

Power BI — for — Jobseekers

Learn how to create interactive dashboards and reports,
and gain insights from the data



Alan Murray



Power BI for Jobseekers

Learn how to create interactive dashboards and reports, and gain insights from the data

Alan Murray



www.bpbonline.com

Copyright © 2023 BPB Online

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior written permission of the publisher, except in the case of brief quotations embedded in critical articles or reviews.

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the author, nor BPB Online or its dealers and distributors, will be held liable for any damages caused or alleged to have been caused directly or indirectly by this book.

BPB Online has endeavored to provide trademark information about all of the companies and products mentioned in this book by the appropriate use of capitals. However, BPB Online cannot guarantee the accuracy of this information.

First published: 2023

Published by BPB Online

WeWork

119 Marylebone Road

London NW1 5PU

UK | UAE | INDIA | SINGAPORE

ISBN 978-93-55518-149

www.bpbonline.com

Dedicated to

My amazing children:

George & Lily

About the Author



Alan Murray is a Microsoft MVP and Excel & Power BI trainer. He has been training and consulting in Microsoft technologies for over 20 years. He loves training and the joy he gets from knowing he is making peoples' working lives easier.

Alan runs his own blog - **Computergaga** (<https://computergaga.com>) and writes for multiple other websites. His Computergaga YouTube channel has over 600 videos and over 40 million views.

He organizes a free monthly **Excel & Power BI meetup** in London where anyone can come, learn, chat, and enjoy each other's company.

About the Reviewers

- ❖ **Dinesh Manoharan** is a veteran in the field of Data and Analytics and has a broad range of expertise in Power BI, Qlik, and Tableau reporting and data storytelling. Has traction towards predictive and prescriptive analytics using R, Python, and Azure ML. He helped customers to make data-driven decisions and possesses good hands-on experience in design, data modeling, performance tuning, and turning data into actionable insights.

As part of the BI Centre of Excellence team, Dinesh is a certified data analyst working in various domains to drive data culture, with his deep data knowledge reduced data-to-insights turnaround time. He actively handles training sessions on Data literacy and Analytical tools.

He likes to play badminton, cycling, listen music and admire nature."

- ❖ **Preeti Sahu** is an expert in Power BI and has an extensive background working with SharePoint and the Power Platform. She is currently employed by TSinfo Technologies as a Power Platform specialist for Power BI, Power Apps, Power Automate, and Power Virtual Agents. She is the author of the Microsoft Power Platform A Deep Dive book and the co-author of the SharePoint Online Modern Experience Practical Guide book. Also, she contributed a significant number of technical articles on the Power Platform to SPGuides.com and EnjoySharePoint.com. In addition to traveling, she enjoys spending time with her family during her free time.

Acknowledgement

I would like to thank the Power BI community – the experts, professionals, learners, and Microsoft. We learn and grow together. Without Power BI and the amazing community providing solutions to others' problems and pushing Microsoft to constantly make it better, there would be no book.

A special thanks to those within my own community. Subscribers of my Computergaga YouTube channel, blog, and members of my London Excel Meetup group. Together we learn and you inspire me to be the best I can be. Thank you for holding me accountable for the quality of my work and for your support.

My gratitude to the team at BPB Publications for being supportive throughout the writing of this book. In this constantly evolving space, and with software updating monthly, there were numerous times when an update required that I go back and update chapters. I needed their support through this demanding process.

Finally, as always, thank you to my children, George, and Lily. I'm very lucky.

Preface

Data is everywhere!

Data is a precious commodity for businesses and there has never been a more exciting time for those who like to work with data.

Businesses store huge volumes of data, which they use to make decisions to stay competitive in their field. This means there is a large demand for people with the skills to make sense of that data.

Having data is useless without meaning. The ability to create reports that result in better understanding and therefore drive effective decision-making is in high demand. Businesses want to maximize the value of their data.

Power BI Desktop is the best tool for self-service business intelligence and analytics. This free tool makes it easy to acquire, shape, model, and present data in an effective and interactive way.

The Power BI Service then provides a cost-effective way to share these reports with other members of your team or with management.

Learning Power BI will give you the edge over others who still use over-priced, rigid, and limited tools. The Microsoft data platform is the leader in the analytics and BI market.

This book will demonstrate the demand for data analytics skills by businesses today, where you can find work and what employers will expect from you.

It then takes you on a practical guide to Microsoft Power BI. No prior knowledge is required. This is a comprehensive beginners guide to Power BI.

Power BI for Job Seekers is divided into **19 chapters**.

Chapter 1: Why Learn Power BI? - The first chapter of this book explains why you should learn Power BI. This is supported by data from job searches, market research and the words of current professionals from recruitment and consultancy firms working with Power BI and data analysis.

Chapter 2: What is Power BI? - In this chapter we start to explore Power BI. We start by explaining the different components of Power BI. We then download and install Power BI Desktop and create a Power BI account.

Chapter 3: Getting Started with Power BI Desktop – In this chapter, we get started using Power BI Desktop. We familiarise ourselves with the Power BI Desktop environment, create a new Power BI file, and explore important options and settings.

Chapter 4: Creating a Simple Power BI report - Things now get very exciting. In this chapter, we create our first Power BI report. This is a simple, but effective report, to showcase the general process of how to create a report in Power BI. We will connect to external data, perform transformations to the data, build visuals and specify how the reader can interact with them.

Chapter 5: Getting and Shaping Data - In this chapter, we look at how to get data from a variety of sources into Power BI. These sources include from PDF files, Excel workbooks, OneDrive, and local folders. Various transformations will be performed to shape the data before loading it to Power BI. This is the beginning of the report that we will focus on creating throughout this book.

Chapter 6: More Data Transformations- In this chapter, we explore further data transformations that I feel deserve some coverage in this book. These transformations are incredibly useful. They include merge queries and the different join types, performing date transformations, calculations in Power Query, and the brilliant unpivot columns feature.

Chapter 7: Creating the Data Model - In this chapter, we fine-tune the data model ready for the DAX calculations and report visuals that are to come. This includes establishing relationships between the tables of the data model, hiding unnecessary fields, and formatting field values correctly.

Chapter 8: Creating a Date Table - In this chapter, we continue building the data model of our report by adding a date table. We begin by detailing why you need a date table in your model and what one consists of. We then create the date table using the DAX formula language and add the columns that we require. We then ensure that the columns are ordered correctly and mark the table as a date table. This ensures that Power BI works effectively with our new date table and does not create its own date tables to perform time intelligence.

Chapter 9: Adding DAX Measures - In this chapter, we delve further into the DAX formula language and look at measures. Measures are DAX calculations that aggregate the values for many rows of a table. We will create measures that we require for our report building in the forthcoming chapters. You will learn a few very useful DAX functions including CALCULATE, RELATED, COUNTROWS, SUMX, and DIVIDE to name a few.

Chapter 10: Cards and Other Text Visuals - In this chapter, we begin looking at the visualizations available in Power BI to present your data effectively. This chapter focuses on text-based visuals, of which there are four that will be covered – the Card, KPI, Table and Matrix.

Chapter 11: Chart Visuals - This chapter will focus on using the chart visuals in Power BI including the column, line, combo, gauge, and treemap visuals. You will see practical examples of their use and learn key formatting attributes that can be changed to increase the effectiveness of the visual.

Chapter 12: Using Maps in Power BI Reports - In this chapter, we look at the map visual of Power BI and use it to efficiently visualize geographic data such as countries, cities, and postcodes. You will learn a few tips to get the most out of your Power BI maps.

Chapter 13: Other Power BI Visualizations - In this chapter, we dive into the AI visuals of Power BI including the Q&A, Decomposition Tree, and Smart Narrative visuals. We also cover how to import custom visuals from the AppSource marketplace to cater for scenarios when the standard visuals do not fulfil your needs.

Chapter 14: Report Interactions, Filters & Slicers - This chapter is about the different ways that we can enable a user to interact and filter the visuals on report pages. We cover visual interactions, slicers, the Filters pane, and Drill through.

Chapter 15: Enhancing your Power BI Reports - In this chapter, we will make some final enhancements to our report before we publish it to the Power BI service and share it with others. We start with inserting objects such as text boxes, images, and buttons. Then we move to creating custom tooltips pages and using bookmarks in your Power BI reports.

Chapter 16: Publishing & Sharing your Reports - In this chapter, we publish our report and explore the Power BI Service. We look at how to share our report with

others both inside and outside of the organization and create new reports from existing datasets that have been published.

Chapter 17: Datasets, Dashboards & Reports - In this chapter, we will dive into more details on some of the key features of the Power BI service – datasets, dashboards, and reports. We will see how to create your own workspaces and specify the access for different users and groups and the permissions that they have. We will also cover the important topic of how to refresh your data and your reports.

Chapter 18: Power BI and Other Apps - This chapter looks at some associated tools to Power BI that are important to know. A key topic is the way that Power BI interacts with Excel. It is very easy to transfer your data models between Excel and Power BI and vice versa. We will also look at how Power BI can interact with PowerPoint and with Microsoft Teams.

Chapter 19: Interview Questions, Certifications & Resources - The final chapter of the book looks at how to land your analysis job. We ask recruiters what they are looking for when hiring an analyst including typical interview questions, experience, and skills, so that you can be fully prepared. We also look at useful certifications and further learning and development opportunities.

Code Bundle and Coloured Images

Please follow the link to download the *Code Bundle* and the *Coloured Images* of the book:

<https://rebrand.ly/dbgic3o>

The code bundle for the book is also hosted on GitHub at **<https://github.com/bpbpublications/Power-BI-for-Jobseekers>**. In case there's an update to the code, it will be updated on the existing GitHub repository.

We have code bundles from our rich catalogue of books and videos available at **<https://github.com/bpbpublications>**. Check them out!

Errata

We take immense pride in our work at BPB Publications and follow best practices to ensure the accuracy of our content to provide with an indulging reading experience to our subscribers. Our readers are our mirrors, and we use their inputs to reflect and improve upon human errors, if any, that may have occurred during the publishing processes involved. To let us maintain the quality and help us reach out to any readers who might be having difficulties due to any unforeseen errors, please write to us at :

errata@bpbonline.com

Your support, suggestions and feedbacks are highly appreciated by the BPB Publications' Family.

Did you know that BPB offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at www.bpbonline.com and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at :

business@bpbonline.com for more details.

At www.bpbonline.com, you can also read a collection of free technical articles, sign up for a range of free newsletters, and receive exclusive discounts and offers on BPB books and eBooks.

Piracy

If you come across any illegal copies of our works in any form on the internet, we would be grateful if you would provide us with the location address or website name. Please contact us at **business@bpbonline.com** with a link to the material.

If you are interested in becoming an author

If there is a topic that you have expertise in, and you are interested in either writing or contributing to a book, please visit **www.bpbonline.com**. We have worked with thousands of developers and tech professionals, just like you, to help them share their insights with the global tech community. You can make a general application, apply for a specific hot topic that we are recruiting an author for, or submit your own idea.

Reviews

Please leave a review. Once you have read and used this book, why not leave a review on the site that you purchased it from? Potential readers can then see and use your unbiased opinion to make purchase decisions. We at BPB can understand what you think about our products, and our authors can see your feedback on their book. Thank you!

For more information about BPB, please visit **www.bpbonline.com**.

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



Table of Contents

INTRODUCTION

1. Why Learn Power BI?	1
Introduction.....	1
Structure.....	1
Objectives.....	2
Gartner Magic Quadrant 2022.....	2
Demand for Power BI skills.....	3
The words of current professionals.....	6
Conclusion.....	8
Points to remember.....	9
2. What is Power BI?	11
Introduction.....	11
Structure.....	12
Objectives.....	12
The three components of Power BI.....	12
<i>Power BI desktop</i>	13
<i>Power query</i>	13
<i>Model the data</i>	14
<i>DAX</i>	15
<i>Visualizations</i>	15
<i>Publish your reports</i>	16
Power BI service.....	16
<i>Workspaces</i>	17
<i>Datasets</i>	18
<i>Reports</i>	18
<i>Dashboards</i>	18
<i>Apps</i>	19
<i>Power BI mobile</i>	19
Installing power BI desktop.....	19
<i>Downloading from the Microsoft store</i>	20

<i>Downloading directly from Microsoft.com</i>	21
Creating a Power BI account	24
Conclusion	25
Questions	25
3. Getting Started with Power BI Desktop	27
Introduction	27
Structure	27
Objectives	27
A first look at Power BI Desktop	28
<i>Welcome screen</i>	28
<i>The Report view</i>	29
<i>The Data view</i>	31
<i>The Model view</i>	33
Exploring useful options and settings	34
<i>Data load</i>	35
<i>Preview features</i>	37
<i>Regional settings</i>	38
<i>Query reduction</i>	38
<i>Report settings</i>	40
Saving a Power BI Desktop file	41
Conclusion	41
Questions	42
4. Creating a Simple Power BI Report	43
Introduction	43
Structure	43
Objectives	44
Getting data from the Web	44
Transforming data in Power Query	47
<i>Removing columns</i>	48
<i>Splitting columns by a delimiter</i>	50
<i>Converting negative values to positive</i>	52
<i>Using trim to remove excess spaces</i>	53
<i>Renaming columns</i>	54
<i>Changing column data types</i>	54

<i>Renaming steps</i>	57
Applying Queries in Power BI.....	58
Creating a column chart	59
Editing existing queries	62
<i>Using column from examples</i>	63
Modifying the column chart visual.....	65
Changing the order of the X-axis.....	67
Creating a table visual.....	68
Interactions between the visuals	74
Conclusion	75
Questions	75
5. Getting and Shaping Data.....	77
Introduction.....	77
Structure.....	77
Objectives.....	78
Creating a New Power BI file.....	78
Getting data with Power Query	79
Import versus DirectQuery mode	80
Getting data from an Excel Workbook	81
<i>Removing rows</i>	83
<i>Splitting columns</i>	85
<i>Data types and renaming columns</i>	87
Getting data from PDF files.....	87
<i>Appending queries</i>	89
<i>Promote the headers of a table</i>	92
<i>Merging columns</i>	93
<i>Data types and renaming columns</i>	95
Importing data from multiple files in a folder	95
<i>Combine the data from all files into one table</i>	98
<i>Removing unwanted characters</i>	101
<i>Changing the case of text</i>	104
<i>Data types and renaming columns</i>	105
Importing data from files stored on OneDrive/SharePoint.....	105
<i>Combine all tables into one</i>	110
<i>Data types and renaming columns</i>	112

Managing data source settings	114
<i>Data sources in the current file</i>	114
<i>Changing global permissions</i>	116
<i>Changing source in the Power Query Editor</i>	117
Conclusion	119
Questions	119
6. More Data Transformations	121
Introduction	121
Structure	122
Objectives	122
Merge queries	122
<i>The different join kinds</i>	122
<i>Left Outer Join — adding a column to a table</i>	124
<i>Left Anti Join — removing rows from a table</i>	129
Establishing the date data type	132
<i>Using the locale data type</i>	132
<i>Converting ISO 8601 Dates</i>	135
Calculations in power query	137
<i>Mathematical calculations</i>	138
<i>Rounding values</i>	140
<i>Adding and editing M code</i>	141
<i>Changing the rounding method</i>	142
<i>Defining a data type</i>	144
<i>Date functions — calculating age</i>	145
<i>Inserting conditional columns</i>	148
Unpivoting columns	151
Conclusion	155
Questions	155
7. Creating the Data Model	157
Introduction	157
Structure	158
Objectives	158
Why create table relationships?	158
Cardinality	160

Creating table relationships	161
Managing the relationships of a model	164
Disabling queries from loading	166
Formatting table columns.....	169
Hiding table fields from the Report view	172
Deleting fields from a model.....	175
Conclusion	177
Questions	177
8. Creating a Date Table.....	179
Introduction.....	179
Structure.....	179
Objectives.....	180
Why create a date table?	180
Introduction to DAX.....	183
CALENDARAUTO and CALENDAR functions	184
<i>CALENDARAUTO function</i>	186
<i>CALENDAR function</i>	189
Creating additional date columns.....	190
<i>Year</i>	191
<i>Month</i>	191
<i>Year and month</i>	193
<i>Day of week name</i>	193
<i>Is it a weekend date or not?</i>	195
<i>Fiscal years</i>	195
<i>Fiscal quarters</i>	196
<i>Fiscal months</i>	197
<i>Week number of the year</i>	198
<i>Create the date table with one formula</i>	200
Sorting table columns by another column	201
Creating the relationships to the date table	203
Marking a table as a date table	204
Hiding other date fields.....	205
Conclusion	206
Questions	207

9. Adding DAX Measures	209
Introduction.....	209
Structure.....	210
Objectives.....	210
What is a measure in Power BI?	210
Implicit versus explicit measures	211
The COUNTROWS function.....	216
Organizing your measures	217
<i>How do I move a measure?</i>	218
<i>Creating folders within tables</i>	219
<i>Creating a measure table</i>	221
SUMX and other iterator functions.....	223
Returning the previous months revenue.....	225
<i>The CALCULATE function</i>	226
<i>The DATEADD function</i>	226
<i>Writing the measure</i>	227
Revenue difference between the two months	229
Calculating the percentage revenue change	231
Creating a year-to-date total	232
Conclusion	234
Questions	235
10. Cards and Other Text Visuals	237
Introduction.....	237
Structure.....	237
Objectives.....	238
The Card visual.....	238
<i>Formatting the callout value</i>	239
<i>Category label versus title</i>	240
<i>Adding a border</i>	242
<i>Format painter</i>	243
KPI	245
<i>Inserting a Slicer</i>	247
<i>Removing the icon</i>	249
<i>Trend axis options</i>	250
<i>Changing the target label</i>	251

<i>Adding the title and border</i>	252
Table visual	253
<i>Formatting the grid and values</i>	254
<i>Formatting the column headers</i>	256
<i>Sorting table data</i>	257
<i>Sorting a table by multiple columns</i>	258
Matrix	260
<i>Formatting the grid and values</i>	261
<i>Formatting the column headers</i>	262
<i>Drill down actions for multiple row headers</i>	263
Conclusion	266
Questions	266
11. Chart Visuals	269
Introduction.....	269
Structure.....	269
Objectives.....	270
Column and bar charts	270
<i>Clustered column chart</i>	270
<i>Adding tooltips</i>	271
<i>Formatting the clustered column chart</i>	273
<i>Changing the chart title</i>	273
<i>Adding a border</i>	275
<i>Changing the column colors</i>	276
<i>Removing axis and axis titles</i>	276
Adding data labels	277
Stacked column chart.....	278
Switching to a bar chart.....	280
Line and area charts	280
<i>Line chart</i>	281
<i>Line chart with multiple data series</i>	283
<i>Stacked area chart</i>	285
Combo charts.....	285
Pie and donut charts.....	289
Gauge charts.....	292
<i>General formatting</i>	294

<i>Visual formatting</i>	294
<i>KPI versus gauge charts</i>	295
Treemap visual	297
Conclusion	299
Questions	300
12. Using Maps in Power BI Reports	301
Introduction	301
Structure	301
Objectives	302
Enable map visuals in Power BI	302
The Map visual	303
<i>Using the Map visual</i>	304
<i>Editing the map title</i>	308
<i>Changing the map style</i>	309
<i>Formatting the bubbles</i>	310
<i>Using a table to filter the map data</i>	310
<i>Disabling the auto-zoom feature</i>	312
Tips for using the map visuals	313
<i>Categorize the geographic fields</i>	314
<i>Using latitude and longitude data</i>	315
<i>Using geographic hierarchies</i>	319
Conclusion	320
Questions	321
13. Other Power BI Visualizations	323
Introduction	323
Structure	324
Objectives	324
The Q&A visual	324
<i>Using the Q&A Visual</i>	324
<i>Formatting the Q&A</i>	330
<i>Configuring the Q&A visual</i>	331
<i>Adding synonyms</i>	332
<i>Creating suggested questions</i>	335
Decomposition tree	336

Smart narrative.....	339
<i>Creating the smart narrative.....</i>	339
<i>Adding and formatting dynamic values</i>	342
Custom visuals in Power BI.....	345
<i>Adding a custom visual from AppSource</i>	345
<i>Removing a custom visual</i>	349
Conclusion.....	350
Questions	351
14. Report Interactions, Filters, and Slicers.....	353
Introduction.....	353
Structure.....	354
Objectives.....	354
Editing visual interactions.....	354
Slicers.....	361
<i>Using date fields in a slicer.....</i>	361
<i>Using text fields in a slicer</i>	366
<i>Filtering visuals on other pages.....</i>	370
The Filters Pane.....	371
<i>Setting a Top N filter</i>	372
<i>Filter all visuals on a page or all pages.....</i>	374
Using drill through.....	376
<i>Creating a drill through.....</i>	377
<i>Performing the drill through</i>	378
<i>Adding a page title.....</i>	380
Conclusion.....	382
Questions	383
15. Enhancing Your Power BI Reports.....	385
Introduction.....	385
Structure.....	385
Objectives.....	386
Adding text boxes, images, and buttons.....	386

<i>Adding text boxes</i>	386
<i>Adding images</i>	387
<i>Adding buttons</i>	390
<i>Changing the drill through back button</i>	391
<i>Page navigator buttons</i>	393
<i>Adding a background</i>	395
Custom tooltip pages	397
<i>Creating a custom tooltip page</i>	399
<i>Adding visuals to the custom tooltip page</i>	401
<i>Adding the tooltip page to a visual</i>	404
Using bookmarks	405
<i>Creating bookmarks</i>	406
<i>Inserting buttons to run the bookmarks</i>	410
Conclusion	411
Questions	411
16. Publishing and Sharing Your Reports	413
Introduction	413
Structure	413
Objectives	414
Publishing your Report	414
Accessing the report on the service	416
Datasets and reports	419
<i>Creating a report from an existing dataset</i>	420
<i>From the Power BI Service</i>	420
<i>From Power BI Desktop</i>	423
Share a link to the report	425
<i>Managing report permissions</i>	427
Conclusion	429
Questions	429
17. Datasets, Dashboards, and Reports	431
Introduction	431

Structure	431
Objectives	432
Workspaces	432
<i>Creating a new workspace</i>	432
<i>Managing access to a workspace</i>	434
<i>Copying a report to another workspace</i>	437
Refreshing datasets and reports	439
Power BI data gateways	439
<i>Installing the standard mode gateway</i>	440
<i>Adding data sources to a gateway</i>	445
<i>Refreshing your data</i>	451
<i>Scheduling data refreshes</i>	452
Dashboards	454
<i>Creating a dashboard</i>	454
<i>Pinning visuals to a dashboard</i>	455
<i>Sharing a dashboard</i>	457
Conclusion	459
Questions	460

18. Power BI and Other Apps.....	461
Introduction	461
Structure	461
Objectives	462
Power BI and Excel	462
Creating Excel Reports from Power BI datasets	462
<i>Using Analyze in Excel</i>	462
<i>Connecting to Power BI datasets from Excel</i>	466
Import Power Pivot models into Power BI	469
<i>From Power BI Desktop</i>	470
<i>From the Power BI service</i>	472
Export to PowerPoint	474
<i>Embed an image</i>	475

<i>Embed live data</i>	477
Publishing to teams	479
<i>Power BI App in Microsoft teams</i>	480
<i>Embed reports in Teams channels and chats</i>	481
Conclusion	483
Questions	483
19. Interview Questions, Certifications, and Resources	485
Introduction.....	485
Structure	485
Objectives.....	486
Interview questions	486
Knowledge.....	486
<i>Experience</i>	487
<i>Passion</i>	488
Certifications	488
Useful resources	489
<i>Websites</i>	489
<i>YouTube channels</i>	490
<i>Podcasts</i>	490
<i>Meetups and conferences</i>	490
Conclusion	490
Questions	491
Multiple Choice Questions Answers	493
Answers	493
<i>Chapter 2</i>	493
<i>Chapter 3</i>	493
<i>Chapter 4</i>	494
<i>Chapter 5</i>	494
<i>Chapter 6</i>	494
<i>Chapter 7</i>	495
<i>Chapter 8</i>	495

<i>Chapter 9</i>	495
<i>Chapter 10</i>	496
<i>Chapter 11</i>	496
<i>Chapter 12</i>	496
<i>Chapter 13</i>	496
<i>Chapter 14</i>	497
<i>Chapter 15</i>	497
<i>Chapter 16</i>	497
<i>Chapter 17</i>	498
<i>Chapter 18</i>	498
<i>Chapter 19</i>	498
Index	499-505

Introduction

Data is everywhere!

Data is a precious commodity for businesses, and there has never been a more exciting time for those who like to work with data.

Businesses store huge volumes of data, which they use to make decisions to stay competitive in their field. This means there is a large demand for people with the skills to make sense of that data.

Having data is useless without meaning. The ability to create reports that result in better understanding and therefore drive effective decision-making is in high demand. Businesses want to maximize the value of their data.

Power BI Desktop is the best tool for self-service business intelligence and analytics. This free tool makes it easy to acquire, shape, model, and present data in an effective and interactive way.

The Power BI Service then provides a cost-effective way to share these reports with other members of your team or with management.

Learning Power BI will give you the edge over others who still use over-priced, rigid, and limited tools. The Microsoft data platform is the leader in the analytics and BI market.

This book will demonstrate the demand for data analytics skills by businesses today, where you can find work, and what employers will expect from you.

It then introduces you to the different tools that form Microsoft Power BI. No prior knowledge is required. This is a comprehensive beginner's guide to Power BI.

Notation

You can expect the following notation in this book.

- The name of a button, keystrokes will be written in Italics. For example, press *Enter*.
- Tab of the Ribbon that we click will be written as Calibri +Font 12 + Bold text. For example, click the **Modelling** tab of the Ribbon.
- Titles and other text that you see on screen and referenced in the book will be written as Calibri +Font 12 + Bold. For example, in the **Merge** window.
- Names of tables, queries, and pages will be shown as Consolas + Font 10 + bold. For example, select the **Products** table.
- Names of fields, columns, and measures will be shown as Consolas + Font 10 + bold . For example, click on the **No of Staff** column.
- Written text will be written as Consolas + Font 10 + Bold. For example, type **Total Revenue** in the name box.

Download the example files

You are encouraged to download the example files used by me throughout the book to practice on. The best way to learn is to follow along and practice the techniques with me. This will benefit your learning greatly.

You can download the example files from the [GitHub link](#).

The files are organized into folders that match the chapters of this book.

CHAPTER 1

Why Learn Power BI?

Introduction

Power BI is quite simply the best self-service BI platform. It is a service for data acquisition, modeling, and reporting. It is easy to learn, is in large demand by employers, and is updated regularly.

Power BI is updated every month, so it is continually getting more powerful, easier to use and takes advantage of advancements in technology. This makes it a very exciting time to start using Power BI if you are seeking a job in business intelligence or data analysis.

In this book, we will learn how to use Power BI effectively, and you will see how easy and fun it can be.

Structure

In this chapter, we will cover the following topics:

- View data that supports the demand for Power BI skills
- Hear from current BI professionals on the importance of PBI

Objectives

After reading this chapter, you will be able to know why you should learn Power BI. You will see data that supports the demand for Power BI skills from popular job sites and other places that you can find work for Power BI.

You will see impartial evidence that Microsoft is the leader in analytics and business intelligence platforms.

And you will also hear from current professionals who work in the BI consultancy or in the recruitment of analysts and other BI professionals.

Gartner Magic Quadrant 2022

In March 2022, Microsoft was recognized as a leader in the Gartner Magic Quadrant for Analytics and Business Intelligence Platforms (*figure 1.1*).

They have been named as a leader for 15 consecutive years. This is the fourth year that they have been positioned the furthest to the right for the completeness of vision and the furthest up for their ability to execute. Please refer to the following figure:



Figure 1.1 Gartner Magic Quadrant showing Microsoft as an ABI leader

This impartial view of different business intelligence platforms provides the confidence that Microsoft's ecosystem of tools and online services is a leader in this landscape.

Although Power BI is not solely responsible for this position in the quadrant, it is a key reason.

Gartner evaluates the quadrant across 15 categories, including data source connectivity, data preparation, data visualization, natural language query, and reporting.

You will see as you progress through this book that Power BI performs highly in all these categories.

Demand for Power BI skills

The demand for Power BI is very strong and is growing year-on-year.

Whether you are looking for a permanent position or contract work, full-time or as a side gig, there are opportunities for you.

A quick search for jobs confirms this demand.

Figure 1.2 shows a search on LinkedIn UK for jobs using the keyword *Power BI*. It returned 4,122 results. Please refer to the following figure:

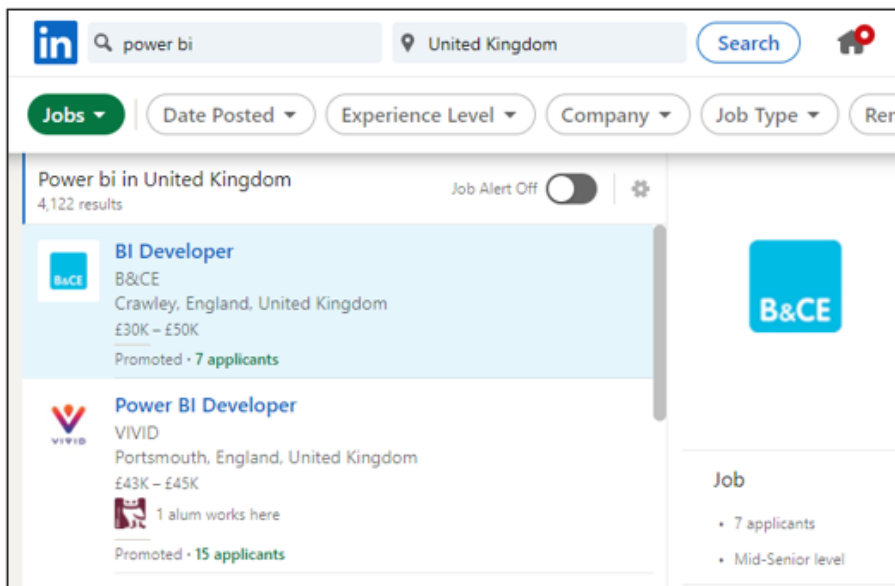


Figure 1.2: Power BI jobs at LinkedIn UK

These results can be filtered by date, location, and experience level, and you can even specify to return roles where you can work remotely.

Figure 1.3 shows a search on Naukri.com, India's No.1 job site, for jobs with the keywords Power BI and data analyst. This search returned 11,931 results.

This search is set to any experience level. So, if you are an entry-level BI analyst or developer, this could be set to 0 years to find the appropriate role. Please refer to the following figure:

The screenshot displays the Naukri.com website interface. At the top, the logo 'naukri.com' is visible along with navigation links for 'JOBS', 'RECRUITERS', 'COMPANIES', 'TOOLS', 'SERVICES', 'MORE', and 'LOGIN'. A search bar contains the text 'Showing jobs for 'power bi, data analyst' Modify'. Below the search bar, there are filter sections for 'All Filters', 'Freshness', 'Location', 'Salary', and 'Experience'. The 'Experience' filter is set to 'Any'. The search results are sorted by 'Relevance' and show 1-20 of 11931 jobs. Three job listings are visible: 'Data Analyst / Power BI / Master data Management (rotational shift)' by Bureau Veritas India Pvt Ltd, 'Power BI Developer/Data Analyst/Data Engineer' by Content Bloom, and 'Data Analyst - Power BI' by Vanguard Talent Hub. Each listing includes details such as years of experience, salary, location, and a 'Save' button.

Figure 1.3: Power BI data analyst jobs at Naukri.com

This next search is on the website of Nigel Frank International, a global leader in Microsoft Recruitment (figure 1.4).

This has returned 589 jobs just from that one recruitment company. This was an international search. Please refer to the following figure:

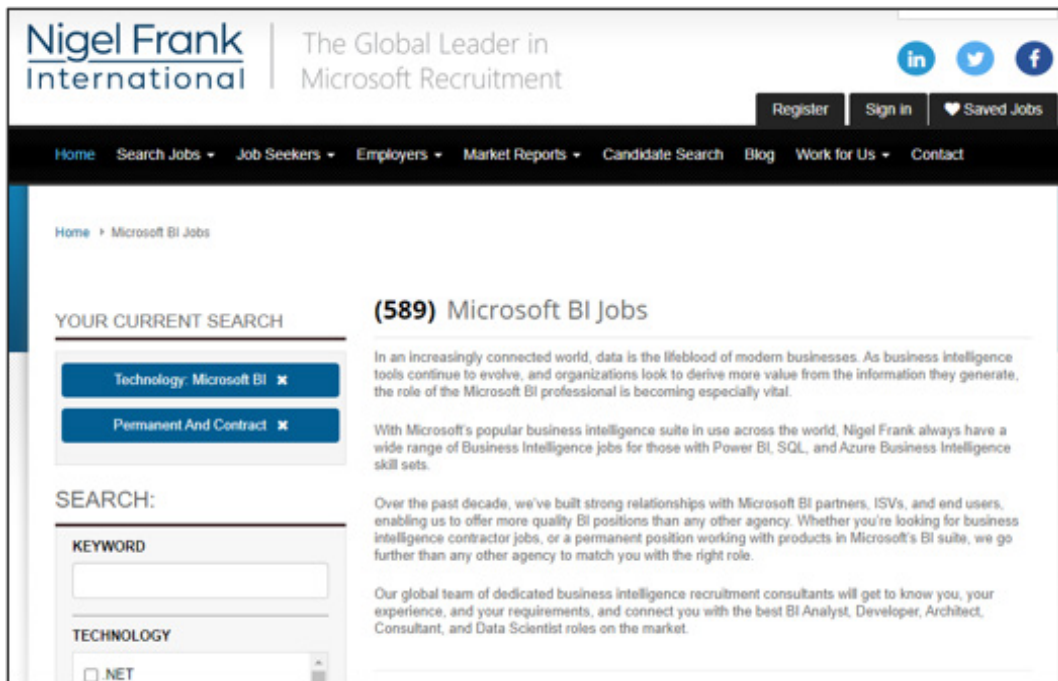


Figure 1.4: Microsoft BI jobs at Nigel Frank International

Just by searching a few common job sites and recruitment company sites returned thousands of opportunities at different experience levels.

Another option when searching for jobs is to search for freelance sites.

These are especially useful if you would like contract work on the side or you are beginning on your Power BI journey.

Figure 1.5 shows a search on Upwork using the keyword Power BI, and it returned 778 jobs to browse and bid for.

You can see filters on the left for experience level, job type, and the number of proposals received.

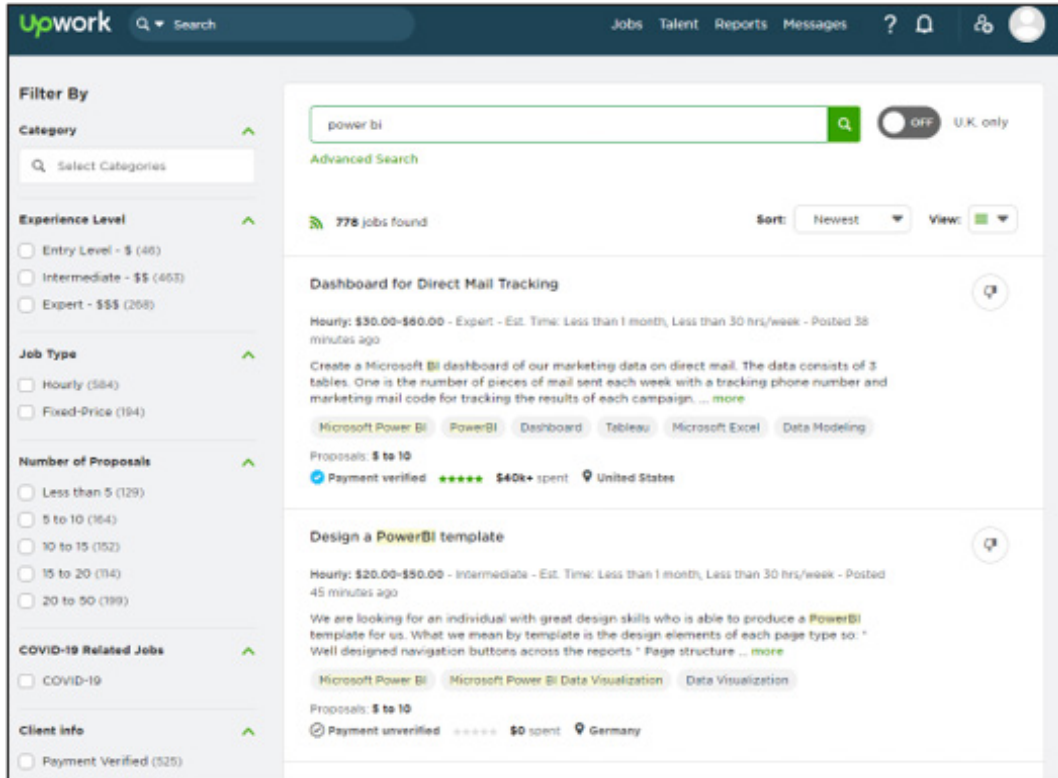


Figure 1.5: The demand for Power BI professionals on Upwork

The jobs can vary from short jobs you can complete in a few hours to long-term contracts. These jobs are a fantastic way to build experience and a portfolio of work for a beginner.

The words of current professionals

To get the inside scoop, I reached out to a few analytics and BI consultancy firms and recruiters to see what they had to say.

I was interested to hear their opinions on the demand for Power BI and any advice they could offer:

Wyn Hopkins

Director, Access Analytic

Wyn Hopkins is the director of Access Analytics based in Perth, Western Australia. He is a Power BI and Excel consultant and trainer. Access Analytic has employed four new BI consultants in the last few years.

“Power BI is taking over everywhere,” says Wyn when asked about the role of Power BI in the future of data analysis and BI.

Access Analytic is a Microsoft Power BI Partner, and their team of consultants and trainers provide solutions for their clients primarily using Power BI and with related tools such as Excel and SharePoint.

“The world of Power BI is very exciting. Regular updates to Power BI mean that it is constantly getting better, and there is always more to learn. The technology has led to a dramatic shift in the speed that reporting solutions can be developed, days and weeks instead of months and years.”

“Power BI offers everything we need, and its integration with Excel and the low entry price is very compelling for businesses looking to improve their reporting and business insights.”

Wyn is a Microsoft MVP, co-organizes the Perth Modern Excel and Power BI user group and is a contributor to the Power BI and Excel communities. So, I was not surprised when he made a special mention of the community.

“The Power Platform community is fantastic. There are many passionate users who share their knowledge and experiences in different forums and blogs. If you need help, reaching out in a Power BI forum can often lead to a quick response.”

Nathalie Prescott

Senior Recruitment Consultant, Nigel Frank International

Nigel Frank International is the global leader in Microsoft recruitment across 15 worldwide locations. Nathalie is a senior recruitment consultant in their London office.

She used to place professionals for a variety of Microsoft technologies, including Azure and Dynamics but now focuses solely on Power BI.

“Due to the increase in demand for BI professionals, my role now is focused on recruitment only in that landscape.”

These roles can include business analysts, data analysts, Power BI Developers, and Power BI consultants.

“There are many different roles that my clients require. Some want to recruit experienced BI professionals, whereas, with others, experience is not necessary.”

“An interest in Microsoft technologies and an attitude to learn and develop is important.”

Jordan Goldmeier

Anarchy Data

Jordan is a self-titled data anarchist, has worked as an analyst and data scientist for more than 13 years and is highly experienced in many technologies, including Power BI, Excel, VBA, Python, and R scripts.

“Yes, Power BI is definitely something that I would recommend an analyst to learn,” says Jordan.

“Power BI and other Microsoft technologies are fantastic, and professionals with knowledge of them are in large demand, which is growing stronger every year.”

When I asked Jordan on advice for someone seeking a job as an analyst or other role in data, he said.

“I recommend that someone chooses their niche and works on specializing in that area. Learn as much as you can learn but do not get too distracted by the many other tools and technologies out there. You cannot know it all.”

“Also, enhance your skills in critical thinking, data statistics and data visualization. Power BI is the tool, but it is also important to have a talent in problem-solving and knowing how to interpret the data.”

Conclusion

In this chapter, we saw the demand for Power BI skills from the job market and the potential for work from freelance sites, which can be useful to gain experience and develop your talents further.

We also heard from current BI professionals on the importance of Power BI in their work with clients.

In the upcoming chapter, we will learn the different components of Microsoft Power BI and the role that each component plays within the platform.

You will also learn how to sign up for a Power BI account and download and install Power BI Desktop on your machine.

Points to remember

Following are a few key takeaways for why you should learn Power BI.

- Microsoft is identified as the leading provider of analytics and business intelligence platforms, and Power BI is at its core.
- Power BI is growing and becoming a part of many people's work. Power BI puts power business intelligence and analytics in the hands of all users.
- There are hundreds of job vacancies every day that are requiring Power BI skills.
- By learning Power BI, you are at the cutting edge of business intelligence and Microsoft's ever-evolving Power Platform.
- Power BI is fun! You never stop learning, and there are often multiple solutions to a challenge. The community is vibrant and helpful, so you never feel alone.

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



CHAPTER 2

What is Power BI?

Introduction

Power BI is a platform that enables us to easily connect to, model and visualize data in a meaningful and insightful way. These visuals can then be shared with others to consume and collaborate on.

Power BI is a scalable platform for self-service or enterprise business intelligence. It is highly customizable and was built for the Microsoft Office user, making it easy to learn and use.

Power BI is for everyone. There are many job roles that require the use of Power BI, and that role will determine how they use it. Some of them are as follows:

- Someone may only use Power BI to consume and interact with reports and dashboards that are created by others in a team.
- Another user may create simple reports to be consumed only by them. These reports may only connect to a single Excel table. This is also a valuable use of Power BI.
- Or someone may create complex reports that use data from multiple different sources that they then share with their team.

Structure

In this chapter, we will cover the following topics:

- The three components of Power BI
 - Power BI Desktop
 - Power BI Service
 - Power BI Mobile
- Installing Power BI Desktop
- Signing up for a Power BI account

Objectives

After reading this chapter, you will be able to understand the different components of Power BI and the role that each of them plays. You will also be aware of the key elements from each of these components.

You will know how to install Power BI Desktop and understand the benefits of downloading the software from the Microsoft store rather than directly from the site.

You will also sign up for a Power BI account, ready for when we publish our reports to the cloud later in the book.

The three components of Power BI

The power platform ecosystem continues to grow, both in its number of tools and the richness of its functionalities.

In addition to Power BI, the power platform includes tools such as Power Automate, Power Apps, and Power Virtual Agents. Power BI has tremendous interconnectivity with these tools and other business applications such as Teams and Excel.

When we use the term “Power BI”, we are really referring to three different tools—Power BI Desktop, Power BI service, and Power BI Mobile.

Let us explore in further detail the role that each of these tools plays in how we use Power BI.

Power BI desktop

Power BI Desktop is the free authoring tool that is used to create your reports.

These reports can be saved, shared, and opened just like any other file, such as an Excel workbook or a Word document would be. Power BI Desktop files have the extension **.pbix**. A user needs to have Power BI Desktop installed on their machine to open a **.pbix** file.

Although you can share Power BI files in this manner, the intended use of Power BI is to publish the reports to the Power BI service. From the Power BI service, they can easily and securely be shared with other members of your team.

So, Power BI Desktop is used to acquire data from different sources, model the data, create effective reports, and then share those reports with others.

Note: You may hear people talk about Power BI in Excel. This is because Excel contains the same tools for data acquisition and modeling as Power BI Desktop. Your datasets can be published from Excel to Power BI and downloaded from Power BI to Excel. This is fantastic, although it can create confusion about what someone means exactly when they talk about Power BI.

Let us have a short introduction to some of the components of Power BI Desktop.

Power query

The first step to creating your Power BI reports is to get the data you need from various sources such as Excel, SharePoint, and SQL Server databases.

Power Query is the tool in Power BI that enables us to connect to external sources and then transform and shape the data how we need it. It is then loaded to Power BI.

With hundreds of connections, Power Query makes it simple to access data from sources such as Excel, SQL Server, PDF files, and SharePoint. *Figure 2.1* shows the list of connectors when getting data in Power BI.

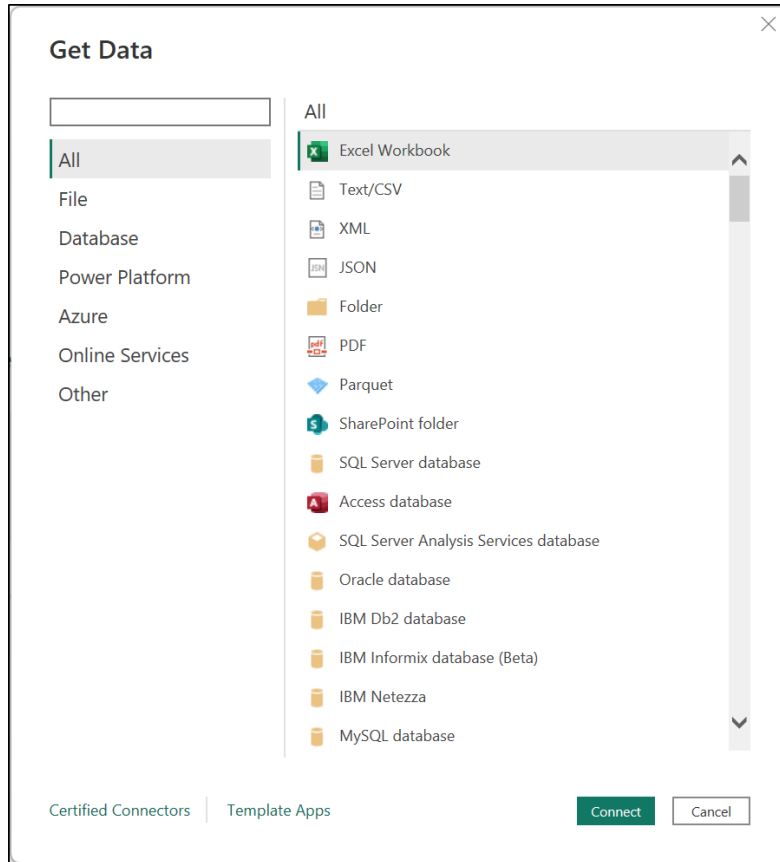


Figure 2.1: Getting data in Power BI

These power queries can be set to automatically refresh to ensure that your data is always up to date.

Model the data

Once the data has been acquired and cleansed using Power Query, you can refine the data model further in preparation for visualizations.

Some of the tasks you can perform when modeling your data include:

- Applying meaningful names to columns, specifying accurate data types, and formatting data appropriately.
- Creating relationships to join the tables into a coherent data structure.
- Hiding any fields that will not be used in the visuals.
- Perform the calculations required for your analyses and reports.

DAX

DAX stands for Data Analysis Expressions and is the formula language of Power BI. It is a rich and powerful language and is used to create tables, calculated columns, and measures for our data model.

Visualizations

The visualizations in Power BI are very impressive. There are many visuals, including column charts, gauges, map charts, and AI visuals. *Figure 2.2* shows the visualizations pane in the Report view of Power BI Desktop:

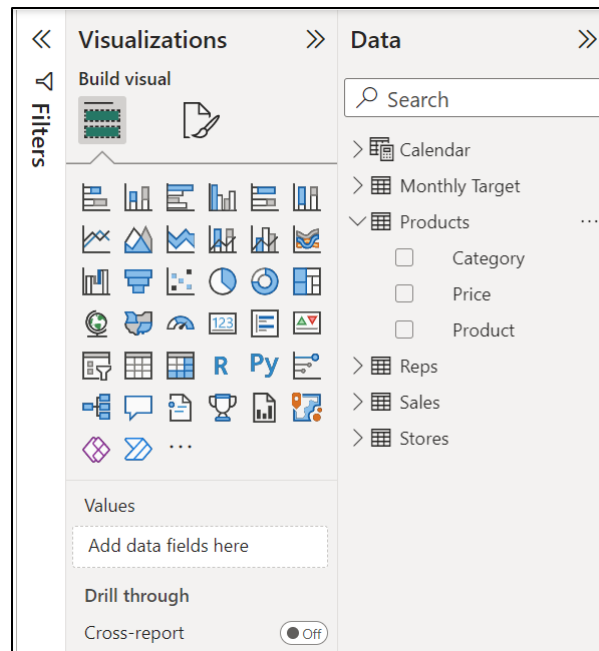


Figure 2.2: Visualizations available in Power BI

These visuals are incredibly easy to use and offer many options to specify exactly how you would like them to look and function. You can also import, or create your own custom visuals, to go beyond those in the standard visualizations pane.

Power BI also provides advanced, yet simple, ways for the visuals to interact with each other. There are many options to filter, highlight, and drill down on visuals at the click of a button.

Those who consume your reports and dashboards will love these dynamic and interactive visuals.

Publish your reports

The completed reports are then published from Power BI Desktop to the Power BI service to share, comment, schedule refreshes, and much more.

To publish the report to the Power BI service, you need a Power BI account. This is free and simple to do. We will explain the process later in this chapter.

Power BI service

The Power BI service is where you share and collaborate with others on your reports and dashboards using a Web browser or mobile app.

There is functionality in the service to create and edit reports, but this is best done using Power BI Desktop.

With the free account, you can view your reports online and perform tasks such as downloading them to PDF and embedding them on a Web page for others to view. However, there is a limitation on the features you can use in the Power BI service for free.

To share and collaborate with others in your organization, you will need a Pro or Premium account. These extra licenses come with a cost and provide greater functionality.

Note: Information regarding licenses and costs is always changing. To view the latest information, go to <https://powerbi.microsoft.com/en-us/pricing/>.

There are a few different components of the Power BI service to know. These will be explained, in detail, in *Chapter 16, Publishing and Sharing your Reports* and *Chapter 17, Datasets, Dashboards and Reports*, later in the book.

Figure 2.3 shows the Power BI service and identifies its different components in the menu down the left. Along the menu at the top are features to export, share, and collaborate on reports. Please refer to the following figure:

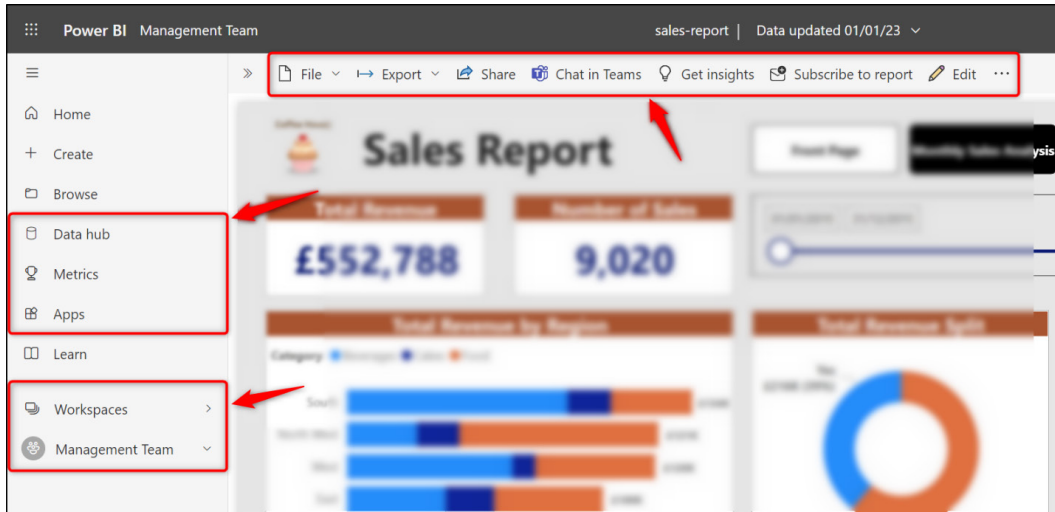


Figure 2.3: The different components and collaboration options in the Power BI service

Let us have a short introduction to some of the components of the Power BI service.

Workspaces

A workspace is a location to store your reports, dashboards, and datasets that can be shared with your team for collaboration.

With your Power BI account, you get a workspace named “**My workspace**”. This is your personal workspace. *Figure 2.4* shows some datasets, reports, and dashboards in my workspace.

You can create other workspaces and add the reports, dashboards, and datasets that you want to include to that workspace before sharing it with others in your team.

You can assign roles to the different users that a workspace is shared with. Some users may have view-only permission, whereas others may be allowed to edit and share content from the workspace.

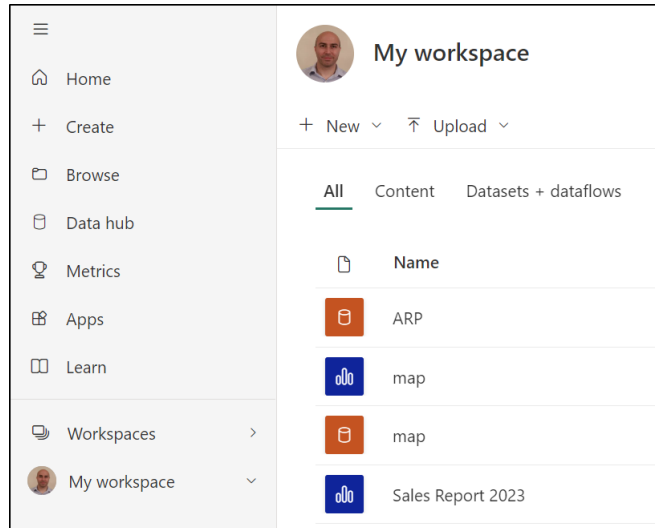


Figure 2.4: Dataset, reports, and dashboards in a workspace

Datasets

A dataset is a collection of data. This could be a single Excel workbook or a model built from multiple different data sources.

When you publish your Power BI report from the Desktop to the service, the dataset is uploaded with it.

A dataset can be used as the source for many Power BI reports and can be included in one or more workspaces.

Reports

A report is created in Power BI Desktop from a single dataset and published to the service. It can be as simple as one or two visuals, or it can contain multiple pages, each full of different visuals.

A report can be connected to only one dataset and associated with only one workspace.

Dashboards

This is a collection of visuals on a single page and can be shared with others. It presents the highlights. The best visuals for the story you are telling.

While a report is created from a single dataset, dashboards can include visuals from multiple different datasets and reports. Like reports, however, dashboards can only be associated with one workspace.

Dashboards are only available in the Power BI service.

Apps

Apps provide an easy way to share a collection of related reports and dashboards.

A user can create an app and add all the reports and dashboards they would like to share with others. On receiving access to an app, the viewer can then easily consume the content all together in one place.

Power BI mobile

The Power BI Mobile app is free and is available on Windows, Android, and iOS devices.

Power BI Mobile enables users to view and collaborate on business data from anywhere. It provides you with easy access to all your workspaces, dashboards, and reports.

You can also set up data alerts and receive push notifications about changes to your data.

Installing power BI desktop

Power BI desktop is the free authoring tool for your Power BI datasets and reports.

There are the following two ways to download and install Power BI Desktop:

1. Download from the Microsoft Store
2. Direct download from microsoft.com

Updates to Power BI Desktop are released regularly, and to receive the automatic updates—you need to download from the Microsoft Store.

If you directly download Power BI Desktop from the website, you will need to manually check and install the updates.

In an organization, access to downloading software is often tightly controlled by the IT department. This can restrict how and when installations occur. Because of this, it is not unusual for different organizations, or even colleagues in the same organization, to have a slightly different version of Power BI Desktop.

Let us look at how to install Power BI Desktop using both methods.

Downloading from the Microsoft store

To download Power BI Desktop, follow the following steps:

1. Open a Web browser and navigate to the following URL—<https://powerbi.microsoft.com/en-us/desktop/>
2. Click on the **Download free** button (*figure 2.5*):

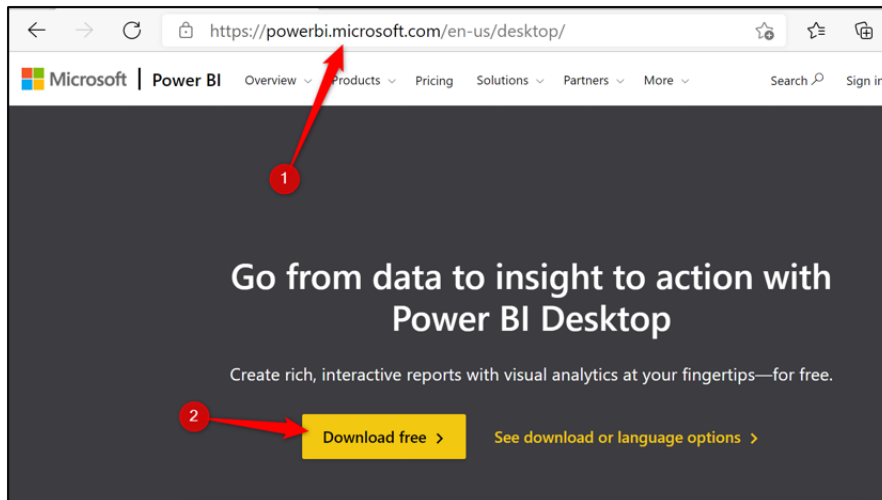


Figure 2.5: Link to the Microsoft store to download Power BI desktop

3. You may be prompted to open the Microsoft store application (*figure 2.6*). Click on **Open**.

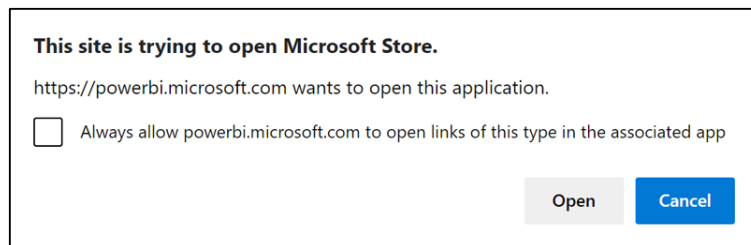


Figure 2.6: Prompt to open the Microsoft Store

4. The Microsoft Store opens and takes you directly to the Power BI Desktop download page (*figure 2.7*). Click on **Install**.

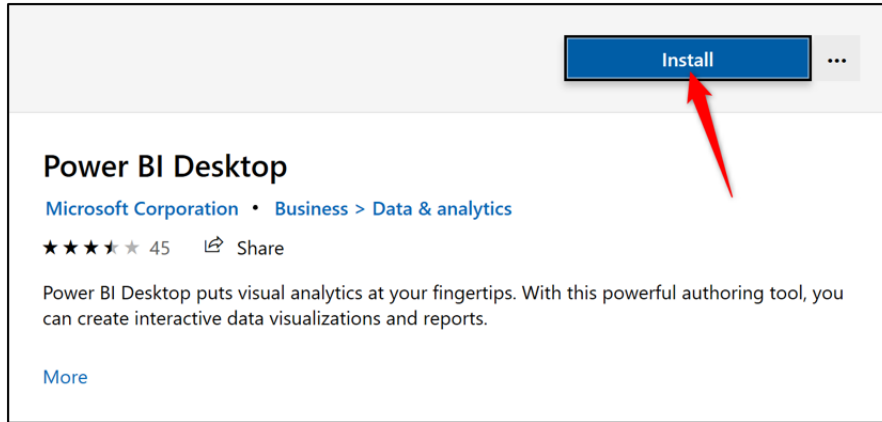


Figure 2.7: Install the Power BI desktop app from the Microsoft store

Tip: You can also open the Microsoft Store app and search for Power BI Desktop instead of navigating via the website.

That is all you need to do. Power BI Desktop will automatically update, so you can be sure that you always have the latest version of the software.

Downloading directly from Microsoft.com

Following are the steps to download Power BI Desktop directly from the website:

1. Open a Web browser and navigate to the following URL—<https://powerbi.microsoft.com/en-us/desktop/>
2. Click on the **See download or language options** link (figure 2.8):

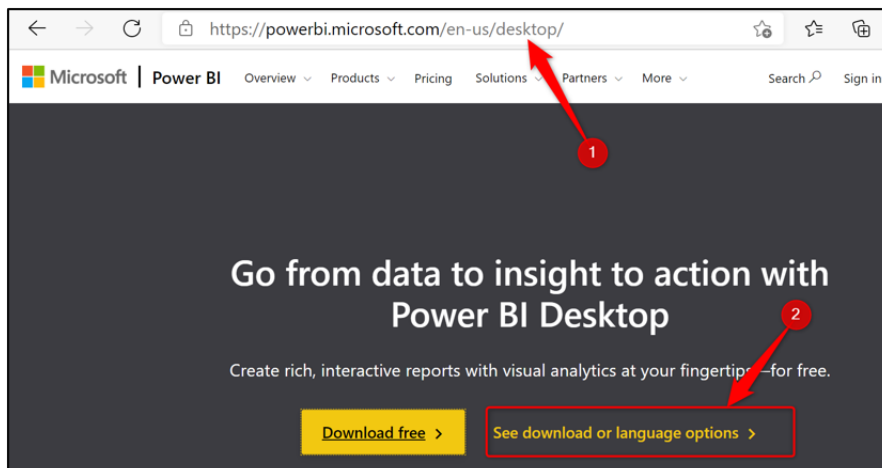


Figure 2.8: Link for the direct download from the website

- The next page presents information about Power BI Desktop, its file size, and the system requirements (*figure 2.9*). Select your language and click on **Download**:

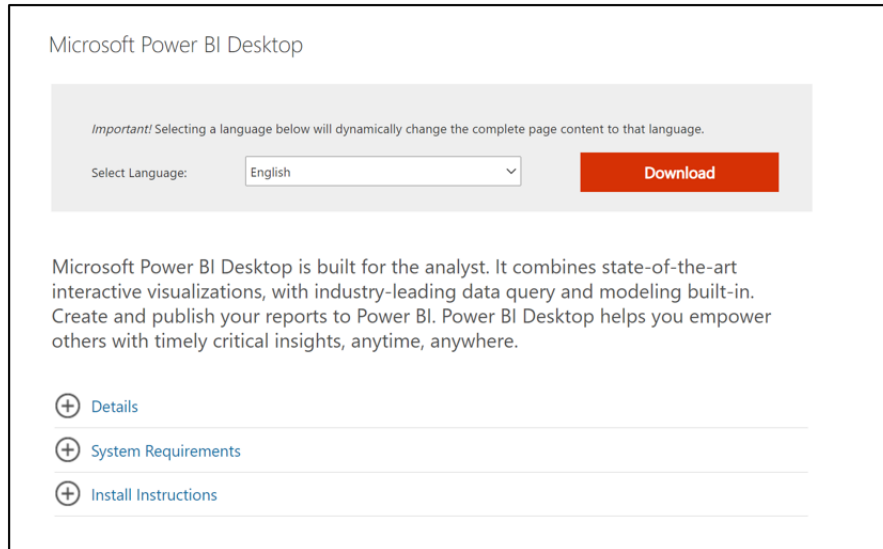


Figure 2.9: Information about the software and system requirements

- The next step is to select either the 32-bit or 64-bit version of the software (*figure 2.10*). Installing the 64-bit version will provide maximum performance benefits when analyzing data. Click on **Next**.

