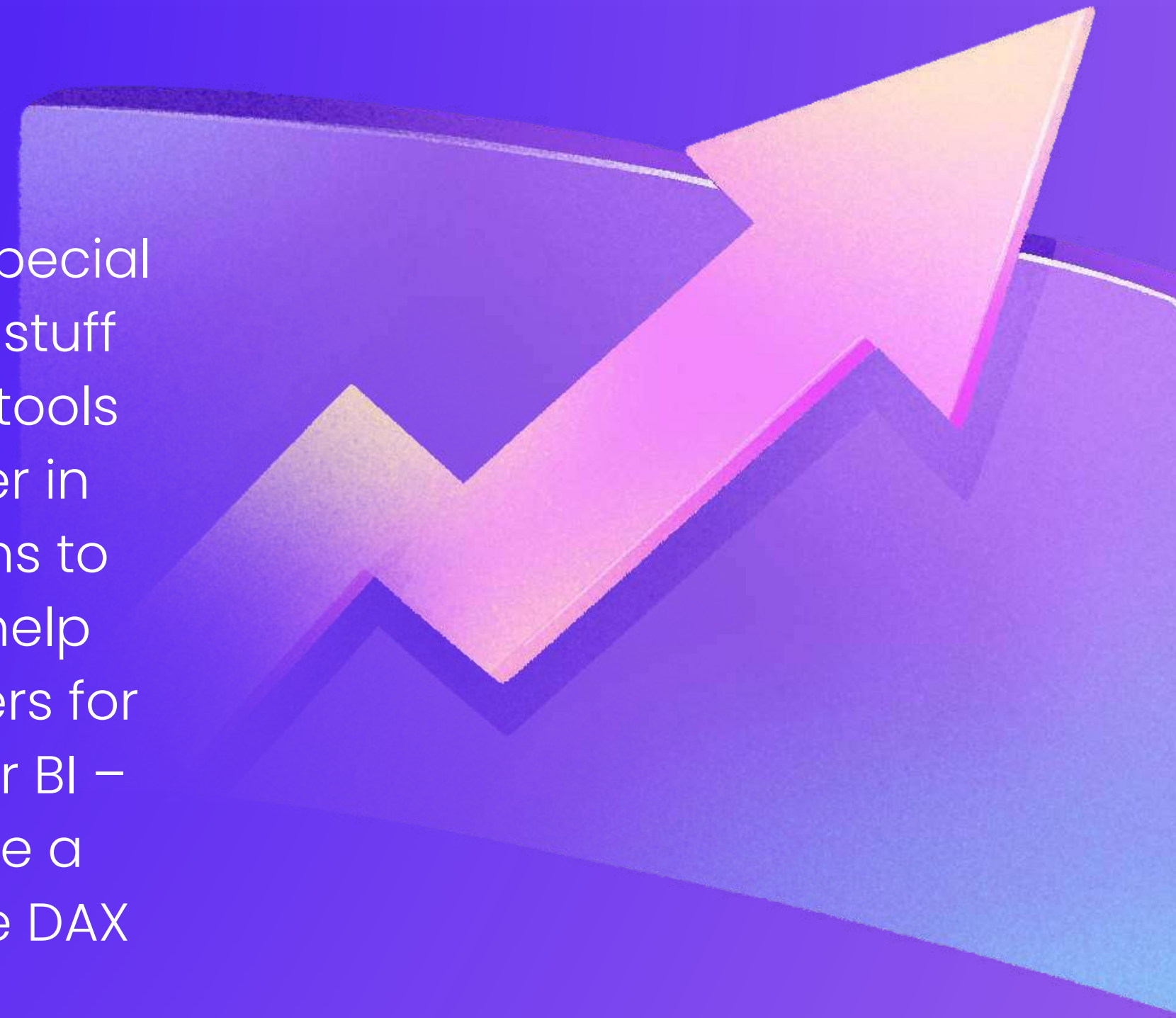
POWER BI DAX FUNCTIONS

UNDERSTANDING AND IMPLEMENTING.

By Lavender Echessa

INTRODUCTION

DAX, which stands for Data Analysis Expressions, is like a special language used in Power BI. It helps us do math and other stuff with our data to make cool reports. Imagine it as a set of tools that allows us to play with and understand our data better in Power BI. When we use DAX, we're using special instructions to tell Power BI what to do with our data. These instructions help us calculate things, group data, and create useful numbers for our reports. Think of DAX as your superhero toolkit in Power BI – it has different functions that act like superpowers. Just like a superhero needs the right powers to save the day, we use DAX functions to make our Power BI reports awesome.



CORE DAX FUNCTION

We will start by understanding the Core DAX functions which are like having a powerful set of tools at your disposal to unlock the potential of your data. These functions act as your data superheroes, performing various tasks that make your reports insightful and meaningful.



SUM

Totals the values in a column. E.g. `SUM(Sales[Revenue])` calculates total revenue.



AVERAGE

Calculates the average of a column eg. `AVERAGE(Sales[Revenue])` gets the average revenue.



