Microsoft

Microsoft Power BI Visual Calculations

Simplifying DAX

Jeroen ter Heerdt | Madzy Stikkelorum | Marc Lelijveld

Practice and solution files on the web

FREE SAMPLE CHAPTER

Microsoft Power BI Visual Calculations: Simplifying DAX

Jeroen ter Heerdt
Madzy Stikkelorum
Marc Lelijveld

Microsoft Power BI Visual Calculations: Simplifying DAX

Published with the authorization of Microsoft Corporation by:

Pearson Education, Inc.

Hoboken, New Jersey

ISBN-13: 978-0-13-539692-6
ISBN-10: 0-13-539692-1

Library of Congress Control Number: 2025946899

$PrintCode

Trademarks

Warning and Disclaimer

EDITOR-IN-CHIEF
Julie Phifer

EXECUTIVE EDITOR
Loretta Yates

ACQUISITIONS EDITOR
Shourav Bose

DEVELOPMENT EDITOR
Kate Shoup

MANAGING EDITOR
Sandra Schroeder

SENIOR PROJECT EDITOR
Tracey Croom

COPY EDITOR
Charlotte Kughen

TECHNICAL EDITOR
Jeffrey Wang

INDEXER
Rachel Kuhn

PROOFREADER
Barbara Mack

COVER DESIGNER
Twist Creative, Seattle

COVER ILLUSTRATION
knssr/stock.adobe.com

COMPOSITOR
codeMantra

GRAPHICS
codeMantra

Dedication

For my wife, Ilse. You selflessly joined me on an adventure moving halfway across the globe, uprooting yourself from everything and everyone to allow me to fulfill my goals, which, among many other things, resulted in me working on visual calculations and writing this book. I am blessed with your unwavering support and love. We started on this adventure as a duo, and we embark on the next chapter as a trio with our daughter Chloë, and I couldn't be more excited.

–Jeroen

For my husband, Wouter, who has put up with me working on this book for so many evenings, weekends, and during nap times of our son without any complaints. You supported me through it all, taking our son for short outings so I could write, making sure I had the comfortable chair when my pregnant belly grew and moved my keyboard ever further away. Thank you for giving me the space to finish this book and even considering Dax as a name for our second son. I'm not saying that's going to be it, though!

–Madzy

For my wife, Kirsten, thank you for your patience, encouragement, and love during the late nights and long weekends spent writing this book. Your support carried me through the tougher moments, and I'm endlessly grateful for having you by my side. For my dad, Martin, your pride and unwavering belief in me have been a constant source of motivation since the start of my career.

–Marc

Contents at a Glance

Contents

This page intentionally left blank

Foreword

DAX is a critical piece of Power BI. It allows authors to build reusable semantic models to power exploration, reporting, and AI. DAX measures express business logic as dynamic, reusable calculations that work across a variety of dimensions and scenarios. However, DAX has a steep learning curve for new users.

Some of the concepts that make DAX so powerful—like filter context and the many features of semantic models—also make DAX challenging to learn. It demands clear thinking about row and filter context, which is hard for beginners, because it asks them to ignore the visual in front of them and reason in the abstract space of the semantic model.

Many common reporting and visualization tasks shouldn't require a complete understanding of these concepts. Tasks like showing the difference between two metrics or adding a running sum to a chart are easier to do using the data in a visualization. These ideas led us to add visual calculations to Power BI. Instead of pushing every calculation into measures that operate independently over base tables, you can work with the result set the visual already produced. That shift narrows the gap between how analysts think and how the engine executes.

Visual calculations are expressions defined as part of a visualization, and they work over the data in that visualization. This lets you write a calculation using specific numbers you can see rather than imagining how filter context might create those numbers from the semantic model. Visual calculations provide functions that already understand the structure of the visual to let you easily write running sums or moving averages. These are just two of the ways visual calculations make it easier to build powerful visualizations in Power BI.

There was a long path to visual calculations. Years before the team debated the investment, Amir Netz asked Jeffrey Wang to look at a performance escalation in a customer DirectQuery model. A measure was generating slow SQL. The customer was confused, and they were right to be. The visual already showed a small set of values from another measure—values that contained everything needed for the more complex result. Yet what looked simple on the canvas triggered a cascade of large, slow, and convoluted SQL. The gap between what is visibly simple and what the engine had to compute was a persistent source of friction. Incidents like this pushed us toward an approach that simplifies the mental model and speeds evaluation by operating at the level of the visual when it makes sense. It still took years and foundational work, including DAX window functions, before visual calculations officially took off.

Both newcomers and seasoned DAX users benefit from this approach. Newcomers can reason from what is on the screen and learn many DAX features without starting in the deep end. Experts can reduce model clutter by moving visual-specific calculations, such as formatting, into the visual and achieve more predictable cost in DirectQuery and composite models. *Microsoft Power BI Visual Calculations: Simplifying DAX* explains how visual calculations work, where they shine, how to keep performance stable as visuals grow more complex, and when classic measures remain the right tool. Visual calculations are still DAX expressions, and the power of DAX is still available when you need it. DAX measures are still a critical tool for building reusable semantic models that are the foundation of exploration, reporting, and AI in Power BI. However, visual calculations give you a way to more easily accomplish many common reporting and visualization tasks.

Use this book to shorten the path from a question to an answer, and to teach patterns your team can apply consistently. That is the value of visual calculations in Power BI. We hope this book helps you take advantage of visual calculations to accomplish your goals with Power BI.

Marius Dumitru, Partner Architect

John Vulner, Partner Architect

Jeffrey Wang, Partner Architect

Microsoft Power BI

Acknowledgments

We would like to express our sincere gratitude to all the people who have supported us while writing this book. Without their help and encouragement, this book would not have been possible. This includes the folks at Pearson/Microsoft Press: Loretta Yates and Shourav Bose for believing there was an audience for this and keeping us on track, and Kate Shoup, who turned our drafts into the book you are reading!

Visual calculations is a powerful and complex feature precisely because of its simplicity. It took years to get it to where it is now. We want to thank the entire Power BI visual calculations team at Microsoft for their efforts to make DAX easier for the rest of us.
It all started with conversations at Microsoft between Jeroen and three architects: Jeffrey Wang, Marius Dumitru, and John Vulner. Without the guidance of Marius, Jeffrey, and John, visual calculations would not have existed.

We would like to thank Jeffrey for being our technical reviewer, keeping us honest and on track. Without him, this book would mostly be a collection of ambiguous information. Any mistakes still left in the book are solely because of the authors.

We'd also like to give a special thanks to Fowmy Abdulmuttalib, Injae Park, and Erik Svensen for showing the Power BI community what visual calculations can do and allowing us to use their examples in our book.

The Power BI community is the warmest and most welcoming community any of us have had the good fortune to be a part of. Its global reach, diversity, and passion are unparalleled. We are grateful that we're members of such an awesome collective. We learn so much from you all every day and look forward to sharing and learning for many years to come.

We want to thank you, the reader, for your interest in visual calculations. Our goal with this book is twofold. First, we want to help you be successful with Power BI by enabling you to do calculations using visual calculations; we hope that when you finish this book, you'll be ready to face any challenge with visual calculations. And second, we want to inspire you to explore new, innovative ways to use visual calculations while also looking beyond them toward the wide world of DAX and Power BI, where the possibilities are truly endless. We welcome your feedback and comments, and if you create a great visual calculation, tell us. We look forward to hearing from you!

Jeroen ter Heerdt
Duvall, Washington, United States of America
Madzy Stikkelorum
De Meern, Utrecht, The Netherlands
Marc Lelijveld
Haastrecht, Zuid-Holland, The Netherlands

This page intentionally left blank

About the Authors

Jeroen ter Heerdt Jeroen (Jay) ter Heerdt is a principal product manager on the Power BI team at Microsoft and one of the masterminds behind visual calculations. With a focus on advancing DAX and modeling in Power BI, Jeroen is passionate about delivering powerful features and improvements that empower users. Known as the “Dutch Data Dude,” he is a seasoned speaker and advocate for data-driven insights, captivating audiences with his entertaining and thought-provoking presentations. Known for incorporating unique and delightfully “weird” elements into his keynotes, Jeroen leaves a memorable impression every time he takes the stage. You can find him sharing insights and connecting with the community on social media as @jaypowerbi, on Reddit as @dutchdatadude, on LinkedIn, and through his website, *https://www.dutchdatadude.com*. He frequently reminds himself to return to blogging more regularly. When not immersed in the world of data, Jeroen enjoys experimenting with artisanal food and drinks, home automation, and exploring the beauty of the Pacific Northwest.

Madzy Stikkelorum Madzy Stikkelorum has a master’s degree in Mathematics, and analyzing data has always been one of her passions. She currently works as a data analytics consultant at Quanto, where she enjoys helping clients solve complex problems with Microsoft Power BI solutions. She shares her love for Power BI and DAX with the community by organizing the Power BI Summer School in the Netherlands, as a public speaker at international conferences, and in blog posts, which can be found at *https://quanto.eu/blog/*. Her very first video training about DAX, created with Michiel Rozema, launched in 2025 for Microsoft Press. Madzy is married to Wouter, and they already had one son together, but during the writing of this book, their second son was baking in the oven—inspiring Madzy to ensure she met the deadlines for her chapters! In her spare time, she loves to read, bake, and work on creative projects of all kinds. This is her first ever book as a co-author; she is extremely proud to have a book of her own making in her bookcase and is sure that more will follow.

Marc Lelijveld Marc Lelijveld is a Microsoft Data Platform Most Valuable Professional (MVP), solution architect, and technical evangelist with a passion for turning data into action. He focuses on Power BI and Microsoft Fabric, helping others make the most of these technologies through practical insights and real-world experience. Recognized as a Microsoft MVP since 2018, Marc is a regular speaker at international conferences, user groups, and community events. He's known for sharing his knowledge, best practices, and contagious enthusiasm—whether he's presenting deep technical content or inspiring others to level up their data journey. Marc blogs about Power BI and Microsoft Fabric at *https://data-marc.com* and can be found on most social platforms under his own name. When he's not working with data, you'll probably find him out exploring the world on one of his many bikes.

Introduction

"Data is the new oil!"

It wasn't too long ago that every consultancy and technology firm in the data space used that phrase, officially or unofficially. What they meant was that data is as important to the data revolution—driven by big data, massive data, and humongous data—as the use of oil was to the industrial revolution. In this exciting new era, all you needed was data, to buy their tools, and/or to hire their consultants, and your organization would magically be data driven!

If only that turned out to be true.

Data may indeed be the new oil. But just like crude oil, data needs refining. It must be transformed into information. This is why we clean, combine, model, and visualize data. The output of all this work—whether you do it on your own, get some help, or use a (semi-)automatic process—includes reports and dashboards that provide insights into various aspects of the organization's dealings, which decision-makers can then consume to make critical business decisions.

Ideally, your organization's data contains the answers needed to make informed decisions. But that data must be teased out—brought to the forefront. Adding calculations to the data is one way to achieve this. For example, providing a humble sum at the end of a list of sales transactions enables someone to understand how much was sold simply by looking at a single number rather than performing mental or computer-aided math to add up individual sales-transaction amounts.

Together with effective data cleansing, data modeling, and data visualization, calculations form the toolset to refine data into much needed information. However, this book focuses only on calculations—not on calculations in general, but rather on a specific way of performing calculations (visual calculations) in specific software (Power BI) that can help you transform your business into one that is data driven.

We assume you already know the basics of Power BI—or at the very least, even if you have a background in Excel and are a bit scared of using Power BI, you understand its role—so we do not include an exhaustive introduction to the software. We also assume you've heard a thing or two about DAX (spoiler alert: people say it's hard!) and perhaps decided it's not for you. However much you do or don't know about Power BI and DAX, it's OK. We're here to help. So, let's not delay any further and jump in.

What is DAX?

Data Analysis Expressions (DAX) is the native formula and query language for Microsoft PowerPivot, Power BI, and SQL Server Analysis Services Tabular models. This book focuses on the use of DAX with Power BI for reasons that will soon become apparent.

