

Aula 4 - Funções DAX

Docupedia Export

Author:Goncalves Donathan (SO/OPM-TS21-BR)

Date:25-Jul-2024 13:39

Table of Contents

1	4.1 - Introdução ao DAX, Medidas e Colunas Calculadas	5
1.1	Origem	5
1.2	Introdução a linguagem DAX	6
1.3	Colunas Calculadas	7
1.4	Medidas	13
1.4.1	Resumo - Colunas Calculadas x Medidas	19
1.4.2	Atenção	19
2	4.2 - Sintaxe, principais operadores e principais fórmulas	23
2.1	Sintaxe	23
2.2	Principais Operadores	23
2.2.1	Operadores Básicos:	23
2.2.2	Operadores de comparação, usados na fórmula SE:	24
2.3	Principais Fórmulas	25
3	4.3 - Fórmulas de Data e Hora	27
3.1	YEAR, MONTH e DAY	27
3.2	WEEKDAY	29
3.3	WEEKNUM	30
3.4	EOMONTH	31
3.5	DATEDIFF, TODAY, NOW	34
3.6	TODAY x NOW	35
3.7	YEARFRAC	35
3.8	HOUR, MINUTE e SECOND	35
4	4.4 - Fórmulas Lógicas	37
4.1	IF	37
4.2	IF Composto (Aninhado)	38

4.3	Função IF com várias condições - OR	40
4.4	Função IF com várias condições - AND	42
4.5	SWITCH	42
5	4.5 - Fórmulas de Texto	44
5.1	CONCATENATE	44
5.2	PROPER	48
5.3	LEFT/MID/RIGHT/UPPER/LOWER	49
5.4	Outras Funções	52
6	4.6 - Fórmulas Matemáticas e Estatísticas	54
6.1	RELATED	54
6.2	Fórmulas Matemáticas	57
6.2.1	SUM	57
6.2.2	DIVIDE	58
6.2.3	AVERAGE	61
6.2.4	MAX/MIN	62
6.3	Fórmulas de Contagem	63
6.3.1	COUNT	64
6.3.2	COUNTA	65
6.3.3	DISTINCTCOUNT	65
6.3.4	COUNTRROWS	69
6.4	Fórmulas X	69
6.4.1	SUMX	70
6.4.2	AVERAGEX	71
6.4.3	MAXX/MINX	71
6.4.4	RANKX	72
6.4.5	COUNTX	73

7	4.7 - Fórmulas de Filtro	74
7.1	CALCULATE	74
7.2	ALL	76
7.3	ALLEXCEPT	80
7.4	FILTER	81
7.5	VALUES	83
7.6	HASONEVALUE	83
7.7	HASONEFILTER	83
7.8	ISFILTERED	83
7.9	SELECTEDVALUE	83
8	4.8 - Fórmulas de Inteligência de Tempo	84
8.1	As fórmulas de inteligência de tempo no Power BI são um conjunto de funções e recursos que permitem realizar análises e cálculos relacionados a datas e horários. Elas ajudam a trabalhar com dados de tempo de forma mais eficiente e fornecem insights temporais em relatórios e visualizações.	84
8.2	DATESYTD	84
8.3	DATEADD	90
8.4	DATESINPERIOD	92
9	4.9 - Organizando Medidas, Resumo das Principais Fórmulas e Boas Práticas	95
9.1	Organizando Medidas	95
9.2	Fórmulas de Texto	100
9.3	Fórmulas Matemáticas	102
9.4	Fórmulas de Contagem	103
9.5	Fórmulas de Inteligência de Tempo	104
9.6	Aula 4 - Boas Práticas	106

1 4.1 - Introdução ao DAX, Medidas e Colunas Calculadas

Importamos os dados, fizemos todos os tratamentos e relacionamentos necessários. Agora chegou o momento de começar a aplicar fórmulas inteligentes nesses dados, para analisar lucros, tendências e etc. usando as **Fórmulas DAX**.

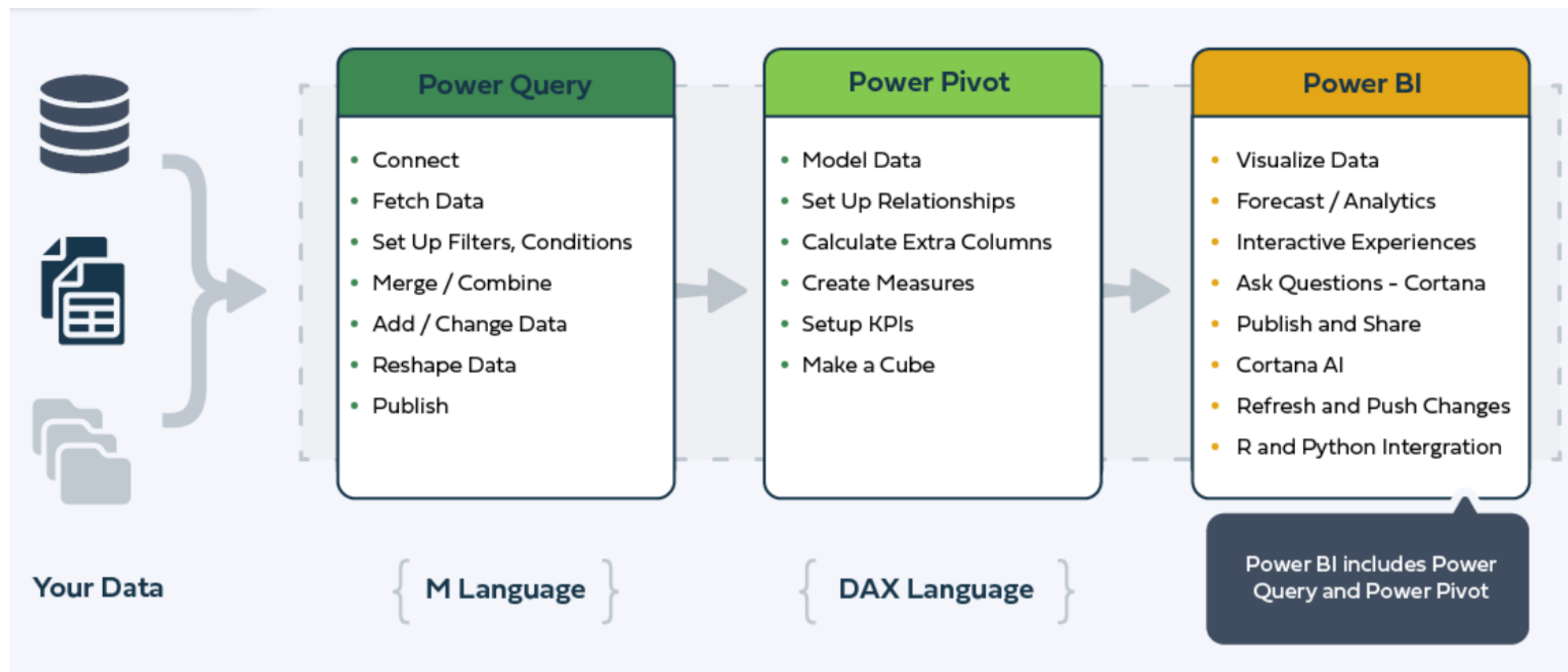
1.1 Origem

A linguagem **DAX** (**Data Analysis Expressions - Expressões para Análise de Dados**) surgiu em 2009 como uma poderosa linguagem de fórmula desenvolvida pela Microsoft. Ela foi introduzida juntamente com o Power Pivot para o Excel 2010.

Ela foi criada para permitir análise e manipulação de dados em modelos de dados tabulares - estrutura semelhante a uma tabela de banco de dados ou uma planilha do Excel. Sua sintaxe é semelhante ao Excel, mas oferece recursos avançados específicos para sua área de aplicação, que nos permitem criar lógicas mais inteligentes de maneira mais simples e intuitiva.

A linguagem DAX é usada principalmente no Power BI, SQL Server Analysis Services (SSAS) e Power Pivot no Excel. Ela permite a criação de medidas calculadas, colunas calculadas, tabelas e consultas para realizar operações como agregações, filtrações e cálculos condicionais.

Diagrama de relação: Power Query, Power Pivot e Power BI



1.2 Introdução a linguagem DAX

A linguagem tem basicamente dois objetivos:

- Permitir adicionar **Colunas Calculadas** e **Medidas** ao nosso modelo, com sintaxes intuitivas;
- A criação de fórmulas inteligentes para ajudar nos nossos diversos tipos de análises.

No geral as fórmulas DAX funcionam de maneira bem parecida com as fórmulas do Excel, sendo muitas delas aproveitadas dentro do Power BI. Vamos ver agora o que são essas medidas e colunas que são geradas com as nossas fórmulas:

1.3 Colunas Calculadas

O objetivo das **Colunas Calculadas** é a criação de uma fórmula para acrescentar novas colunas as nossas tabelas. Para entender, vamos voltar na nossa base de vendas:

SKU	Tamanho Pedido	Loja	Data da Venda	Código Cliente	Tipo do Pedido
TE10019	4	Vila Velha	quarta-feira, 1 de janeiro de 2020	239	Pedido Múltiplo
TE10018	3	Curitiba	quarta-feira, 1 de janeiro de 2020	234	Pedido Múltiplo
TE10005	3	Fortaleza	quarta-feira, 1 de janeiro de 2020	567	Pedido Múltiplo
TE10009	3	Cuiabá	quarta-feira, 1 de janeiro de 2020	588	Pedido Múltiplo
TE10016	3	Fortaleza	quarta-feira, 1 de janeiro de 2020	114	Pedido Múltiplo
TE10006	5	Brasília	quarta-feira, 1 de janeiro de 2020	222	Pedido Múltiplo
TE10016	5	Brasília	quarta-feira, 1 de janeiro de 2020	401	Pedido Múltiplo
TE10021	5	Recife	quarta-feira, 1 de janeiro de 2020	576	Pedido Múltiplo
TE10004	5	Porto Alegre	quarta-feira, 1 de janeiro de 2020	766	Pedido Múltiplo
TE10016	5	Belo Horizonte	quarta-feira, 1 de janeiro de 2020	810	Pedido Múltiplo
TE10018	5	Salvador	quarta-feira, 1 de janeiro de 2020	258	Pedido Múltiplo
TE10011	5	Vila Velha	quarta-feira, 1 de janeiro de 2020	28	Pedido Múltiplo
TE10024	5	Cuiabá	quarta-feira, 1 de janeiro de 2020	59	Pedido Múltiplo
TE10022	3	Porto Alegre	quarta-feira, 1 de janeiro de 2020	260	Pedido Múltiplo
TE10001	2	Cuiabá	quarta-feira, 1 de janeiro de 2020	704	Pedido Múltiplo
TE10024	2	Florianópolis	quarta-feira, 1 de janeiro de 2020	370	Pedido Múltiplo
TE10008	2	Curitiba	quarta-feira, 1 de janeiro de 2020	444	Pedido Múltiplo
TE10021	2	Brasília	quarta-feira, 1 de janeiro de 2020	342	Pedido Múltiplo
TE10011	2	Porto Alegre	quarta-feira, 1 de janeiro de 2020	41	Pedido Múltiplo
TE10008	2	Salvador	quarta-feira, 1 de janeiro de 2020	557	Pedido Múltiplo

A coluna **Tipo do Pedido** não existia inicialmente na nossa base de vendas, nós que acrescentamos essa coluna no Power Query usando a ferramenta de **Coluna Condicional**. A lógica basicamente foi a seguinte:

- Se Tamanho do Pedido = 1: "Pedido Único";
- Se Tamanho do Pedido > 1: "Pedido Múltiplo".

O que fizemos na verdade foi utilizar a lógica SE por trás da ferramenta, de uma maneira que nem foi preciso se preocupar com as fórmulas. Como estamos entrando no módulo de fórmulas, nada mais justo do que refazer essa coluna usando efetivamente a fórmula SE para isso!

Vamos começar excluindo a coluna Tipo do Pedido: basta clicar no nome dela com o botão direito e clicar na opção **Excluir (Delete)**:

SKU	Tamanho Pedido	Loja	Data da Venda	Código Cliente	Tipo do Pedido
TE10019	4	Vila Velha	quarta-feira, 1 de janeiro de 2020	239	Pedido M
TE10018	3	Curitiba	quarta-feira, 1 de janeiro de 2020	234	Pedido M
TE10005	3	Fortaleza	quarta-feira, 1 de janeiro de 2020	567	Pedido M
TE10009	3	Cuiabá	quarta-feira, 1 de janeiro de 2020	588	Pedido M
TE10016	3	Fortaleza	quarta-feira, 1 de janeiro de 2020	114	Pedido M
TE10006	5	Brasília	quarta-feira, 1 de janeiro de 2020	222	Pedido M
TE10016	5	Brasília	quarta-feira, 1 de janeiro de 2020	401	Pedido M
TE10021	5	Recife	quarta-feira, 1 de janeiro de 2020	576	Pedido M
TE10004	5	Porto Alegre	quarta-feira, 1 de janeiro de 2020	766	Pedido M
TE10016	5	Belo Horizonte	quarta-feira, 1 de janeiro de 2020	810	Pedido M
TE10018	5	Salvador	quarta-feira, 1 de janeiro de 2020	258	Pedido M
TE10011	5	Vila Velha	quarta-feira, 1 de janeiro de 2020	28	Pedido M
TE10024	5	Cuiabá	quarta-feira, 1 de janeiro de 2020	59	Pedido M
TE10022	3	Porto Alegre	quarta-feira, 1 de janeiro de 2020	260	Pedido M
TE10001	2	Cuiabá	quarta-feira, 1 de janeiro de 2020	704	Pedido M
TE10024	2	Florianópolis	quarta-feira, 1 de janeiro de 2020	370	Pedido M
TE10008	2	Curitiba	quarta-feira, 1 de janeiro de 2020	444	Pedido M
TE10021	2	Brasília	quarta-feira, 1 de janeiro de 2020	342	Pedido M
TE10011	2	Porto Alegre	quarta-feira, 1 de janeiro de 2020	41	Pedido M
TE10008	2	Salvador	quarta-feira, 1 de janeiro de 2020	557	Pedido Múltiplo

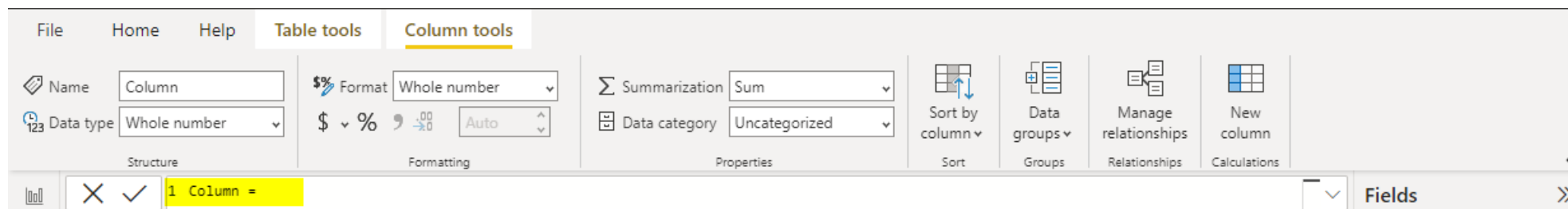
Feito isso, agora vamos criar uma fórmula seguindo a mesma lógica utilizada na nossa coluna condicional.

Para acrescentar uma nova coluna clicamos com o botão direito do mouse em qualquer parte da tabela e selecionar a opção **Nova Coluna** (New Column):

SKU	Tamanho Pedido	Loja	Data da Venda	Código
TE10019	4	Vila Velha	quarta-feira, 1 de janeiro de 2020	
TE10018	3	Curitiba	quarta-feira, 1 de janeiro de 2020	
TE10005	3	Fortaleza	quarta-feira, 1 de janeiro de 2020	
TE10009	3	Cuiabá	quarta-feira, 1 de janeiro de 2020	
TE10016	3	Fortaleza	quarta-feira, 1 de janeiro de 2020	
TE10006	5	Brasília	quarta-feira, 1 de janeiro de 2020	
TE10016	5	Brasília	quarta-feira, 1 de janeiro de 2020	
TE10021	5	Recife	quarta-feira, 1 de janeiro de 2020	
TE10004	5	Porto Alegre	quarta-feira, 1 de janeiro de 2020	
TE10016	5	Belo Horizonte	quarta-feira, 1 de janeiro de 2020	
TE10018	5	Salvador	quarta-feira, 1 de janeiro de 2020	
TE10011	5	Vila Velha	quarta-feira, 1 de janeiro de 2020	
TE10024	5	Cuiabá	quarta-feira, 1 de janeiro de 2020	
TE10022	3	Porto Alegre	quarta-feira, 1 de janeiro de 2020	
TE10001	2	Cuiabá	quarta-feira, 1 de janeiro de 2020	
TE10024	2	Florianópolis	quarta-feira, 1 de janeiro de 2020	
TE10008	2	Curitiba	quarta-feira, 1 de janeiro de 2020	
TE10021	2	Brasília	quarta-feira, 1 de janeiro de 2020	
TE10011	2	Porto Alegre	quarta-feira, 1 de janeiro de 2020	
TE10008	2	Salvador	quarta-feira, 1 de janeiro de 2020	
				557

- Sort ascending
- Sort descending
- Clear sort
- Clear filter
- Clear all filters
- Copy
- Copy table
- New measure
- New column**
- Refresh data
- Edit query
- Rename
- Delete
- ✓ Hide in report view
- Unhide all
- New group

Quando clicarmos na opção a barra de fórmulas será marcada, inicialmente com o texto "Coluna =":



É nessa barra de fórmulas que vamos criar a fórmula para preencher a nossa coluna. Sempre que criarmos uma fórmula no Power BI, ela vai seguir a seguinte estrutura:

Nome da Coluna = Fórmula

Vamos começar alterando o nome da coluna para **Tipo do Pedido**:

1 Tipo do Pedido =					
SKU	Tamanho Pedido	Loja	Data da Venda	Código Cliente	Tipo do Pedido
TE10019	4	Vila Velha	quarta-feira, 1 de janeiro de 2020	239	
TE10018	3	Curitiba	quarta-feira, 1 de janeiro de 2020	234	
TE10005	3	Fortaleza	quarta-feira, 1 de janeiro de 2020	567	

Como podemos ver, o Power BI já criou a nossa coluna. Como ainda não temos a fórmula escrita, ela está vazia. Então, logo depois do sinal de igual vamos inserir a nossa expressão. Como vimos anteriormente, usaremos a fórmula **SE**. No Power BI todas as fórmulas são escritas em **inglês**. Portanto, vamos nos referir à fórmula **SE** como **IF**.