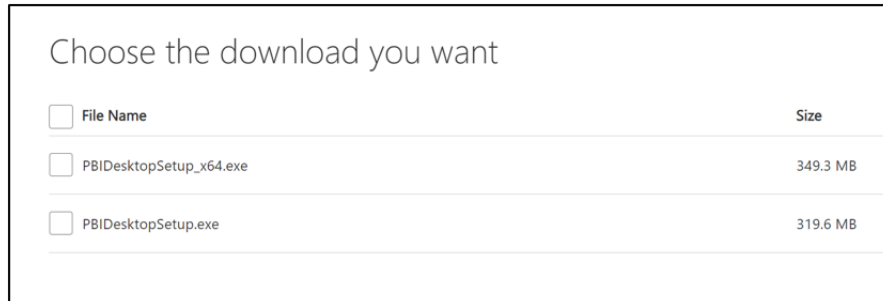


Figure 2.10: Choose between the 32-bit and 64-bit downloads

- Power BI desktop begins downloading (*figure 2.11*):

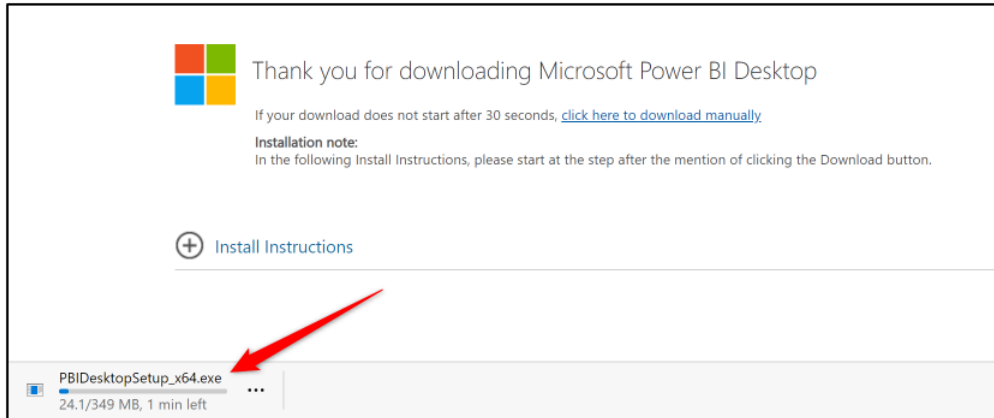


Figure 2.11: Power BI desktop is downloading

- When the download has been completed, open the file. This begins with the Setup Wizard (*figure 2.12*). Select your language and click on **Next**.

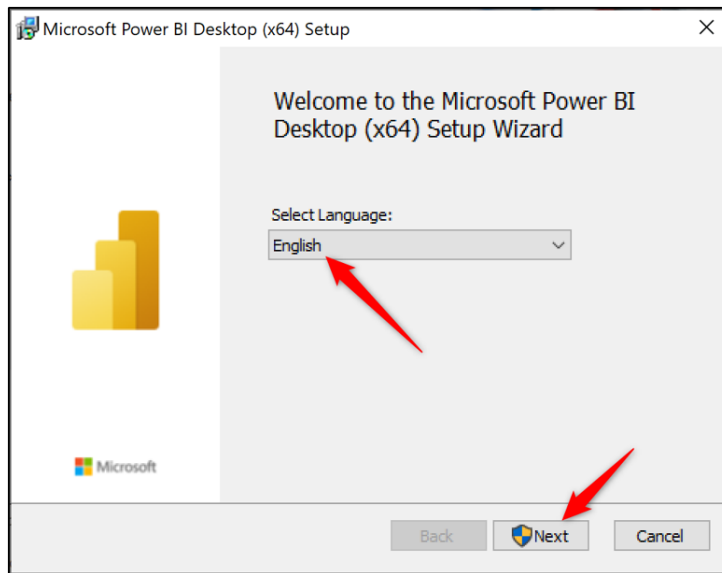


Figure 2.12: Select your language and begin the Setup Wizard

- Follow the steps of the wizard to confirm the license terms, file path of the software, and complete the installation by clicking on Finish.

If it is your responsibility, keep an eye out for updates to the software. You are usually notified by a message on the Status Bar in the bottom-right corner of the window.

Perform the updates to ensure that you have the latest version and get the most out of Power BI Desktop.

Creating a Power BI account

You do not need a Power BI account to get started learning Power BI Desktop. However, you will need a Power BI account to publish reports and use the Power BI service.

There are two ways to obtain a Power BI account:

- Sign up for a free, pro, or premium account as an individual. This requires a work or school email account. E-mail accounts such as outlook.com and gmail.com are not supported.
- You may be assigned a license from an administrator at your organization.

Follow the following steps to sign up for a Power BI account:

1. Navigate to the following URL—powerbi.microsoft.com
2. Click either on the Start free or Try free link (*figure 2.13*).

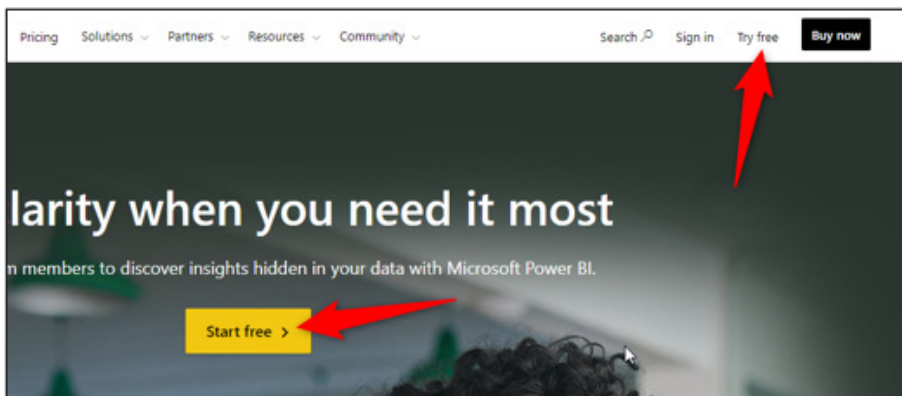


Figure 2.13: Try a Power BI account for free

3. You may be taken to another page, where you need to click on the **Try Power BI for free** or **Try free** link again (*figure 2.14*).



Figure 2.14: Click the try Power BI for free link to begin to sign up

4. You will probably be recognized as already using a Microsoft account, at which point you can sign in (*figure 2.15*).

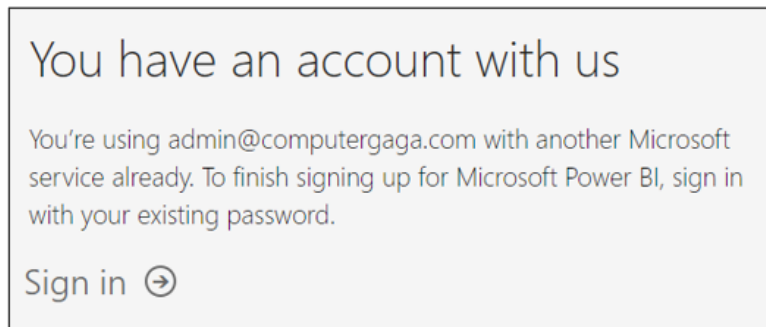


Figure 2.15: Prompt to sign in with my Microsoft account

5. If you are not recognized, you will need to enter a valid e-mail address to sign up. You will be requested to review and accept the terms and conditions.

It may take a short while to set up your account. You should be taken to the Power BI service in a Web browser on completion.

Conclusion

In this chapter, we explained the different components of Power BI and their purpose. We installed Power BI Desktop and created a Power BI account in preparation for the rest of this book.

In the upcoming chapter, we will take our first peek at Power BI Desktop and learn about the three different views.

We will start a new PBI file and explore useful options and settings that you might consider changing to tailor your experience in Power BI Desktop.

Questions

Here are some questions to test what you have learnt in this chapter:

1. Which of the following are parts of Power BI Service?
 - a. Dashboards
 - b. Reports
 - c. Workspaces
 - d. All of the above.

2. You need a Power BI account to use Power BI Desktop.
 - a. True
 - b. False
3. What does DAX stand for?
4. What role does Power Query play within Power BI Desktop?
 - a. Insert visualizations such as column charts and tables.
 - b. Create measures for our data model.
 - c. Get, transform, and shape data ready for analysis.
 - d. Provides a method of sharing reports and dashboards.

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



CHAPTER 3

Getting Started with Power BI Desktop

Introduction

Power BI Desktop is a very powerful tool, and you may be wondering where to begin when you first open the software. For this reason, it is nice to have a sneak peek at Power BI Desktop before we use it for our first report.

Structure

In this chapter, we will cover the following topics:

- Orientation of the Power BI Desktop screen and the different views.
- Explore important options and settings.
- Start and save a new Power BI file.

Objectives

After reading this chapter, you will understand the different views of Power BI desktop and their roles in creating reports.

You will be aware of important options and settings that you can change to tailor your PBI Desktop experience. The default settings are not to everyone's preference.

As you use the software more, you will learn the functionality that you like and customize the settings for yourself.

You will start and save a new Power BI file. This is the file that we will continue within the upcoming chapter when we create our first Power BI report.

A first look at Power BI Desktop

Power BI desktop was built with the office user in mind, so when you open it for the first time, aspects of it will have that familiar Microsoft Office feel.

This is excellent when you are learning for the first time. It gives you the confidence to explore further into the more complicated aspects.

Let us jump in and explore the different parts of Power BI Desktop.

Welcome screen

On opening Power BI Desktop, you will see the start screen (*figure 3.1*).

This provides a nice welcome to the tool and gives you fast access to recently used files, information on recent updates to the software, and links to the forums for help from the active Power BI community.

You can prevent this screen from appearing every time you open Power BI Desktop by unchecking the **Show this screen on startup** box at the bottom of the screen:

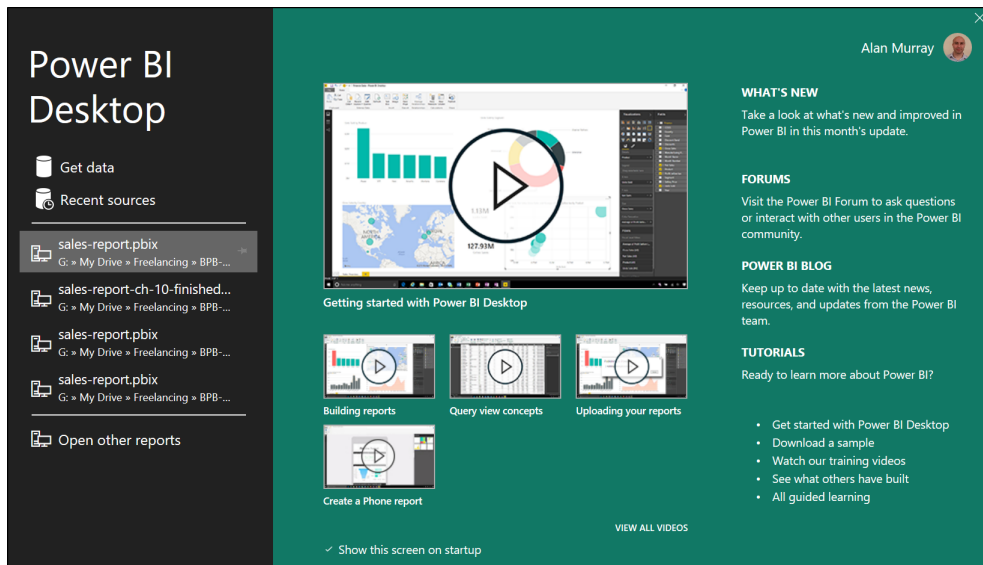


Figure 3.1: The start screen on opening the Power BI desktop

Click on the **X** in the corner to close the getting started screen.

There are three views in Power BI Desktop—Report view, Data view, and Model view. You can switch between the views using the buttons on the menu on the left of the window (*figure 3.2*):

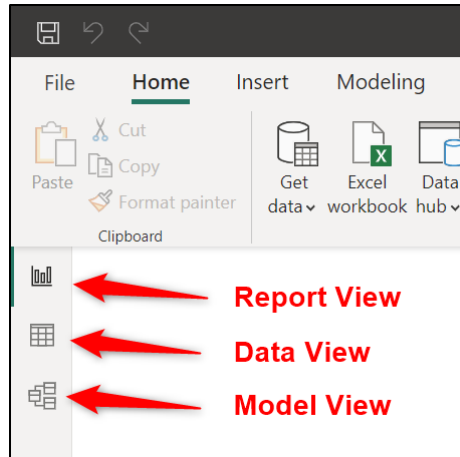


Figure 3.2: The three views of the Power BI desktop

The Report view

The default view in the Power BI desktop is the report view. This is where we will design our reports and build our visuals.

The **Home** tab of the Ribbon in each view gives a good indication of its purpose.

In the Report view, you can see the different phases of a typical Power BI report on the Home tab (*figure 3.3*):

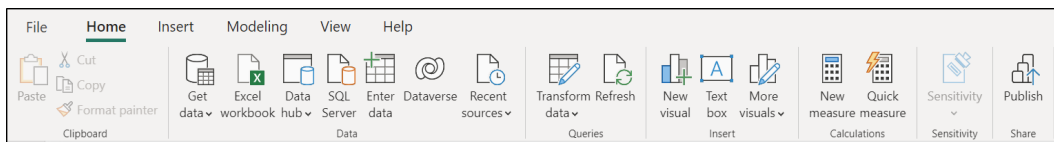


Figure 3.3: The Home tab of the Ribbon in the Report view

There are five distinct groups of buttons for getting data, transforming the data, inserting visuals, creating measures, and then publishing the report. These are tasks that are commonly performed in this view.

Note: Power BI Desktop is updated regularly, and the images of buttons on the Ribbon, and other screenshots, are likely to be different at the time of reading this book.

The canvas is the main element of the Report view. This blank canvas is where we will insert our visuals and design our reports.

Figure 3.4 shows a collection of visuals, including a Slicer, two cards, a stacked bar chart, a doughnut chart, and a line chart placed on the canvas. There is also an image, a text box, and navigation buttons.

Below the canvas are the page tabs. A Power BI report can contain multiple pages. By default, the page is aligned to the top of the canvas:



Figure 3.4: The canvas and page tabs of the Report view

To the right of the canvas are three panes—**Filters**, **Visualizations**, and **Data** (figure 3.5).

These panes can be expanded and collapsed to ensure that you have room on your screen to design your reports. The **Filters** pane has collapsed in the image.

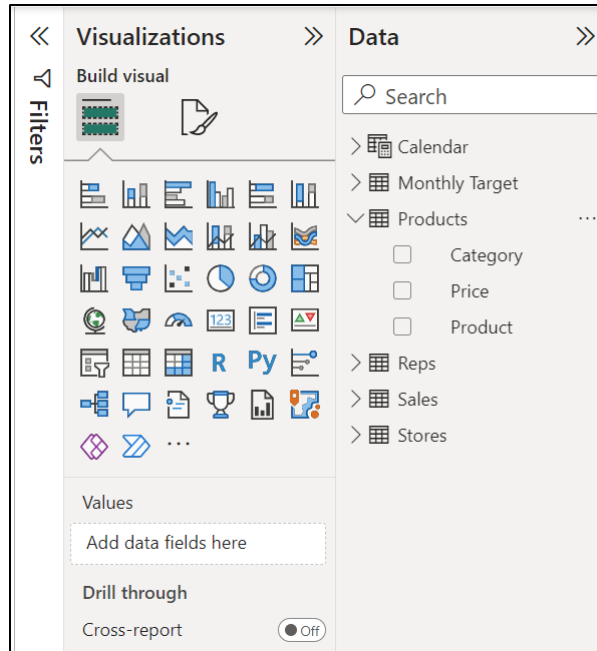


Figure 3.5: The filters, visualizations, and data panes

The **Visualizations** and **Data** panes are essential for working in the Report view.

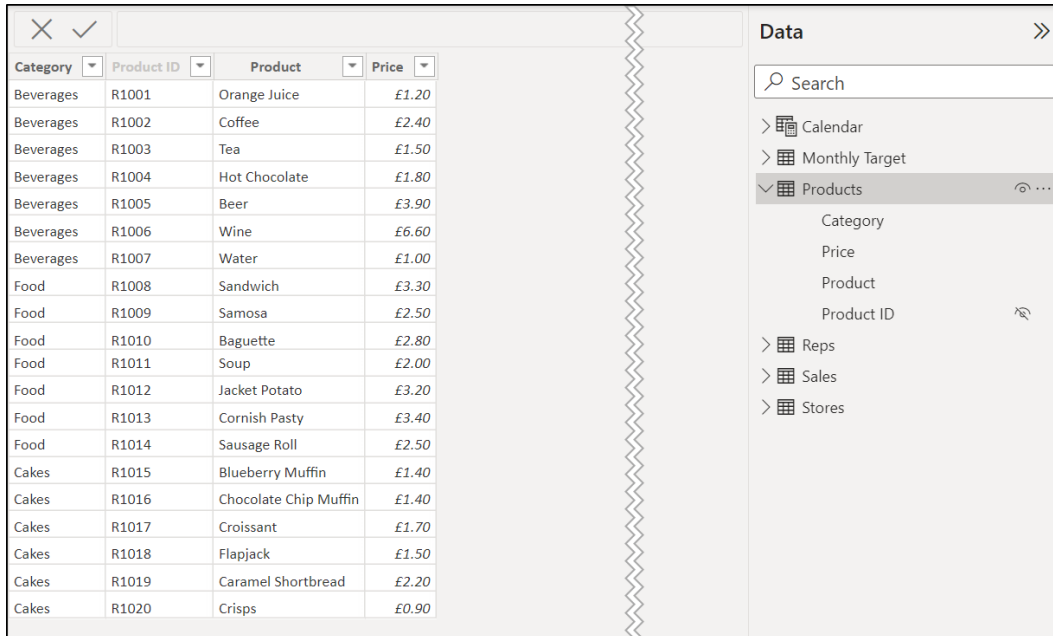
You can insert a visual onto a page by either clicking or dragging the visual from the **Visualizations** pane. You can then drag the fields of your model from the Data pane into the areas of the visual to build it. This is very similar to working with PivotTables in Excel.

The data pane shown in *figure 3.5* contains six tables. The **Products** table has been expanded to show its fields.

The Data view

The data view is primarily used for performing tasks on columns/fields, such as formatting, changing data types, and specifying the default summarization. It is also useful for creating calculated tables and columns.

As its name implies, this view shows you the data of the selected table. In *figure 3.6*, the **Products** table has been selected in the data pane:



Category	Product ID	Product	Price
Beverages	R1001	Orange Juice	£1.20
Beverages	R1002	Coffee	£2.40
Beverages	R1003	Tea	£1.50
Beverages	R1004	Hot Chocolate	£1.80
Beverages	R1005	Beer	£3.90
Beverages	R1006	Wine	£6.60
Beverages	R1007	Water	£1.00
Food	R1008	Sandwich	£3.30
Food	R1009	Samosa	£2.50
Food	R1010	Baguette	£2.80
Food	R1011	Soup	£2.00
Food	R1012	Jacket Potato	£3.20
Food	R1013	Cornish Pasty	£3.40
Food	R1014	Sausage Roll	£2.50
Cakes	R1015	Blueberry Muffin	£1.40
Cakes	R1016	Chocolate Chip Muffin	£1.40
Cakes	R1017	Croissant	£1.70
Cakes	R1018	Flapjack	£1.50
Cakes	R1019	Caramel Shortbread	£2.20
Cakes	R1020	Crisps	£0.90

The right-hand pane shows the 'Data' view with a search bar and a list of tables: Calendar, Monthly Target, Products (selected), Repls, Sales, and Stores. Under 'Products', the columns Category, Price, Product, and Product ID are visible.

Figure 3.6: Products table selected in data view

Because you can see the data from the selected table, it makes it the ideal view for tasks such as formatting columns and creating calculated columns because you can see the results of your actions.

Figure 3.7 shows the **Home** tab of the Ribbon in the Data view. When compared to the **Home** tab of the Report view, you may notice that there are no buttons for inserting visuals. This cannot be done in the Data view.

Those buttons have been replaced with a Manage Relationships button. And in the Calculations group, there are buttons for a New Column and a New Table. These tasks are commonly performed in the Data view:

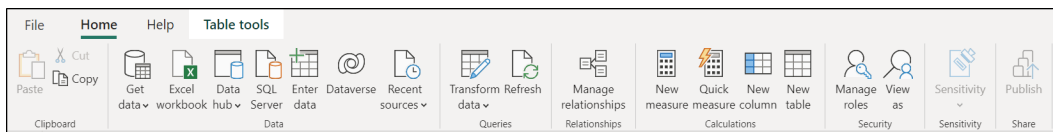


Figure 3.7: The Home tab of the Ribbon in the data view

The **Table tools** tab of the Ribbon provides functionality such as defining the data types and format of the columns.

The Model view

The Model view is primarily used to create and view the relationships between the tables of the data model. The model is shown as a diagram making it easier to understand and work with.

Figure 3.8 shows a data model with five tables. The relationships between these tables are presented visually in the model view:

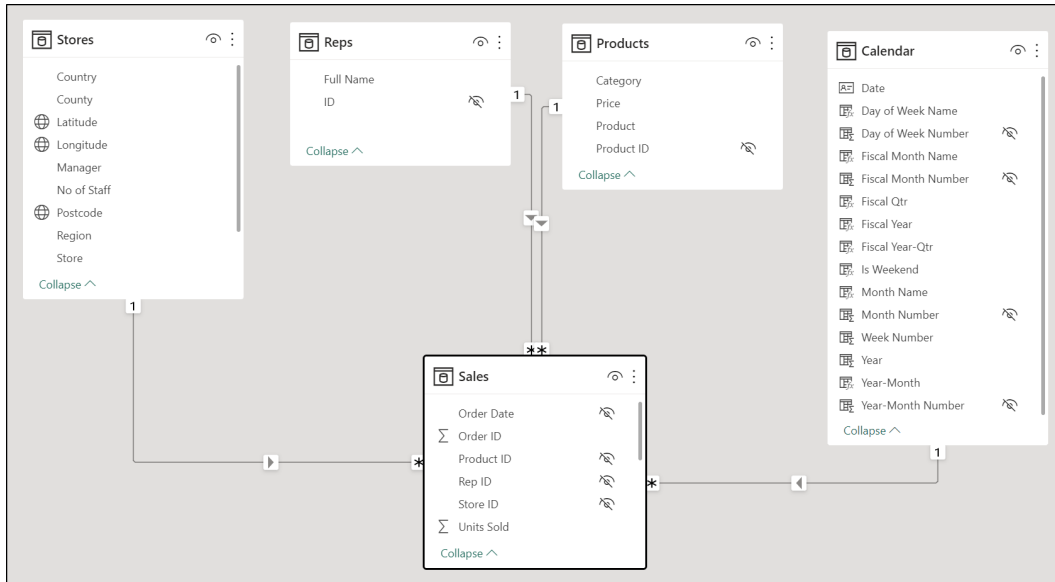


Figure 3.8: Relationships between tables shown in the Model view

When you look at the **Home** tab of the Ribbon (figure 3.9), you may notice that there are no buttons for inserting visuals. These tasks are to be performed in the Report views only:

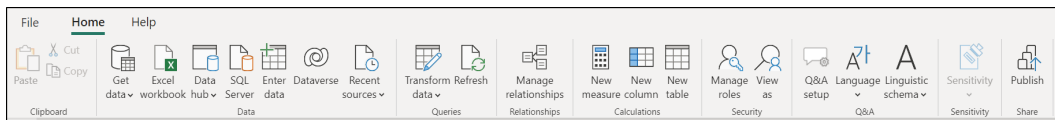


Figure 3.9: The Home tab of the Ribbon in the Model view

The **Model** view does not lack functionality, though. You can perform calculations, format fields, create folders to organize fields, and more.

Figure 3.10 shows the **Properties** and **Data** panes that are displayed on the right side of the window. The Properties pane shows the settings related to the **Sales** table, as that table has been selected in the Data pane.

It contains two categories—**General** and **Advanced**. Both categories are collapsed. There are some useful properties available from here, such as changing the storage mode of a table:

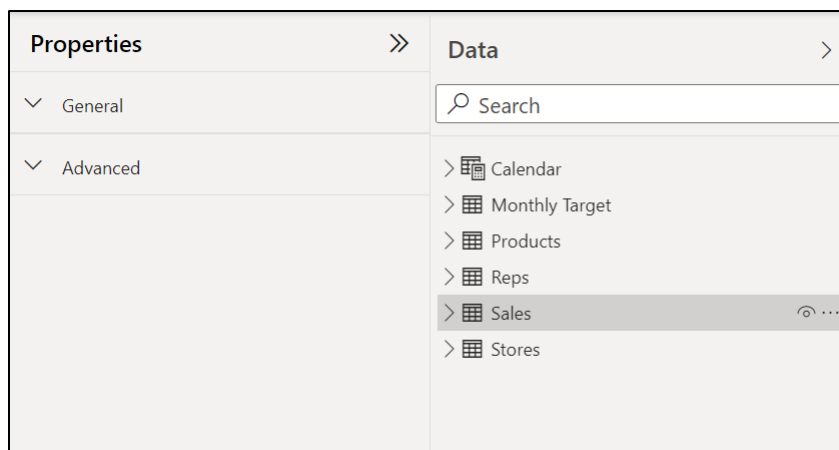


Figure 3.10: Properties and Data panes of the Model view

The **Properties** pane can also be used to change settings for the fields of a table. From the **Properties** pane, you can achieve many useful tasks, such as changing a field data type and hiding the field in the Report view.

Exploring useful options and settings

There are many options and settings in Power BI Desktop that you can change to customize the tool to your needs.

Some of these options are commonly used by Power BI users. We will focus on these options and explain why you may or may not want to change them.

Some of these options are global and will affect future reports you create on the Power BI desktop. Others you can change to only affect the file in use.

To open the Power BI options, click on **File | Options and Settings | Options** (figure 3.11):

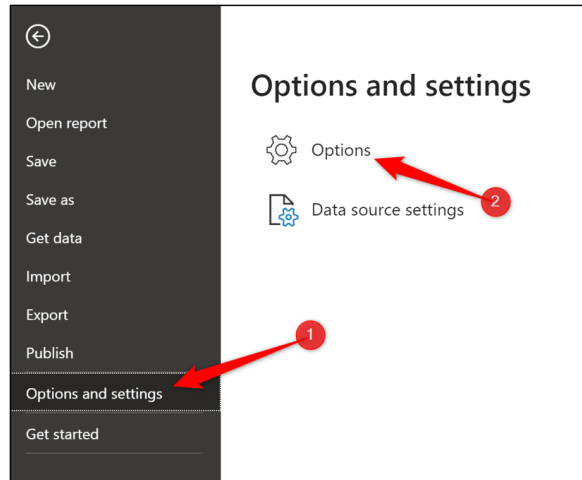


Figure 3.11: Opening the options window in Power BI Desktop

Now, let us explore some useful options from different categories in the **Options** window.

We will not change any of the options at this stage, as it is better to learn Power BI desktop with its default behaviors. For now, it is useful just to know how these options can be changed to personalize your experience.

Note: The reasons for changing some of these settings may not be clear until you see Power BI in action during the book. They are covered in this chapter because they are better to change before you begin with the report, and it also provides an easy reference point to come back to.

Data load

Figure 3.12 shows the global data load options in Power BI Desktop. So, changing these would affect future reports. There is also a data load category for the current file only. Useful for changing settings for specific reports only.

Some common options in this category include the following:

- **Type Detection:** The default setting is for Power Query to automatically detect the data type for columns. This happens early in the query and repeatedly if a task creates a new column in the data. This can slow query performance and produce errors.

This setting can be switched to **Never detect column types and headers for unstructured sources**. The data types can then be handled manually by us later in the query.

- **Background Data:** When working with large data models, refresh times can lag. In these circumstances, some users will set this to **Never allow data previews to download in the background**. This can significantly improve query refresh time.
- **Time intelligence:** By default, Power BI Desktop creates a hidden date table for all columns in the lookup tables of a model that are the date data type. It also establishes a relationship between the date column of the hidden date table and the lookup tables date column.

This is done to enable the use of time intelligence calculations by the user. However, this can cause multiple hidden date tables to be created.

It is more efficient to disable the Auto date/time for new files option and create your own date table with all the columns you need for time intelligence work. We will see how to do this in *Chapter 8, Creating a Date Table* of this book.

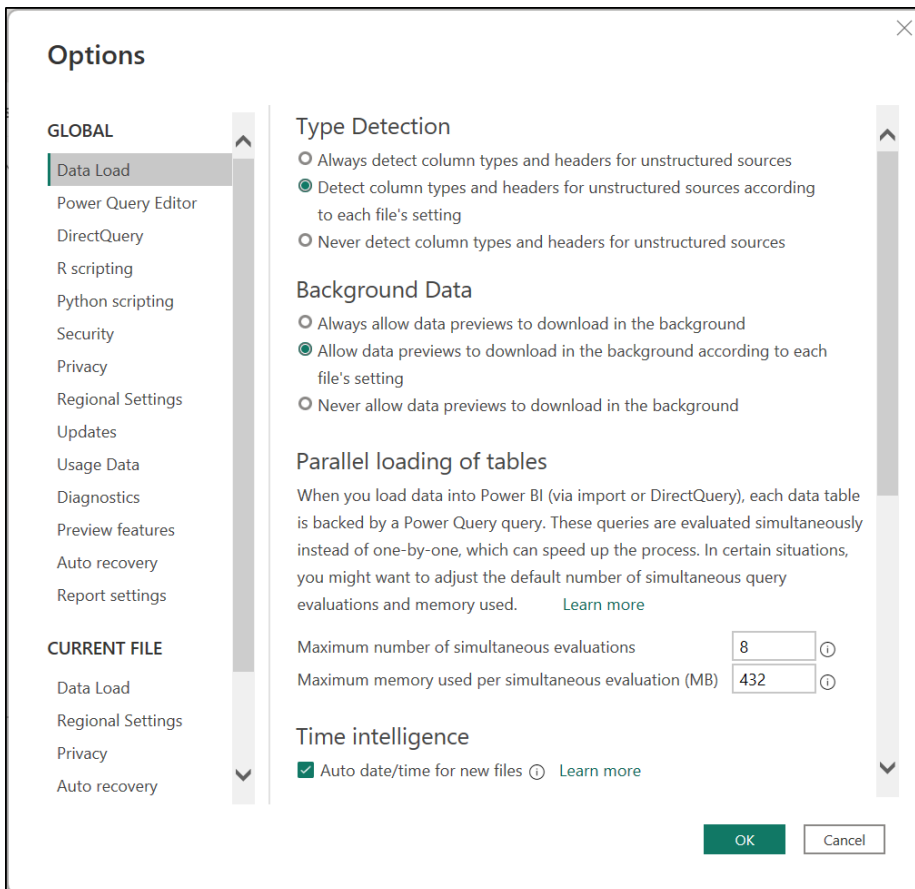


Figure 3.12: The global data load options in the Power BI desktop

Preview features

The preview features are an interesting area of the Power BI options. From here, you can enable new features that are available to preview in your release.

Figure 3.13 shows the preview features currently available in my version of Power BI Desktop. There are links to learn more about these new features.

At the time that you read this book, these features may have been released fully in Power BI or even removed:

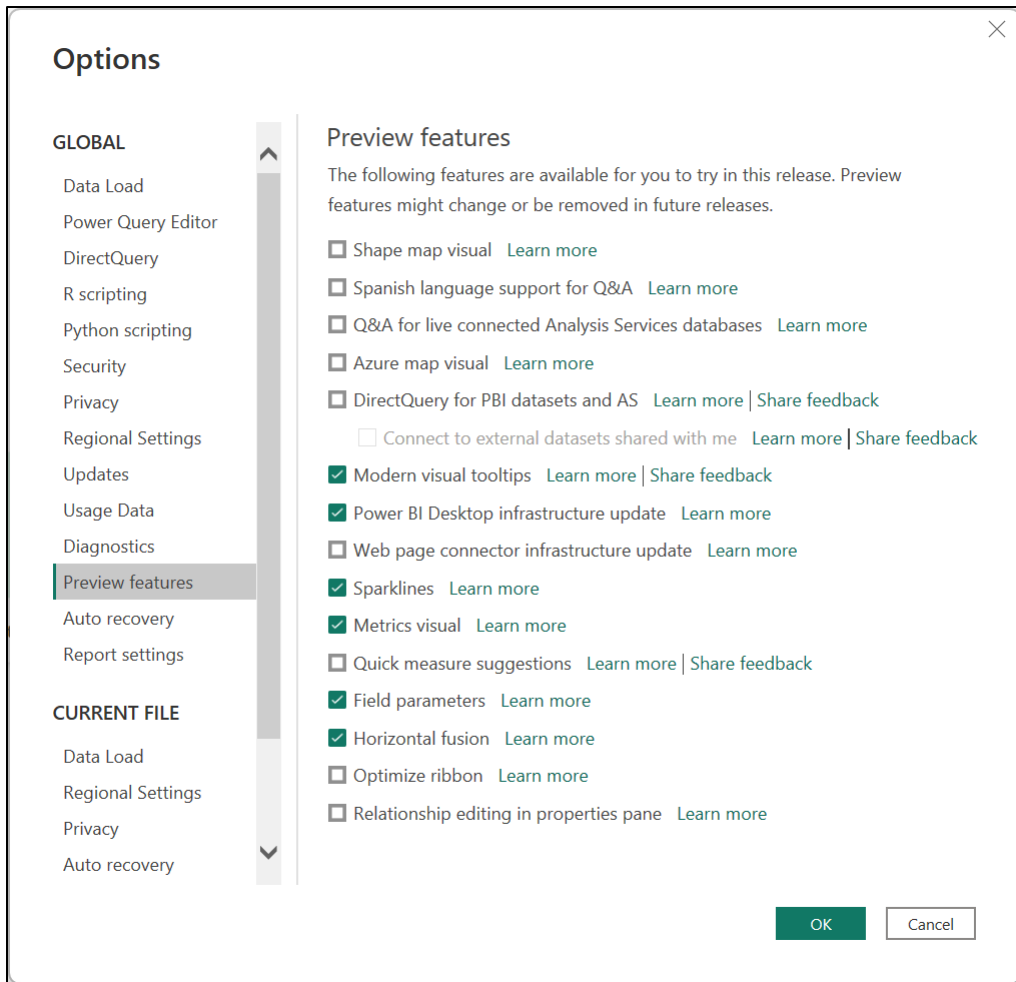


Figure 3.13: Preview features available to try on the Power BI desktop

Regional settings

By default, the regional settings for Power BI Desktop are taken from your computer settings. This is used to interpret date, time, and number fields on import.

This can be changed globally and for specific files. *Figure 3.14* shows the regional settings for the current file only.

If you are working with data sourced from a region different to your own local settings, it may make sense to change this setting. Power BI will then be better prepared for the format of the date, time, and number fields. Please refer to the following figure:

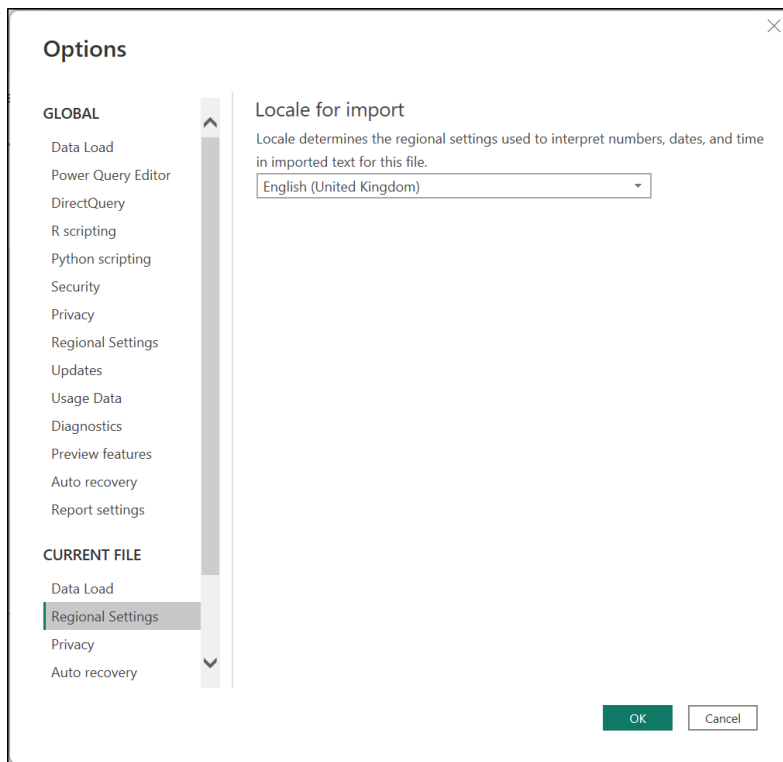


Figure 3.14: Regional settings for the current file

Query reduction

Power BI reports are very interactive, “straight out of the gate”. This is exciting and very useful.

However, it is not always what you want, and this consistent querying of your data every time you engage with the report can be time-consuming.

Some useful options in the *Query Reduction* category (figure 3.15) include the following:

- **Disabling cross highlighting/filtering by default:** Every visual on the report page interacts with the other visuals by default. When you click on a table cell or chart data point, it highlights or filters the other visuals on the page. You can edit or remove this interaction for each visual individually. However, if you are planning to remove this for the majority or all the visuals, it makes sense to disable this setting in the Power BI options. This will save you a lot of time later.
- **Slicers:** The default setting is to Instantly apply slicer changes. There is the option to Add an Apply button to each slicer to apply changes when you are ready.

This setting is useful if you have multiple Slicers. You can click on Apply when ready instead of the data being queried each time you change a Slicer value:

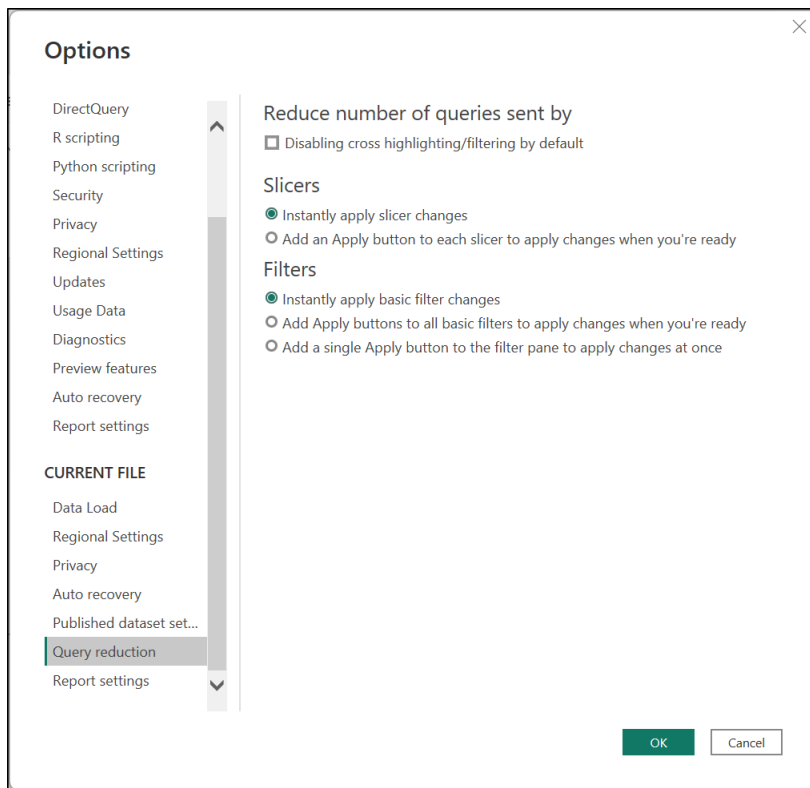


Figure 3.15: Query reduction settings for the current file

Report settings

Figure 3.16 shows the Report settings category for the current file.

There is also a Report settings category for global changes; however, those options are very different to the ones for the current file.

Useful settings in this category include the following:

- **Visual options:** The default interaction between visuals on a page is set to cross-highlighting. In my experience, this is rarely the preference for users. This is notable in visuals such as column charts and pie charts.

You can change the default interaction to cross-filtering by selecting the **Change default visual interaction from cross highlighting to cross filtering** option.

- **Export data:** It is possible for users to export data from the Power BI service by default. This could cause problems, and it may be better to keep that single source and prevent the chances of having multiple versions of the same data.

To prevent users from exporting the data, change the option to **Don't allow end users to export any data from the service or Report Server**.

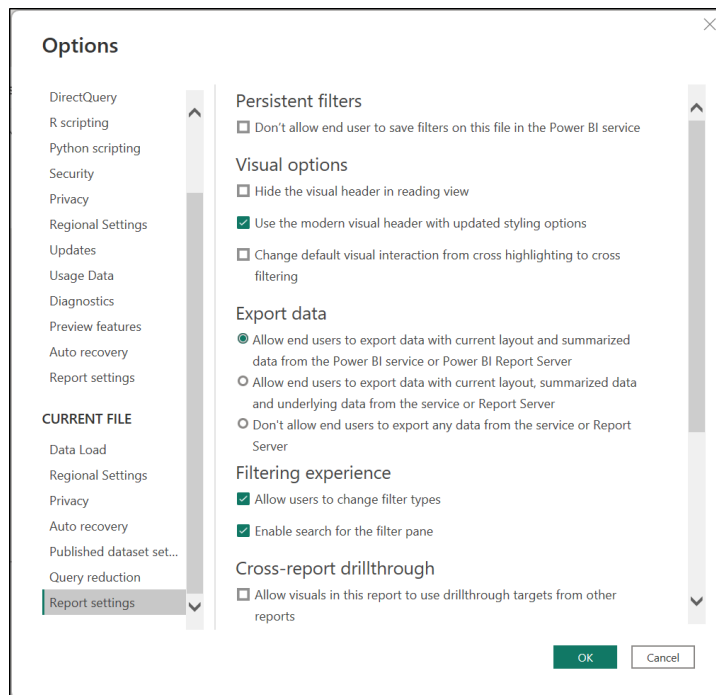


Figure 3.16: Report settings for the current file

Saving a Power BI Desktop file

We will continue with this file in the upcoming chapter when we create our first Power BI report. So, let us save what we have done so far.

Saving a Power BI Desktop file is no different to saving other MS Office documents, but in terms of comprehension, it is important not to skip this step.

1. Click on **File | Save As**
2. Navigate to the location to save the file and enter a meaningful file name. Click on **Save**.

Figure 3.17 shows the **Save As** window in Power BI Desktop. I have named the file **top-films-analysis** as our first example will work with film data.

You may notice that the file type is a Power BI file and has a **.pbix** extension:

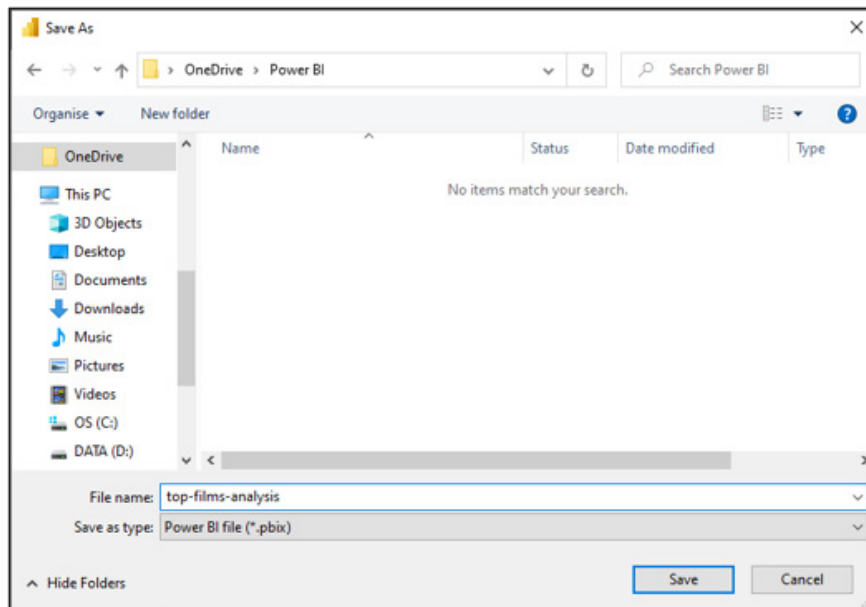


Figure 3.17: Saving a report in Power BI Desktop

The file is saved. We will continue to work on this file in the upcoming chapter.

Conclusion

In this chapter, we explained the three different views of Power BI Desktop and how each of them is best used. We explored useful options and settings to customize the PBI desktop to your needs. And we started and saved a new Power BI file ready for our first report.

In the upcoming chapter, we will create our first Power BI report. It will be a simple report that introduces us to a few of the different elements of the Power BI desktop and the process of creating reports.

We will get data from the Web, perform some data transformations, and load it to Power BI. Then, we will create a couple of visualizations to gain insights into the data.

Questions

Here are some questions to test what you have learnt in this chapter.

1. In which view would you create the visuals of your report?
 - a. Model view
 - b. Report view
 - c. Data view
 - d. All of the above.
2. Which view is best for viewing the relationships between the tables of your model?
 - a. Model view
 - b. Report view
 - c. Data view
3. What is the extension of a Power BI file?
4. By default, PBI Desktop automatically creates a hidden date table for all columns of a date data type.
 - a. True
 - b. False

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



CHAPTER 4

Creating a Simple Power BI Report

Introduction

It is time to create our first **Power BI (PBI)** report. This will be a nice introduction to parts of the process in creating a Power BI report.

This will be a simple example that connects to a single table of data, demonstrates the magic of Power Query, and introduces you to a couple of PBI visuals. This will be exciting, so let us dive in.

Structure

In this chapter, we will cover the following topics:

- Getting data from the Web
- Performing data transformations
- Using Column From Examples
- Creating a column chart visual and changing the order of the data series
- Creating a table visual

Objectives

After reading this chapter, you will be able to connect to structured data on a Web page, perform useful data transformations and load the data into Power BI.

You will know how to edit the existing queries to make improvements or fixes later.

You will be able to create column chart and table visuals and make simple modifications to improve their effectiveness.

Getting data from the Web

The first step to creating our report is to get the data that we need. In this report, we will be using the top 250 film data from imdb.com. Let us perform the following steps:

1. Open the **top-films-analysis.pbix** file that we saved in the previous chapter.
2. Click on **Home | Get data | Web** (refer to *figure 4.1*):

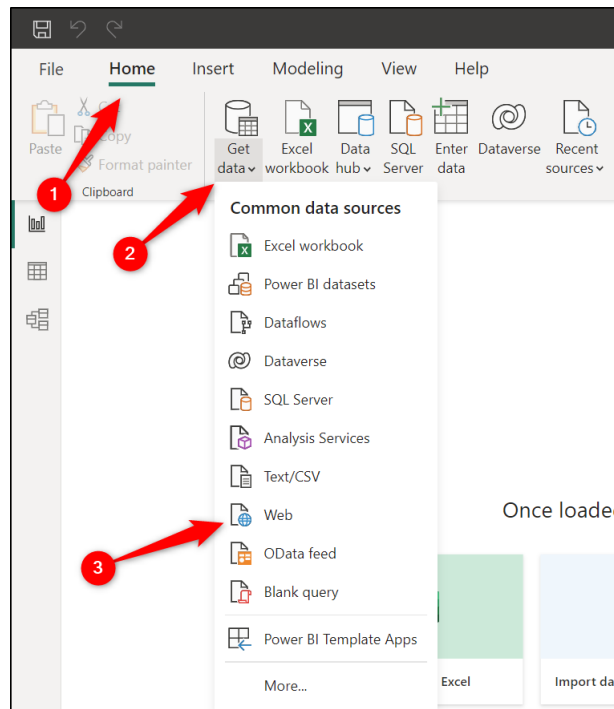


Figure 4.1: Getting data from the Web

A list of common data sources is shown from the **Get data** button. The **More** button gives access to many more sources that Power BI provides connectors for.

1. In the **From Web** window, type or paste the following URL and click on **OK** (refer to *figure 4.2*).

https://www.imdb.com/chart/top/?ref_=nv_mv_250

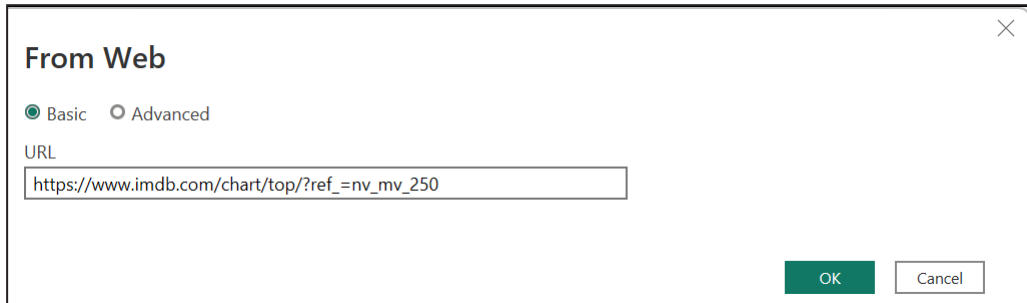


Figure 4.2: The From Web window to enter the URL

Note: You can get this URL by navigating to imdb.com in a browser. Then, navigate to the Top Rated Movies page using the menu provided at the top of the page. This information is correct at the time of writing this book. Please bear in mind that the website can change.

This is just a basic Web connection. The **Advanced** option provides the ability to build a URL from different parts.

You may be prompted for an authentication method (refer to *figure 4.3*). This happens when using a URL for a Web connection for the first time.

2. Select the **Anonymous** method, as no credentials are required to access this content. Leave the level as **https://www.imdb.com/**. Click on **Connect**.

These credentials are now stored. So, we will not be prompted again if we connect to more data from this URL.

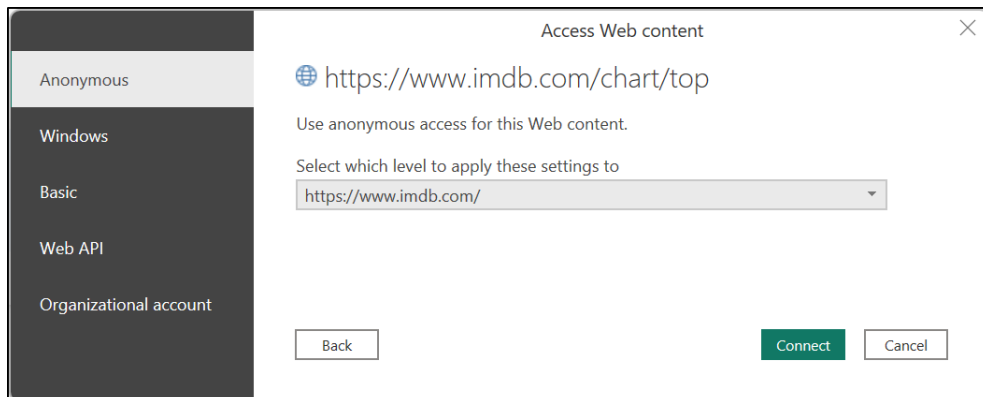


Figure 4.3: Window prompting for an authentication method for the URL

The **Navigator** window is shown listing the tables that were identified on the Web page, as shown in *figure 4.4*.

The two tables shown are the same data. An HTML table and a suggested table are provided. Either of these would work. We will use Table 1 for this example.

There are some transformations to be made to prepare this data before we can create our visuals.

3. Check the box to select **Table 1** and click on **Transform Data**.

Tip: If the data looks good, you could click on **Load** and load it straight into Power BI. However, it is a best practice to always click on **Transform Data** so that you can get a better look at the data before using it for analysis. It will ensure fewer mistakes in the future.

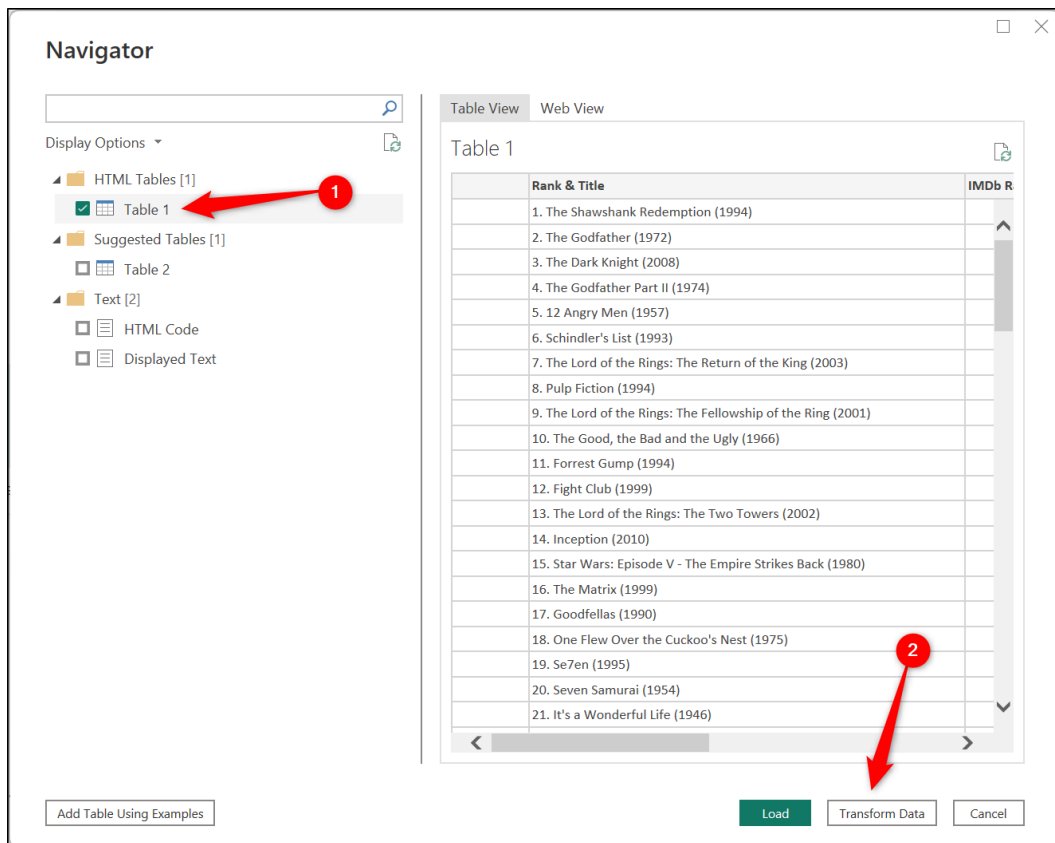


Figure 4.4: Selecting the table(s) to import from the Navigator window

Transforming data in Power Query

The Power Query window is opened, as shown in *figure 4.5*. This is a very impressive component of Power BI Desktop that makes it very easy to perform powerful data transformations.

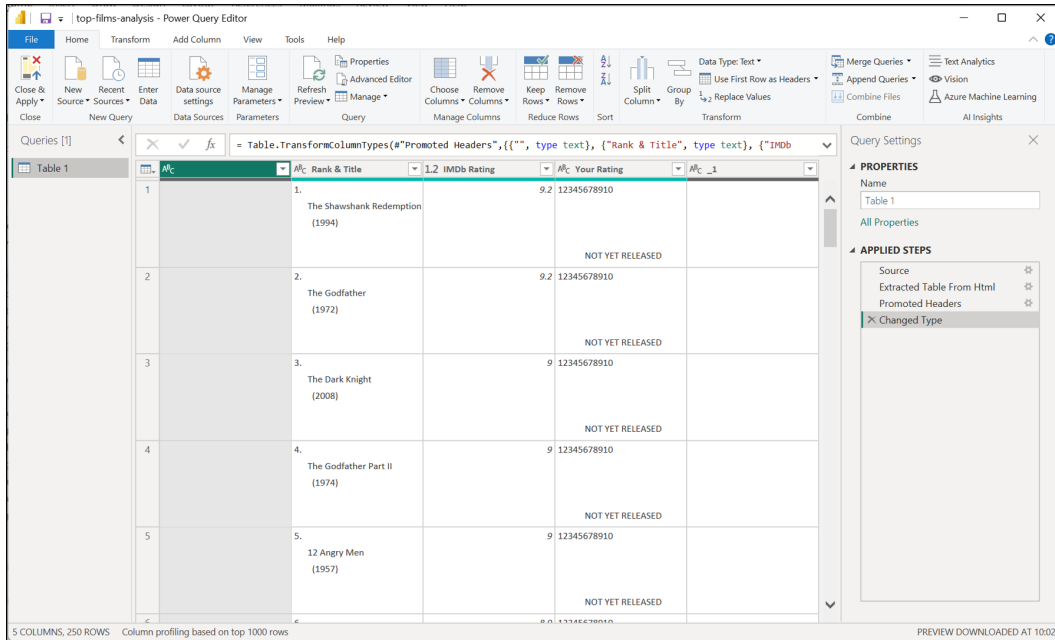


Figure 4.5: The Power Query Editor

This is the first time in the book that we have seen Power Query, so let us point out the key elements of the window, which are as follows:

- The Ribbon contains the following tabs—**Home**, **Transform**, **Add Column**, **View**, **Tools**, and **Help**. There are many useful functions found on these tabs, including merge queries, split columns, and changing data types. We will explore many of these in the next few chapters
- The Formula Bar under the Ribbon shows the code for the selected step. This is M code, the formula language of Power Query. This is very powerful; however, many transformations can be handled easily using just the UI.
- The Formula Bar is not shown by default. It can be shown by clicking on **View | Formula Bar** or in the Power BI Desktop options covered in the previous chapter.
- A Preview of the table data is shown in the main part of the window.

- The **Queries** pane is on the left. There is only one query in this example, but this area will be busier as we progress through the book.
- There is a **Query Settings** pane on the right (refer to *figure 4.6*). This is an important element of the window. Each transformation we perform is recorded as a step. You can then refresh the query to run all the steps again.

There are four steps already recorded, which are the connection to the source (Web URL), extraction of the table in the Navigator window, promotion of the header row and then the automatic **Changed Type** step.

Let us take this opportunity to rename the query using the Name box. The query was named Table 1 (taken from the Navigator window). Let us name this query as **Top Films**, as shown in *figure 4.6*:

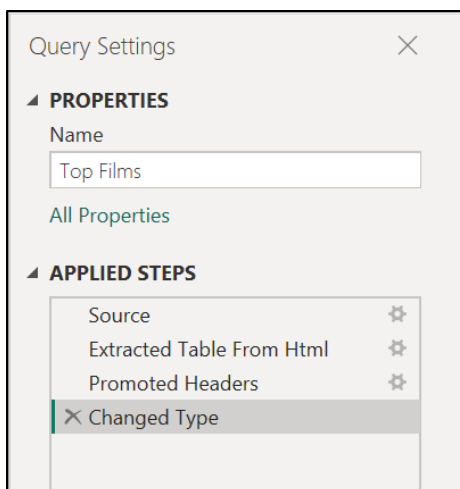


Figure 4.6: The Query Settings pane with the query name and the applied steps

Note: There is no undo button in Power Query. To reverse an action, you delete the step from the list of applied steps. You can do this with the X that appears to the left of the step name when you position the cursor on it.

Removing columns

There are a few transformations that we need to perform on this table. Let us begin by removing the unnecessary columns.

In this data, Columns 1, 4, and 5 are not required. We have two main approaches to removing these columns. We could select Columns 1, 4, and 5 and remove those specific columns. Or, we could select Columns 2 and 3 and remove the other columns.

When you work with data, you need to consider which method would be best in your scenario. In a scenario when there may be a sixth column next week, and you now need to remove columns 1, 4, 5, and 6, the approach of removing the other columns is better.

We will use the remove other columns method:

1. Click on the **Rank & Title** column, hold *Ctrl* and click the **IMDb Rating** column.
2. Right-click on one of the column headers and click on **Remove Other Columns** (refer to *figure 4.7*).

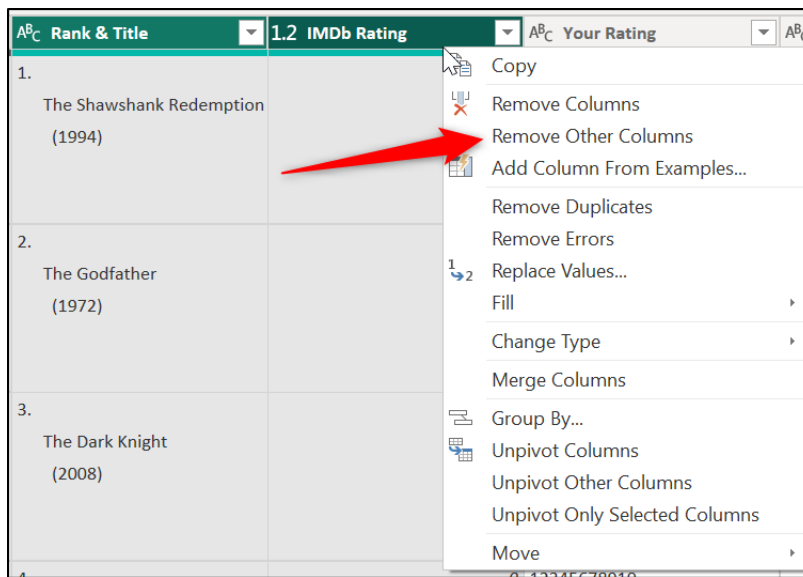


Figure 4.7: Remove other columns in Power Query

Tip: The **Choose Columns** button on the **Home** tab is very useful for removing columns. Especially when working with many columns.

The columns are removed, and this action is added as a step to the **Applied Steps**, as shown in *figure 4.8*.

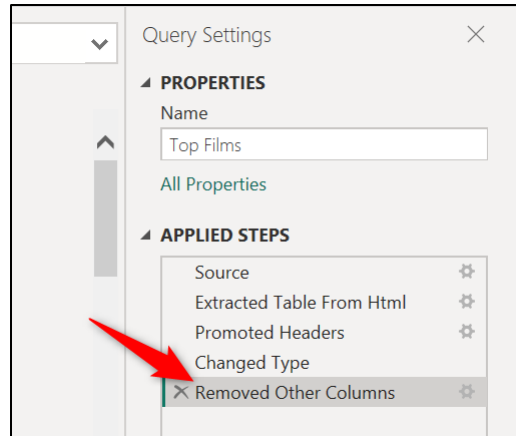


Figure 4.8: Removed other columns added as a step

Splitting columns by a delimiter

For our next step, we need to split the rank, film title and the year of release into separate columns (refer to figure 4.9).

Currently, they are all in the same column but on separate lines as they are delimited by a line feed character.

	A ^B C Rank & Title	1.2 IMDb Rating
1	1. The Shawshank Redemption (1994)	9.2
2	2. The Godfather (1972)	9.2
3	3. The Dark Knight (2008)	9

Figure 4.9: Rank, title, and year needs separating

Let us perform the following steps:

1. Select the **Rank & Title** column.
2. Click on **Home | Split Column** or **Transform | Split Column** (refer to figure 4.10).
3. Click **By Delimiter**.