DAX is one of two languages you might use in Power BI to add calculations to your data. The other language is M. However, although you can do calculations with M, it's best suited for data transformations, such as data cleansing or grouping. If you must choose between M and DAX, follow this rule: Perform calculations in DAX—which is by far the superior option in terms of capabilities and speed—and transformations in M. Only deviate from this rule if you really must.

At first glance, DAX looks a lot like Excel formulas. It follows a similar structure in that you define an expression that applies functions to data and returns some result. In fact, many of the function names are the same. IF is IF, SUM is SUM, AVERAGE is AVERAGE, and so on. They do the same (or very similar) things. However, there are numerous functions that don't exist in Excel but do exist in DAX, and vice versa.

The fact that DAX looks a lot like Excel formulas is both a blessing and a curse. It's a blessing because new users, who most likely have some prior knowledge of Excel (because who doesn't?), feel empowered to add the calculations they need. But it's a curse because hidden beneath the seemingly simple, familiar expression syntax are concepts that are so alien to Excel that they might have come from a distant star. After dipping their toes into DAX waters, new users quickly discover that DAX supports functions they know nothing about, and that familiar functions they *do* know about from Excel are completely missing in DAX.

In some ways, it might be better if DAX did not seem so similar to Excel. Yes, it would have added hurdles in adoption, and users would have had to learn a new language. But beyond very basic usage, users still need to learn a new language because of the conceptual differences. The similarities in syntax provide a false sense of security, encouraging users to maintain an Excel mindset when using DAX and Power BI instead of preparing them to learn a fresh perspective on performing calculations.

What this book is about

This book is about visual calculations, a feature of Microsoft Power BI, aimed at making DAX easier to work with. Development of this feature started in June 2020, with visual calculations entering public preview in February 2024. At the time this book went to print, the visual calculations feature was still in preview; therefore, things might have changed by the time you read this. (We indicate features of visual calculations for which we anticipate changes in the book.)

Organization of this book

Here are the contents of this book at a glance:

Part I: Visual calculations fundamentals This part covers the fundamentals of visual calculations and consists of the following chapters:

- **Chapter 1: Introduction to visual calculations** This chapter introduces visual calculations and explains the problems with DAX that they are meant to solve.
- **Chapter 2: My first visual calculation** This chapter provides a hands-on tutorial for building your first visual calculation.
- **Chapter 3: Visual calculations concepts** In this chapter, you'll learn the core concepts behind visual calculations.
- **Chapter 4: Organization and use of visual calculations** This chapter shows you how to work with visual calculations successfully and how to govern them.

Part II: Visual calculations functions This part dives into the details of visual calculations exclusive functions and their foundations in DAX window functions. Because this is a hefty topic, it's divided into three chapters, so you can more easily find what you're looking for:

- **Chapter 5: The pillars: supportive functions** This chapter introduces supportive functions, such as ORDERBY, that pave the way for the foundational functions.
- **Chapter 6: The foundation: foundational functions** This chapter focuses on the foundational functions that form the basis of the functions that are exclusive to visual calculations.
- **Chapter 7: The floors and rooms: visual calculations exclusive functions** This chapter discusses the exciting functions that are exclusive to visual calculations.

Part III: Mastering visual calculations This part helps you master visual calculations. It consists of the following chapters:

- **Chapter 8: Comparing calculation options** This chapter compares the various calculation options available to you in Power BI and discusses what to use when.

- **Chapter 9: Breaking down visual calculations execution** This chapter explores the inner workings of visual calculations and their performance characteristics.

Part IV: The art of practical visual calculations This final part focuses on practical applications of visual calculations and provides a detailed description of various use cases. It includes the following chapters:

- **Chapter 10: Applied visual calculations** This chapter provides practical, real-world examples of how to use visual calculations to implement calculations that can be applied to many visuals.
- **Chapter 11: Charting with visual calculations** Here, we explore how to create specific visuals that are missing from Power BI with the help of visual calculations.

Chapter 42: Beyond the page Evaluates the goals we had for this book and provides guidance on what's next.

Appendices Provides code snippets that support and expand on the main content.

Who should read this book?

This book is for anyone who wants to use the visual calculations feature of Power BI, as well as those who seek a deeper understanding of visual calculations, their foundation, and their inner workings. We present the DAX required to gain that understanding, but by no means is this book intended to cover everything there is to know about DAX. If learning DAX is your goal, there are way better materials available to you than this book.

We expect you, our reader, to have a basic knowledge of Power BI and some experience in the analysis of numbers. If you have already been exposed to DAX, this may be helpful because you might be able to skip some sections. However, previous experience with DAX is not required. Having used Excel formulas before will help.

Conventions and features in this book

This book uses the following conventions:

- We use US notation for numbers, such as a period (.) for decimal separators and a comma (,) for thousand separators.
- Power BI features (such as visual calculations, measures, and so on) are not capitalized.

- DAX functions are uppercase. For example: CALCULATE and RUNNINGSUM.
- Values supplied to parameters are uppercase. For example, HIGHESTPARENT and DEFAULT.
- Code is presented as follows:

  `TheAnswerToTheUltimateQuestionOfLifeTheUniverseAndEverything = 42.`
- Names of measures, visual calculations, and calculated columns are in italic: *Total Sales* measure, *VsPrevious* visual calculation.
- Parameter names are in italic: *axis*, *reset*.
- Whenever we reference a value you can find in the dataset, it's capitalized: Bikes, Yellow, Security products.
- Direct references to columns and tables are written with a capital: Product Category, Sales, Birth Date. Descriptive references are lowercase: the products listed.
- Of course, generic grammar still applies.

Here are some example snippets that show these conventions:

- This expression will return the total sales for products in the Computers category, within the current context. If it's executed in the same context as our *Total Sales* expression was in our last example, then *Total Computer Sales* represents the total amount for sales transactions of products in the Computers category for the year 2023 for stores in the United States.
- To calculate this, the column Category in the table Product Category is filtered based on the value Computers in the filter context to the DAX expression.
- To calculate the running sum of *Sales*, we are using the RUNNINGSUM function of DAX instead of CALCULATE, with the *axis* parameter set to the columns axis by specifying COLUMNS. This means the visual calculation will look like `MyVC = RUNNINGSUM ( [Sales], COLUMNS )`.

Signature, syntax, expression, and statement

The distinction between signature, syntax, expression, and statement sparked a debate among the authors of this book. We will not repeat the debate here, but it's important to know that throughout the book we use signature to refer to the unique identification of a function. You can think of it as the blueprint of a function, or what you would generally put in the header (.h) file if you were programming in languages such as C or C++. A function's signature includes its name and the parameters it accepts. A function's signature helps you to understand what a function does and how it should be used. The signature of a DAX function by itself is not executable. We will use the word signature throughout this book with this definition in mind.

In contrast, syntax refers to the rules and conventions that define the structure and format of valid statements in a programming language like DAX. It's like grammar in human languages. And just as improper grammar can make a sentence hard to understand, incorrect syntax can cause errors in code or make it difficult to read. In this book, we aim to avoid the word syntax and instead refer to pieces of executable DAX code in descriptive terms, such as "this visual calculation". If we need to, we will refer to pieces of executable DAX code as expressions or statements, such as "These three statements are equivalent because...."

These four terms also interact because statements can be syntactically correct but use functions with ambiguous or poorly designed signatures.

System requirements

We strongly urge you to follow along with the examples in this book and the tutorial provided in Chapter 2. To do so, we recommend you use the most recent Power BI Desktop version, but at minimum, you should be using the February 2024 version. In Power BI Desktop, be sure to turn on the visual calculations preview feature if it isn't turned on already. To do so, choose Options and Settings > Options > Preview features. Then select Visual calculations and click OK. Finally, restart Power BI Desktop to enable visual calculations.

If you're using Power BI in the web exclusively, you can also follow along, but keep in mind that the web version of Power BI Desktop doesn't have a ribbon at the time of this writing, so you will have to use the context menu of a visual (right-click), the three-dot menu of the visual to enter visual calculations edit mode, or the button in the bar at the top of the report.

When you have a supported visual selected while editing a report, there are multiple ways of entering visual calculations edit mode:

- Clicking the New visual calculation button in the ribbon, as shown in Figure 0-1 (Power BI Desktop only).

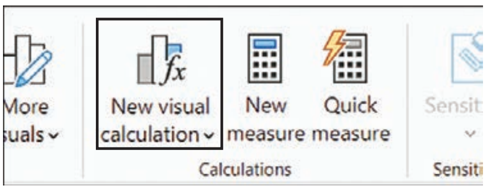


FIGURE 0-1 The New visual calculation button in the Calculations section of the Home ribbon.

- Right-clicking anywhere on the visual to open the context menu, choosing New visual calculation, and selecting the type of visual calculation you want to create (see Figure 0-2).
- Clicking the three-dot menu in the top-right corner of the visual, choosing New visual calculation, and selecting the type of visual calculation you want to create (see Figure 0-3).

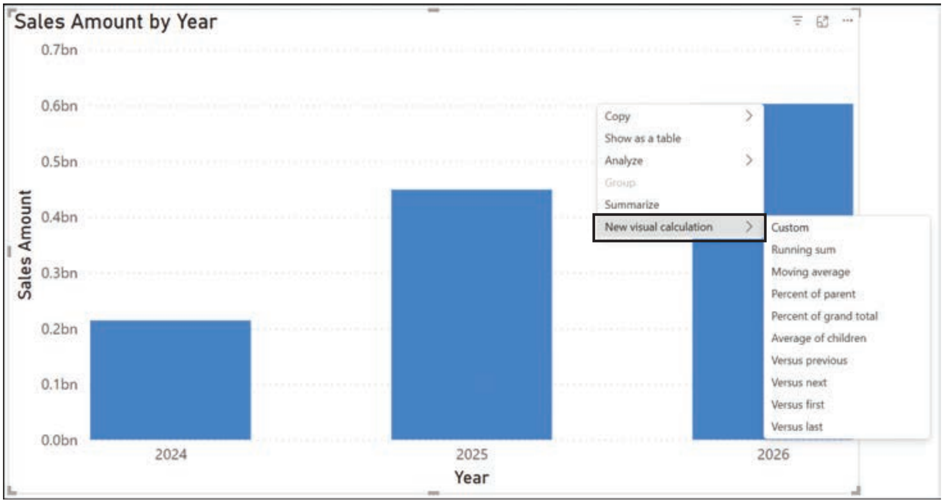


FIGURE 0-2 The context menu of visuals includes a New visual calculation entry.

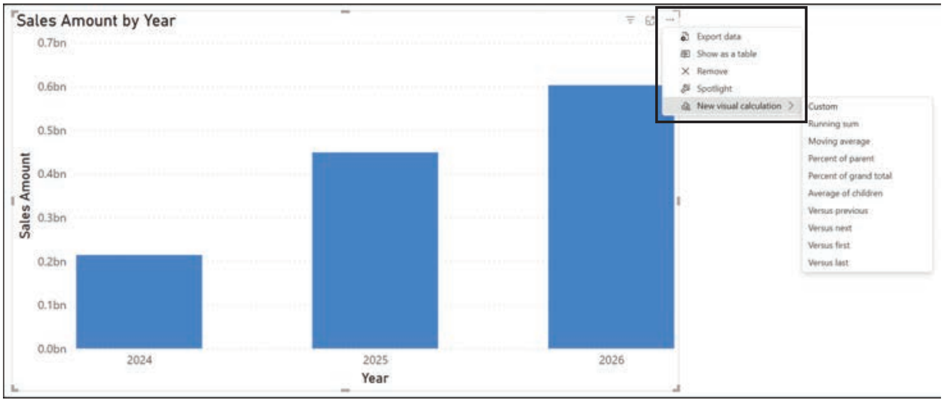


FIGURE 0-3 The three-dot menu of visuals includes a New visual calculation entry.

- Clicking the New visual calculation button in the bar at the top of the report, as shown in Figure 0-4 (browser only).