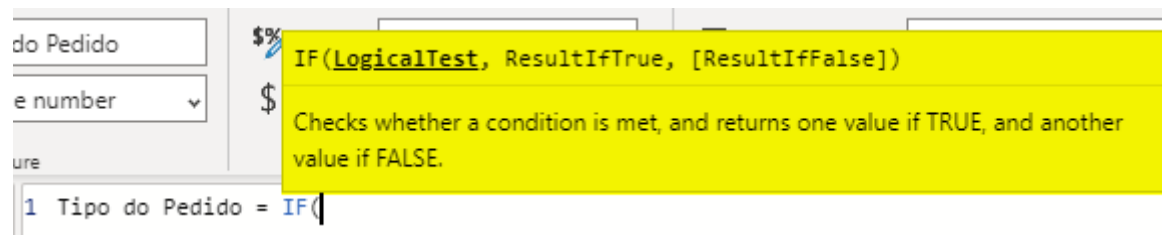
Diferente do Excel, aqui não iremos fazer comparações entre uma célula e outra, mas sim de toda a coluna. **As fórmulas no Power BI não são aplicadas apenas à uma célula, e sim à uma coluna inteira ou à uma tabela inteira de uma só vez.**

Quando começamos a escrever o nome da fórmula, o Power BI lista todas as fórmulas possíveis de acordo com o que digitamos. Para simplificar o trabalho, podemos selecioná-la e apertar tanto a tecla **Tab** quanto o **Enter**:

The screenshot shows the Power BI formula bar with the formula `1 Tipo do Pedido = if`. A dropdown menu is open, listing the following functions: `fx IF`, `fx IF.EAGER`, `fx IFERROR`, `fx ISFILTERED`, `fx ISCROSSFILTERED`, `fx ISAFter`, and `fx ISONORAFTER`. Below the formula bar, a table is visible with columns for SKU, Tamanho Pedido, and Location. The data rows are:

SKU	Tamanho Pedido	Localização
TE10019	4	Vila Velha
TE10018	3	Curitiba
TE10005	3	Fortaleza
TE10009	3	Cuiabá
TE10016	3	Fortaleza
TE10006	5	Brasília

Assim como no Excel, quando selecionamos uma fórmula ele nos apresenta uma caixa com as informações referentes à essa fórmula:



Importante: no Power BI, a separação dos argumentos das fórmulas - as informações necessárias para a fórmula ser executada da maneira correta - pode variar. Na versão na qual o curso foi feito, a separação é feita por **vírgula (,)**; em outras versões a separação pode ser por **ponto e vírgula (;)**. Quando não souber qual usar, basta ter atenção a indicação que o próprio Power BI apresenta.

A função **IF** verifica a condição de uma expressão lógica e retorna um valor caso a expressão seja verdadeira e outro caso a expressão seja falsa. Na nossa coluna, faremos isso da seguinte forma:

Condição/Teste Lógico (Logical Test)	Resultado se Verdadeira (Result if True)	Resultado se Falsa (Result if False)
Tamanho do pedido > 1	"Pedido Múltiplo"	"Pedido Único"

Como dito antes, no Power BI não vamos criar comparação entre duas células, mas sim analisar uma coluna inteira. Sabendo disso, a fórmula a ser usada é a seguinte:

```
1 Tipo do Pedido = IF(TotalVendas[Tamanho Pedido]>1, "Pedido Múltiplo", "Pedido Único")
```

Basicamente, o que estamos falando é que se o tamanho do pedido for maior que 1 a resposta da fórmula deve ser "Pedido Múltiplo", e caso contrário "Pedido Único". E pronto! Criamos a nossa coluna de **Tipo do Pedido**!

1 Tipo do Pedido = IF(TotalVendas[Tamanho Pedido]>1, "Pedido Múltiplo", "Pedido Único")						
SKU	Tamanho Pedido	Loja	Data da Venda	Código Cliente	Tipo do Pedido	
TE10019	4	Vila Velha	quarta-feira, 1 de janeiro de 2020	239	Pedido Múltiplo	
TE10018	3	Curitiba	quarta-feira, 1 de janeiro de 2020	234	Pedido Múltiplo	
TE10005	3	Fortaleza	quarta-feira, 1 de janeiro de 2020	567	Pedido Múltiplo	
TE10009	3	Cuiabá	quarta-feira, 1 de janeiro de 2020	588	Pedido Múltiplo	
TE10016	3	Fortaleza	quarta-feira, 1 de janeiro de 2020	114	Pedido Múltiplo	
TE10006	5	Brasília	quarta-feira, 1 de janeiro de 2020	222	Pedido Múltiplo	
TE10016	5	Brasília	quarta-feira, 1 de janeiro de 2020	401	Pedido Múltiplo	
TE10021	5	Recife	quarta-feira, 1 de janeiro de 2020	576	Pedido Múltiplo	
TE10004	5	Porto Alegre	quarta-feira, 1 de janeiro de 2020	766	Pedido Múltiplo	
TE10016	5	Belo Horizonte	quarta-feira, 1 de janeiro de 2020	810	Pedido Múltiplo	
TE10018	5	Salvador	quarta-feira, 1 de janeiro de 2020	258	Pedido Múltiplo	
TE10011	5	Vila Velha	quarta-feira, 1 de janeiro de 2020	28	Pedido Múltiplo	
TE10024	5	Cuiabá	quarta-feira, 1 de janeiro de 2020	59	Pedido Múltiplo	
TE10022	3	Porto Alegre	quarta-feira, 1 de janeiro de 2020	260	Pedido Múltiplo	
TE10001	2	Cuiabá	quarta-feira, 1 de janeiro de 2020	704	Pedido Múltiplo	
TE10024	2	Florianópolis	quarta-feira, 1 de janeiro de 2020	370	Pedido Múltiplo	
TE10008	2	Curitiba	quarta-feira, 1 de janeiro de 2020	444	Pedido Múltiplo	
TE10021	2	Brasília	quarta-feira, 1 de janeiro de 2020	342	Pedido Múltiplo	
TE10011	2	Porto Alegre	quarta-feira, 1 de janeiro de 2020	41	Pedido Múltiplo	
TE10008	2	Salvador	quarta-feira, 1 de janeiro de 2020	557	Pedido Múltiplo	

1.4 Medidas

Acabamos de ver como funciona uma coluna calculada - cada linha da coluna criada possui uma informação com base nos outros dados da linha. E a **medida**? Qual a sua função?

Quando queremos um **valor único** como resposta, como a soma total de produtos vendidos, não faz sentido usarmos uma coluna. Vamos ver na prática um exemplo:

Vamos imaginar que agora que temos as informações sobre o tipo dos pedidos queremos criar uma matriz com a soma de vendas de cada um. Podemos criar uma coluna chamada Total Pedidos e usar a fórmula **Soma** (*Sum*) para somar todos os valores da coluna Tamanho do Pedido:

```
1 Total Pedidos = SUM(TotalVendas[Tamanho Pedido])
```

O resultado que obtemos aparece abaixo:

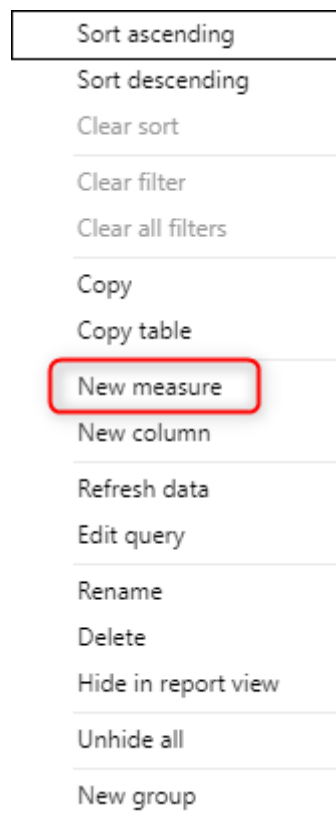
SKU	Tamanho Pedido	Loja	Data da Venda	Código Cliente	Tipo do Pedido	Total Pedidos
TE10019	4	Vila Velha	quarta-feira, 1 de janeiro de 2020	239	Pedido Múltiplo	71442
TE10018	3	Curitiba	quarta-feira, 1 de janeiro de 2020	234	Pedido Múltiplo	71442
TE10005	3	Fortaleza	quarta-feira, 1 de janeiro de 2020	567	Pedido Múltiplo	71442
TE10009	3	Cuiabá	quarta-feira, 1 de janeiro de 2020	588	Pedido Múltiplo	71442
TE10016	3	Fortaleza	quarta-feira, 1 de janeiro de 2020	114	Pedido Múltiplo	71442
TE10006	5	Brasília	quarta-feira, 1 de janeiro de 2020	222	Pedido Múltiplo	71442
TE10016	5	Brasília	quarta-feira, 1 de janeiro de 2020	401	Pedido Múltiplo	71442
TE10021	5	Recife	quarta-feira, 1 de janeiro de 2020	576	Pedido Múltiplo	71442
TE10004	5	Porto Alegre	quarta-feira, 1 de janeiro de 2020	766	Pedido Múltiplo	71442
TE10016	5	Belo Horizonte	quarta-feira, 1 de janeiro de 2020	810	Pedido Múltiplo	71442
TE10018	5	Salvador	quarta-feira, 1 de janeiro de 2020	258	Pedido Múltiplo	71442
TE10011	5	Vila Velha	quarta-feira, 1 de janeiro de 2020	28	Pedido Múltiplo	71442
TE10024	5	Cuiabá	quarta-feira, 1 de janeiro de 2020	59	Pedido Múltiplo	71442
TE10022	3	Porto Alegre	quarta-feira, 1 de janeiro de 2020	260	Pedido Múltiplo	71442
TE10001	2	Cuiabá	quarta-feira, 1 de janeiro de 2020	704	Pedido Múltiplo	71442
TE10024	2	Florianópolis	quarta-feira, 1 de janeiro de 2020	370	Pedido Múltiplo	71442
TE10008	2	Curitiba	quarta-feira, 1 de janeiro de 2020	444	Pedido Múltiplo	71442
TE10021	2	Brasília	quarta-feira, 1 de janeiro de 2020	342	Pedido Múltiplo	71442
TE10011	2	Porto Alegre	quarta-feira, 1 de janeiro de 2020	41	Pedido Múltiplo	71442
TE10008	2	Salvador	quarta-feira, 1 de janeiro de 2020	557	Pedido Múltiplo	71442

Na nossa nova coluna, ele repetiu o valor 71442 para todas as linhas, o que não faz muito sentido. Vimos que dentro de uma coluna calculada, a fórmula avalia um valor linha a linha e retorna um resultado com base nesse valor, o que fez sentido na criação da coluna com o tipo do pedido; entretanto, como ao realizar a soma nós pegamos todos os valores de uma coluna e transformamos em um único valor, não faz sentido criar uma coluna para armazenar isso.

É aqui que entram as **Medidas**: elas guardam na memória para nós um único valor, também chamado de **agregamento**, que não aparece efetivamente em nenhuma tabela mas pode ser usado futuramente nos nossos relatórios para demonstrar esses valores.

Como nossa coluna não faz sentido, vamos excluí-la clicando com o botão direito na coluna e indo na opção **Excluir (Delete)**.

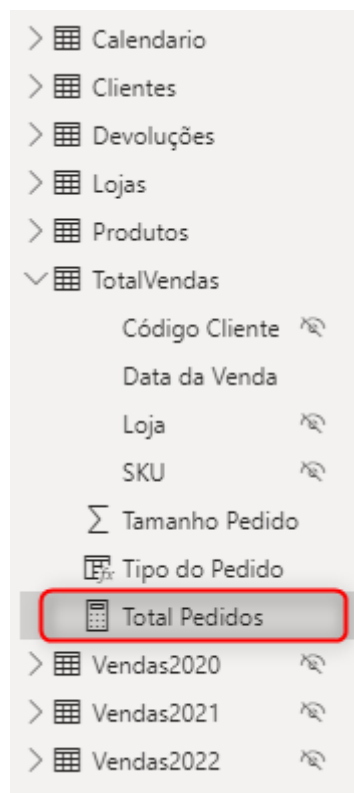
Feito isso, vamos criar uma Medida: o processo é idêntico a criação de uma coluna, com a diferença que na hora de selecionar a opção após ter clicado na tabela com o botão direito escolhemos **Nova Medida (New Measure)**:



Mais uma vez será criada uma fórmula na barra de fórmulas, com o nome "Medida =". Vamos chamá-la de **Total Pedidos** e utilizar a fórmula **SUM** que usamos anteriormente:

```
1 Total Pedidos = SUM(TotalVendas[Tamanho Pedido])
```

A fórmula é exatamente a mesma que usamos anteriormente, porém agora para a criação de uma medida e não mais uma coluna. Não foi criada uma nova coluna na tabela, mas sim uma nova medida chamada **Total Pedidos**:

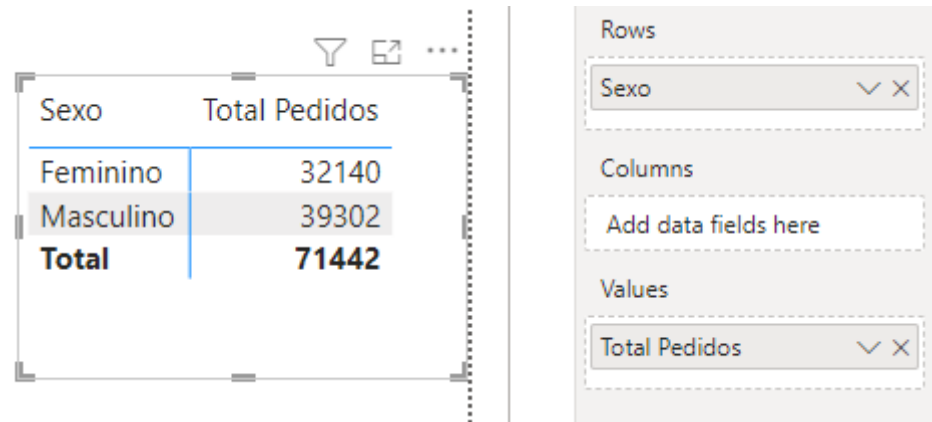


O resultado da conta realizada nessa medida está armazenado na memória do nosso Power BI. Para visualizar o resultado, precisamos voltar para o nosso relatório. Vamos criar uma matriz e arrastar o campo **Total Pedidos** para Valores:

Total Pedidos	
	71442

A princípio ele vai nos mostrar apenas a soma dos valores, que é o mesmo 71442 que vimos anteriormente. Aqui entra outra vantagem das Medidas, que é a possibilidade de aplicar filtros livremente para subdividir o nosso resultado.

Por exemplo, se quisermos saber o total de produtos vendidos de acordo com o gênero da pessoa, podemos arrastar o campo Sexo para a linha da matriz:



Sexo	Total Pedidos
Feminino	32140
Masculino	39302
Total	71442

Rows

Sexo

Columns

Add data fields here

Values

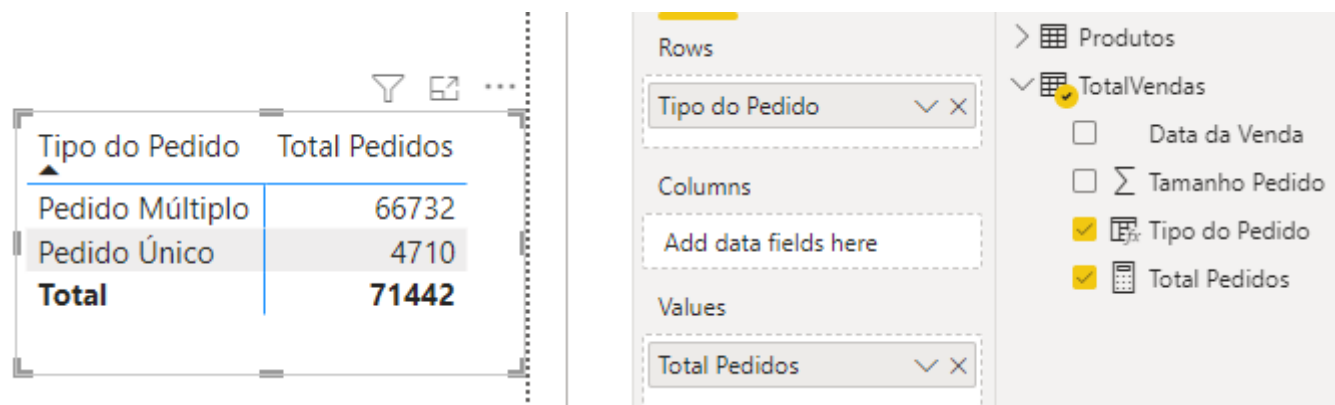
Total Pedidos

Ao fazermos isso, ele consegue dividir os valores de acordo com Masculino e Feminino. Podemos aplicar qualquer filtro na nossa matriz e a medida vai conseguir separar os valores de acordo com cada um dos critérios. Vamos ver outro exemplo, usando a coluna das lojas agora:

Loja	Total Pedidos
Belo Horizonte	5070
Brasília	4948
Campinas	4950
Cuiabá	9895
Curitiba	5080
Florianópolis	5273
Fortaleza	5267
Porto Alegre	5159
Recife	5149
Rio de Janeiro	5250
Salvador	5242
São Paulo	4957
Vila Velha	5202
Total	71442

Podemos ver que as possibilidades são infinitas, e tudo isso criando apenas uma única medida.

Para fechar, vamos ver uma diferença sutil de como utilizamos **Colunas Calculadas** e **Medidas** nos nossos relatórios: geralmente, a Coluna Calculada vai entrar nas Linhas de uma matriz no relatório, enquanto as Medidas vão entrar nos Valores, se adaptando aos critérios da matriz:



The image shows a Power BI report. On the left is a table with two columns: 'Tipo do Pedido' and 'Total Pedidos'. The table has three rows: 'Pedido Múltiplo' with a value of 66732, 'Pedido Único' with a value of 4710, and a 'Total' row with a value of 71442. On the right is the 'Fields' pane. The 'Rows' shelf contains 'Tipo do Pedido'. The 'Columns' shelf is empty with the text 'Add data fields here'. The 'Values' shelf contains 'Total Pedidos'. In the background, the 'TotalVendas' measure is expanded, showing 'Data da Venda' (unchecked), 'Tamanho Pedido' (unchecked), 'Tipo do Pedido' (checked), and 'Total Pedidos' (checked).

Tipo do Pedido	Total Pedidos
Pedido Múltiplo	66732
Pedido Único	4710
Total	71442

1.4.1 Resumo - Colunas Calculadas x Medidas

Colunas Calculadas	Medidas
Os valores são calculados baseados na informação de cada linha da tabela (são calculados individualmente para cada linha, chamado de " Contexto de Linha ")	Os valores são calculados baseados nos filtros aplicados nas Matrizes (chamado de " Contexto de Filtro ")
Acrescenta uma nova coluna na tabela	Não acrescenta novas colunas nas tabelas
Recalcula de acordo com alterações nas células	Recalcula de acordo com alterações nos filtros do Relatório (em uma Matriz ou Gráfico)
Geralmente usado nos campos das Linhas, Colunas ou Filtros nas Matrizes/Gráficos	Quase sempre usado no campo Valores de uma Matriz/Gráfico

1.4.2 Atenção

O Power Query é responsável pela etapa de carregamento e transformação dos dados antes de serem importados para o modelo de dados do Power BI. As alterações feitas no Power Query, como filtragem, combinação de tabelas ou criação de colunas calculadas, afetam a forma como os dados são preparados e estruturados para análise no Power BI.

Por outro lado, a linguagem DAX é usada para criar cálculos e medidas personalizadas no modelo de dados do Power BI. Ela é aplicada **após** os dados serem carregados e transformados pelo Power Query; portanto, **as alterações feitas usando DAX**, como a criação de medidas calculadas, colunas calculadas ou tabelas virtuais, são específicas para o modelo de dados do Power BI e **não afetam o processo de carregamento e transformação no Power Query**.