There are some other useful methods for splitting columns, including **By Number of Characters** and **By Digit to Non-Digit**. We will explore more of these in *Chapter 6, More Data Transformations*.

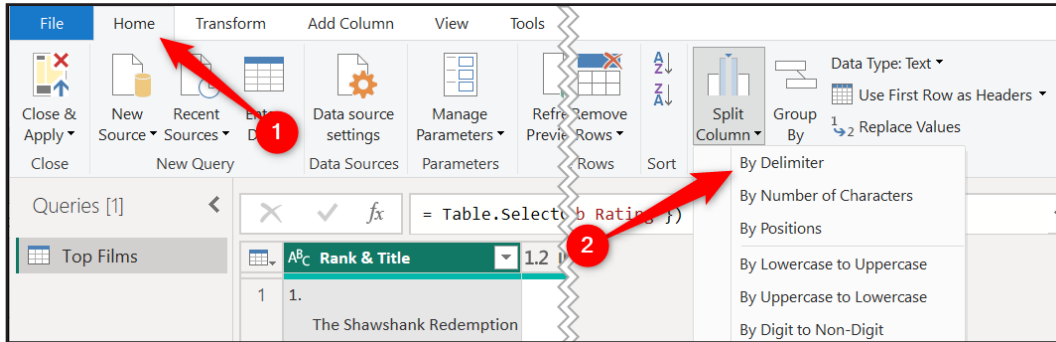


Figure 4.10: Split column by delimiter

4. The line feed character is automatically detected and suggested as the delimiter (refer to figure 4.11). This is represented by the **#(lf)**.
5. Leave the **Split at** option as Each occurrence of the delimiter. This ensures that it splits at both line feeds, one after the rank and the other after the movie title. Click **OK**.

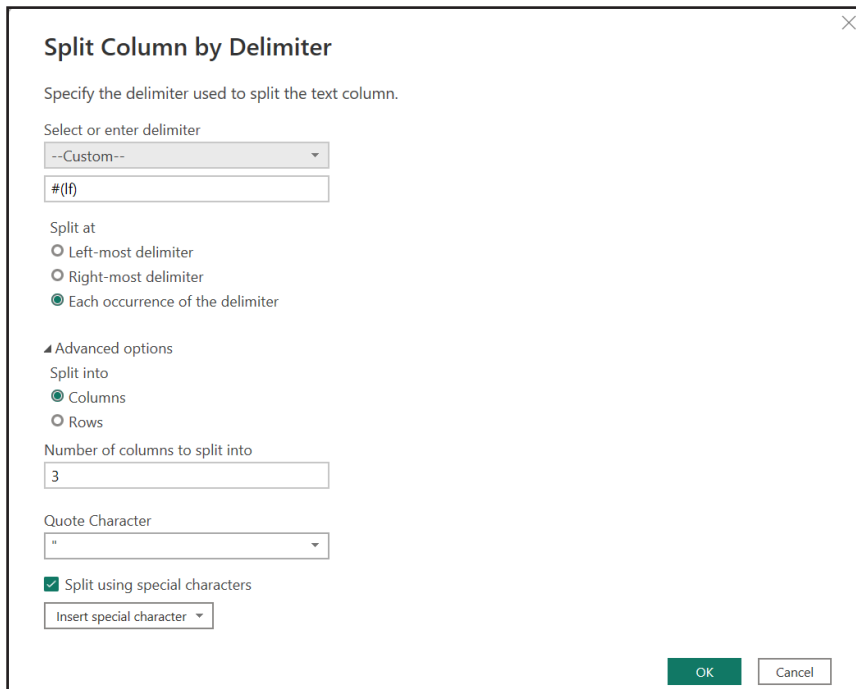


Figure 4.11: Specifying a line feed character as the delimiter

The column is split into three separate columns, as shown in *figure 4.12*. We will rename these columns in one of our final steps.

The year of release column has been stored as negative values, which is incorrect. This is due to the automatic changed type step that was run when the new columns were added. If we had disabled the automatic changed type step, discussed in the previous chapter, this would not have happened.

Because the year was enclosed in brackets, it was perceived to be a negative value. These actions are dependent upon your regional settings, and they may not occur for you.

This changed type step also converted the rank column values to a whole number data type. This is because the period (.) was assumed to be a decimal point. In your region, it may not behave this way. Please refer to the following figure:

1 ² ₃ Rank & Title.1	ABC Rank & Title.2	1 ² ₃ Rank & Title.3
1	1	The Shawshank Redemption
2	2	The Godfather
3	3	The Dark Knight
4	4	The Godfather Part II
5	5	12 Angry Men
6	6	Schindler's List
7	7	The Lord of the Rings: The Return of the King
8	8	Pulp Fiction
9	9	The Lord of the Rings: The Fellowship of the Ring
10	10	The Good, the Bad and the Ugly

Figure 4.12: Rank and Title column split into three separate columns

Converting negative values to positive

To convert the negative values for the year of release to positive values, we will multiply the values in the column by -1 .

The Power Query UI provides access to many calculations, including statistical, date, rounding and standard calculations such as division. Let us perform the following steps:

1. Select the **Rank & Title.3** column.
2. Click on **Transform** | **Standard** | **Multiply** (refer to *figure 4.13*).

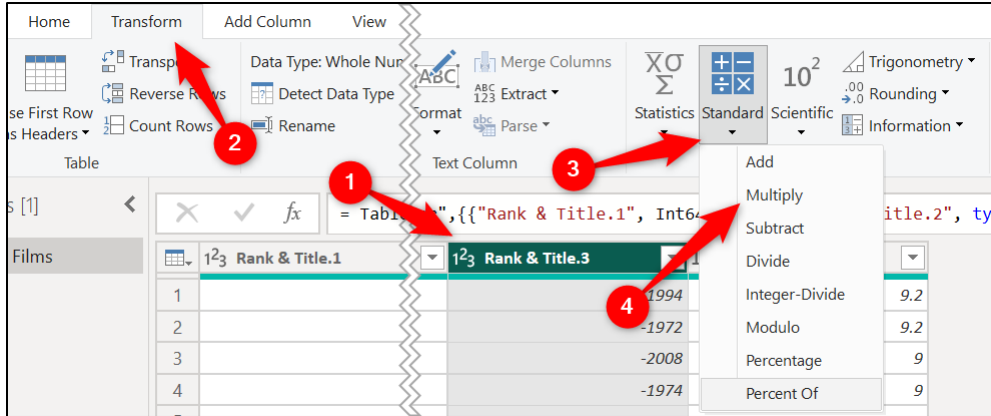


Figure 4.13: Standard calculations on the transform tab

3. Type **-1** into the **Value** field and click **OK** (refer to figure 4.14).

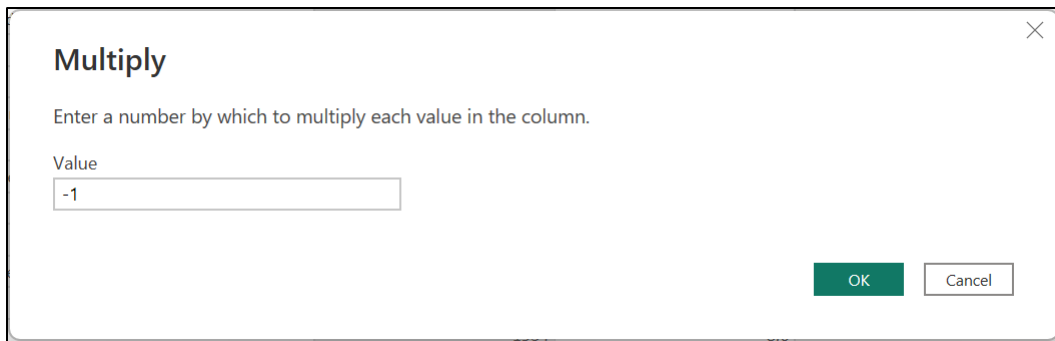


Figure 4.14: Multiply by -1 to convert to a positive value

The results of this step can be seen in Figure 4.16.

Using trim to remove excess spaces

The **Rank & Title.2** column contains excess leading spaces before the movie title. These were left over from splitting the column at the line feed characters.

These spaces can be removed easily by trimming the text in the column as follows:

1. Select the **Rank & Title.2** column.
2. Click on **Transform | Format | Trim** (refer to figure 4.15):

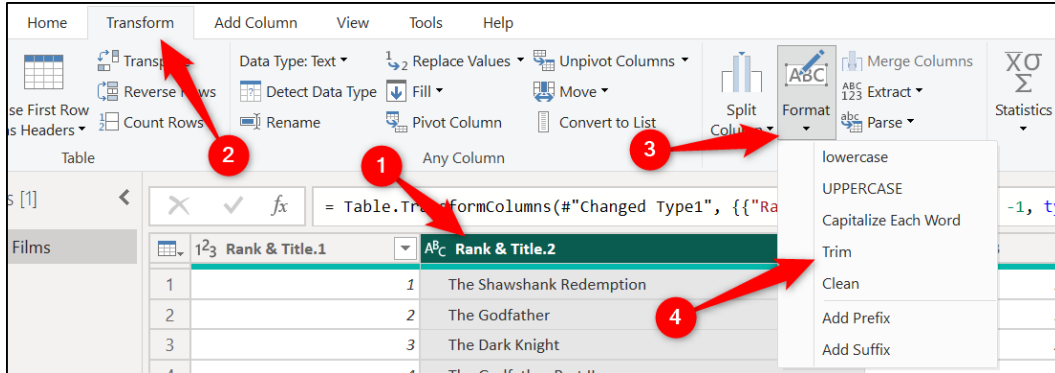


Figure 4.15: Trim text to remove excess spaces

The results of this step can be seen in *figure 4.16*.

Renaming columns

The first three column headers have non-descriptive titles. Let us rename them as follows:

1. Double-click a column header, type the new name, and press *Enter*.
2. Repeat for all the other columns.

All column header changes are recorded as one step. *Figure 4.16* shows the renamed headers:

Rank	Movie Title	Year	IMDb Rating
1	The Shawshank Redemption	1994	9.2
2	The Godfather	1972	9.2
3	The Dark Knight	2008	9
4	The Godfather Part II	1974	9
5	12 Angry Men	1957	9
6	Schindler's List	1993	8.9
7	The Lord of the Rings: The Return of the King	2003	8.9
8	Pulp Fiction	1994	8.8
9	The Lord of the Rings: The Fellowship of the Ring	2001	8.8
10	The Good, the Bad and the Ugly	1966	8.8

Figure 4.16: Table columns with meaningful headers

Changing column data types

Now that we have clean structured data, we need to check the data types of each column and make any necessary changes. The data type needs to be correct, or else we will not be able to perform the analysis that we want.

In the header of each column is a small data type button showing an icon that indicates the current data type. The **Rank** column is a whole number, **Movie Title** is text, and the **Year** and **IMDb Rating** columns are both currently decimal number data types.

We should change the data type of the **Year** column to a whole number which can be done as follows:

1. Click on the Data Type button on the **Year** column header.
2. Click **Whole Number**. Refer to *figure 4.17*.

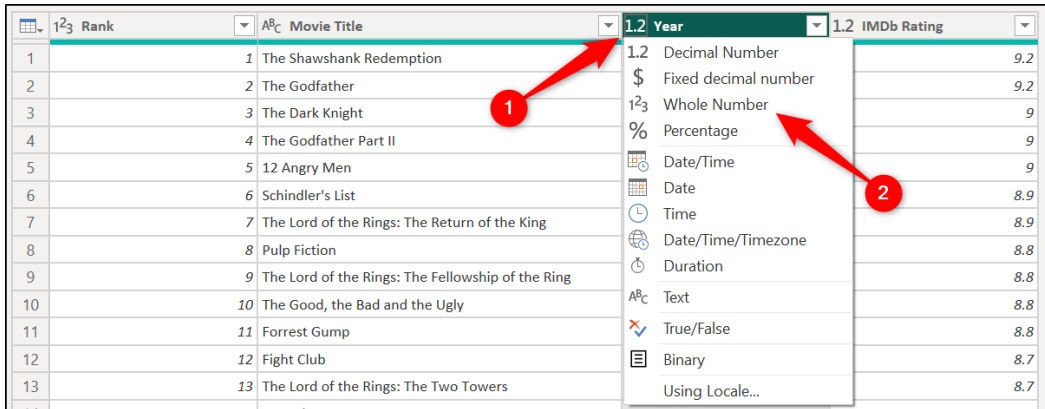


Figure 4.17: Changing the data type of a column

Power Query added a changed type step on importing the data, then again when we split a column, and we have added a further changed type step when changing the data type for the **Year** column. This results in three changed type steps, as shown in *Figure 4.18*:

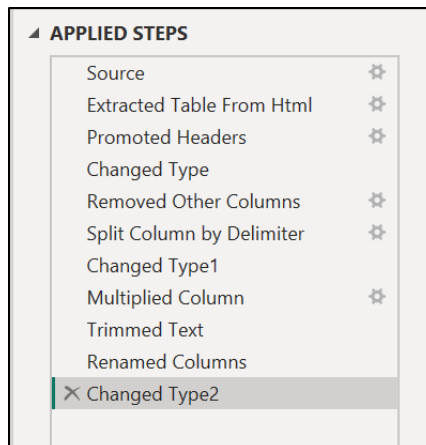


Figure 4.18: Multiple changed type steps

There is no need for this many changed type steps, and we could remove the first one to reduce the number of steps in the query and make it more readable.

Click the **X** to the left of **Changed Type**.

You are warned of potential damage to subsequent steps. This first changed type step is harmless, so click **Delete** (refer to *figure 4.19*):

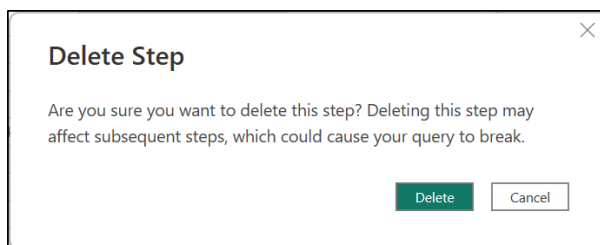


Figure 4.19: Warning that deleting a step can affect subsequent steps

Deleting that step has caused the **IMDb Rating** column to revert to a text data type. Click on the **Changed Type2** step (final step) to make it active and change the data type of the **IMDb Rating** column to Decimal Number. This action is now also included in the final step.

If we were to delete the **Changed Type1** step (the one that occurred because of the split column step), it would break the subsequent steps (refer to *figure 4.20*). It is this step that produced the negative value for the year, which we multiplied by -1 .

If we had deleted that step, the year would appear in brackets stored as text instead of a negative value, thus causing an error when Power Query attempted to multiply it. Please refer to the following figure:

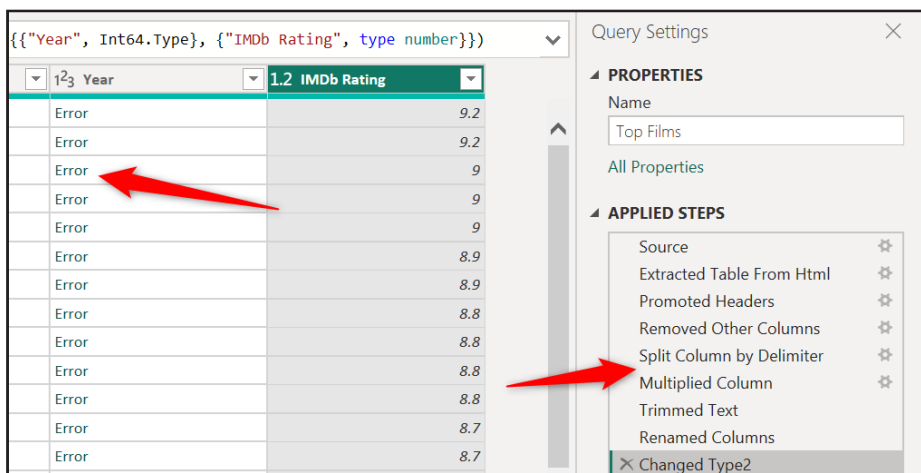


Figure 4.20: Removing the changed type1 step causes an error

Tip: Clicking on a step shows you a preview of the data at that step in the process. This can be used to check the effect the step has on our data and what would happen if we were to delete it.

We could delete both the **Changed Type1** and the **Multiplied Column** steps, remove the brackets from the year column, and then change the data type of the year column from text to a whole number in the final step. These different paths to shaping our data make Power Query so much fun.

However, this is a simple query, so let us leave that changed type step in there. Normally, the only changed type step required, except for specific circumstances, is at the end of the query.

Note: Do not confuse changing data types with formatting columns. This is a common mistake. We will format the columns when modelling the data in Power BI, not Power Query.

You may notice that the values in the [IMDb Rating] column have inconsistent decimal places, but this is not a problem.

Renaming steps

Finally, we will rename some of the steps to document the query better. The default step names are not very descriptive.

This may be a small query, but when you look back at this query in the future, it would be frustrating to read step names such as **Multiplied Column** and **Trimmed Text**. They do not detail what is happening at each step of the query well.

Right-click the step you want to rename in the **Applied Steps** list and click **Rename** (refer to *figure 4.21*):

Note: When users get more familiar with Power Query and start to write M code, they will often exclude the uses of spaces in the names of steps. This is to improve the way that steps can be referenced in the M code and is outside the scope of this book, but it is good to be aware of.

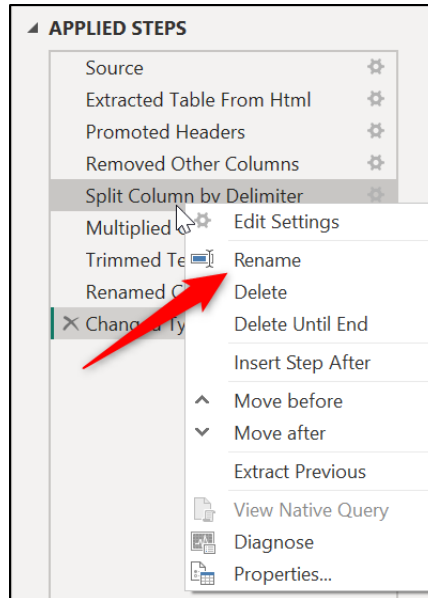


Figure 4.21: Renaming steps of the query

Continue this process to rename all the steps that you feel require the change.

Figure 4.22 shows the completed list of renamed steps.

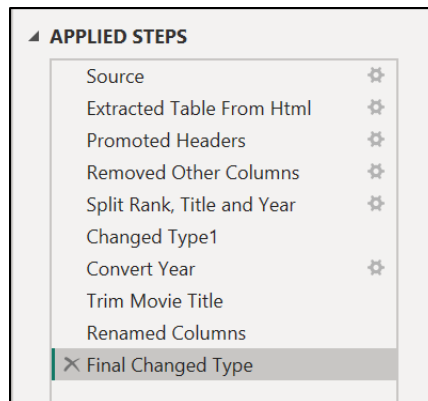


Figure 4.22: Descriptive query steps

Applying Queries in Power BI

The query is now ready to load to Power BI. Click **Home** | **Close & Apply**.

When Power BI has finished applying the query, the **Top Films** table can be seen in the **Data** pane (refer to figure 4.23).

You can expand the table by clicking the arrow to the left of its name. The columns are listed in A–Z order.

The three numeric columns (**IMDb Rating**, **Rank**, and **Year**) have a sigma next to their name. This indicates that they will perform a summarization (sum, count, average, and so on) when used on a visual. We will look at how to disable this for specific fields soon. Please refer to the following figure:

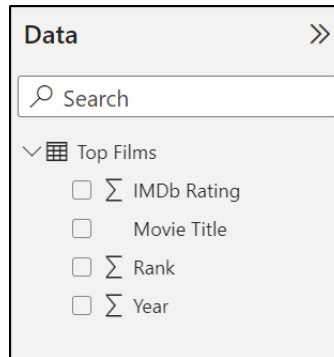


Figure 4.23: Top films table loaded to Power BI

Creating a column chart

A column chart displays each category as a vertical column (bar charts for horizontal), allowing us to compare their values easily. We will use a column chart to compare the count of films from each year.

We will use the clustered column chart visual as we only have one data series to compare. We will see the use of the stacked column and bar chart variation in *Chapter 11, Chart Visuals*.

Let us create the first visual of our report—a clustered column chart as follows:

In the **Visualizations** pane, click the Clustered column chart visual (refer to *figure 4.24*).

An empty tile for the visual is placed on the report canvas. This tile can be moved and resized as necessary. Please refer to the following figure:

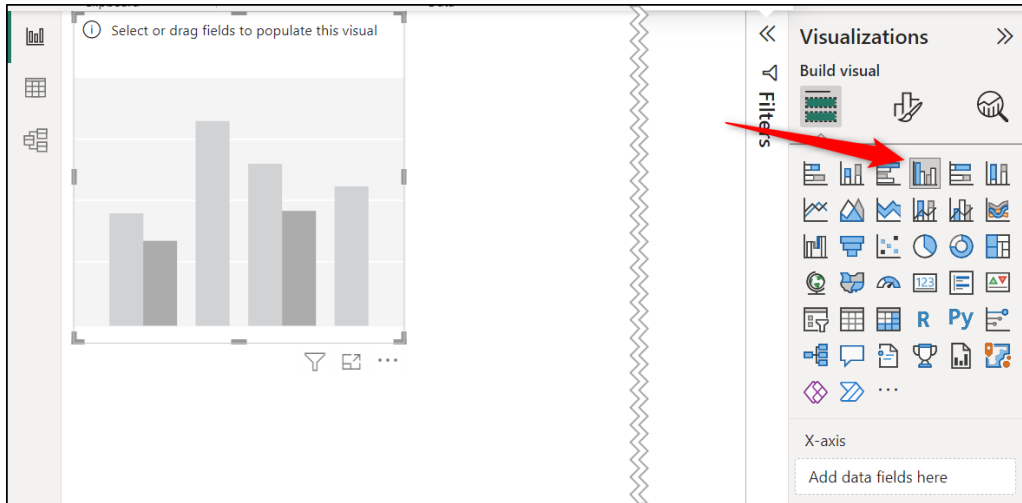


Figure 4.24: Insert a clustered column chart

1. Click and drag the **Year** field from the **Data** pane to the X-axis well of the visual.
2. Click and drag the **Movie Title** field from the **Data** pane to the Y-axis well (refer to figure 4.25).

The **Movie Title** field has a text data type, so the values are counted when they are added to the Y-axis. Please refer to the following figure:

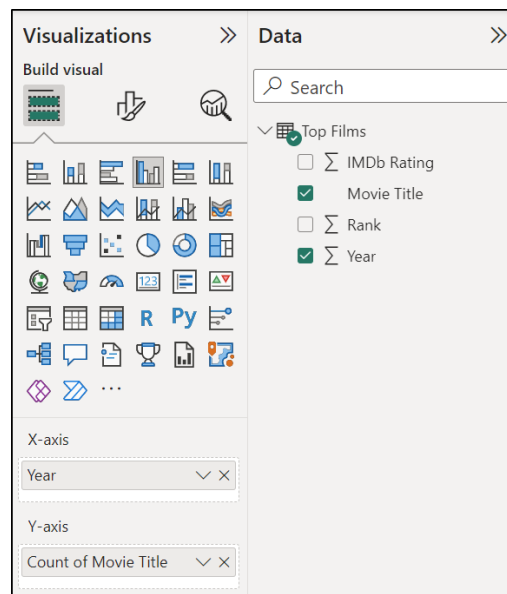


Figure 4.25: Adding fields to the column chart

The column chart shows the count of movies by year (refer to figure 4.26).

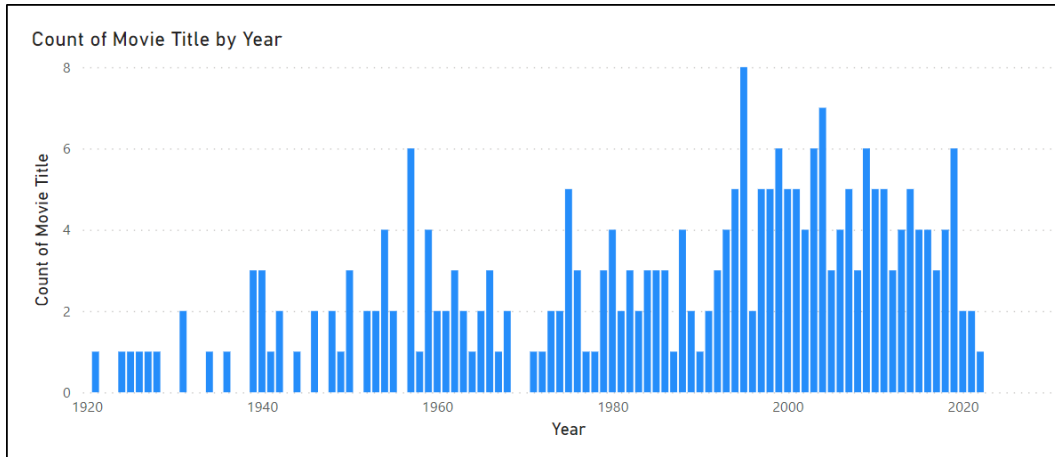


Figure 4.26: Column chart showing the count of movies by year

You can see the clustering around recent years. There are also gaps for years when no movie has made it into the top 250 movies of all time.

There is a high-performing year in the 1950s, but it is not clear which year it is. Position the mouse over the column to view the tooltip (we will get more creative with these later in the book).

The tooltip displays the year and count of movies (refer to figure 4.27). Please refer to the following figure:

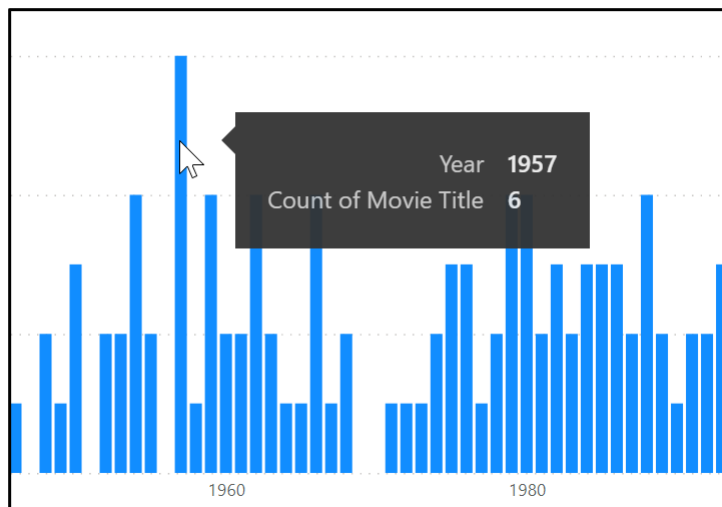


Figure 4.27: Tooltip showing further details

If you decide that a column chart is not a good fit for what you are trying to present, it is very easy to change the visual in Power BI. Let us change the visual to a line chart.

With the column chart selected, click the Line chart visual in the **Visualizations** pane to convert it into a line chart (refer to *figure 4.28*):

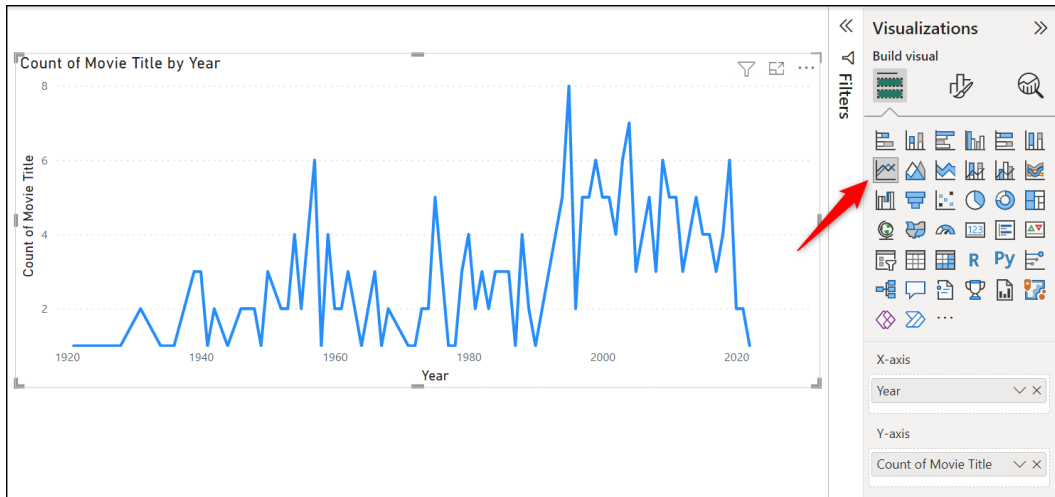


Figure 4.28: Change the column chart to a line chart

I prefer the clustered column chart, so let us change it back.

With the line chart selected, click the Clustered column chart visual to switch back.

There are many improvements that can be made to this visual. However, we will save the topic of formatting visuals until later in the book. There are chapters dedicated to the different types of visuals in Power BI. Then, we will dive deeper.

Editing existing queries

It is simple to go back and make changes to the queries at any time. Any changes are then applied to Power BI, and the report gets updated.

We have been asked to make a change to the report. Instead of showing the count of movies by year, the recipient is requesting that the chart shows the count of movies by decade.

Now, this is not data that we currently have. We have a **Year** column, but not one which groups the years by decade.

Let us edit the existing query and add a new column for the decade as follows:

Click on **Home** | **Transform data**.

Using column from examples

To create the new column, we will use a brilliant feature named **Column From Examples**. This feature will greatly simplify what may otherwise be a complicated formula.

Column From Examples will write the formula for you based on the examples you provide it with, as per the following steps:

1. With the query selected in the Power Query Editor, select the **Year** column.
2. Click on **Add Column** | **Column From Examples** list arrow | **From Selection** (refer to *figure 4.29*):

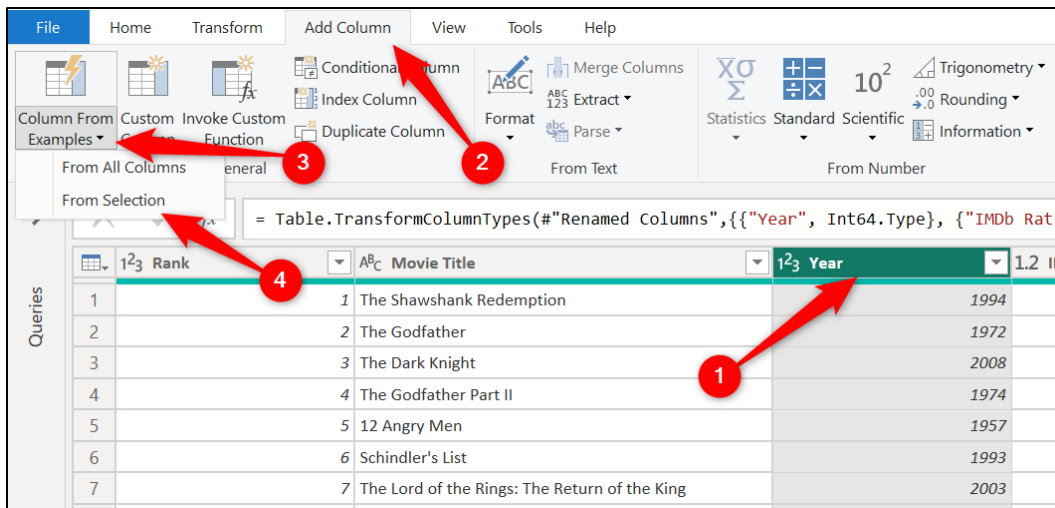


Figure 4.29: Add a column using Column from examples with the From Selection option

The **From Selection** option focuses **Column From Examples** on that column, ignoring the others. In this example, we only have one-year column, so this action was not that necessary.

However, imagine if we had more than one-year column. If we asked **Column From Examples** to look at all columns, it may not understand what column we are trying to work with.

The option you choose is dependent upon your requirements.

A column appears to the right of the data preview. To use Column From Examples, we will enter an example into one of the cells in that column.

1. Click in the first cell of the column, type “1990-1999,” and press Enter (refer to *figure 4.30*). The year of the first movie is 1994, so the decade we want is 1990-1999.

Column From Examples returns some possible results based on this example.

Unfortunately, this is not what we want. The second movie should be 1970–1979, not 1972–1981, as suggested. We need to provide it with another example:

1 ² ₃ Year	<input checked="" type="checkbox"/>	1.2 IMDb Rating	<input type="checkbox"/>	Range
1994		9.2		1990-1999
1972		9.2		1972-1981
2008		9		2008-2017
1974		9		1972-1981
1957		9		1954-1963
1993		8.9		1990-1999
2003		8.9		1999-2008
1994		8.8		1990-1999

Figure 4.30: The first example is not what we need

You can provide an example for any row in the column that is showing an incorrect result. It makes sense to pick a row that gives the best example. In our data, it does not really matter what row we use.

1. Click in a different cell, type the appropriate decade, and press **Enter**. In *figure 4.31*, we have used the fifth movie in the list and entered “1950–1959”:

1 ² ₃ Year	<input checked="" type="checkbox"/>	1.2 IMDb Rating	<input type="checkbox"/>	Range
1994		9.2		1990-1999
1972		9.2		1970-1979
2008		9		2000-2009
1974		9		1970-1979
1957		9		1950-1959
1993		8.9		1990-1999
2003		8.9		2000-2009
1994		8.8		1990-1999
2001		8.8		2000-2009
1966		8.8		1960-1969

Figure 4.31: The second example has returned the correct results

This has returned the results we wanted. Each row is showing the correct decade.

2. Press *Ctrl + Enter* or click the **OK** button above the column to confirm that we want to keep these results.
3. Rename the column to “**Decade**” and click and drag to move the column to the left of the **IMDb Rating** column (refer to *figure 4.32*).

123 Year	ABC Decade	1.2 IMDb Rating
1994	1990-1999	9.2
1972	1970-1979	9.2
2008	2000-2009	9
1974	1970-1979	9
1957	1950-1959	9
1993	1990-1999	8.9
2003	2000-2009	8.9
1994	1990-1999	8.8
2001	2000-2009	8.8
1966	1960-1969	8.8
1994	1990-1999	8.8
1999	1990-1999	8.7

Figure 4.32: Rename and move the column

Click **Home** | **Close & Apply** to apply the query to Power BI.

Modifying the column chart visual

The **Decade** field is now available in the **Data** pane. We need to swap the **Decade** field for the **Year** field in the charts’ *x*-axis for our column chart to show the count of movies by decade. This can be done as follows:

1. Click on the column chart visual to activate it.
2. Click on the **X** on the **Year** field in the *X*-axis well of the chart to remove it.
3. Click and drag the **Decade** field into the *X*-axis well of the chart (refer to *figure 4.33*).

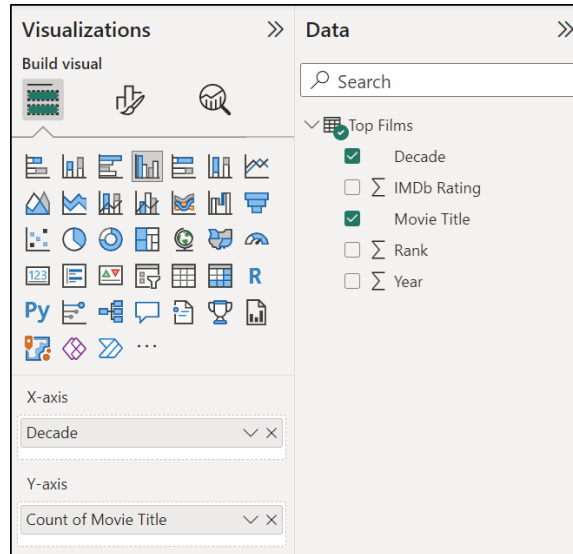


Figure 4.33: Using the decade column in the chart x-axis

The column chart now shows the count of movies by decade, as shown in figure 4.34.

By default, column charts order the data from the largest value to the smallest value when a field with the text data type is used in the *x*-axis. This is normally very good, for example, to order product names by their sales value—largest to smallest.

However, in this example, we want to order the columns by the decade from oldest to newest. So, the *x*-axis should begin with 1920–1929 and end with 2020–2029.

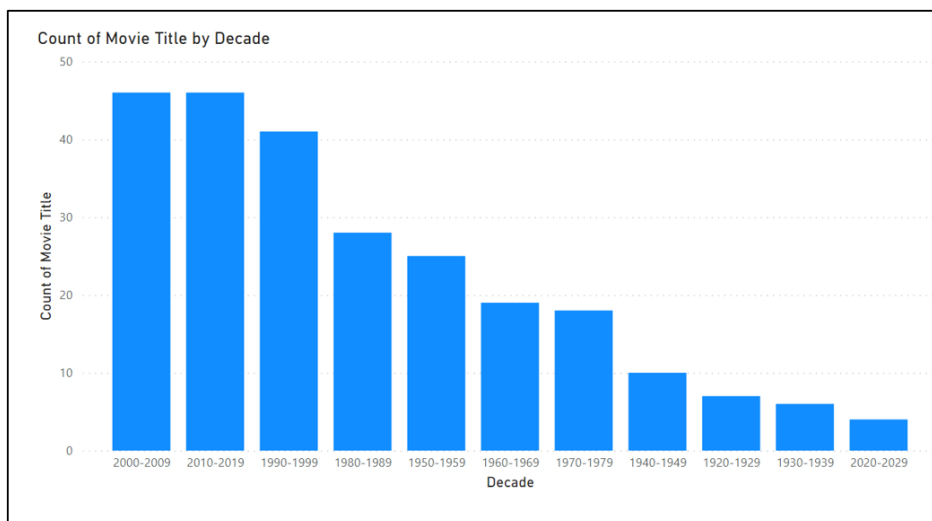


Figure 4.34: Count of movies by decade

Changing the order of the X-axis

Power BI has advanced charting functionalities. This makes sorting the values of a column chart very simple.

1. Click on the More options ellipsis in the top right corner of the column chart.
2. Position the cursor over **Sort axis** and click **Decade** (refer to *figure 4.35*).

This changes the chart to sort by the decade instead of the count of the movie title. However, it still sorts the decade in descending order. Please refer to the following figure:

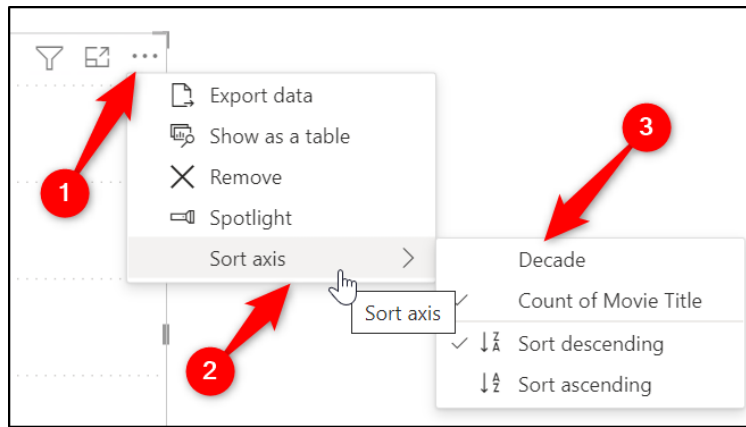


Figure 4.35: Sort the column chart values by decade

3. Click on the More options ellipsis | **Sort axis** | **Sort ascending** (refer to *figure 4.36*):

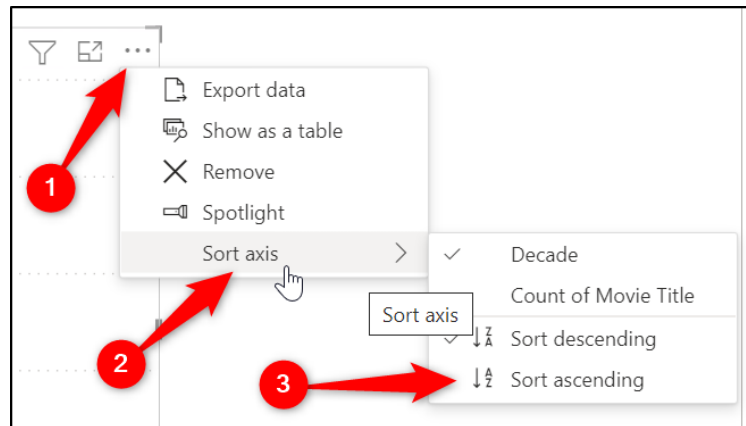


Figure 4.36: Sort chart axis in ascending order

The column chart is now ordered correctly, as shown in *figure 4.37*.

This was a nice demonstration of the built-in functionality of the rich visuals in Power BI. No formulas and no code are required.

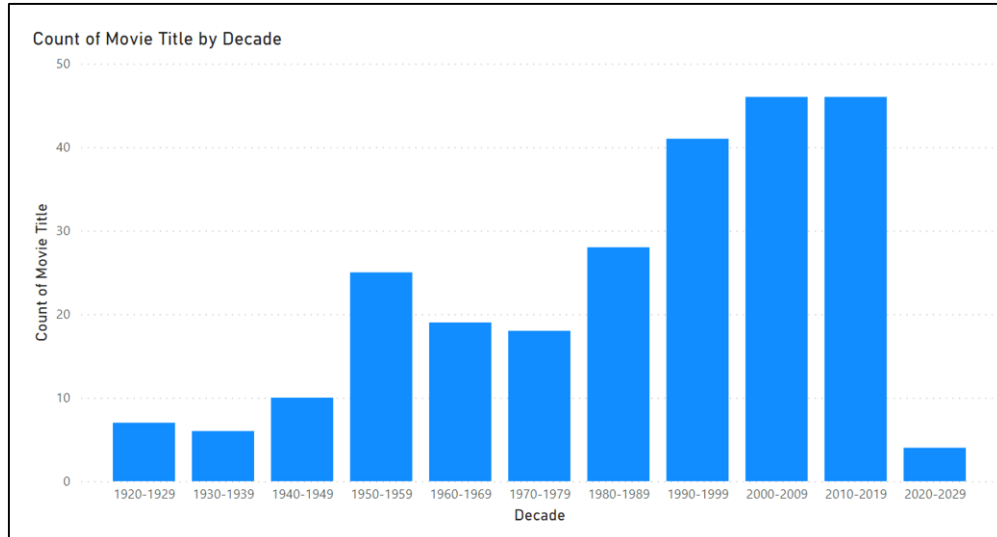


Figure 4.37: Column chart values sorted correctly

Creating a table visual

Let us add a second visual to our report. This will be the table visual.

The goal is to use the table visual to list the top 10 movies. We also want the table to be dependent on any selection made in the column chart. When a user clicks a decade in the column chart, the table will be filtered and list the top 10 movies for that decade only.

This will be very easy because that functionality is provided by default.

Note: The interactions are an option that is enabled by default but can be disabled. We have covered this option in the previous chapter.

Click somewhere on the report canvas to ensure that the column chart is deselected (otherwise, we will change the column chart to a table instead of inserting the table as a new visual).

1. Click the Table visual in the **Visualizations** pane (refer to *figure 4.38*).

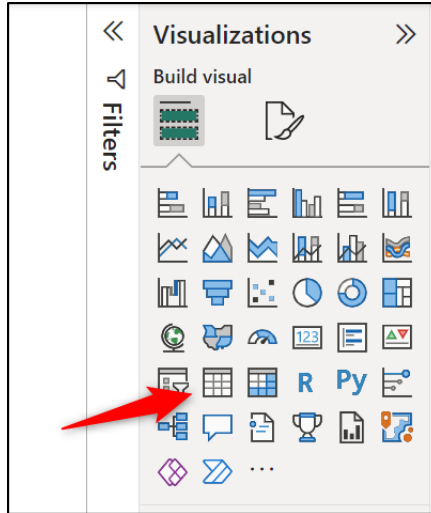


Figure 4.38: Inserting the table visual

2. Move the **Rank**, **Movie Title**, **Year**, and **IMDb Rating** fields into the **Columns** well of the table and be sure to position the fields in that order (refer to *figure 4.39*):

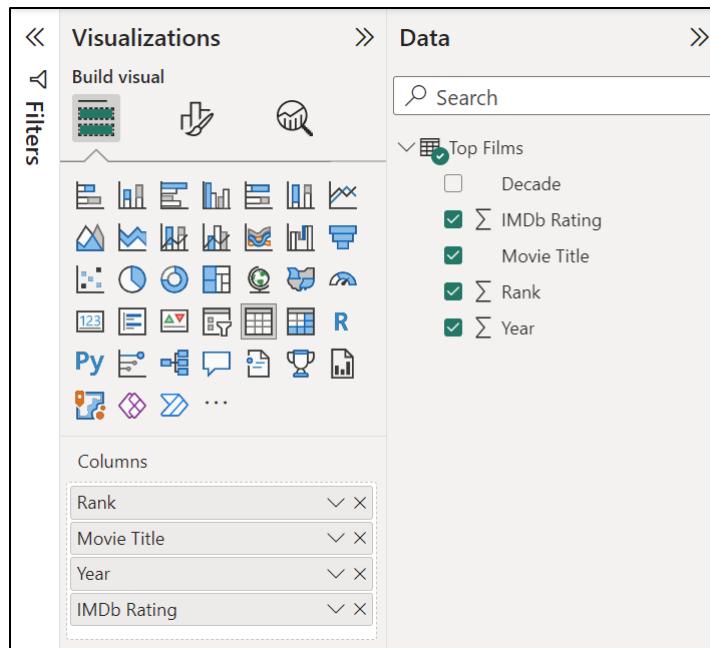


Figure 4.39: Adding fields to the table visual

3. Move and resize the table as required. I have resized the column chart to fit 10 movies only, as shown in *figure 4.40*.

Rank	Movie Title	Year	IMDb Rating
5	12 Angry Men	1957	9.00
180	12 Years a Slave	2013	8.10
123	1917	2019	8.20
91	2001: A Space Odyssey	1968	8.30
85	3 Idiots	2009	8.30
142	A Beautiful Mind	2001	8.20
104	A Clockwork Orange	1971	8.20
115	A Separation	2011	8.20
248	Aladdin	1992	8.00
51	Alien	1979	8.40
31375		496590	2,063.10

Figure 4.40: The initial table

There are a few changes we need to make to this table. They are as follows:

- Order the movies by **Rank** in ascending order (it is currently ordered A–Z by **Movie Title**).
- Format the **IMDb Rating** column to show 1 decimal place.
- Remove the totals at the bottom of the table.
- Filter the table to only show the top 10 movies. This will also remove the vertical scroll bar.

To order the movies by the **Rank** column, simply click on the **Rank** header in the table. The first click will order the movies by rank in descending order. Click a second time to order them in ascending order.

An arrow is shown in the column header to indicate the order of the table items (refer to *figure 4.41*).

Rank	Movie Title	Year	IMDb Rating
1	The Shawshank Redemption	1994	9.20
2	The Godfather	1972	9.20
3	The Dark Knight	2008	9.00
4	The Godfather Part II	1974	9.00
5	12 Angry Men	1957	9.00
6	Schindler's List	1993	8.90
7	The Lord of the Rings: The Return of the King	2003	8.90
8	Pulp Fiction	1994	8.80
9	The Lord of the Rings: The Fellowship of the Ring	2001	8.80
10	The Good, the Bad and the Ugly	1966	8.80
31375		496590	2,063.10

Figure 4.41: List of movies ordered by rank in ascending order

The next task is to format the **IMDb Rating** column to one decimal place. This can be done as follows:

1. Click on the **IMDb Rating** field in the **Data** pane.
2. From the **Column tools** tab, change the number of decimal places to **1** by using the spin box arrows, or type **1** into the box (refer to figure 4.42):

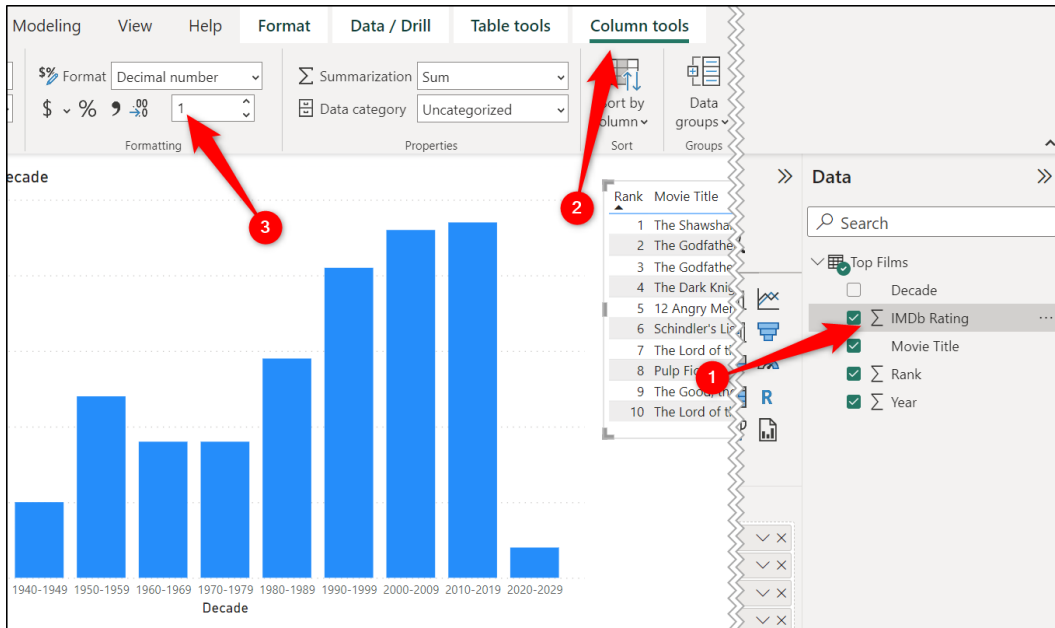


Figure 4.42: Formatting the IMDb rating column to one decimal place

Next, we will remove the totals from the table by performing the following steps. This is one of the many formatting options for table visuals, and we will explore further options in more detail in *Chapter 10, Cards and Other Text Visuals*.

1. Click on the table visual to activate it.
2. Click the **Format your visual** button in the **Visualizations** pane to switch to the formatting options, as shown in figure 4.43.
3. In the **Visual** category, click **Totals** to expand that section and click the **Values On/Off** slider to **Off**.

Figure 4.43 shows the table with the totals removed, the **IMDb Rating** column formatted to 1 decimal place, and the movies ordered by the **Rank** column in ascending order:

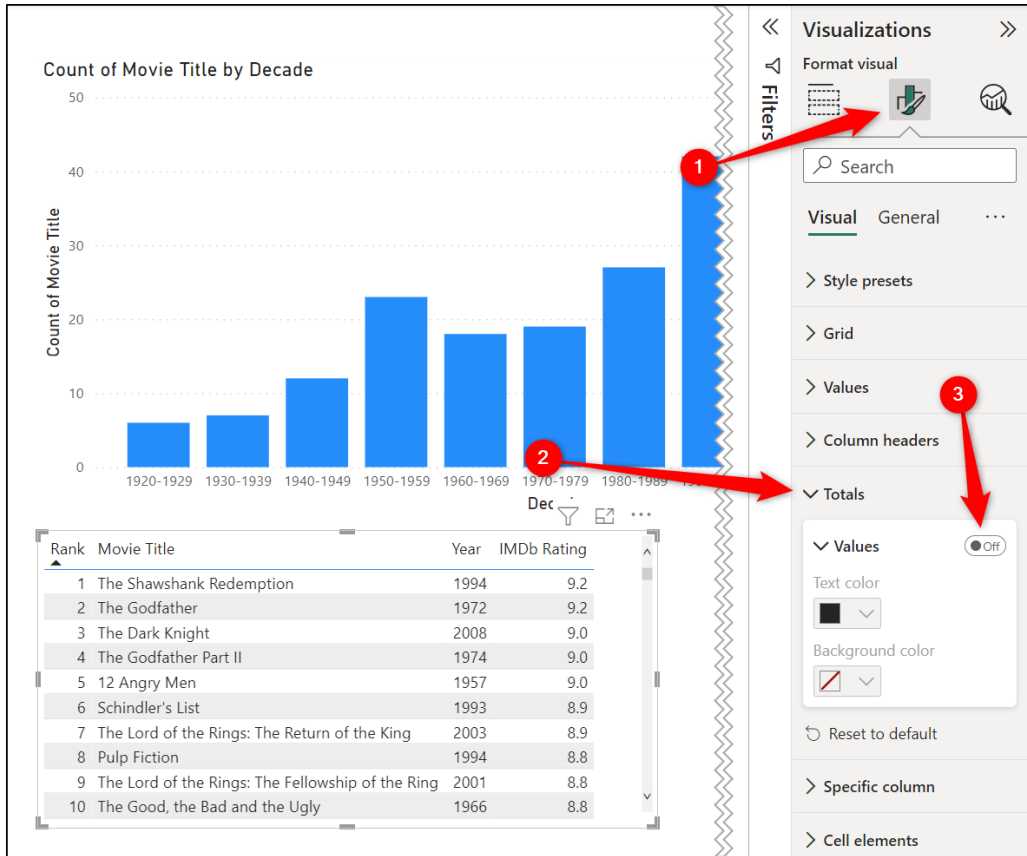


Figure 4.43: Switching the total values off for the table

The final task is to set a filter on the table to only show the top 10 movies. This is determined by their rank. So, we are really filtering for the 10 smallest rank values as follows:

1. Click on the table visual so that it is active.
2. Expand the **Filters** pane if necessary.
3. Click on **Movie Title** in the **Filters** on this visual section to expand the options (refer to figure 4.44).
4. Select **Top N** from the **Filter type** list.

5. For **Show items**, Select **Bottom**, and type **10** in the box provided.
6. Click and drag the **Rank** field from the **Data** pane into the **By value** box.
7. Click the **Apply filter** link. Please refer to the following figure:

Rank	Movie Title	IMDb Rating
1	The Shawshank Redemption	9.2
2	The Godfather	9.1
3	The Godfather: Part II	9.0
4	The Dark Knight	9.0
5	12 Angry Men	8.9
6	Schindler's List	8.9
7	The Lord of the Rings: The Return of the King	8.9
8	Pulp Fiction	8.8
9	The Good, the Bad and the Ugly	8.8
10	The Lord of the Rings: The Fellowship of the Ring	8.8

Filters

Search

Filters on this visual ...

IMDb Rating is (All)

Movie Title
bottom 10 by Rank

Filter type ⓘ
Top N

Show items:
Bottom 10

By value
Rank

Apply filter

Figure 4.44: Filtering the table to show the top 10 movies only

The vertical scroll bar is removed from the table visual, as now only 10 movies are shown at any given time.

We now have our completed table visual, as shown in *figure 4.45*. We will cover more formatting options for tables in *Chapter 10, Cards and Other Text Visuals*.

Rank	Movie Title	Year	IMDb Rating
1	The Shawshank Redemption	1994	9.2
2	The Godfather	1972	9.2
3	The Dark Knight	2008	9.0
4	The Godfather Part II	1974	9.0
5	12 Angry Men	1957	9.0
6	Schindler's List	1993	8.9
7	The Lord of the Rings: The Return of the King	2003	8.9
8	Pulp Fiction	1994	8.8
9	The Lord of the Rings: The Fellowship of the Ring	2001	8.8
10	The Good, the Bad and the Ugly	1966	8.8

Figure 4.45: The completed table visual

Interactions between the visuals

One of our report requirements is that when a decade is selected from the column chart, the table is filtered to show the top 10 movies for that decade only.

Interactions between visuals are enabled by default. So, this functionality is already active.

Click on one of the decade columns in the chart. The table reacts and filters to only show movies from the selected decade. In *figure 4.46*, the **1980–1989** column is selected:

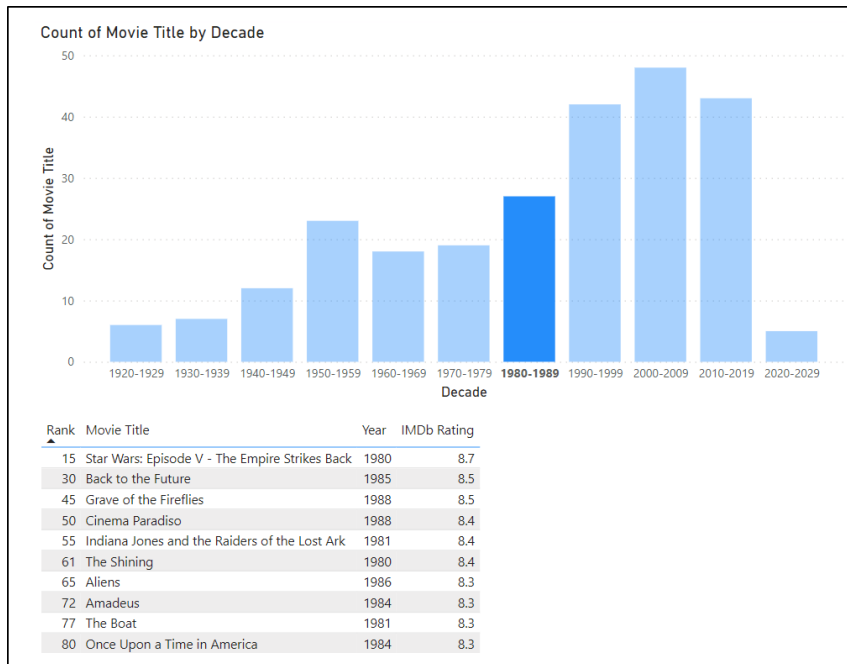


Figure 4.46: Column chart filtering the top 10 movies table

To remove the filter, click on the same column again or elsewhere in the chart.

We will explore interactions between report visuals in more depth in *Chapter 14, Report Interactions, Filters and Slicers*.

Note: If the interactions did not work when you clicked on a column, click File | Options and settings | Options | Query reduction and uncheck the Disabling cross highlighting / filtering by default option.

This was a simple and effective report to get started with Power BI.

The report is connected to the data on the Web and can be updated at any time by clicking **Home** | **Refresh**.

Conclusion

In this chapter, we learned how to connect to data on the Web and perform transformations to shape the data ready for analysis. We then created a couple of visuals to gather insights into the data.

In the upcoming chapter, we will explore more of the connectors available in Power BI Desktop. We will use these connectors to import data from sources such as a PDF, a folder, an Excel workbook, and files stored on SharePoint.

We will also see more data transformation features available in Power Query to shape our data.

Questions

Here are some questions to test what you have learned in this chapter.

1. Which of the following are transformation tools in Power Query?
 - a. Column From Examples
 - b. Split Column
 - c. Choose Columns
 - d. All of the above
2. You can disable the automatic Changed Type step of Power Query.
 - a. True
 - b. False

3. Specifying a data type is the same as formatting values.
 - a. True
 - b. False
4. Which transformation in Power Query removes excess spaces?
 - a. Clean
 - b. Proper
 - c. Trim
 - d. Split Column
5. You can change the order of values in a column chart.
 - a. True
 - b. False
6. Which steps would open the Power Query Editor to edit an existing query?
 - a. View > Power Query Editor
 - b. Home > Transform Data
 - c. Home > Edit Queries
 - d. Modelling > Transform Data

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



CHAPTER 5

Getting and Shaping Data

Introduction

To create a Power BI report, you need data. Unless you have an existing dataset to use, this means you need to get data from one or more sources into Power BI.

In this chapter, we look at how to get data from a variety of sources into Power BI. This is the beginning of the report that we will focus on creating throughout this book.

We will import data from common sources such as Excel workbooks, PDF files, and SharePoint. On importing the data, we will use Power Query to perform simple data transformations to clean and shape our data ready for use.

Structure

In this chapter, we will cover the following topics:

- Getting data from an Excel workbook
- Importing data from multiple files in a folder
- Getting data from PDF files

- Importing data from files stored on OneDrive/SharePoint
- Preparing data for analysis with Power Query

Objectives

After reading this chapter, you will be able to import data from different sources, including Excel files on SharePoint, multiple files stored in a folder, and PDF documents into Power BI.

You will also know more data transformations to clean and shape your data in Power Query, ready for analysis.

Creating a New Power BI file

Files: sales-report-data folder

In this chapter, we will begin to create a sales report. This report will feature in many of the examples in this book as we build it from scratch into a multi-page report.

Our first task is to import the data that we need from the different data sources. We need to import data about our sales transactions, the products that we sell, our sales representatives who sell our products, and the different stores that we have.

The following are the sources of that data.

- **Stores data:** Excel workbook
- **Sales reps data:** PDF document
- **Sales transactions:** Multiple CSV files stored within a single folder
- **Products data:** Excel workbook stored on SharePoint

First, let us create a new PBI file as we did in *Chapter 3: Getting Started with Power BI Desktop*.

1. Start Power BI Desktop.
2. Click **File** | **Save as**.
3. Navigate to the location you wish to save the file and enter “**sales-report**” as the filename. Click **Save**.

The examples we work with throughout this book will assume the settings covered in *Chapter 3, Getting Started with Power BI Desktop*, are left as their defaults.

Getting data with Power Query

Power Query makes it very easy to get data into Power BI for modelling and analysis. This is due to the many certified connectors available. This list of connectors is constantly growing.

To see a list of all certified connectors, click **Home** | **Get data** | **More**.

The Get Data window (*figure 5.1*) lists the certified connectors and categorizes them to make it easier to locate the one you need:

Note: You can also use custom connectors that are not yet certified by Microsoft. For example, to connect to a REST API. These may be distributed by a vendor that your organization use.

By default, Power BI only allows the use of certified connectors. To allow the use of custom connectors, you need to adjust the security options in Power BI Desktop. This is done by clicking **File** | **Options and settings** | **Options** | **Security**.

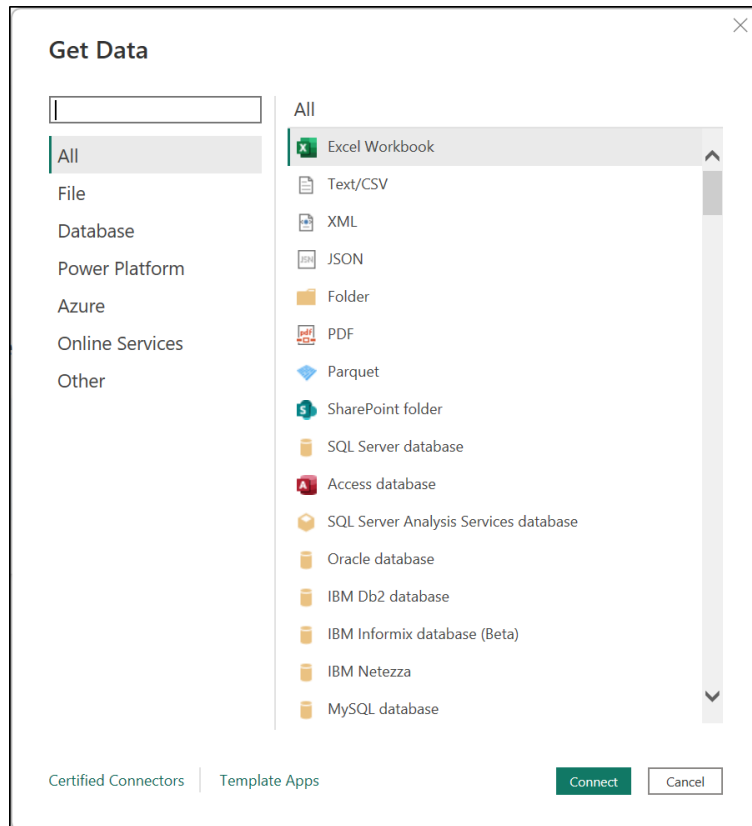


Figure 5.1: The Get Data window showing all certified connectors

There are some very useful connectors here, including SQL Server database, SharePoint folder, JSON files, and much more.