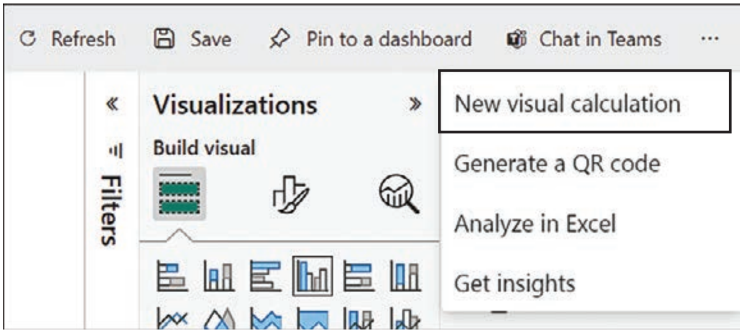


FIGURE 0-4 The bar at the top of the report while editing in the browser includes a New visual calculation button.

- Right-clicking an existing visual calculation and selecting Edit calculation (see Figure 0-5).

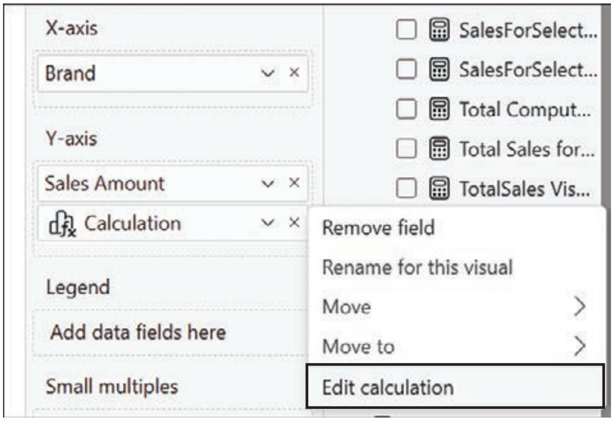


FIGURE 0-5 The context menu of a visual calculation is another way to enter visual calculations edit mode.

GitHub Repo

All examples and code snippets in the book are available on the book's website at *MicrosoftPressStore.com/powerbivis/downloads*.

The examples throughout the book use the same semantic model, available in the Contoso Sales.pbix file, which you can download from the aforementioned website.

The pbix file not only contains the semantic model but also includes multiple report pages showing the examples in the chapters in this book. The semantic model has some dimension tables like Customer, Date, Product, and Store and contains three fact tables with data about sales, orders, and order details.

The data in the model was generated using the free Contoso Data Generator published by SQLBI, which is available at *docs.sqlbi.com/contoso-data-generator/*.

Chapter 2, which walks you through creating your first visual calculation, will introduce the example model in more detail.

We also provide two more pbix files to download: Contoso Sales CH2 Start.pbix and Contoso Sales CH2 End.pbix. The Contoso Sales CH2 Start.pbix file is the starting point for following the tutorial in Chapter 2, and the Contoso Sales CH2 End.pbix is the result of the tutorial in that same chapter.

Errata, updates, and book support

We've made every effort to ensure the accuracy of this book and its companion content. You can access updates to this book—in the form of a list of submitted errata and their related corrections—at

MicrosoftPressStore.com/powerbivis/errata

If you discover an error that is not already listed, please submit it to us at the same page.

For additional book support and information, please visit *MicrosoftPressStore.com/Support*.

Please note that product support for Microsoft software and hardware is not offered through the previous addresses. For help with Microsoft software or hardware, go to *support.microsoft.com*.

CHAPTER 2

My first visual calculation

Now that you have learned about visual calculations, how easy it is to use them, why they matter, and what problem they solve, you must be eager to try them out for yourself. To that end, this chapter switches from the theoretical approach presented in the previous chapter to real-world examples in which you could use visual calculations. This chapter starts by introducing the semantic model we will be using throughout the book. After that, it guides you through the process of creating visual calculations. You will be introduced to the process of creating visual calculations. Meanwhile, we will touch upon a few new concepts, which will be further explained in the next chapters of this book. The first examples are also building blocks for practical, real-world examples, which will be described in Part IV, "The art of practical visual calculations."

Introduction to the example semantic model

To help you create your first visual calculations, we have created an example of a semantic model that you can use, which you can find on this book's companion website (see Introduction). The pbix file Contoso Sales CH2 Start provides a starting point from which you can follow along to do all the steps described in this chapter. The pbix file not only contains the semantic model but also includes a report page called Chapter 2, which serves as the starting point on which we'll build in this chapter. On the website, you can also find the Contoso Sales CH2 End pbix file, which will match the final version that you will have created if you follow along with all the steps in this chapter. You can check that model to determine whether you did it correctly.

> **Note** It's also possible to create visual calculations directly in the service, but for the purpose of these first examples, we'll use the desktop version. In the service, the steps are similar.

To begin, open the example semantic model in Power BI Desktop. As shown in Figure 2-1, the example model contains data about sales transactions. It's a simple model that has some dimension tables like Customer, Date, Product, and Store, and contains three fact tables with data about Sales, Orders, and Order Detail. To enable you to start straight away with the visual calculations, we've already prepared the necessary relationships and some simple measures, which can be found in a table where all measures are stored. This table is named _Measures.

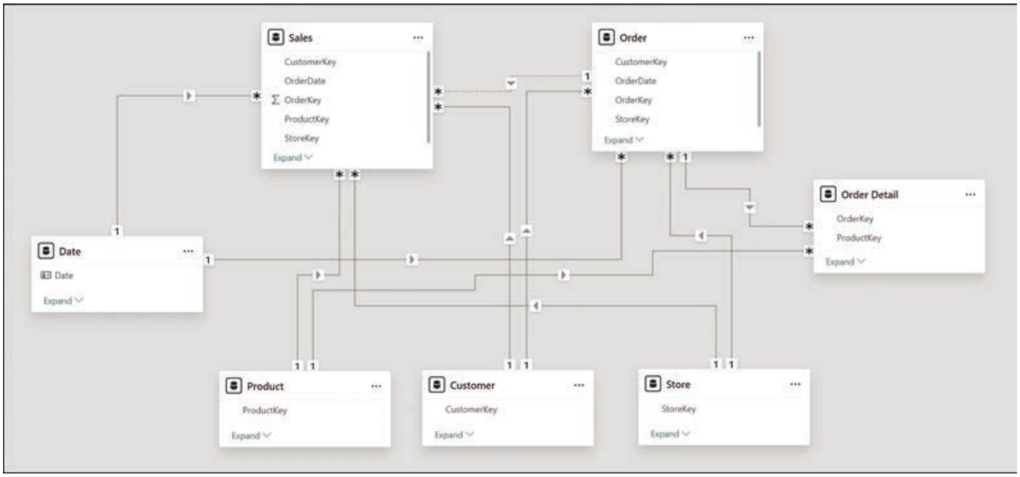


FIGURE 2-1 The model view of the example semantic model.

On the Chapter 2 report page, the clustered bar chart on the left shows the *Costs* and *Sales* per product category, and the clustered column chart on the right gives the *Sales* per Month. There is also a slicer to enable you to select different years.

Questions that could arise from this example model include the following:

- What is the profit per product category?
- What are the total year-to-date sales?
- How is the total profit divided over the different countries?

Of course, it's possible to create measures in Power BI that answer these questions and show them on the report page. However, we'll employ visual calculations instead because they're so easy to use.

Create your first visual calculation

A visual calculation is created inside a visual. For your first visual calculation, you'll enhance the sample visual that contains *Sales* and *Costs* per product category with the total profit per product category. Do so by first selecting the clustered bar chart on the left of the report page. When the visual is selected, there are multiple ways to add a visual calculation to this bar chart:

- Click the New visual calculation button in the Calculations section of the Home ribbon, as shown in Figure 2-2.

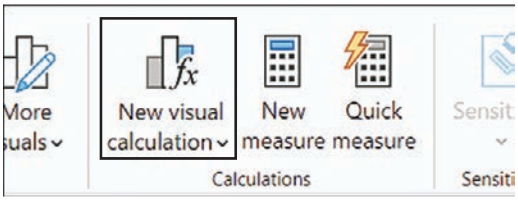


FIGURE 2-2 The New visual calculation button in the Calculations section of the Home ribbon.

- Click the More options button (it features three dots) in the top-right corner of the visual, and select the New visual calculation > Custom option, as shown in Figure 2-3.

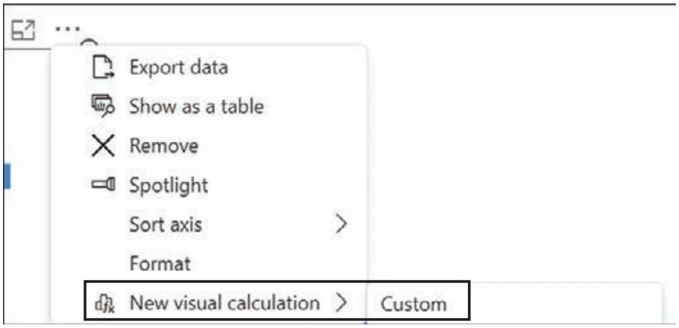


FIGURE 2-3 Click the New visual calculation option, then the Custom option in the More options button.

- Right-click the visual and select the New visual calculation option in the context menu that opens, as shown in Figure 2-4.

FIGURE 2-4 Right-clicking the visual opens a context menu with a New visual calculation > Custom option.

Performing one of these actions displays the visual calculations edit mode window, which shows your visual as it appeared on the report page at the top and a matrix with data beneath it. The matrix is a representation of the data in the visual and is conveniently called the visual matrix. In this case, the visual matrix contains values for the sales and costs per product category as well as the total sales and costs for all product categories combined. An empty Calculation column appears to the right of these values, as is shown in Figure 2-5. This is where you will create your first visual calculation.

The first question we want to answer is, what is the total profit per product category? The formula for this involves subtracting the costs from the sales. This is exactly what you'll do to create your first visual calculation.

As explained in Chapter 1, "Introduction to visual calculations," visual calculations are written in the DAX language. They have a similar syntax as regular measures and should thus begin with a name. We'll call our first visual calculation *Profit*.

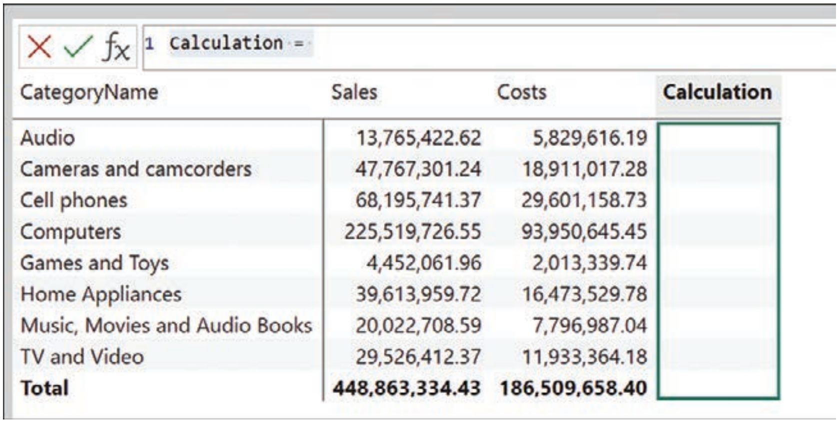


FIGURE 2-5 Add a visual calculation to an existing visual in the visual matrix.

To create this visual calculation, place your cursor in the formula bar and use ordinary DAX to write the formula. Notice that the IntelliSense feature provides suggestions as you type. In this example, after you type the name for your visual calculation and begin typing the formula, as you press the S key, you will see a list of suggestions that includes [Sales], as shown in Figure 2-6.

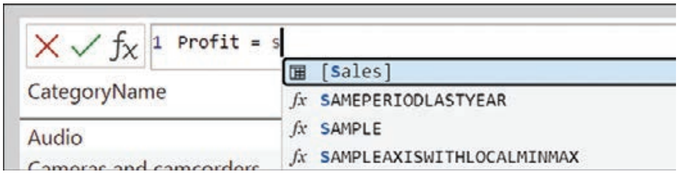


FIGURE 2-6 Defining *Profit* as a visual calculation.