MIN/MAX

Find minimum and maximum values in a column. eg `MIN(Sales[Revenue])` gets the lowest revenue.



COUNT

Counts rows in a column. eg `COUNT(Sales[Transactions])` counts transactions.

CORE DAX FUNCTION



DISTINCT COUNT
Counts distinct values
in a column.eg
`DISTINCTCOUNT(Sales[CustomerID])` counts
unique customers.



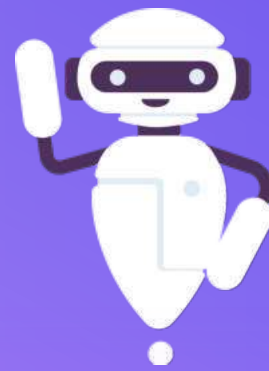
FILTER
Filters a table based
on a condition. eg
`FILTER(Sales,
Sales[Region]="West")` filters for Western
region.



CALCULATE
Evaluates an
expression in a filtered
context.
`CALCULATE(SUM(Sales[Revenue]),
Sales[Region]="West")`
sums Western revenue.

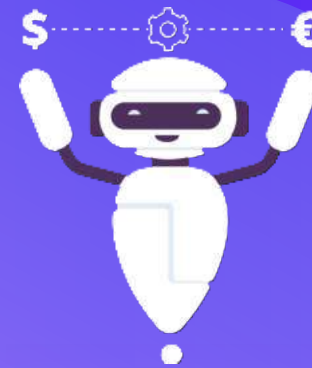
TIME INTELLIGENCE FUNCTION

These functions empower you to dive into period-over-period and moving window analytics, all driven by flexible date filters. They provide the means to unravel insights that unfold over time, making your data analysis more responsive and insightful. With these functions, you can seamlessly navigate through the temporal dimensions of your data, allowing for a deeper understanding of trends and patterns. Whether you're exploring historical trends or forecasting future possibilities,



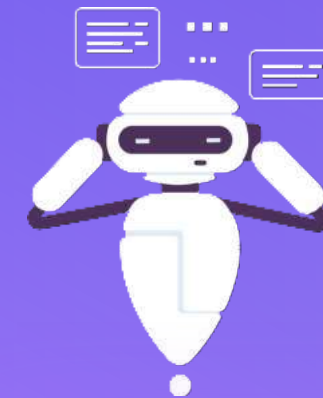
SAMEPERIODLASTYEAR

Compares to the same period last year. E.g. Calculates Western revenue last year.



DATESYTD

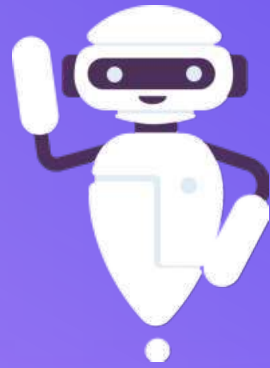
Calculates year-to-date totals.
`DATESYTD(SUM(Sales[Revenue]))` gets YTD revenue.



DATESMTD

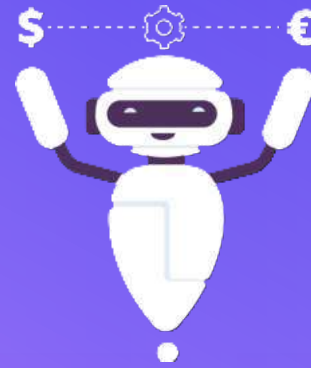
Calculates month-to-date totals.
`DATESMTD(SUM(Sales[Revenue]))` gets MTD totals.

TIME INTELLIGENCE FUNCTION



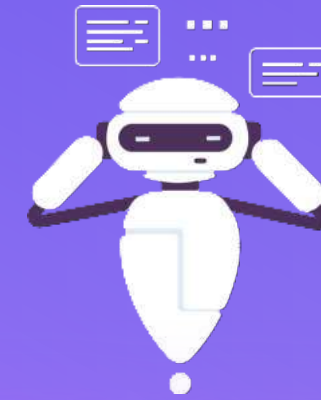
PARALLELPERIOD

Compares to a parallel period relative to a date.



OPENINGBALANCEMONTH

Calculates opening balances by month.