In this chapter, we cannot demonstrate all these connectors. We will import data from just a few common data sources required for our report.

Import versus DirectQuery mode

There are three storage modes available when getting data into Power BI. These are Import, DirectQuery, and Dual (also known as a composite mode).

The most popular storage mode is import. This is the default mode and the one we will use for all examples in this book.

Import mode creates and stores a copy of your data in Power BI. All Power BI service features can be performed on this data, but it must be refreshed manually. It is easy for you and others to access the underlying data of a Power BI report using this method. This is great, but there may be security implications with this direct access.

The DirectQuery mode is only possible with database connections such as a SQL Server database connection. This mode creates a connection to the data and does not import a copy of it. Because you are creating a direct connection to the data, you are always working with the current data. This does limit the data manipulation options available when compared to the import mode.

DirectQuery mode is used when you have large data that can take a long time to import when data is refreshed. Or when you are creating near real-time reports and need to be viewing the most up-to-date version of your data.

With the Dual mode, a table is the product of both the Import and DirectQuery modes. Power BI chooses the method to retrieve your data.

Figure 5.2 shows the storage mode options when using a SQL Server database connection:

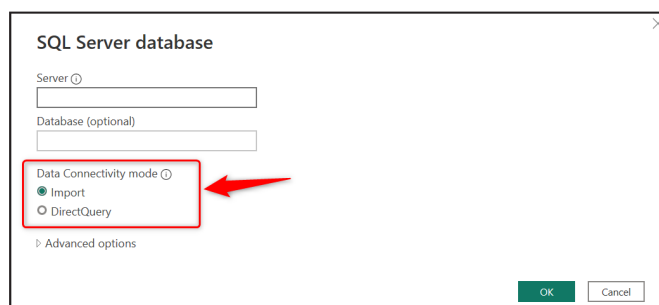


Figure 5.2: Specifying the storage mode for tables from a SQL Server database connection

We do not delve deeper into this topic in the book, but it is important to be aware of the different modes available.

You can change the storage mode of a table at any time.

1. Switch to the model view.
2. Select the table that you want to change the storage mode for.
3. From the **Properties** pane, click the **Storage mode** list and select the required option (figure 5.3).

Note: If the properties pane is not visible, right-click the table and click properties.

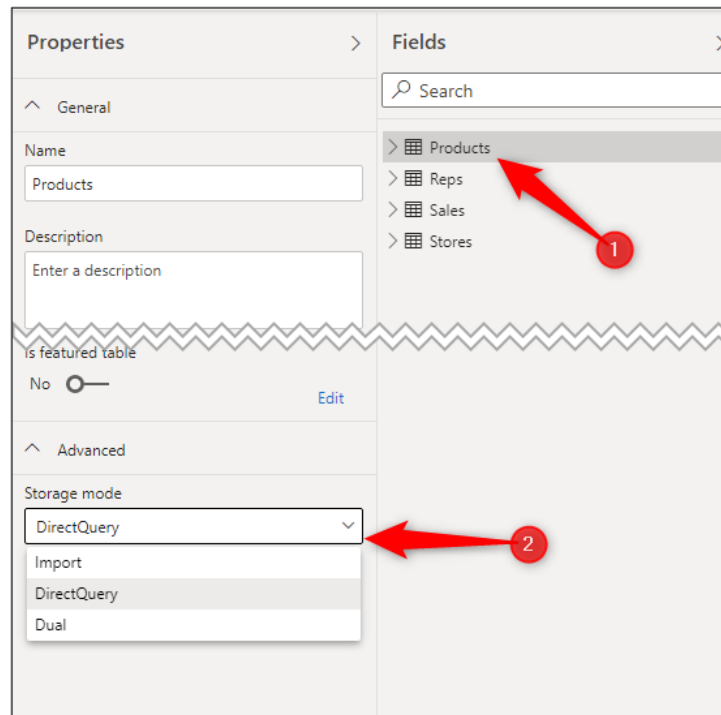


Figure 5.3: Changing the storage mode of a table

Getting data from an Excel Workbook

It comes as no surprise that Excel is a very common data source. In this example, we will import data from an Excel workbook stored locally. This workbook contains information about our stores that are distributed around the UK.

1. Click **Home** | **Excel workbook**.
2. Navigate to the **sales-report-data** folder, click the **Stores.xlsx** workbook and click **Open**.

The **Navigator** window opens (*figure 5.4*). This lists all the worksheets, tables, and defined names from the Excel workbook.

The **Stores.xlsx** workbook contains three worksheets—**Sheet1**, **Sheet2**, and **Stores**. Click on a worksheet name to see a preview of the data to the right. Sheet1 and Sheet2 are empty worksheets. The Stores worksheet contains the data we want to import.

You can see the preview of the **Stores** worksheet in *figure 5.4*. There are a few required transformations to perform, for example, removing the blank rows and splitting the region and ID in the **Store ID** column.

1. Check the box for the **Stores** worksheet only. Click **Transform Data**.

The screenshot shows the Power BI Navigator window. On the left, under 'Stores.xlsx [3]', the 'Stores' worksheet is selected with a checkmark. A red arrow with the number '1' points to this selection. On the right, a preview of the 'Stores' data is shown, including a table with columns 'Store ID', 'Store', 'Street', and 'County'. A red arrow with the number '2' points to the 'Transform Data' button at the bottom right of the window.

Store ID	Store	Street	County
Last updated 2 days ago			
		null	null
West 1	Bristol	19 Sandlings drive	Bristol
West 2	Gloucester	58 Lodge Road	Gloucestershire
West 3	Cardiff	39 Heron Way	Glamorgan
West 4	Belfast	84 Orchard Drive	Belfast
West 5	Birmingham	11 Plough Lane	West Midlands
	null	null	null
East 6	Harlow	22 Kipling Road	Essex
East 7	Ipswich	98 Oxford Drive	Suffolk
East 8	Lincoln	80 Penhill Road	Lincolnshire
East 9	Norwich	64 Corks Lane	Norfolk
	null	null	null
South 10	Brighton	71 Ashgrove	East Sussex
South 11	Guildford	65 High Street	Surrey
South 12	Portsmouth	10 Bishops Road	Hampshire
	null	null	null
North West 13	Glasgow	89 Jessops Road	Lanarkshire
North West 14	Manchester	30 Wright Lane	Lancashire
North West 15	Aberdeen	50 Church Road	Aberdeenshire
North West 16	Leeds	117 Hadfield Road	West Yorkshire
North West 17	Liverpool	42 Wellow Lane	Merseyside
North West 18	Carlisle	93 Vermont Drive	Cumbria

Figure 5.4: Selecting the sheets and tables to import from an Excel workbook

The Power Query Editor window is opened (*figure 5.5*), and we can begin to perform the required transformations to the data.

There are already four applied steps in our query. These are shown in the **Applied Steps** list in *figure 5.5*.

- **Source:** The connection to the **Stores.xlsx** file
- **Navigation:** Extracting the **Stores** worksheet only from the workbook.
- **Promoted Headers:** The first row in the data has been promoted and used for the header row.
- **Changed Type:** The data type of each column is automatically detected. In this example, the values in the **No of Staff** column have been stored as whole numbers, and all other columns have a text data type applied.

The screenshot shows the Power Query Editor interface. The main area displays a table with the following columns: Store ID, Store, Street, County, and Postcode. The data is as follows:

Store ID	Store	Street	County	Postcode	
1	West 1	Bristol	19 Sandlings drive	Bristol	BS1 1JG
2	West 2	Gloucester	58 Lodge Road	Gloucestershire	GL1 2EQ
3	West 3	Cardiff	39 Heron Way	Glamorgan	CF10 4UW
4	West 4	Belfast	84 Orchard Drive	Belfast	BT1 1AA
5	West 5	Birmingham	11 Plough Lane	West Midlands	B1 1BB
6	East 6	Harlow	22 Kipling Road	Essex	CM20 1WG
7	East 7	Ipswich	98 Oxford Drive	Suffolk	IP1 2DE
8	East 8	Lincoln	80 Penhill Road	Lincolnshire	LN1 1DD
9	East 9	Norwich	64 Corks Lane	Norfolk	NR2 1NH
10	South 10	Brighton	71 Ashgrove	East Sussex	BN3 2LS
11	South 11	Guildford	65 High Street	Surrey	GU2 4BB
12	South 12	Portsmouth	10 Bishops Road	Hampshire	PO1 2AL
13	North West 13	Glasgow	89 Jessops Road	Lanarkshire	G1 1AB
14	North West 14	Manchester	30 Wright Lane	Lancashire	M2 5DB
15	North West 15	Aberdeen	50 Church Road	Aberdeenshire	AB10 1AB
16	North West 16	Leeds	117 Hadfield Road	West Yorkshire	LS1 1UR
17	North West 17	Liverpool	42 Wellow Lane	Merseyside	L11 6JL
18	North West 18	Carlisle	93 Vermont Drive	Cumbria	CA3 8QG
19	North East 19	Newcastle	15 Roundwood Road	Tyne and Wear	NE4 9LU
20	North East 20	York	24 Cherry Lane	Yorkshire	YO1 6GA

The 'No of Staff' column is highlighted in green, indicating it has been changed to a whole number data type. The 'Applied Steps' list on the right shows: Source, Navigation, Promoted Headers, and Changed Type.

Figure 5.5: Stores query in the Power Query Editor

Removing rows

For our first task, we will remove the unrequired rows. These are the first row, which contains recent update status information, and the blank rows.

We can accomplish this task by filtering out the null values from any column except the **Store ID** column. This is shown in *figure 5.6*, where the filter on the **Street** column is used.

Note: In Power Query, null values are cells with no value. A blank cell is not necessarily a cell with a null value, as cells that contain a space or an empty string "" will not be identified in Power Query as null.

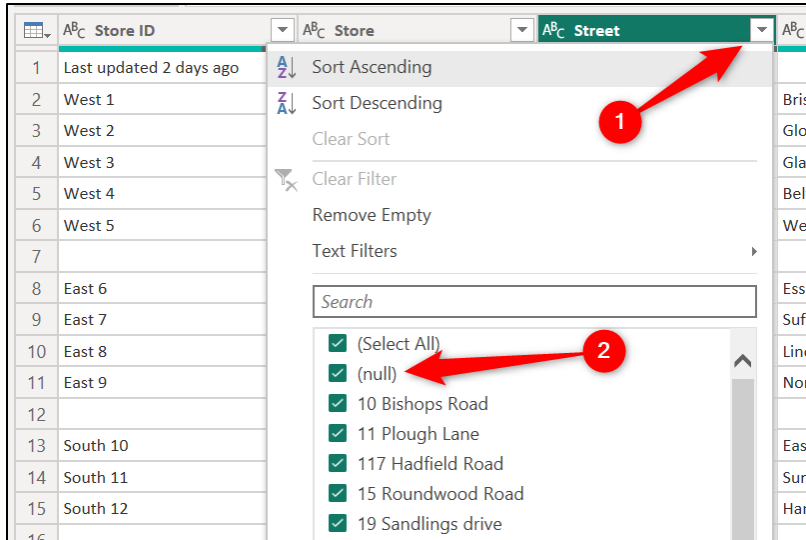


Figure 5.6: Filtering out the rows with null values

This accomplishes the task in one step. However, let us look at removing the rows in two different steps. This gives us a chance to explore further transformations. We will remove the blank rows and then remove the first row.

Tip: The approach you take when transforming your data ultimately depends on your data. The correct approach depends on the specific case. If this is a query that you will refresh regularly, which approach is more reliable for the future.

1. Click **Home | Remove Rows | Remove Blank Rows** (figure 5.7).

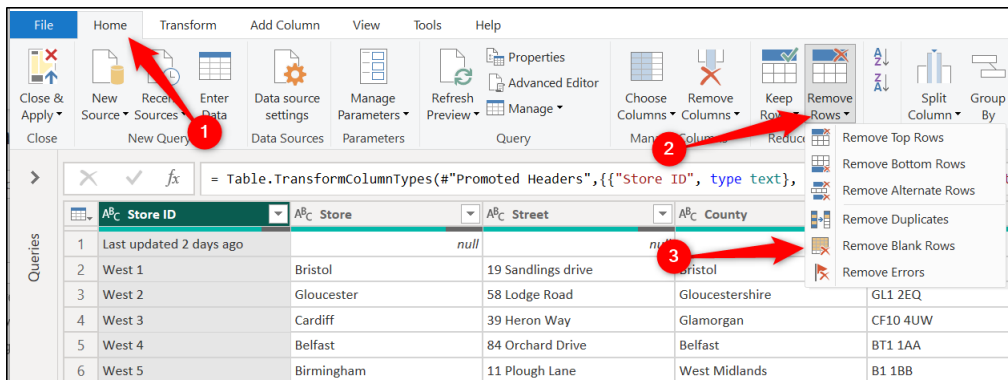
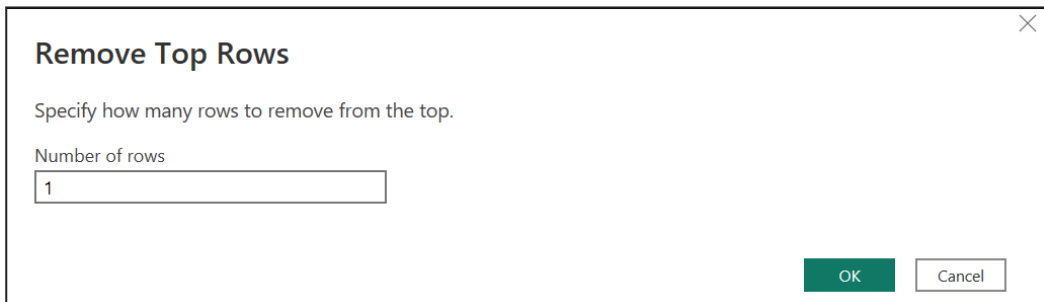


Figure 5.7: Removing blank rows in a query

2. Click **Home | Remove Rows | Remove Top Rows**.
3. Type “1” for the Number of rows and click **OK** (figure 5.8).



Remove Top Rows

Specify how many rows to remove from the top.

Number of rows

1

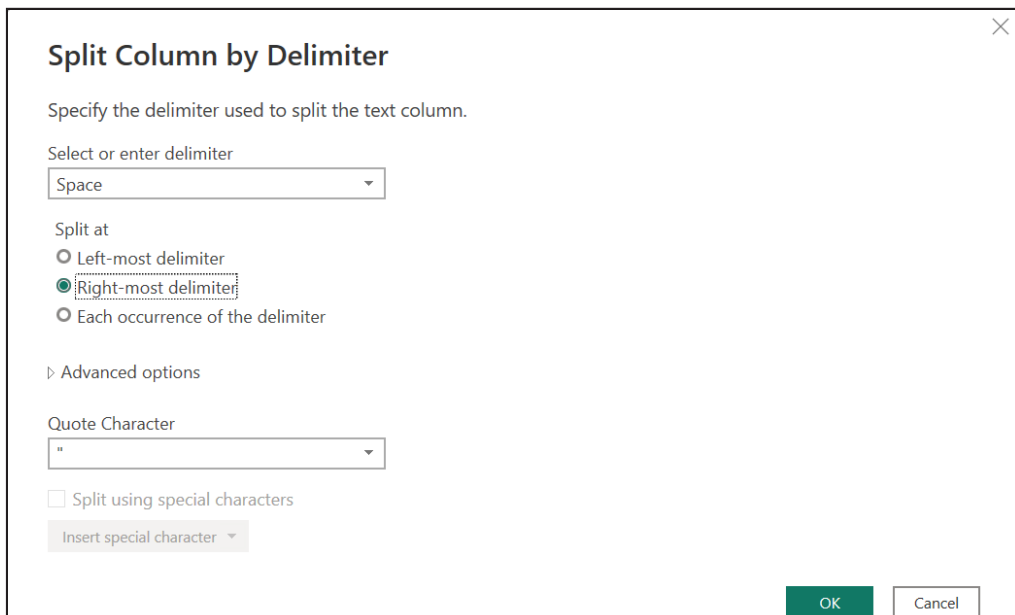
OK Cancel

Figure 5.8: Entering the number of rows to remove

Splitting columns

The next task is to split the region and ID from the **Store ID** column into different columns.

1. Click on the **Store ID** column.
2. Click **Home | Split Column | By Delimiter**.
3. Specify a **Space** for the delimiter and select the right-most delimiter for the Split at option (figure 5.9). Click **OK**.



Split Column by Delimiter

Specify the delimiter used to split the text column.

Select or enter delimiter

Space

Split at

Left-most delimiter

Right-most delimiter

Each occurrence of the delimiter

Advanced options

Quote Character

"

Split using special characters

Insert special character

OK Cancel

Figure 5.9: Splitting a column at the final occurrence of a space

Specifying the right-most delimiter was important; otherwise, the regions named **North West** and **North East** would have been split at their space (figure 5.10):

	A ^B _C Store ID.1	1 ² ₃ Store ID.2
1	West	1
2	West	2
3	West	3
4	West	4
5	West	5
6	East	6
15	North West	15
16	North West	16
17	North West	17
18	North West	18
19	North East	19
20	North East	20
21	North East	21

Figure 5.10: Region names containing spaces remain intact

You can click and drag columns to easily re-order them in Power Query. Let us re-order the ID and region columns so that the ID column is first (figure 5.11).

	1 ² ₃ Store ID.2	A ^B _C Store ID.1	A ^B _C Store
1	1	West	Bristol
2	2	West	Gloucester
3	3	West	Cardiff
4	4	West	Belfast
5	5	West	Birmingham
6	6	East	Harlow
7	7	East	Ipswich
8	8	East	Lincoln

Figure 5.11: Re-ordering the columns in the query

When the query is applied, and the data is loaded into Power BI, the Fields pane will order the columns in A–Z order and not in the order we see them in the query.

So, this step is not hugely helpful. But it may still be nice to have this order while working in Power Query.

Data types and renaming columns

The last steps in any query are to check the data types and header names for each column of the query.

The names of the first two columns require changing in this query.

Double-click on the column header to change the **Store ID.2** column to **Store ID** and the **Store ID.1** column to **Region** (figure 5.12).

	1 ² ₃ Store ID	A ^B _C Region	A ^B _C Store
1	1	West	Bristol
2	2	West	Gloucester
3	3	West	Cardiff
4	4	West	Belfast
5	5	West	Birmingham
6	6	East	Harlow
7	7	East	Ipswich
8	8	East	Lincoln
9	9	East	Norwich

Figure 5.12: Renaming the columns

With this query, there are no changes to be made to the column data types, but it is important to perform these checks at the end of a query.

Many times, an error has occurred when modelling the data, and the cause has been that a column has an incorrect data type.

We will not close and apply this query to Power BI yet, as we have more data to connect to. We will remain in Power Query and connect to a PDF file next.

Getting data from PDF files

We will now get our sales rep data from a PDF document stored locally. We will do this from within the Power Query editor, following immediately from the previous data source connection.

1. Click **Home** | **New Source** | **More** (figure 5.13).

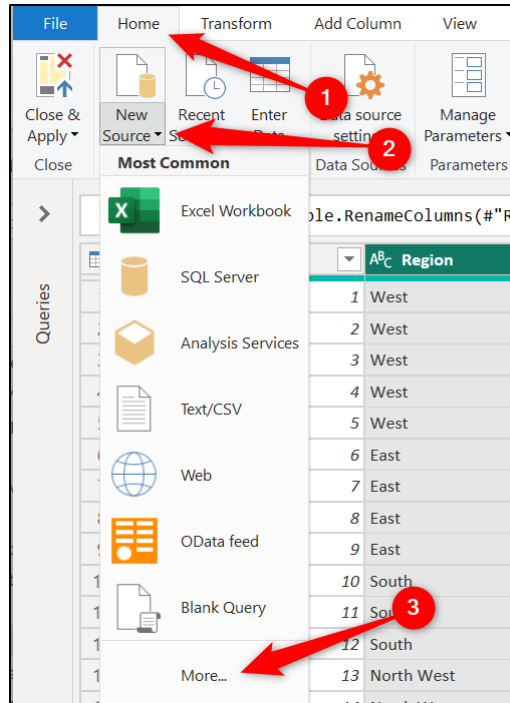


Figure 5.13: Getting data from within the Power Query Editor

2. Click the **PDF** connector and click **Connect** (figure 5.14).

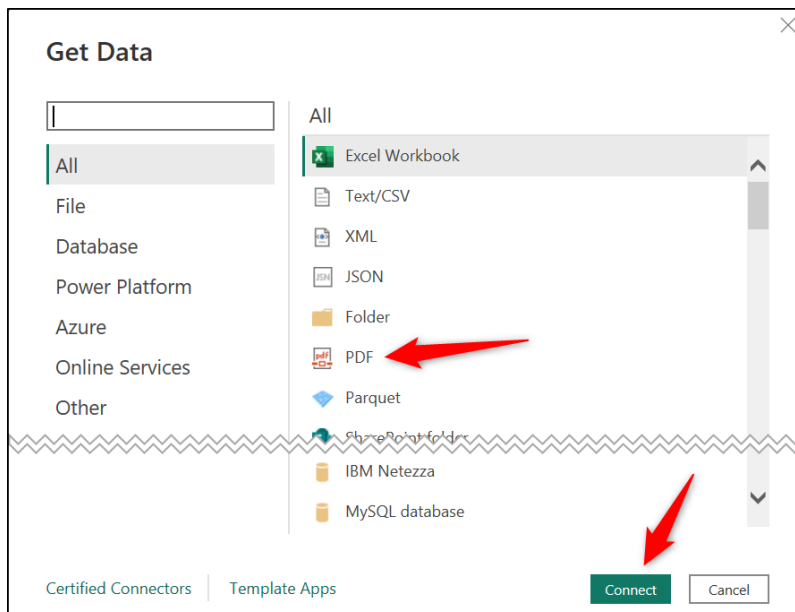


Figure 5.14: Connecting to a PDF file

- In the Open window, select the **Sales Reps.pdf** file and click **Open**.

The Navigator window lists all the tables and pages that the PDF connector can identify in the document (*figure 5.15*). A preview of the data is shown when a table or page is selected.

This document contains two pages with one table that spans both pages. Unfortunately, the connector has identified the table as two separate tables. We will load both tables to Power Query and then combine them into a single table.

Check the box for **Table001** and **Table002** and click **OK** (*figure 5.15*).

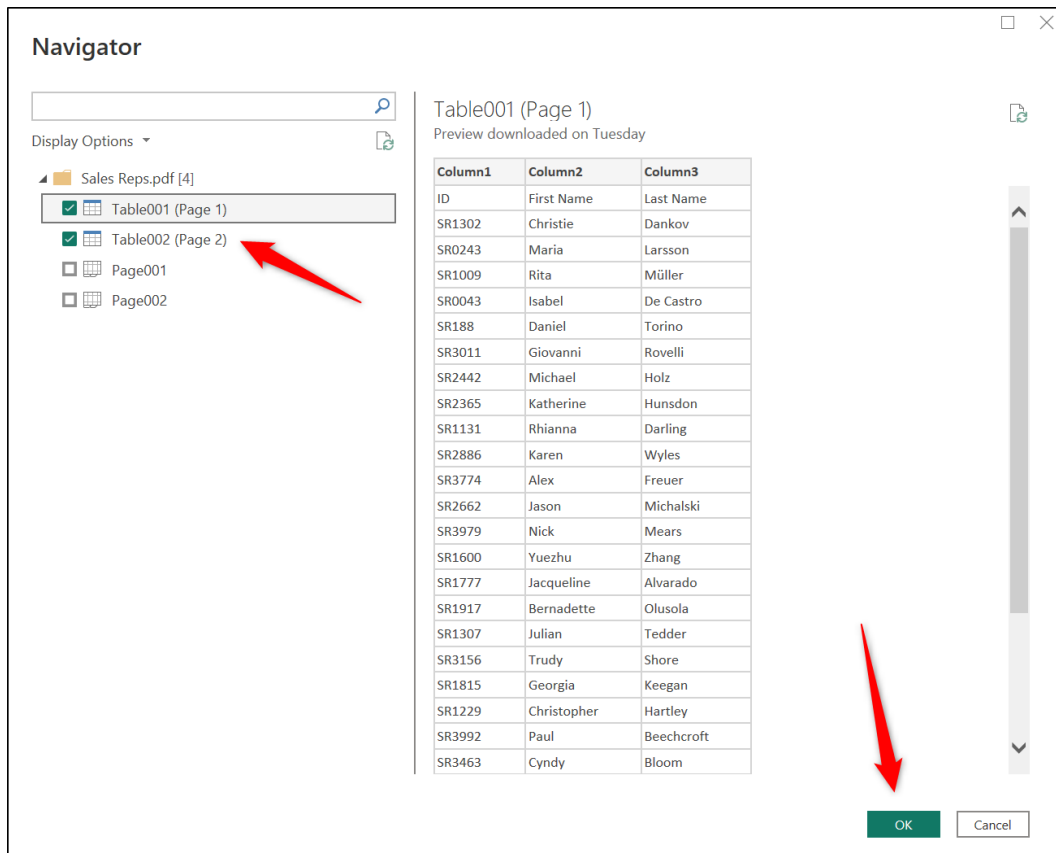
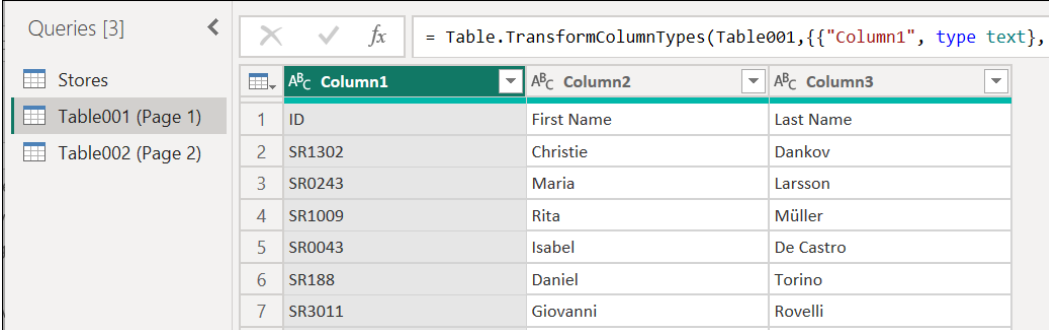


Figure 5.15: Specifying the tables to import in the Navigator window

Appending queries

The two tables are loaded to Power Query and are shown in the Queries pane along with the **Stores** query from before (*figure 5.16*).

The headers in **Table001** have not been recognized and are used as the first row of data in the table. This is incorrect; however, we do not want to correct this just yet.



Column1	Column2	Column3
ID	First Name	Last Name
1	SR1302	Christie
2	SR0243	Maria
3	SR1009	Rita
4	SR0043	Isabel
5	SR188	Daniel
6	SR3011	Giovanni
7		Rovelli

Figure 5.16: Two tables with no headers recognized

We need to append **Table002** to **Table001** to form one table. For this, we need the headers of both tables to match, and currently, they do. Both tables have **Column1**, **Column2**, and **Column3** as their headers.

1. Click **Table001** in the **Queries** pane.
2. Click **Home** | the list arrow for **Append Queries** | **Append Queries as New** (figure 5.17)

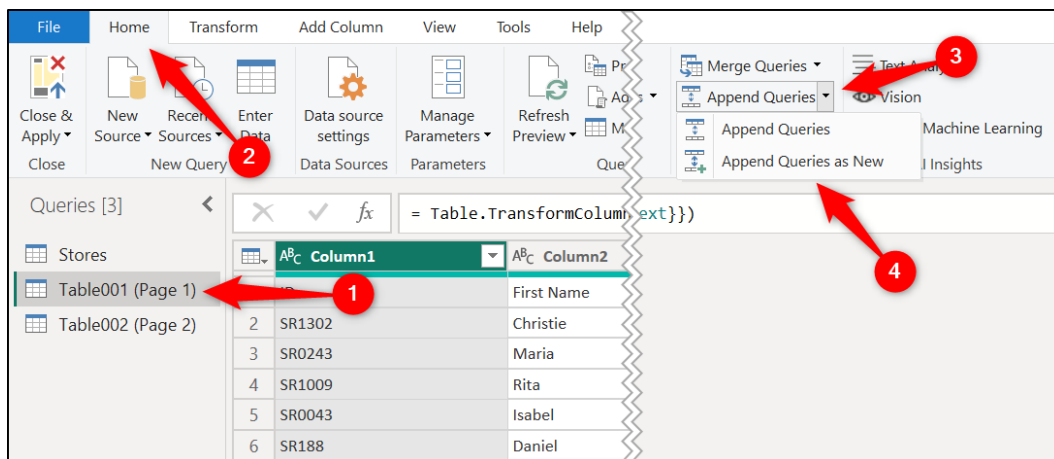


Figure 5.17: Append queries as a new query

The **Append** window allows you to easily select all the tables that you want to append. **Table001** is already selected as the first table because we clicked it before starting the append query (figure 5.18).

Note: The three or more tables option provides a ListBox so you can add all the queries/tables that you want to append to.

- Click the list for the **Second table** and select **Table002**. Click **OK**.



Figure 5.18: Selecting the tables to append

A new query is created that includes both tables combined into a single table. The rows from **Table002** have been appended beneath the rows of **Table001**. This combined table is the table that we will use in our sales report data model.

Right-click on the **Append1** query, click **Rename** and name it **Reps** (*figure 5.19*).

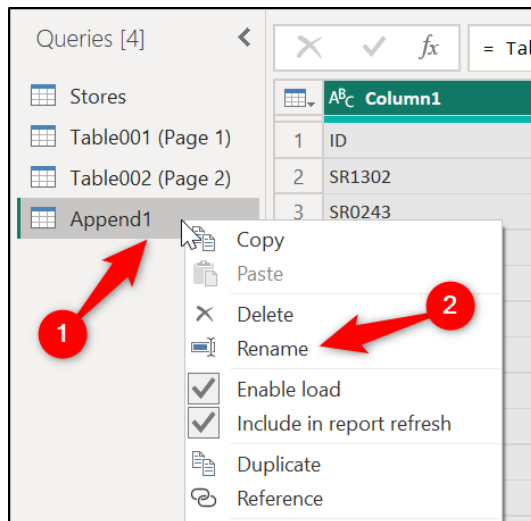


Figure 5.19: Renaming a query

This query references the other two queries, so it is dependent upon them. Although we will not use **Table001** or **Table002** directly for data in our Power BI report, they are important, as they are our connection to the **Sales Reps.pdf** file.

When working with multiple queries in Power Query, the Query Dependencies window can be a useful way to help understand how the queries work together.

To open the Query Dependencies window, click **View | Query Dependencies**. In *figure 5.20*, you can see the dependencies of our queries so far:

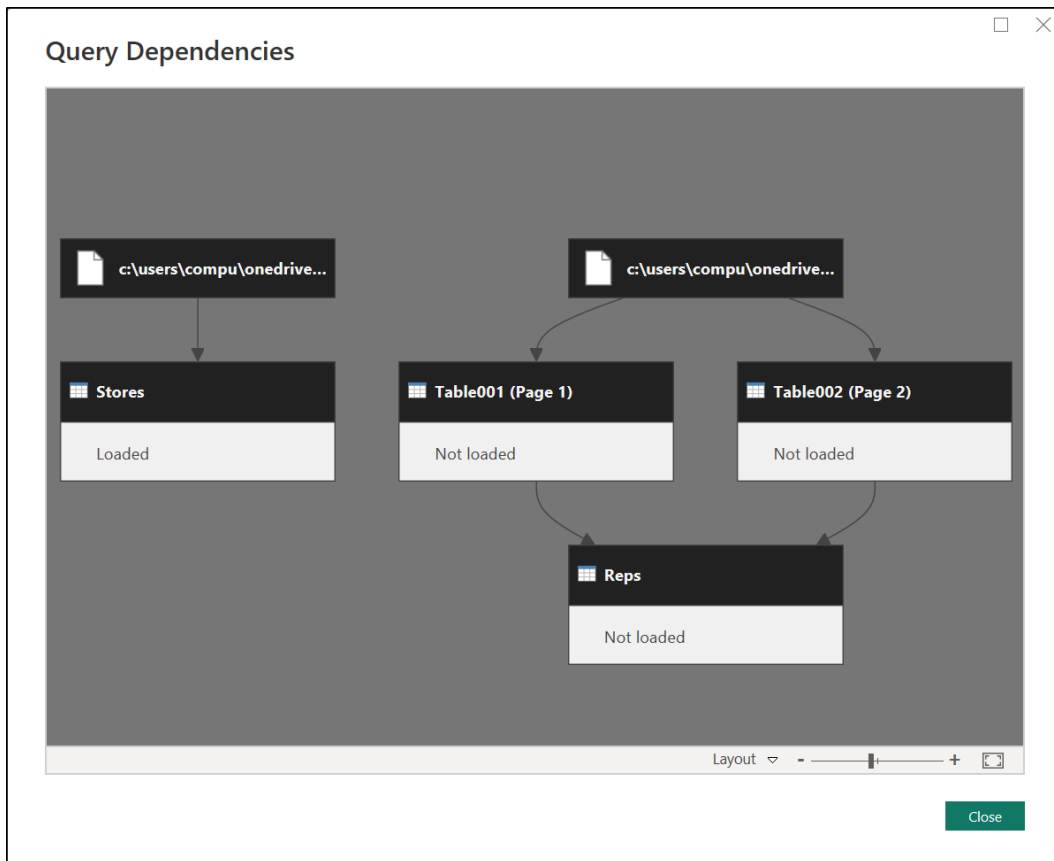


Figure 5.20: Viewing the query dependencies

Promote the headers of a table

Now that we have appended the two queries into one, we can promote the first row to be used as headers.

Click **Home | Use First Row as Headers**.

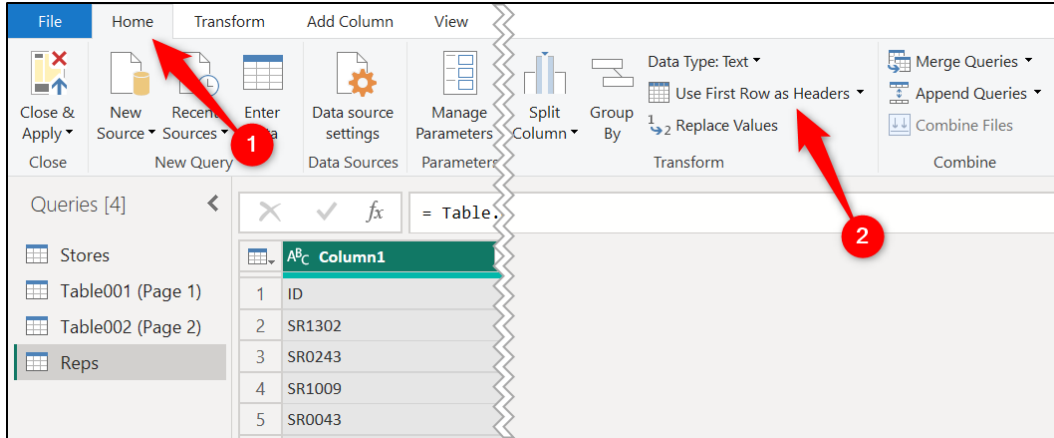


Figure 5.21: Promote the first row as headers

Merging columns

Our final transformation is to merge the **First Name** and **Last Name** columns together to create a **Full Name** column.

1. Click the **First Name** column.
2. Press and hold *Ctrl* and click the **Last Name** column.
3. Click **Transform | Merge Columns** (figure 5.22).

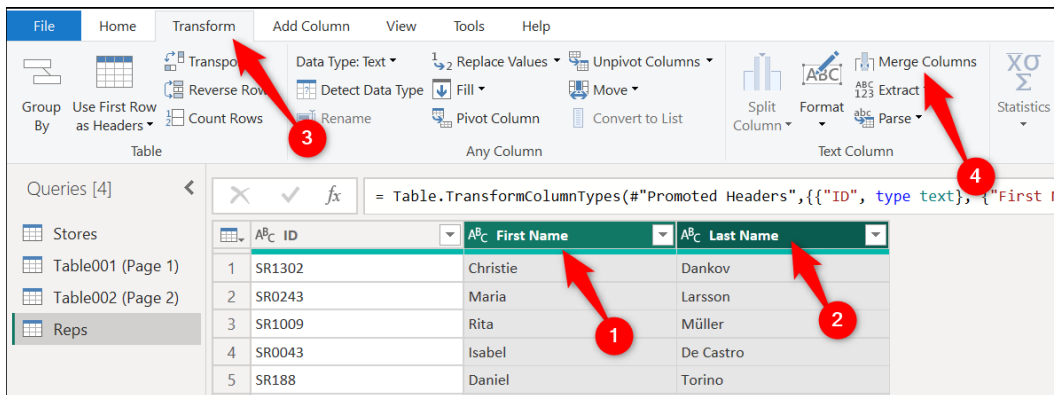
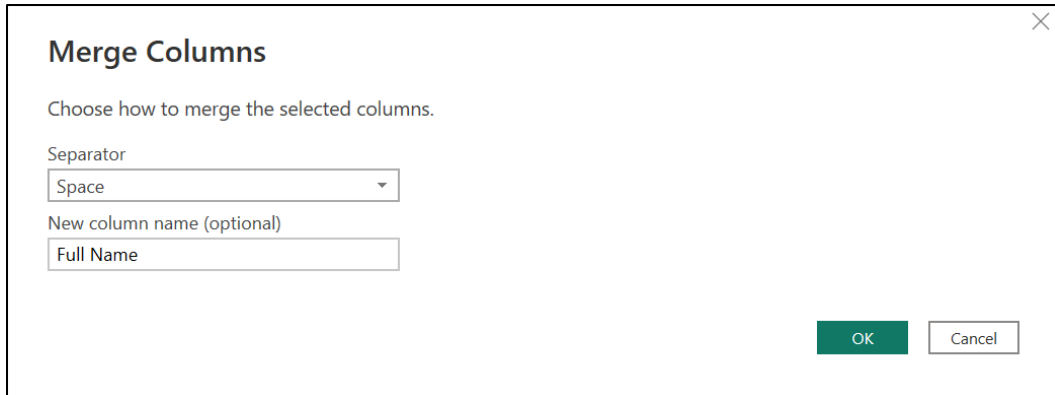


Figure 5.22: Merge columns in Power Query

The order in which the columns are selected dictates the order in which the content is merged, so it is important to select the **FirstName** column first in this example.

1. In the **Merge Columns** window (figure 5.23), Click the **Separator** list and click **Space**.

2. Type **Full Name** for the New column name. Click **OK**.



Merge Columns

Choose how to merge the selected columns.

Separator
Space

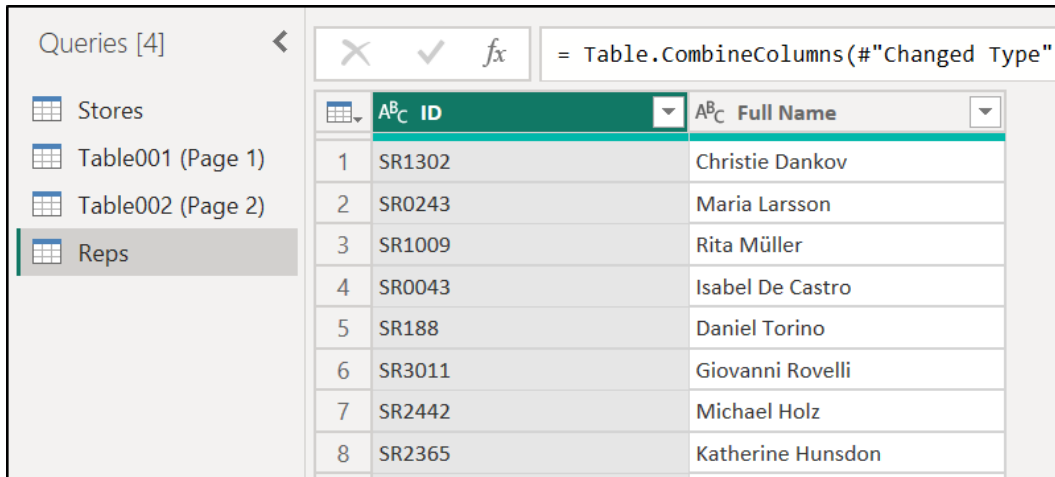
New column name (optional)
Full Name

OK Cancel

Figure 5.23: Specify the separator and name for the new column

The two columns are merged and have been replaced by a new **Full Name** column (Figure 5.24). This is because we clicked the **Merge Columns** button on the **Transform** tab.

There is also a **Merge Columns** button on the **Add Column** tab that would have kept the **First Name** and **Last Name** columns and created a third column for the new merged column. Please refer to the following figure:



Queries [4] <

Stores

Table001 (Page 1)

Table002 (Page 2)

Reps

✕ ✓ fx = Table.CombineColumns(#"Changed Type", ...)

ID	Full Name
1	Christie Dankov
2	Maria Larsson
3	Rita Müller
4	Isabel De Castro
5	Daniel Torino
6	Giovanni Rovelli
7	Michael Holz
8	Katherine Hunsdon

Figure 5.24: Two name columns replaced by the merged full name column

Data types and renaming columns

There are no data types to change or columns to rename at the end of this query. However, I wanted to keep this step here to build on the previous query and the importance of performing these checks, especially the column data types.

Importing data from multiple files in a folder

Staying in the Power Query Editor, our next source of data is a folder. We have five CSV files stored in a local folder. These files contain our sales transaction data. Each file contains the sales data for a different region.

We need to connect to the folder to import all files within it and combine them into a single table of sales transactions.

1. Click **Home** | **New Source**.
2. Click on the folder connector and click **Connect**.
3. Click **Browse** and locate the sales folder containing the five CSV files (*figure 5.25*). Alternatively, copy and paste the folder path into the box provided. Click **OK**.

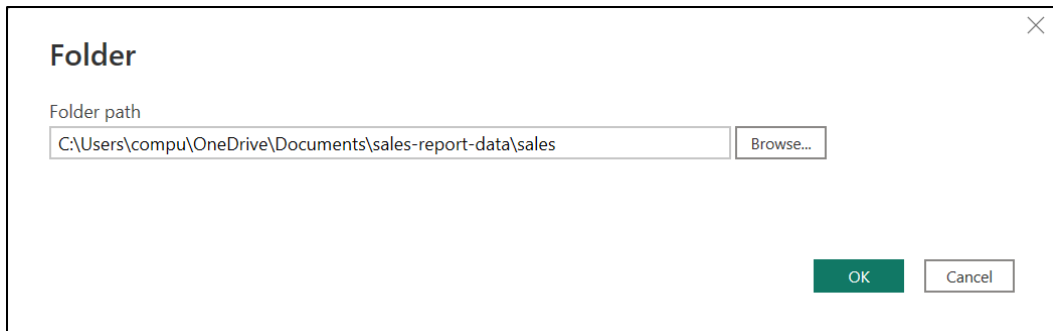


Figure 5.25: Browse to the folder you want to connect to

4. A window lists all the files from the folder, including some of their properties, such as the file extension and the date last accessed (*figure 5.26*). Click **Transform Data** to load this data into the Power Query Editor.

Note: We need to import all the files from this folder and combine them, so we could have clicked the Combine and Transform Data button to skip a step. However, I want to discuss the potential to filter out unrequired files from the folder before combining.

Content	Name	Extension	Date accessed	Date modified	Date created	Attributes	
Binary	east.csv	.csv	04/01/2023 16:39:22	23/08/2021 20:15:51	04/09/2022 07:47:53	Record	C:\Users\compu\O
Binary	north-east.csv	.csv	04/01/2023 16:39:34	23/08/2021 21:15:30	04/09/2022 07:47:53	Record	C:\Users\compu\O
Binary	north-west.csv	.csv	04/01/2023 16:39:34	23/08/2021 20:15:05	04/09/2022 07:47:53	Record	C:\Users\compu\O
Binary	south.csv	.csv	16/11/2022 05:38:53	23/08/2021 21:14:44	04/09/2022 07:47:53	Record	C:\Users\compu\O
Binary	west.csv	.csv	04/01/2023 16:39:34	24/12/2021 14:24:04	04/09/2022 07:47:53	Record	C:\Users\compu\O

Figure 5.26: List of files in the folder and their properties

With the list of files loaded into Power Query, we can filter out any files that are not required for our Power BI report. These could be files that we know are in the folder or a fail-safe method to protect the query from files that may accidentally be saved to the folder in the future.

For example, we could set a filter to only list the files that contain the words “east”, “west”, “south,” or “north” in their name.

Tip: Power Query is highly case-sensitive. So, when setting filters, ensure that the text you type is in the correct case.

A clever technique is to add a step before the filter that sets the case. This helps to ensure that the text in the column is the same case as the one that you specify in the filter criteria.

1. Click on the **Name** column and click **Transform** | **Format** | **lowercase** (figure 5.27).

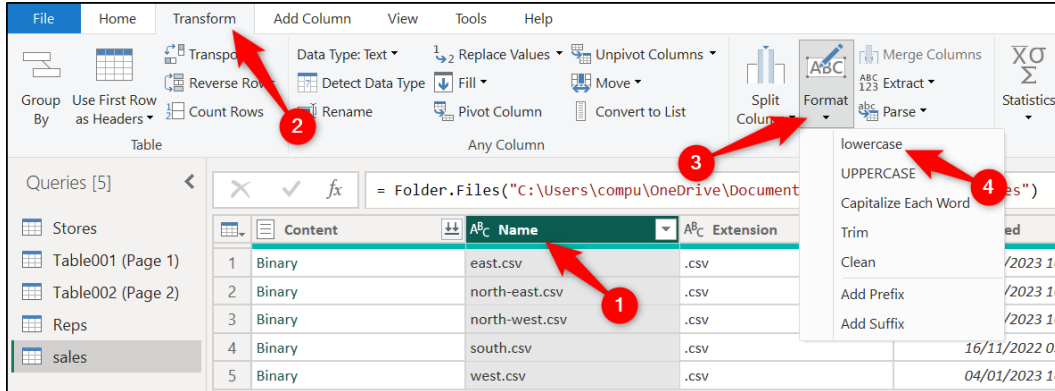


Figure 5.27: Formatting text in lowercase

2. Click the filter arrow for the **Name** column and click **Text Filters** | **Contains** (figure 5.28).

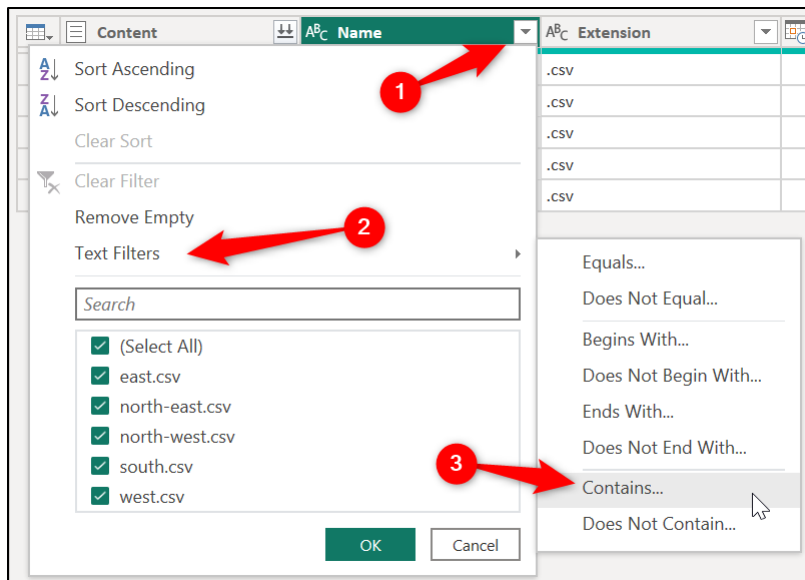


Figure 5.28: Setting a text contains filter on the Name column

3. In the **Filter Rows** window, click the **Advanced** option to enable the setting of more than two criterion.
4. Set the filter criteria as shown in figure 5.29. Click the **Add Clause** button to add the additional two clauses that we require. Ensure that all rows in the **And/Or** column are set to **Or** and that all values are entered in lowercase. Click **OK**.

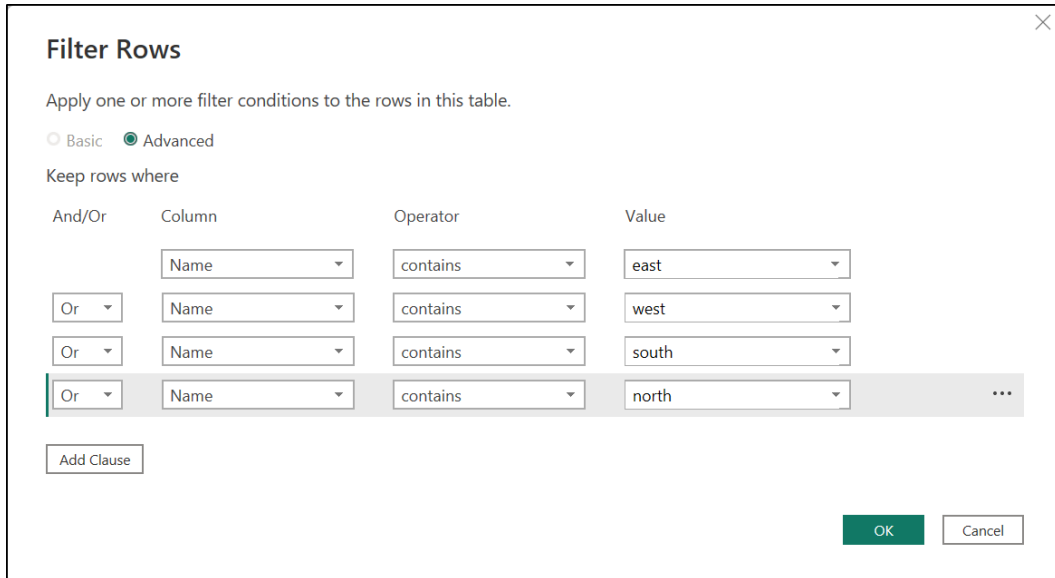


Figure 5.29: Setting multiple “contains” criteria to filter rows

Combine the data from all files into one table

We now want to extract the content from each of the five files and stack them into one table/query. This is simple, as Power Query will do the heavy lifting for you.

1. Click the **Combine Files** button in the header of the **Content** column (figure 5.30).

	Content	Name	Extension
1	Binary	east.csv	.csv
2	Binary	north-east.csv	.csv
3	Binary	north-west.csv	.csv
4	Binary	south.csv	.csv
5	Binary	west.csv	.csv

Figure 5.30: The Combine Files button

2. The **Combine Files** window opens. You are prompted for a **Sample File** (figure 5.31). Each of the five CSV files has the exact same headers and contains the same steps, so it is no problem which file is used as the sample. Leave it as the **First file** and click **OK**.

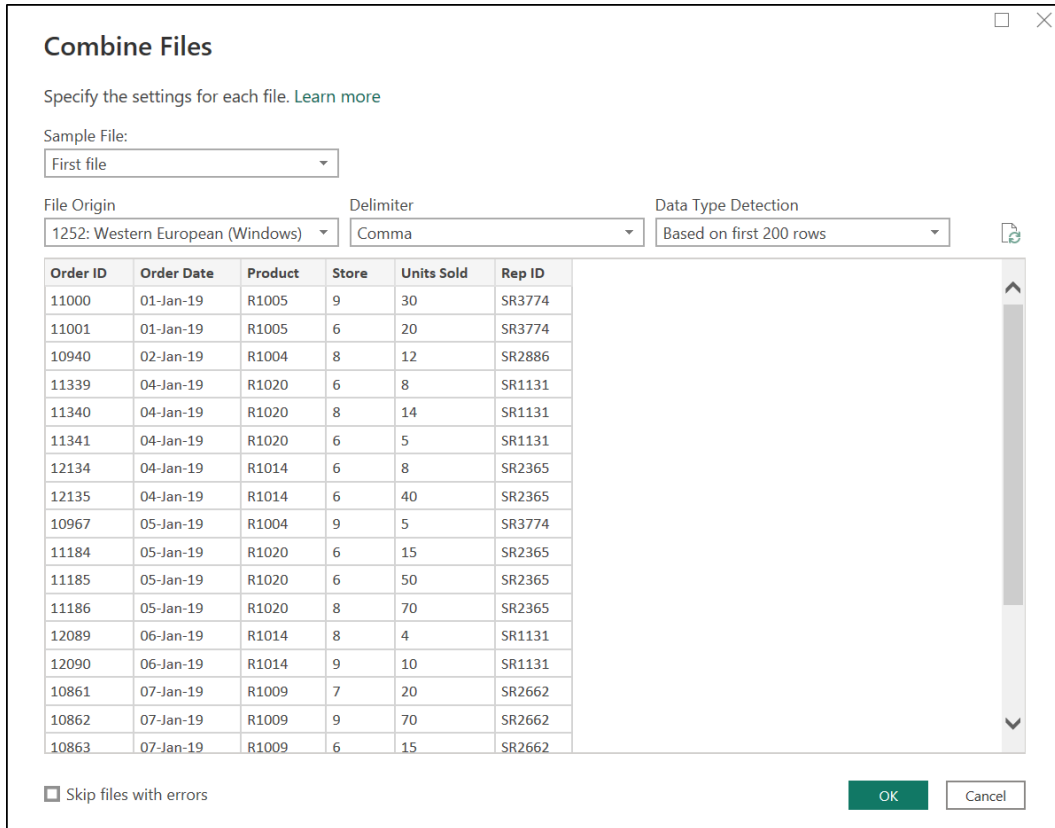


Figure 5.31: Specifying a sample file in the Combine Files window

The **sales** query now contains the data from all five regional transaction files. A **Source.Name** column has been added containing the name of the file that the data is from (figure 5.32).

Note: There are techniques to work around common issues, such as files with inconsistent header names. However, they are beyond the scope of this book. You can find solutions to these scenarios in the Power BI community. Click [Help | Community](#).

	Source.Name	Order ID	Order Date	Product
1	east.csv	11000	01/01/2019	R1005
2	east.csv	11001	01/01/2019	R1005
3	east.csv	10940	02/01/2019	R1004
4	east.csv	11339	04/01/2019	R1020
5	east.csv	11340	04/01/2019	R1020
6	east.csv	11341	04/01/2019	R1020
7	east.csv	12134	04/01/2019	R1014
8	east.csv	12135	04/01/2019	R1014
9	east.csv	10967	05/01/2019	R1004
10	east.csv	11184	05/01/2019	R1020
11	east.csv	11185	05/01/2019	R1020

Figure 5.32: Source.Name column added with the name of the file

Power Query created some helper queries, including a custom function, parameter, and the sample file. And applied a few steps to combine the files into one table/query for us.

Figure 5.33 shows the **Queries** pane with the helper queries and the combined **Sales** query. The **Applied Steps** is also shown. We have taken this opportunity to rename the query from **sales** to **Sales** for consistency in the names of our queries. Please refer to the following figure:

The screenshot displays the Power BI Desktop interface. On the left, the **Queries** pane shows a tree view with the following structure:

- Transform File f...
 - Helper Querie...
 - Parameter1 (...)
 - Sample File
 - Transform File
 - Transform Sa...
 - Other Queries [5]
 - Stores
 - Table001 (Pag...
 - Table002 (Pag...
 - Reps
 - Sales** (highlighted)

The main area shows a data table with columns: Source.Name, Rep ID, and others. The data rows are numbered 1 through 14. A red arrow points from the 'Sales' query in the Queries pane to the data table.

On the right, the **Query Settings** pane is open for the 'Sales' query. The **APPLIED STEPS** section is highlighted with a red box and contains the following steps:

- Source
- Filtered Rows
- Filtered Hidden Files1
- Invoke Custom Function1
- Renamed Columns1
- Removed Other Columns1
- Expanded Table Column1
- Changed Type** (highlighted)

Figure 5.33: Added helper queries and applied steps

We do not need to understand this process, as it is all performed by using the Power Query UI. However, as you progress your knowledge with Power Query, you will begin to learn more about parameters and custom functions and will be able to edit these queries if required.

Removing unwanted characters

We want to keep the **Source.Name** column as it helps us to identify which region the sale is from. However, we need to make improvements to the text, as it currently shows each file name, including its extension, for example, **east.csv**.

Let us begin by removing the extension from each region. As they are all CSV files, therefore, have the same extension, we will use the Replace Values tool of Power Query.

1. Click on the **Source.Name** column.
2. Click **Home | Replace Values**.
3. Type **".csv"** in the **Value to Find** box and leave the **Replace With** box empty (figure 5.34). Click **OK**.

Tip: If there were different file extensions such as **.xlsx**, **.xlsb**, and **csv**, then we could split the column at the period **"."** And then remove the column containing the extension.

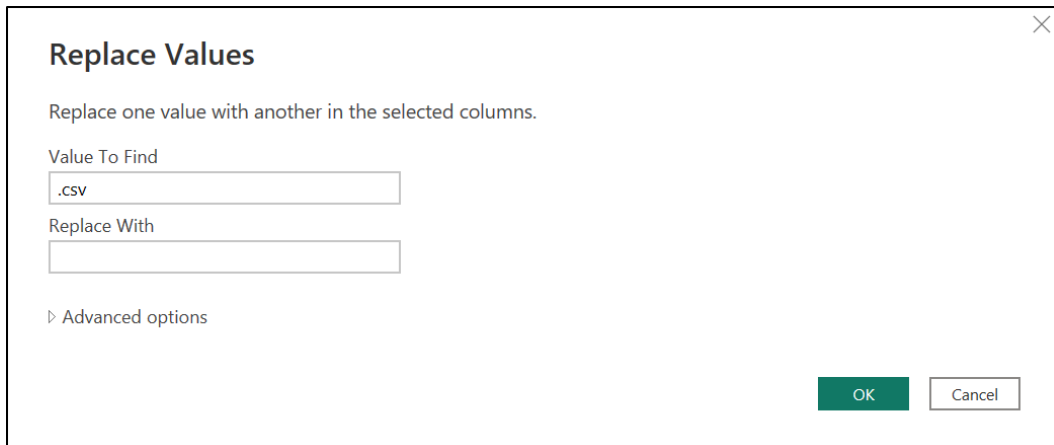


Figure 5.34: Removing the .csv values

There is another unwanted character to replace in the **Source.Name** column. However, it is difficult to notice this character because the only region we can see on the screen currently is the east.

The filter tool, in addition to performing the wonderful task of filtering values, is also brilliant for checking the values in a column.

1. Click the filter arrow on the **Source.Name** column.

The filter also only shows the *east* value as it looks at the first 1,000 rows only (*figure 5.35*). It states that the **List may be incomplete**. However, we can ask to see the complete list.

2. Click the **Load more** link.

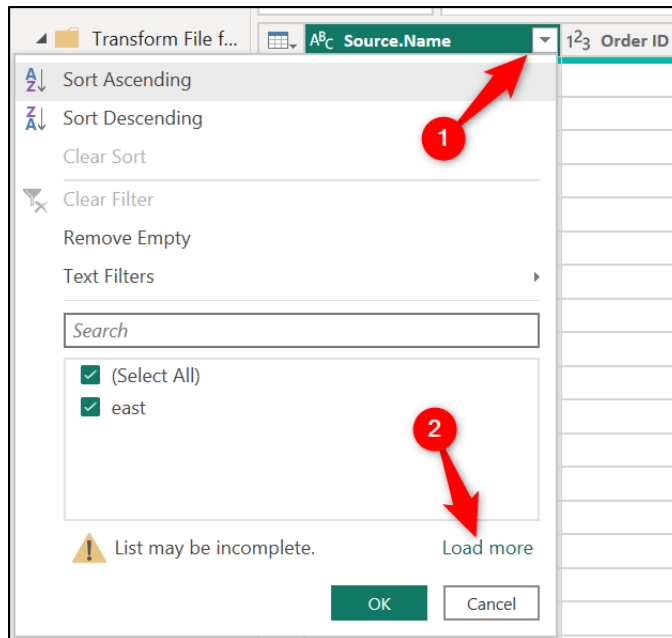


Figure 5.35: Load the complete list of values in filter

The complete list is displayed (*figure 5.36*). You can now see that there is a hyphen (-) character in the region names of the north-east and north-west. We will replace the hyphen with a space.

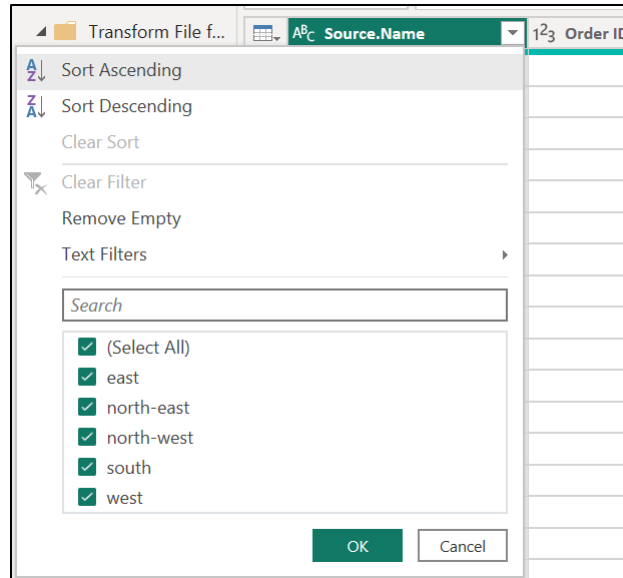


Figure 5.36: Hyphen characters in two of the region names

1. Click on the **Source.Name** column.
2. Click **Home | Replace Values**.
3. Type “-” in the **Value To Find** box and type a space “ ” in the **Replace With** box (figure 5.37). Click **OK**.