> **Note** The suggestion, [Sales], has square brackets, like a column or measure. The icon that precedes this indicates a table column. However, we have used a measure in the visual. Furthermore, it does not have a table name in front of the square brackets, as would be typical when you are referring to a table column. This is because this [Sales] object actually refers to the column of Sales that is present in the visual matrix, which is visible in the visual calculations edit mode window. This concept will be further explained in the next section.

When you type a square bracket in the formula bar, you will see all the available columns for use in this visual matrix. In this example, there are three: CategoryName, Sales, and Costs. CategoryName is a column of the Product table. The Sales and Costs columns are the results of the DAX measures *Sales* and *Costs* that are defined in the _Measures table. These measures were added to the visual and return values for the context that is evaluated for the visual in which we are creating a visual calculation.

Back to our example. We wanted to create output for the profit, which is defined as the difference between sales and costs. In the formula bar, you can simply type the following DAX formula:

```
Profit = [Sales] - [Costs]
```

Press Enter or click the green check mark to finish. This will create a new column in the visual matrix with the name Profit, and in each row, the corresponding value of sales minus costs is evaluated, as shown in Figure 2-7. Congratulations, you have successfully created your first visual calculation!

CategoryName	Sales	Costs	Profit
Audio	13,765,422.62	5,829,616.19	7,935,806.43
Cameras and camcorders	47,767,301.24	18,911,017.28	28,856,283.96
Cell phones	68,195,741.37	29,601,158.73	38,594,582.64
Computers	225,519,726.55	93,950,645.45	131,569,081.09
Games and Toys	4,452,061.96	2,013,339.74	2,438,722.22
Home Appliances	39,613,959.72	16,473,529.78	23,140,429.95
Music, Movies and Audio Books	20,022,708.59	7,796,987.04	12,225,721.55
TV and Video	29,526,412.37	11,933,364.18	17,593,048.19
Total	**448,863,334.43**	**186,509,658.40**	**262,353,676.03**

FIGURE 2-7 *Profit* as a visual calculation in the visual matrix.

This new calculation is also added to your visual. As shown in Figure 2-8, the visual should now contain three values: *Sales* and *Costs* (which are measures) and *Profit* (which is your very first visual calculation).

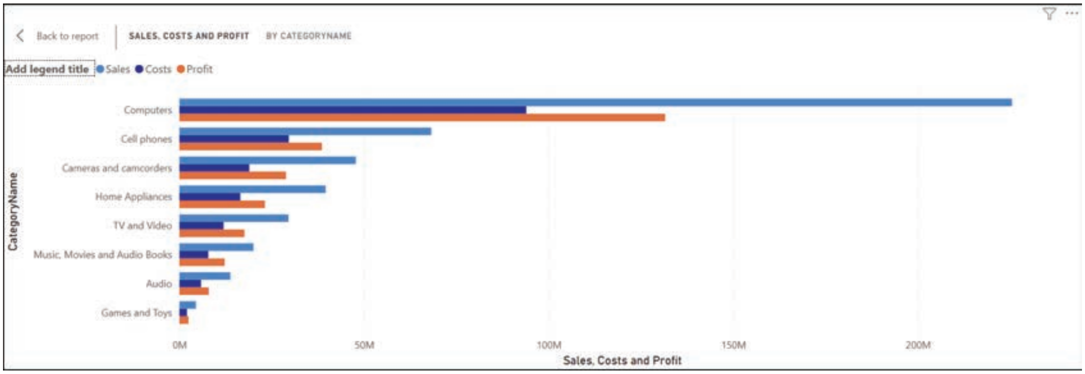


FIGURE 2-8 *Profit* as a visual calculation in the visual preview.

You can add a new visual calculation by writing another DAX statement in the formula bar. You can even use your newly created visual calculation in another visual calculation. For example, you can create a visual calculation to calculate the profit including a 10 percent tax amount by typing the following DAX formula in the formula bar and pressing Enter:

```
Profit incl tax = 1.1 * [Profit]
```

This will create a second visual calculation, and thus a new column in your visual matrix and a new bar in the visual preview.

It's also possible to go back to your report by clicking the Back to report button in the top-left corner of the window, as shown in Figure 2-9.

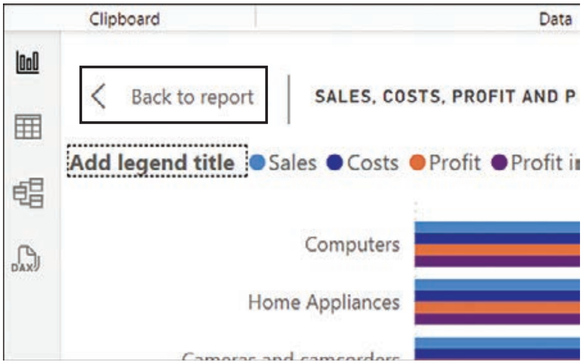


FIGURE 2-9 You can use the Back to report button to go back to the report.

In the Visualizations pane of the visual that you just edited, there are two new values on the X-axis, called *Profit* and *Profit incl tax* (assuming you created both visual calculations described so far), as shown in Figure 2-10. To help you differentiate between measures, columns and visual calculations, the visual calculations are identified by a small icon that shows a column chart with fx in front of it.

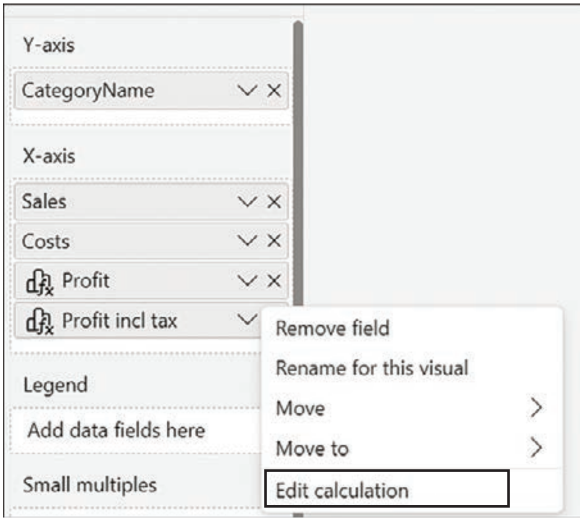


FIGURE 2-10 The X-axis of the visual contains four fields.

There are also different options in the dropdown on the right side of the data fields. For regular measures and columns, there are options for Show value as and New quick measure. But for visual calculations, there is an Edit calculation option instead. Clicking this option returns you to the visual calculations edit mode window, with the visual preview on top and the visual matrix, where you can edit your visual calculation, below it.

Create a year-to-date sales calculation

The report page contains a visual that shows the sales per month, but it's very common to look at sales on a year-to-date basis, instead—hence, our second question earlier in this chapter: what are the total year-to-date sales? Let's see if we can create another visual calculation for year-to-date sales. In this example, you will add a new year-to-date sales calculation to the clustered column chart that contains the sales per month, shown in Figure 2-11.

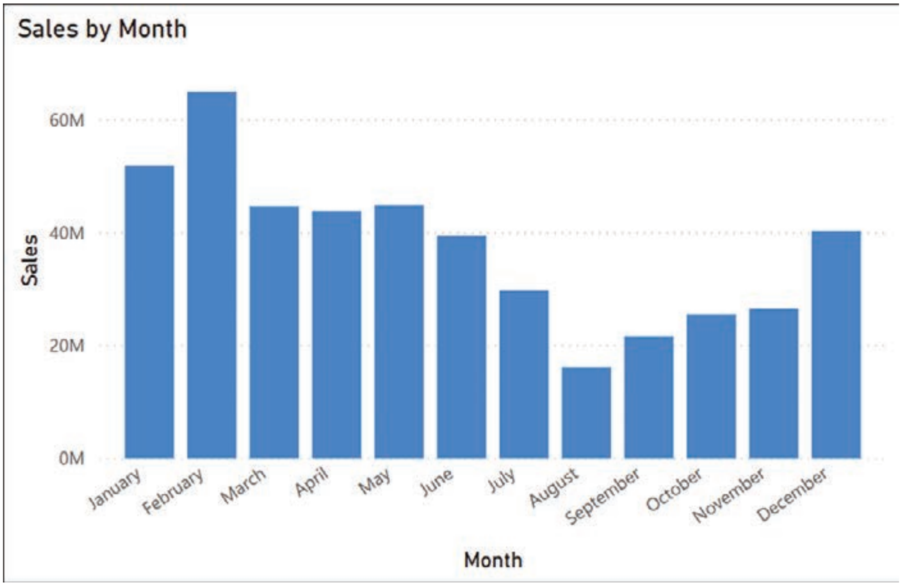


FIGURE 2-11 This clustered column chart visual shows the *Sales* per Month.

As before, begin by selecting the visual you want to work on—in this case, the column chart. Then add a visual calculation to open the visual calculations edit mode window, with the column chart on top and the visual matrix below it. This time, the visual matrix has one column for the month, which lists all the month names, and one column called Sales, where the values for the data in the column chart are listed. At the bottom, there's a total row that shows the total sales over all the months.

To calculate year-to-date sales, you add up the sales for all previous months, plus the sales for the current month. So, for January, the year-to-date sales would be the same as the regular sales because it's the first month of the year. But for February, the year-to-date sales would be the sum of the January sales and the February sales. And for March, it would be the sum of January, February, and March.

> **Note** If you're familiar with DAX, you could use time intelligence functions to obtain the year-to-date value. However, because visual calculations are meant to be simple, and are available to any user who is not ready to use time intelligence functions, we will use visual calculations here.

Because users frequently need to obtain running sums, like year-to-date sales, visual calculations offer an expression template for this operation. An expression template is a predefined template to compute certain calculations in visual calculations. To view and insert available expression templates (of which there are several), click the fx button for the visual calculation to the left of the formula bar (see Figure 2-12). These expression templates can be selected in the visual calculations edit mode, as well as in the ribbon (refer to Figure 2-2), the More options button (refer to Figure 2-3), and the context menu (refer to Figure 2-4).

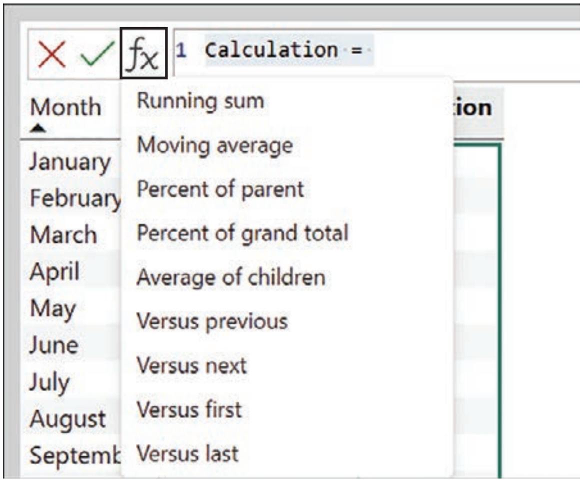


FIGURE 2-12 The fx button allows you to view and insert expression templates.

For this example, select the first expression template in the list, Running sum. As shown in Figure 2-13, expression templates provide easy-to-use parameter pickers that enable you to quickly select parameter values without having to type them. In addition, this expression template automatically names your visual calculation *Running sum* and adds the RUNNINGSUM DAX function. For this example, let's change the name of the visual calculation to *YTD Sales* instead of *Running sum* because that is a better description of what you want to calculate.

FIGURE 2-13 Using the Running Sum expression template is made easy thanks to the parameter picker.

The RUNNINGSUM function

The RUNNINGSUM DAX function is not available for measures, calculated columns, and calculated tables. Rather, it's a brand-new function created specifically for visual calculations. It calculates the sum of values by adding the current value to the preceding values—exactly what you need to calculate the year-to-date sales. However, the RUNNINGSUM function has much broader uses, and can be applied to all values on the axis. For example, suppose you have a process, and each step in that process takes a certain number of seconds to complete, and you've created a visual with all the steps on the axis. Using the RUNNINGSUM function, you can calculate exactly how much time has passed after each step. And if you add the starting timestamp for the first step, you even get the exact timestamp of when each process step completes. That's with just one DAX function! Truly, the possibilities are endless. We will discuss axis and related concepts in Chapter 3 and dive much deeper into the available functions, including RUNNINGSUM, in Part II, "Visual calculations functions", of this book.