CLOSING BALANCE MONTH

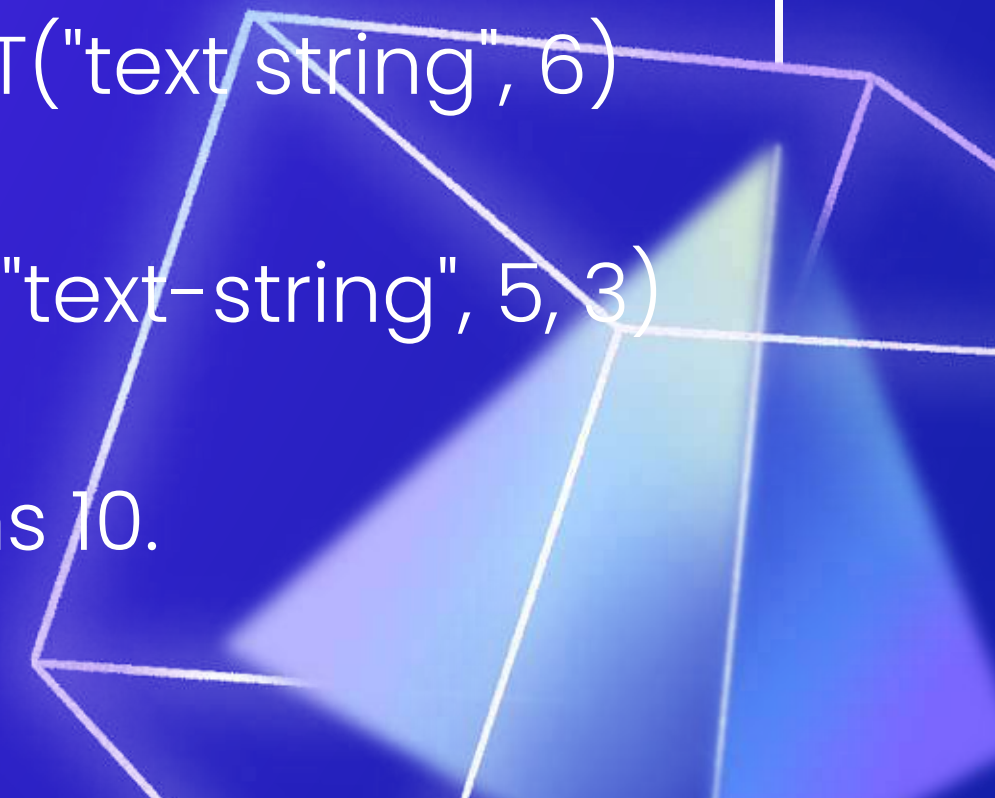
Calculates closing balances by month.



TEXT FUNCTION

Text functions in DAX provide powerful tools for manipulating and analyzing text data within Power BI. Among these functions are:

- UPPER – Converts text to upper case. E.g. UPPER("text") becomes "TEXT".
- LOWER – Converts text to lowercase. LOWER("TEXT") becomes "text".
- CONCATENATE – Joins two or more text strings. CONCATENATE("text","string") becomes "text string".
- LEFT – Returns leftmost characters of a text string. LEFT("text string", 4) returns "text".
- RIGHT – Returns the rightmost characters of a text string. RIGHT("text string", 6) returns "string".
- MID – Returns characters from the middle of a text string. MID("text-string", 5, 3) returns "tri".
- LEN – Returns the length of a text string. LEN("text string") returns 10.



LOGICAL FUNCTIONS

Logical Functions in DAX are like the decision-makers of your data, helping you implement smart conditions and choices in your Power BI reports. They enable you to introduce logic into your measures and calculated columns, making your data analysis more dynamic and responsive.



IF

IF evaluates a condition and returns one value if true and another if false. For instance, you can use it to categorize revenue as "High" or "Low" based on a threshold.

SWITCH

Meet the decision-maker that evaluates a list of values and returns the result for the first match. It's like a dynamic roadmap for assigning priorities or categories based on specific conditions.

LOGICAL FUNCTIONS

AND/OR

Logical AND/OR operators are your allies when you need to combine conditions. They help you build complex conditions, like checking if revenue is above a certain threshold in a specific region.

NOT

Logical NOT is your negator. It flips a condition, making it handy when you want to check for everything except a specific scenario. For example, identifying regions that are not "East."



IMPLEMENTING DAX FUNCTION



- Implementing DAX effectively requires a strategic approach to ensure your measures are not only accurate but also maintainable and comprehensible. Here are some elaborated best practices:

01

- Use Descriptive Measure Names:

Opt for clear and descriptive names for your measures. A name should convey the purpose of the measure, making it easy for others (or yourself) to understand its role in the analysis.

-

02

- Break Down Complex Expressions:

If you find yourself creating complex expressions, consider breaking them down into smaller, more manageable parts. This not only enhances readability but also simplifies troubleshooting and debugging.

03

- Use Variables to Simplify Logic:

Leverage DAX variables to store intermediate results or complex expressions. This not only simplifies your overall logic but also improves performance by avoiding redundant computations.

04

- Add Comments to Document Measures:

Comment on your DAX measures to provide context and explanations. This is especially crucial when dealing with intricate calculations. A well-documented measure ensures that anyone reviewing the code can quickly grasp its purpose and functionality.

05


- Validate with Test Data:

Before deploying your DAX measures into production, validate them with test data. This helps uncover any unexpected behavior or errors. Create a set of test scenarios that cover various conditions your measures may encounter.



CONCLUSION

DAX plays a pivotal role in transforming and analyzing data within Power BI. The examples covered in this overview include crucial functions such as time intelligence, text manipulation, filtering, and aggregation. Adhering to DAX best practices enables the creation of clear and sustainable measures. As you consistently apply DAX in your Power BI reporting, it will progressively become a second-nature tool, enhancing your data analysis capabilities.



THANK YOU!