Portanto, é importante entender que o Power Query e a linguagem DAX são duas etapas distintas e complementares no processo de preparação e análise de dados no Power BI. O Power Query é usado para carregar e transformar os dados, enquanto a linguagem DAX é usada para criar cálculos e medidas personalizadas no modelo de dados.

Por isso, quando voltamos para o Power Query as tabelas e medidas que criamos não estão presentes lá, pois elas fazem parte de uma etapa posterior. A única mudança verificada foi que ao apagar uma coluna no Power BI, ela é removida no Power Query também.

Tabela atual no editor do Power Query:

✕ ✓ fx

= Table.RemoveColumns(#"Changed Type1",{"Tipo do Pedido"})

	A ^B _C SKU	1 ² ₃ Tamanho Pedido	A ^B _C Loja	Data da Venda	1 ² ₃ Código Cliente
1	TE10021	5	Recife	01/01/2020	576
2	TE10021	2	Brasília	01/01/2020	342
3	TE10019	1	Cuiabá	01/01/2020	437
4	TE10023	1	Florianópolis	01/01/2020	809
5	TE10016	5	Belo Horizonte	01/01/2020	810
6	TE10022	4	Campinas	01/01/2020	91
7	TE10001	2	Cuiabá	01/01/2020	704
8	TE10005	3	Fortaleza	01/01/2020	567
9	TE10011	2	Porto Alegre	01/01/2020	41
10	TE10015	4	Porto Alegre	01/01/2020	626
11	TE10006	4	Salvador	01/01/2020	186
12	TE10004	5	Porto Alegre	01/01/2020	766
13	TE10006	5	Brasília	01/01/2020	222
14	TE10018	5	Salvador	01/01/2020	258
15	TE10003	4	Belo Horizonte	01/01/2020	696
16	TE10011	4	Cuiabá	01/01/2020	458
17	TE10009	3	Cuiabá	01/01/2020	588
18	TE10016	5	Brasília	01/01/2020	401
19	TE10016	3	Fortaleza	01/01/2020	114
20	TE10015	4	Porto Alegre	01/01/2020	346

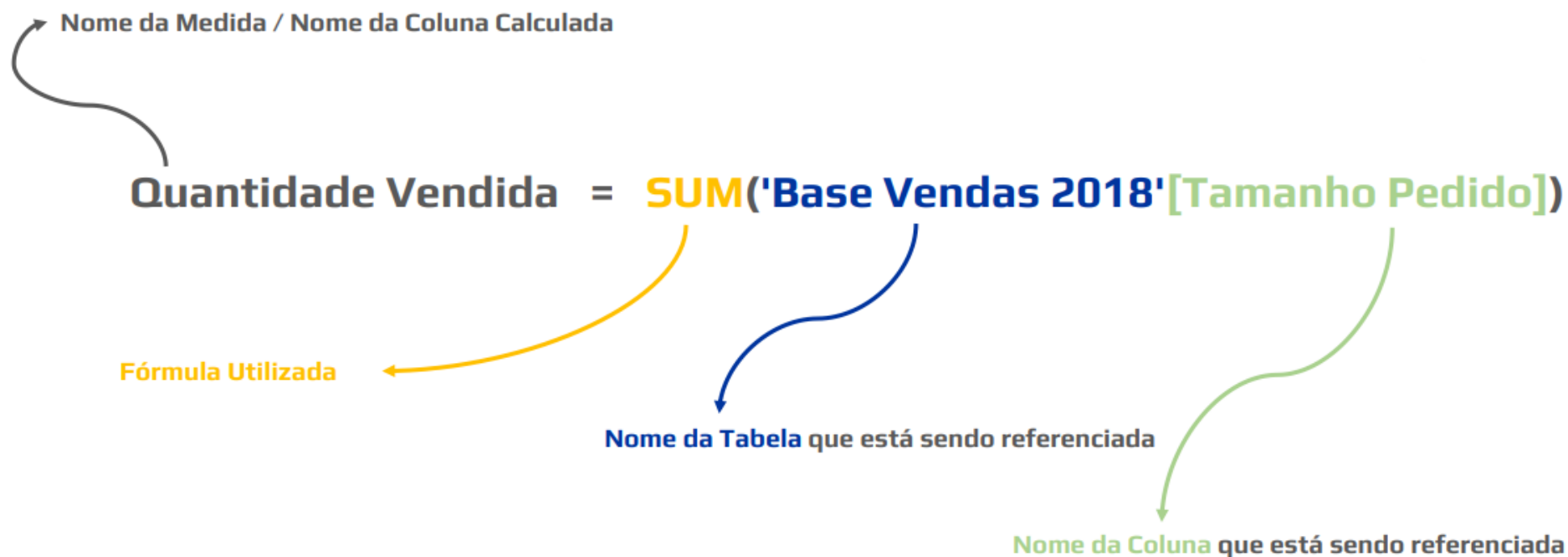
Tabela atual no Power BI:

SKU ▼	Tamanho Pedido ▼	Loja ▼	Data da Venda ▼↑	Código Cliente ▼	Tipo do Pedido ▼
TE10019	4	Vila Velha	quarta-feira, 1 de janeiro de 2020	239	Pedido Múltiplo
TE10018	3	Curitiba	quarta-feira, 1 de janeiro de 2020	234	Pedido Múltiplo
TE10005	3	Fortaleza	quarta-feira, 1 de janeiro de 2020	567	Pedido Múltiplo
TE10009	3	Cuiabá	quarta-feira, 1 de janeiro de 2020	588	Pedido Múltiplo
TE10016	3	Fortaleza	quarta-feira, 1 de janeiro de 2020	114	Pedido Múltiplo
TE10006	5	Brasília	quarta-feira, 1 de janeiro de 2020	222	Pedido Múltiplo
TE10016	5	Brasília	quarta-feira, 1 de janeiro de 2020	401	Pedido Múltiplo
TE10021	5	Recife	quarta-feira, 1 de janeiro de 2020	576	Pedido Múltiplo
TE10004	5	Porto Alegre	quarta-feira, 1 de janeiro de 2020	766	Pedido Múltiplo
TE10016	5	Belo Horizonte	quarta-feira, 1 de janeiro de 2020	810	Pedido Múltiplo
TE10018	5	Salvador	quarta-feira, 1 de janeiro de 2020	258	Pedido Múltiplo
TE10011	5	Vila Velha	quarta-feira, 1 de janeiro de 2020	28	Pedido Múltiplo
TE10024	5	Cuiabá	quarta-feira, 1 de janeiro de 2020	59	Pedido Múltiplo
TE10022	3	Porto Alegre	quarta-feira, 1 de janeiro de 2020	260	Pedido Múltiplo
TE10001	2	Cuiabá	quarta-feira, 1 de janeiro de 2020	704	Pedido Múltiplo
TE10024	2	Florianópolis	quarta-feira, 1 de janeiro de 2020	370	Pedido Múltiplo
TE10008	2	Curitiba	quarta-feira, 1 de janeiro de 2020	444	Pedido Múltiplo
TE10021	2	Brasília	quarta-feira, 1 de janeiro de 2020	342	Pedido Múltiplo
TE10011	2	Porto Alegre	quarta-feira, 1 de janeiro de 2020	41	Pedido Múltiplo
TE10008	2	Salvador	quarta-feira, 1 de janeiro de 2020	557	Pedido Múltiplo

2 4.2 - Sintaxe, principais operadores e principais fórmulas

2.1 Sintaxe

Podemos resumir a estrutura de uma Medida/Coluna Calculada da seguinte maneira:



2.2 Principais Operadores

2.2.1 Operadores Básicos:

Operador Matemático	Significado	Exemplo
+	Adição	2 + 4
-	Subtração	7 - 3
*	Multiplicação	3 * 2
/	Divisão	10 / 2
^	Expoente	5 ^ 2

2.2.2 Operadores de comparação, usados na fórmula SE:

Operador de Comparação	Significado	Exemplo
=	É igual a	[Loja] = "Salvador"
>	É maior do que	[Tamanho Pedido] > 2
<	É menor do que	[Tamanho Pedido] < 2
>=	Maior ou igual a	[Quantidade Devolvida] >= 3
<=	Menor ou igual a	[Quantidade Devolvida] <= 3
<>	Diferente de	[Categoria] <> "Celular"

Operadores

Operador de Texto/Lógico	Significado	Exemplo
&	Concatena dois ou mais textos em um único texto	"Alon" & "Pinheiro"
&&	Cria uma condição E entre dois ou mais testes lógicos	[Loja] = "Salvador" && [Tamanho Pedido] > 2
	Cria uma condição OU entre dois ou mais testes lógicos	[Loja] = "Salvador" [Loja] = "Curitiba"
IN	Cria uma condição OU com base em valores de uma lista (usando chaves)	[Loja] IN {"Salvador"; "Curitiba"; "Belo Horizonte"}

2.3 Principais Fórmulas

Na imagem abaixo temos as fórmulas mais comuns a serem usadas na linguagem DAX. Não veremos todas durante esse treinamento, pois priorizamos o foco nas principais dentre elas.

Fórmulas Matemáticas e Estatísticas	Fórmulas Lógicas	Fórmulas de Texto	Fórmulas de Filtro	Fórmulas de Data e Hora
<p><i>Exemplos Comuns:</i></p> <ul style="list-style-type: none"> SUM AVERAGE MAX/MIN DIVIDE COUNT/COUNTA COUNTROWS DISTINCTCOUNT <p><i>Fórmulas Iterativas:</i></p> <ul style="list-style-type: none"> SUMX AVERAGEX MAXX/MINX RANKX COUNTX 	<p><i>Exemplos Comuns:</i></p> <ul style="list-style-type: none"> IF IFERROR AND OR NOT SWITCH TRUE FALSE 	<p><i>Exemplos Comuns:</i></p> <ul style="list-style-type: none"> CONCATENATE FORMAT LEFT/MID/RIGHT UPPER/LOWER PROPER LEN SEARCH/FIND REPLACE REPT SUBSTITUTE TRIM UNICHAR 	<p><i>Exemplos Comuns:</i></p> <ul style="list-style-type: none"> CALCULATE FILTER ALL ALLEXCEPT RELATED RELATEDTABLE DISTINCT VALUES EARLIER/EARLIEST HASONEVALUE HASONEFILTER ISFILTERED USERRELATIONSHIP 	<p><i>Exemplos Comuns:</i></p> <ul style="list-style-type: none"> DATEDIFF YEARFRAC YEAR/MONTH/DAY HOUR/MINUTE/SECOND TODAY/NOW WEEKDAY/WEEKNUM <p><i>Fórmulas de Inteligência de Tempo:</i></p> <ul style="list-style-type: none"> DATESYTD DATESQTD DATESMTD DATEADD DATESINPERIOD

3 4.3 - Fórmulas de Data e Hora

Vamos começar nossos estudos vendo as fórmulas DAX de datas.

As fórmulas que serão vistas:

- YEAR/MONTH/DAY
- WEEKDAY/WEEKNUM
- EOMONTH
- DATEDIFF
- TODAY/NOW
- YEARFRAC
- HOUR/MINUTE/SECOND

3.1 YEAR, MONTH e DAY

Vamos voltar a nossa tabela Calendário: criamos diversas colunas nessa base com informações relacionadas as datas usando as Ferramentas de Data do Power Query; algumas delas também poderiam ter sido criadas com fórmulas DAX. Vamos ver um exemplo:

A coluna de **Ano** pode ser criada usando a fórmula **YEAR** (Ano). Para fazer isso, primeiro excluimos a nossa coluna já feita; em seguida, selecionamos a opção de criar uma coluna calculada, damos a ela o nome de "Ano" e selecionamos a coluna que contém a data da qual queremos saber o ano:

```
1 Ano = YEAR(Calendario[Data])
```

Pronto! Criamos a nossa coluna de Ano usando uma fórmula DAX.

Data	Ano
quarta-feira, 1 de janeiro de 2020	2020
quinta-feira, 2 de janeiro de 2020	2020
sexta-feira, 3 de janeiro de 2020	2020
sábado, 4 de janeiro de 2020	2020
domingo, 5 de janeiro de 2020	2020
segunda-feira, 6 de janeiro de 2020	2020
terça-feira, 7 de janeiro de 2020	2020
quarta-feira, 8 de janeiro de 2020	2020
quinta-feira, 9 de janeiro de 2020	2020
sexta-feira, 10 de janeiro de 2020	2020
sábado, 11 de janeiro de 2020	2020
domingo, 12 de janeiro de 2020	2020
segunda-feira, 13 de janeiro de 2020	2020
terça-feira, 14 de janeiro de 2020	2020
quarta-feira, 15 de janeiro de 2020	2020

Agora, faça a mesma coisa com a coluna **Mês** e crie uma nova coluna com o **Dia** usando as fórmulas **MONTH** e **DAY**.

Resposta

Para a coluna **Mês**:

```
1 Mês = MONTH(Calendario[Data])
```

Para a coluna **Dia**:

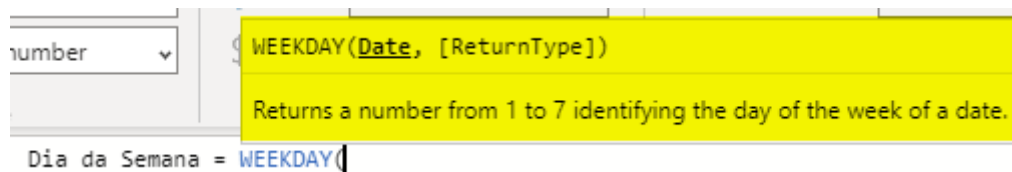
```
1 Dia = DAY(Calendario[Data])
```

→ Como não vamos usar a coluna com os dias no nosso relatório, podemos excluí-la.

3.2 WEEKDAY

Outra coisa interessante de se fazer é extrair o dia da semana de cada uma das datas. Podemos usar essa coluna, por exemplo, para criar um resumo de vendas por dia da semana. Para fazer isso vamos usar a fórmula **WEEKDAY**.

Vamos chamar a nossa coluna de **Dia da Semana**:



Essa função vai retornar para nós um número entre 1 e 7 identificando o dia da semana. Ela precisa de dois argumentos: o primeiro é a **coluna que contém essa data**, e o segundo é o número de retorno de acordo com a semana - basicamente, podemos escolher o dia da semana que queremos como inicial, como indicado na tabela:

Opção	Significado
1	Dias indo de 1 a 7 começando no Domingo
2	Dias indo de 1 a 7 começando na Segunda-Feira
3	Dias indo de 0 a 6 começando na Segunda-Feira

No nosso caso vamos escolher a primeira opção. A fórmula fica assim:

```
1 Dia da Semana = WEEKDAY(Calendario[Data],1)
```

E o resultado:

Data	Dia da Semana
quarta-feira, 1 de janeiro de 2020	4
quinta-feira, 2 de janeiro de 2020	5
sexta-feira, 3 de janeiro de 2020	6
sábado, 4 de janeiro de 2020	7
domingo, 5 de janeiro de 2020	1
segunda-feira, 6 de janeiro de 2020	2
terça-feira, 7 de janeiro de 2020	3
quarta-feira, 8 de janeiro de 2020	4
quinta-feira, 9 de janeiro de 2020	5
sexta-feira, 10 de janeiro de 2020	6
sábado, 11 de janeiro de 2020	7
domingo, 12 de janeiro de 2020	1
segunda-feira, 13 de janeiro de 2020	2
terça-feira, 14 de janeiro de 2020	3
quarta-feira, 15 de janeiro de 2020	4

3.3 WEEKNUM

A fórmula **WEEKNUM** retorna para nós o número da semana no **Ano**, como temos na coluna Semana do Ano. Ela funciona da mesma maneira que a fórmula WEEKDAY, com dois argumentos: o primeiro sendo a coluna que temos a informação da data, e o segundo nos permite escolher se queremos que a semana comece no Domingo (1) ou na Segunda-Feira (2).

Para fazer uma coluna igual a que foi criada anteriormente, a fórmula ficaria assim:

```
1 Semana do Ano (2) = WEEKNUM(Calendario[Data],1)
```

3.4 EOMONTH

Para dar continuidade, vamos voltar para a nossa Base de Vendas.

A fórmula **EOMONTH** (*End of Month* - Final do Mês) consegue retornar o final do mês de uma data - por exemplo, 15/07/2023 → 31/07/2023. Essa informação pode ser interessante para nós, por exemplo, se a nossa loja emitir um boleto de pagamento no final de cada mês; nesse caso, usando a fórmula vamos saber o dia exato para a emissão desse boleto.

A fórmula irá pedir dois argumentos: a **data de início** e o **meses**, que representa o número de meses antes ou depois da data inicial - como queremos o próprio mês, colocamos o número zero (0):

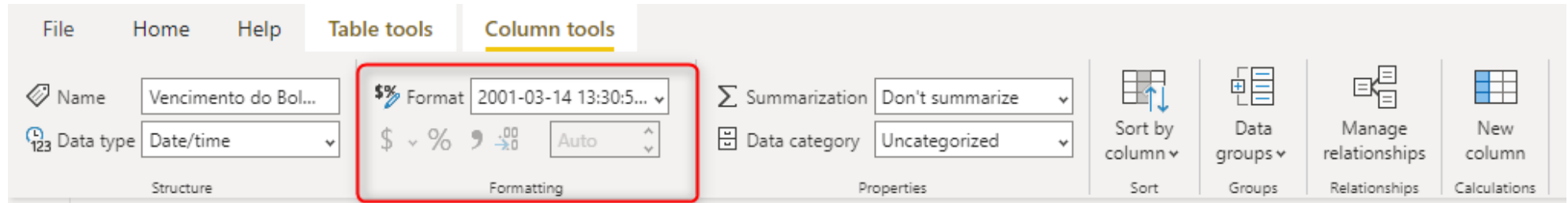
```
1 Vencimento do Boleto = EOMONTH(Calendario[Data], 0)
```

Dando um OK criamos a nossa coluna:

Vencimento do Boleto
31/01/2020 00:00:00
31/01/2020 00:00:00
31/01/2020 00:00:00
31/01/2020 00:00:00
31/01/2020 00:00:00
31/01/2020 00:00:00
31/01/2020 00:00:00
31/01/2020 00:00:00
31/01/2020 00:00:00
31/01/2020 00:00:00
31/01/2020 00:00:00
31/01/2020 00:00:00
31/01/2020 00:00:00
31/01/2020 00:00:00
31/01/2020 00:00:00
31/01/2020 00:00:00

Como podemos ver, ele retornou tanto a data quanto um valor em horas; como não precisamos desses valores, vamos formatar a nossa coluna para que apareçam apenas as informações sobre as datas.

Para isso, selecionamos a coluna e vamos na guia **Ferramentas de Coluna** (*Column Tools*) na opção **Formatar** (*Format*), onde procuramos o formato que queremos que nossa data fique:



Nesse caso vamos escolher o formato padrão dd/mm/aaaa:

The screenshot shows the Excel ribbon with the 'Number' tab active. The 'Format' dropdown menu is open, displaying a list of common and date-specific formats. The input field at the top contains the date and time '2001-03-14 13:30:55'. In the 'Common formats' section, the format '14/03/2001 (dd/mm/yyyy)' is highlighted in yellow. Below this, the 'Date formats' section lists various other date representations.