As shown in Figure 2-13, the DAX function RUNNINGSUM asks for at least one parameter, as indicated by the `[Field]` parameter picker. However, our data does not contain a Field column, as indicated by the red squiggle. You need to replace this field with the column for which you want to calculate the running sum. To do that, either click the parameter picker and pick [Sales] as shown in Figure 2-14 or delete the `[Field]` placeholder in the formula bar and type a square opening bracket to see a list of all the available options for the current visual (in this case, Month and Sales).

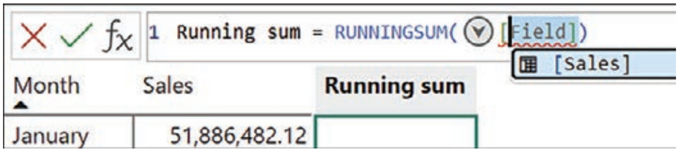


FIGURE 2-14 Use the parameter picker to select a column.

To calculate the running sum of the sales data, select `[Sales]` to generate the following DAX statement:

```
YTD Sales = RUNNINGSUM ( [Sales] )
```

Then press Enter or click the green check mark to the left of the formula bar to create the new visual calculation in your visual. The resulting visual calculation should appear in the visual preview in the top half of the screen (see Figure 2-15).

The values are exactly as you would expect, which is best seen in the visual matrix at the bottom of the screen (see Figure 2-16).

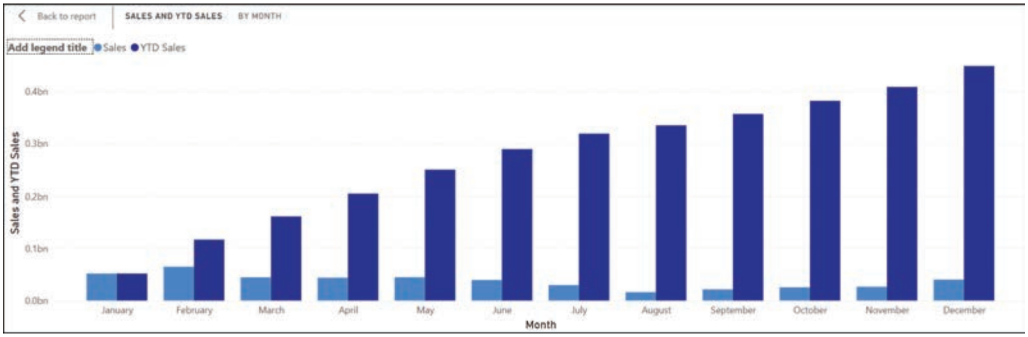
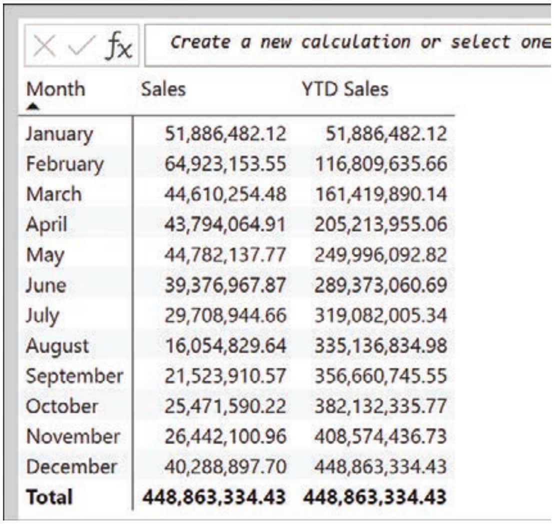


FIGURE 2-15 The *YTD Sales* visual calculation correctly calculates the running sum of *Sales*.



Month	Sales	YTD Sales
January	51,886,482.12	51,886,482.12
February	64,923,153.55	116,809,635.66
March	44,610,254.48	161,419,890.14
April	43,794,064.91	205,213,955.06
May	44,782,137.77	249,996,092.82
June	39,376,967.87	289,373,060.69
July	29,708,944.66	319,082,005.34
August	16,054,829.64	335,136,834.98
September	21,523,910.57	356,660,745.55
October	25,471,590.22	382,132,335.77
November	26,442,100.96	408,574,436.73
December	40,288,897.70	448,863,334.43
Total	**448,863,334.43**	**448,863,334.43**

FIGURE 2-16 The visual matrix of the visual shows *Sales* per Month and *YTD Sales*.

As expected, the two values for January are the same, because the YTD sales in January consist only of sales during that single month. The other values are also as expected, with YTD sales in February equaling the sum of the January and February sales, and YTD sales in March equaling the sum of the YTD sales in February and the March sales. And, as expected, at the end of the year, the YTD sales in December equal the grand total of all the individual sales per month, resulting in a total of approximately 449 million—which is also the total for the YTD Sales column.

> **Note** We are well aware that the TOTALYTD function restarts the sum at Year change, when the semantic model has a dimension marked as Date table. For this example, with visual calculations, we assume you only work with one year of data, and therefore, RUNNINGSUM will suffice as well. At year change, the RUNNINGSUM function will continue summing up. This behavior can be influenced by using the *reset* parameter as described in Chapter 3, "Visual calculations concepts." We will also add this *reset* parameter later in this chapter to a visual calculation.

Suppose you want to view the YTD sales per month, but not the sales for each individual month. You cannot simply remove the *Sales* measure from the visual because you need that data to calculate the YTD sales. You can, however, hide that data. To do so, make sure you're in the visual calculations edit mode, locate the *Sales* measure in the Visualizations pane, and click the eye icon. (This is the same icon you use to hide or show columns or tables in the table view of a Power BI semantic model.) A diagonal line will appear through the icon, indicating that it's now hidden (see Figure 2-17).

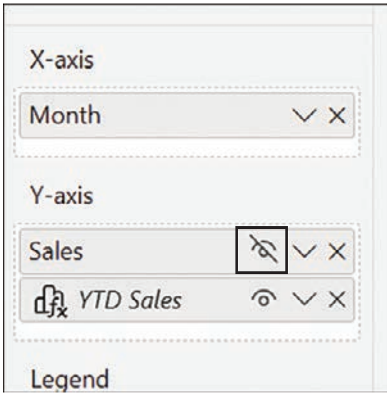


FIGURE 2-17 The eye icon next to measures and visual calculations makes it possible to hide measures or visual calculations from the visual.

Notice that the visual matrix at the bottom of the visual calculations edit mode window is not affected by hiding the *Sales* measure. It still contains two columns, Sales and YTD Sales, and rows for each Month. However, the visual preview in the top part of the screen contains only the *YTD Sales* for each Month, as does the visual on the report page.

Notice, too, that the Y-axis setting in the Visualizations pane for the visual on the report page lacks an entry for the *Sales* measure. However, if you hover your mouse pointer over the information icon above the Y-axis setting (see Figure 2-18), Power BI informs you that there are more fields in the visual, but they are hidden. You can still access and use them when you open the visual calculations edit mode window.

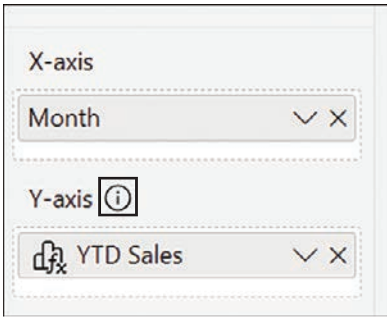


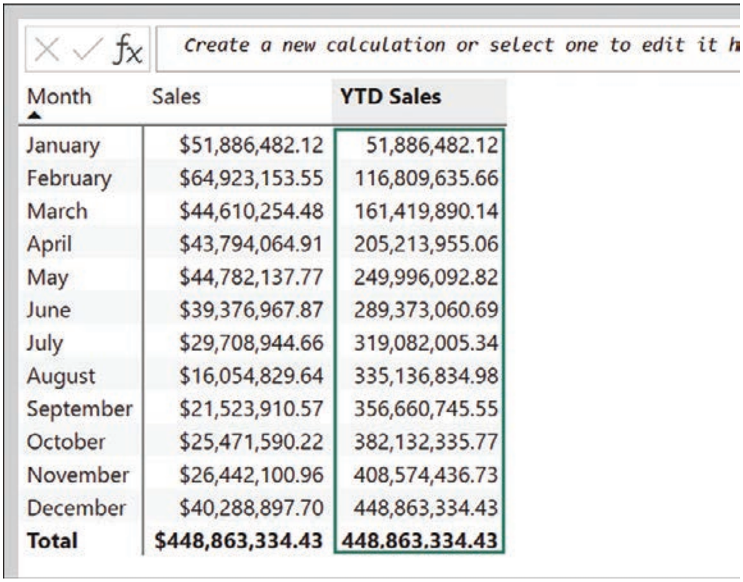
FIGURE 2-18 The information icon indicates that this visual has hidden fields.

> **Note** You can create very complex visual calculations in steps and hide any irrelevant intermediate results. We'll look at an example of a more complex calculation that requires some intermediate steps later in this chapter. Chapter 3 includes more information about the concept of hiding elements in visuals.

Set a format

When you create a measure, you often need to format it, so it looks how you want it to in your report. But how does this work for visual calculations? To find out, let's use the example you just created for year-to-date sales.

The *YTD Sales* visual calculation doesn't have any format set; neither does the *Sales* measure from which it is calculated. So, let's start with that. If you change the *Sales* measure to a currency format with a dollar sign in front of it, you won't see any changes in the visual. This makes sense because even though the *Sales* measure is in that visual, it's hidden from view. If you switch to the visual calculations edit mode, though, you do see the change in the visual matrix, with all the values in the Sales column preceded by a dollar sign (see Figure 2-19). However, the YTD Sales column has not changed. This is because visual calculations do not take on the format of any measures used to create them—although it does try to guess, based on the function you used and your input data types. In this case, the result is not what we want. We want to format it with a dollar sign in front, just like the *Sales* measure.



Month	Sales	YTD Sales
January	$51,886,482.12	51,886,482.12
February	$64,923,153.55	116,809,635.66
March	$44,610,254.48	161,419,890.14
April	$43,794,064.91	205,213,955.06
May	$44,782,137.77	249,996,092.82
June	$39,376,967.87	289,373,060.69
July	$29,708,944.66	319,082,005.34
August	$16,054,829.64	335,136,834.98
September	$21,523,910.57	356,660,745.55
October	$25,471,590.22	382,132,335.77
November	$26,442,100.96	408,574,436.73
December	$40,288,897.70	448,863,334.43
Total	**$448,863,334.43**	**448,863,334.43**

FIGURE 2-19 The visual matrix shows the Sales with a currency format, but the YTD Sales are not formatted in the same way.

Fortunately, it's possible to set a format for a visual calculation. You do this in the visual's Visualizations pane. Simply click the General tab, open Data format, and choose YTD Sales from the Apply settings to menu. Then open the Format menu under Format options and choose Currency. Make sure that the Currency format menu is set to $ English (United States) as shown in Figure 2-20.

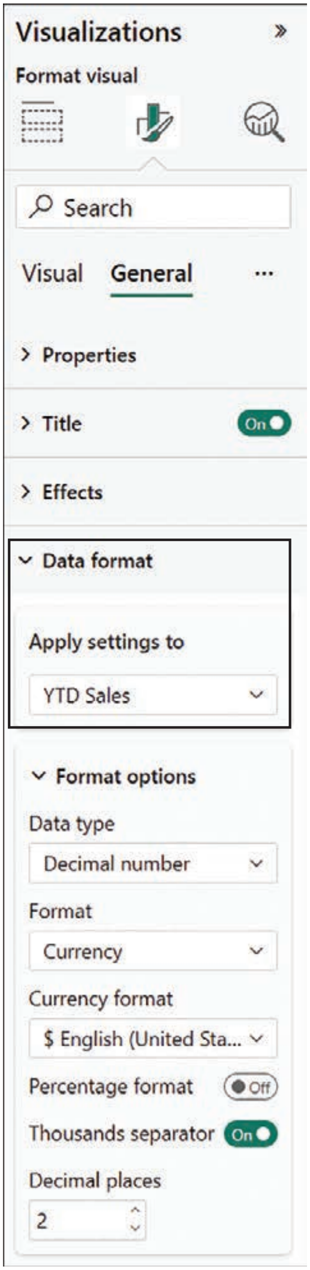


FIGURE 2-20 You can set data formats for visual calculations in the General section of the Visualizations pane, under the Data Format header.