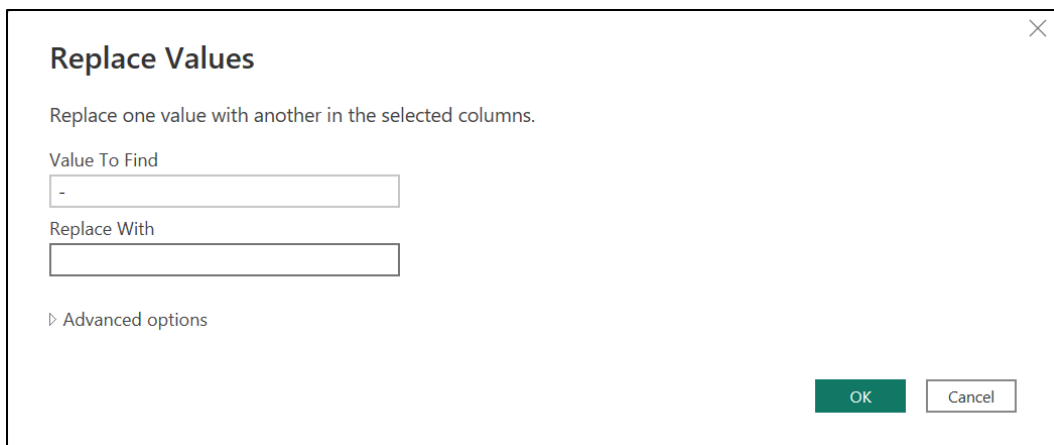


Figure 5.37: Replacing the hyphens with spaces

The filter can be used again to check the success of the replace values operation. Figure 5.38 shows that the hyphens were successfully replaced.

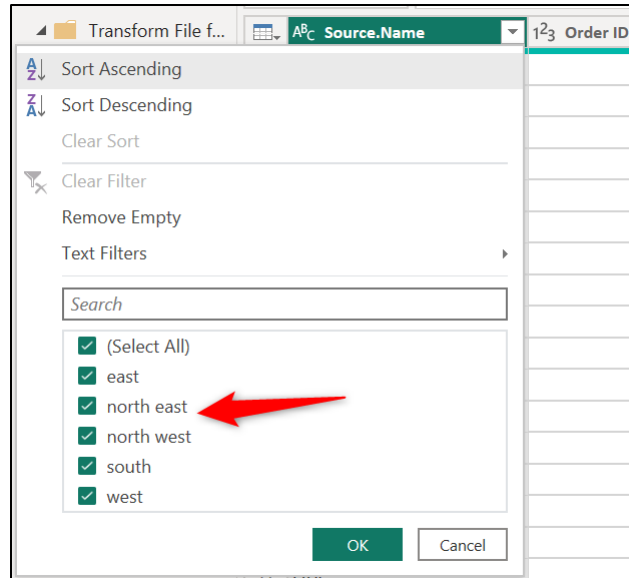


Figure 5.38: Using a filter to check that the replace values operation worked

Changing the case of text

Next, we will change the case of the text in the [Source.Name] column from lowercase to capitalize each word.

1. Click on the **Source.Name** column.
2. Click **Transform | Format | Capitalize Each Word** (figure 5.39).

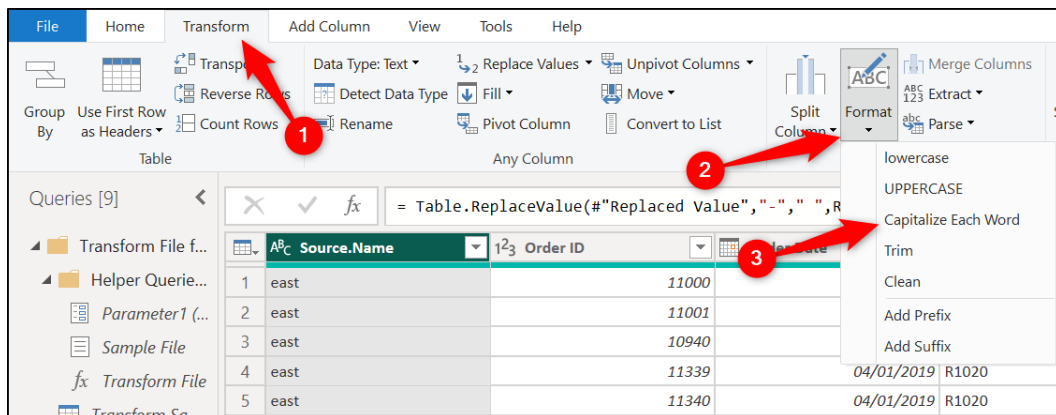


Figure 5.39: Capitalizing each word of the Source.Name column

Data types and renaming columns

We will now perform our final checks on the names of the columns and their data types.

There are a few columns that need renaming. Double-click on the headers and rename the following columns, as shown in *figure 5.40*.

- **Source.Name** to **Region**
- **Product** to **Product ID**
- **Store** to **Store ID**

There are no data type changes to make; however, it is always wise to check the data type of each column:

	Region	Order ID	Order Date	Product ID	Store ID	Units Sold	Rep ID
1	East	11000	01/01/2019	R1005		9	30 SR3774
2	East	11001	01/01/2019	R1005		6	20 SR3774
3	East	10940	02/01/2019	R1004		8	12 SR2886
4	East	11339	04/01/2019	R1020		6	8 SR1131
5	East	11340	04/01/2019	R1020		8	14 SR1131
6	East	11341	04/01/2019	R1020		6	5 SR1131
7	East	12134	04/01/2019	R1014		6	8 SR2365
8	East	12135	04/01/2019	R1014		6	40 SR2365
9	East	10967	05/01/2019	R1004		9	5 SR3774
10	East	11184	05/01/2019	R1020		6	15 SR2365
11	East	11185	05/01/2019	R1020		6	50 SR2365
12	East	11186	05/01/2019	R1020		8	70 SR2365

Figure 5.40: Completed column names and data types

Importing data from files stored on OneDrive/SharePoint

Our final query requires importing data from an Excel workbook stored on SharePoint/OneDrive. This Excel file contains data relating to the products that we sell.

When creating a connection to the document, in this case, an Excel file, we want to use the Web connector and provide the OneDrive file path. This will ensure that the connection works from other computers and for other users.

Connecting to the local file, as we did in the previous example, is also fine. However, it would only work on this computer.

So, before we start the Web connector, we need to find and take a copy of the OneDrive file path. There are a few methods to do this. Let us see three different methods.

The first method is to get the file path from SharePoint online.

1. From SharePoint online, locate the **products.xlsx** Excel workbook.
2. Click on the **products.xlsx** workbook to select it.
3. From the details pane on the right, click the Copy direct link button (*figure 5.41*):

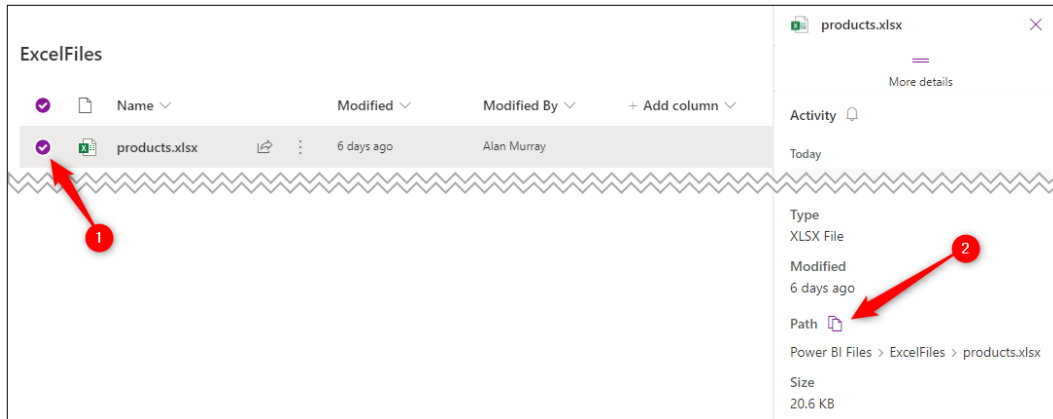


Figure 5.41: Getting the file path of the document from SharePoint online

The copied file path will look like the following. My **products.xlsx** file has been saved into a document library named ExcelFiles.

<https://computergaga.sharepoint.com/sites/PowerBIFiles/ExcelFiles/products.xlsx>

Note: If the details pane is not visible, click the information icon at the end of the toolbar in SharePoint online.

The next method is to copy the file path from OneDrive online.

1. From OneDrive online, locate the **products.xlsx** workbook.
2. Click on the workbook to select it.
3. Click the Open the details pane button (*figure 5.42*).
4. Click the Copy direct link button to copy the file path.

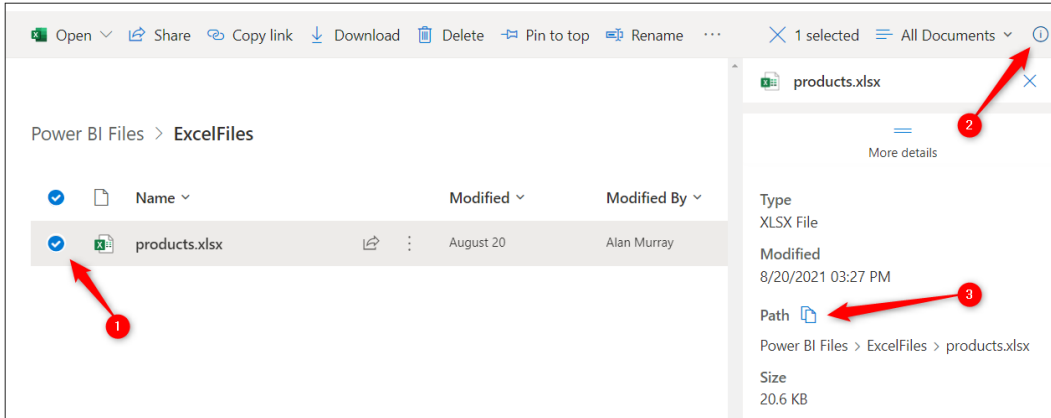


Figure 5.42: Copying the file path of a document from OneDrive online

The copied file path is the same as the one copied from SharePoint online.

The final method is to copy the path from within the Excel application. The path that is copied from here is slightly different to the previous methods, so it requires a minor edit before we can use it in the Web connector.

1. With the **products.xlsx** workbook open in Excel, click **File** | **Info**.
2. Click the **Copy path** button (figure 5.43).

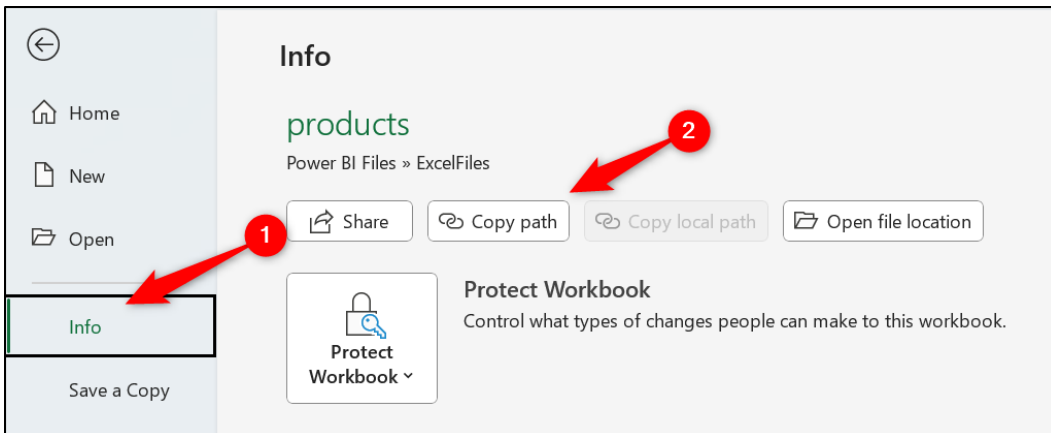


Figure 5.43: Copying the file path from within the Excel app

The copied file path will look like the following. Before it can be used in the Web connector, we need to remove the “?web=1” text from the end.

<https://computergaga.sharepoint.com/sites/PowerBIFiles/ExcelFiles/products.xlsx?web=1>

Now that we have the SharePoint/OneDrive file path, we can connect to this data source using the Web connector.

1. From within the Power Query Editor, click the **New Source list | Web**.
2. In the **From Web** window (figure 5.44), paste the OneDrive file path into the URL box. Click **OK**.

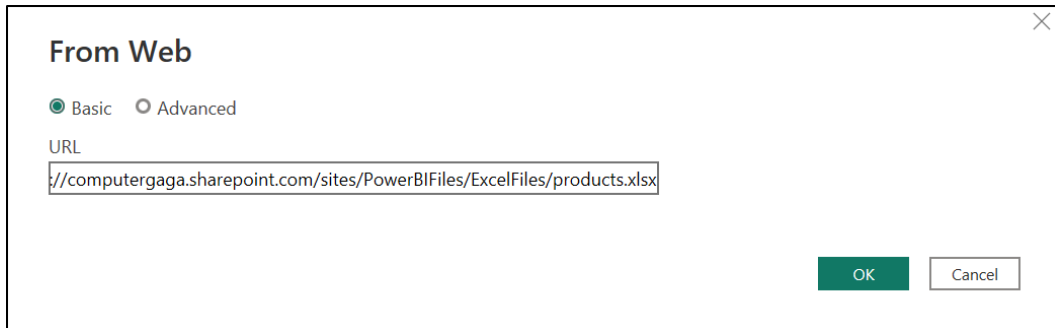


Figure 5.44: Pasting the OneDrive file path into the URL box of the Web connector

You may be prompted for how you want to access the Web content, especially if it is your first time connecting to this Web content in Power BI (figure 5.45).

There are five authentication methods:

- **Anonymous:** Used when the Web page does not require any credentials. For example, public sites such as Wikipedia.
- **Windows:** Used if the Web page requires your Windows credentials.
- **Basic:** Used if the Web page requires a simple username and password.
- **Web API:** Used if an API key is used to authenticate access to the Web resource.
- **Organizational account:** Used if the Web page requires organizational account credentials. For example, access to a SharePoint site.

I will use my organizational account to access the **products.xlsx** file that is stored on my Computergaga SharePoint site.

1. Click **Organizational account** (figure 5.45).
2. Click **Sign in** and follow the process of signing in using your credentials.
3. Select the level that you want to use these credentials for and click **Connect**.

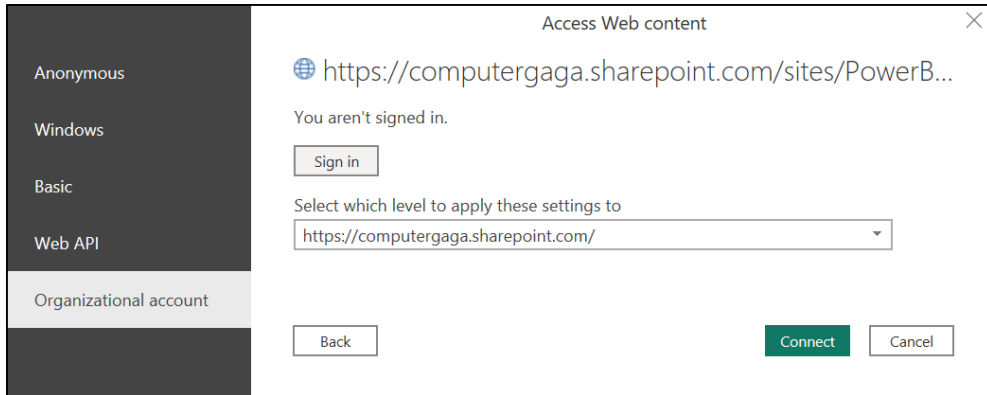


Figure 5.45: Signing into SharePoint with your account credentials

The Navigator window lists all the tables, worksheets, and defined names from the Excel workbook (*figure 5.46*). Our product data is split into three separate tables—Beverages, Cakes, and Food.

We want to import all the tables from the workbook. There are currently three tables, but this may change, so we will import everything from the workbook and filter out any data structure that is not a table.

1. Right-click on the folder (the Excel workbook) and click **Transform Data** (*figure 5.46*).

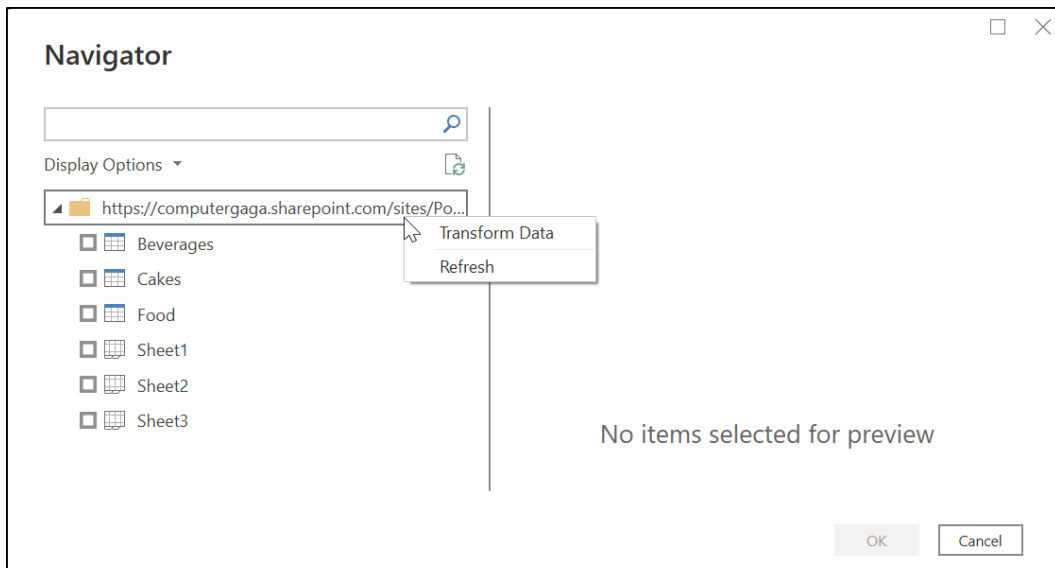


Figure 5.46: Loading all the objects of the workbook into Power Query

All the tables and worksheets are loaded into the Power Query Editor (*figure 5.47*).

Note: You may have noticed that when you click on the folder, the OK button is disabled. You can see this in *figure 5.46*. So, right-clicking is a neat trick to import everything from the workbook. Bear in mind functionality such as this may have changed at the time you read this book.

	ABC Name	Data	ABC Item	ABC Kind	Hidden
1	Sheet3	Table	Sheet3	Sheet	FALSE
2	Sheet2	Table	Sheet2	Sheet	FALSE
3	Sheet1	Table	Sheet1	Sheet	FALSE
4	Beverages	Table	Beverages	Table	FALSE
5	Food	Table	Food	Table	FALSE
6	Cakes	Table	Cakes	Table	FALSE

Figure 5.47: Tables and worksheets loaded to the Power Query Editor

Combine all tables into one

We want to combine the data from all the tables into one table, but before we do this, we must first filter out the worksheets. These sheets contain the same data as the tables but in a less structured format. The tables are preferable.

1. Click the filter arrow for the **Kind** column.
2. Uncheck the **Sheet** box and click **OK** (*figure 5.48*).

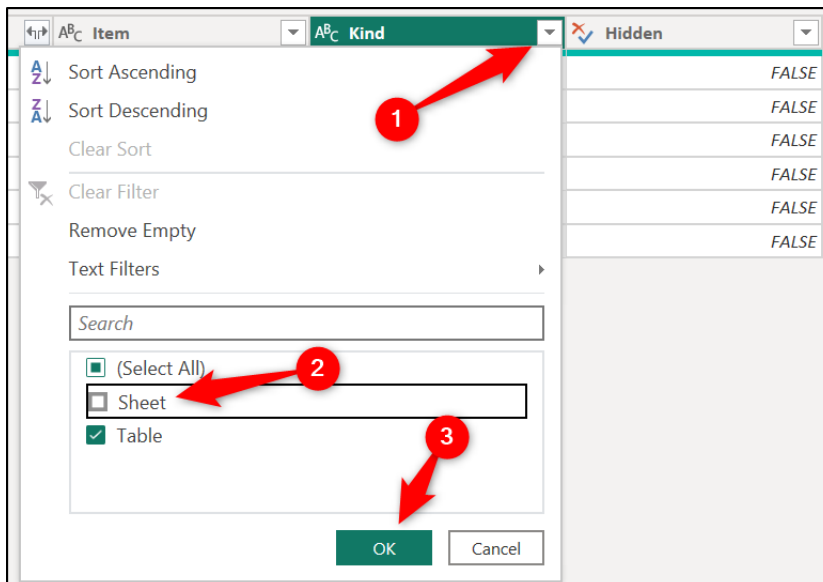


Figure 5.48: Filtering out the sheets

3. Select both the **Name** and **Data** columns, right-click on the **Data** column and click **Remove Other Columns** (figure 5.49).

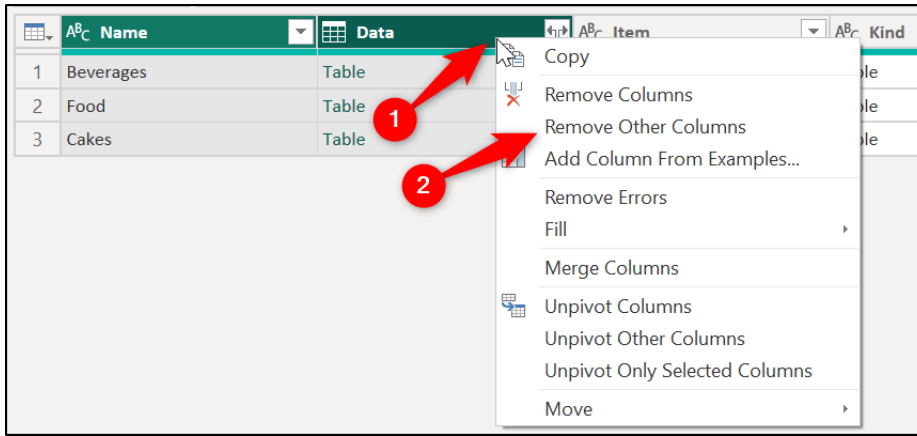


Figure 5.49: Removing other columns that are not required

This removes columns that are not required in the final combined table.

Now, we will combine the data from all the tables into one.

Tip: You can click in the cell to the right of the word “Table” (do not click on the word “table”) to see a preview of its data below.

1. Click the **Expand data** button in the **Data** column header (figure 5.50).
2. From the list, you can choose which columns to extract from the tables. In this example, we want all of them. So, leave all the column boxes checked.
3. Uncheck the **Use original column name as prefix** box. Click **OK**.

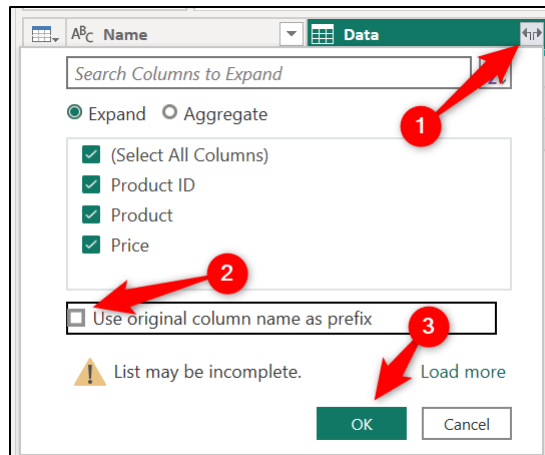


Figure 5.50: Expanding the data from the tables

The data from all the tables is combined into one. *Figure 5.51* shows the first 10 rows of the combined table:

	ABC Name	ABC 123 Product ID	ABC 123 Product	ABC 123 Price
1	Beverages	R1001	Orange Juice	1.2
2	Beverages	R1002	Coffee	2.4
3	Beverages	R1003	Tea	1.5
4	Beverages	R1004	Hot Chocolate	1.8
5	Beverages	R1005	Beer	3.9
6	Beverages	R1006	Wine	6.6
7	Beverages	R1007	Water	1
8	Food	R1008	Sandwich	3.3
9	Food	R1009	Samosa	2.5
10	Food	R1010	Baguette	2.8

Figure 5.51: Combined data from all the tables.

Data types and renaming columns

As usual, our final steps are to check the names and data types of the columns of the table and make any necessary changes.

The **Name** column contains the name of the tables that the data has come from. These are the category that the product is assigned to.

1. Double-click on the **Name** column header and name it **Category** (*figure 5.52*).

While we are renaming objects, the query also requires renaming. It is currently using the OneDrive path as its name.

2. Right-click the query name in the **Queries** pane and click **Rename**. Name the query **Products**.

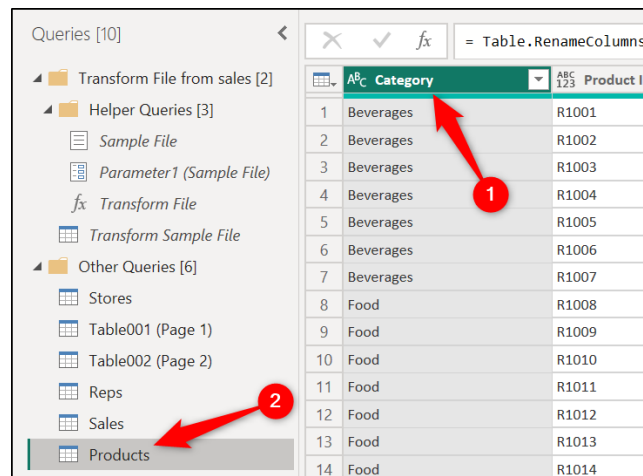


Figure 5.52: Renamed category column and Products query

The **Product ID**, **Product**, and **Price** columns all require their data types to be changed.

3. Change the data type of the **Product ID** and **Product** columns to Text.
4. Change the **Price** column data type to a Fixed decimal number. *Figure 5.53* shows this being done by clicking the data type button in the column header.

Product ID	Product	Price
R1001	Orange Juice	1.2
R1002	Coffee	\$
R1003	Tea	123
R1004	Hot Chocolate	%
R1005	Beer	Date/Time
R1006	Wine	Date
R1007	Water	Time
R1008	Sandwich	Date/Time/Timezone
R1009	Samosa	Duration
R1010	Baguette	Text
R1011	Soup	True/False
R1012	Jacket Potato	Binary
R1013	Cornish Pasty	Using Locale...

Figure 5.53: Changing the price column to a Fixed decimal number data type

All the data has now been connected to and transformed. It is time to load it to Power BI for us to begin modeling further.

Click **Home | Close & Apply**.

Figure 5.54 shows all the tables loaded into Power BI Desktop and shown in the Data pane of the report view.

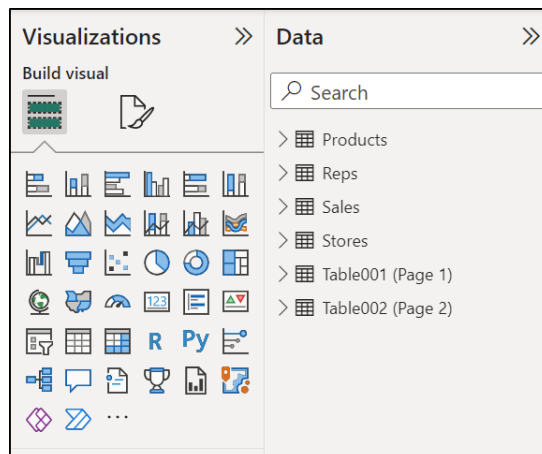


Figure 5.54: All tables loaded into Power BI

Managing data source settings

Power BI provides an easy way for you to manage your data source settings. For example, to change the location of a file or edit the sign-in credentials for data sources such as SharePoint.

Note: This is an important topic and especially prevalent in this book. If you follow along using the provided Power BI files at each chapter, their file paths will be different to those on your computer.

To open the data source settings, click **Home** | **Transform data** list | **Data source settings** (figure 5.55).

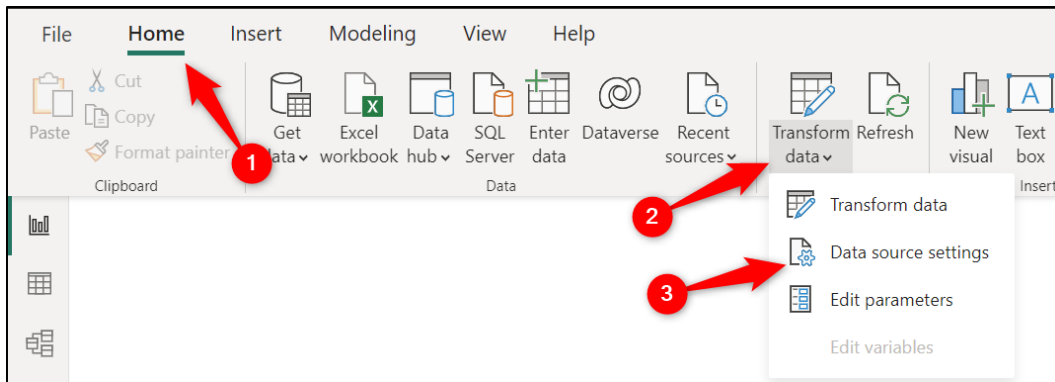


Figure 5.55: Accessing the data source settings

Data sources in the current file

The **Data source settings** window lists all the data sources in the current file (figure 5.56).

This is the default option, but you can switch between the **Data sources in current file** and **Global permissions** using the option buttons near the top of the window.

Let us say we needed to change the folder path of our **sales** folder. Maybe a colleague has downloaded the CSV files to a different folder.

1. Click on the sales folder data source.
2. Click **Change Source** (figure 5.56).

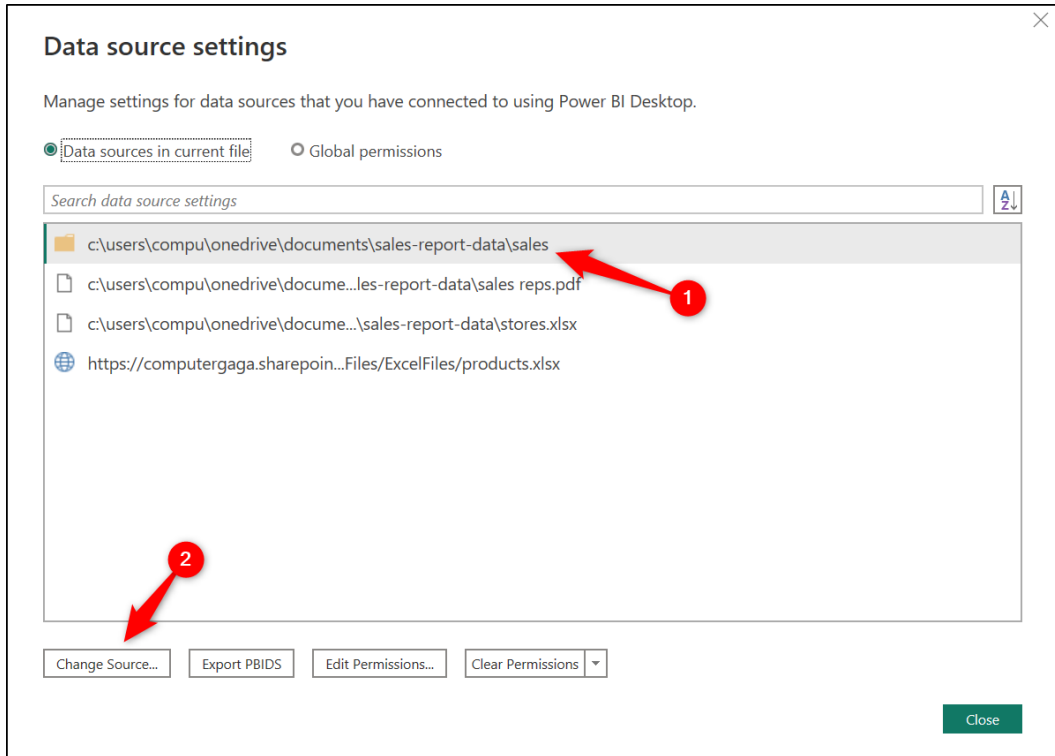


Figure 5.56: Changing the location of a data source

3. Either edit the folder path from the Folder path box or click the **Browse** button to navigate to and select the new folder path (*figure 5.57*). Click **OK**.

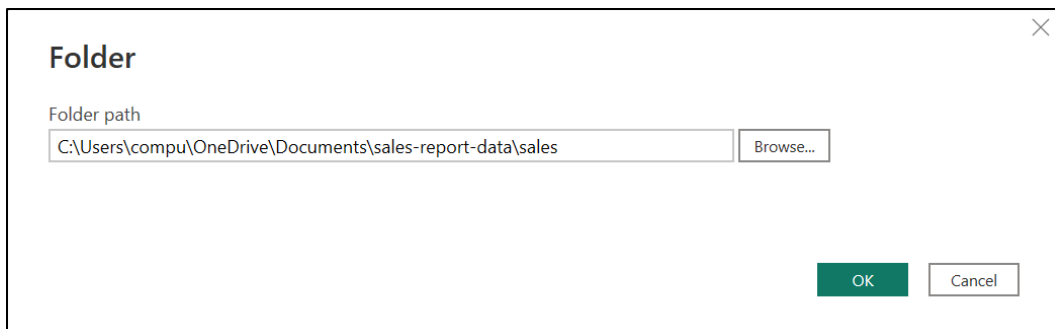


Figure 5.57: Edit the folder path or browse to its new location

Changing global permissions

You can also edit the global permissions of Power BI in this window. For example, different websites, apps, and OneDrive permissions you may have created in Power BI.

Click the **Global permissions** option in the window to view all your permissions (*figure 5.58*).

You can see the permissions for accessing SharePoint and the imdb.com website from the previous chapter in the window.

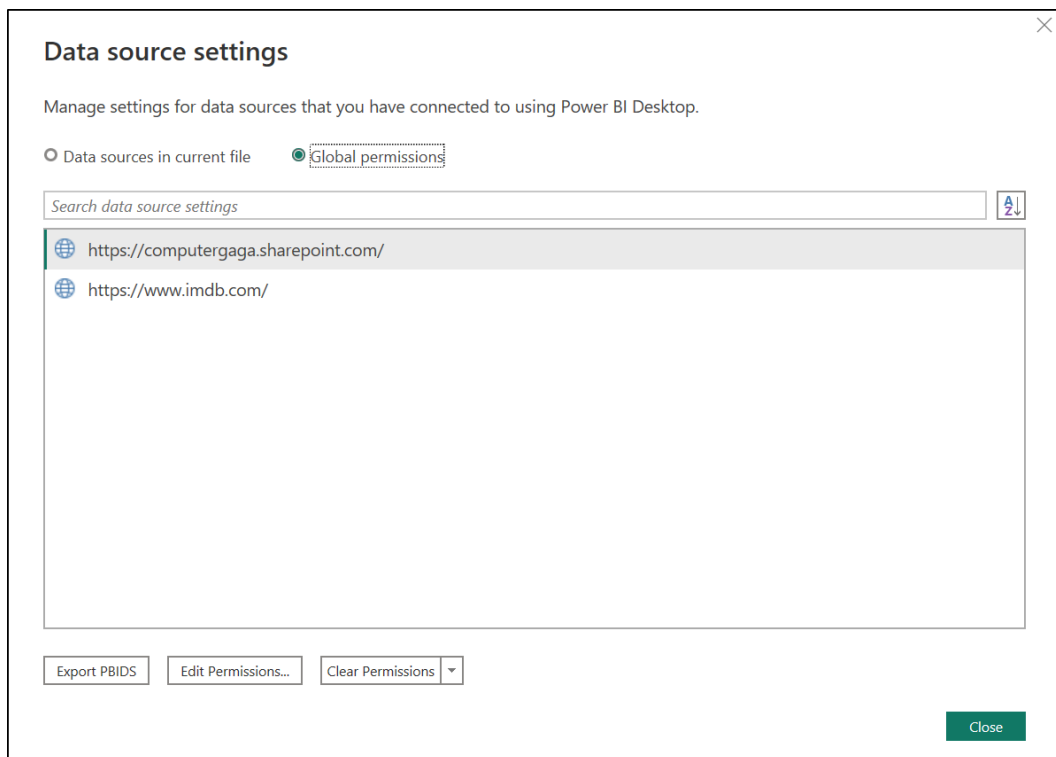


Figure 5.58: Global permissions for your Power BI connections

To edit the permissions for SharePoint,

1. Click on the SharePoint/OneDrive data source.
2. Click **Edit Permissions**.
3. From the **Edit Permissions** window (*figure 5.59*), click **Edit** to change your access credentials.

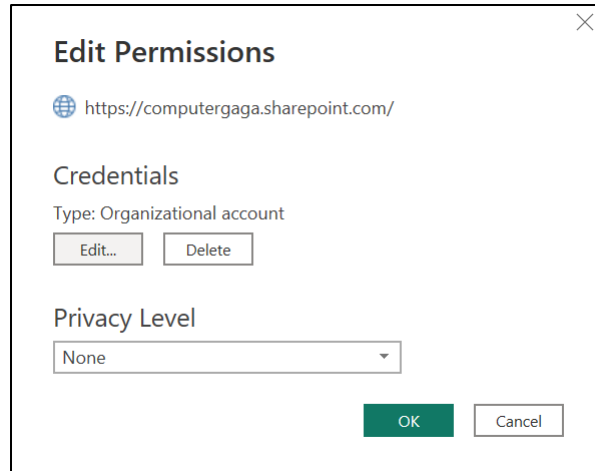


Figure 5.59: Editing the credentials for a SharePoint connection

4. Click **Sign in as different user** and follow the process to change the account or login with a different password. Click **Save**:

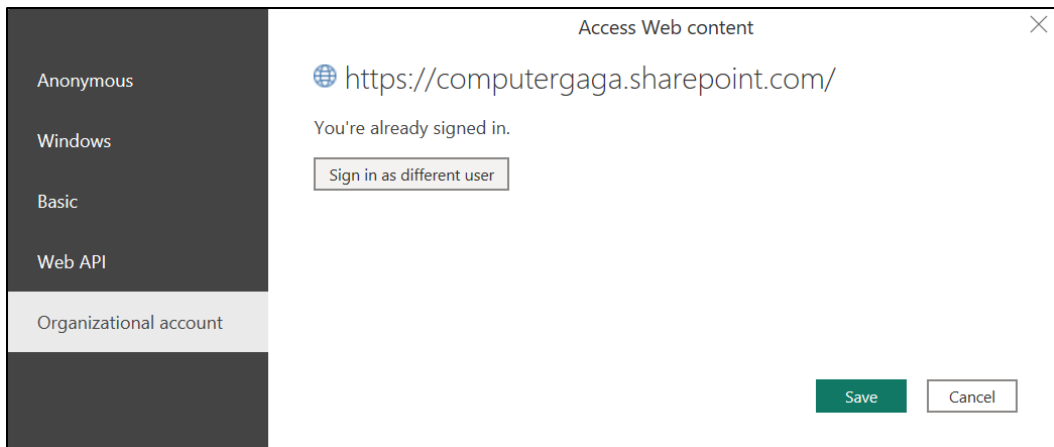


Figure 5.60: Sign in as a different user to change the login credentials

Changing source in the Power Query Editor

As an alternative to using the data source settings, you can edit the data source within the Power Query editor.

1. From any view of the Power BI window, click **Home** | **Transform Data** to open the Power Query Editor.

Note: You can also access the data source settings from the Power Query Editor window by clicking Home | Data source settings.

Let us imagine that the **Stores.xlsx** file that we imported earlier has moved location, and we need to re-direct our connection to the new file path.

2. Click the **Stores** query in the **Queries** pane to select it (figure 5.61).

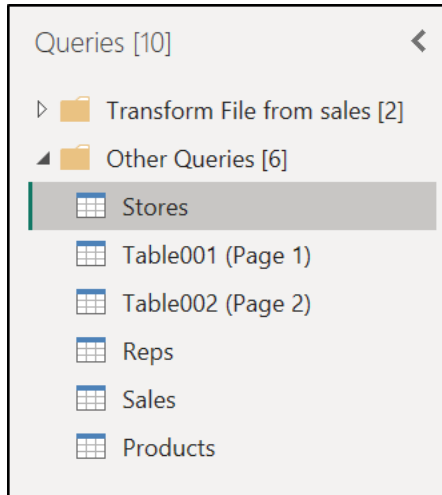


Figure 5.61: Selecting the query to change from the Queries pane

3. Click the **Source** step in the Applied Steps.
4. Edit the file path or paste the new file path in place of the old one in the Formula bar (figure 5.62).

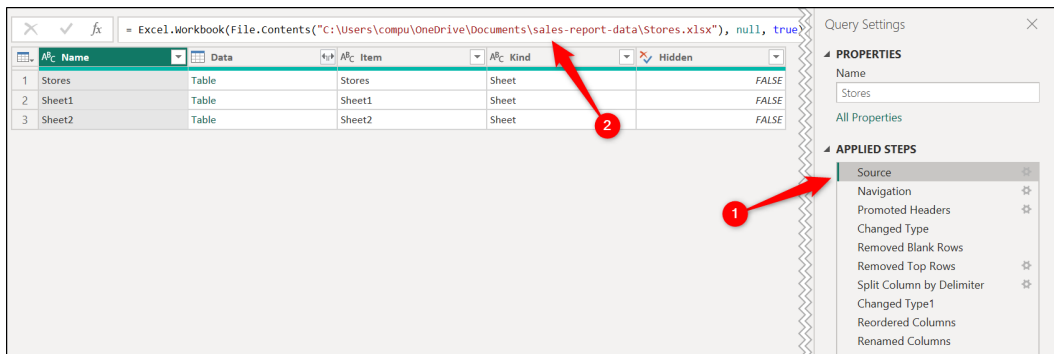


Figure 5.62: Changing the file path of a connection in the Formula bar

Alternatively, click the gear icon on the **Source** step and browse to the new file path using a window. Click **OK**.

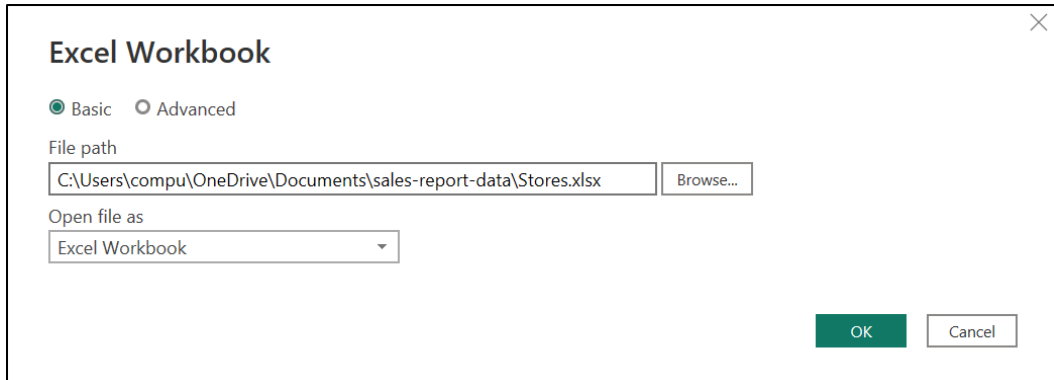


Figure 5.63: Changing the file path of a connection using the UI

5. Click **Home | Close & Apply** to load the queries back to Power BI.

Note: You can edit any step of a query at any time by coming back to the Power Query Editor. Some steps offer a gear icon to make a change using the UI, and others require editing in the Formula bar.

Conclusion

In this chapter, we learnt how to connect to multiple different data sources using Power Query. This included multiple files from a folder, files stored on OneDrive or SharePoint and PDF files.

We performed simple transformations using the Power Query UI to prepare the data for analysis in Power BI.

In the upcoming chapter, we explore more data transformations possible in Power Query. This includes performing merge and append queries and working with international number formats and calculated columns.

Questions

Here are some questions to test what you have learnt in this chapter:

1. Which of the following are different types of storage modes in Power BI?
 - a. DirectQuery
 - b. Import
 - c. Dual
 - d. All of the above.

2. Power Query is case-sensitive. You must ensure when setting filters to match the correct case.
 - a. True
 - b. False
3. What is the correct path to follow to change the case of a column in Power Query?
 - a. Click on the column > Transform > Format
 - b. Click on the column > Home > Change Case
 - c. Click on the column > Home > Format
 - d. Click on the column > View > Text > Change Case
4. Which is the correct connector to use when importing data from a file on OneDrive or SharePoint?
 - a. Excel Workbook
 - b. Text/CSV
 - c. SQL Server
 - d. Web.

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



CHAPTER 6

More Data Transformations

Introduction

In the previous two chapters, we have used Power Query to perform a variety of data transformations to prepare our data for analysis. These include splitting columns, changing the case of text, replacing values, and appending queries.

In this chapter, we explore further data transformations that I feel deserve some coverage in this book. These transformations are incredibly useful. It is important that a user of Power BI is familiar with these functionalities.

We will start by looking at merge queries. These queries have multiple uses. We will outline the different join types and then see two types of merge queries in action.

Next, we will explore a few date transformations in Power Query. Dates can appear in all shapes and sizes, so we will see a few examples of how to handle different date formats that you may receive when importing data from sources such as Excel and CSV files.

Then, we will look at performing different calculations in Power Query, including conditional columns, rounding values, and some date calculations. And finally, we see unpivoting columns. This is a favorite tool for many Power Query users.

Structure

In this chapter, we will cover the following topics:

- Merge queries
- Date transformations
- Calculations in Power Query
- Unpivoting columns

Objectives

After reading this chapter, you will be able to perform some of the most important transformations in Power Query, including merge queries, unpivoting columns, and adding calculated columns to a table.

Following on from the previous two chapters, this concludes the Power Query part of this book. By the end of this chapter, you will have a detailed understanding of the role that Power Query plays and how to use it to work with data and prepare it for analysis in Power BI.

Merge queries

File: `merge-queries.pbix`

Merge queries are a very powerful transformation in Power BI. There are many uses of merge queries and six different join kinds to cater to those needs.

We saw an example of an append query in the previous chapter to stack the data rows from one query to another. Although append queries are used to add rows to a query/table, Merge Queries are typically used to add columns from one query to another.

In Power BI, we are often getting data from different sources culminating in a model with many queries. Merge Queries is a useful tool to clean our model by combining the data from different queries in Power Query.

The different join kinds

There are six different types of join available when performing a merge query. These are known as Join Kinds and are used to specify how we would like to join the two queries.

Each Join Kind will return a different result, so it is important to become familiar with the different types of joins and their behavior.

Figure 6.1 shows the list of different Join Kinds in the Merge window. When merging queries, the first query/table is referred to as the Left table, and the second table is referred to as the Right table. Please refer to the following figure:

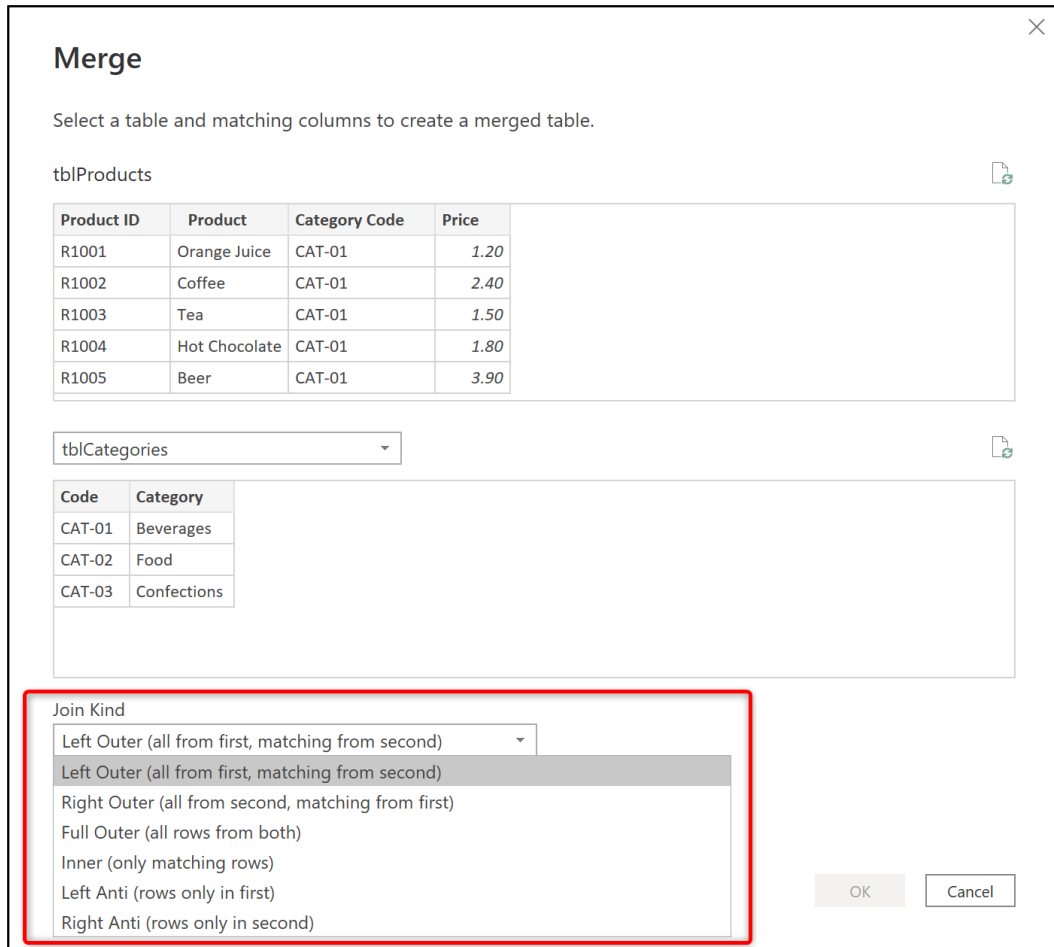


Figure 6.1: Join kinds in the Merge window

The following is an explanation of the six different join types.

1. **Left Outer:** All rows from the first table and only those that match from the second. This type of merge is often referred to as a typical lookup that you may perform in Excel. This is the default and most used type of merge. It is used to add columns to a table from another table. If no match is found in the second table, a null value is returned.

2. **Right Outer:** Returns all rows from the second table regardless of whether there is a match in the first. The rows where there is no match in the first table will return null values. This merge is a reverse of the first join kind.
3. **Full Outer:** The merge will return all rows from both the first and second tables matching the rows that it can. This ensures that all records from both tables are returned.
4. **Inner:** This merge type only returns the matching rows (the records that appear in both tables). Any rows without a match in the first or second tables will not be returned.
5. **Left Anti:** The Anti merge types return the rows that appear in one table and not the other. For the Left Anti, this means it returns all rows in the first table that do not have a match in the second table. The columns returned for the second table will all contain null values due to the nature of this merge.
6. **Right Anti:** The Right Anti is the reverse of the Left Anti join type. It will return only the rows that appear in the second table and not the first. The columns returned for the first table will all contain null values.

Let us look at two merge query examples.

Left Outer Join—adding a column to a table

For the first example, we will merge the **tblProducts** and **tblCategories** tables in the **merge-queries.pbix** file provided.

A snapshot of the **tblProducts** table is shown in *figure 6.2*. The table contains information about products that includes a **Category Code** column. This column uniquely identifies the different categories that a product is assigned to.

Product ID	Product	Category Code	Price
R1010	Baguette	CAT-02	£2.8
R1005	Beer	CAT-01	£3.9
R1015	Blueberry Muffin	CAT-03	£1.4
R1019	Caramel Shortbread	CAT-03	£2.2
R1016	Chocolate Chip Muffin	CAT-03	£1.4
R1002	Coffee	CAT-01	£2.4
R1013	Cornish Pasty	CAT-02	£3.4

Figure 6.2: tblProducts table with category code column

Figure 6.3 shows the **tblCategories** table. It contains three different categories. The **Code** column has data that contains matches to rows in the **Category Code** column of **tblProducts**.

Code	Category
CAT-01	Beverages
CAT-02	Food
CAT-03	Confections

Figure 6.3: *tblCategories* table with three product categories

We will perform a merge query using the default Left Outer join merge type to add the category name information to the **tblProducts** table.

1. Click the **tblProducts** table in the Queries pane to select it.
2. Click **Home | Merge Queries** (figure 6.4):

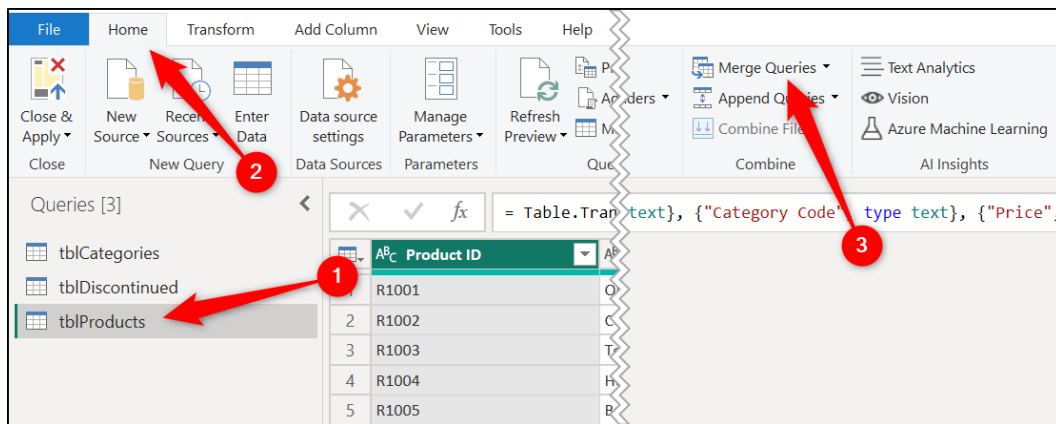


Figure 6.4: Starting a merge query from the *tblProducts* table

When merging queries, there is a list arrow on the **Merge Queries** button that provides the options to merge to the existing query or merge to a new query (figure 6.5).

In this example, we will merge to the existing **tblProducts** query to add a column to that query.

If we choose to merge to a new query, it will create a new query that references the **tblProducts** query as its source. **tblProducts** would remain unchanged.

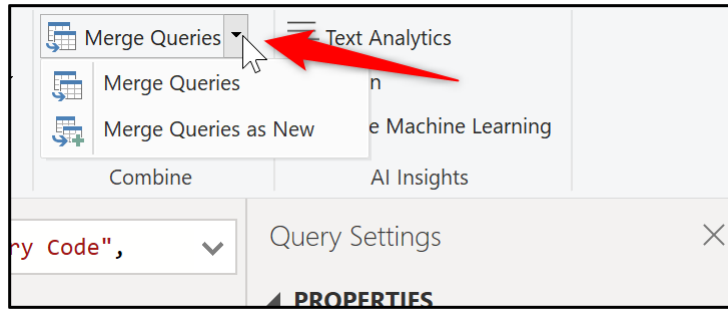


Figure 6.5: Merge to the existing query or a new query

In the *Merge* window (figure 6.6), the **tblProducts** query is already selected as the first table.

1. Click the list arrow and select **tblCategories** as the second table for the merge operation.
2. To select the matching columns, click the **Category Code** column in the first table and then the **Code** column in the second table.

The columns highlight in grey to indicate their selection, and a message appears at the bottom of the window to show how successful the matching process was.

In figure 6.6, the message says that it successfully matched 20 of 20 rows. So, all rows in the first table have a matching value in the second table. Please refer to the following figure:

Note: You can select multiple matching columns when merging tables. Simply select the matching column pairs in order whilst holding the Ctrl key on the keyboard.

1. Click **OK**.

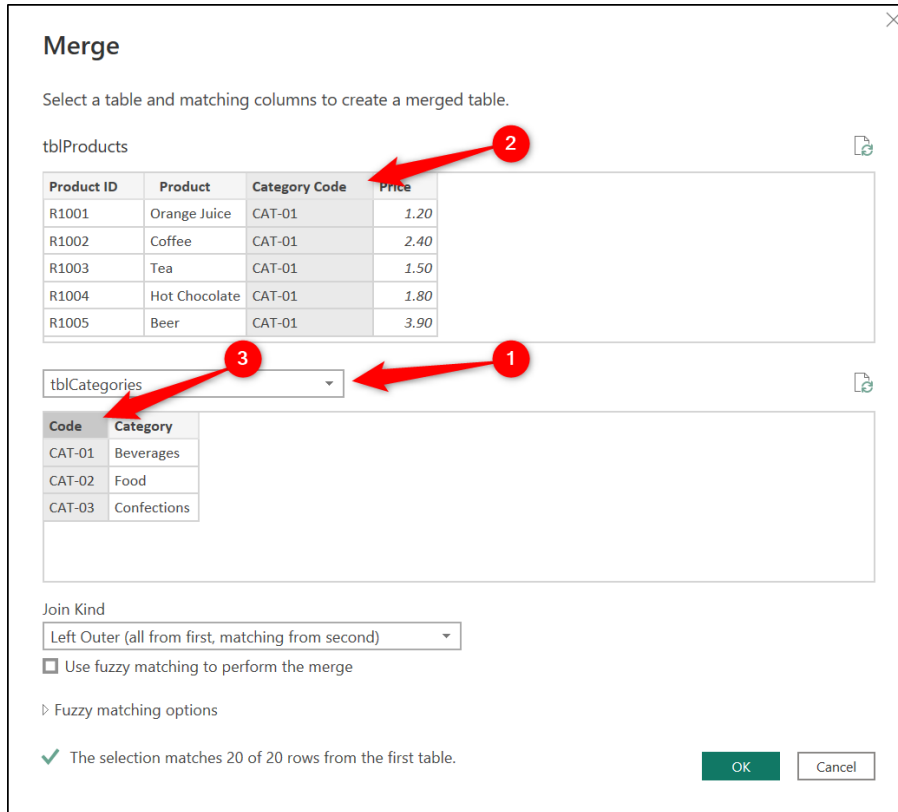


Figure 6.6: Merging two queries with a left outer join

Figure 6.7 shows the result of the merge. A **tblCategories** sub-table has been added in a column to the end of the **tblProducts** query. And the **Merged Queries** step has been added to the **Applied Steps** list.

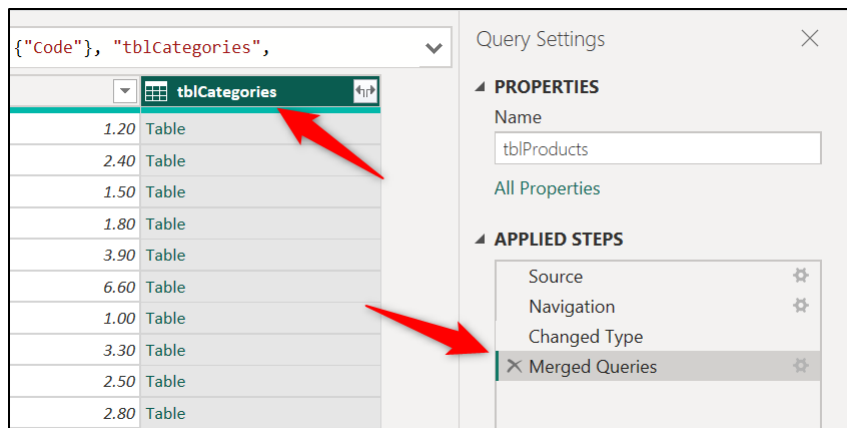


Figure 6.7: Merged Queries step and new column added to the tblProducts query

We now need to expand the table to choose the columns that we want to add to the **tblProducts** table. In this example, we only want the **Category** column, but you can include as many columns as required.

1. Click the double-arrow button next to the **tblCategories** column name (figure 6.8).
2. Uncheck the **Use original column name as prefix** box.
3. Uncheck the **Code** box, as we only want to add the Category column.
4. Click **OK**.

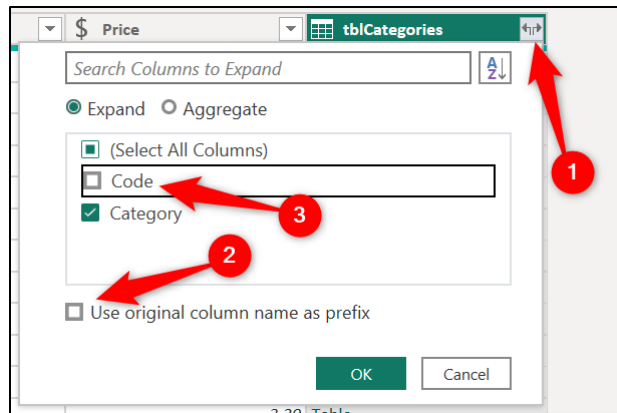


Figure 6.8: Expanding the table to choose the columns to merge

The **Category** column is added to the existing **tblProducts** query, and the **Expanded tblCategories** step is added to the **Applied Steps** list (figure 6.9).

Tip: The expanded **tblCategories** applied step has a gear icon. So, later, you can easily re-visit the step and change the columns that are included in the merge operation.

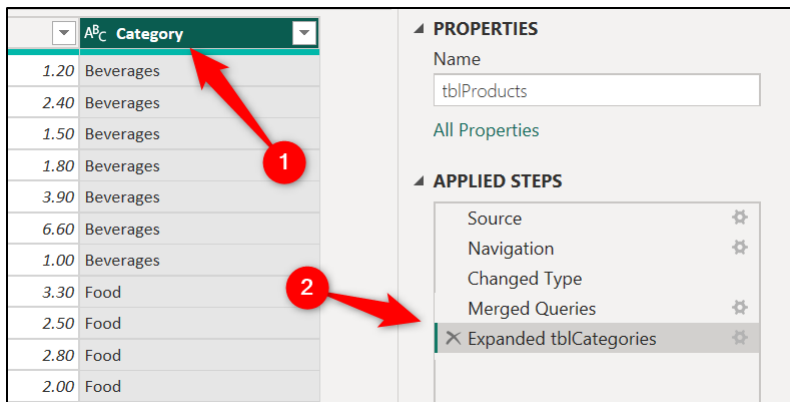


Figure 6.9: Category column added to the existing **tblProducts** table

With the **Category** column added, we no longer require the **Category Code** column in **tblProducts**. It has fulfilled its purpose of creating a match between the rows of the two tables.

5. Right-click on the **Category Code** column header and click **Remove**.

Figure 6.10 shows the **Removed Columns** step added to the **Applied Steps** list. The act of removing the column used to match the two queries can be confusing when you first start using merge queries. Many expect that it will break the merge.

However, because this step occurs after the *Merged Queries* step, this is perfectly fine. And we only want to load the tables and columns to Power BI that we are going to use from that point on. So, removing the column is good practice.

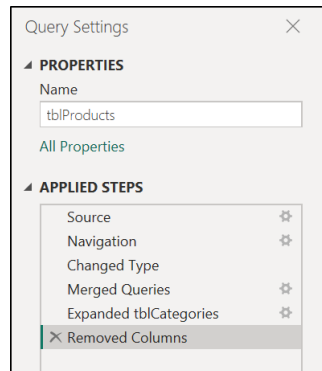


Figure 6.10: Removed columns step is performed after the merge operation

In figure 6.11, you can see the completed merged table:

	ABC Product ID	ABC Product	\$ Price	ABC Category
1	R1001	Orange Juice	1.20	Beverages
2	R1002	Coffee	2.40	Beverages
3	R1003	Tea	1.50	Beverages
4	R1004	Hot Chocolate	1.80	Beverages
5	R1005	Beer	3.90	Beverages
6	R1006	Wine	6.60	Beverages
7	R1007	Water	1.00	Beverages
8	R1008	Sandwich	3.30	Food

Figure 6.11: *tblProducts* with the merge operation complete

Left Anti Join—removing rows from a table

For a second example of the use of merge queries, we will see the Left Anti Join type.