Format	Description
*14/03/2001 13:30:55	(General Date)
*14/03/2001	(Short Date)
*quarta-feira, 14 de março de 2001	(Long Date)
*13:30	(Short Time)
*13:30:55	(Long Time)
14/03/2001	(dd/mm/yyyy)
2001-03-14 13:30:55	(yyyy-mm-dd hh:mm:ss)
2001-03-14	(yyyy-mm-dd)
2001-03	(yyyy-mm)
Date formats	
quarta-feira, 14 de março de 2001	(dddd, d" de "mmm"
14 de março de 2001	(d" de "mmm"
14/03/01	(dd/mm/yy)
14/3/2001	(d/m/yyyy)
14/3/01	(d/m/yy)
14-03-01	(dd-mm-yy)
14-03-2001	(dd-mm-yyyy)
14-3-01	(d-m-yy)
14-3-2001	(d-m-yyyy)
14.03.01	(dd.mm.yy)
14.03.2001	(dd.mm.yyyy)

E pronto! Conseguimos ajustar a nossa coluna!

Data	Vencimento do Boleto
quarta-feira, 1 de janeiro de 2020	31/01/2020
quinta-feira, 2 de janeiro de 2020	31/01/2020
sexta-feira, 3 de janeiro de 2020	31/01/2020
sábado, 4 de janeiro de 2020	31/01/2020
domingo, 5 de janeiro de 2020	31/01/2020
segunda-feira, 6 de janeiro de 2020	31/01/2020
terça-feira, 7 de janeiro de 2020	31/01/2020
quarta-feira, 8 de janeiro de 2020	31/01/2020
quinta-feira, 9 de janeiro de 2020	31/01/2020
sexta-feira, 10 de janeiro de 2020	31/01/2020
sábado, 11 de janeiro de 2020	31/01/2020
domingo, 12 de janeiro de 2020	31/01/2020
segunda-feira, 13 de janeiro de 2020	31/01/2020
terça-feira, 14 de janeiro de 2020	31/01/2020
quarta-feira, 15 de janeiro de 2020	31/01/2020

Como não vamos usar essa coluna, podemos apagá-la também.

3.5 DATEDIFF, TODAY, NOW

Essa fórmula retorna a diferença entre duas datas. Podemos calcular novamente a idade dos clientes, agora usando uma fórmula DAX.

Para isso, voltamos até a tabela de clientes. A fórmula **DATEDIFF** pede 3 argumentos: a data inicial, a data final e o intervalo. Nesse caso, essas informações são:

- **Data Inicial:** a data de nascimento dos clientes;
- **Data Final:** a data mais recente; como queremos calcular a idade e queremos que ela sempre atualize, vamos usar a fórmula **TODAY()** para pegar esse valor;

- e o **Intervalo**: a forma que queremos essa data, seja em dias, horas, minutos, etc.; como queremos calcular a idade, faz sentido pedirmos que o intervalo seja em anos.

Para calcular a idade, a fórmula fica da seguinte maneira:

```
1 Idade do Cliente = DATEDIFF(Clientes[Data de Nascimento], TODAY(), YEAR)
```

3.6 TODAY x NOW

Enquanto a função TODAY retorna apenas a data atual - "05/07/2023" - , a função NOW retorna data e hora atuais "05/07/2023 15:00:00". Ambas retornam essas informações com base no sistema de data e hora do computador em que o modelo está sendo executado.

3.7 YEARFRAC

A função YEARFRAC retorna a fração do ano entre duas datas, permitindo que a diferença em anos entre duas datas seja calculada de forma mais precisa - quando é necessário calcular a diferença entre duas datas com uma granularidade mais refinada do que apenas anos inteiros. Sua sintaxe:

```
YEARFRAC(StartDate, EndDate, [Basis])
```

Returns the year fraction representing the number of whole days between start_date and end_date.

Onde *StartDate* representa a data inicial, *EndDate* a data final e *Basis*, que é um argumento opcional (0 por padrão) responsável por controlar o método usado para calcular a fração do ano - se será considerado o número real de dias no mês e no ano, se o cálculo vai assumir um ano com 360 dias, entre outras opções.

3.8 HOUR, MINUTE e SECOND

Enquanto as funções YEAR/MONTH/DAY são responsáveis por extraírem as informações específicas sobre as datas, as funções HOUR/MINUTE/SECOND exercem o mesmo papel extraindo informações específicas sobre o componente de hora de uma data ou horário. Recebem como argumento um *datetime* contendo a hora completa, e separam as informações conforme necessário.

4 4.4 - Fórmulas Lógicas

Vamos agora para as fórmulas lógicas. Veremos as seguintes fórmulas:

- IF
- AND
- OR
- NOT
- SWITCH

4.1 IF

Agora que temos a idade dos clientes, queremos criar uma coluna com informações sobre quem tem mais que 40 anos e quem tem menos que 40 anos. Para isso, usaremos a fórmula IF - que já vimos anteriormente.

Vamos criar uma nova coluna chamada **Categoria do Cliente**, onde os clientes com idade superior a 40 anos são da Categoria 1 e clientes com idade menor que 40 anos são da Categoria 2. A nossa fórmula fica assim:

```
1 Categoria do Cliente = IF(Clientes[Idade do Cliente] > 40, "Categoria 1", "Categoria 2")
```

O resultado na nossa coluna:

Idade do Cliente	Categoria do Cliente
72	Categoria 1
41	Categoria 1
23	Categoria 2
51	Categoria 1
67	Categoria 1
25	Categoria 2
51	Categoria 1
23	Categoria 2
25	Categoria 2
39	Categoria 2
44	Categoria 1
51	Categoria 1
72	Categoria 1
69	Categoria 1
56	Categoria 1

4.2 IF Composto (Aninhado)

E se quisermos criar apenas 2 categorias, optarmos por separar nossos clientes em 4 categorias de acordo com a idade?

Para isso precisamos tornar nosso IF um pouco mais sofisticado: na opção **Result if False**, podemos colocar outro IF, de maneira que criamos mais categorias para o nosso valor de idade. Isso é chamado de **IF composto**.

Dica: ao clicar na setinha ao lado da barra de fórmulas conseguimos aumentar o editor, facilitando a visualização:

```
1 Categoria do Cliente = IF(Clientes[Idade do Cliente] > 40, "Categoria 1", "Categoria 2")
```

O critério das categorias é o seguinte:

Idade do Cliente	Categoria
Acima de 60	Categoria 1
Entre 45 e 59	Categoria 2
Entre 30 e 44	Categoria 3
Abaixo de 30	Categoria 4

Não precisamos criar outra coluna, basta editar a que criamos previamente clicando na coluna criada e então na fórmula de barras. Nossa nova fórmula é a seguinte:

```
1 Categoria do Cliente = IF(Clientes[Idade do Cliente] > 60, "Categoria 1",  
2 IF(Clientes[Idade do Cliente] > 45, "Categoria 2",  
3 IF(Clientes[Idade do Cliente] > 30, "Categoria 3", "Categoria 4")))  
4
```

Shift + Enter
pula linhas

Nosso novo resultado:

Idade do Cliente	Categoria do Cliente
72	Categoria 1
41	Categoria 3
23	Categoria 4
51	Categoria 2
67	Categoria 1
25	Categoria 4
51	Categoria 2
23	Categoria 4
25	Categoria 4
39	Categoria 3
44	Categoria 3
51	Categoria 2
72	Categoria 1
69	Categoria 1
56	Categoria 2

Com mais categorias é possível fazer uma análise melhor, de forma a adequar a estratégia de vendas de acordo com as faixas etárias que mais compram determinados produtos.

4.3 Função IF com várias condições - OR

Nosso objetivo agora é criar uma fórmula que seja capaz de diferenciar os dias da semana, na base Calendário, de acordo com duas possibilidades: **Dia da Semana** e **Fim de Semana**.

Com a fórmula WEEKDAY nós conseguimos os números referentes a cada dia da semana, com o 1 representando o domingo. Agora, precisamos criar uma estrutura lógica que retorne os textos correspondentes da seguinte maneira:

- Se os números forem 1 (domingo) **OU** 7 (sábado), retornaremos o texto "Fim de Semana";
- Nos outros casos a fórmula irá retornar "Dia da Semana"

A ênfase no **OU (OR)** vem justamente da necessidade de termos duas condições na nossa fórmula: se o número for 1 ou 7. Então, para criar a nossa nova coluna vamos usar a estrutura **OR** junto com a estrutura **IF**.

Podemos juntar essas duas estruturas de duas maneiras:

Primeira opção: **usando a função OR**:

```
1 Fim de Semana = IF(OR(Calendario[Dia da Semana] = 1, Calendario[Dia da Semana] = 7), "Fim de Semana", "Dia da Semana")
```

Segunda opção: **usando o operador lógico ||**:

```
1 Fim de Semana = IF(Calendario[Dia da Semana] = 1 || Calendario[Dia da Semana] = 7, "Fim de Semana", "Dia da Semana")
```

A diferença é apenas de sintaxe. Para a condição do nosso IF ser verdadeira, basta que uma das condições dentro do OR seja verdadeira - por isso que estamos usando essa estrutura. O resultado:

Dia da Semana	Fim de Semana
4	Dia da Semana
5	Dia da Semana
6	Dia da Semana
7	Fim de Semana
1	Fim de Semana
2	Dia da Semana
3	Dia da Semana
4	Dia da Semana
5	Dia da Semana
6	Dia da Semana
7	Fim de Semana
1	Fim de Semana
2	Dia da Semana
3	Dia da Semana
4	Dia da Semana

4.4 Função IF com várias condições - AND

Como vimos, na estrutura OR basta que uma condição seja verdadeira para o nosso resultado ser verdadeiro. Agora vamos estudar a estrutura **OR**, que **apenas tem sua saída como verdadeira quando todas as suas condições são verdadeiras**.

Na tabela Clientes, vamos criar uma coluna em que determinamos quais mulheres vão ter direito a descontos dos dias das mães. Para isso, elas precisam ter acima de 50 anos.

Então agora temos duas condições que precisam ser verdadeiras: elas precisam ser mulheres e ter mais de 50 anos. Vamos criar uma coluna chamada **Promoção Dia das Mães**. Assim como na estrutura OR, na estrutura AND também podemos usar tanto a função OR quanto seu operador lógico.

Usando a função OR:

```
1 Promoção Dia das Mães = IF(OR(Clientes[Sexo] = "Feminino", Clientes[Idade do Cliente] > 50), "Sim", "Não")
```

Usando o operador lógico:

```
1 Promoção Dia das Mães = IF(Clientes[Sexo] = "Feminino" && Clientes[Idade do Cliente] > 50, "Sim", "Não")
```

Se as duas condições forem verdadeiras o IF entra na condição verdadeira; caso contrário, ele retornará o resultado ligado à condição falsa.

4.5 SWITCH

A função SWITCH no Power BI é uma função condicional que permite avaliar uma expressão e retornar um resultado correspondente ao caso ligado à expressão. A sintaxe básica é a seguinte:

SWITCH(expressão, caso1, resultado1, caso2, resultado2, ..., casoN, resultadoN, [resultadoPadrão])

A expressão é avaliada em relação a cada caso, e ao achar o correspondente o retorno é o resultado ligado ao caso. Se nenhum caso corresponder à expressão e um resultado padrão for fornecido, esse resultado será retornado; se não houver nenhum resultado padrão, a função retornará um valor em branco.

Um exemplo da função SWITCH:

SWITCH(DiaDaSemana, 1, "Domingo", 2, "Segunda-feira", 3, "Terça-feira", 4, "Quarta-feira", 5, "Quinta-feira", 6, "Sexta-feira", 7, "Sábado", "Outro")

Nesse exemplo, a função SWITCH avalia a variável "DiaDaSemana" e retorna o nome correspondente ao dia da semana. Se o valor da variável não for um número de 1 a 7, o resultado padrão "Outro" será retornado.

5 4.5 - Fórmulas de Texto

As fórmulas de texto que veremos:

- CONCATENATE
- PROPER
- LEFT/MID/RIGHT
- UPPER/LOWER
- FORMAT
- LEN
- SEARCH/FIND
- REPLACE
- REPT
- SUBSTITUTE
- TRIM
- UNICHAR

5.1 CONCATENATE

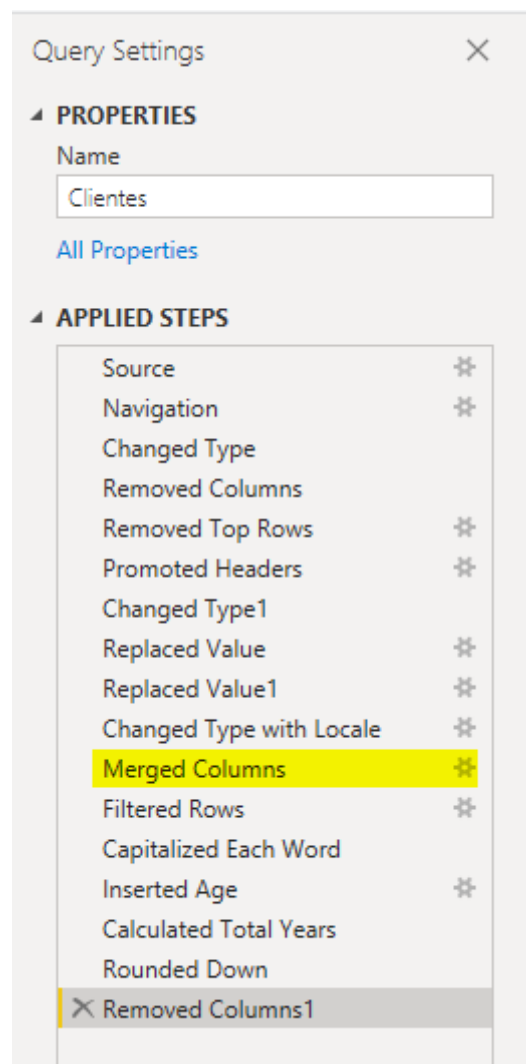
Vamos começar vendo uma fórmula de texto que junta 2 ou mais textos - ou seja, **concatena** textos.

Usamos uma ferramenta parecida no Editor de Consultas para juntar os textos de duas colunas. Fizemos isso para juntar a coluna de Nome e Sobrenome e formar a coluna Nome Completo.

Para ver a aplicação da fórmula de texto, vamos excluir a coluna de Nome Completo e criar outra, utilizando o operador **&** para juntar diversos textos (um único *ampersand*, diferente da aplicação da fórmula AND).

Detalhe: a função chamada CONCATENATE também existe, porém ela só é capaz de unir duas cadeias de texto em uma só - para unir três ou mais utilizamos o operador. Na nossa coluna, por exemplo, queremos juntar nome e sobrenome com um espaço entre eles, o que caracteriza três cadeias de texto.

Como quando mesclamos as colunas não criamos uma nova, se apenas removermos a coluna com o nome completo dos clientes perdemos essa informação. Então antes de aplicar a nossa ferramenta, precisamos voltar ao Power Query e apagar a etapa na qual mesclamos as colunas:



Clicando com o botão direito em cima dessa etapa e clicando em **Apagar** (*Delete*) e confirmando a nossa escolha, voltamos a ter as colunas Nome e Sobrenome. Entretanto, quando voltamos a última etapa aplicada, nos deparamos com um erro. O que aconteceu?

Explicação

A mensagem que apareceu na caixa de confirmação dizia o seguinte: "Você tem certeza se que deseja excluir esta etapa? Excluir esta etapa pode afetar as etapas subsequentes, o que pode fazer com que sua consulta seja interrompida."

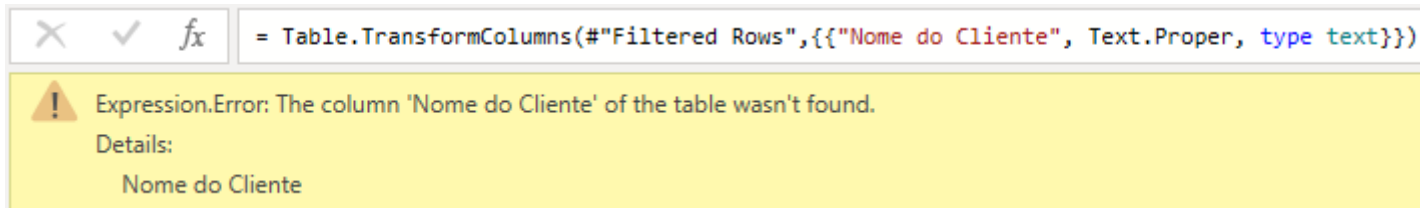
Delete Step

Are you sure you want to delete this step? Deleting this step may affect subsequent steps, which could cause your query to break.