> **Note** You can create custom formats for visual calculations. This gives you the freedom to display your visual calculations however you like. Advanced formatting options for visual calculations are described in Chapter 4, "Organization and use of visual calculations."

This change is also visible in the visual matrix, with entries in the YTD Sales column formatted as a currency with the dollar sign in front. Furthermore, because the visual matrix is just a matrix representation of the data in the visual, the change is also visible in the visual preview, with the axis now containing a dollar sign. The values in the tooltip are formatted as currencies as well.

Calculate the profit per country

The third question posed at the beginning of this chapter was, how is the total profit divided over the different countries?

Earlier you created a visual calculation to calculate the profit, but that visual calculation is not available in the Data pane of our model for use in a new visual. However, it's available in a visual that you have already created—the one that shows *Sales*, *Costs*, *Profit*, and *Profit incl tax* by CategoryName. So, you can copy that visual to a new report page and change it as needed to answer our question.

To start, copy the aforementioned visual and paste it on a new report page. Then remove the *Profit incl tax* entry from the Visualizations pane because you don't need that data. Finally, change the Y-axis data field from CategoryName to the CountryName from the Store table. The result will be a clustered bar chart that shows *Sales*, *Costs*, and *Profit* by CountryName, as shown in Figure 2-21.

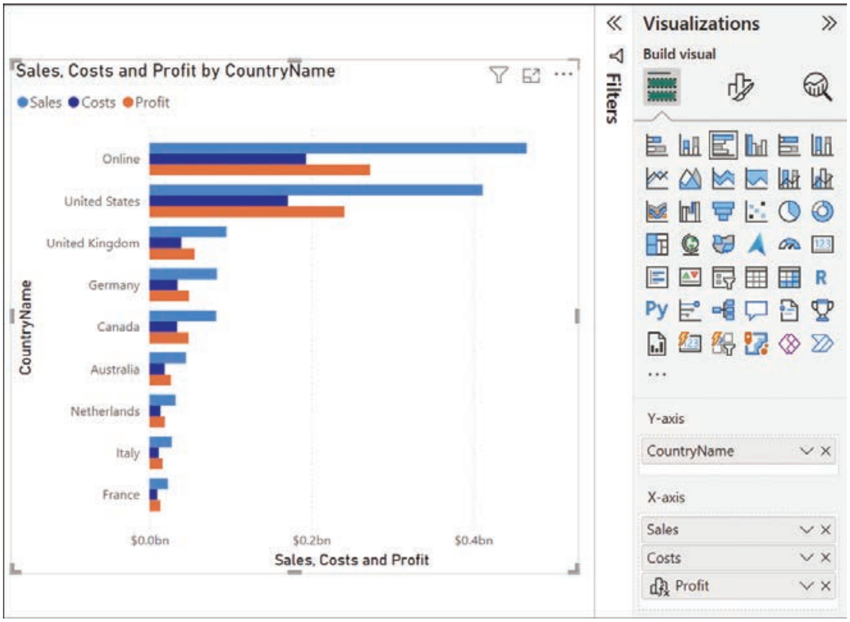


FIGURE 2-21 A clustered bar chart visual showing *Sales*, *Costs*, and *Profit* by CountryName, next to the Visualizations pane for the visual.

This visual currently contains *Sales* and *Costs* measures, which you don't need to show there. However, you do need these measures to calculate the profit. So, rather than removing the *Sales* and *Costs* measures from the visual, you need to hide them. To do so, open the visual calculations edit mode window and click the eye icon next to the Sales and Costs entries (see Figure 2-22).

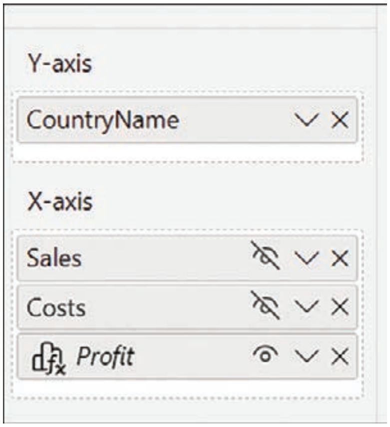


FIGURE 2-22 The Visualizations pane of the visual where the *Sales* and *Costs* measures are hidden, and only the *Profit* visual calculation is visible.

You now have a visual that shows the profit for each country. But the visual does not show the percentage of the total profit for each country, which is what you're really after. One way to obtain this is to simply change the visual from a clustered bar chart to a pie chart. This will display the profit per country as part of the total profit, and it displays the percentages as well. However, because there is a known aversion to pie charts among the Power BI community (including some of the authors of this book), we will take a different approach: creating a visual calculation that shows these percentages in a bar chart.

To calculate the profit of each country as a percentage of the total profit, you need to divide the profit for each country by the total profit (which is already visible in the Total row of the Profit column). To achieve this, you can create a column that displays the total profit for all countries on each country row, and then divide each country's profit by this total profit to obtain a percentage.

If you were to do this using DAX measures, you would need to change the context to remove the filter that is set to a specific country for each row. So, you could use CALCULATE and a function like REMOVEFILTERS or ALL to obtain the total profit on the rows for each country. But in the spirit of making DAX easier, we'll use a pair of functions created specifically for visual calculations: COLLAPSE and COLLAPSEALL. These two functions make it possible to pick a field in the visual matrix and collapse the lattice, returning the collapsed value of the column. We will discuss how this works in Chapter 3 and discuss the functions in detail in Part II.

> **Note** In this case, you only have one field on the rows (CountryName), so you can only collapse to one higher level—meaning COLLAPSE and COLLAPSEALL will generate the same result. However, if you have multiple levels on your axis—for example, the year, quarter, and month—you can use COLLAPSEALL to display the total value for all rows (including the subtotals).

> **Note** You can use the COLLAPSE function to return different kinds of results, such as the total value for all rows (same as COLLAPSEALL), the quarter total for each month, the yearly total for each quarter (so, one level collapsed), and more. We'll dive deeper into this in Part II.

To start, create a visual calculation with the following DAX statement:

```
Total Profit = COLLAPSEALL ( [Profit], ROWS )
```

This creates another column in the visual matrix with the total profit displayed on each individual row, as well as the Total row (see Figure 2-23).

Create a new calculation or select one to edit it here

CountryName	Sales	Costs	Profit	Total Profit
Australia	$45,089,152.58	18,690,555.54	26,398,597.04	740,693,124.92
Canada	$82,316,364.56	34,119,408.06	48,196,956.50	740,693,124.92
France	$22,745,351.50	9,471,036.05	13,274,315.44	740,693,124.92
Germany	$83,272,724.41	34,548,906.68	48,723,817.72	740,693,124.92
Italy	$27,467,104.93	11,357,110.69	16,109,994.25	740,693,124.92
Netherlands	$32,290,656.77	13,396,522.28	18,894,134.48	740,693,124.92
Online	$465,981,615.52	193,372,577.81	272,609,037.70	740,693,124.92
United Kingdom	$94,934,581.28	39,254,650.66	55,679,930.62	740,693,124.92
United States	$411,730,738.85	170,924,397.70	240,806,341.15	740,693,124.92
Total	**$1,265,828,290.39**	**525,135,165.47**	**740,693,124.92**	**740,693,124.92**

FIGURE 2-23 The *Total Profit* visual calculation displays the total profit for each individual row in the visual matrix.

> **Note** The second parameter, ROWS, indicates to the function that you want to collapse the rows. This implies that you can also collapse over different axes, which is something that we will explore later in this chapter.

The next step is to create yet another visual calculation to calculate the percentage of the profit per country as a part of the total profit. You can use the following DAX to obtain a new column that contains these values:

```
% Profit = DIVIDE ( [Profit], [Total Profit] )
```

Then use the General Format settings in the Visualizations pane to select the Percentage format for the visual calculation (see Figure 2-24). Figure 2-25 shows the resulting visual matrix.

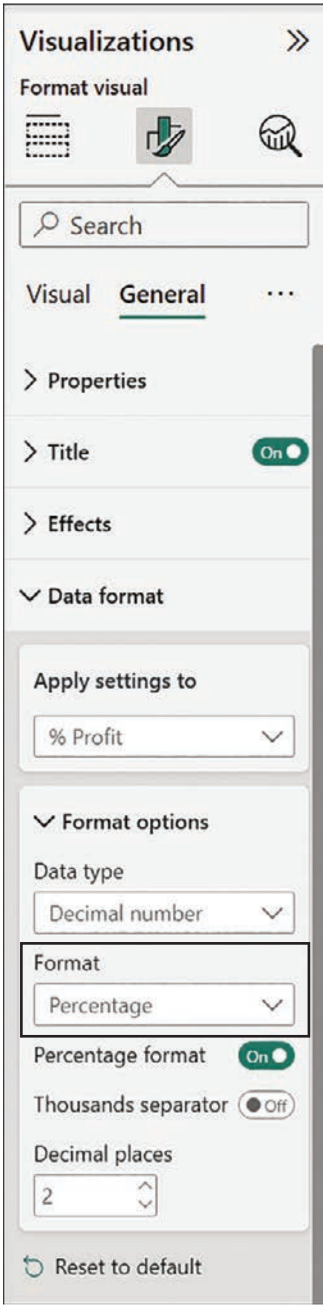


FIGURE 2-24 Use the General Format settings in the Visualizations pane to set the data format of the *% Profit* visual calculation to a Percentage.

*

Create a new calculation or select one to edit it here

CountryName	Sales	Costs	Profit	Total Profit	% Profit
Australia	$45,089,152.58	18,690,555.54	26,398,597.04	740,693,124.92	3.56%
Canada	$82,316,364.56	34,119,408.06	48,196,956.50	740,693,124.92	6.51%
France	$22,745,351.50	9,471,036.05	13,274,315.44	740,693,124.92	1.79%
Germany	$83,272,724.41	34,548,906.68	48,723,817.72	740,693,124.92	6.58%
Italy	$27,467,104.93	11,357,110.69	16,109,994.25	740,693,124.92	2.17%
Netherlands	$32,290,656.77	13,396,522.28	18,894,134.48	740,693,124.92	2.55%
Online	$465,981,615.52	193,372,577.81	272,609,037.70	740,693,124.92	36.80%
United Kingdom	$94,934,581.28	39,254,650.66	55,679,930.62	740,693,124.92	7.52%
United States	$411,730,738.85	170,924,397.70	240,806,341.15	740,693,124.92	32.51%
Total	**$1,265,828,290.39**	**525,135,165.47**	**740,693,124.92**	**740,693,124.92**	**100.00%**

FIGURE 2-25 The visual matrix showing *Sales*, *Costs*, *Profit*, *Total Profit*, and *% Profit* by country.

You can now hide both the *Profit* and the *Total Profit* visual calculations to obtain a bar chart with just the percentages of the profit per country, as was our goal (see Figure 2-26).