Continuing with the **tblProducts** query from the previous merge, we will merge it with a query named **tblDiscontinued** to remove the product rows where that product is no longer sold.

Figure 6.12 shows the **tblDiscontinued** query. It contains the two products that are no longer sold, and we want them removed from the **tblProducts** query.

	Product ID	Product Name
1	R1013	Cornish Pasty
2	R1018	Flapjack

Figure 6.12: Query with products that have been discontinued

1. Click on the **tblProducts** query to select it.
2. Click **Home | Merge Queries**.
3. In the **Merge** window (figure 6.13), click the list arrow and select **tblDiscontinued** for the second table.
4. Click on the **Product ID** column in the first table, then click the **Product ID** column in the second table to specify them as the matching columns.
5. Click on the **Join Kind** list arrow and click **Left Anti**.

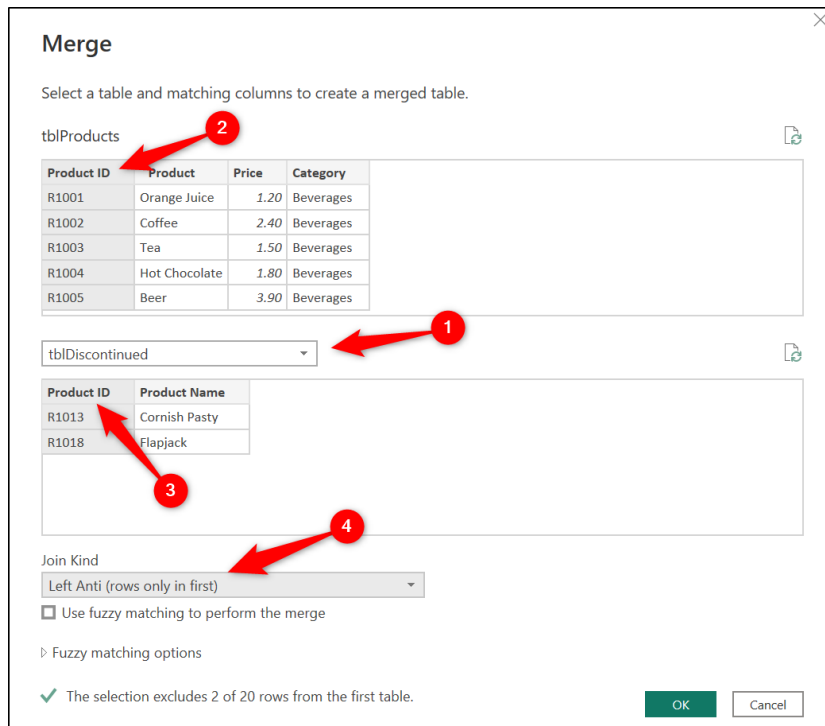


Figure 6.13: Selecting the matching columns and the Left Anti join kind

A message is shown at the bottom of the window stating that **The selection excludes 2 of 20 rows from the first table.**

- Click **OK**.

Figure 6.14 shows the result of the Left Anti merge operation. The **tblDiscontinued** sub-table is added, and only 18 rows are returned in the query:

	APC Product ID	APC Product	\$ Price	APC Category	tblDiscontinued
1	R1001	Orange Juice	1.20	Beverages	Table
2	R1002	Coffee	2.40	Beverages	Table
3	R1003	Tea	1.50	Beverages	Table
4	R1004	Hot Chocolate	1.80	Beverages	Table
5	R1005	Beer	3.90	Beverages	Table
6	R1006	Wine	6.60	Beverages	Table
7	R1007	Water	1.00	Beverages	Table
8	R1008	Sandwich	3.30	Food	Table
9	R1009	Samosa	2.50	Food	Table
10	R1010	Baguette	2.80	Food	Table
11	R1011	Soup	2.00	Food	Table
12	R1012	Jacket Potato	3.20	Food	Table
13	R1014	Sausage Roll	2.50	Food	Table
14	R1015	Blueberry Muffin	1.40	Confections	Table
15	R1016	Chocolate Chip Muffin	1.40	Confections	Table
16	R1017	Croissant	1.70	Confections	Table
17	R1018	Caramel Shortbread	2.20	Confections	Table
18	R1020	Crisps	0.90	Confections	Table

Figure 6.14: The two rows are excluded from the merge result

The purpose of the merge was to remove the matched rows, so we have no requirement for the column that has been added. Let us remove it.

- Right-click on the **tblDiscontinued** column header and click **Remove**.

Remember, these actions are all recorded in the **Applied Steps** list and can be edited or removed at any time if required. In figure 6.15, you can see both merge query operations that we performed listed in the Applied Steps.

These steps can be renamed for better documentation of your queries. The default step names are not very meaningful.

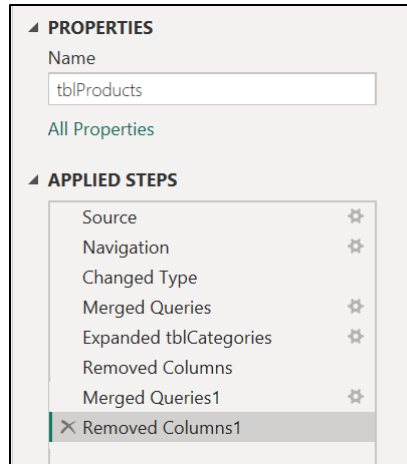


Figure 6.15: Completed steps with the two merge queries

Note: In practice, it would be better to perform these two merge queries in reverse order to how they are demonstrated here. Removing unnecessary rows before looking up their category would be more logical.

They are shown in this order for a better learning experience. To show the default Left Outer join before seeing another example.

Establishing the date data type

File: **dates-and-calculations.pbix**

Many queries contain one or more date columns, especially fact tables.

Dates come in many shapes and sizes, and because of this, they are not always recognized by Power Query when it runs the automatic Changed Type step.

Let us see a couple of examples of unrecognized dates and how to correct them and apply the date data type.

Using the locale data type

Figure 6.16 shows the **Customers** table in the **dates-and-calculations.pbix** file. The **DoB** column contains the date of birth for each customer.

Power Query has performed its automatic **Changed Type** step, but the dates have not been correctly understood. You can see some of the dates are aligned to the left of the column, and others are aligned to the right:

A ^B C Lastname	A ^B C Firstname	A ^B C ₁₂₃ DoB	A ^B C Dept
Becker	Steve	10/21/1981	Finance
Beckford	Darren	05/08/1972	HR
Patel	Gita	09/01/1987	Customer Services
Jewell	Anna	03/30/2000	Customer Services
Frood	Ian	11/06/1993	HR
Dalby	Carly	12/17/1998	Technical services
Hill	Lauren	02/09/1979	HR
Percy	Ashley	04/08/1991	Sales
Francis	Ian	06/19/1980	Finance
Smith	Hollie	02/10/1978	Sales

Figure 6.16: DoB column contains the unrecognized date

This is because the dates are in an MDY format. The regional settings on my version of Power BI Desktop are set to English (United Kingdom). Therefore, my version of Power Query is expecting a date in the DMY format.

Some dates are not recognized due to the month number (this is the day really) being greater than 12. And other dates, which Power Query believes it understands, are stored as the wrong date.

The data type of the **DoB** column is currently set to alphanumeric. If we tried to change the data type to the Date type, the unrecognized dates would return errors (*figure 6.17*).

A ^B C Firstname	A ^B C ₁₂₃ DoB	A ^B C Dept
Steve	Error	Finance
Darren	05/08/1972	HR
Gita	09/01/1987	Customer Services
Anna	Error	Customer Services
Ian	11/06/1993	HR
Carly	Error	Technical services
Lauren	02/09/1979	HR
Ashley	04/08/1991	Sales
Ian	Error	Finance
Hollie	02/10/1978	Sales

Figure 6.17: Errors returned when applying the Date data type

Fortunately, there is a very simple fix for this issue, the Locale data type.

1. Click the change type button in the **DoB** column header and click **Using Locale** (*figure 6.18*):

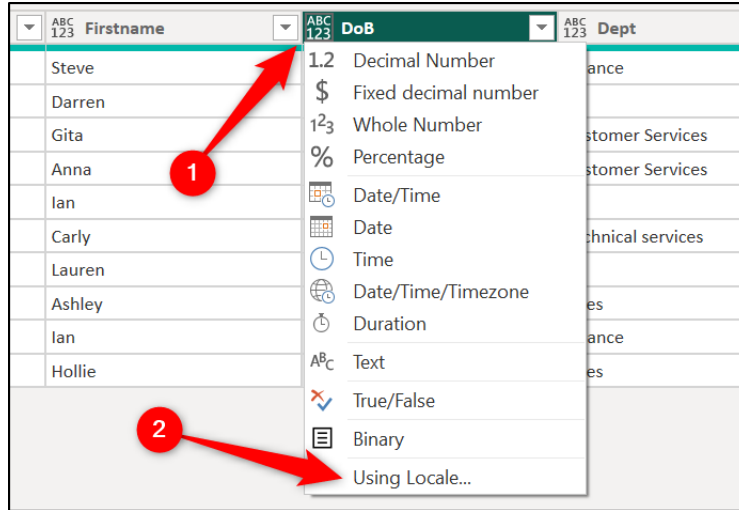


Figure 6.18: Using the Locale data type

- In the **Change Type with Locale** window, select **Date** from the **Data Type** list (figure 6.19).
- Select **English (United States)** from the **Locale** list. Click **OK**.

Tip: A quick way to jump to locales such as **English (United Kingdom)** or **English (United States)** is to type “f” when in the list of locales. It jumps to the locales that begin with an “f”. The English locales are just before this. Faster than scrolling at times.

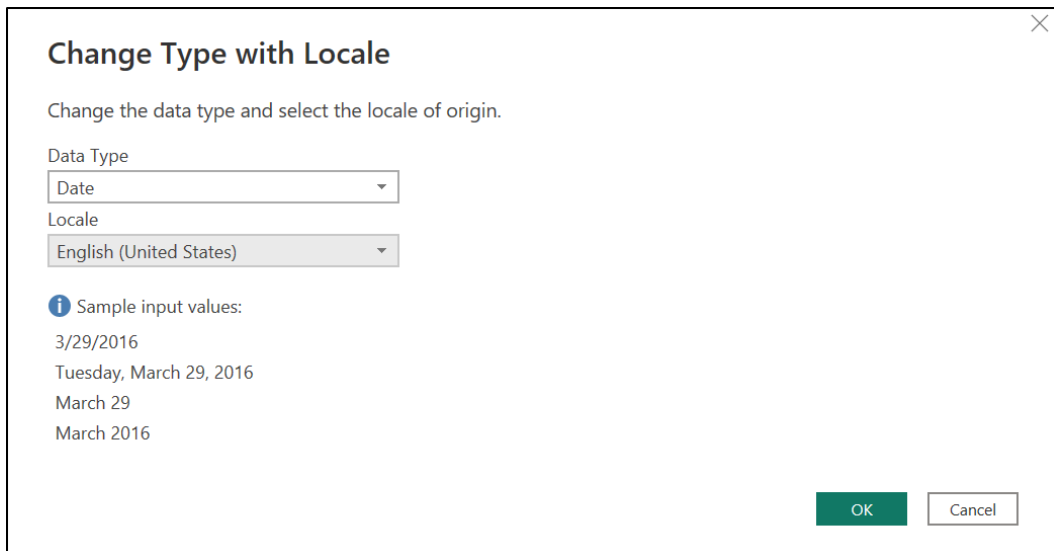


Figure 6.19: Change type with locale to recognize English (United States) dates

In *figure 6.20*, you can see the corrected dates.

Using the **English (United States)** locale fixed them, and they are formatted in the **English (United Kingdom)** format. This is specified by my regional settings:

ABC 123	Firstname	DoB	ABC 123 Dept
	Steve	21/10/1981	Finance
	Darren	05/08/1972	HR
	Gita	09/01/1987	Customer Services
	Anna	30/03/2000	Customer Services
	Ian	11/06/1993	HR
	Carly	17/12/1998	Technical services
	Lauren	02/09/1979	HR
	Ashley	04/08/1991	Sales
	Ian	19/06/1980	Finance
	Hollie	02/10/1978	Sales

Figure 6.20: Dates fixed by using the locale data type

The **Using Locale** data type is very useful when working with international number formats and is not specific to only handling dates.

Converting ISO 8601 Dates

In this example, we will look at converting dates stored in the ISO 8601 format of *yyyymmdd* to the Date data type.

Figure 6.21 shows the **Customers** table in the **dates-and-calculations.pbix** file.

The **Date Joined** column contains dates in the ISO format. As there are no delimiters in the date format, the Whole Number data type was automatically applied by Power Query.

ABC 123	Dept	ABC 123 Date Joined
1	Finance	20210209
2	HR	20050120
7	Customer Services	20100904
0	Customer Services	20120430
3	HR	20190801
3	Technical services	20151105
9	HR	20200502
4	Sales	20210915
0	Finance	20161210
3	Sales	20180606

Figure 6.21: ISO dates stored as whole numbers

We cannot change it to the Date data type directly. This would result in errors for all the values.

First, we change it to the Text data type. Then we change it to the Date data type as a separate step.

Click the change type button in the **Date Joined** header and click **Text**.

Figure 6.22 shows the Text data type applied. Not much improvement yet.

ABC 123 Dept	ABC Date Joined
Finance	20210209
HR	20050120
Customer Services	20100904
Customer Services	20120430
HR	20190801
Technical services	20151105
HR	20200502
Sales	20210915
Finance	20161210
Sales	20180606

Figure 6.22: Converting the ISO dates to text values

1. Click the change type button in the **Date Joined** header and click **Date**.
A message appears asking if you would like to change the existing changed type step or add a new changed type step for this conversion (figure 6.23).
2. Click **Add new step**. It is important that we preserve the previous change to the Text data type.

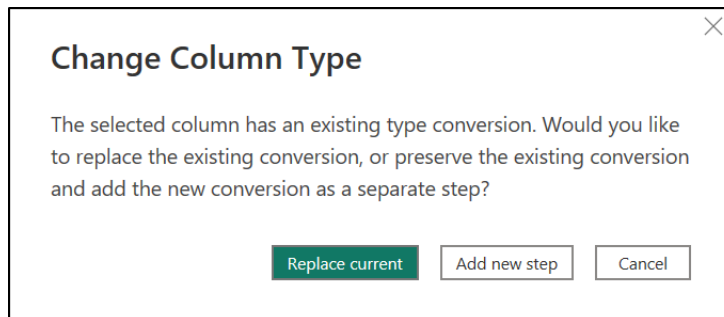


Figure 6.23: Adding a new Changed Type step

The dates are now formatted correctly. In *figure 6.24*, you can see the corrected dates and the two separate steps added to the Applied Steps list. The steps have been renamed for better documentation of the query.

Note: Another approach to convert the ISO dates could have been to use the Column from Examples feature shown in *Chapter 4, Creating a Simple Power BI Report*. There are often many methods to achieving the transformation that you need in Power Query.

ABC 123	Dept	Date Joined
	Finance	09/02/2021
	HR	20/01/2005
	Customer Services	04/09/2010
	Customer Services	30/04/2012
	HR	01/08/2019
	Technical services	05/11/2015
	HR	02/05/2020
	Sales	15/09/2021
	Finance	10/12/2016
	Sales	06/06/2018

PROPERTIES

Name
Customers

All Properties

APPLIED STEPS

- Source
- Navigation
- Changed Type with Locale
- Converted ISO to Text
- Converted ISO to Date

Figure 6.24: ISO dates validated with the Date data type

Calculations in power query

File: **dates-and-calculations.pbix**

There are many calculations that you can perform in Power Query while writing little to no M code (the language of Power Query).

Let us look at some examples, including basic mathematical calculations, date functions, inserting conditional columns, and rounding values.

The Power Query UI provides buttons to perform a variety of calculations on both the Transform and Add Column tabs.

Figure 6.25 shows the **Transform** tab on the Ribbon. The Number Column and Date and Time Column groups provide easy access to many formulas. Using the formulas on the Transform tab will change the values in the existing column.

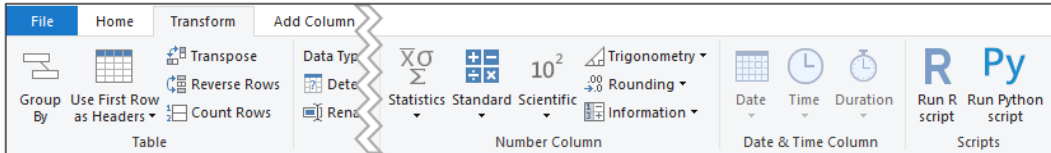


Figure 6.25: Calculations on the Transform tab of Power Query

Figure 6.26 shows the formulas on the **Add Column** tab of the Ribbon. These formulas would not change the existing column values but would add a new column with the formula results.

The same formulas are available here, as were seen on the **Transform** tab. There are additional formulas at the start of the Ribbon, though, including **Custom Column**, **Conditional Column**, and **Index Column**. Please refer to the following figure:

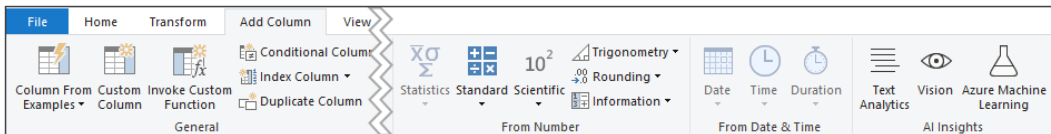


Figure 6.26: Calculations on the Add Column tab of Power Query

Mathematical calculations

Power Query provides a list of standard mathematical calculations on both the **Transform** and **Add Column** tabs to make them easier to apply.

These calculations include adding, multiplying, percentage of, and modulo operations, among others.

For our example, we will use the **Data** table in the **dates-and-calculations.pbix** file. We will apply a Pound Sterling to US Dollar exchange rate formula to the **Value** column.

The Pound Sterling to US Dollar exchange rate is currently at 1.32. So, we will multiply each of the values in the column by 1.32.

1. Click on the **Value** column to select it.
2. Click **Transform** | **Standard** | **Multiply** (figure 6.27).

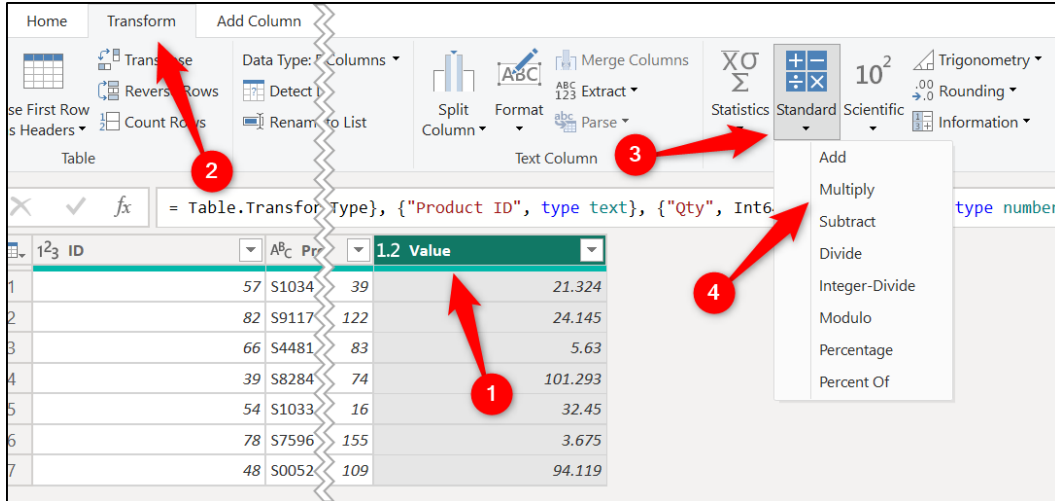


Figure 6.27: Applying the multiply calculation to values

3. Type "1.32" in the **Value** box of the **Multiply** window. Click **OK**.

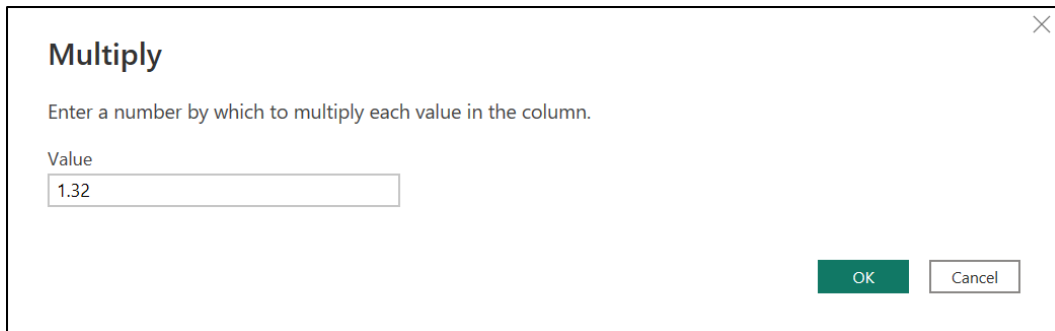


Figure 6.28: Entering the number to multiply the column values by

Each of the values in the column is multiplied by 1.32, and the results returned.

In figure 6.29, you can see the resulting **Value** column. The image also shows the **Multiplied Column** step added to the **Applied Steps** list. Please refer to the following figure:

	Product ID	Qty	1.2 Value
57	S1034	39	28.14768
82	S9117	122	31.8714
66	S4481	83	7.4316
39	S8284	74	133.70676
54	S1033	16	42.834
78	S7596	155	4.851
48	S0052	109	124.23708

Figure 6.29: Results of the exchange rate formula

In the formula, you can see that `table.TransformColumns` function was used. You can also make out the `* 1.32` calculation, and the data type is set as `number`. By exploring the results of Power Query actions, you will become more familiar with the M language.

Rounding values

Following the previous formula, let us now look at rounding the values returned from that calculation. We will round the values to two decimal places.

1. Click on the **Value** column to select it.
2. Click **Transform** | **Rounding** | **Round** (figure 6.30).

Three rounding calculations are available from the UI in Power Query—Round Up, Round Down, and Round.

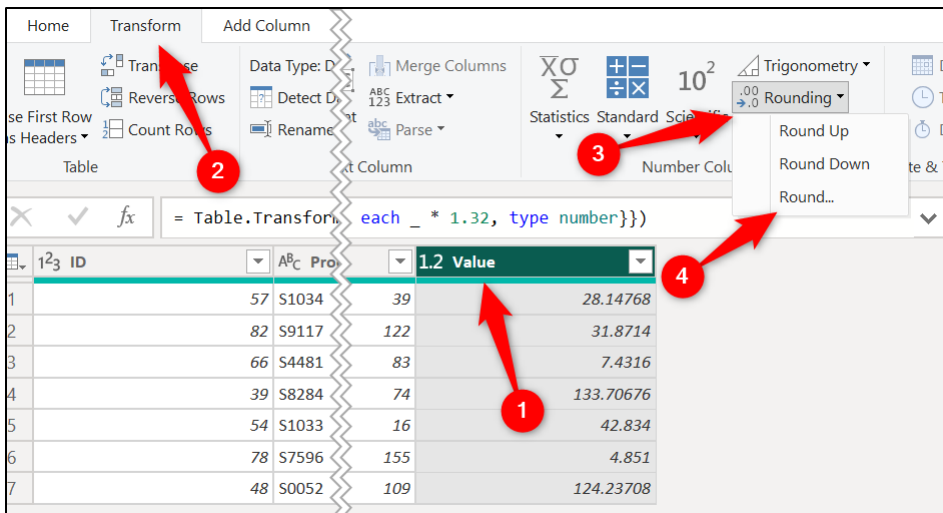


Figure 6.30: Rounding calculations available from the UI in Power Query

- In the **Round** window (figure 6.31), type “2” for the number of **Decimal Places**. Click **OK**:

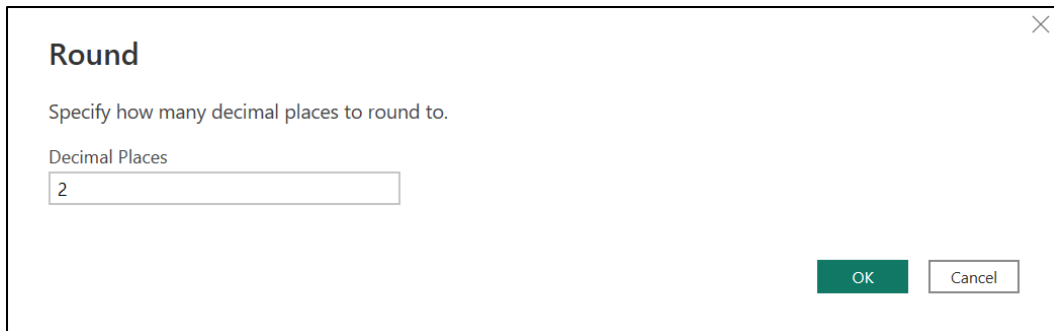


Figure 6.31: Rounding values to two decimal places

The numbers in the **Value** column are rounded to two decimal places, and a **Rounded Off** step is added to the query (figure 6.32).

Note: The rounding method used by Power Query is to round tied values to the nearest even number (known as “Bankers rounding”). This means that the value 6.45 would round to 6.4 if rounded to one decimal place. However, 6.55 would round to 6.6.

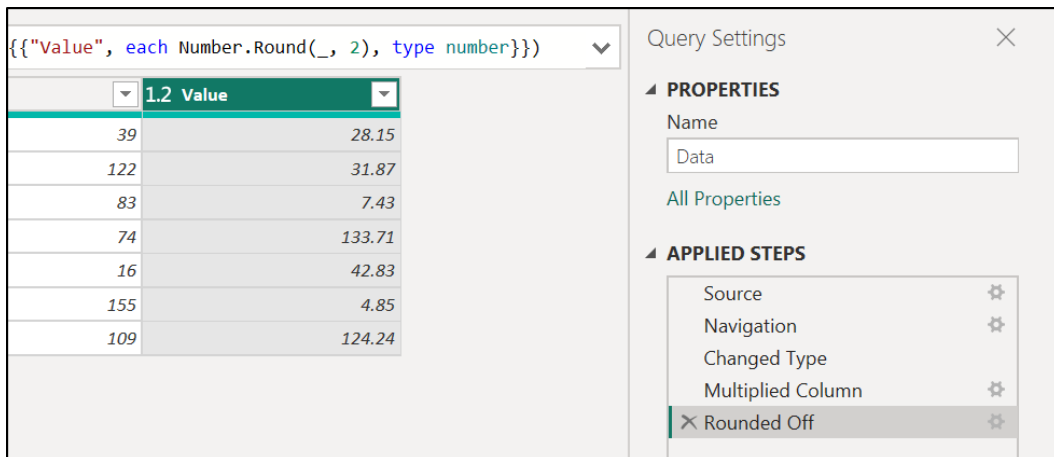


Figure 6.32: Rounded-off step added, and the values transformed

Adding and editing M code

Many calculations are available from the UI in Power Query. However, as you progress your Power Query knowledge, you will begin to write your own calculations, beyond the UI.

Power Query offers a few areas where you can write your own M code. These include the Formula Bar, Custom Columns, and the Advanced Editor.

Let us look at doing a little Formula Bar editing now to change the rounding method used in the previous example, and then, to define a data type in the same step.

Changing the rounding method

By default, when you use the Round calculation offered from the **Transform** and **Add Column** tabs (as we did in the last calculation), Power Query performs the rounding method called **RoundingMode.ToEven**.

This means that Power Query rounds all tied values to the nearest even number. This is known as “**Bankers Rounding**”.

So, the value 9.225, when rounded to two decimal places, would be round to 9.22. This is because 9.22 is a closer even number than 9.24, and Power Query rounds to the nearest even number.

You may have expected and preferred that the result of this tied value be 9.23. To specify this, we will edit the Round function to change the rounding method that is used.

1. Ensure that the last step in the query, **Rounded Off**, is selected. This will show the M code for that step in the Formula Bar.

Tip: If you cannot see the Formula Bar, click View | Formula Bar.

2. Click in the **Round** function before the closing bracket and type a comma (,), as shown in *figure 6.33*. A syntax box appears showing the arguments of the Round function and additional helpful information:

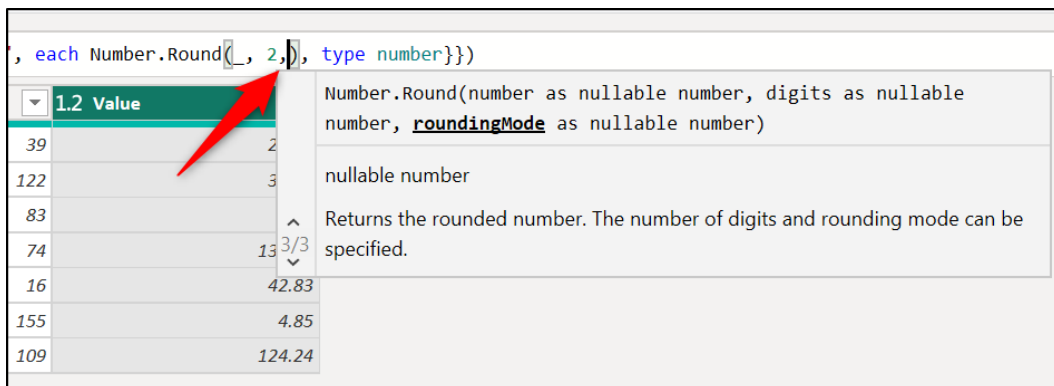


Figure 6.33: Formula bar showing the arguments for the Round function

The box shows that we are in the **roundingMode** argument of the function.

3. Start typing “**Rounding**” into the Formula Bar, and the Intellisense list appears, showing the options that match the entered text (figure 6.34).

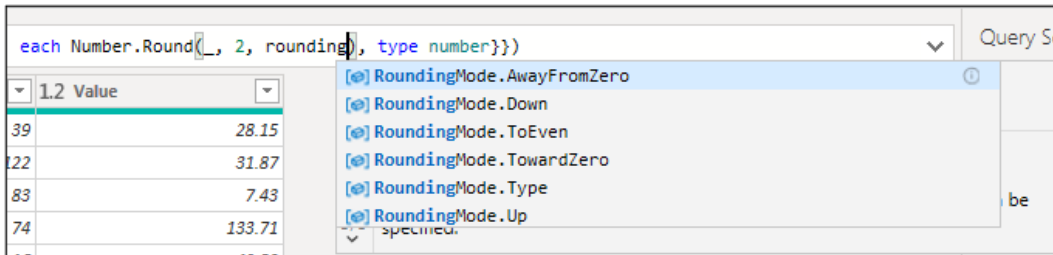


Figure 6.34: The different rounding modes of Power Query

4. Click on the **RoundingMode.AwayFromZero** option, or press *Tab* on the keyboard, to enter it into the formula.

This rounding mode is the most common rounding method used. Numbers less than 5 round down, and numbers 5 and above round up.

Figure 6.35 shows the completed Round function and the resulting values. In this example, the values have not changed because there are no tied values. However, this may change when the data is refreshed, so it is important to specify the desired rounding method. Please refer to the following figure:

1.2 Value	Value
39	28.15
122	31.87
83	7.43
74	133.71
16	42.83
155	4.85
109	124.24

Figure 6.35: Round function with the rounding mode specified

The following M code is the complete code for this step:

```
= table.TransformColumns("#Multiplied Column",{"Value", each Number.Round(_, 2, RoundingMode.AwayFromZero), type number}})
```

Defining a data type

Let us make another edit to the M code of that step to define a data type. This is a useful technique, especially when adding columns, as it reduces the number of steps in a query, therefore, reducing clutter.

In *figure 6.36*, you can see that the last step of our query currently defines the data type as a decimal number. This is visible in the M code but also by the icon in the header of the **Value** column.

Let us change this to the Fixed Decimal Number data type, also known as the Currency data type.

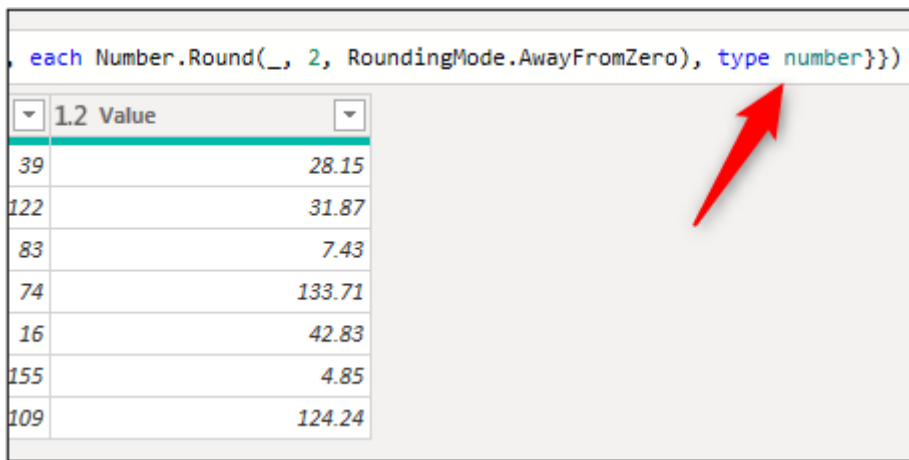


Figure 6.36: Changing the data type within the transformation step

Click in the formula and replace the text “**type number**” with “**Currency.Type**”. Press **Enter**.

Tip: You can learn the M code for a specific data type easily by performing the data type change and then looking at the M code created for that step. You can then delete the step and type the code that you saw it generate.

In *figure 6.37*, you can see the Currency data type applied to the same **Rounded Off** step. This is a more efficient approach than creating an additional **Changed Type** step (unless other types need to be specified):

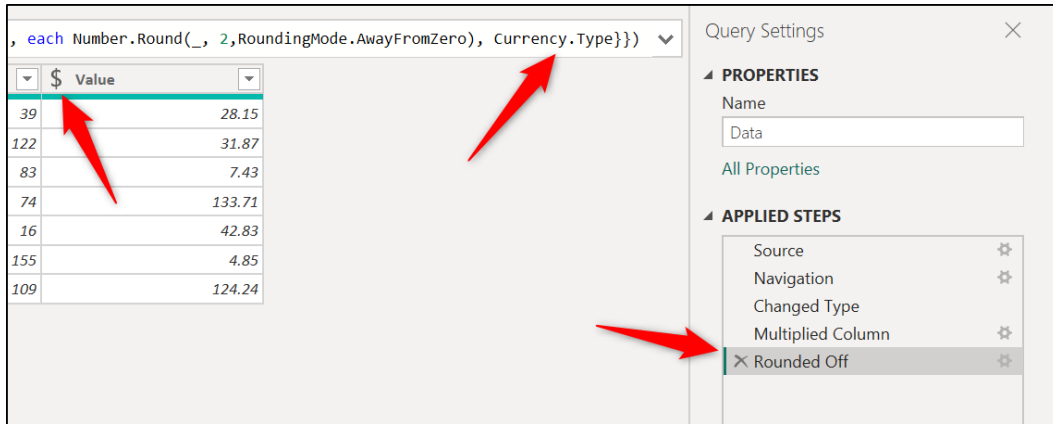


Figure 6.37: Currency data type specified within the transformation step

Date functions—calculating age

Power Query contains many useful date calculations within the UI, making them very quick and easy to use.

Figure 6.38 shows the list of date calculations available from the Transform or Add Column tabs of the Ribbon. There are some terrific calculations in here.

When the Month option is selected, a sub-menu is activated to show the options like End of Month, Days in Month, and Name of Month. Please refer to the following figure:

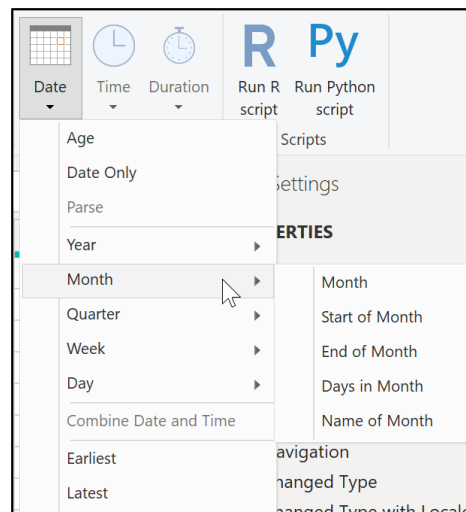


Figure 6.38: Date calculations in Power Query

For this example, we will add a column and calculate the age of customers using a date of birth column.

We will be using the **Customers** query in the **dates-and-calculations.pbix** file.

- Click on the **DoB** column to select it.
- Click **Add Column | Date | Age** (figure 6.39).

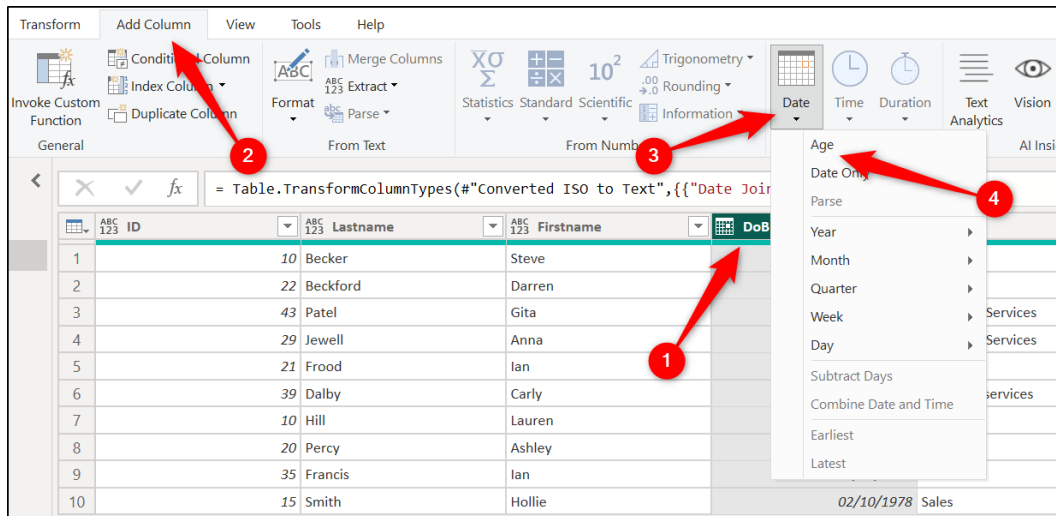


Figure 6.39: Calculate age from date of birth

A new column named **Age** is added to the end of the table. In figure 6.40, you can see that the results have been returned in the duration data type. It has returned the number of days difference and a time value.

The screenshot shows the Power BI 'Transform' view. The ribbon includes 'Date', 'Time', 'Duration', 'Text', and 'Vision'. The 'Date' dropdown menu is open, showing 'Age' selected. Below the ribbon, a table with columns 'DoB', 'Dept', 'Date Joined', and 'Age' is visible. A red arrow with the number 1 points to the 'Date' column.

DoB	Dept	Date Joined	Age
21/10/1981	Finance	09/02/2021	15059.00:00:00
05/08/1972	HR	20/01/2005	18423.00:00:00
09/01/1987	Customer Services	04/09/2010	13153.00:00:00
30/03/2000	Customer Services	30/04/2012	8324.00:00:00
11/06/1993	HR	01/08/2019	10808.00:00:00
17/12/1998	Technical services	05/11/2015	8793.00:00:00
02/09/1979	HR	02/05/2020	15839.00:00:00
04/08/1991	Sales	15/09/2021	11485.00:00:00
19/06/1980	Finance	10/12/2016	15548.00:00:00
02/10/1978	Sales	06/06/2018	16174.00:00:00

Figure 6.40: Age calculation returns the result in the duration data type

This is not the result that we need, so we will add further transformation steps.

We will convert the result into the number of years and then round it down to the nearest integer.

- Click on the **Age** column to select it.
- Click **Transform | Duration | Total Years** (figure 6.41):

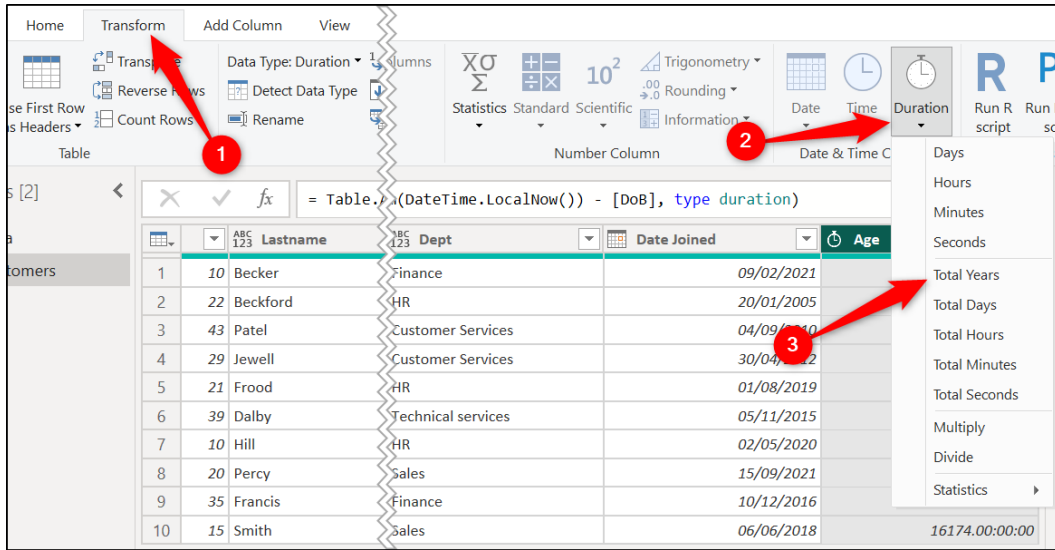


Figure 6.41: Transform the duration to total years

The **Age** column now shows the total years as a decimal value (figure 6.42):

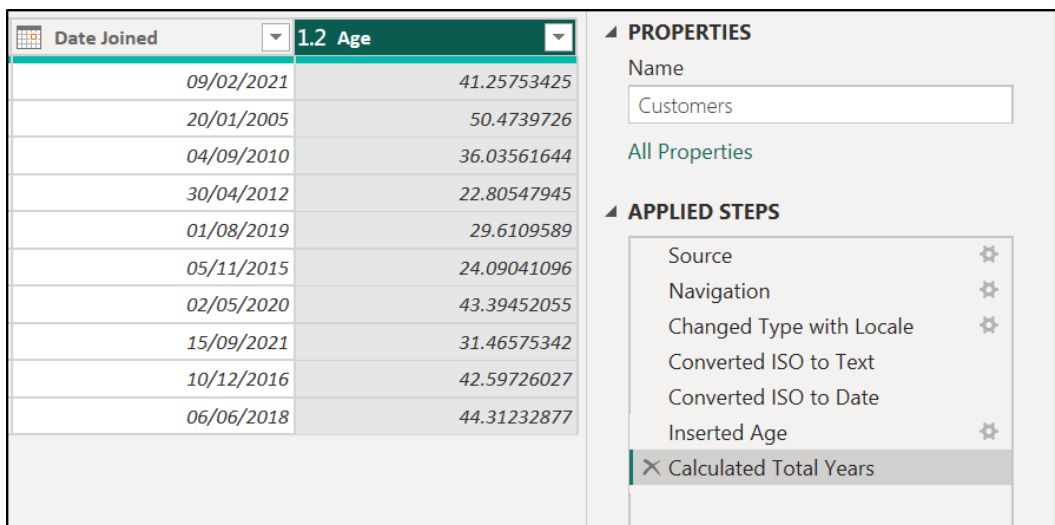


Figure 6.42: Total years calculation and added step

To remove the decimal part of the result, we will round down to the nearest integer. As we are calculating their age, it is important we round down at this step.

- Click on the **Age** column to select it.
- Click **Transform** | **Rounding** | **Round Down** (figure 6.43).

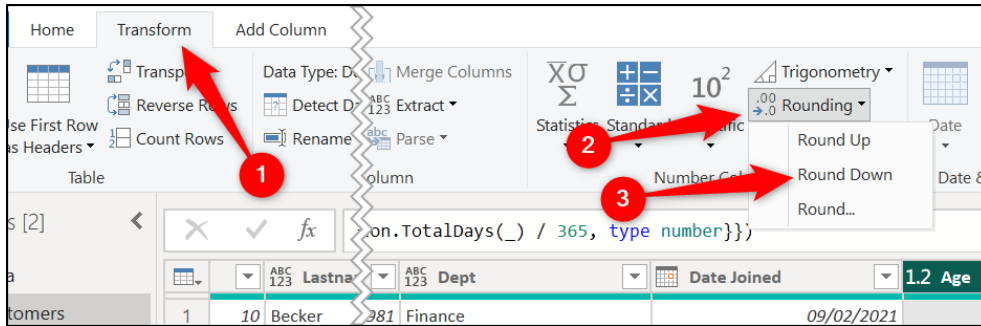


Figure 6.43: Rounding down the age values to remove the decimal part

Figure 6.44 shows the completed **Age** column with the three steps added to the **Applied Steps** list:

Date Joined	Age
09/02/2021	41
20/01/2005	50
04/09/2010	36
30/04/2012	22
01/08/2019	29
05/11/2015	24
02/05/2020	43
15/09/2021	31
10/12/2016	42
06/06/2018	44

PROPERTIES

Name: Customers

APPLIED STEPS

- Source
- Navigation
- Changed Type with Locale
- Converted ISO to Text
- Converted ISO to Date
- Inserted Age
- Calculated Total Years
- × Rounded Down**

Figure 6.44: Rounded down step added to complete the age calculation

Inserting conditional columns

Power Query provides a command on the **Add Column** tab of the Ribbon to insert basic conditional columns to your queries.

In this example, we will use the **Data** query in the **dates-and-calculations.pbix** file.

This query has a column that contains the quantity of an item ordered. We will test this column and assign the groupings X Large, Big, Medium, and Small, depending on the number ordered.

1. From the **Data** query, click **Add Column | Conditional Column**.
The **Add Conditional Column** window appears. We will now complete all the relevant fields, as shown in *figure 6.45*.
2. Type **Qty Size** for the **New column name**.
3. Click the **Column Name** list arrow and select **Qty** for the column to test.
4. Click the Operator list arrow and select **is greater than or equal to** as the operator.
5. Type **"120"** for the **Value** field.
6. Type **"X Large"** for the **Output**. It is important that we begin with testing the largest quantity and work toward the smallest.

Note: You can also select a column using the button to the left of the Value and Output fields instead of typing values. This can be useful to test one column's value against another column.

Click the **Add Clause** button to insert another row for the second condition.

Add Conditional Column

Add a conditional column that is computed from the other columns or values.

New column name
Qty Size

	Column Name	Operator	Value	Output
If	Qty	is greater than or...	120	X Large
Else If	Qty	is greater than or...	85	Big
Else If	Qty	is greater than or...	35	Medium

Add Clause

Else
Small

OK Cancel

Figure 6.45: Adding a conditional column in Power Query

7. For this condition, select the **Qty** column, is greater than or equal to for the Operator, type **"85"** for the Value, and type **"Big"** for the Output.
8. Click **Add Clause**.

9. For the third logical test, select the **Qty** column, which is greater than or equal to for the Operator, type “35” for the Value, and type “Medium” for the Output.
10. In the **Else** field, type “Small”. Click **OK**.

Note: Power Query is case-sensitive, so if you are testing text values, ensure to enter the values in the correct case.

Figure 6.46 shows the **Qty Size** column added to the query. Each group name has been correctly assigned based on the value in the **Qty** column:

ID	Product ID	Qty	Value	Qty Size
1	S1034	39	28.15	Medium
2	S9117	122	31.87	X Large
3	S4481	83	7.43	Medium
4	S8284	74	133.71	Medium
5	S1033	16	42.83	Small
6	S7596	155	4.85	X Large
7	S0052	109	124.24	Big

Figure 6.46: Qty Size column added with the formula shown

Check the Formula Bar for this step. It is useful to familiarize yourself with the M language when possible. You can make edits to the formula in the bar or by clicking the gear icon beside the step in the Applied Steps list.

Let us edit the code in the Formula Bar to define the text data type within the same step.

Click at the end of the formula inside the final closing bracket and type the following small bit of code. This is shown in *figure 6.47*.

, type text

The comma gives us access to the **columnType** argument of the **table.AddColumn** function. Type text defines the text data type.

The following is the complete M code for this step:

```
= table.AddColumn(#"Rounded Off", "Qty Size", each if [Qty] >= 120 then "X Large" else if [Qty] >= 85 then "Big" else if [Qty] >= 35 then "Medium" else "Small", type text)
```

Query Settings

PROPERTIES

Name: Data

All Properties

APPLIED STEPS

- Source
- Navigation
- Changed Type
- Multiplied Column
- Rounded Off
- Added Conditional Column

```
[Qty] >= 120 then "X Large" else if [Qty] >= 85 then "1", type text)
```

	\$ Value	Qty Size
39	26	Medium
122	1.87	X Large
83	7.43	Medium
74	133.71	Medium
16	42.83	Small
155	4.85	X Large
109	124.24	Big

Figure 6.47: Editing the formula to define the data type

Inserting a Conditional Column is great, but it is only useful for creating basic conditional statements. Inserting a Custom Column and writing your *if* statements using M offers much greater flexibility than the Conditional Column method.

By writing M, you can create more complex logic that includes operators such as **And** or **Or**, you can use other M functions within the conditional statements, and you can perform calculations with other column values.

However, this goes beyond the scope of this chapter. The aim is to look at powerful calculations in Power Query, mainly using the options available on the Ribbon. I encourage you to learn more and try and create more complex conditional statements for practice.

Unpivoting columns

File: **unpivot.pbix**

When you ask a Power Query user, what is it that first made you “Wow” in Power Query? Unpivoting Columns is a very commonly heard answer.

It, therefore, had to be included as one of the data transformations before we dive into other aspects of Power BI.

Unpivoting columns convert header values into a single column and place all values that belong to that header in a column alongside them. This is also commonly referred to as flattening a table.

Note: The Unpivoting Columns tool is often mistakenly confused with a tool called Transpose. The Transpose tool flips the entire table from rows to columns or vice versa.

Figure 6.48 shows the results of unpivoting columns in Power Query:

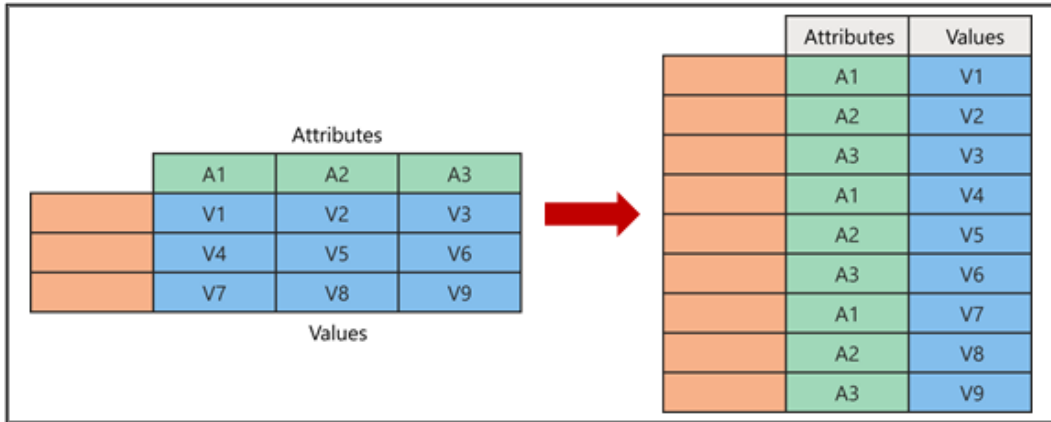


Figure 6.48: Diagram that illustrates the result of unpivoting columns ¹

The header values in green are converted into a single column named **Attributes**. The values in blue are placed in a single column named **Values** and sit beside the header they belong to.

Let us see unpivoting columns in action. In this example, we will use the **SalesbyCity** table in the **unpivot.pbix** file. This table is shown in figure 6.49.

APc Product	1 ² Chicago	1 ² Boston	1 ² Denver	1 ² Houston
1 Cheese	47337	22245	27809	36092
2 Juice	31248	26159	48868	49372
3 Wine	46746	42622	23766	23315
4 Beef	20689	50411	30708	41397
5 Chocolate	45169	45080	49920	49874
6 Bread	31725	48173	37145	47219
7 Biscuits	23798	45017	52972	28506
8 Pepper	52611	49818	26913	44473

Figure 6.49: SalesbyCity table with city values in the header row

We have product values in a column, city values in a header row, and sales values that correspond to that product and city.

1 <https://support.microsoft.com/en-us/office/unpivot-columns-power-query-0f7bad4b-9ea1-49c1-9d95-f588221c7098>

When unpivoting columns, you get the option to unpivot the selected columns or to unpivot the unselected columns. It is worth considering what the best approach for your data is. Does your data have a regular set of columns? Or is it irregular?

With the **SalesbyCity** table, we want to unpivot the columns that contain the city header values.

We will use the method to unpivot all columns except the **Product** column. This will ensure that our query continues to work when we refresh our reports in the future. Even if different city columns appear in the table.

1. Click on the **Product** column to select it.
2. Click **Transform** | **Unpivot Columns** list arrow | **Unpivot Other Columns** (figure 6.50).

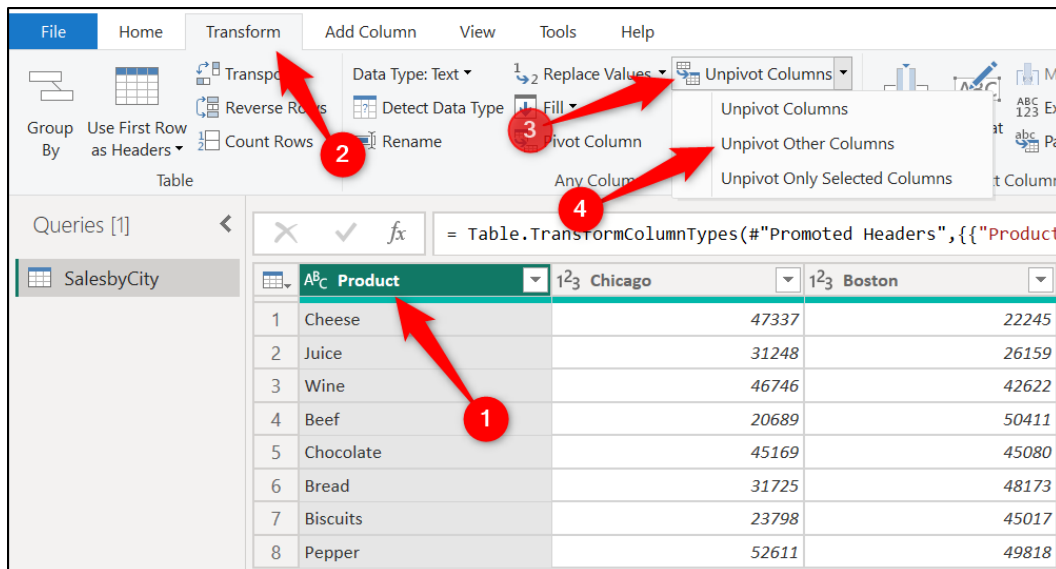


Figure 6.50: Unpivot all columns except the Product column

Figure 6.51 shows the result of unpivoting the columns. The **Attribute** and **Value** columns are created for the unpivoted columns' values.

Tip: You can also right-click on the selected columns to access all three Unpivot Columns options.

This flattened table is the result that we needed to effectively analyze the sales data by city:

	Product	Attribute	Value
1	Cheese	Chicago	47337
2	Cheese	Boston	22245
3	Cheese	Denver	27809
4	Cheese	Houston	36092
5	Juice	Chicago	31248
6	Juice	Boston	26159
7	Juice	Denver	48868
8	Juice	Houston	49372
9	Wine	Chicago	46746
10	Wine	Boston	42622

Figure 6.51: Attribute and Value columns created for the unpivoted column values

To complete the table, we will rename the Attribute and Value columns to **City** and **Total**, respectively.

To avoid adding an additional step to the query, let us edit the M code for this step. The following M code is shown in the formula bar for the Unpivoted Other Columns step:

```
= table.UnpivotOtherColumns("#Changed Type", {"Product"}, "Attribute", "Value")
```

Edit the **Attribute** header to **City** and the **Value** header to **Total**, as shown in the following completed M code.

```
= table.UnpivotOtherColumns("#Changed Type", {"Product"}, "City", "Total")
```

Figure 6.52 shows the completed query with the renamed headers:

	Product	City	Total
1	Cheese	Chicago	47337
2	Cheese	Boston	22245
3	Cheese	Denver	27809
4	Cheese	Houston	36092
5	Juice	Chicago	31248
6	Juice	Boston	26159
7	Juice	Denver	48868
8	Juice	Houston	49372
9	Wine	Chicago	46746
10	Wine	Boston	42622

Figure 6.52: Completed query with the renamed headers

Conclusion

In this chapter, we learnt more data transformations that are possible in Power Query. We saw the power of merge queries and learnt the different joint types available and when to use them.

Working with dates is an essential skill for someone working with data, and we learnt how to perform different transformations on different date formats and structures.

We also saw the different calculations built into Power Query that can be used easily through the UI. But also how you can make some simple edits to the M code to take your calculations further and reduce the steps of your queries.

Finally, we used the Unpivot Columns tool of Power Query. An incredibly popular tool to convert pivoted columns into columns of a table.

In the upcoming chapter, we return to the **sales-report.pbix** report that we were preparing in *Chapter 5, Getting and Shaping Data*. We will prepare the data model for the report by creating relationships between the tables, formatting the columns, and hiding any fields that will not be used within the visuals of our report.

Questions

Here are some questions to test what you have learnt in this chapter.

1. Which of the following can be found in the date calculations list in Power Query?
 - a. Days in Month
 - b. End of Month
 - c. Age
 - d. All of the above
2. Which of the following is not a join type?
 - a. Inner
 - b. Complete
 - c. Left Outer
 - d. Left Anti
3. What is the correct path to unpivot columns in Power Query?
 - a. Click on the columns > Transform > Unpivot Columns
 - b. Click on the columns > File > Unpivot Columns

- c. Click on the columns > Add Column > Unpivot Columns
 - d. Click on the columns > Home > Unpivot Columns
4. Which join type is used to pull across values for all matching items in the first table?
- a. Lookup
 - b. Left Anti
 - c. Full Outer
 - d. Left Outer
5. Which of the following are calculation groups on the Transform tab?
- a. Standard
 - b. Rounding
 - c. Statistics
 - d. All of the above

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



CHAPTER 7

Creating the Data Model

Introduction

In *Chapter 5, Getting and Shaping Data*, we imported the data required for our report and then performed numerous transformations to prepare the data before loading it into the Power BI data model.

In this chapter, we fine-tune the data model to be ready for the **Data Analysis Expressions (DAX)** calculations and report visuals that are to come.

We will start by creating the relationships between the tables of our model. We will understand why this is important and how we can manage these relationships once they are established.

There are two queries that have been loaded into the data model in Power BI that will not be used. They were useful in preparing our data but will not be used at the reporting stage. So, we will look at how we can disable these queries from loading.

Next, we will work on the fields of the tables in the model. We will format the table fields as required, hide fields that will not be used in the report visuals and delete fields that are not required at all.

Structure

In this chapter, we will cover the following topics:

- Creating relationships between tables
- Managing the relationships of a model
- Disabling queries from loading to Power BI
- Formatting table fields
- Hiding table fields from Report view
- Deleting fields from the model

Objectives

After reading this chapter, you will be able to create many-to-one relationships between the tables of your data model. You will also understand how to check, edit, and remove relationships between tables.

You will be able to accurately format the fields of your tables, hide fields that will not be used in the report visuals and prevent specific queries from loading to the model in Power BI.

Why create table relationships?

Files: `sales-report-ch-7-start.pbix`

Relationships are created to connect the different tables of a data model, allowing us to filter the data from one table to another. The tables are related using a key column from each table.

By default, Power BI is set to autodetect table relationships when data is loaded. Switch to the Model view of Power BI to see a diagram of these relationships. *Figure 7.1* shows the relationships autodetected for our sales report model.

Note: You can disable the setting to autodetect relationships between tables. Click File | Options and settings | Options | Data Load (for the current file only) | Autodetect new relationships after data is loaded.

Relationships have been created between the **Stores** and **Sales** tables and between the **Products** and **Sales** tables, but not between **Reps** and **Sales**. These relationships are identified by the lines that link the two tables.

When you first open the Model view of Power BI, the tables are often scattered across the canvas. You will want to click and drag them in an organized way to easily view and understand how they relate to each other.

In *figure 7.1*, you can see that I have moved the **Sales** table underneath the **Stores**, **Reps**, and **Products** tables. The **Table001** and **Table002** tables will be prevented from loading later in this chapter, so we can ignore them for now:

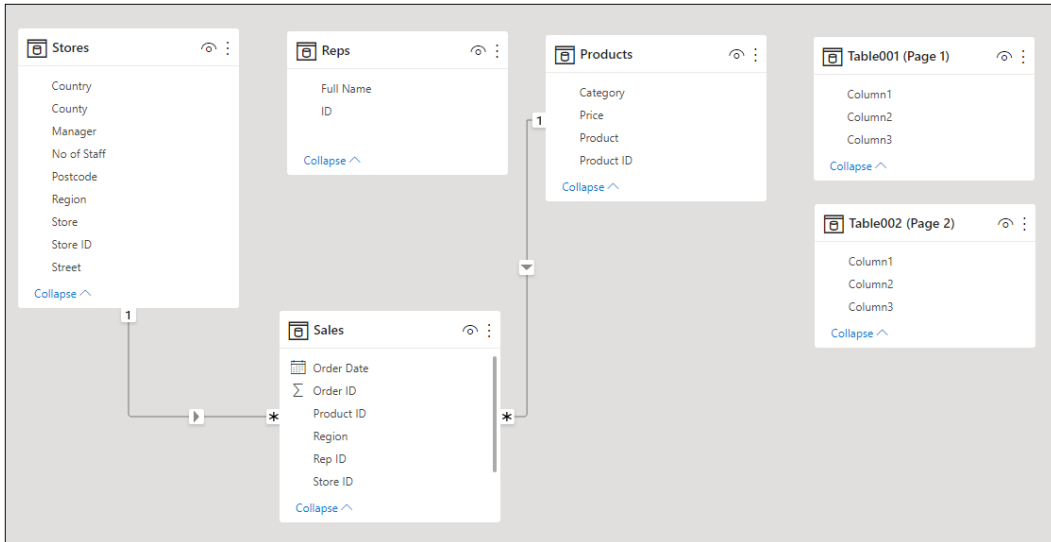


Figure 7.1: Autodetected relationships in the Model view

I have organized the tables in this manner to visualize the filter direction from the three dimension tables (Stores, Reps, and Products) to the fact table (Sales) easily. The filter direction is shown by the arrow on the line linking the tables.