Delete

Cancel

Toda vez que vamos apagar uma etapa que não é a última etapa do nosso processo, precisamos saber como isso impactará as etapas subsequentes. Por exemplo, no nosso caso, após juntarmos as colunas usamos a ferramenta **Colocar Cada Letra em Maiúsculo** para deixar os nomes formatados corretamente. Isso foi aplicado a coluna Nome do Cliente; porém acabamos de excluir a etapa na qual essa coluna era criada, e por isso ele não foi capaz de identificá-la:



Para corrigir esse problema precisamos apagar também a etapa onde usamos a ferramenta mencionada. Nesse caso, como essas informações não foram mais usadas conseguimos apagar corretamente as etapas. Feito isso, temos as colunas que queríamos na nossa tabela Clientes novamente:

Código Cliente	Sexo	Nº de Filhos	Data de Nascimento	E-mail	CPF	Idade do Cliente	Categoria do Cliente	Promoção Dia das Mães	Primeiro Nome	Sobrenome
2	Masculino	0	segunda-feira, 9 de abril de 1951	roberto46@gmail.com	33264169658	72	Categoria 1	Não	ROBERTO	FIGUEIREDO
5	Masculino	0	sábado, 19 de junho de 1982	breno49@icloud.com	95867983463	41	Categoria 3	Não	BRENO	RIBEIRO
6	Masculino	0	quarta-feira, 10 de maio de 2000	lucas23@hotmail.com	41957227222	23	Categoria 4	Não	LUCAS	MESQUITA
8	Masculino	0	domingo, 14 de maio de 1972	alexandre42@yahoo.com	00779784787	51	Categoria 2	Não	ALEXANDRE	GUEDES
13	Masculino	0	domingo, 5 de agosto de 1956	eduardo22@hotmail.com	41565439828	67	Categoria 1	Não	EDUARDO	SONE
21	Masculino	0	terça-feira, 15 de setembro de 1998	stefan96@live.com	52748863555	25	Categoria 4	Não	STEFAN	ALIANI
22	Masculino	0	quarta-feira, 18 de outubro de 1972	david14@live.com.br	83186335732	51	Categoria 2	Não	DAVID	LIMA
29	Masculino	0	sábado, 16 de setembro de 2000	carlos91@icloud.com	12884594881	23	Categoria 4	Não	CARLOS	FONSECA
35	Masculino	0	sexta-feira, 5 de junho de 1998	norman40@outlook.com	08426546494	25	Categoria 4	Não	NORMAN	BOTELHO
39	Masculino	0	sexta-feira, 20 de janeiro de 1984	joao29@uol.com.br	13321597438	39	Categoria 3	Não	JOÃO	NASCIMENTO
40	Masculino	0	sábado, 13 de outubro de 1979	guilherme62@outlook.com	85633667181	44	Categoria 3	Não	GUILHERME	CASTILHO
41	Masculino	0	domingo, 10 de setembro de 1972	anizio58@hotmail.com	29933624299	51	Categoria 2	Não	ANÍZIO	QUINTO
52	Masculino	0	quarta-feira, 11 de julho de 1951	jose78@live.com	89129293667	72	Categoria 1	Não	JOSÉ	MIRANDA
53	Masculino	0	sexta-feira, 24 de dezembro de 1954	gabriel43@msn.com	67765236967	69	Categoria 1	Não	GABRIEL	SILVA
58	Masculino	0	sábado, 25 de março de 1967	joao27@live.com.br	04839212913	56	Categoria 2	Não	JOÃO	GONÇALVES
60	Masculino	0	domingo, 21 de outubro de 1990	thomaz35@terra.com.br	23222999545	33	Categoria 3	Não	THOMÁZ	SANTOS
61	Masculino	0	terça-feira, 14 de julho de 1964	barbara98@yahoo.com.br	47439921997	59	Categoria 2	Não	BÁRBARA	PEREIRA
63	Masculino	0	sexta-feira, 10 de agosto de 1984	raphael78@gmail.com	56731747661	39	Categoria 3	Não	RAPHAEL	RODRIGUES
65	Masculino	0	sábado, 10 de agosto de 1996	joao54@icloud.com	67976985864	27	Categoria 4	Não	JOAO	FREITAS
70	Masculino	0	terça-feira, 14 de junho de 1977	raphael89@gmail.com	39922544822	46	Categoria 2	Não	RAPHAEL	BARROS
75	Masculino	0	terça-feira, 22 de março de 1994	lucas11@live.com.br	36765489848	29	Categoria 4	Não	LUCAS	ARAÚJO
79	Masculino	0	segunda-feira, 31 de julho de 2000	rilson32@icloud.com	82952646182	23	Categoria 4	Não	RILSON	MENDONÇA
87	Masculino	0	segunda-feira, 28 de novembro de 1977	renan25@yahoo.com	12717427466	46	Categoria 2	Não	RENAN	VILLANOVA
92	Masculino	0	quarta-feira, 13 de julho de 1994	matheus59@gmail.com	15964211686	29	Categoria 4	Não	MATHEUS	SILVA
93	Masculino	0	quinta-feira, 3 de maio de 1979	rubens44@icloud.com	55164737194	44	Categoria 3	Não	RUBENS	SENA
103	Masculino	0	sexta-feira, 5 de setembro de 1980	gustavo43@gmail.com	24695843175	43	Categoria 3	Não	GUSTAVO	SOUZA
105	Masculino	0	quinta-feira, 18 de maio de 1978	thiago81@gmail.com	43543338515	45	Categoria 3	Não	THIAGO	BINELLO
108	Masculino	0	quarta-feira, 17 de abril de 1974	allan54@gmail.com	72824266677	49	Categoria 2	Não	ALLAN	NÓBREGA
115	Masculino	0	domingo, 30 de maio de 1954	gabriel36@icloud.com	16776184995	69	Categoria 1	Não	GABRIEL	GONÇALVES
117	Masculino	0	segunda-feira, 23 de agosto de 1965	julio71@yahoo.com	23429781457	58	Categoria 2	Não	JÚLIO	PUNTEL
118	Masculino	0	quinta-feira, 6 de setembro de 1951	daniel54@msn.com	49677877839	72	Categoria 1	Não	DANIEL	CARNEIRO
119	Masculino	0	quarta-feira, 28 de julho de 1954	mateus12@terra.com.br	36687764282	69	Categoria 1	Não	MATEUS	RIBEIRO
125	Masculino	0	sábado, 25 de fevereiro de 1950	silvio27@yahoo.com.br	71179786275	73	Categoria 1	Não	SILVIO	VIRÍSSIMO
129	Masculino	0	sábado, 11 de dezembro de 1982	jackson58@gmail.com	81586292917	41	Categoria 3	Não	JACKSON	GUADAGNINO
131	Masculino	0	sábado, 15 de setembro de 1956	lazar015@icloud.com	04488735588	67	Categoria 1	Não	LÁZARO	MATTOS
141	Masculino	0	terça-feira, 4 de setembro de 1973	pedro68@msn.com	89868879664	50	Categoria 2	Não	PEDRO	ERTHAL
144	Masculino	0	terça-feira, 8 de janeiro de 1991	raphael65@msn.com	21392649838	32	Categoria 3	Não	RAPHAEL	BITTENCOURT
145	Masculino	0	sábado, 11 de novembro de 1995	pedro31@terra.com.br	47525328743	28	Categoria 4	Não	PEDRO	WANDERLEY

Fields

Search

Calendário

Clientes

Categoria do Cliente

Código Cliente

CPF

Data de Nascimento

E-mail

Idade do Cliente

Nº de Filhos

Primeiro Nome

Promoção Dia das ...

Sexo

Sobrenome

Devoluções

Lojas

Produtos

TotalVendas

Vendas2020

Vendas2021

Vendas2022

A fórmula fica da seguinte maneira:

```
1 Nome Completo = Clientes[Primeiro Nome] & " " & Clientes[Sobrenome]
```

Nossa coluna:

Nome Completo
NATÁLIA BARBOSA
ROBERTO FIGUEIREDO
STELA FERREIRA
RAFAEL ARAUJO
BRENO RIBEIRO
LUCAS MESQUITA
DEBORA RAMOS
ALEXANDRE GUEDES
JULIANA MONTANHOLI
GABRIELA FONTOURA
HENRIQUE CORREA
JULIANA FERREIRA
EDUARDO SONE
CAROLINA HEIMLICH
FABIO MARINS

5.2 PROPER

A função **PROPER** é responsável por converter a primeira letra de cada palavra em maiúscula se for minúscula, e converte todas as outras letras em minúsculas. Ela também remove múltiplos espaços em branco consecutivos, substituindo-os por um único espaço. Para usá-la, basta apenas selecionar a função e passar o texto a ser formatado como argumento.

A versão do Power BI utilizada nas aulas não apresenta essa função :(

Por isso, para formatarmos o Nome do Cliente de maneira mais simples vamos utilizar a formatação do próprio Power Query. Caso a versão usada apresente essa função, use ela.

5.3 LEFT/MID/RIGHT/UPPER/LOWER

As funções **LEFT**, **MID** e **RIGHT** retornam caracteres da **esquerda, meio e direita**, respectivamente. Já as fórmulas **UPPER** e **LOWER** transformam os textos em **MAIÚSCULO e minúsculo**.

Vamos ver alguns exemplos:

LEFT/RIGHT

Quando criamos um relatório/gráfico/*dashboard*, criamos da forma mais visual possível. Uma prática muito comum é criar análises mensais para analisarmos as nossas informações; por isso, criamos a nossa coluna com o nome dos meses.

Uma prática muito comum em relação aos meses é abreviá-los (janeiro = jan), para a visualização das informações nos gráficos ficarem melhores. E para pegar as três primeiras letras, vamos usar a função **LEFT** que pede dois argumentos: o **texto de onde queremos extrair uma parte** e **quantos caracteres (da esquerda para a direita) queremos extrair**. Criamos uma nova coluna chamada Mês Abreviado, com a seguinte fórmula:

```
1 Mês Abreviado = LEFT(Calendario[Nome do Mês], 3)
```

A coluna criada:

Nome do Mês ▾	Mês Abreviado ▾
Janeiro	Jan
Janeiro	Jan
Janeiro	Jan
Janeiro	Jan
Janeiro	Jan
Janeiro	Jan
Janeiro	Jan
Janeiro	Jan
Fevereiro	Fev
Fevereiro	Fev
Fevereiro	Fev
Fevereiro	Fev
Fevereiro	Fev
Fevereiro	Fev
Fevereiro	Fev
Fevereiro	Fev
Março	Mar

As fórmulas LEFT e RIGHT funcionam da mesma forma; a diferença é que enquanto a fórmula LEFT pega os caracteres mais a esquerda do nosso texto, a fórmula RIGHT pega o número de caracteres especificado mais a direita.

UPPER/LOWER

Uma outra aplicação que costumamos fazer é colocar o texto abreviado em maiúsculo, porque em apresentações costumamos seguir esse modelo. Poderíamos criar outra coluna para fazer isso, mas isso daria mais trabalho; uma maneira mais simples de fazer isso é colocar a fórmula LEFT, que retorna o nosso texto, dentro da fórmula **UPPER**, de forma que todo esse texto será formatado para maiúsculo:

```
1 Mês Abreviado = UPPER(LEFT(Calendario[Nome do Mês], 3))
```

E pronto! Temos nossa coluna formatada com os meses abreviados e em maiúsculo:

Nome do Mês ▾	Mês Abreviado ▾
Janeiro	JAN
Janeiro	JAN
Janeiro	JAN
Janeiro	JAN
Janeiro	JAN
Janeiro	JAN
Janeiro	JAN
Janeiro	JAN
Fevereiro	FEV
Fevereiro	FEV
Fevereiro	FEV
Fevereiro	FEV
Fevereiro	FEV
Fevereiro	FEV
Fevereiro	FEV
Fevereiro	FEV
Março	MAR

MID

A função MID pega os caracteres ao centro do nosso texto. Ele recebe três argumentos: o **texto de qual queremos extrair uma parte (Text)**, o **número referente a posição do primeiro caractere que queremos pegar (Start Position)** e **quantos caracteres a partir dele (NumberOfCharacters)**.

```
MID(Text, StartPosition, NumberOfCharacters)
```

Returns a string of characters from the middle of a text string, given a starting position and length.

Por exemplo: vamos imaginar que queremos uma coluna apenas com a parte numérica do código dos nossos produtos. Poderíamos usar a função RIGHT; entretanto, isso apenas serviria para produtos com um código contendo 5 números - o que nos limitaria a 100000 opções com o mesmo código TE. Apesar de esse número representar várias opções, em outros casos a quantidade de algarismos pode variar, o que pode acabar trazendo problemas futuros.

Usando a função MID podemos colocar uma seleção maior de números e ela só vai nos retornar os números existentes:

```
1 Numeração do Produto = MID(Produtos[SKU], 3, 10000)
```

SKU	Numeração do Produto
TE10001	10001
TE10002	10002
TE10003	10003
TE10004	10004
TE10005	10005
TE10006	10006
TE10007	10007
TE10008	10008
TE10009	10009
TE10010	10010
TE10011	10011
TE10012	10012
TE10013	10013
TE10014	10014
TE10015	10015

5.4 Outras Funções

FORMAT: permite formatar um valor numérico ou de data/hora em um formato específico;

LEN: retorna o número de caracteres em uma determinada expressão de texto;

SEARCH/FIND: localizam a posição de uma *substring* em uma *string*, onde o FIND é *case-sensitive* (diferencia letras maiúsculas e minúsculas);

REPLACE: troca uma *substring* por outra *substring*;

REPT: repete uma determinada *string* um número específico de vezes;

SUBSTITUTE: usada para substituir todas as ocorrências de uma *substring* por outra em uma *string*. Ela permite fazer substituições de texto em uma coluna, medida ou variável que contenha valores de texto;

TRIM: remove os espaços em branco no começo e no final de uma *string*;

UNICHAR: usada para retornar o caractere unicode específico de um determinado código.

6 4.6 - Fórmulas Matemáticas e Estatísticas

Veremos dois tipos de fórmulas: as **Fórmulas Comuns** e as **Fórmulas Iterativas**.

Fórmulas Comuns:

- SUM
- AVERAGE
- MAX/MIN
- DIVIDE
- COUNT/COUNTA
- COUNTROWS
- DISTINCTCOUNT

Fórmulas Iterativas:

- SUMX
- AVERAGEX
- MAXX/MINX
- RANKX
- COUNTX

6.1 RELATED

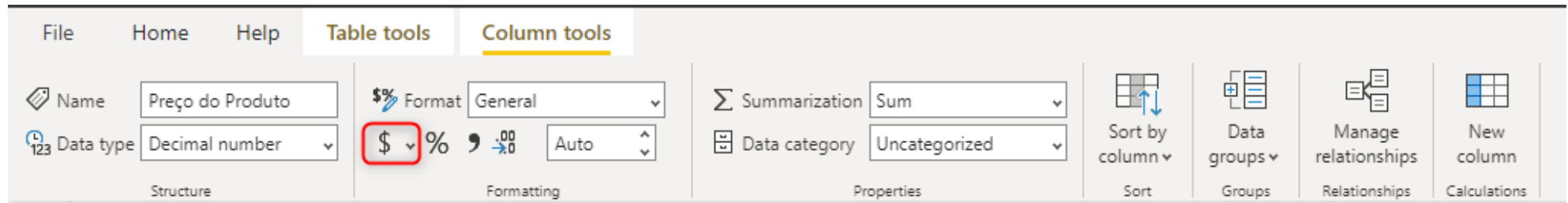
Antes de seguirmos propriamente para as fórmulas da nossa aula, precisamos ver uma fórmula que faz parte das fórmulas de filtro - a fórmula **RELATED**. O que essa fórmula faz basicamente é puxar colunas de uma tabela para outra - de forma parecida com o que a função PROCV do Excel faz. Por isso, apesar de muito simples ela é uma fórmula muito importante para nós.

Na nossa base de vendas temos diversas informações sobre as vendas, mas não temos o valor unitário de cada produto, por exemplo. Essa informação está na base de produtos; então, o que vamos fazer é puxar essa informação para dentro da tabela de vendas.

Para trazer essas informações basta criar uma nova coluna com a fórmula, e passar como argumento a coluna que queremos trazer de outra tabela:

```
1 Preço do Produto = RELATED(Produtos[Custo Unitario])
```

Um detalhe muito importante: **a fórmula RELATED só consegue trazer colunas de uma base para a outra se as bases estiverem RELACIONADAS!** Para formatar a coluna como moeda, basta ir na opção de formatação de moeda, clicar na setinha para abrir a lista de opções e encontrar o real brasileiro:



R\$ Portuguese (Brazil)

E pronto!

SKU	Tamanho Pedido	Loja	Data da Venda	Código Cliente	Tipo do Pedido	Preço do Produto	
TE10009	3	Cuiabá	quarta-feira, 1 de janeiro de 2020	588	Pedido Múltiplo	R\$ 2.700,00	
TE10005	3	Cuiabá	sábado, 4 de janeiro de 2020	48	Pedido Múltiplo	R\$ 5.000,00	
TE10015	3	Cuiabá	terça-feira, 7 de janeiro de 2020	340	Pedido Múltiplo	R\$ 250,00	
TE10009	3	Cuiabá	quarta-feira, 8 de janeiro de 2020	25	Pedido Múltiplo	R\$ 2.700,00	
TE10010	3	Cuiabá	sábado, 11 de janeiro de 2020	744	Pedido Múltiplo	R\$ 800,00	
TE10021	3	Cuiabá	sábado, 11 de janeiro de 2020	323	Pedido Múltiplo	R\$ 260,00	
TE10008	3	Cuiabá	segunda-feira, 13 de janeiro de 2020	53	Pedido Múltiplo	R\$ 3.000,00	
TE10006	3	Cuiabá	segunda-feira, 13 de janeiro de 2020	65	Pedido Múltiplo	R\$ 4.000,00	
TE10017	3	Cuiabá	sábado, 18 de janeiro de 2020	257	Pedido Múltiplo	R\$ 750,00	
TE10003	3	Cuiabá	quinta-feira, 23 de janeiro de 2020	742	Pedido Múltiplo	R\$ 4.100,00	
TE10014	3	Cuiabá	quinta-feira, 23 de janeiro de 2020	700	Pedido Múltiplo	R\$ 150,00	
TE10011	3	Cuiabá	quinta-feira, 23 de janeiro de 2020	703	Pedido Múltiplo	R\$ 2.600,00	
TE10011	3	Cuiabá	sábado, 25 de janeiro de 2020	334	Pedido Múltiplo	R\$ 2.600,00	
TE10003	3	Cuiabá	segunda-feira, 27 de janeiro de 2020	194	Pedido Múltiplo	R\$ 4.100,00	
TE10006	3	Cuiabá	sábado, 1 de fevereiro de 2020	326	Pedido Múltiplo	R\$ 4.000,00	
TE10001	3	Cuiabá	sábado, 1 de fevereiro de 2020	115	Pedido Múltiplo	R\$ 3.500,00	
TE10009	3	Cuiabá	terça-feira, 4 de fevereiro de 2020	120	Pedido Múltiplo	R\$ 2.700,00	
TE10007	3	Cuiabá	sábado, 8 de fevereiro de 2020	97	Pedido Múltiplo	R\$ 2.100,00	
TE10019	3	Cuiabá	domingo, 9 de fevereiro de 2020	738	Pedido Múltiplo	R\$ 350,00	
TE10013	3	Cuiabá	domingo, 9 de fevereiro de 2020	601	Pedido Múltiplo	R\$ 175,00	
TE10003	3	Cuiabá	domingo, 9 de fevereiro de 2020	807	Pedido Múltiplo	R\$ 4.100,00	

Search

> Calendario

> Clientes

> Devoluções

> Lojas

> Produtos

▼ TotalVendas

Código Cliente

Data da Venda

Loja

Preço do Produto

SKU

Tamanho Pedido

Tipo do Pedido

Total Pedidos

> Vendas2020

> Vendas2021

> Vendas2022

Agora que temos a coluna com o preço dos produtos, podemos calcular o faturamento de cada venda. Para isso, basta multiplicar a coluna com o preço do produto e a coluna com o tamanho do pedido:

```
1 Faturamento da Venda = TotalVendas[Tamanho Pedido] * TotalVendas[Preço do Produto]
```

Nosso resultado:

Tamanho Pedido ▾	Preço do Produto ▾	Faturamento da Venda ▾
3	R\$ 2.700,00	R\$ 8.100,00
3	R\$ 5.000,00	R\$ 15.000,00
3	R\$ 250,00	R\$ 750,00
3	R\$ 2.700,00	R\$ 8.100,00
3	R\$ 800,00	R\$ 2.400,00
3	R\$ 260,00	R\$ 780,00
3	R\$ 3.000,00	R\$ 9.000,00
3	R\$ 4.000,00	R\$ 12.000,00
3	R\$ 750,00	R\$ 2.250,00
3	R\$ 4.100,00	R\$ 12.300,00
3	R\$ 150,00	R\$ 450,00
3	R\$ 2.600,00	R\$ 7.800,00
3	R\$ 2.600,00	R\$ 7.800,00
3	R\$ 4.100,00	R\$ 12.300,00

Para finalizar, vamos puxar a informação do nome do gerente para a tabela de devoluções.