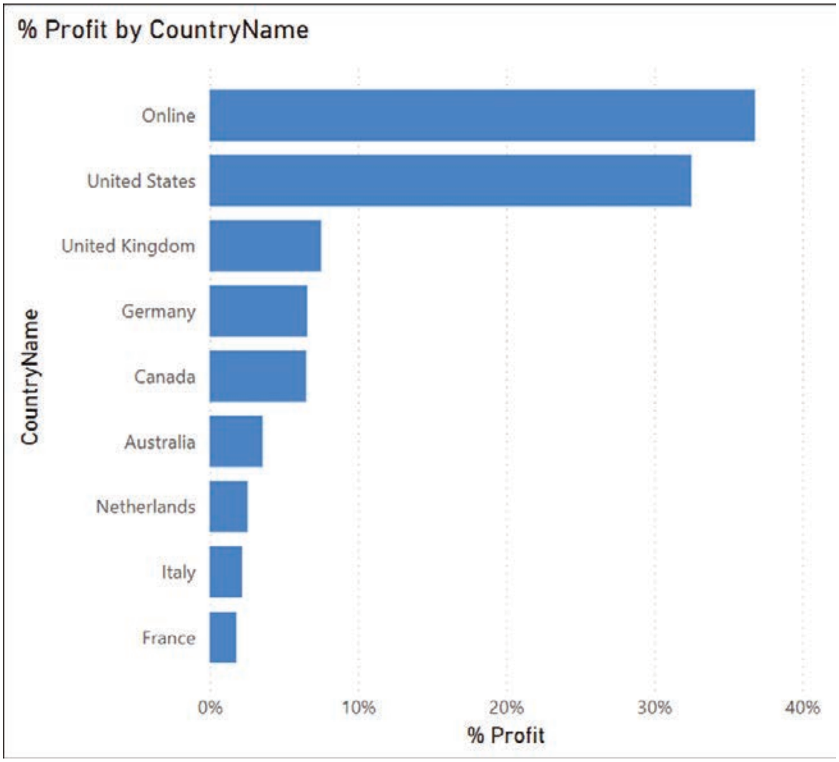


FIGURE 2-26 The resulting bar chart shows the percentage of the total profit per country.

Now, this approach works just fine. But you can obtain the percentage of the total profit per country more easily by using an expression template—specifically the Percent of grand total expression template. This expression template creates a visual calculation that does exactly what we did here, but all in one step. It uses the following syntax:

```
Percent of grand total = DIVIDE ( [Field], COLLAPSEALL ( [Field], Axis ) )
```

Simply replace the `[Field]` parameter picker with `[Profit]` (because you want to divide the profit by the total profit) and the Axis parameter picker with the keyword ROWS (to indicate that you want to collapse the rows to obtain the total profit) and press Enter. Change the format to a percentage, and you'll get the result as shown in Figure 2-27. To indicate that this calculation uses the Profit column, the header of the Profit column is lined with a dotted line. This uses the highlight functionality of visual calculations, as described in more detail in Chapter 4. Notice that the values of the *Percent of grand total* visual calculation are identical to those of the *% Profit* visual calculation.

CountryName	Sales	Costs	Profit	Total Profit	% Profit	Percent of grand total
Australia	$45,089,152.58	18,690,555.54	26,398,597.04	740,693,124.92	3.56%	3.56%
Canada	$82,316,364.56	34,119,408.06	48,196,956.50	740,693,124.92	6.51%	6.51%
France	$22,745,351.50	9,471,036.05	13,274,315.44	740,693,124.92	1.79%	1.79%
Germany	$83,272,724.41	34,548,906.68	48,723,817.72	740,693,124.92	6.58%	6.58%
Italy	$27,467,104.93	11,357,110.69	16,109,994.25	740,693,124.92	2.17%	2.17%
Netherlands	$32,290,656.77	13,396,522.28	18,894,134.48	740,693,124.92	2.55%	2.55%
Online	$465,981,615.52	193,372,577.81	272,609,037.70	740,693,124.92	36.80%	36.80%
United Kingdom	$94,934,581.28	39,254,650.66	55,679,930.62	740,693,124.92	7.52%	7.52%
United States	$411,730,738.85	170,924,397.70	240,806,341.15	740,693,124.92	32.51%	32.51%
Total	**$1,265,828,290.39**	**525,135,165.47**	**740,693,124.92**	**740,693,124.92**	**100.00%**	**100.00%**

FIGURE 2-27 The two visual calculations—*% Profit* and *Percent of grand total*—return the same results.

Reset your calculation

Now that you have seen some simple examples, it's time to explore additional functionality within visual calculations. If you recall the calculation you created for year-to-date sales per month, you might remember that there's a slicer on the report page to select only one year of data. This is on purpose; a year-to-date calculation that sums up values over different years might not give as much insight to a business user. So, you'll probably want to start a new calculation each year. In other words, you want to reset the calculation for each year. Fortunately, resetting a calculation is quite simple and involves a function you've already used: RUNNINGSUM.

When employing RUNNINGSUM to calculate the year-to-date sales, you used the following visual calculation:

```
YTD Sales = RUNNINGSUM ( [Sales] )
```

This statement has only one parameter, which is the field you want to sum over. However, the function RUNNINGSUM supports the use of additional parameters to enable you to create more complex calculations—specifically the *axis*, *orderby*, *blanks*, and *reset* parameters—as shown in the following function signature:

```
RUNNINGSUM ( Expression, [Axis], [OrderBy], [Blanks], [Reset] )
```

Only one of these parameters—the first one—is mandatory. The other four are optional. In this example, you want to apply the fifth parameter (*reset*) to reset your calculation. When you apply this parameter, the function resets the running sum whenever the parameter's value changes.

In this example, you want to reset the running sum each year. The effect of the *reset* parameter is best seen if you create a new matrix in the report with the Year and Month columns from the Date table on the rows and the *Sales* measure as the expression, or value. The expression for this is as follows:

```
YTD Sales = RUNNINGSUM ( [Sales], [Year] )
```

This expression is very similar to the one you used earlier, but in this case, you have added a *reset* parameter ([Year]) to indicate that you want to reset the calculation when the Year value changes on January 1 of the next year.

> **Note** In all the previous examples, we have omitted the *reset* parameter—meaning it's set to NONE by default. So, the calculation is not reset in any of these examples.

Figure 2-28 shows the resulting visual matrix. Notice that the values for the first year are the same as in the example where we did not use the *reset* parameter. That is, for each month, the previous months are summed up to calculate the year-to-date sales, and the totals for 2024 are the same for both the sales and the year-to-date sales. The values are identical again for January 2025 because the calculation for the year-to-date sales has been reset for the new year. Finally, the value of the year-to-date sales for February 2025 is the sum of January 2025 and February 2025, and so on. Now the calculation does exactly what you want it to do, even if you show multiple years in your visual.

Create a new calculation or select one to edit it here

Year	Month	Sales	YTD Sales
2024	January	$14,091,130.12	14,091,130.12
	February	$18,652,745.19	32,743,875.31
	March	$13,707,027.63	46,450,902.94
	April	$13,617,953.40	60,068,856.34
	May	$16,025,489.67	76,094,346.00
	June	$15,987,048.83	92,081,394.84
	July	$15,846,857.82	107,928,252.65
	August	$9,561,371.47	117,489,624.12
	September	$16,331,213.97	133,820,838.08
	October	$20,374,371.26	154,195,209.35
	November	$23,371,076.56	177,566,285.91
	December	$36,497,380.10	214,063,666.00
	Total	**$214,063,666.00**	**214,063,666.00**
2025	January	$51,886,482.12	51,886,482.12
	February	$64,923,153.55	116,809,635.66

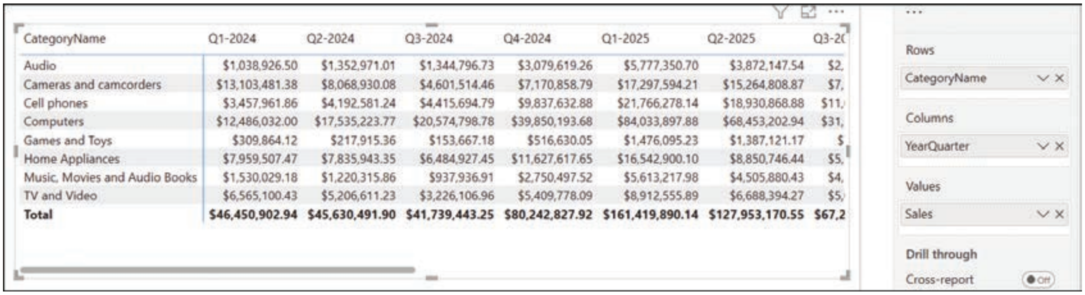
FIGURE 2-28 The visual matrix with a visual calculation for year-to-date sales that resets each year.

> **Note** Chapter 3 dives more deeply into the possibilities associated with resetting your calculations.

Create calculations that navigate over the columns

So far, all of our examples have involved calculating over rows within columns in the visual matrix—summing up different values within one column in the running sum or collapsing all the values in a column to a total value. But what if you want to calculate over columns instead of rows?

Enter the *axis* parameter. You've already seen this parameter twice in this chapter. It was a mandatory parameter for the COLLAPSEALL function and an optional one for the RUNNINGSUM function. To see how this parameter works, you first need to create a visual with a field dimension on the columns in the visual matrix. In the example semantic model, you can create a report page with a matrix that contains CategoryName from the Product table on the rows, YearQuarter from the Date table on the columns, and the familiar *Sales* measure. Figure 2-29 shows the resulting matrix visual and part of the Visualizations pane.



CategoryName	Q1-2024	Q2-2024	Q3-2024	Q4-2024	Q1-2025	Q2-2025	Q3-2(
Audio	$1,038,926.50	$1,352,971.01	$1,344,796.73	$3,079,619.26	$5,777,350.70	$3,872,147.54	$2,
Cameras and camcorders	$13,103,481.38	$8,068,930.08	$4,601,514.46	$7,170,858.79	$17,297,594.21	$15,264,808.87	$7,
Cell phones	$3,457,961.86	$4,192,581.24	$4,415,694.79	$9,837,632.88	$21,766,278.14	$18,930,868.88	$11,
Computers	$12,486,032.00	$17,535,223.77	$20,574,798.78	$39,850,193.68	$84,033,897.88	$68,453,202.94	$31,
Games and Toys	$309,864.12	$217,915.36	$153,667.18	$516,630.05	$1,476,095.23	$1,387,121.17	$
Home Appliances	$7,959,507.47	$7,835,943.35	$6,484,927.45	$11,627,617.65	$16,542,900.10	$8,850,746.44	$5,
Music, Movies and Audio Books	$1,530,029.18	$1,220,315.86	$937,936.91	$2,750,497.52	$5,613,217.98	$4,505,880.43	$4,
TV and Video	$6,565,100.43	$5,206,611.23	$3,226,106.96	$5,409,778.09	$8,912,555.89	$6,688,394.27	$5,
Total	**$46,450,902.94**	**$45,630,491.90**	**$41,739,443.25**	**$80,242,827.92**	**$161,419,890.14**	**$127,953,170.55**	**$67,2**

FIGURE 2-29 The matrix visual shows the sales per category and per quarter.

If you add a visual calculation to this visual, you'll notice that the visual matrix now has a different format than in all the previous examples. You'll still see the CategoryName on the rows, but because the YearQuarter is on the columns, you'll get a new column for each value of YearQuarter. If you first create a new visual calculation, you'll see two columns for each YearQuarter: Sales (with the value of the *Sales* measure) and Calculation (which is empty), as shown in Figure 2-30. This is where you'll create your new visual calculation.

For the sake of example, let's create a calculation that returns the moving average for multiple quarters by applying the MOVINGAVERAGE function and specifying an axis. The signature of this function is as follows:

```
MOVINGAVERAGE ( <Column>, <WindowSize>[, <IncludeCurrent>][, <Axis>][, <OrderBy>]
[, <Blanks>][, Reset] )
```

1 Calculation =

YearQuarter	Q1-2024		Q2-2024		Q3-2024		Q4-2024
CategoryName	Sales	**Calculation**	Sales	**Calculation**	Sales	**Calculation**	Sales
Audio	$1,038,926.50		$1,352,971.01		$1,344,796.73		$3,079,61
Cameras and camcorders	$13,103,481.38		$8,068,930.08		$4,601,514.46		$7,170,85
Cell phones	$3,457,961.86		$4,192,581.24		$4,415,694.79		$9,837,63
Computers	$12,486,032.00		$17,535,223.77		$20,574,798.78		$39,850,19
Games and Toys	$309,864.12		$217,915.36		$153,667.18		$516,63
Home Appliances	$7,959,507.47		$7,835,943.35		$6,484,927.45		$11,627,61
Music, Movies and Audio Books	$1,530,029.18		$1,220,315.86		$937,936.91		$2,750,49
TV and Video	$6,565,100.43		$5,206,611.23		$3,226,106.96		$5,409,77
Total	**$46,450,902.94**		**$45,630,491.90**		**$41,739,443.25**		**$80,242,82**

FIGURE 2-30 A new calculation was added for each YearQuarter on the columns.

