**Study Dax**

DAX functions can also be nested inside each other to perform multiple operations efficiently. This can save a lot of time when writing DAX formulas. For example, it is often useful to have multiple nested IF statements or to use the IFERROR function to wrap around another function, so that any errors in the formula are represented by the value you specify.

Some of the most common DAX functions used in reports are:

1. Simple calculations: COUNT, DISTINCTCOUNT, SUM, AVERAGE, MIN, MAX.

**Geralmente são usadas para criar cards das medidas principais no relatório.**

1. SUMMARISE: Returns a table typically used to further apply aggregations over different groupings.

**Geralmente usadas para criar novas tabelas com iformações agregadas**

1. CALCULATE: Performs an aggregation along with one or more filters. When you specify more than one filter, the function will perform the calculation where all filters are true.

**É utilizada para realizamos cálculos com medidas especificas como calcular a soma de vendas de um pais**

1. IF: Based on a logical condition, it will return a different value for if it is true or false. This is similar to the CASE WHEN operation in SQL.

**Utilizadas para operações lógicas**

1. IFERROR: Looks for any errors for an inner function and returns a specified result
2. ISBLANK: Checks if the rows in a column are blank and returns true or false. Useful to use in conjunction with other functions like IF.

**Geralmente usadas para retornar condições onde há linhas vazias**

1. EOMONTH: Returns the last day of the month of a given date (column reference in a date format) for as many months in the past or the future.
2. DATEDIFF: returns the difference between 2 dates (both as column references in date formats) in days, months, quarters, years, etc.

There are two main types of context in DAX: row context and filter context.

**Row Context**

### Row Context

This refers to just “the current row” across all columns of a table and also extends to all columns in related tables too. This type of context lets the DAX formula know which rows to use for a specific formula.

Here is an example of a formula for a calculated column that has a row context:

**Cost Price Per Unit = financials[COGS] / financials[Units Sold]**

To explicitly define a row context in a measure, you need to use a special function called an iterator**. Examples of iterator functions are SUMX, AVERAGEX, and COUNTX. These functions will first perform a calculation on a row-by-row basis,** and then perform the final aggregation on the result (ie. sum, average, count, etc.). In this way, the row context is defined explicitly by the use of these iterators.

Let’s take a look at an example of an iterator function in action:

**Average Cost Per Unit = AVERAGEX ( financials, financials[COGS] / financials[Units Sold] )**

This example performs two calculations: first, the expression is evaluated on a row-by-row basis, **and then the result is applied to the AVERAGE function.**

An alternative way of reaching this same result is to first create the calculated column **‘Cost Price Per Unit’ as we did above and then create a separate AVERAGE measure of that column**

**Filter Context**

Filter context is applied on top of a row context and refers to a subset of rows or columns that are specified as filters in the report. Filters can be applied in a few ways:

* Directly in a DAX formula
* Using the filters pane
* Using a slicer visual
* Through the fields that make up a visual (such as the rows and columns in a matrix)

A good example of how to add a filter context to a DAX formula is to use the CALCULATE function, which allows you to add one or more filter parameters to the measure. In the example below, we create a profit margin measure filtered for USA only:

No caso queremos calcular a margem de lucro apenas para os EUA então podemos usar o calculate para isso.

**USA Profit Margin =**

**CALCULATE ( SUM ( financials[Profit] ) / SUM ( financials[Sales] ),  financials[Country] = "United States of America")**

The **CALCULATE function is arguably one of Power BI's most important (and most popular)** DAX functions. It is easy to use and extremely versatile, allowing you to expand your data analyses and develop even more valuable Power BI reports.

Below is a simple example of the **CALCULATE function using SUM to find total revenue and filtering for Country = United Kingdom.** We’ll discuss this example again in more detail at the end of the tutorial.

### How does the total monthly revenue in the UK compare to all other countries?

### Para fazer esse cálculo podemos calcular o total de receita para UK e depois calcular o total de receitas para os países que não são UK e depois compara-las.

### Podemos usar essa dax para calcular UK:

**UK Revenue = CALCULATE(SUM('Online Retail'[Revenue]),**

**'Online Retail'[Country] = "United Kingdom")**

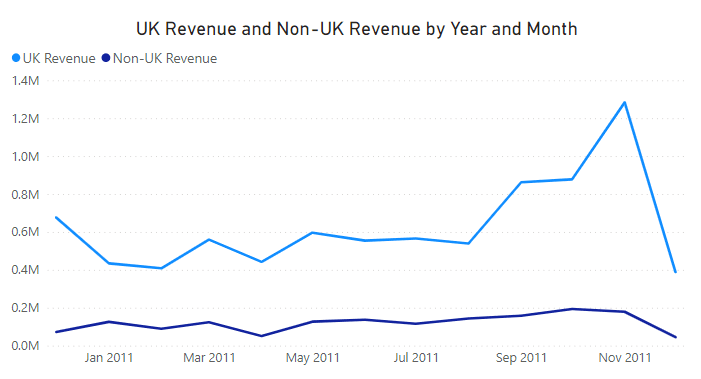
**Nesse caso nosso filtro apenas calcula o total da soma para o pais igual a United Kingdom.**

**Para os países que não são UK (países diferentes) fazemos:**

**Non-UK Revenue = CALCULATE(SUM('Online Retail'[Revenue])**

**'Online Retail'[Country] <>"United Kingdom")**

**Graficamente temos:**



### What percentage of total revenue is from the UK?

### Para responder essa pergunta temos que realizar 2 cálculos, o total de receita incluindo todos os países. Depois o total de reino unido e calcular a porcentagem.

### Nesse caso utilizamos o all para pegar de todos os países.

### Total Revenue = CALCULATE(SUM('Online Retail'[Revenue]),

### ALL('Online Retail'[Country]))

### Aqui pegamos apenas de UK e calculamos a soma de reino unido em relação ao total.

### UK % of Revenue = CALCULATE(SUM('Online Retail'[Revenue])/[Total Revenue],

### 'Online Retail'[Country] = "United Kingdom")

### 84%

Podemos fazer também de outra maneira. Calculando a diferença:

UK % of Revenue (Diferença) =

VAR TotalRevenue = CALCULATE(SUM('Online Retail'[Revenue]), ALL('Online Retail'[Country]))

VAR UKRevenue = CALCULATE(SUM('Online Retail'[Revenue]), 'Online Retail'[Country] = "United Kingdom")

RETURN

DIVIDE((TotalRevenue - UKRevenue), TotalRevenue) \* 100