Fórmula

```
1 Gerente da Loja = RELATED(Lojas[Gerente])
```

Agora que já temos todas as informações que precisamos, vamos as fórmulas:

6.2 Fórmulas Matemáticas

6.2.1 SUM

Vamos começar criando uma **medida** para guardar a soma total do faturamento que acabamos de calcular. Para isso, utilizaremos a fórmula SUM, onde passamos como argumento a coluna que queremos somar:

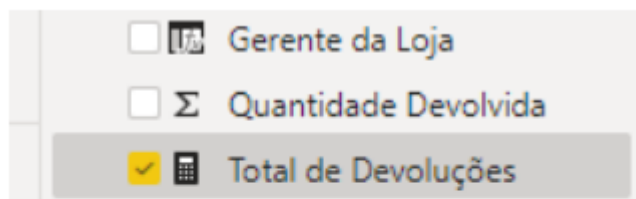
```
1 Soma Total = SUM(TotalVendas[Faturamento da Venda])
```

Lembrando que o valor da medida não aparece na nossa tabela. Para que seja possível visualizar, precisamos colocar essa informação nos *dashboards*/relatórios.




Agora tente criar uma medida com o total de devoluções de produtos na tabela Devoluções.

Resposta

```
1 Total de Devoluções = SUM('Devoluções'[Quantidade Devolvida])
```



Uma coisa interessante da gente notar é o símbolo que aparece logo ao lado de cada informação.

- Uma coluna calculada vai ter um ícone do tipo 
- Uma coluna de valores vai ter um ícone do tipo 
- Já uma medida vai ter um ícone do tipo 

6.2.2 DIVIDE

Também é possível criar Medidas onde fazemos cálculos usando outras Medidas. Vamos ver isso calculando a taxa de devolução dos produtos. Basicamente, precisamos dividir a soma de devoluções pela soma de total de pedidos. Poderíamos fazer isso colocando a fórmula de soma dessa maneira:

```
1 Percentual de Devoluções = SUM('Devoluções'[Quantidade Devolvida]) / SUM(TotalVendas[Tamanho Pedido])
```

Entretanto, estamos apenas repetindo os cálculos que já fizemos nas medidas. Então para simplificar podemos escrever assim:

```
1 Percentual de Devoluções = [Total de Devoluções] / TotalVendas[Total Pedidos]
```

Para formatar como porcentagem, na guia Dados temos as **Ferramentas de Medida** (*Measure tools*), onde podemos escolher a formatação desejada - para todas as nossas colunas e medidas.

Outra maneira de fazer essa divisão é usando a fórmula **DIVIDE**. Essa função basicamente faz a divisão dos valores - tem outras formas de utilizar, mas nos atentaremos a essa -, com uma ferramenta extra: a opção de colocar uma expressão alternativa a ser retornado caso o divisor da nossa operação seja zero - é como se usássemos o IFERROR, porém em apenas uma expressão. Não é obrigatória essa expressão, mas é possível utilizá-la.

Nossa fórmula usando a função **DIVIDE**:

```
1 Percentual de Devoluções = DIVIDE([Total de Devoluções], TotalVendas[Total Pedidos])
```

Para visualizar o resultado podemos criar uma matriz, colocando todas as nossas medidas no campo de valores:

	Total Pedidos	Total de Devoluções	Percentual de Devoluções
	71442	2037	2,85%

Podemos separar esse percentual por produtos colocando o nome deles no campo de linhas:

Nome do Produto	Total Pedidos	Total de Devoluções	Percentual de Devoluções
AirPods Pro (3ª geração)	3075	121	3,93%
Asus MD15	3081		
Buds 4 Pro	3024		
Controle DualSense PS5	3028		
Controle Dualshock 4 PS4	3063	48	1,57%
Controle p/ XBOX sem fio	3061	95	3,10%
EarPods	2854	88	3,08%
Edge 30 Ultra	2958	63	2,13%
Galaxy S23 Ultra	2955	115	3,89%
Galaxy Tab S7 FE	2969	206	6,94%
Galaxy Waych 4	2903	69	2,38%
Headset Gamer Cloud Stinger	3143	75	2,39%
Headset Gamer G332	2875	131	4,56%
Headset Gamer Lamia 2	2961	105	3,55%
iPad 10ª Geração	2961	42	1,42%
iPhone 12 Pro Max	3024	104	3,44%
Lenovo Gamer Gaming I3	3030	74	2,44%
Monitor Gamer Hero	2977	114	3,83%
Monitor Gamer LG 34 Ultra Wide	2914	189	6,49%
Monitor Gamer Snow	2821	28	0,99%
Notebook Gamer Acer Nitro 5	3001	92	3,07%
Notebook Gamer Dell G15	2864	46	1,61%
Watch Series 6	2852	170	5,96%
Zenfone 8	3048	62	2,03%
Total	71442	2037	2,85%

Também podemos classificar o nosso percentual do menor para o maior, do maior para o menor, entre outros. Dessa maneira que as operações que fizemos com as fórmulas DAX vão dando formato ao nosso projeto.

6.2.3 AVERAGE

Para calcular a média de faturamento dos nossos produtos, vamos criar uma nova medida na nossa tabela de Vendas usando a fórmula da média (**AVERAGE**). A medida ficará desta forma:

```
1 Média de Faturamento = AVERAGE(TotalVendas[Faturamento da Venda])
```

Podemos também calcular a média de devoluções por produto, usando a coluna de **Quantidade Devolvida**:

```
1 Média de Devoluções = AVERAGE('Devoluções'[Quantidade Devolvida])
```

Para não visualizar um único valor, podemos colocar as Lojas nas linhas de uma matriz. As medidas que acabamos de criar ficam assim:

Loja	Média de Faturamento	Média de Devoluções
Belo Horizonte	R\$ 5.524,81	1,86
Brasília	R\$ 5.762,59	1,76
Campinas	R\$ 5.558,07	1,82
Cuiabá	R\$ 5.750,14	1,77
Curitiba	R\$ 5.804,95	1,76
Florianópolis	R\$ 5.465,11	1,85
Fortaleza	R\$ 5.955,06	1,84
Porto Alegre	R\$ 5.641,46	1,82
Recife	R\$ 6.005,06	1,70
Rio de Janeiro	R\$ 5.628,31	1,73
Salvador	R\$ 5.709,90	1,88
São Paulo	R\$ 5.562,35	1,65
Vila Velha	R\$ 6.046,73	1,83
Total	R\$ 5.727,03	1,79

6.2.4 MAX/MIN

Vamos descobrir os valores **Mínimo** e **Máximo** de faturamento. Para isso, vamos criar duas novas medidas na base de vendas.

Mínimo:

```
1 Mínimo Faturado = MIN(TotalVendas[Faturamento da Venda])
```

Máximo:

```
1 Máximo Faturado = MAX(TotalVendas[Faturamento da Venda])
```

Temos agora os valores de mínimo e máximo faturado:

	Mínimo Faturado	Máximo Faturado
	R\$ 125,00	R\$ 25.000,00

Podemos ver esse valor, por exemplo, pela categoria dos produtos:

Categoria	Mínimo Faturado	Máximo Faturado
Celular	R\$ 2.100,00	R\$ 25.000,00
Fone de Ouvido	R\$ 125,00	R\$ 9.000,00
Headset	R\$ 150,00	R\$ 1.250,00
Monitor	R\$ 750,00	R\$ 9.000,00
Notebook	R\$ 2.400,00	R\$ 25.000,00
Smart Watch	R\$ 800,00	R\$ 13.500,00
Tablet	R\$ 2.600,00	R\$ 16.500,00
Video Game	R\$ 200,00	R\$ 1.750,00
Total	R\$ 125,00	R\$ 25.000,00

*Para renomear um dos títulos da matriz, basta dar um duplo clique no campo selecionado na parte da **Visualização** e renomea-lo.

6.3 Fórmulas de Contagem

6.3.1 COUNT

Vamos contar a quantidade de vendas feitas - que é diferente de quantidade de produtos vendidos. Para isso, vamos criar uma nova medida usando a fórmula **COUNT**. Essa fórmula vai **contar a quantidade de números** em uma determinada coluna.

No caso, queremos contar a quantidade de vendas a partir da coluna com o Tamanho do Pedido:

```
1 Número de Vendas = COUNT(TotalVendas[Tamanho Pedido])
```

Agora vamos contar a quantidade de devoluções feitas na nossa base Devoluções:

```
1 Número de Devoluções = COUNT('Devoluções'[Quantidade Devolvida])
```

Visualizando o resultado:

Categoria	Número de Vendas	Número de Devoluções
Celular	3993	212
Fone de Ouvido	2964	100
Headset	3004	160
Monitor	2889	178
Notebook	3999	127
Smart Watch	1915	133
Tablet	1968	140
Video Game	3065	90
Total	23797	1140

6.3.2 COUNTA

Além da fórmula COUNT, que conta a quantidade de valores numéricos em uma coluna, temos a fórmula **COUNTA**, que conta a quantidade de valores em uma coluna independente se são números ou textos.

Vamos verificar isso contando o número de clientes cadastrados na nossa loja. Com a função COUNTA não importa a coluna que escolhermos - pode ser a coluna de nome, por exemplo. Se estivéssemos usando a fórmula COUNT, seria necessário usar a coluna com o código do cliente.

```
1 Número de Clientes = COUNTA(Clientes[Nome Completo])
```

Podemos visualizar, por exemplo, a quantidade de clientes de acordo com o sexo:

Sexo	Número de Clientes
Feminino	374
Masculino	448
Total	822

6.3.3 DISTINCTCOUNT

A ideia agora é criarmos uma fórmula que consegue fazer uma **contagem distinta**. Vamos usá-la para saber a quantidade de produtos devolvidos, mas dessa vez de forma distinta.

Ou seja, na nossa base de devoluções, na coluna de SKU, eu não quero saber a quantidade total de linhas da tabela, porque muitos produtos são devolvidos mais de uma vez. Nesse caso, queremos saber, de maneira única e distinta, quantos produtos foram devolvidos.

A fórmula fica da seguinte maneira:

```
1 Quantidade de Produtos Devolvidos = DISTINCTCOUNT('Devoluções'[SKU])
```

Quantidade de Produtos Devolvidos

21

Temos portanto 21 produtos devolvidos. Mas quantos produtos temos ao todo?

Para saber esse valor podemos criar outra medida, contando quantos produtos temos na base dos produtos:

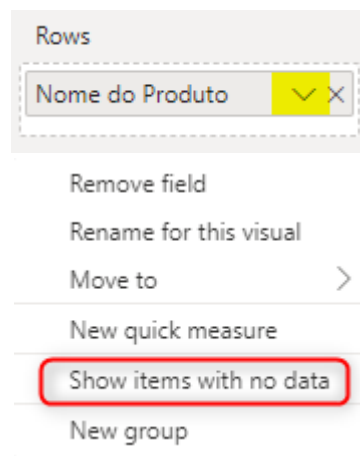
```
1 Quantidade de Produtos Distintos = DISTINCTCOUNT(Produtos[SKU])
```

Quantidade de Produtos Distintos

24

Mas como saber quais produtos não foram devolvidos? Pelas contas, 3 produtos não sofreram devoluções.

Basta criarmos uma matriz com os produtos nas linhas e a medida de devoluções em valores. Os produtos que não foram devolvidos não irão aparecer; para mudar isso, basta irmos nos campos da matriz, na opção Produto, clicarmos na seta para baixo e marcarmos a opção **Mostrar itens sem dados** (*Show items with no data*):



Nosso resultado:

Nome do Produto	Total de Devoluções
AirPods Pro (3ª geração)	121
Asus MD15	
Buds 4 Pro	
Controle DualSense PS5	
Controle Dualshock 4 PS4	48
Controle p/ XBOX sem fio	95
EarPods	88
Edge 30 Ultra	63
Galaxy S23 Ultra	115
Galaxy Tab S7 FE	206
Galaxy Waych 4	69
Headset Gamer Cloud Stinger	75
Headset Gamer G332	131
Headset Gamer Lamia 2	105
iPad 10ª Geração	42
iPhone 12 Pro Max	104
Lenovo Gamer Gaming I3	74
Monitor Gamer Hero	114
Monitor Gamer LG 34 Ultra Wide	189
Monitor Gamer Snow	28
Notebook Gamer Acer Nitro 5	92
Notebook Gamer Dell G15	46
Watch Series 6	170
Zenfone 8	62
Total	2037

6.3.4 COUNTROWS

Para fecharmos a parte de fórmulas de contagem, vamos ver a fórmula **COUNTROWS**. Ela basicamente vai contar a quantidade de linhas que existem em uma tabela. Podemos refazer a fórmula da medida Número de Devoluções, por exemplo, e refazê-la utilizando a fórmula COUNTROWS. Como ele conta quantas linhas existem na nossa tabela, não precisamos passar uma coluna específica, apenas o nome da tabela:

```
1 Número de Devoluções = COUNTROWS('Devoluções')
```

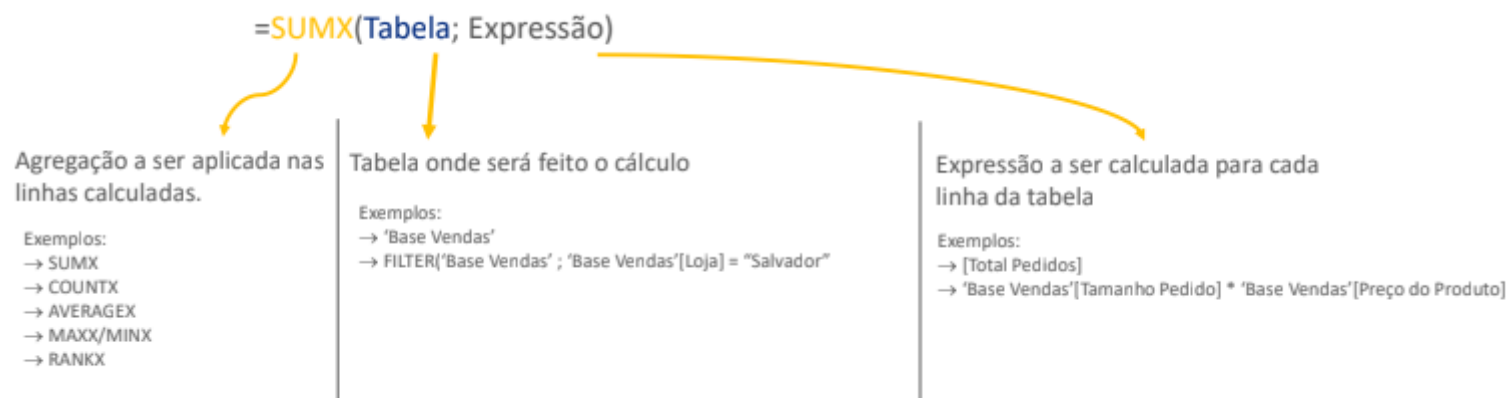
Obtemos o mesmo resultado:

Número de Devoluções	
	1140

6.4 Fórmulas X

As fórmulas X fazem algum cálculo linha por linha da tabela e depois aplicam alguma agregação (soma, máximo, mínimo, média, etc).

Exemplo:



6.4.1 SUMX

Para começarmos a usar as fórmulas X, vamos voltar na base de Vendas. Nessa tabela temos uma coluna chamada Faturamento da Venda que basicamente calcula, linha a linha, o produto entre Tamanho Pedido e Preço do Produto.

Essa coluna que criamos não necessariamente precisaria ser criada de fato. Uma vez que existem as fórmulas, poderíamos simplesmente usar uma delas ao invés de criar uma nova coluna.

Criamos uma medida chamada Faturamento Total que calcula a soma total de faturamento. Essa medida faz o **SUM** da coluna de Faturamento da Venda - se não tivéssemos essa coluna, não seria possível calcularmos o faturamento total. Vamos ver como podemos usar apenas o **SUMX** ao invés de criar uma nova coluna e depois somar seus valores.

Vamos criar uma medida chamada Faturamento Total X para fazer uma comparação. Nossa fórmula fica da seguinte maneira:

```
1 Faturamento Total X = SUMX(TotalVendas, TotalVendas[Tamanho Pedido] * TotalVendas[Preço do Produto])
```

Dentro dela já dizemos qual a operação deverá ser feita, linha a linha, para obtermos um resultado final - o valor de faturamento. Agora, se colocarmos nossa medida Faturamento Total X na nossa matriz, vemos que chegamos ao mesmo resultado que a medida que calculou a soma da coluna de Faturamento.

Soma Total	Faturamento Total X
R\$ 136.286.195,00	R\$ 136.286.195,00

Vemos então que é possível calcular o faturamento total sem necessariamente criar uma coluna extra na Base Vendas para isso.

6.4.2 AVERAGEX

Agora vamos usar a fórmula **AVERAGEX** para calcular a média de faturamento das vendas. Nós já havíamos criado uma medida chamada Média de Faturamento que realizava esse cálculo; porém, novamente, ela depende da coluna de faturamento das vendas. Usaremos a nossa fórmula X para acabar com essa dependência. A nossa medida Média de Faturamento X:

```
1 Média de Faturamento X = AVERAGEX(TotalVendas, TotalVendas[Tamanho Pedido] * TotalVendas[Preço do Produto])
```

Podemos verificar que, novamente, obtivemos o mesmo resultado:

Média de Faturamento	Média de Faturamento X
R\$ 5.727	R\$ 5.727,03

Mais uma vez vimos que a coluna Faturamento na base de vendas é desnecessária, dado que temos as fórmulas X.

6.4.3 MAXX/MINX

Temos outras duas medidas que usam a coluna de faturamento: Mínimo Faturado e Máximo Faturado. Recrie essas medidas, agora usando as fórmulas **MAXX** e **MINX**.

Fórmulas

Máximo Faturado X:

```
1 Máximo Faturado X = MAXX(TotalVendas, TotalVendas[Tamanho Pedido] * TotalVendas[Preço do Produto])
```

Mínimo Faturado X:

```
1 Mínimo Faturado X = MINX(TotalVendas, TotalVendas[Tamanho Pedido] * TotalVendas[Preço do Produto])
```

6.4.4 RANKX

A função **RANKX** é usada para atribuir um *ranking* às linhas em uma tabela ou expressão com base em um valor específico. Ela itera (ou seja, percorre linha a linha da tabela) sobre os registros e classifica as linhas com base no valor especificado.

A sintaxe básica da função RANKX é a seguinte:

RANKX(<tabela ou expressão>, <valor>, [<expressão de classificação>], [<ordem>], [<empate>])

O parâmetro **tabela/expressão** representa a tabela ou expressão que você deseja classificar, e o parâmetro **valor** é o valor específico pelo qual as linhas serão classificadas.

Opcionalmente, você pode fornecer uma <expressão de classificação> para determinar a ordem de classificação das linhas; Também é opcional fornecer uma <ordem> para especificar se a classificação é ascendente (1) ou descendente (0). Além disso, você pode fornecer um valor opcional <empate> para especificar como tratar os empates no ranking.

Por exemplo, vamos imaginar que queremos atribuir um *ranking* aos produtos de uma tabela de acordo com o valor das vendas.

A função ficaria da seguinte maneira:

RankingVendas = RANKX(Vendas, Vendas[Valor], , DESC)

Nesse exemplo, a função RANKX itera sobre a tabela "Vendas" e classifica as linhas com base no valor da coluna "Valor". Ela retornará o ranking para cada produto com base no valor das vendas, em ordem decrescente.

6.4.5 COUNTX

A função **COUNTX** é usada para contar o número de linhas em uma tabela ou expressão que atendem a uma determinada condição. Ela itera sobre os registros da tabela/expressão e conta quantos deles satisfazem a condição especificada.