> **Note** The MOVINGAVERAGE function is also available as an expression template. However, we have chosen not to use that expression template here, so you'll better grasp how the function works.

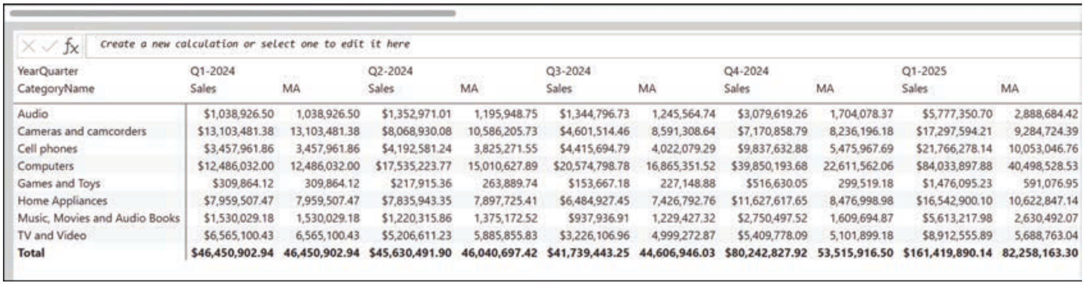
The first two parameters are mandatory. However, all the others are optional, as indicated by the square brackets. In this example, it makes sense to calculate a moving average over the last four values so that for each quarter you calculate the average for a whole year (all four previous quarters). This will help business users look at their data without a seasonality pattern. For example, if you sell ice cream, it's expected that you'll sell much more during the months when the temperature is higher. To see if your sales have increased since the previous year, you can compare this year's monthly value with the same month of the previous year, but that only gives you insight about that one month. If you calculate the average sales over the last 12 months, however, you can see per month whether your sales have increased on a yearly basis.

> **Note** The default behavior of the *axis* parameter is to calculate over the first axis in the visual shape. In most cases, this corresponds to the ROWS value that you saw in the COLLAPSEALL function, for example. If the *axis* parameter is optional in the function you are using and you omit it, it falls back to the default setting—so the first axis in the visual shape. Chapter 3 discusses the other possible values.

In our example, to calculate the moving average over the last four quarters, you can use the following visual calculation:

```
MA = MOVINGAVERAGE ( [Sales], 4, TRUE, COLUMNS )
```

The *windowSize* parameter is set to 4, indicating that you would like to average over the last four values. The *includeCurrent* parameter is set to TRUE because you want to include the current value when you calculate the moving average for the four values that you are averaging. (You could skip this, because it's an optional parameter and its default value is TRUE, but we've included it here for clarity.) Finally, the *axis* parameter is set to COLUMNS, indicating that you want to calculate this moving average over the columns. Figure 2-31 shows the result.



Create a new calculation or select one to edit it here

YearQuarter	Q1-2024		Q2-2024		Q3-2024		Q4-2024		Q1-2025	
CategoryName	Sales	MA	Sales	MA	Sales	MA	Sales	MA	Sales	MA
Audio	$1,038,926.50	1,038,926.50	$1,352,971.01	1,195,948.75	$1,344,796.73	1,245,564.74	$3,079,619.26	1,704,078.37	$5,777,350.70	2,888,684.42
Cameras and camcorders	$13,103,481.38	13,103,481.38	$8,068,930.08	10,586,205.73	$4,601,514.46	8,591,308.64	$7,170,858.79	8,236,196.18	$17,297,594.21	9,284,724.39
Cell phones	$3,457,961.86	3,457,961.86	$4,192,581.24	3,825,271.55	$4,415,694.79	4,022,079.29	$9,837,632.88	5,475,967.69	$21,766,278.14	10,053,046.76
Computers	$12,486,032.00	12,486,032.00	$17,535,223.77	15,010,627.89	$20,574,798.78	16,865,351.52	$39,850,193.68	22,611,562.06	$84,033,897.88	40,498,528.53
Games and Toys	$309,864.12	309,864.12	$217,915.36	263,889.74	$153,667.18	227,148.88	$516,630.05	299,519.18	$1,476,095.23	591,076.95
Home Appliances	$7,959,507.47	7,959,507.47	$7,835,943.35	7,897,725.41	$6,484,927.45	7,426,792.76	$11,627,617.65	8,476,998.98	$16,542,900.10	10,622,847.14
Music, Movies and Audio Books	$1,530,029.18	1,530,029.18	$1,220,315.86	1,375,172.52	$937,936.91	1,229,427.32	$2,750,497.52	1,609,694.87	$5,613,217.98	2,630,492.07
TV and Video	$6,565,100.43	6,565,100.43	$5,206,611.23	5,885,855.83	$3,226,106.96	4,999,272.87	$5,409,778.09	5,101,899.18	$8,912,555.89	5,688,763.04
Total	**$46,450,902.94**	**46,450,902.94**	**$45,630,491.90**	**46,040,697.42**	**$41,739,443.25**	**44,606,946.03**	**$80,242,827.92**	**53,515,916.50**	**$161,419,890.14**	**82,258,163.30**

FIGURE 2-31 The visual calculation successfully calculates a moving average.

To confirm that the values are as you would expect, look at the row for the Audio category. You'll see that the moving average value of Q1-2024 is the same as the sales value. This is because there are not yet any additional values to include in the moving average. The moving average for Q2-2024, however, is different from the sales value, because it now averages two values (Q1-2024 sales and Q2-2024 sales). The same is true of the moving average for Q3-2024 (which averages sales values of the first three quarters of the year) and Q4-2024 (which averages sales values of the first four quarters). The first quarter of 2025 is the interesting one. Here, you would expect the moving average to average sales values of Q2-2024 ($1.35 million), Q3-2024 ($1.34 million), Q4-2024 ($3.08 million), and Q1-2025 ($5.78 million). The average of these numbers is approximately $2.89 million, which equals the value returned by the moving average calculation for Q1-2025. So, it works exactly as anticipated.

> **Note** This example shows how a more complex calculation can be performed using just one simple line of DAX. This is exactly what visual calculations are meant to do: make DAX easier.

How does this work?

Now that you've seen a few examples of visual calculations and how easy it is to create them, and you've caught a glimpse of the possibilities associated with visual calculations, it's time for us to explain how they work.

Figure 2-32 shows the visual calculations edit mode window for a visual calculation, with all relevant elements highlighted. These include the following:

- **Visual preview** The visual preview shows you what the visual will look like when you leave the visual calculations edit mode and return to your report. You can see what impact newly added visual calculations have on your visual, and what your visual will look like if you hide certain measures or visual calculations.
- **Visual matrix** The visual matrix is the data representation of your visual. As mentioned, it shows you the outcomes of all newly added calculations.
- **Formula bar** You use the formula bar to write and edit visual calculations.

- **Panes** The visual calculations edit mode window includes the same panes as the report, such as the Visualizations pane (shown here). You use these panes to select other visual calculations that you might want to edit or to format them.

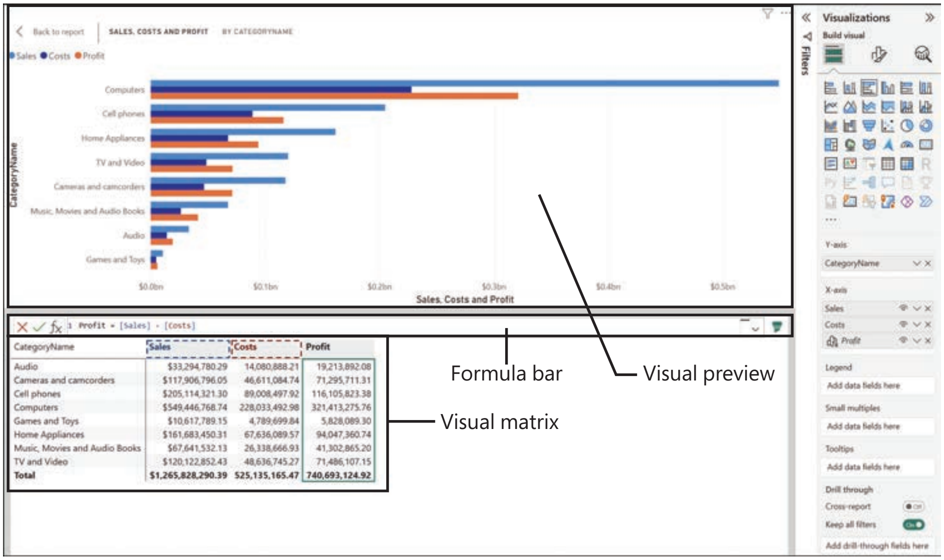


FIGURE 2-32 The visual calculations edit mode window consists of many elements.

The most important concept you need to grasp is that every visual—whether it's a clustered bar chart, a table, or even a simple card visual—can be viewed as a visual matrix, with the different dimensions of the visual on the visual matrix's rows and columns, and the calculated values for each combination of dimensions within the visual matrix. Viewing a visual as a visual matrix enables you to add more calculations, which can be seen as new columns in the visual matrix.

It's important to note that you can only use the values that are already inside a visual matrix for any new calculations you want to add. You cannot create new calculations based on measures or dimensions that are not present in the visual matrix. You can, however, add the necessary measures in the visual and then hide those measures so that they don't appear in the visual, but they do exist in the visual matrix and can therefore be used in new calculations. It's even possible to create a visual that displays only visual calculations, as you saw in the year-to-date sales example. In this approach, you can hide intermediate steps in your calculations and show only the final result.

As you have seen, you can perform a calculation in two different ways: by writing a DAX expression that uses measures or visual calculations that are already present in the visual, or by applying expression templates, which provide you with the structure of the calculation, leaving you to fill in the specifics by typing or using the provided parameter pickers.

Power BI includes several new functions created specifically for visual calculations. These include the RUNNINGSUM, COLLAPSEALL, and MOVINGAVERAGE functions discussed in this chapter, but (spoiler alert) there are many more! Part II describes these functions in greater detail.

These new functions include some new parameters, like the *axis* and *reset* parameters, which you saw earlier in this chapter. Both of these parameters are discussed in greater detail in Chapter 3. The *reset* parameter enables you to reset your visual calculation so it starts again from a specific data point. This is very useful if, for example, you are calculating year-to-date values for multiple years in the same visual. As for the *axis* parameter, it enables you to navigate within the visual matrix in various ways, such as by rows, by columns, or other patterns (discussed further in Chapter 3).

In summary

Chapter 1 explained why visual calculations were created: to enable every end user to get more out of Power BI *without* having to master DAX. The purpose of this chapter was to demonstrate this using a few basic examples.

This chapter showed you how easy it is to create visual calculations for existing visuals in a Power BI report, without the need to write complex DAX measures. If you are familiar with Excel formulas, adding visual calculations to your Power BI model should be easy. Alternatively, you can use one of several expression templates to create quite complex calculations with just a few clicks.

The learning curve for visual calculations is short, enabling every business user to create quick insights based on the existing reports and enhanced with visual calculations. Moreover, because the outcome of the new visual calculation appears in the visual calculation edit mode window's visual preview and in the visual itself, you're not left to sift through a big table of data to glean these insights. Rather, you can visualize them directly in your report. You can even create complex calculations with multiple steps and hide intermediate results. In this way you can more easily follow exactly what is being calculated in the visual calculation edit mode window but display only the final result on the report page.

Visual calculations also offer a performance improvement compared to measures because visual calculations work only on the data available in the visual matrix (which is usually a small set). Chapter 9, "Breaking down visual calculations execution," explains this in more detail.

We hope you have grasped some of the possibilities that arise with visual calculations. The rest of this book dives deeper into these possibilities, as well as important concepts that pertain to visual calculations (Chapter 3), functions used for visual calculations (Part II), and how visual calculations compare to other calculation options within Power BI (Chapter 8, "Comparing calculation options").

Index

N

O

P

Q

R

S

T

V

W