Dimension tables (also known as lookup tables) describe the entities in the model. In this example – products, staff (reps), and stores. Dimension tables often have a small number of rows and contain descriptive columns.

Fact tables (also known as data tables) contain transactional data. In this example, it is sales. Other examples include training data, deliveries, stock updates, and test data. Fact tables contain many rows that continually grow (hopefully, the company performs many more sales transactions).

In *figure 7.2*, I have inserted two table visuals to the canvas in the Report view (we saw how to insert a table visual in *Chapter 4, Creating a Simple Power BI Report*).

The first table contains the **Full Name** field from the **Reps** table, and the **Units Sold** field from the **Sales** table. Power BI has applied the default summarization of Sum to the **Units Sold** field.

Power BI is unable to filter the sales data by the rep’s full name because there is no relationship between the **Reps** and **Sales** tables (*figure 7.1*). This is shown by each rep having the same unit sold value:

Full Name	Units Sold
Alex Freuer	215540
Audrey White	215540
Bernadette Olusola	215540
Christie Dankov	215540
Christopher Hartley	215540
Clare Jorquera	215540
Cyndy Bloom	215540
Daniel Torino	215540
Elizabeth Kendrick	215540
Georgia Keegan	215540
Gina Croft	215540
Giovanni Rovelli	215540
Total	215540

Store	Units Sold
Aberdeen	8914
Belfast	11737
Birmingham	8401
Brighton	19580
Bristol	8449
Cardiff	9872
Carlisle	5011
Glasgow	8741
Gloucester	12653
Guildford	6270
Harlow	23046
Hartlepool	11323
Total	215540

Figure 7.2: The same value for each item is a clear sign that the data cannot be filtered

The second table contains the **Store** field from the **Stores** table, and the **Units Sold** field from the **Sales** table. The units sold data is successfully filtered by stores as there is a working relationship between the two tables.

This highlights the importance of creating relationships between the tables of the model because, without them, we cannot apply filters between tables.

Cardinality

The relationship between two tables is defined by cardinality. This cardinality is identified by the symbols at the “from” and “to” ends of the line linking the two tables in the relationship.

Figure 7.3 shows the relationship between the **Stores** and **Sales** tables. The cardinality type of this relationship is known as one-to-many or many-to-one (these are essentially the same) and is identified by the one and the infinity symbols.

In this relationship, the **Stores** table contains unique values (the one side), and the **Sales** table can contain duplicates (the many side). So, a store can, and hopefully would have many sales related to it. But a single sales transaction would be related to only one store. Please refer to the following figure:

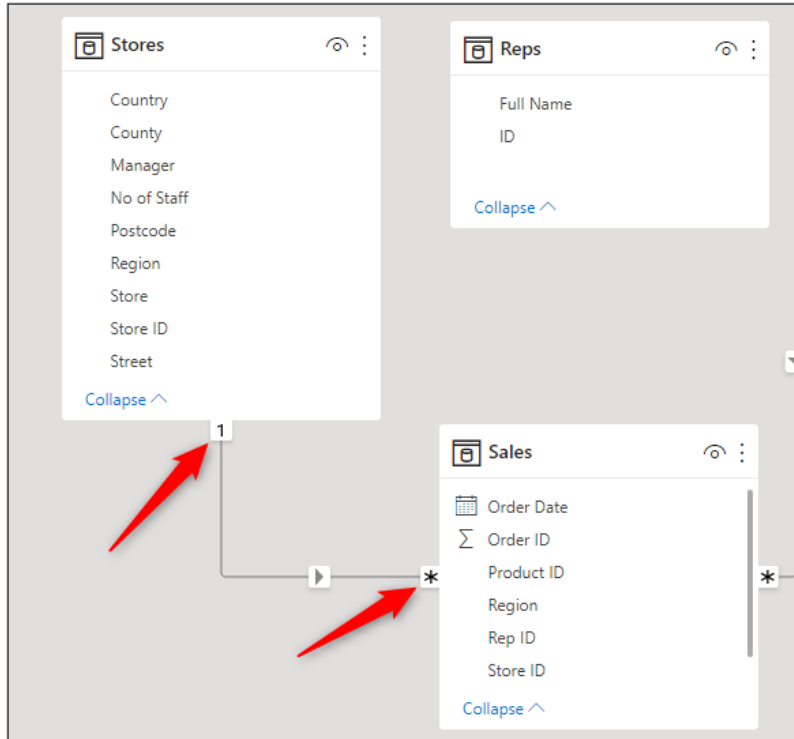


Figure 7.3: One-to-many or many-to-one cardinality between the Stores and Sales tables

There are four types of cardinalities.

1. One-to-many (1:*)
2. Many-to-one (*:1)
3. One-to-one (1:1)
4. Many-to-many (*:*)

A one-to-one cardinality means that the key columns linking the two tables both have unique values. And a many-to-many cardinality means that the key columns can both contain duplicate values.

The most common type of cardinality is the one-to-many or many-to-one. In this model, that is the only cardinality type that we will use.

Creating table relationships

Let us create the relationship between the **Reps** and **Sales** tables of the model.

There are two methods for creating table relationships in Power BI.

In Model view, you can drag and drop from the key column in one table to the key column in the other table. In doing this, Power BI investigates the two columns and assigns the cardinality based on whether the column contains duplicates or not.

Or you can click the **Manage Relationships** button found on the Home tab in the Model view. This method is slower but allows us control over specifying the cardinality and other relationship settings.

Note: The Manage Relationships button is available in all three views of Power BI. And sometimes on more than one tab. It is a very accessible button and is not the only button to be found on multiple tabs and views in Power BI.

For this example, we will use the drag-and-drop method. We will see the Manage Relationships window shortly when we discuss editing and deleting existing table relationships.

1. Switch to the Model view.
2. Click and drag from the **Rep ID** column in the **Sales** table to the **ID** column in the **Reps** table (figure 7.4).
3. Release the button click, and the relationship is established.

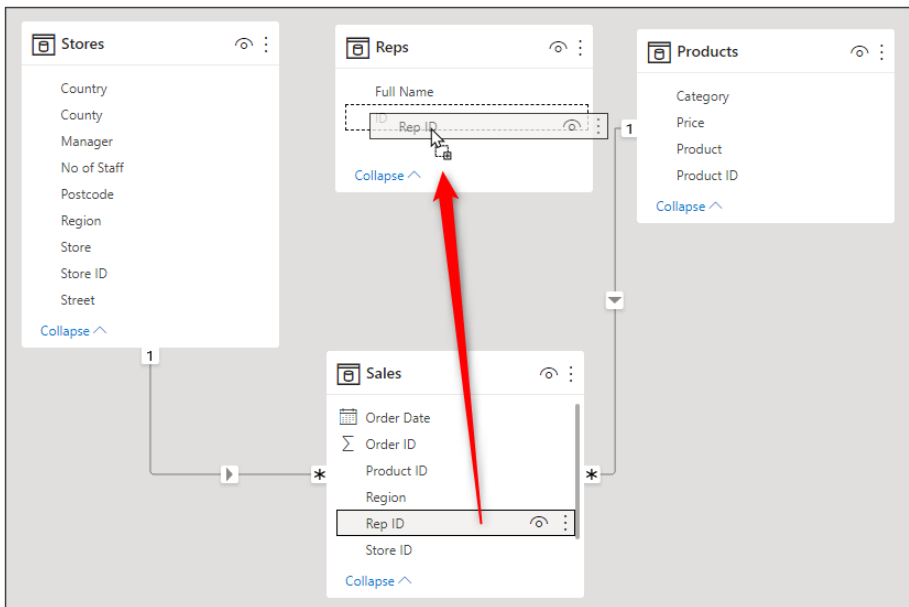


Figure 7.4: Creating a relationship between the Sales and Reps tables

Power BI has defined the relationship cardinality as many-to-one, as shown in figure 7.5, and the line is shown between the two tables illustrates this. The filter direction is also indicated by the arrow on the relationship line.

To quickly see which two columns are used to connect the two tables, hover the mouse cursor over the line connecting them (figure 7.5). The two key fields are highlighted in grey:

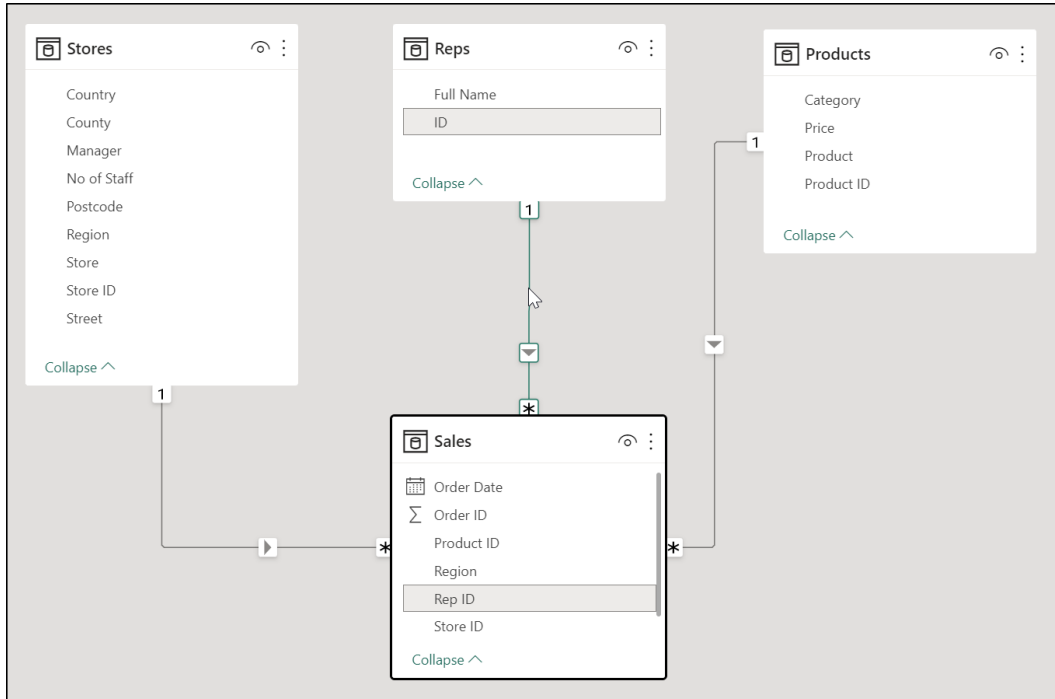


Figure 7.5: Checking the key columns of a table relationship

Back in the Report view, the table visual showing units sold by the sales rep, which was not working previously, now works wonderfully (figure 7.6):

Full Name	Units Sold	Store	Units Sold
Alex Freuer	5308	Aberdeen	8914
Audrey White	8963	Belfast	11737
Bernadette Olusola	5177	Birmingham	8401
Christie Dankov	7852	Brighton	19580
Christopher Hartley	2514	Bristol	8449
Clare Jorquera	2397	Cardiff	9872
Cyndy Bloom	7151	Carlisle	5011
Elizabeth Kendrick	5669	Glasgow	8741
Georgia Keegan	3260	Gloucester	12653
Gina Croft	5071	Guildford	6270
Giovanni Rovelli	7739	Harlow	23046
Isabel De Castro	7324	Hartlepool	11323
Total	215540	Total	215540

Figure 7.6: Units sold by sales rep table now working

Power BI can filter the **Units Sold** field in the **Sales** table by the **Full Name** field in the **Reps** table due to the working relationship between the two tables.

Managing the relationships of a model

Creating relationships between tables of the model is quick and simple to do with the drag-and-drop method in the Model view. However, you may want to edit or delete a table relationship later.

It is possible to edit or delete a relationship using the diagram in the model view.

Right-click on the line of the relationship that you want to change (*figure 7.7*), click **Delete** to remove the relationship, or click **Properties** to open the Edit relationship window (*figure 7.9*).

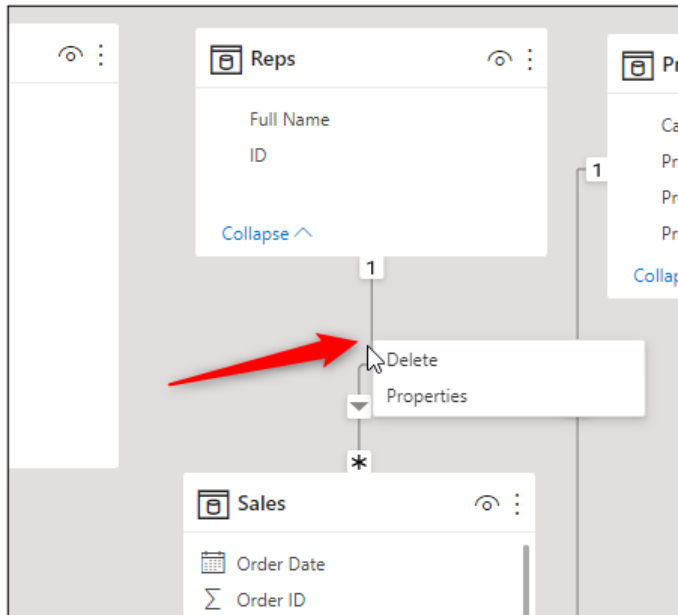


Figure 7.7: Editing a relationship using the diagram in the Model view

However, an easier method to right-click that thin line is to use the **Manage relationships** window (*figure 7.8*).

Click the **Manage relationships** button on the **Home** tab of the Ribbon in Model view (this button is accessible in all three views of PBI Desktop).

All the table relationships of the model are listed in the **Manage relationships** window (Figure 7.8). All the relationships are listed with the **Sales** table on the **From** side and the relevant dimension (lookup) table on the **To** side:

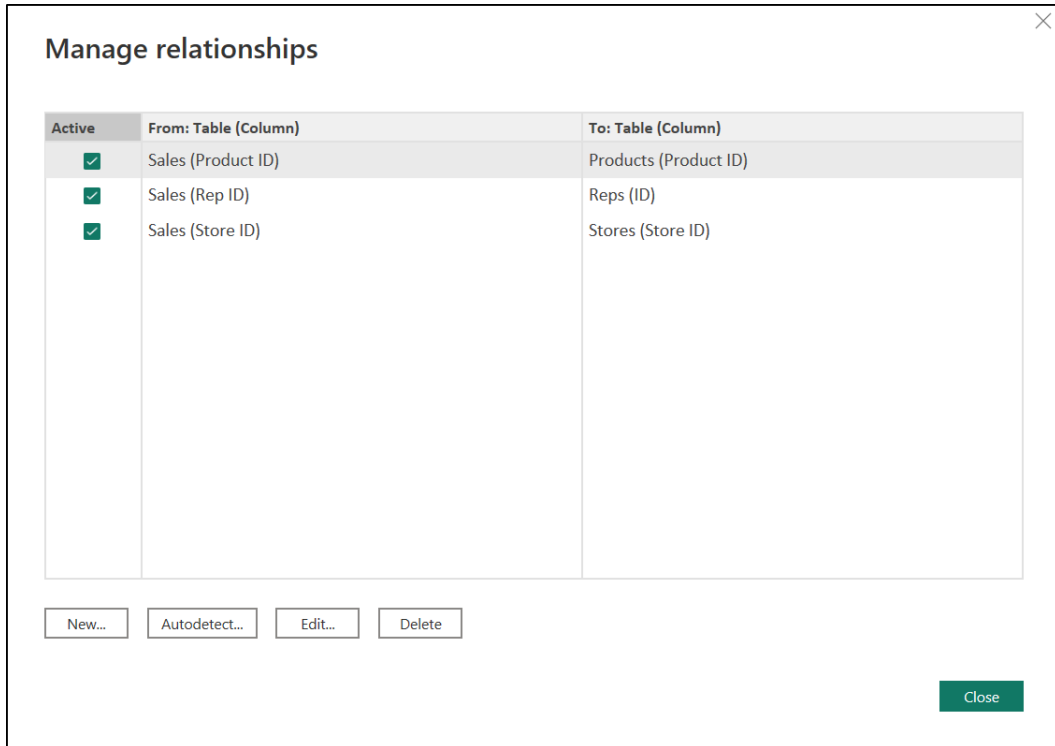


Figure 7.8: The Manage relationships window in Power BI desktop

There are buttons at the bottom of the window to perform different relationship tasks, such as creating new relationships or deleting existing ones.

Click the **Edit** button to open the **Edit relationship** window (figure 7.9).

This window contains everything you need to fine-tune your table relationships. You can change the key columns used in the relationships, their cardinality, cross-filter direction, and specify whether it is the active relationship or not.

Many of these settings go beyond the scope of this book. However, it is important to be aware of the options available and how to access them. Please refer to the following figure:

Edit relationship ✕

Select tables and columns that are related.

Sales ▼

Region	Order ID	Order Date	Product ID	Store ID	Units Sold	Rep ID
West	19302	01 January 2019	R1001	4	30	SR1009
West	19303	01 January 2019	R1001	4	20	SR1009
West	19312	04 January 2019	R1002	4	5	SR1009

Reps ▼

ID	Full Name
SR1302	Christie Dankov
SR0243	Maria Larsson
SR1009	Rita Müller

Cardinality Cross filter direction

Many to one (*:1) Single

Make this relationship active Apply security filter in both directions

Assume referential integrity

OK
Cancel

Figure 7.9: Editing a relationship in Power BI

The columns used to relate the tables can be changed by simply clicking on the columns that you want to use instead. Click **OK** to confirm any changes.

Disabling queries from loading

Not all queries need to be loaded to Power BI for analysis and use in report visuals. Some queries have other purposes. They are used as intermediate steps to only perform the data connections and transformations part of the Power Query process.

In this data model, there are two such queries—**Table001** and **Table002**.

These queries were created when we imported the sales rep data from a PDF file. We appended the data from these two queries into a new query named **Reps**.

So, these queries are very important because the **Reps** query is dependent on them. However, they are intermediate steps and are not required once we are in the Power BI modeling and reporting environment.

Therefore, we will ensure that they are not deleted and continue to refresh with the report but are not loaded to Power BI along with the other queries.

1. Click **Home | Transform data** to open the Power Query Editor.
2. Right-click on the **Table001** query and click the **Enable load** option in the context menu to uncheck it (*figure 7.10*):

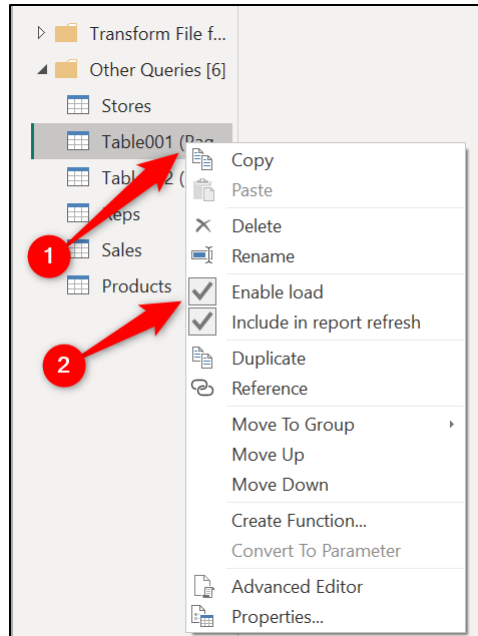


Figure 7.10: Preventing the query from loading to Power BI

Unchecking this option will prevent the query from loading to Power BI as a table. However, it will remain in the Power Query Editor and still perform its role in producing the **Reps** query.

In this example, we will keep the Include in report refresh option checked. This means that the query will be refreshed along with the other queries when the report is either manually or automatically refreshed in PBI Desktop or in the Power BI service.

If you have a query that does not update or does not update very often, it can be a good idea to uncheck the **Include in report refresh** option. This will help improve report refresh time as unrequired queries are not included in the report refresh.

Note: The Enable load and Include in report refresh settings can also be edited in the Properties window of a query.

A message is shown warning you about possible data loss (*figure 7.11*). Preventing the load of a query could break existing visuals in a report. These queries are not used by any visual, so click **Continue**.

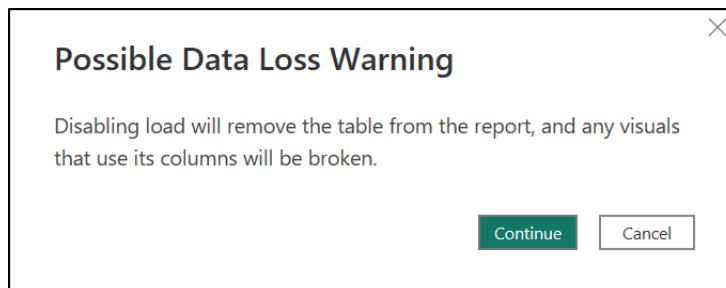


Figure 7.11: Warning that disabling the loading of a query could break report visuals

3. Repeat Steps 2 and 3 for the **Table002** query.

When the queries are prevented from loading, they are shown in italic font (*figure 7.12*).

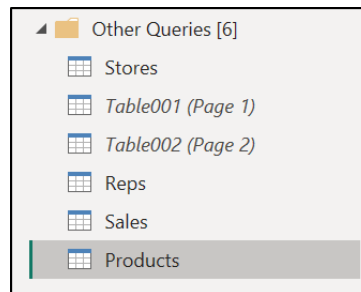


Figure 7.12: Queries that are not set to load or shown in an italic font

4. Click **Home | Close & Apply** to close the editor and load the queries to the Power BI model.

The **Table001** and **Table002** tables are no longer loaded and are not shown in the **Data** pane (*figure 7.13*).

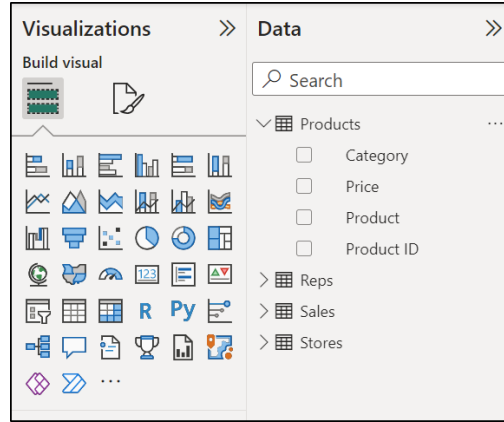


Figure 7.13: Table001 and Tables002 not shown in the Data pane

Formatting table columns

Another element of the modeling process is formatting the numeric columns of your tables in the desired manner.

Figure 7.14 shows the two table visuals that were inserted earlier in the chapter. You can see that the **Units Sold** column in the tables is unformatted.

The whole number data type was specified when we imported and transformed the data in the **Sales** table. But with numbers of this size, they would look nice when formatted to show a thousand separators.

Full Name	Units Sold
Alex Freuer	5308
Audrey White	8963
Bernadette Olusola	5177
Christie Dankov	7852
Christopher Hartley	2514
Clare Jorquera	2397
Cyndy Bloom	7151
Elizabeth Kendrick	5669
Georgia Keegan	3260
Gina Croft	5071
Giovanni Rovelli	7739
Isabel De Castro	7324
Total	215540

Store	Units Sold
Aberdeen	8914
Belfast	11737
Birmingham	8401
Brighton	19580
Bristol	8449
Cardiff	9872
Carlisle	5011
Glasgow	8741
Gloucester	12653
Guildford	6270
Harlow	23046
Hartlepool	11323
Total	215540

Figure 7.14: Unformatted units sold columns in the table visuals

In this model, we will format only the **Price** column in the **Products** table, and the **Units Sold** column in the **Sales** table.

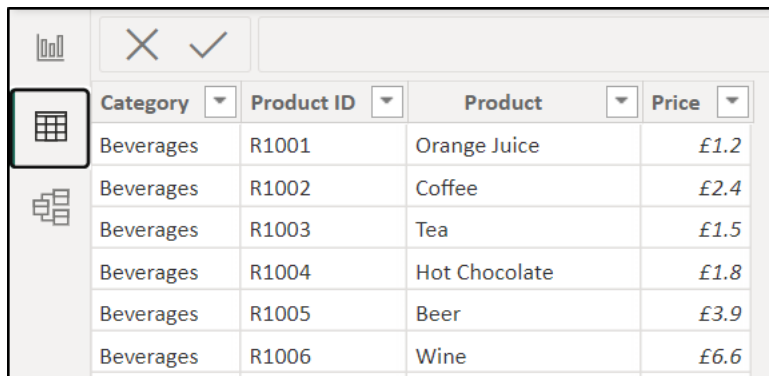
You can format the columns of a table in all three Power BI views (Report, Data, and Model). However, I like to format columns using the Data view. The reason for this is that in the Data view, I can see the data in the columns being formatted.

Now, this is also true, to some extent, of the Report view. Because we can see the formatting of the **Units Sold** column in the table visuals. But the Data view makes it easy to see the data for all table columns.

So, in this example, let us proceed to format the columns using the Data view.

1. Switch to the Data view in Power BI Desktop.
2. Click on the **Products** table in the **Data** pane.

Figure 7.15 shows the **Products** table in the Data view. You can see that the formatting of the **Price** column is a mess. We will format the column to show a currency with two decimal places.



Category	Product ID	Product	Price
Beverages	R1001	Orange Juice	£1.2
Beverages	R1002	Coffee	£2.4
Beverages	R1003	Tea	£1.5
Beverages	R1004	Hot Chocolate	£1.8
Beverages	R1005	Beer	£3.9
Beverages	R1006	Wine	£6.6

Figure 7.15: Unformatted price column shown in the Data view

3. Click on the **Price** column. You can do this in the **Data** pane or on the table column itself in the Data view.

Note: The terms “field” and “column” are effectively interchangeable. In previous versions of Power BI, the Data pane was named the Fields pane, yet when a field is selected, a Column tools tab is shown on the Ribbon. In Excel, when using PivotTables, your columns, and measures are shown in a pane named Fields.

4. Change the number of decimal places to 2 (figure 7.16). You can do this by typing two into the decimal places box provided on the **Column tools** tab or by using the spin buttons to the right of the decimal places box.

The currency shown for the **Price** column in my model is the British pound. This was not specified by me. It was automatically applied due to my regional settings

being the United Kingdom and because the Fixed Decimal Number data type was applied in Power Query.

There is a currency format button in the **Formatting** group of the **Column tools** tab (figure 7.16), so this can easily be changed from the automatically applied currency format if required.

Now, let us change the format of the **Units Sold** column in the **Sales** table.

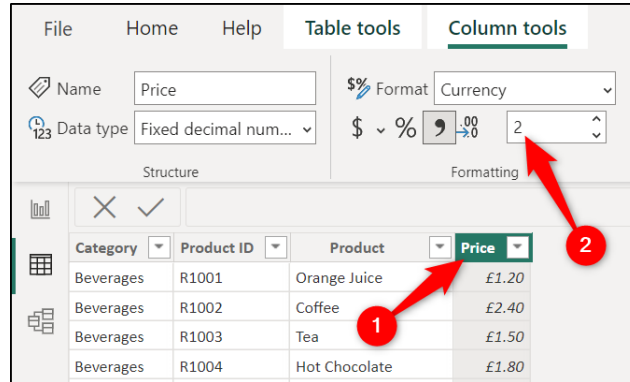


Figure 7.16: Formatting the price column to two decimal places

1. Click on the **Sales** table in the **Data** pane.
2. Click on the **Units Sold** column.
3. Click the thousands separator button (figure 7.17).

Note: The regional settings in Power BI dictate the formatting options shown. The options on your screen may differ from those shown in the images of this book. For example, the comma is shown as the thousand separator for my locale.

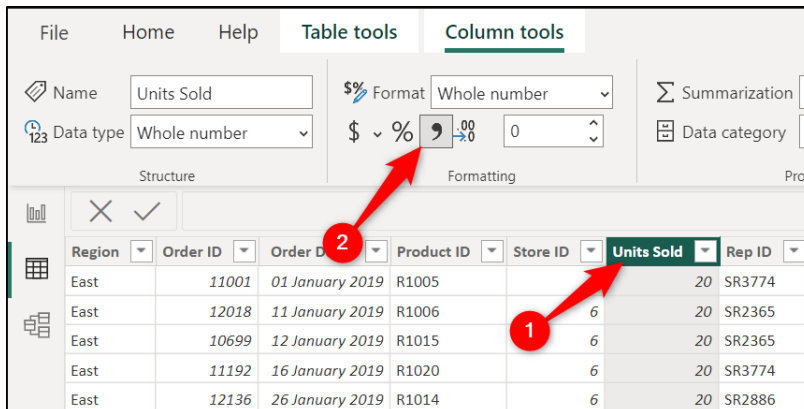


Figure 7.17: Formatting the Units Sold column to display the thousand separator

The thousands separator is applied to the numbers in the **Units Sold** column. However, because the numbers in each row are small, the change cannot easily be seen.

Note: We will not concern ourselves with formatting the Order Date column in this table. The reason for this is that we will create a date table in the upcoming chapter and will be using the columns from that date table in our report visuals. So, formatting this Order Date column would be pointless.

Switch to the Report view to see that the numbers in the **Units Sold** column of the two table visuals are showing the change in format (*figure 7.18*).

Full Name	Units Sold
Alex Freuer	5,308
Audrey White	8,963
Bernadette Olusola	5,177
Christie Dankov	7,852
Christopher Hartley	2,514
Clare Jorquera	2,397
Cyndy Bloom	7,151
Elizabeth Kendrick	5,669
Georgia Keegan	3,260
Gina Croft	5,071
Giovanni Rovelli	7,739
Isabel De Castro	7,324
Total	215,540

Store	Units Sold
Aberdeen	8,914
Belfast	11,737
Birmingham	8,401
Brighton	19,580
Bristol	8,449
Cardiff	9,872
Carlisle	5,011
Glasgow	8,741
Gloucester	12,653
Guildford	6,270
Harlow	23,046
Hartlepool	11,323
Total	215,540

Figure 7.18: Units Sold columns showing their new format in the table visuals

Hiding table fields from the Report view

Hiding fields from the Report view in Power BI is a great way to reduce the clutter of your working environment.

When you are adding fields and measures to visualizations in the Report view, you want a clean and tidy working area. Any field that you do not plan to use in a report visual has no reason to be shown in the Report view.

Duplicate fields can lead to confusion over which field should be used, and having too many fields will cause a lot of scrolling and, therefore, unnecessary aggravation.

So, which fields should you hide? To put it simply, you hide any field that you will not use in visualization.

This typically includes the following:

- The key fields that are used to create the relationships between tables
- Fields that are used to sort other fields/columns
- Date fields, once the date table is set up
- DAX measures that will not be used directly in report visuals

Note: A hidden field can still be used in a DAX calculation, but it needs to be visible to be used in a visual.

Right now, we will hide the key fields that are used in the table relationships. We have no plans to use any of them in a visualization. In the upcoming chapter, we will need to hide a date field and a couple of fields that are used to sort other fields.

You can easily hide fields from the Report view using any of the three Power BI Desktop views (Report, Data, and Model). My preference is to hide fields in the Model view.

1. Switch to the Model view
2. Right-click on the **Product ID** field in the **Sales** table and click **Hide** in the Report view (*figure 7.19*).

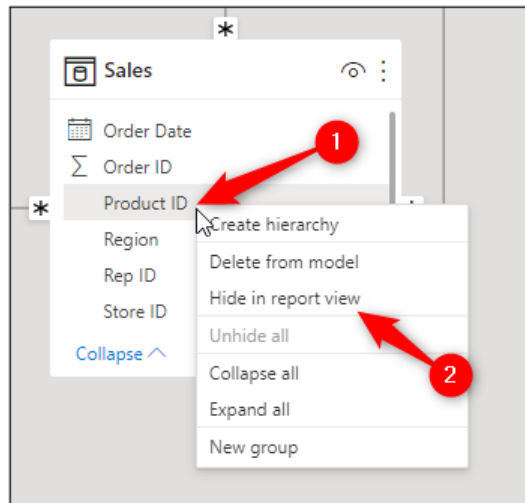


Figure 7.19: Hiding the key fields in the Report view

3. Repeat this step for all fields that need to be hidden.

In *figure 7.20*, all the key fields have been hidden. The Model view indicates a hidden field by displaying an eye icon with a strikethrough. You can also click this eye icon to show and hide a field. Please refer to the following figure:

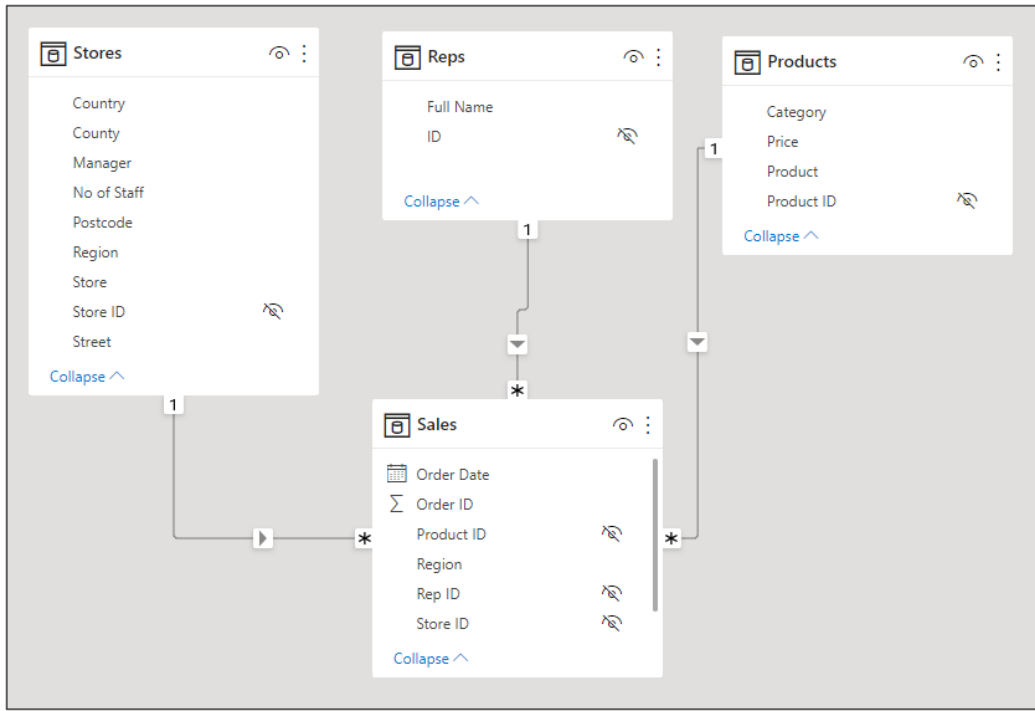


Figure 7.20: All chosen fields have been hidden

These fields are only hidden in the Report view, so the fields are still visible in the Data view. In this view, they are shown with the header text in a light grey font (figure 7.21):

Region	Order ID	Order Date	Product ID	Store ID	Units Sold	Rep ID
East	11001	01 January 2019	R1005	6	20	SR3774
East	12018	11 January 2019	R1006	6	20	SR2365
East	10699	12 January 2019	R1015	6	20	SR2365
East	11192	16 January 2019	R1020	6	20	SR3774
East	12136	26 January 2019	R1014	6	20	SR2886
East	10628	29 January 2019	R1015	6	20	SR1131
East	10705	31 January 2019	R1015	6	20	SR2886

Figure 7.21: Hidden fields are shown in the Data view

In the Report view, the fields are hidden. Figure 7.22 shows a clean and tidy Data pane with all key fields hidden but still performing the important role of relating the different tables of the model:

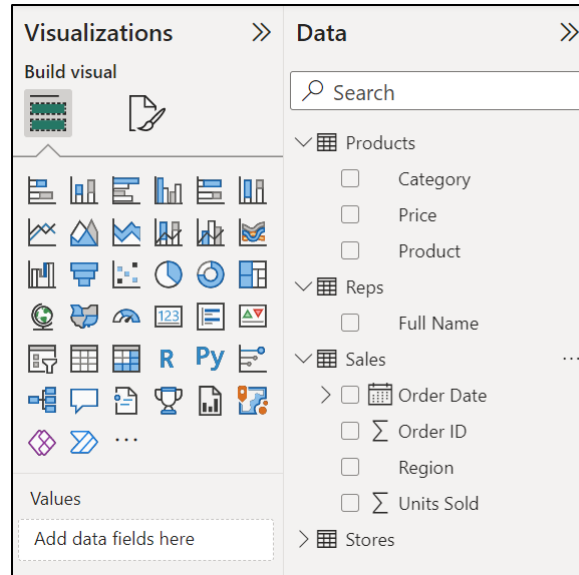


Figure 7.22: The hidden fields are not visible in the Report view

Deleting fields from a model

If a field is not used within the model or the report, then it should be removed.

Loading too many fields to the Power BI model is a common mistake made by Power BI users. If a field is not going to be used in a visual, in a DAX calculation, to relate two tables, or some other purpose, why is it there?

In this model, we have two **Region** fields. One in the **Stores** table and another in the **Sales** table (figure 7.23). There is no advantage to having duplicate fields like this. So, we will remove one.

Which one?

Well, we will remove the **Region** field from the **Sales** table because that table has more rows, so it will have a greater impact on cleaning our model. But not only this, the correct home for the **Region** field is the **Stores** table because it relates to the store. It belongs to that dimension.

Once again, we will use the Model view for this task. However, deleting a field from the model can be achieved using the **Data** pane in any of the three Power BI views.

1. From the Model view, right-click the **Region** field in the **Sales** table.
2. Click **Delete from model** (figure 7.23).

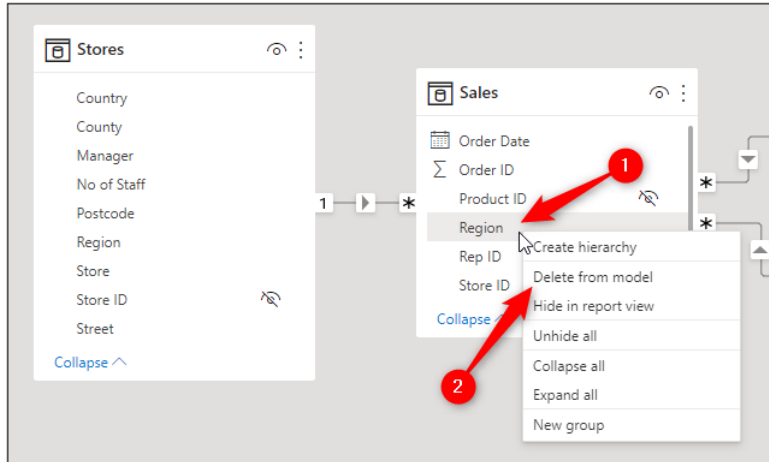


Figure 7.23: Deleting a field from the model

As you may have expected, a warning message appears, checking if you are sure that you want to delete this field from the model (*figure 7.24*).

5. Click **Yes**.

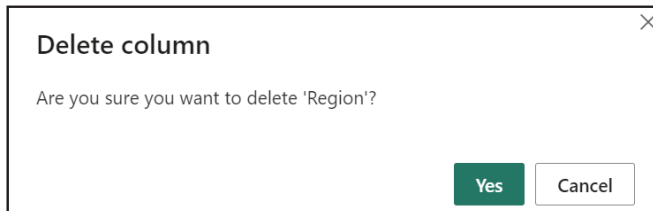


Figure 7.24: Warning message about deleting a column

The field is removed, leaving only the **Region** field in the **Stores** dimension table (*figure 7.25*):

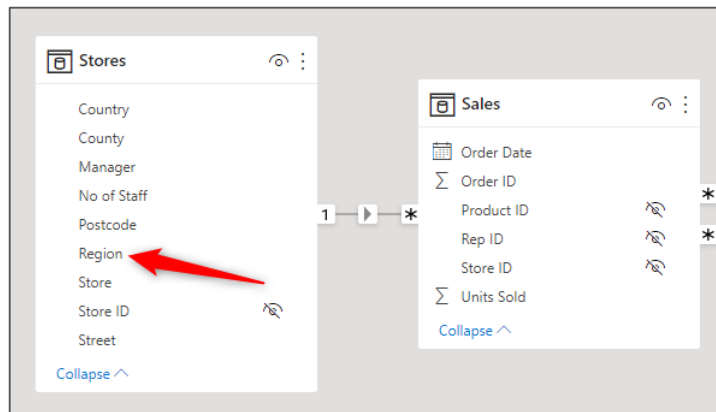


Figure 7.25: Only the Region field in the Stores table remains

Tip: This column could and should have been removed in Power Query when we were importing and transforming the data and before it was loaded to Power BI. However, deleting it at a later stage is not a problem. You can also perform tasks such as changing the data type of a column in Power BI without having to re-open the Power Query Editor.

Conclusion

In this chapter, we learnt why and how to create relationships between the tables of your data model. We also saw how to manage these relationships and edit them later if required.

We then formatted the fields of the tables, hid fields that were not required for the report visuals, and prevented the loading of specific queries.

In the upcoming chapter, we will create a date table for the data model. We will learn why this is important and then create the table using the Data Analysis Expressions (DAX) formula language.

This will be our first sight of DAX in this book, so we will have a gentle yet detailed introduction to the language. We will then use DAX to create the date table and the additional columns that we will use in our report, including year, quarter, financial year, month, day of week, and more.

Questions

Here are some questions to test what you have learnt in this chapter.

1. Which of the following is a type of relationship cardinality?
 - a. One-to-one
 - b. Many-to-one
 - c. Many-to-many
 - d. All of the above.
2. By default, Power BI is set to autodetect table relationships when data is loaded.
 - a. True
 - b. False

3. Which fields should not be hidden in Power BI?
 - a. A field that is used as a key field in a table relationship
 - b. A field needs for a DAX calculation
 - c. A field that will be used in a report visualization
 - d. A field used to sort another field
4. Preventing a query from loading to Power BI means it does not refresh when the report is refreshed.
 - a. True
 - b. False
5. You can format columns of a table using any of the three Power BI views.
 - a. True
 - b. False

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



CHAPTER 8

Creating a Date Table

Introduction

In this chapter, we continue building the data model of our report by adding a date table (also known as a calendar table or a date dimension).

We begin this chapter by detailing why you need a date table in your model and what one consists of. This is a very common question from Power BI users.

We will then begin to create the date table using the DAX formula language. We start with an introduction to the DAX language and then proceed to build the date table and create the different columns that we need for our report.

Next, we will sort a few of the table columns by another column. A cool feature of Power BI that adds flexibility to how we sort text values in a column.

Finally, we mark the table as a date table and create a relationship between the date table and the sales table in our model.

Structure

In this chapter, we will cover the following topics:

- Why create a date table?

- Introduction to the DAX formula language.
- CALENDAR and CALENDARAUTO functions.
- Creating additional date fields for our report, including year, month, quarter, and more.
- Sorting columns by another column.
- Marking a table as a date table.

Objectives

After reading this chapter, you will know how and why to create a date table in your Power BI data model. You will have a solid understanding of the benefits of a custom date table versus the built-in date hierarchy automatically created by Power BI.

By the end of this chapter, you will have gained some experience writing DAX formulas to create tables and columns in Power BI. This prepares us nicely for the upcoming chapter, where we start to write DAX measures.

Why create a date table?

To perform date-based reporting in Power BI, you need a date table. Examples of date-based reporting include:

- Year-to-date calculations.
- This period is compared to the same period in last year's analysis.
- Total revenue at the weekend compared to during the week.
- Total expenses by fiscal year, quarter, and month.

A date table consists of a column containing date values and additional columns for each of the required date attribute fields, such as year, quarter, month, and so on.

The date column must contain a complete set of dates that span a whole year (this could be a fiscal year). The column must not contain duplicates, as it will be used as the key field in table relationships or have any gaps. Each row of the table represents a single date in the model.

Power BI provides a built-in date table that does this for us. This is great for beginners. They can immediately start analyzing their data using date-based fields such as year, quarter, and month.

This date table is hidden, so you will not see it in the same way that you can see the other physical tables of your data model. However, the presence of this date table is shown by the hierarchy applied to a date field (*figure 8.1*).

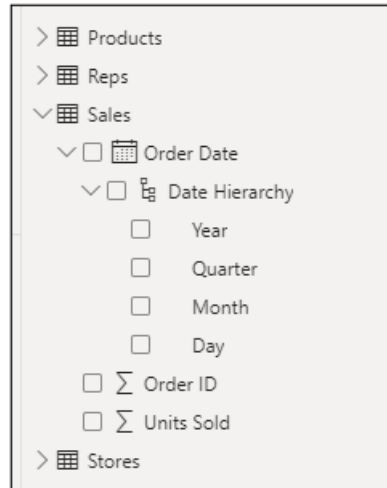


Figure 8.1: Date hierarchy automatically created for the order date field of the model

This automatic date table creation can be very useful for beginners.

Figure 8.2 shows the **Month** field from the hierarchy being used in a table visual to sum units sold by the month. Also, the **Order Date** field is used in the axis of a Clustered column chart. The hierarchy has created drill-down functionality on the column chart. This is shown by the arrow buttons along the top of the visual:

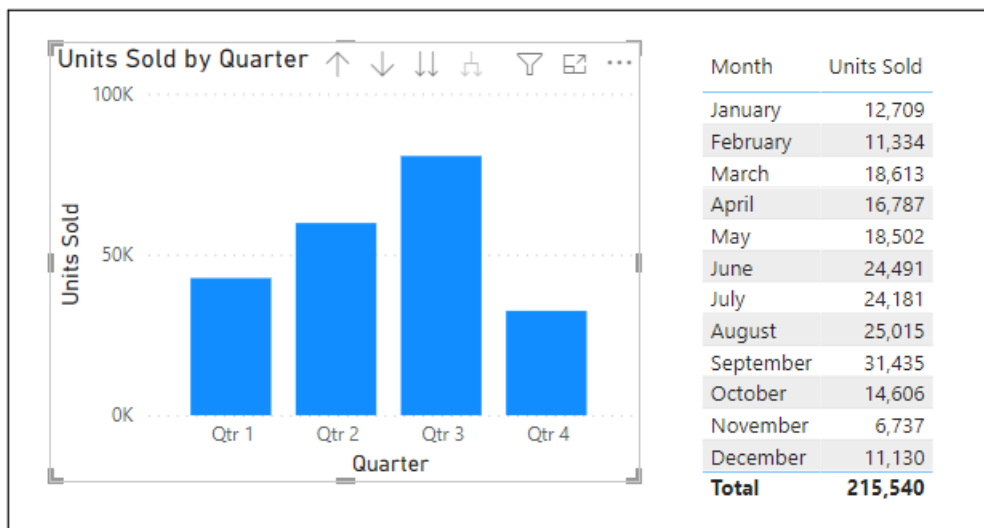


Figure 8.2: Default date table fields being used in visuals

However, there are a couple of issues with the automatically generated date tables.

First, a hidden date table is created for every date field in your model. So, if you have eight date fields from different tables in your model. For example, the due date, payment date, refund date, date joined, and so on. Power BI creates eight hidden date tables. This can bloat your model.

Second, the fields provided in the hierarchy are limited. The default date table only provides year, quarter, month, and day of month fields. So, if you want to perform analysis using fields such as day of week, fiscal quarter, or week number of the year, then this table is not sufficient.

Note: You can disable the setting to automatically create these date tables in Power BI. Click File | Options and settings | Options | Data Load (Global) | Auto date/time for new files. This setting can also be disabled in the current file only by clicking File | Options and settings | Options | Data Load (Current File) | Auto date/time.

Creating your own custom date table ensures that you only have one date table in the model. This avoids the bloating of the model caused by many hidden tables. It also provides consistency and simplicity for those using the model. For example, there will only be one month field instead of a month field in every date field of the model.

It also ensures that we have all the date attribute fields that we need for our analysis—day of the week, fiscal quarter, and so on. *Figure 8.3* shows an image of a completed date table named **Calendar** visible in the **Data** pane of the Report view:

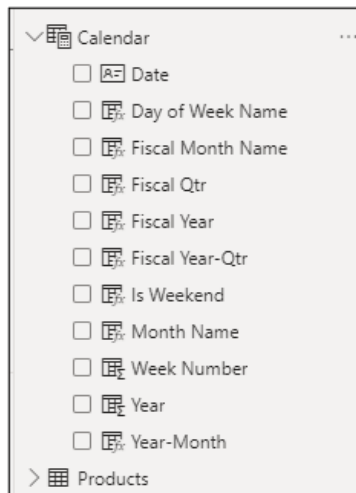


Figure 8.3: Custom date table with all attribute fields ready for use in the report

Custom date tables can be created using Power Query or by using DAX in Power BI. In this chapter, we will stay in Power BI and use DAX to create the date table.

Introduction to DAX

Data Analysis Expressions (DAX) is a very rich formula language that takes a lot of practice to learn. There are some advanced concepts involved in DAX that can be difficult when you are new to the language.

However, do not be deterred. DAX is fun. And if you are familiar with writing worksheet formulas in Excel, the DAX formula language is syntactically very similar.

There are three types of DAX calculation in Power BI—calculated tables, calculated columns, and measures.

When typing a formula, a helpful IntelliSense list appears with the functions, tables, table columns, and measures appropriate for that formula (*figure 8.4*):

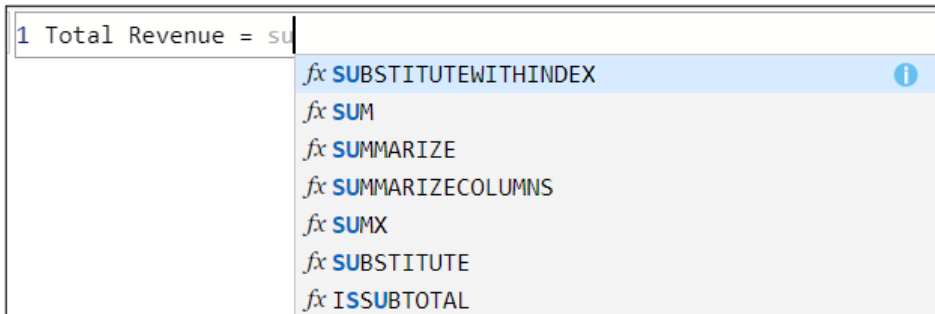


Figure 8.4: IntelliSense list of functions as you type a DAX formula

Notice the information icon to the right of the function name to get further details on that function.

A complete list of DAX functions can be found at the DAX Reference Library (<https://docs.microsoft.com/en-us/dax/dax-function-reference>). This list is constantly updated.

A functions' arguments are listed in a tooltip as you type (*figure 8.5*). Optional arguments are recognized by the square brackets around their name (the **ResultIfFalse** argument is optional in this example). This tooltip helps to keep track of where you are in a function.

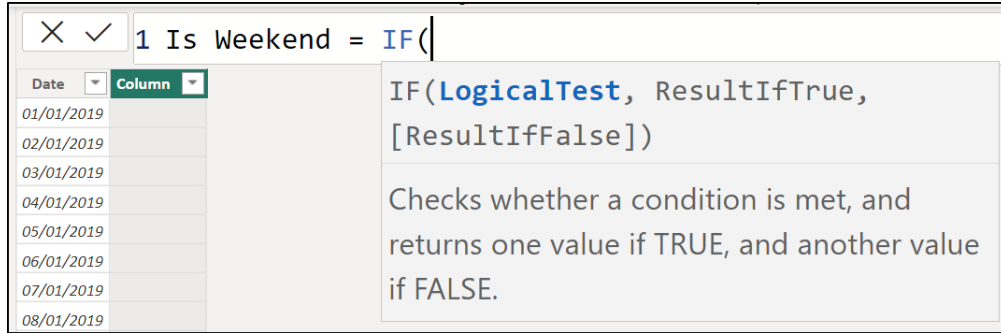


Figure 8.5: The arguments of a function shown in a tooltip

An argument can have its own list of options. Click the arrow icon for further detail on the options available (figure 8.6).

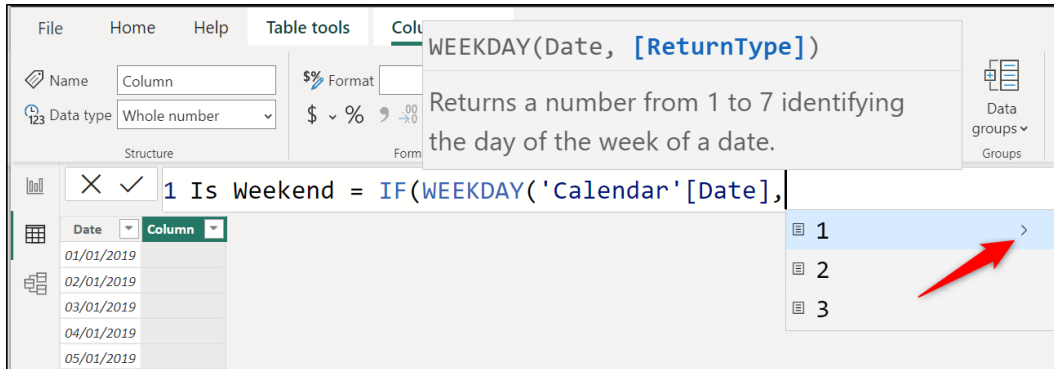


Figure 8.6: Get further information on the options available for an argument

The formula bar in Power BI allows the use of the *Shift + Enter* keyboard shortcut to insert a line break and the *Tab* key to indent the lines of a formula. It also has line numbers for the different lines of your formula.

Large DAX formulas should be presented in an easy-to-read manner. You will get used to this as you see further examples of DAX formulas.

CALENDARAUTO and CALENDAR functions

Files: `sales-report-ch-8-start.pbix`

There are two functions in DAX that can be used to create a date table. The **CALENDARAUTO** and **CALENDAR** functions are used to create a calculated table with a single distinct column of dates that will form the spine of our date table.

- **CALENDARAUTO**: creates a column of dates that spans all dates between a start and end date that it automatically calculates from all date fields in the model.
- **CALENDAR**: creates a column of dates that spans all dates between a specified start and end date. This function gives us the control to define a date range or calculate one with DAX functions.

It is best to switch to the Data view in Power BI for creating the date table. In this view, we will see the results of our calculated tables and columns.

1. Switch to the Data view in Power BI
2. Click **Table tools | New table** (figure 8.7):

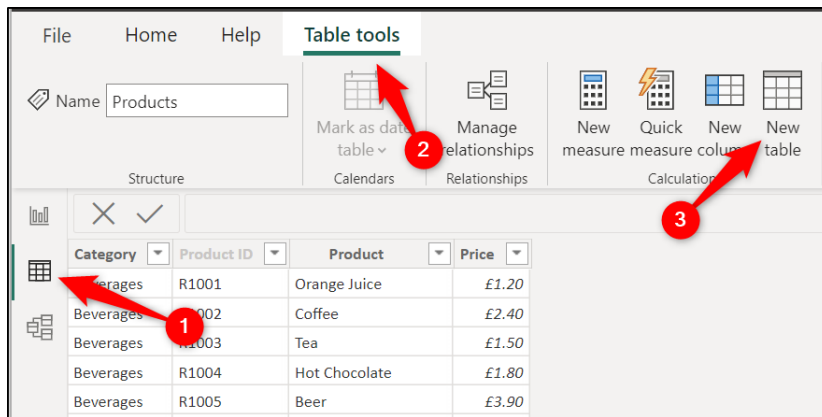


Figure 8.7: Creating a new table using DAX

Note: Notice the buttons for the three types of DAX calculation here—New measure, New column, and New table.

The new table appears in the **Data** pane (A–Z order) with the calculated table icon beside it. The Formula bar is activated and prompts for the table to be assigned a name and the formula to be written, as shown in figure 8.8:

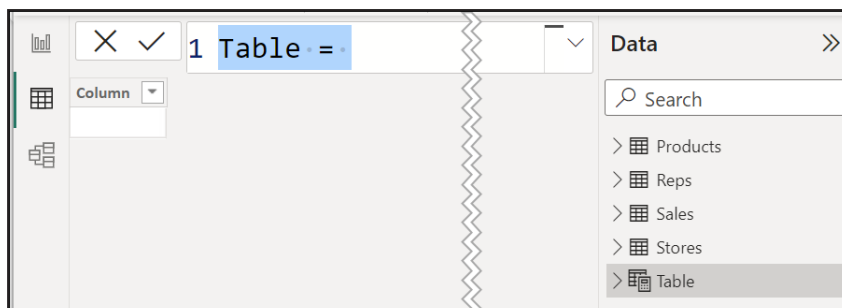


Figure 8.8: New table shown in the Data pane and the Formula bar

CALENDARAUTO function

The **CALENDARAUTO** function is certainly the easiest way to generate the all-important single column of distinct dates than spans complete years.

CALENDARAUTO calculates the date range required for the date column automatically by checking all date fields in the model. It will find the oldest date across all date fields, the newest date across all date fields, extend them to make a complete year if necessary, and then fill in any missing dates so that there are no gaps.

The following is the syntax of the **CALENDARAUTO** function:

```
=CALENDARAUTO([FiscalYearEndMonth])
```

There is an optional argument for the fiscal year-end month. By entering the fiscal year-end month into the function, **CALENDARAUTO** will ensure that the date range covers whole fiscal years. For example, entering the following formula would create a list of dates from October 1 to September 30 for all the fiscal years in the dataset.

```
=CALENDARAUTO(9)
```

If this is omitted, the **CALENDARAUTO** function will ensure the date range covers whole calendar years of January 1 to the December 31.

In *figure 8.9*, the following formula has been entered into the Formula bar for creating a new table in our sales report.

```
Calendar = CALENDARAUTO()
```

The table is named **Calendar** and returns dates from January 1, 2019 to December 31, 2019. The values have a date/time data type applied by default.

The sales report data that we are working with contains dates for the year 2019 only.

If it did contain sales data ranging from February 10, 2018 to October 31, 2020, the **CALENDARAUTO** function would have returned all dates for the date range January 1, 2018 to December 31, 2020. Please refer to the following figure:

Date
01/01/2019 00:00:00
02/01/2019 00:00:00
03/01/2019 00:00:00
04/01/2019 00:00:00
05/01/2019 00:00:00
...
28/01/2019 00:00:00
29/01/2019 00:00:00
30/01/2019 00:00:00
31/01/2019 00:00:00

Figure 8.9: Column of dates generated by CALENDARAUTO

We do not have time data in this model, so we will change the data type to date only and format the dates to how we would like them to appear.

1. Click the **Date** column to select it, if necessary
2. On the **Column tools** tab, click the list arrow for the **Data type** option and click **Date** (figure 8.10).
3. Click the list arrow for the **Format** option and click the format you want to apply. I have applied the short date format in this example (Power BI applies the Long Date format by default to columns with a date data type).

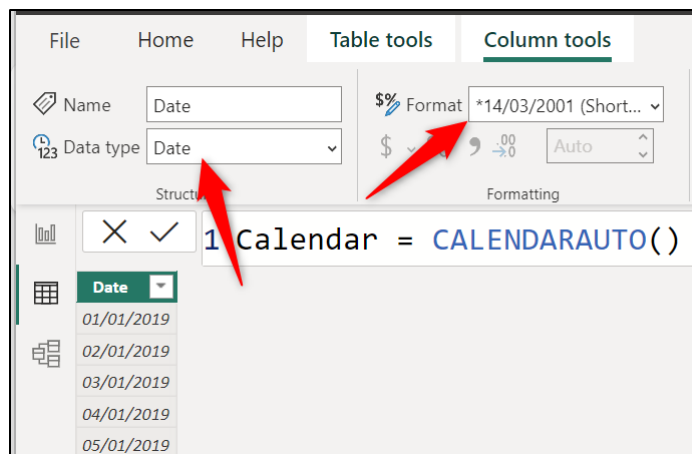


Figure 8.10: Changing the column's data type and format

This is perfect for the data we are using in our sales report, but let us see an example of creating a list of dates with CALENDARAUTO using fiscal years.

In *figure 8.11*, the following formula is used to create a column of dates than spans whole fiscal years from April 1 to March 31. Number 3 has been entered for the **FiscalYearEndMonth** argument.

Calendar = CALENDARAUTO(3)

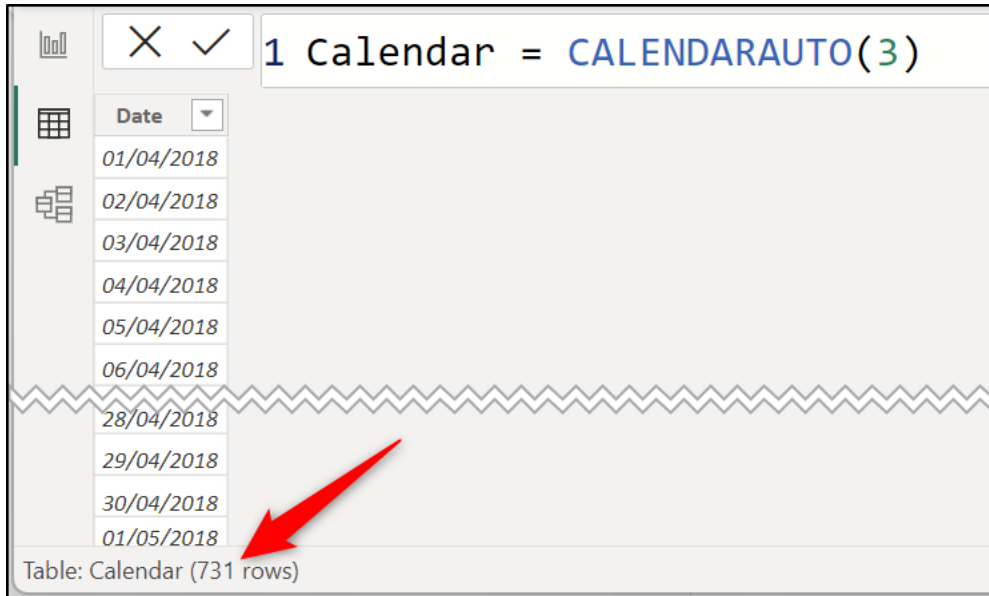


Figure 8.11: CALENDARAUTO returning dates for fiscal years

This has been applied to our sales report data which contains dates from 2019 only. Notice that two years of dates are returned (731 distinct dates). **CALENDARAUTO** automatically spans complete fiscal years starting from April 1, 2018 to ending on March 31, 2020.

The **CALENDARAUTO** function is the fastest and simplest way to create a table and generate the column of dates required. However, there are factors to bear in mind.

The **CALENDARAUTO** function looks at all date fields of the model to ascertain the start and end dates for the column of dates. When using this approach, you should be careful that other date fields do not distort the results. A common example of this is date fields such as dates of birth and dates that an employee started with a company.

If you plan to create a report using sales data for the last three years, you do not require dates that span outside of that three-year range for time intelligence. You may require a date column to calculate fields such as age or years with the company, but not for time intelligence calculations in the report.

To avoid date fields such as this causing confusion, you can prevent these date fields from being loaded to the model (covered in *Chapter 7, Creating the Data Model*) or opt for the `CALENDAR` function over `CALENDARAUTO` to gain more control.