A sintaxe básica da função:

COUNTX(<tabela ou expressão>, <condição>)

Enquanto o primeiro parâmetro representa a tabela ou expressão em que queremos contar as linhas, o segundo é uma expressão que define a condição que cada linha deve satisfazer para ser contada.

Por exemplo: vamos supor que você tenha uma tabela chamada "Vendas" com as colunas "Produto", "Região" e "Valor" e queremos contar quantas vendas tiveram um valor maior que R\$100. A função fica da seguinte maneira:

Vendas Acima 100 = COUNTX(FILTER(Vendas, Vendas[Valor] > 100), Vendas[Produto])

Nesse exemplo, a função COUNTX itera sobre a tabela "Vendas" após aplicar um filtro para selecionar apenas as vendas com valor acima de R\$100. Ela conta quantas linhas atendem a essa condição e retornará o resultado.

7 4.7 - Fórmulas de Filtro

Fórmulas de filtro que veremos:

- CALCULATE
- ALL
- ALLEXCEPT
- FILTER
- VALUES
- HASONEVALUE
- HASONEFILTER
- ISFILTERED
- SELECTEDVALUE

7.1 CALCULATE

Essa função é muito importante no DAX, pois ela permite que nós controlemos o contexto de avaliação dos cálculos. Dessa maneira conseguimos realizar cálculos como soma, média e etc. de algum valor aplicando de maneira simples um filtro - como somar todas as vendas de um determinado produto ou de determinada loja. Para quem é mais familiarizado com Excel, essa fórmula tem uma função muito parecida com as fórmulas CONT.SE(S) e SOMASE(S).

Para a primeira aplicação da nossa fórmula, vamos calcular a soma total de pedidos de **celulares** na base de vendas. Basicamente, queremos que a fórmula faça uma soma, mas só se o Tipo do Produto for **Celular**. Sabendo disso, nossa fórmula fica assim:

```
1 Vendas de Celular = CALCULATE(SUM(TotalVendas[Tamanho Pedido]), Produtos[Tipo do Produto] = "Celular")
```

Vamos ver a resposta e fazer uma comparação:

Tipo do Produto	Total Pedidos
Celular	11985
Fone de Ouvido	8953
Headset	8979
Monitor	8712
Notebook	11976
Smart Watch	5755
Tablet	5930
Video Game	9152
Total	71442

11985
Vendas de Celular

Exercício: faça uma matriz igual a matriz da imagem.

Total de Pedidos e Vendas no Final de Semana x Loja

Loja	Total Pedidos	Vendas no Final de Semana
Belo Horizonte	5070	1446
Brasília	4948	1503
Campinas	4950	1405
Cuiabá	9895	2794
Curitiba	5080	1487
Florianópolis	5273	1546
Fortaleza	5267	1475
Porto Alegre	5159	1430
Recife	5149	1481
Rio de Janeiro	5250	1495
Salvador	5242	1527
São Paulo	4957	1410
Vila Velha	5202	1409
Total	71442	20408

Fórmula da Medida

```
1 Vendas no Final de Semana = CALCULATE(SUM(TotalVendas[Tamanho Pedido]), Calendario[Fim de Semana] = "Fim de Semana")
```

7.2 ALL

O principal objetivo da fórmula **ALL** é cancelar filtros. Ela retorna todas as linhas de uma tabela ou todas as células de uma coluna, desconsiderando e ignorando qualquer filtro.

Para entender melhor, vamos pensar na seguinte situação: temos uma matriz com todas as lojas e o total de pedidos feitos em cada loja. O que queremos saber é o valor percentual de vendas de cada loja em relação ao todo.

Então, o que precisaríamos é de uma coluna com o total absoluto dos pedidos, para dividir o total por loja pelo total absoluto. Para conseguir esse total absoluto precisamos da fórmula ALL, de modo que, independente de a matriz estar dividida por loja, em cada uma das linhas, eu quero que ele retorne o total absoluto, e não o total filtrado, que vimos que é o que acontece quando colocamos a medida em uma tabela com filtros de linha.

Vamos aplicar o ALL junto com o CALCULATE. Primeiro, trabalharemos com uma matriz que tenha as Lojas nas linhas e a medida Total Pedidos em valores:

Loja	Total Pedidos
Belo Horizonte	5070
Brasília	4948
Campinas	4950
Cuiabá	9895
Curitiba	5080
Florianópolis	5273
Fortaleza	5267
Porto Alegre	5159
Recife	5149
Rio de Janeiro	5250
Salvador	5242
São Paulo	4957
Vila Velha	5202
Total	71442

Quando colocamos a medida Total Pedidos na nossa matriz, ele filtra por loja; só que para calcular o percentual de vendas de cada loja, precisamos dividir o total de vendas de cada loja pelo total absoluto. Obtemos o total absoluto através da fórmula CALCULATE + ALL:

```
1 Total Pedidos (Absoluto) = CALCULATE(TotalVendas[Total Pedidos],ALL(TotalVendas))
```

O primeiro argumento da fórmula é o que queremos somar e o segundo argumento vai ser o de Filtrar. Como queremos todos os valores da nossa base de vendas, usamos o ALL para tirar qualquer filtro.

Colocando dentro da matriz a medida que acabamos de criar, chegamos a esse resultado:

Loja	Total Pedidos	Total Pedidos (Absoluto)
Cuiabá	9895	71442
Florianópolis	5273	71442
Fortaleza	5267	71442
Rio de Janeiro	5250	71442
Salvador	5242	71442
Vila Velha	5202	71442
Porto Alegre	5159	71442
Recife	5149	71442
Curitiba	5080	71442
Belo Horizonte	5070	71442
São Paulo	4957	71442
Campinas	4950	71442
Brasília	4948	71442
Total	71442	71442

Agora vamos calcular o percentual das vendas:

```
1 Percentual de Vendas = TotalVendas[Total Pedidos] / TotalVendas[Total Pedidos (Absoluto)]
```

Agora sim:

Loja	Total Pedidos	Percentual de Vendas
Belo Horizonte	5070	7,10%
Brasília	4948	6,93%
Campinas	4950	6,93%
Cuiabá	9895	13,85%
Curitiba	5080	7,11%
Florianópolis	5273	7,38%
Fortaleza	5267	7,37%
Porto Alegre	5159	7,22%
Recife	5149	7,21%
Rio de Janeiro	5250	7,35%
Salvador	5242	7,34%
São Paulo	4957	6,94%
Vila Velha	5202	7,28%
Total	71442	100,00%

A vantagem de fazer isso é que quando aplicamos algum outro filtro na nossa matriz, os valores se dividem de maneira diferente, mas tendo o total absoluto é possível calcular os valores mesmo assim:

Ano	Total Pedidos	Percentual de Vendas
2020	18233	25,52%
2021	23207	32,48%
2022	30002	41,99%
Total	71442	100,00%

Exercício: criem a seguinte matriz agora:

Tipo do Produto	Total de Devoluções	Percentual de Devoluções
Celular	344	16,89%
Fone de Ouvido	209	10,26%
Headset	311	15,27%
Monitor	331	16,25%
Notebook	212	10,41%
Smart Watch	239	11,73%
Tablet	248	12,17%
Video Game	143	7,02%
Total	2037	100,00%

Resposta

Para criar o Total de Devoluções (Absoluto):

```
1 Total de Devoluções (Absoluto) = CALCULATE(SUM('Devoluções'[Quantidade Devolvida]), ALL('Devoluções'))
```

Percentual de Devoluções:

```
1 Percentual de Devoluções = [Total de Devoluções] / [Total de Devoluções (Absoluto)]
```

7.3 ALLEXCEPT

A função ALLEXCEPT no Power BI é usada para remover todos os filtros de contexto de uma tabela ou coluna, exceto os filtros aplicados em colunas específicas. Ela permite controlar quais colunas devem ser consideradas para o cálculo de uma medida, ignorando os filtros aplicados em outras colunas. Sua sintaxe é:

ALLEXCEPT(<tabela>, <coluna1>, <coluna2>, ...)

Onde **tabela** é a tabela na qual desejamos remover os filtros de contexto, e **coluna** representa as colunas para as quais queremos manter os filtros de contexto. A função ALLEXCEPT é comumente usada em medidas para isolar o cálculo em um subconjunto específico de colunas, ignorando os filtros aplicados em outras colunas. Isso permite criar cálculos mais granulares e controlados.

7.4 FILTER

A função FILTER filtra uma tabela de acordo com algum critério e retorna uma tabela menor só com as linhas que atendem ao critério. Vamos começar criando uma medida que vai retornar para nós quantas vendas tivemos acima da média de faturamento.

Criamos uma medida que armazena a média de faturamento. Pela lógica, é mais fácil usarmos essa medida para criar um filtro junto com a função CALCULATE e então usar a função COUNT para contarmos quantas vendas foram acima da média. Entretanto, temos um problema: a fórmula CALCULATE não aceita que uma medida seja utilizada no filtro. Logo, essa opção se torna inviável.

Escrever toda a fórmula de média novamente dentro de um CALCULATE também não é a melhor opção, pois nossa fórmula ficaria extensa e confusa; aqui entra a função **FILTER**, que usaremos combina a função CALCULATE para conseguirmos contar os valores com base no nosso filtro.

Primeiramente precisamos criar uma média absoluta, para impedir que seu valor seja alterado de acordo com as vendas:

```
1 Média Absoluta de Faturamento = CALCULATE([Média de Faturamento], ALL(TotalVendas))
```

Feito isso conseguimos contar os valores acima da média:

```
1 Vendas Acima da Média = CALCULATE(COUNT(TotalVendas[Preço do Produto]),  
2 FILTER(TotalVendas, TotalVendas[Faturamento da Venda] > [Média Absoluta de Faturamento]))
```

O que essa fórmula fez, basicamente, foi ir até a nossa base de vendas e aplicar um filtro de número!

Loja	Total Pedidos	Vendas Acima da Média
Belo Horizonte	5070	588
Brasília	4948	633
Campinas	4950	600
Cuiabá	9895	1184
Curitiba	5080	614
Florianópolis	5273	602
Fortaleza	5267	661
Porto Alegre	5159	641
Recife	5149	658
Rio de Janeiro	5250	651
Salvador	5242	645
São Paulo	4957	594
Vila Velha	5202	653
Total	71442	8724

Exercício: calculem a quantidade de vendas realizadas com preços acima da média. Para isso:

- Criem uma medida absoluta com a média de preços;
- Calculem a quantidade de produtos vendidos acima dessa média de preços

Resposta

Criando a medida absoluta:

```
1 Média Absoluta de Preços = CALCULATE(AVERAGE(Produtos[Preço Unitário]), ALL(TotalVendas))
```

Quantidade de produtos vendidos acima da média:

```
1 Vendas High Ticket = CALCULATE(TotalVendas[Número de Vendas], FILTER(TotalVendas, TotalVendas[Preço do Produto] > [Média Absoluta de Faturamento]))
```

7.5 VALUES

7.6 HASONEVALUE

7.7 HASONEFILTER

7.8 ISFILTERED

7.9 SELECTEDVALUE

8 4.8 - Fórmulas de Inteligência de Tempo

8.1 As fórmulas de inteligência de tempo no Power BI são um conjunto de funções e recursos que permitem realizar análises e cálculos relacionados a datas e horários. Elas ajudam a trabalhar com dados de tempo de forma mais eficiente e fornecem insights temporais em relatórios e visualizações.

Algumas das principais fórmulas de inteligência de tempo no Power BI incluem:

- **DATESYTD:** retorna um intervalo de datas representando o ano até a data especificada. Ela é útil para realizar cálculos de acumulação ao longo do ano, como somas acumuladas ou médias acumuladas.
- **DATESQTD:** usada para retornar um intervalo de datas representando o trimestre até a data especificada.
- **DATESMTD:** retorna um intervalo de datas representando o mês até a data especificada. Ela é útil para realizar cálculos de acumulação ao longo do mês, como somas acumuladas ou médias acumuladas.
- **DATEADD:** usada para adicionar um valor específico a uma data. Ela permite adicionar anos, trimestres, meses, semanas, dias, horas, minutos ou segundos a uma data.
- **DATESINPERIOD:** usada para criar um intervalo de datas com base em um período de tempo específico, como ano, trimestre, mês, semana ou dia.

8.2 DATESYTD

Para começarmos a trabalhar com essas fórmulas de inteligência de tempo, vamos criar uma matriz para analisar como está se comportando o faturamento ao longo dos meses.

Quando montamos uma matriz com o nome dos meses (coluna da base calendário), ele não diferencia os anos, apenas coloca os meses nessa matriz:

Nome do Mês

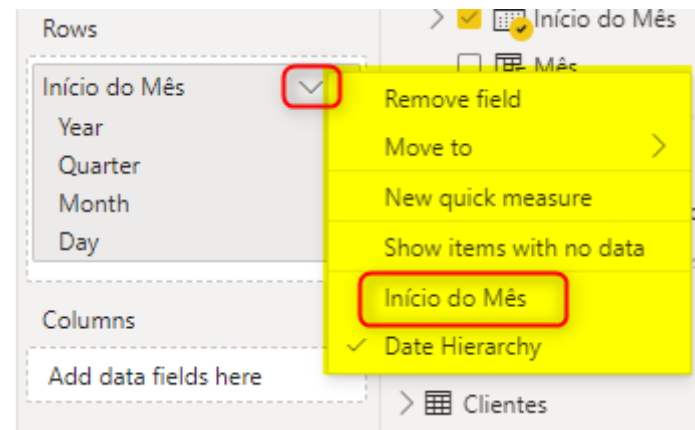
Abril
Agosto
Dezembro
Fevereiro
Janeiro
Julho
Junho
Maio
Março
Novembro
Outubro
Setembro

Na prática, quando precisarmos de um grau mais detalhado dessas datas, vamos usar a coluna de Início do Mês que criamos na nossa base. Colocando a coluna no nosso relatório:

Year

⊕ 2020
⊕ 2021
⊕ 2022

A princípio ele vai agrupar as datas por anos, mas conseguimos configurar isso para ele retirar esse agrupamento:



Ele desagrupou os anos, mas as datas ficaram em um formato estranho:

Início do Mês

quarta-feira, 1 de janeiro de 2020
sábado, 1 de fevereiro de 2020
domingo, 1 de março de 2020
quarta-feira, 1 de abril de 2020
sexta-feira, 1 de maio de 2020
segunda-feira, 1 de junho de 2020
quarta-feira, 1 de julho de 2020
sábado, 1 de agosto de 2020
terça-feira, 1 de setembro de 2020
quinta-feira, 1 de outubro de 2020
domingo, 1 de novembro de 2020
terça-feira, 1 de dezembro de 2020
sexta-feira, 1 de janeiro de 2021
segunda-feira, 1 de fevereiro de 2021
segunda-feira, 1 de março de 2021
quinta-feira, 1 de abril de 2021
sábado, 1 de maio de 2021
terça-feira, 1 de junho de 2021
quinta-feira, 1 de julho de 2021

Podemos alterar isso mudando a formatação da coluna na guia Dados.

Início do Mês
01/01/2020
01/02/2020
01/03/2020
01/04/2020
01/05/2020
01/06/2020
01/07/2020
01/08/2020
01/09/2020
01/10/2020
01/11/2020
01/12/2020
01/01/2021
01/02/2021
01/03/2021
01/04/2021

Agora sim!

Para ver o faturamento mês a mês, basta arrastar a nossa medida de Faturamento Total X (calculado com a fórmula X) para o valor da nossa matriz:

Início do Mês	Faturamento Total X
01/01/2020	R\$ 3.624.250,00
01/02/2020	R\$ 2.734.880,00
01/03/2020	R\$ 2.034.390,00
01/04/2020	R\$ 2.734.300,00
01/05/2020	R\$ 2.354.175,00
01/06/2020	R\$ 2.755.245,00
01/07/2020	R\$ 3.556.345,00
01/08/2020	R\$ 2.462.185,00
01/09/2020	R\$ 2.831.520,00
01/10/2020	R\$ 2.445.920,00
01/11/2020	R\$ 3.586.390,00
01/12/2020	R\$ 3.933.005,00
01/01/2021	R\$ 5.338.840,00
01/02/2021	R\$ 2.557.680,00
01/03/2021	R\$ 2.126.245,00
Total	R\$ 136.286.195,00

Dessa forma ele retorna para nós o **valor faturado em cada mês**. Para ver o valor **acumulado mês a mês**, vamos criar uma nova medida usando uma das nossas fórmulas de inteligência de tempo: o Faturamento Acumulado do Ano. A fórmula fica da seguinte maneira:

```
1 Faturamento Acumulado do Ano = CALCULATE(TotalVendas[Faturamento Total X], DATESYTD(Calendario[Data]))
```

A matriz com a nova medida fica assim:

▲ Início do Mês	Faturamento Total X	Faturamento Acumulado do Ano
01/01/2020	R\$ 3.624.250,00	R\$ 3.624.250,00
01/02/2020	R\$ 2.734.880,00	R\$ 6.359.130,00
01/03/2020	R\$ 2.034.390,00	R\$ 8.393.520,00
01/04/2020	R\$ 2.734.300,00	R\$ 11.127.820,00
01/05/2020	R\$ 2.354.175,00	R\$ 13.481.995,00
01/06/2020	R\$ 2.755.245,00	R\$ 16.237.240,00
01/07/2020	R\$ 3.556.345,00	R\$ 19.793.585,00
01/08/2020	R\$ 2.462.185,00	R\$ 22.255.770,00
01/09/2020	R\$ 2.831.520,00	R\$ 25.087.290,00
01/10/2020	R\$ 2.445.920,00	R\$ 27.533.210,00
01/11/2020	R\$ 3.586.390,00	R\$ 31.119.600,00
01/12/2020	R\$ 3.933.005,00	R\$ 35.052.605,00
01/01/2021	R\$ 5.338.840,00	R\$ 5.338.840,00
01/02/2021	R\$ 2.557.680,00	R\$ 7.896.520,00
01/03/2021	R\$ 2.126.245,00	R\$ 10.022.765,00
Total	R\$ 136.286.195,00	R\$ 56.751.340,00

Como podemos ver, ela nos mostra o total acumulado durante os meses ano a ano.

8.3 DATEADD

Agora vamos trabalhar com uma fórmula que tem como objetivo basicamente fazer uma comparação com um período anterior.

Por exemplo: podemos fazer uma comparação entre o mês de julho de 2019 com julho de 2018, ou então fazer uma comparação entre o mês de maio de 2019 com o mês de abril de 2019.

Como primeiro passo vamos criar a nossa medida. Como queremos saber sempre o valor faturado do mês anterior, vamos usar a fórmula **DATEADD** da seguinte maneira:

```
1 Faturamento do Mês Anterior = CALCULATE(TotalVendas[Faturamento Total X],DATEADD(Calendario[Data], -1, MONTH))
```

Basicamente, a fórmula vai calcular o faturamento total e vai mostrar sempre o faturamento do mês anterior. Vamos colocar essa medida na nossa matriz que o resultado ficará mais claro:

Início do Mês	Faturamento Total X	Faturamento do Mês Anterior
01/01/2020	R\$ 3.624.250,00	
01/02/2020	R\$ 2.734.880,00	3.624.250,00
01/03/2020	R\$ 2.034.390,00	2.734.880,00
01/04/2020	R\$ 2.734.300,00	2.034.390,00
01/05/2020	R\$ 2.354.175,00	2.734.300,00
01/06/2020	R\$ 2.755.245,00	2.354.175,00
01/07/2020	R\$ 3.556.345,00	2.755.245,00
01/08/2020	R\$ 2.462.185,00	3.556.345,00
01/09/2020	R\$ 2.831.520,00	2.462.185,00
01/10/2020	R\$ 2.445.920,00	2.831.520,00
01/11/2020	R\$ 3.586.390,00	2.445.920,00
01/12/2020	R\$ 3.933.005,00	3.586.390,00
01/01/2021	R\$ 5.338.840,00	3.933.005,00
01/02/2021	R\$ 2.557.680,00	5.338.840,00
01/03/2021	R\$ 2.126.245,00	2.557.680,00
Total	R\$ 136.286.195,00	129.661.920,00

Vamos pensar em uma utilidade para essa coluna:

Imagina que a nossa empresa tem uma meta de crescimento de 5% ao mês. Isso significa que o faturamento de 02/16 deve ser 5% maior que o de 01/16; o faturamento de 03/16 deve ser 5% maior que o de 02/16 e assim sucessivamente.

Vamos criar uma medida chamada Meta de Faturamento Mensal que vai calcular essa meta para nós:

```
1 Meta de Faturamento Mensal = [Faturamento do Mês Anterior] * 1.05
```

Criada a nossa coluna, podemos montar a matriz:

Início do Mês	Meta de Faturamento Mensal	Faturamento Total X
01/01/2020		R\$ 3.624.250,00
01/02/2020	R\$ 3.805.462,50	R\$ 2.734.880,00
01/03/2020	R\$ 2.871.624,00	R\$ 2.034.390,00
01/04/2020	R\$ 2.136.109,50	R\$ 2.734.300,00
01/05/2020	R\$ 2.871.015,00	R\$ 2.354.175,00
01/06/2020	R\$ 2.471.883,75	R\$ 2.755.245,00
01/07/2020	R\$ 2.893.007,25	R\$ 3.556.345,00
01/08/2020	R\$ 3.734.162,25	R\$ 2.462.185,00
01/09/2020	R\$ 2.585.294,25	R\$ 2.831.520,00
01/10/2020	R\$ 2.973.096,00	R\$ 2.445.920,00
01/11/2020	R\$ 2.568.216,00	R\$ 3.586.390,00
01/12/2020	R\$ 3.765.709,50	R\$ 3.933.005,00
01/01/2021	R\$ 4.129.655,25	R\$ 5.338.840,00
01/02/2021	R\$ 5.605.782,00	R\$ 2.557.680,00
01/03/2021	R\$ 2.685.564,00	R\$ 2.126.245,00
Total	R\$ 136.145.016,00	R\$ 136.286.195,00

Com essa análise observamos que em diversos meses o faturamento ficou abaixo da meta. Essa é uma das maneiras que podemos usar as medidas que acabamos de calcular nas análises da nossa empresa.

8.4 DATESINPERIOD

O objetivo dessa fórmula é calcular o acumulado de um certo período. Qual é o acumulado das vendas nos últimos 10 dias? Ou nos últimos 3 meses?

Vamos criar uma medida para 15 dias de faturamento:

```
1 15 dias de Faturamento = CALCULATE([Faturamento Total X], DATESINPERIOD(Calendario[Data], MAX(Calendario[Data]), -15, DAY))
```

Explicando a fórmula:

DATESINPERIOD(Dates, StartDate, NumberOfIntervals, Interval)

Returns the dates from the given period.

- *Dates*: lista de datas do nosso calendário;
- *StartDate*: usamos a função MAX para que ela retorne a data máxima da linha em questão, pensando em um intervalo de datas que vai do dia inicial até o dia em questão - ou seja, a própria data relacionada a linha;
- *NumberOfIntervals*: número que queremos avançar (+) ou recuar (-) no tempo a partir da data inicial;
- *Interval*: intervalo de tempo que queremos fazer a soma; no nosso caso, os últimos 15 dias.

Resumindo, o que a nossa fórmula vai fazer é pegar o dia atual, voltar 15 dias a partir do dia atual e calcular a soma do faturamento de forma acumulada.

Para visualizar o resultado vamos criar uma matriz com a coluna de Data nas linhas e os faturamentos Total X e 15 dias em valores. A matriz fica da seguinte maneira:

Year	Faturamento Total X	15 dias de Faturamento
⊕		R\$ 216.435,00
⊕ 2020	R\$ 35.052.605,00	R\$ 1.915.140,00
⊕ 2021	R\$ 44.482.250,00	R\$ 2.564.635,00
⊕ 2022	R\$ 56.751.340,00	R\$ 3.222.275,00
Total	R\$ 136.286.195,00	R\$ 3.222.275,00

Repita os passos que vimos anteriormente para chegar na seguinte tabela:

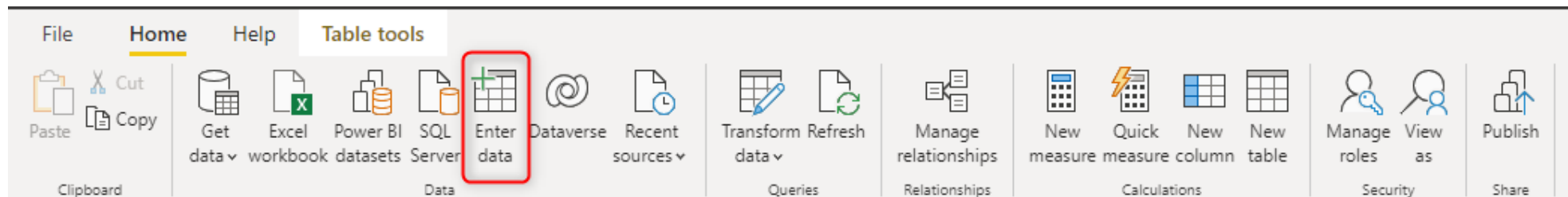
Data	Faturamento Total X	15 dias de Faturamento
		R\$ 216.435,00
01/01/2020	R\$ 216.435,00	R\$ 216.435,00
02/01/2020	R\$ 171.395,00	R\$ 387.830,00
03/01/2020	R\$ 165.330,00	R\$ 553.160,00
04/01/2020	R\$ 161.445,00	R\$ 714.605,00
05/01/2020	R\$ 115.125,00	R\$ 829.730,00
06/01/2020	R\$ 108.675,00	R\$ 938.405,00
07/01/2020	R\$ 128.125,00	R\$ 1.066.530,00
08/01/2020	R\$ 58.915,00	R\$ 1.125.445,00
09/01/2020	R\$ 149.200,00	R\$ 1.274.645,00
10/01/2020	R\$ 113.325,00	R\$ 1.387.970,00
11/01/2020	R\$ 115.100,00	R\$ 1.503.070,00
12/01/2020	R\$ 73.850,00	R\$ 1.576.920,00
13/01/2020	R\$ 140.360,00	R\$ 1.717.280,00
14/01/2020	R\$ 91.040,00	R\$ 1.808.320,00
15/01/2020	R\$ 136.990,00	R\$ 1.945.310,00
16/01/2020	R\$ 72.875,00	R\$ 1.801.750,00
17/01/2020	R\$ 96.625,00	R\$ 1.726.980,00
18/01/2020	R\$ 170.095,00	R\$ 1.731.745,00
Total	R\$ 136.286.195,00	R\$ 3.222.275,00

Nesse ponto, chegamos ao resultado da soma acumulada, sempre 15 dias para trás. Podemos editar a fórmula para colocar a quantidade de dias que quisermos, fazendo o que chamamos de um acumulado móvel.

9 4.9 - Organizando Medidas, Resumo das Principais Fórmulas e Boas Práticas

9.1 Organizando Medidas

Para organizar melhor nosso arquivo de Power BI, vamos aprender a armazenar todas as medidas criadas em um único local. Em qualquer uma das abas, vamos até a **Página Inicial (Home)** e selecionamos **Inserir Dados (Enter Data)**:



Na janela que será aberta, podemos mudar o nome da tabela que será criada. Aqui a chamaremos de **Medidas**:

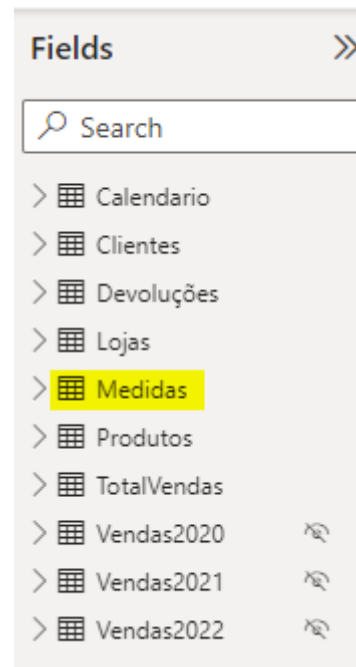
Create Table



	Column1	+
1		
+		

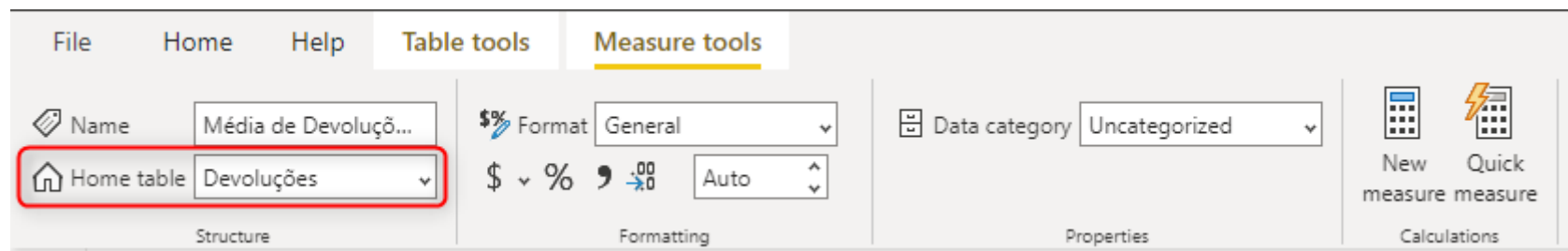
Name: Medidas

Essa tabela será colocada na aba **Dados** (*Fields*):



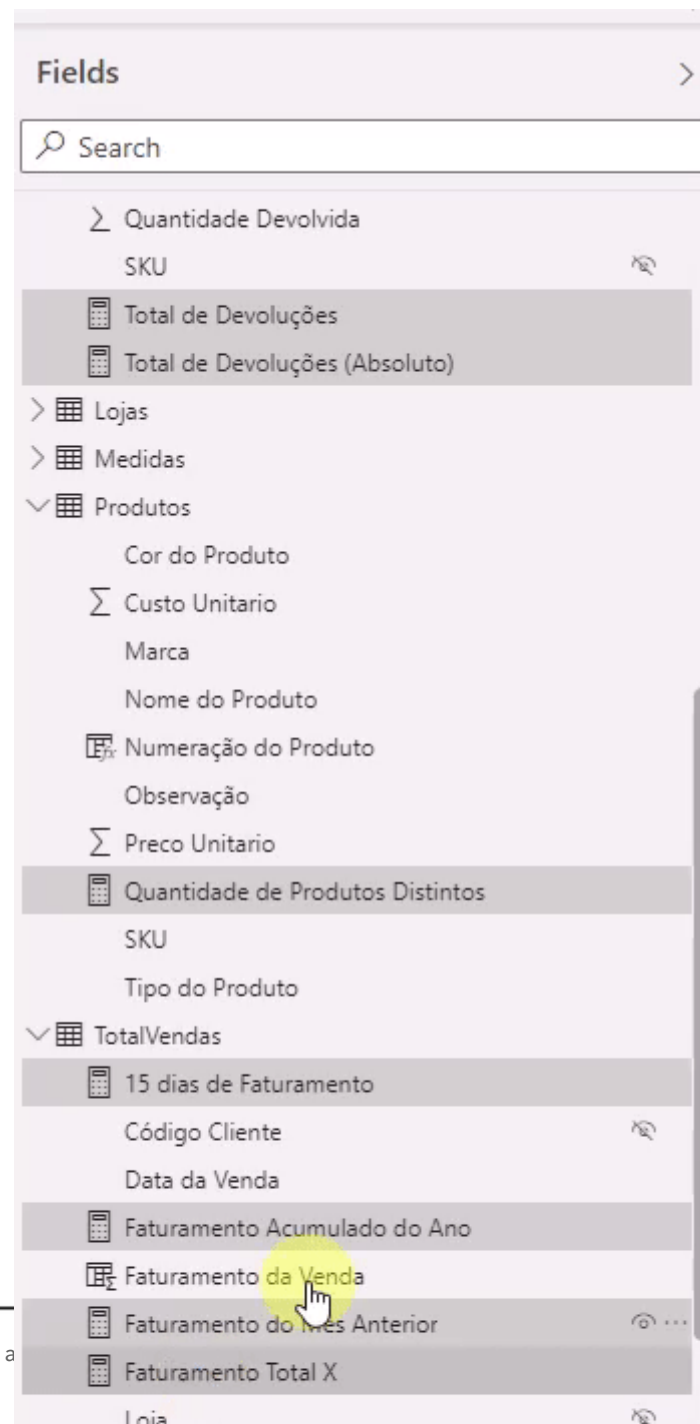
Feito isso, temos duas formas de mudar as medidas para a nova tabela criada:

1: Quando clicamos sobre uma medida, a aba **Ferramentas de medida** (*Measure tools*) aparece para nós; entrando nessa aba, conseguimos mudar a tabela onde se localiza a medida no campo **Tabela inicial** (*Home table*):



O problema é que, desta forma, temos que mudar a localização de uma medida de cada vez.

2: Na aba **Modelo** (*Model*) conseguimos selecionar várias medidas ao mesmo tempo clicando nelas com o botão *ctrl* pressionado. Sabendo disso, podemos selecionar todas as medidas e arrastá-las até a nossa tabela criada, alterando todas elas de lugar.



Feito isso, podemos excluir a **Coluna 1** (*Column 1*) que foi criada quando criamos a tabela.

Algumas medidas que usavam outras medidas na expressão apareceram com erro, pois como elas mudaram de lugar essa informação está desatualizada:

```
1 Percentual de Vendas = TotalVendas[Total Pedidos] / TotalVendas[Total Pedidos (Absoluto)]
```

! Column 'Total Pedidos' in table 'TotalVendas' cannot be found or may not be used in this expression.

Para corrigir esse erro basta alterar a base de dados nas expressões!

```
1 Faturamento Acumulado do Ano = CALCULATE([Faturamento Total X], DATESYTD(Calendario[Data]))
```

Agora temos nossas medidas organizadas em uma única tabela, o que facilita o trabalho quando formos utilizar elas na criação dos gráficos!!
Elas vão inclusive aparecer na parte superior dos nossos dados, o que facilita a localização!

9.2 Fórmulas de Texto

LEN	Retorna o número de caracteres de um texto	=LEN (Texto) Ex: =LEN ("Alon") -> 4
CONCATENATE ou &	Concatena vários textos em um só	= Texto1 & Texto2 & Texto3 ... Ex: = "Alon" & " " & "Pinheiro" -> Alon Pinheiro
LEFT/MID/RIGHT	Retorna caracteres da esquerda/meio/direita	=LEFT (Texto; Núm_Caracteres) =MID (Texto;Pos_Inicial;Qtd_Caracteres) Ex: =LEFT ("Alon Pinheiro"; 4) -> Alon Ex: =MID ("Diego Amorim Santos";7;6) -> Amorim
UPPER/LOWER	Converte o texto em MAIÚSCULO/minúsculo	=UPPER (Texto) =LOWER (Texto) Ex: =UPPER ("Alon") -> ALON
SUBSTITUTE	Substitui um conjunto de caracteres por outro	=SUBSTITUTE (Texto; Texto Antigo; Texto Novo) Ex: =SUBSTITUTE ("João Paulo Martins";"Martins";"Lira") -> João Paulo Lira
SEARCH	Retorna a posição de um caracter no texto	=SEARCH (Texto Procurado; No Texto) Ex: =SEARCH ("P";"Alon Pinheiro") -> 6

9.3 Fórmulas Matemáticas

SUM	Soma os valores de uma coluna	= SUM (Nome_da_Coluna) Ex: = SUM ([Tamanho Pedido])
AVERAGE	Retorna a média (aritmética) dos valores de uma coluna	= AVERAGE (Nome_da_Coluna) Ex: = AVERAGE ([Faturamento])
MAX	Retorna o maior valor de uma coluna	= MAX (Nome_da_Coluna) Ex: = MAX ([Quantidade Devolvida])
MIN	Retorna o menor valor de uma coluna	= MIN (Nome_da_Coluna) Ex: = MIN ([Preço do Produto])
DIVIDE	Faz uma divisão entre dois valores	= DIVIDE (Numerador; Denominador; Resultado Alternativo) Ex: = DIVIDE (SUM[Custo Unitário]; SUM[Preço Unitário]; 0)

9.4 Fórmulas de Contagem

COUNT

Conta a quantidade de células em uma coluna que contém números

=**COUNT**(Nome_da_Coluna)
Ex: =**COUNT**([Tamanho Pedido])

COUNTA

Conta a quantidade de células preenchidas em uma coluna (numérico e não numérico)

=**COUNTA**(Nome_da_Coluna)
Ex: =**COUNTA**(['Nome do Cliente'])

DISTINCTCOUNT

Conta a quantidade de valores únicos em uma coluna (sem as duplicatas)

=**DISTINCTCOUNT**(Nome_da_Coluna)
Ex: = **DISTINCTCOUNT**([SKU])

COUNTROWS

Conta a quantidade de linhas em uma tabela

=**COUNTROWS**(Nome da Tabela)
Ex: =**COUNTROWS**('Cadastro Produtos')

9.5 Fórmulas de Inteligência de Tempo

**FÓRMULAS de
Inteligência de Tempo**

Permitem criar comparações e análises inteligentes como acumulado do ano, acumulado dos últimos 10 dias, comparação com o mesmo período do ano anterior, etc

**Acumulado do ano
até uma data**

=**CALCULATE**(Medida; **DATESYTD**(Calendario[Data]))

**Período
anterior/posterior**

=**CALCULATE**(Medida; **DATEADD**(Calendario[Data]; -1; **MONTH**))

**Acumulado de um
período**

=**CALCULATE**(Medida;
DATESINPERIOD(**MAX**(Calendario[Data]); -15; **DAY**))

Escolha um intervalo (dias, meses, anos...) e a quantidade

9.6 Aula 4 - Boas Práticas

Colunas Calculadas x Medidas

- Só use Colunas Calculadas quando você quiser calcular um valor diferente para cada linha da tabela;
- Quando quiser fazer cálculos simples, que retornam um único valor, como soma, média, contagem, use sempre uma Medida.

Quebre os cálculos em partes pequenas

- Quando for possível, evite criar várias fórmulas uma dentro da outra. Isso pode deixar a fórmula mais confusa de ler e complexa de mexer.
-