CALENDAR function

The `CALENDAR` function is used when you want to define the date range for the column of dates. This can be done by typing a fixed date range in the function or, for a more dynamic solution, using formulas to calculate the start and end date to be used.

The following is the syntax of the `CALENDAR` function:

```
CALENDAR(StartDate, EndDate)
```

To provide a fixed date range for the complete calendar year of 2019 required for our sales report, the following formula could be used:

```
Calendar = CALENDAR(DATE(2019,01,01),DATE(2019,12,31))
```

The `DATE` function is used within both the `StartDate` and `EndDate` arguments to enable the entering of date values. The elements of a date are entered in year, month, and day order within the `DATE` function.

Tip: You can make the text in the Formula bar larger by pressing `Ctrl` and scrolling the mouse wheel simultaneously. No mouse, no problem. You can also do this by pressing the `Ctrl + +` keys.

This formula works as a treat for what we need. However, if we required a long-term solution where the date table would automatically adjust its date range when refreshed, DAX expressions can be used to calculate the start and end dates.

In *figure 8.12*, the following formula is used to calculate the start and end dates using the `Order Date` field in the `Sales` table. The formula has been split into multiple lines to make it easier to digest. This is good practice when writing DAX expressions.

```
Calendar =
```

```
    CALENDAR(  
        DATE(YEAR(MIN(Sales[Order Date])),1,1),  
        DATE(YEAR(MAX(Sales[Order Date])),12,31)  
    )
```

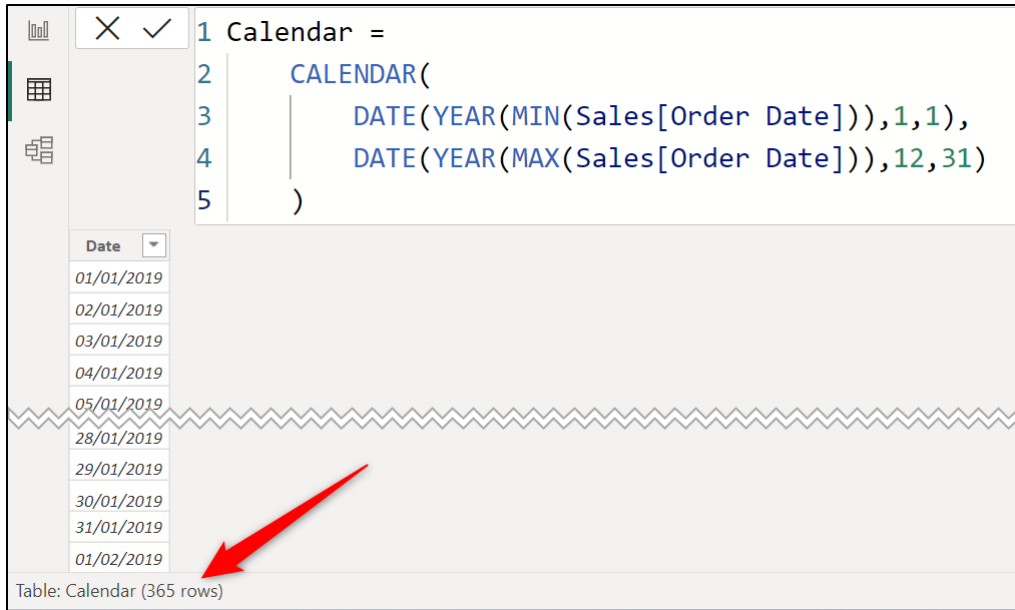


Figure 8.12: Creating a date table with the CALENDAR function

In the formula, the **MIN** and **MAX** functions are used to return the smallest and largest dates from the **Order Date** column. The **YEAR** function then extracts the year from each of these dates to return the start and end years.

The month and day are entered as fixed values for January 1 and December 31.

You can see how the **CALENDAR** function provides more control when creating a date table. It enabled us to focus on the **Order Date** field when calculating the date range. But with **CALENDARAUTO**, it looked at all date fields in the model to calculate the date range.

Remember, this date column must contain a distinct column of dates for full years. Now, that does not need to range from January to December. We could have specified any start and end date.

Creating additional date columns

With the date column created, we will now proceed to create all the date attribute columns that we will need for our reports. These additional columns will be used for filtering and slicing the data in our report.

For this date table, we will create columns including year, month, year-month, day of week, fiscal year, fiscal quarter, and week number of year.

Note: Many of these columns will not be used in the report that we create in this book. They are shown in this chapter for demonstration purposes only.

You could create many more columns, and you can see examples of this in different date tables on the Web. This date table includes some of the most common date attribute fields one would want.

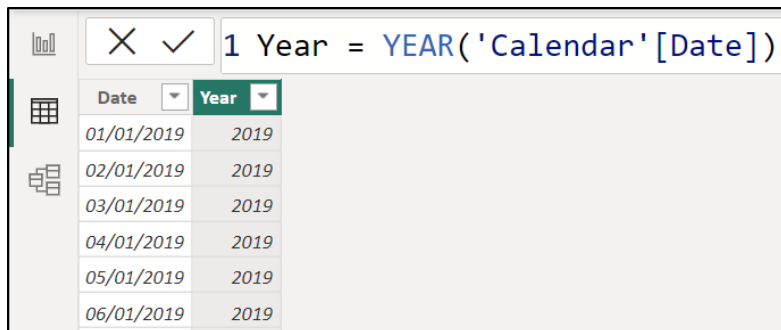
To insert a calculated column, click **Home** | **New column** or **Table tools** | **New column**.

Year

Inserting a year column is straightforward, as the function is simply **YEAR**. The **YEAR** function needs only to be provided with the column that contains the date values, and it will extract the four-digit year value.

The following formula creates a column named **Year** and extracts the year for our **Calendar** table (*figure 8.13*).

Year = YEAR('Calendar'[Date])



Date	Year
01/01/2019	2019
02/01/2019	2019
03/01/2019	2019
04/01/2019	2019
05/01/2019	2019
06/01/2019	2019

Figure 8.13: YEAR function to extract the four-digit year from a date

When referencing a column, an explicit column reference that includes the table name is always used, even when the new column and referenced column are in the same table.

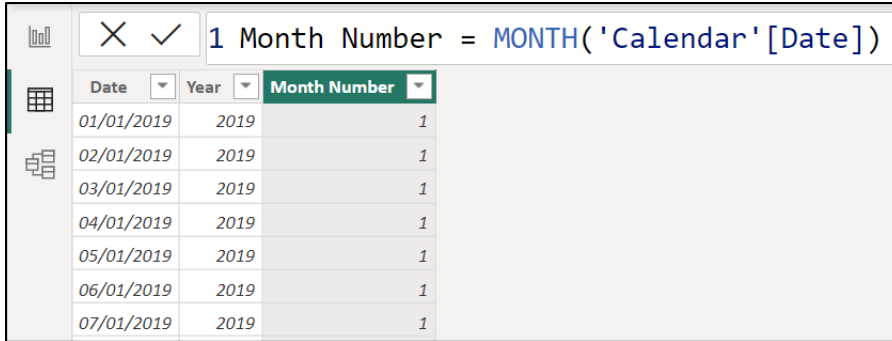
The IntelliSense list will appear as you type the formula to assist with selecting columns. You cannot click the column to select it.

Month

For the month, we will insert two columns. One column for the month number and another for the month name. You may be thinking, why do we need both? Well, the month number column will play an important role soon.

The formula for the month number is like that used for the year. The **MONTH** function will extract the month number from a given date (*figure 8.14*).

Month Number = MONTH('Calendar'[Date])



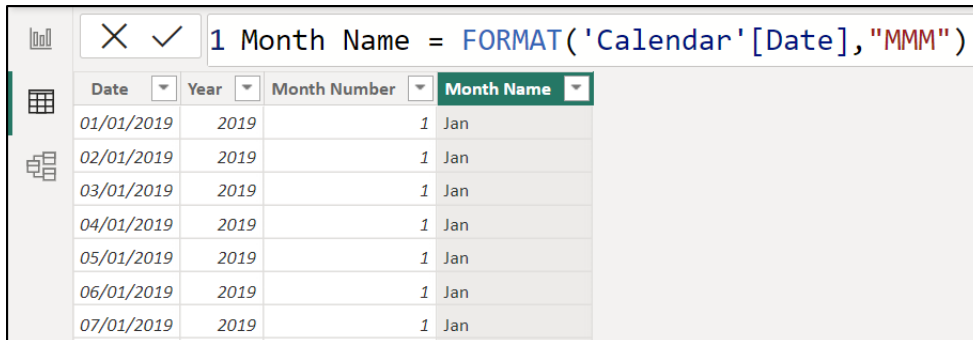
Date	Year	Month Number
01/01/2019	2019	1
02/01/2019	2019	1
03/01/2019	2019	1
04/01/2019	2019	1
05/01/2019	2019	1
06/01/2019	2019	1
07/01/2019	2019	1

Figure 8.14: Inserting the month number with the MONTH function

For the month name, the **FORMAT** function will be used. The **FORMAT** function converts a value to text and applies a number format.

In *figure 8.15*, the following formula creates a column named “Month Name” and shows the value in the **Date** column in a short month name format (MMM).

Month Name = FORMAT('Calendar'[Date], "MMM")



Date	Year	Month Number	Month Name
01/01/2019	2019	1	Jan
02/01/2019	2019	1	Jan
03/01/2019	2019	1	Jan
04/01/2019	2019	1	Jan
05/01/2019	2019	1	Jan
06/01/2019	2019	1	Jan
07/01/2019	2019	1	Jan

Figure 8.15: Month name column using FORMAT

You may prefer the long month name format. For example, January, February, and March. For this, enter the format string “MMMM”. This could be created in addition to the short month name, and then you have both columns available for filtering data in a report.

The short month name format is suggested, especially in a chart’s axis, as it uses less space and has a consistent length.

Note: The string of characters to determine the month name format can be entered in uppercase or lowercase.

Year and month

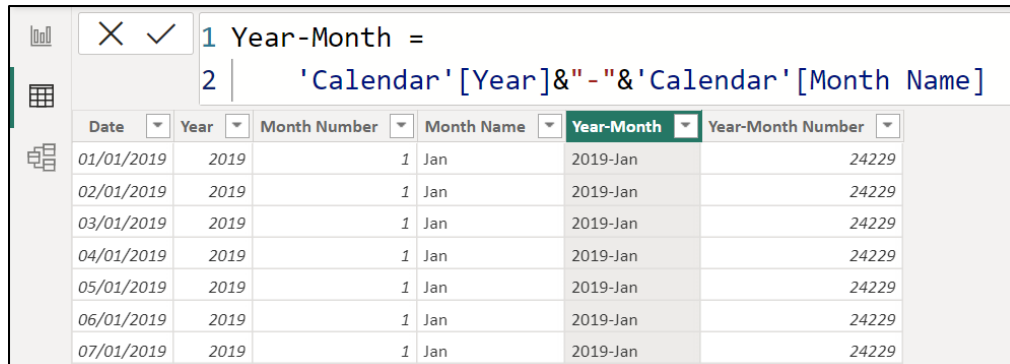
To create a Year–Month column, we will merge the content of the **Year** and **Month Name** columns together and separate them with the “–” delimiter.

In *figure 8.16*, the following formula is used to create the **Year–Month** column. The ampersand (&) is used to concatenate the three values together into a single string.

Year–Month = ‘Calendar’[Year]&”-”&’Calendar’[Month Name]

We will also create a merged year and month number column using the following formula. The result of this formula is also shown in *figure 8.16*.

Year–Month Number = ‘Calendar’[Year] * 12 + ‘Calendar’[Month Number]



The screenshot shows a spreadsheet interface. At the top, a formula bar contains the following text:

```
1 Year-Month =
2 'Calendar'[Year]&"-"&'Calendar'[Month Name]
```

Below the formula bar is a table with the following columns and data:

Date	Year	Month Number	Month Name	Year-Month	Year-Month Number
01/01/2019	2019	1	Jan	2019-Jan	24229
02/01/2019	2019	1	Jan	2019-Jan	24229
03/01/2019	2019	1	Jan	2019-Jan	24229
04/01/2019	2019	1	Jan	2019-Jan	24229
05/01/2019	2019	1	Jan	2019-Jan	24229
06/01/2019	2019	1	Jan	2019-Jan	24229
07/01/2019	2019	1	Jan	2019-Jan	24229

Figure 8.16: Merged year and month name column

The **Year–Month Number** column is a whole number data type because it is the result of a mathematical calculation. This is important so that the column can be used to sort the **Year–Month** column later in this chapter.

The formula in the **Year–Month Number** column produces an incremental number sequence without gaps. The multiplied by 12 part of the formula ensured that it would remain a perfect sequence across multiple years.

Day of week name

We will return two columns for the day of the week, just like we did with the previous two date attribute columns. One column for the day of week number and another for the day of week name.

The **WEEKDAY** function will be used to identify the day of the week for a given date. This will be returned as an index number that represents the day of the week based on a given start day.

The second argument of **WEEKDAY** allows you to specify the start day of the week. The following three options are available:

- Option 1 states that Sunday = 1 to Saturday = 7
- Option 2 states that Monday = 1 to Sunday = 7
- Option 3 states that Monday = 0 to Sunday = 6

In *figure 8.17*, the following formula is used to return the day of week number for the values in the **Date** column. Option 2 is used to specify that the week begins on a Monday, starting with an index number of 1.

Day of Week Number = WEEKDAY('Calendar'[Date],2)

Date	Year	Month Number	Month Name	Year-Month	Year-Month Number	Day of Week Number
01/01/2019	2019	1	Jan	2019-Jan	24229	2
02/01/2019	2019	1	Jan	2019-Jan	24229	3
03/01/2019	2019	1	Jan	2019-Jan	24229	4
04/01/2019	2019	1	Jan	2019-Jan	24229	5
05/01/2019	2019	1	Jan	2019-Jan	24229	6
06/01/2019	2019	1	Jan	2019-Jan	24229	7
07/01/2019	2019	1	Jan	2019-Jan	24229	1

Figure 8.17: Day of week number column with WEEKDAY

The **FORMAT** function can be used to return the day of the week name, just as we did with the **Month Name** column (*figure 8.18*).

Day of Week Name = FORMAT('Calendar'[Date], "DDD")

Date	Year	Month Number	Month Name	Year-Month	Year-Month Number	Day of Week Number	Day of Week Name
01/01/2019	2019	1	Jan	2019-Jan	24229	2	Tue
02/01/2019	2019	1	Jan	2019-Jan	24229	3	Wed
03/01/2019	2019	1	Jan	2019-Jan	24229	4	Thu
04/01/2019	2019	1	Jan	2019-Jan	24229	5	Fri
05/01/2019	2019	1	Jan	2019-Jan	24229	6	Sat
06/01/2019	2019	1	Jan	2019-Jan	24229	7	Sun
07/01/2019	2019	1	Jan	2019-Jan	24229	1	Mon

Figure 8.18: FORMAT function returning the day of week name

Again, the three non-digits of “DDD” are used to display the short name for the day of the week. The full name could be shown by using the format string “DDDD”, if preferred.

Is it a weekend date or not?

As part of our reporting, it would be nice to know the sales performance at the weekend compared to other weekdays. To help with this, we will create a column to display “Yes” or “No”, as to whether the date is a weekend date or not. For this example, the weekend is a Saturday and Sunday.

The following formula uses an IF function to test if the **Day of Week Number** column contains a value greater than 5. If it does, then display “Yes”; otherwise, display “No” (figure 8.19).

Is Weekend = IF('Calendar'[Day of Week Number]>5,"Yes","No")

The screenshot shows a Power BI interface. At the top, a formula bar contains the DAX formula: `1 Is Weekend =` and `2 IF('Calendar'[Day of Week Number]>5,"Yes","No")`. Below the formula bar is a table with the following columns: Year, Month Number, Month Name, Year-Month, Year-Month Number, Day of Week Number, Name, and Is Weekend. The table contains 8 rows of data for the year 2019, all for the month of January. The 'Day of Week Number' column shows values 1 through 7. The 'Is Weekend' column shows 'No' for days 1-5 and 'Yes' for days 6 and 7.

Year	Month Number	Month Name	Year-Month	Year-Month Number	Day of Week Number	Name	Is Weekend
2019	1	Jan	2019-Jan	24229			No
2019	1	Jan	2019-Jan	24229			No
2019	1	Jan	2019-Jan	24229			No
2019	1	Jan	2019-Jan	24229			No
2019	1	Jan	2019-Jan	24229			Yes
2019	1	Jan	2019-Jan	24229			Yes
2019	1	Jan	2019-Jan	24229			No

Figure 8.19: Is the date a weekend or not

Remember, when using the **WEEKDAY** function previously, we specified that the week begins on a Monday. Therefore, Saturday and Sunday are the day of week numbers 6 and 7, respectively.

Fiscal years

We will not be using any fiscal date fields in the reports that we create, but for demonstration purposes, let us look at how to create some fiscal date attribute fields starting with the fiscal year.

Tip: If you are using fiscal data, the column of dates in the date table should contain all dates for the fiscal period.

For this example, the fiscal year spans from April to March.

The following formula returns the fiscal year in the format “YYYY/YYYY” that is, 2017/2018 (figure 8.20).

Fiscal Year =

```
IF(MONTH('Calendar'[Date])>=4,
YEAR('Calendar'[Date])&"/"&YEAR('Calendar'[Date])+1,
YEAR('Calendar'[Date])-1&"/"&YEAR('Calendar'[Date]))
```

Year	Month Number	Month Name	Year-Month	Year-Month Number	Day of Week Number	Day of Week	Fiscal Year
2019	1	Jan	2019-Jan	24229	2	Tue	2018/2019
2019	1	Jan	2019-Jan	24229	3	Wed	2018/2019
2019	1	Jan	2019-Jan	24229	4	Thu	2018/2019
2019	1	Jan	2019-Jan	24229	5	Fri	2018/2019
2019	1	Jan	2019-Jan	24229	6	Sat	2018/2019
2019	1	Jan	2019-Jan	24229	7	Sun	2018/2019

Figure 8.20: Calculated column to return the fiscal year

The formula uses an **IF** function to test if the month number of the date is greater than or equal to 4 (April). If the result is **TRUE**, then the fiscal year is constructed of the current year to the next year. And if **FALSE**, the fiscal year is constructed from the previous year to this year.

Yes, we could have referenced the **Month Number** column that we created previously instead of using the **MONTH** function on the **Date** column. But I wanted to demonstrate a complete formula that does not require that “helper” column.

This formula can easily be adapted for other fiscal years by simply changing the month number used in the logical test of the **IF** function.

Fiscal quarters

To calculate the fiscal quarter, we can use an **IF** function again to test the calendar quarter and then offset the result dependent on when the fiscal year begins.

The following formula is used to return the fiscal quarter (figure 8.21). For this example, the fiscal year runs from April to March.

Fiscal Qtr =

“Q”&

IF(QUARTER(‘Calendar’[Date])>=2,

QUARTER(‘Calendar’[Date])-1,

QUARTER(‘Calendar’[Date])+3)

The screenshot shows a spreadsheet interface. At the top, a formula bar contains the following formula:

```
1 Fiscal Qtr =
2 "Q"&
3 IF(QUARTER('Calendar'[Date])>=2,
4 QUARTER('Calendar'[Date])-1,
5 QUARTER('Calendar'[Date])+3)
```

Below the formula bar is a table with the following columns: Date, Year, Month Number, Month Name, Year-Month, Year-Month Number, Fiscal Year, Fiscal Qtr, and Fiscal Year-Qtr. The data rows are as follows:

Date	Year	Month Number	Month Name	Year-Month	Year-Month Number	Fiscal Year	Fiscal Qtr	Fiscal Year-Qtr
01/01/2019	2019	1	Jan	2019-Jan	24229	2018/2019	Q4	18/19 Q4
02/01/2019	2019	1	Jan	2019-Jan	24229	2018/2019	Q4	18/19 Q4
03/01/2019	2019	1	Jan	2019-Jan	24229	2018/2019	Q4	18/19 Q4
04/01/2019	2019	1	Jan	2019-Jan	24229	2018/2019	Q4	18/19 Q4
05/01/2019	2019	1	Jan	2019-Jan	24229	2018/2019	Q4	18/19 Q4

Figure 8.21: Fiscal quarter based on an April to March fiscal year

The **QUARTER** function is used to return the calendar quarter from the **Date** column. The **IF** function then tests if the quarter number is greater than or equal to 2 (April–June) and if **TRUE** subtracts 1. If the result of the logical test is **FALSE**, then add 3 to the quarter. So, quarter 1 (January–March) is quarter 4 of the fiscal year.

We can then create any additional columns that may be used in the labels or filters of our report, such as a combined fiscal year and quarter column. The following formula creates a combined **Fiscal Year-Qtr** column:

Fiscal Year-Qtr =

MID(‘Calendar’[Fiscal Year],3,2)&”/”&

RIGHT(‘Calendar’[Fiscal Year],2)&” “&

‘Calendar’[Fiscal Qtr]

In this formula, the fiscal year is changed from the “YYYY/YYYY” format used in the **Fiscal Year** column to a “YY/YY” format. The **MID** and **RIGHT** functions are used to extract the last two digits from the start and end year of the fiscal period.

Fiscal months

For the fiscal month, we will create two columns. One to calculate the fiscal month number and another for the fiscal month name.

The following formula is used to create the **Fiscal Month Number** column (figure 8.22). The IF function tests if the month number is greater than or equal to 4, and if **TRUE**, 3 is subtracted from the month number. So, 4 (April) becomes month number 1, and calendar month 5 is fiscal month 2, and so on.

If the result of the test is **FALSE**, then 9 is added to the month number. So, 1 (January) becomes month number 10, and so on.

Fiscal Month Number =

IF(MONTH('Calendar'[Date])>=4,

MONTH('Calendar'[Date])-3,

MONTH('Calendar'[Date])+9)

Date	Year	Month Number	Month Name	Year-Month	Year-Month Number	Fiscal Year-Qtr	Fiscal Month Number	Fiscal Month Name
01/01/2019	2019	1	Jan	2019-Jan	24	18/19 Q4	10	Jan
02/01/2019	2019	1	Jan	2019-Jan	24	18/19 Q4	10	Jan
03/01/2019	2019	1	Jan	2019-Jan	24	18/19 Q4	10	Jan
04/01/2019	2019	1	Jan	2019-Jan	24	18/19 Q4	10	Jan
05/01/2019	2019	1	Jan	2019-Jan	24	18/19 Q4	10	Jan

Figure 8.22: Returning the fiscal month number and month name

The formula to be used for the fiscal month name is actually the same as the one used earlier for the **Month Name** column. So, why duplicate this column? Well, the month name is the same, but the first month of the year is different (April instead of January in this example). We will specify the order of the months in each column soon.

Fiscal Month Name = FORMAT('Calendar'[Date], "MMM")

Week number of the year

The final column to be added to the **Calendar** table is for the week number of the year. For this column, we can use the **WEEKNUM** function.

The following formula uses **WEEKNUM** to return the week number of the year from the **Date** column (Figure 8.23). **WEEKNUM** defaults to a week beginning on a Sunday. So, the first week of 2019 actually started on Sunday, December 30, 2018. And week 2 begins on Sunday, January 6, 2019.

Week Number = WEEKNUM('Calendar'[Date])

Month Name	Year-Month	Year-Month Number	Day of Week Number	Day Name	Week Number
n	2019-Jan	24229	2	Tu	1
n	2019-Jan	24229	3	We	1
n	2019-Jan	24229	4	Th	1
n	2019-Jan	24229	5	Fr	1
n	2019-Jan	24229	6	Sa	1
n	2019-Jan	24229	7	Su	2
n	2019-Jan	24229	1	Mo	2
n	2019-Jan	24229	2	Tu	2

Figure 8.23: Week number of the year with WEEKNUM

You may want to specify a different method for numbering the weeks of the year. For example, to use the ISO 8601 standard, which is that week numbering begins on a Monday.

The following formula specifies option 2 for the return type in the **WEEKNUM** function (figure 8.24). This applies to the ISO standard of week numbering.

Week Number = WEEKNUM('Calendar'[Date],2)

Month Name	Year-Month	Year-Month Number	Day of Week Number	Day Name	Week Number
n	2019-Jan	24229	2	Tue	1
n	2019-Jan	24229	3	We	1
n	2019-Jan	24229	4	Thu	1
n	2019-Jan	24229	5	Fri	1
n	2019-Jan	24229	6	Sat	1
n	2019-Jan	24229	7	Su	1
n	2019-Jan	24229	1	Mo	2
n	2019-Jan	24229	2	Tue	2

Figure 8.24: ISO week number using option 2

With this formula, Week 1 started on December 31, 2018 and Week 2 started on January 7, 2019.

Create the date table with one formula

The date table for this report was created by inserting the columns individually. However, the entire date table can be created with a single formula.

This approach is quicker, and whenever you need the date table, you can simply copy the formula and have the new date table at the click of your fingers.

To create the **Calendar** table with one formula, the **ADDCOLUMNS** function is used. As its name implies, this function adds calculated columns to a given table.

The syntax of the **ADDCOLUMNS** function is as follows:

ADDCOLUMNS(table, name1, expression1, [name2], [expression2], ...)

- **Table:** Any DAX expression that returns a table.
- **Name:** The name to assign to the added column, enclosed in double quotes.
- **Expression:** The DAX expression to return the values for the added column.

You can provide as many Name and Expression pairs as required for the columns you want to add to the table.

The following formula is used to create the new table. The **CALENDAR** function is used for the table argument of **ADDCOLUMNS** followed by the Name and Expression pairs.

Calendar =

ADDCOLUMNS(

CALENDAR(

DATE(**YEAR**(**MIN**(Sales[Order Date])),1,1),

DATE(**YEAR**(**MAX**(Sales[Order Date])),12,31)

),

"Year", **YEAR**([Date]),

"Month Number", **MONTH**([Date]),

"Month Name", **FORMAT**([Date],"MMM"),

"Year-Month Number", **YEAR**([Date])&**MONTH**([Date]),

"Year-Month", **YEAR**([Date])&"-"&**FORMAT**([Date],"MMM"),

"Day of Week Number", **WEEKDAY**([Date],2),

"Day of Week Name", **FORMAT**([Date],"DDD"),


```

    "Is Weekend", IF(WEEKDAY([Date],2)>5,"Yes","No"),
    "Week Number", WEEKNUM([Date],2)
)

```

Note: You will need to specify the data types of any columns that need changing. For example, the [Date] column.

Sorting table columns by another column

For many of the date attribute columns that we created in the **Calendar** table, we created an index column in addition to the column that contains the name of the date attribute, that is, **Day of Week Number** and **Day of Week Name**.

The reason that we did this was to correctly order the **Month Name**, **Day of Week Name**, **Fiscal Month Name**, and **Year-Month** columns of our calendar table.

Figure 8.25 shows the columns used in four distinct tables in the Report view. In each table, the order of the columns is incorrect. Power BI orders each column in A–Z order and needs to be instructed otherwise. This is where our index columns come to the rescue:

Month Name	Year-Month	Fiscal Month Name	Day of Week Name
Apr	2019-Apr	Apr	Fri
Aug	2019-Aug	Aug	Mon
Dec	2019-Dec	Dec	Sat
Feb	2019-Feb	Feb	Sun
Jan	2019-Jan	Jan	Thu
Jul	2019-Jul	Jul	Tue
Jun	2019-Jun	Jun	Wed
Mar	2019-Mar	Mar	
May	2019-May	May	
Nov	2019-Nov	Nov	
Oct	2019-Oct	Oct	
Sep	2019-Sep	Sep	

Figure 8.25: Columns from the date table not ordered correctly

Let us correct the sort order for each of these columns, starting with the **Month Name** column.

1. In the Data view, click on the **Month Name** column.
2. Click **Column tools** | **Sort by column** | **Month Number** (figure 8.26).

The screenshot shows the Power BI interface with the 'Column tools' ribbon active. The 'Month Name' column is selected, and its dropdown menu is open. The table below shows data for January 2019. Red arrows and numbers 1-4 indicate the steps to sort a column by another column:

- Click the **Month Name** column.
- Click **Sort by column** | **Day of Week Number**.
- Click the **Year-Month** column.
- Click **Sort by column** | **Year-Month Number**.

Figure 8.26: Sorting a column by another column

Nice and simple. Ok, now that we know how to do this, let us re-order all other columns that require our assistance.

1. Click the **Day of Week Name** column.
2. Click **Sort by column** | **Day of Week Number**.
3. Click the **Year-Month** column.
4. Click **Sort by column** | **Year-Month Number**.
5. Click the **Fiscal Month Name** column.
6. Click **Sort by column** | **Fiscal Month Number**.

Back in the Report view, the columns in each table are now ordered as required (figure 8.27).

Month Name	Year-Month	Fiscal Month Name	Day of Week Name
Jan	2019-Jan	Apr	Mon
Feb	2019-Feb	May	Tue
Mar	2019-Mar	Jun	Wed
Apr	2019-Apr	Jul	Thu
May	2019-May	Aug	Fri
Jun	2019-Jun	Sep	Sat
Jul	2019-Jul	Oct	Sun
Aug	2019-Aug	Nov	
Sep	2019-Sep	Dec	
Oct	2019-Oct	Jan	
Nov	2019-Nov	Feb	
Dec	2019-Dec	Mar	

Figure 8.27: Columns from the date table are now ordered correctly

Creating the relationships to the date table

We now need to connect the **Calendar** table to the **Sales** table of our model, just like we did with the other tables of the model in *Chapter 7, Creating the Data Model*.

Let us switch to the Model view and create a many-to-one relationship between the two tables.

In Model view, click and drag from the **Order Date** field in the **Sales** table to the **Date** field in the **Calendar** table. On releasing the button, a relationship line will appear, visualizing the many-to-one relationship (*figure 8.28*).

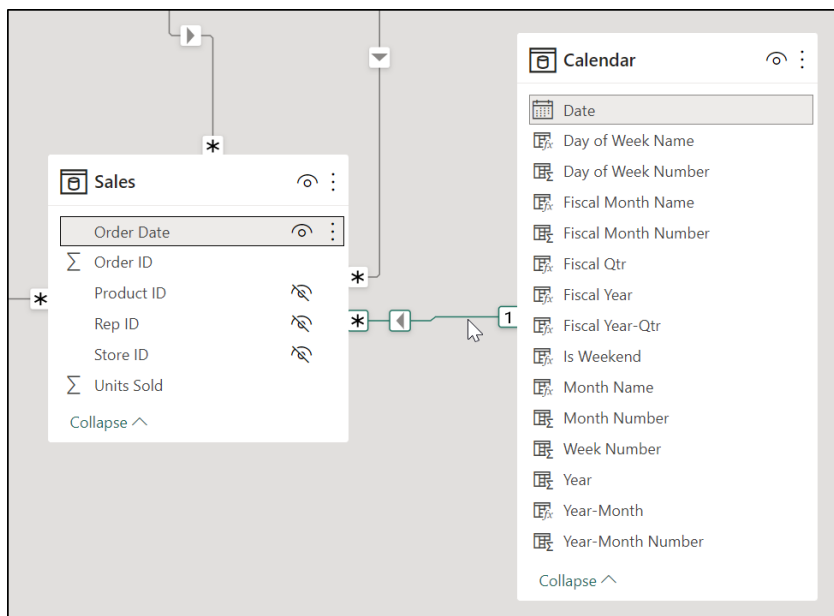


Figure 8.28: Creating the relationship between the date and fact table

Relationships can also be created using the **Manage Relationships** button on the **Home** tab of the Report, Data, and Model views in Power BI.

Marking a table as a date table

We must now specify to Power BI to use the **Calendar** table as the date table. Doing so gives us control, and removes the automatically created date tables. This is noticeable in the removal of the date hierarchies in each of the date columns of the model.

Figure 8.29 shows the **Order Date** column of the **Sales** table after marking a date table in Power BI. The date hierarchy shown previously in figure 8.1 has now disappeared:

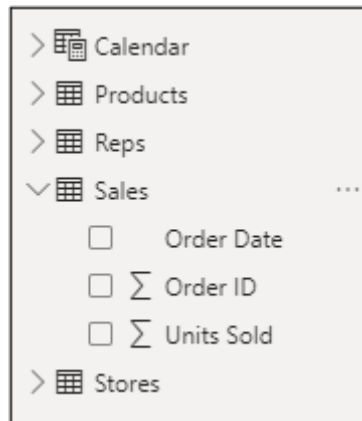


Figure 8.29: No date hierarchy on the order date field of the model

To mark the **Calendar** table as the date table:

1. In the Data view, switch to the **Calendar** table.
2. Click **Table tools** | **Mark as date table** | **Mark as date table** (figure 8.30).

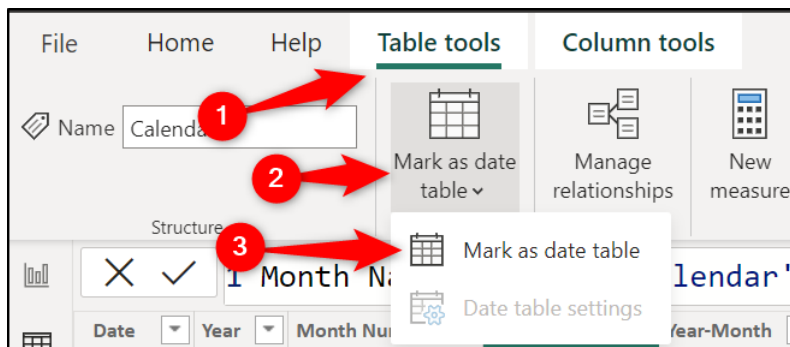


Figure 8.30: Marking a table as a date table

The **Mark as date table** window appears and asks us to state which column of the table contains the date values.

3. Click the Data column list and click **Date** (figure 8.31). Click **OK**.

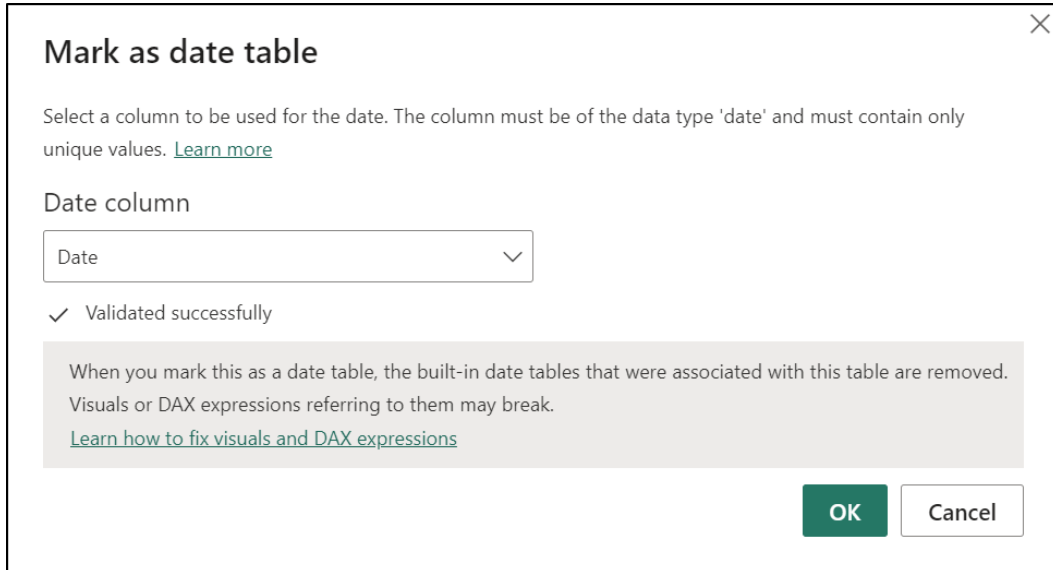


Figure 8.31: Specifying the date column for the date table

When specifying the date column for the date table, Power BI performs validation checks for the date column requirements, detailed earlier in this chapter. These include that the date column contains a unique list of dates, that the dates are contiguous, and that there are no null values. In *figure 8.31*, it states that the validation was successful.

Hiding other date fields

The final step is to hide the fields in the report view that we will not be used directly in any visual or filter.

1. Switch to the Model view (this is the best view for this task).
2. Click the eye icon to hide the **Order Date** field in the **Sales** table (*figure 8.32*). This field is no longer needed in the Report view, as we have the **Date** field in the **Calendar** table.
3. Click the eye icon to hide the **Month Number**, **Fiscal Month Number**, **Day of Week Number**, and **Year-Month Number** fields. These index fields were used to sort columns and are not required directly in the report.

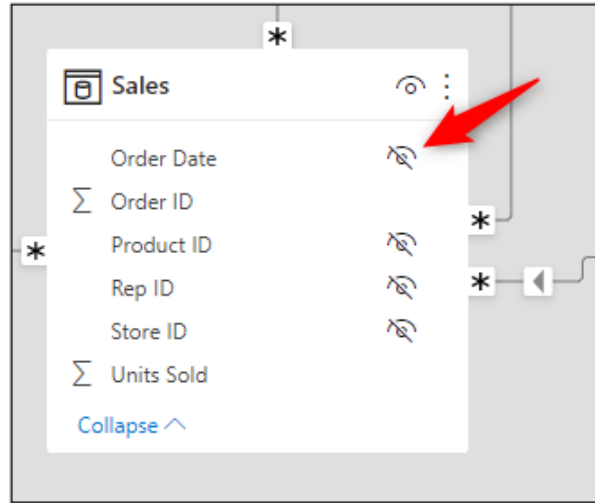


Figure 8.32: Hiding the Order Date field from the report view

Conclusion

In this chapter, we learnt the role that a date table plays in our data model and the many benefits of creating one. We created a dynamic date table using DAX formulas that update dependent on the dates in the model whenever the PBI report is refreshed. There are different date attributes that we plan to use in the measures and visuals of our report. We learnt how to create different date attribute columns using DAX.

Finally, we made sure that the different columns of the table were ordered correctly; we connected the table to the other tables in the model, marked it as a date table, and hid the unrequired fields from the report view.

In the upcoming chapter, we will start writing DAX measures. This is a logical progression from the calculated tables and calculated columns created with DAX in this chapter. A measure is a named calculation, often used to aggregate values, that is saved in the model and used wherever required in a report.

Questions

Here are some questions to test what you have learnt in this chapter.

1. Which of the following is a reason to create your own date table?
 - a. Avoid a bloated model caused by numerous hidden date tables
 - b. The ability to create any date attribute fields you require
 - c. It avoids duplicate fields and provides a simpler, cleaner model
 - d. All of the above
2. Which function was used to return the month name and day of week name from a date?
 - a. WEEKDAY
 - b. FORMAT
 - c. MONTH
 - d. TEXT
3. What is the correct path to the Sort by column button in Power BI?
 - a. Data view > click on the column > Table tools > Sort by column
 - b. Report view > click on the column > Insert > Sort by column
 - c. Report view > click on the column > Modeling > Sort by column
 - d. Data view > click on the column > Column tools > Sort by column
4. Which of the following are DAX functions?
 - a. CALENDARAUTO
 - b. QUARTER
 - c. IF
 - d. All of the above

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



CHAPTER 9

Adding DAX Measures

Introduction

In this chapter, we delve further into the DAX formula language and look at measures. Measures are DAX calculations that aggregate the values for many rows of a table.

As you have witnessed already, DAX is a rich formula language, and this book serves only as an introduction and the beginning of your DAX journey.

We begin the chapter by describing what a measure is. We answer the questions of why you should use measures and what is the difference between implicit and explicit measures?

Next, we will look at different approaches to organizing your measures so that they are easy to find and use.

We will also learn a few of the more commonly used DAX functions, including **SUMX**, **CALCULATE**, and **RELATED**. These also include some time intelligence calculations that make use of the **Calendar** table created in the previous chapter.

Structure

In this chapter, we will cover the following topics:

- What is a measure in Power BI?
- The difference between implicit and explicit measures.
- Common techniques for organizing your measures.
- Examples of seven different DAX functions, including **SUMX**, **CALCULATE**, and **RELATED**.
- Time intelligence calculations.

Objectives

After reading this chapter, you will understand the importance of creating your own explicit measures in your Power BI datasets. You will know the difference between measures and calculated columns and the many advantages that using measures provide.

By the end of this chapter, you will have learnt some of the most used DAX functions, including **CALCULATE**, **SUMX**, and **RELATED**. You will also be aware of the different approaches to storing measures.

What is a measure in Power BI?

A measure is a DAX calculation that aggregates the values from many rows of a table. This is different from calculated columns that evaluate the values for each row of a table.

Calculated columns return values that can be seen in the column added to a table. We created calculated columns in the previous chapter to return the year, week number, month name, and more for our date table. We used functions such as **MONTH**, **IF**, and **WEEKNUM** that evaluate for each row of a table.

A measure cannot be seen in the Data view in Power BI like a calculated column. The results of a measure are only seen when it is added to a visual on a report page. DAX functions that aggregate values from many rows of a table are used in measures, for example, **SUM**, **AVERAGE**, and **COUNTRWS**.

Figure 9.1 shows the **Month Name** column from the **Calendar** table we created in the previous chapter and a measure named **Total Revenue** used in a table visual. Notice the different icons by their name. The calculator icon indicates a measure in your model.

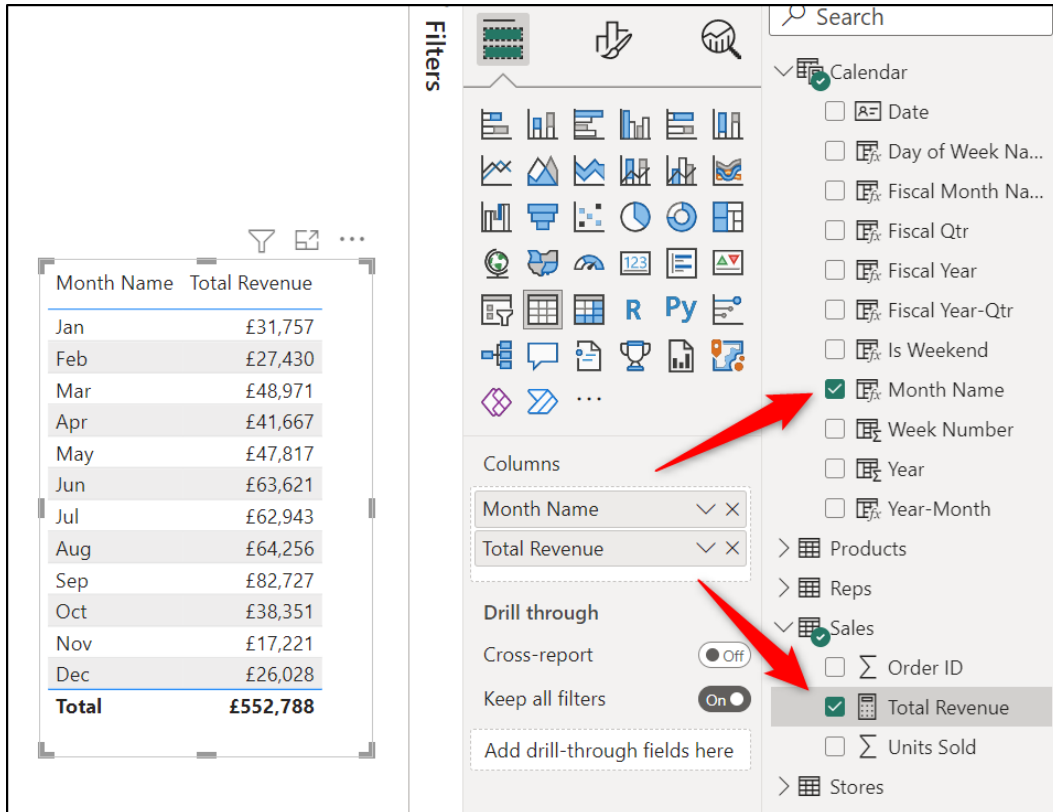


Figure 9.1: Calculated columns and measures in a table visual

For users with a background of using PivotTables in Excel, you can think of a measure as a field that you would use in the Values area of the PivotTable. A calculated column would be used in the Rows, Columns, or Filter area of the PivotTable.

Implicit versus explicit measures

Files: **sales-report-ch-9-start.pbix**

When we talk about DAX measures, what we are really referring to, are explicit measures. Let us understand what this means.

In the **sales-report-ch-9-start.pbix** file, we will create a measure to sum the total units sold and view the results in a table visual. The table will show the total units sold by the month.

One way to do this would be to simply insert a table visual and move the **Month Name** column from the **Calendar** table, and the **Units Sold** column from the **Sales** table into the Columns area of the table visual (figure 9.2).

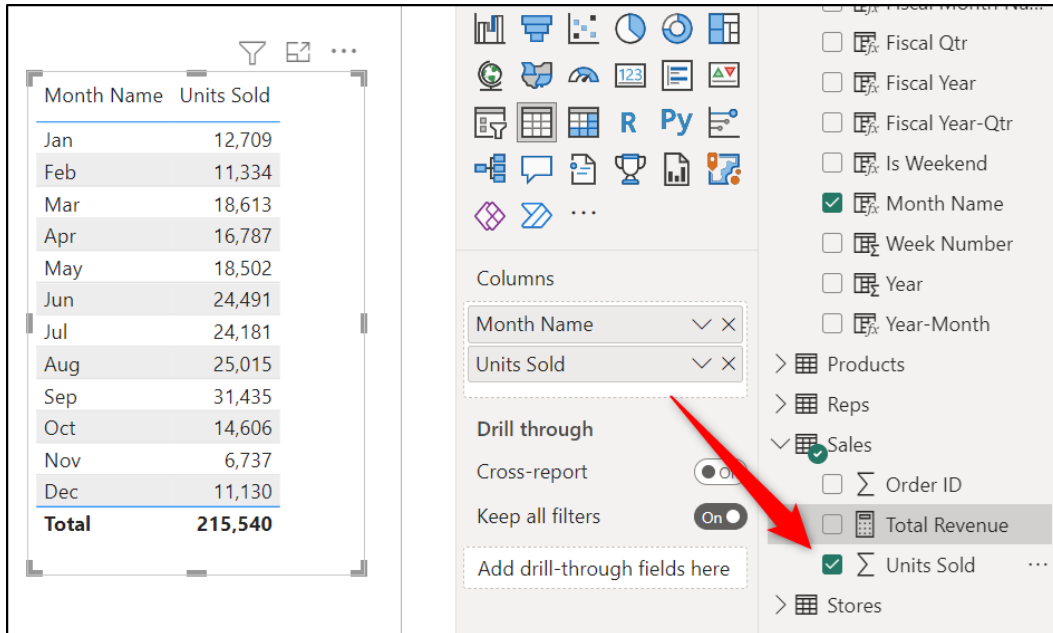


Figure 9.2: Implicit measure to calculate the total units sold

The sum of the units sold values by the month are returned. The **Units Sold** field contains number values, so **Sum** is the default method for summarizing the values (the default summarization method can be changed).

This is the result that we wanted, and it was easily achieved without writing any DAX measures of our own. This type of measure is known as an implicit measure.

The calculation, or summarization method, is applied only for that instance of the field. If the **Units Sold** field was added to another visual, and instructed to sum the values, then this is a second calculation.

The summarization method applied can be changed for each instance of a field by clicking the arrow next to the field name in the *Columns* area of the table and clicking the required summarization option (figure 9.3).

This is a valid way of calculating the values in a visual, and we used this approach for our report in *Chapter 4, Creating a Simple Power BI Report*.

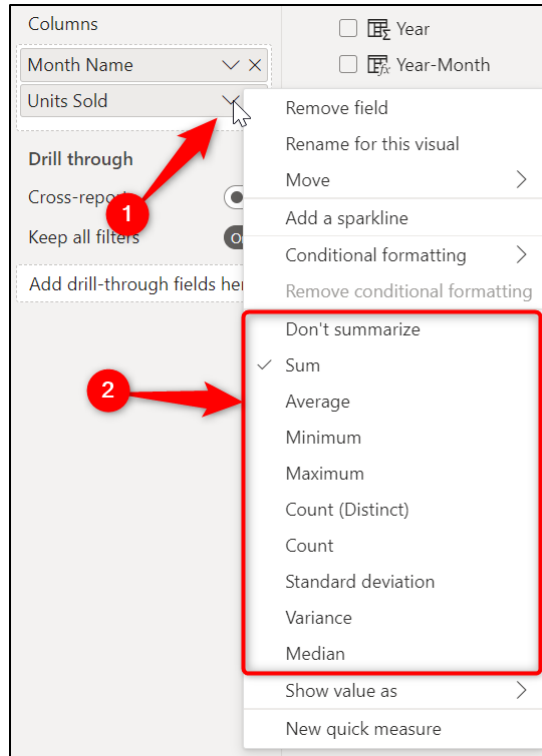


Figure 9.3: Different ways to summarize the values in a field

However, this approach has its limitations when compared to creating our own explicit measures.

These limitations include the following:

- Implicit measures perform separate calculations for each instance of their use. This increases the load on calculations.
- They do not have context as they are not built-for-purpose. Creating a measure to total the units sold versus applying a calculation on-the-fly to the **Units Sold** field has greater meaning.
- You are limited only to the summarization options provided in the list.

Let us see how we can create our own explicit measure to return the total units sold.

1. Click on the **Sales** table in the **Data** pane (this is so that the measure is stored in this table).
2. Click **Home | New measure** (this option is also found on the Table tools tab or by right-clicking a table name).

3. A new measure appears in the **Sales** table, and the Formula bar is active for you to write the measure.
4. Enter the following formula and press Enter (*figure 9.4*).

Total Units Sold = SUM(Sales[Units Sold])

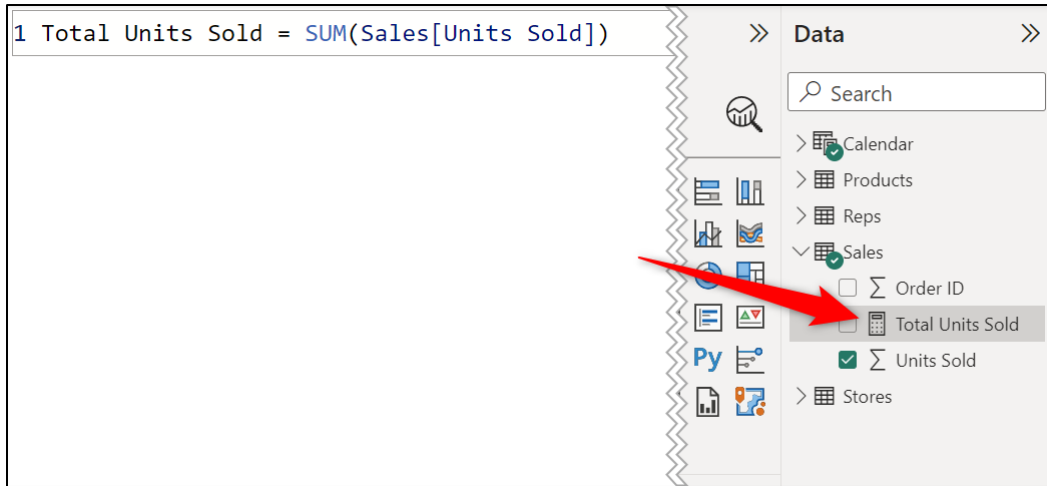


Figure 9.4: Measure to sum the values from the Units Sold column

A few things to note about the formula.

- The name of the measure is entered to the left of the equals.
- The formula is entered to the right of the equals.
- The SUM function is used, and the column is referenced explicitly. Column references are always entered in a **TableName[ColumnName]** syntax.

Tip: You can make the text in the Formula bar larger by pressing **Ctrl** and scrolling the mouse wheel simultaneously. No mouse, no problem. You can also do this by pressing the **Ctrl + +** keys.

1. With the measure selected, assign a format to the measure using the buttons on the **Measure tools** tab (*figure 9.5*). In this example, the button to display values with commas as a thousand separator was clicked.

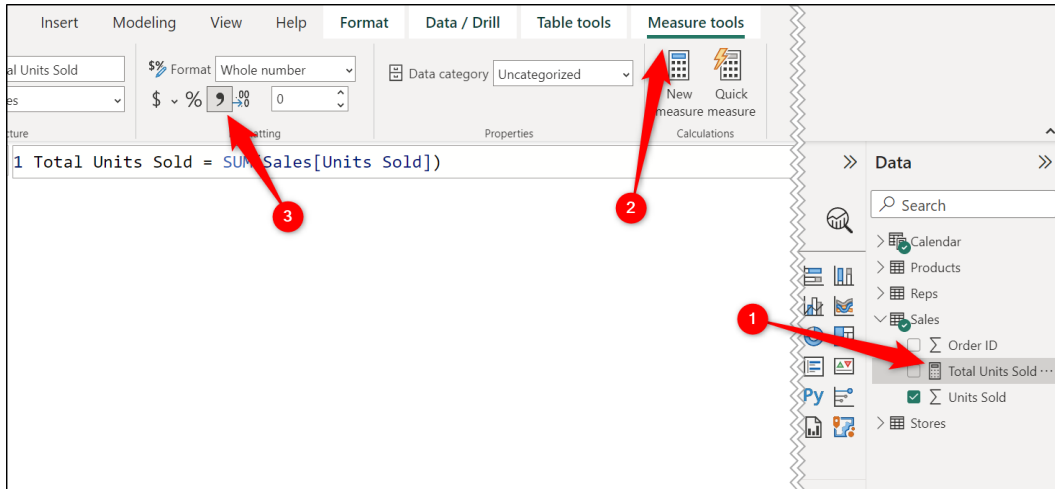


Figure 9.5: Formatting the measure with the thousand separator

2. Remove the **Units Sold** field from the table that we used previously by clicking the **X** by the field name and drag the **Total Units Sold** measure into the **Columns** area or check the box by the measure name (figure 9.6).

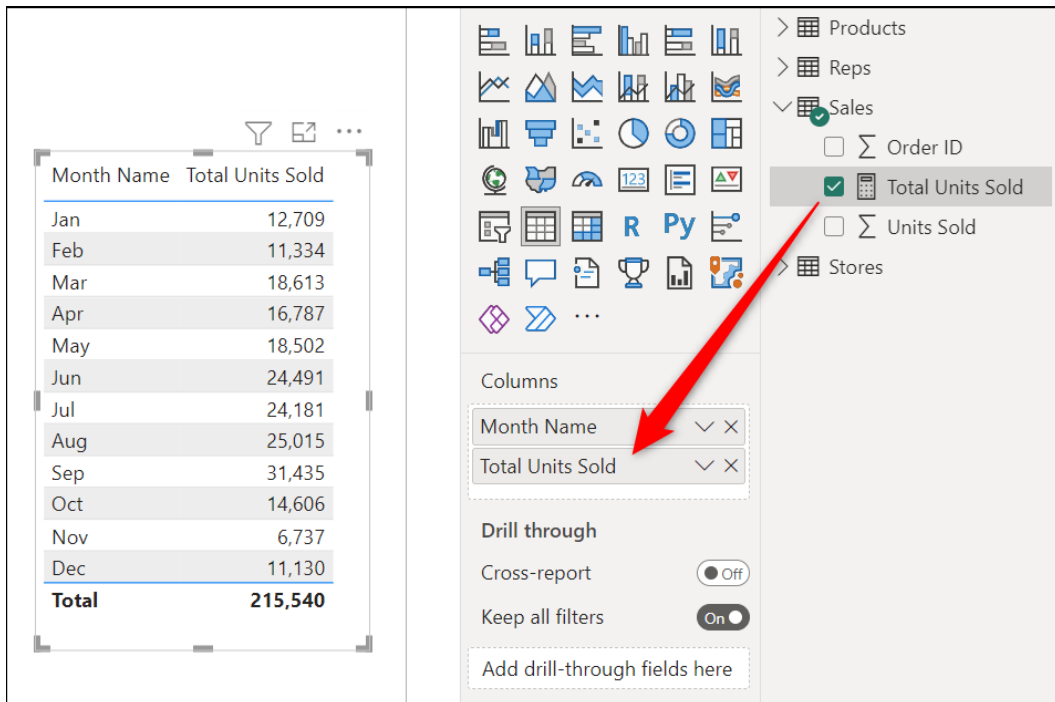


Figure 9.6: Using the measure in the table visual

The benefits of using explicit measures include the following:

- They only calculate once.
- They can be reused again and again in other visuals and within other measures. This reduces calculation time and makes more concise and meaningful formulas.
- They are built for a purpose and are descriptive.
- They can be formatted.
- You can take advantage of the rich DAX language for more powerful calculations.

From this point forward, explicit measures will be referred to simply as measures.

The COUNTROWS function

Let us jump straight in and create a second measure. This measure will count the number of sales.

For this measure, we will use the **COUNTROWS** function. Its purpose is to count the number of rows in a table.

This makes it the perfect choice for this measure, as it can be used to return the number of rows in the **Sales** table. Each row is an individual sales transaction, so the number of rows is equal to the number of sales.

Note: Functions such as COUNT and COUNTA could have been used instead, but COUNTROWS really is perfect for this task.

1. Click on the **Sales** table and click the **New measure** button on the **Home** tab.
2. Type the following formula into the Formula bar.
Count of Sales = COUNTROWS(Sales)
3. With the measure selected, click the button to display the thousands separator on the **Measure tools** tab.
4. Select the table visual and drag the **Count of Sales** measure into the **Columns** area to display the results in the table.

Tip: Using a table visual (or some other visual) to test the results of your measure is good practice. You do not immediately see the results of your measures as you do with calculated columns. Just because you do not receive an error does not mean the measure is working correctly. Always test them.

Figure 9.7 shows the **Count of Sales** measure added to the table. The measure is selected, so the formula can also be seen in the Formula bar. Please refer to the following figure:

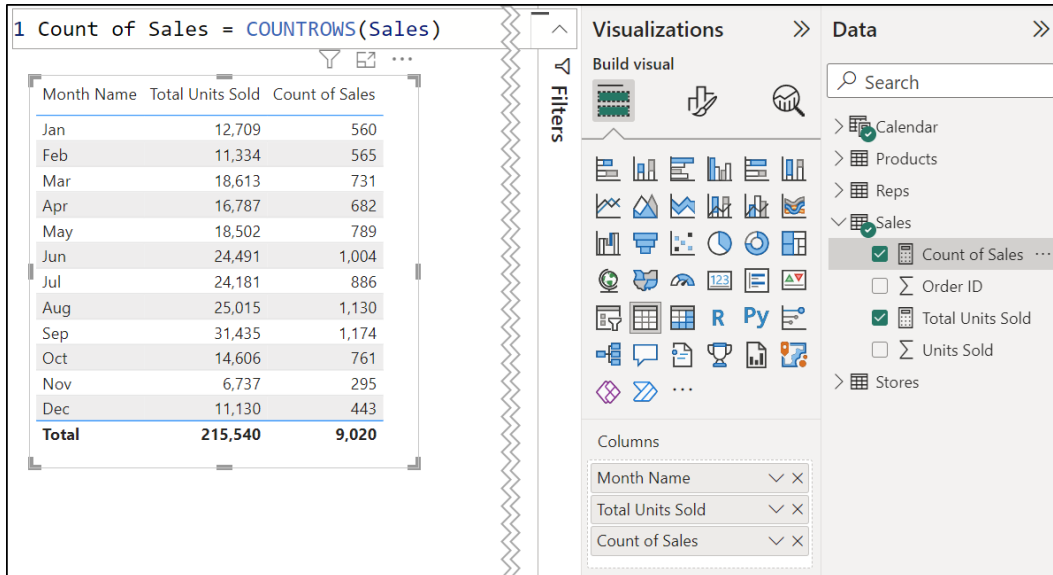


Figure 9.7: COUNTROWS function to count the number of sales

Organizing your measures

So far, we have created only two measures, but we have a few more to create. Before we do so, we should consider the best method to organize our measures, so that they are easy to locate.

The two measures we have created so far have both been stored in the **Sales** table. This is a logical decision, and with only two measures, it works fine. However, listing 10, 20, or even 50 measures in one table without any organization is not a savvy way to operate.

There are two methods that can be used to efficiently organize your measures.

- Storing them in folders and subfolders within the appropriate table.
- Creating a measures table to store your measures separate from the other tables and fields of the model.

How do I move a measure?

First, let us cover how to move a measure from one table to another. If you did not select the table before creating a measure, it might have been stored in the wrong table.

Figure 9.8 shows the **Count of Sales** measures stored, by mistake, in the **Products** table. Notice that the measure functions correctly. A measure can be stored in any table.

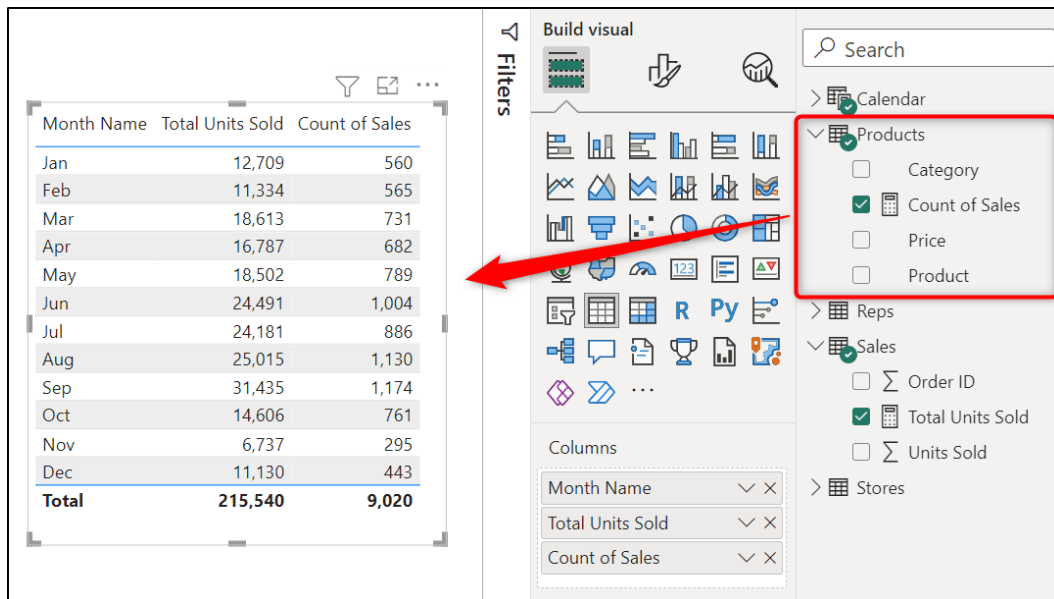


Figure 9.8: Measure in the wrong table but functions correctly

Now, although it functions fine, it makes no sense to store a measure that counts the sales in a table of data about products. So, we will move it to the **Sales** table.

1. From the Report view, click on the measure to be moved. In this example, that is **Count of Sales**.
2. Click the **Home** table list arrow on the **Measure tools** tab and click the table that you want to move the measure to (figure 9.9).

Tip: In Model view, you can move a measure by simply clicking and dragging it between the tables and folders. Unfortunately, this is not possible in the Report view.

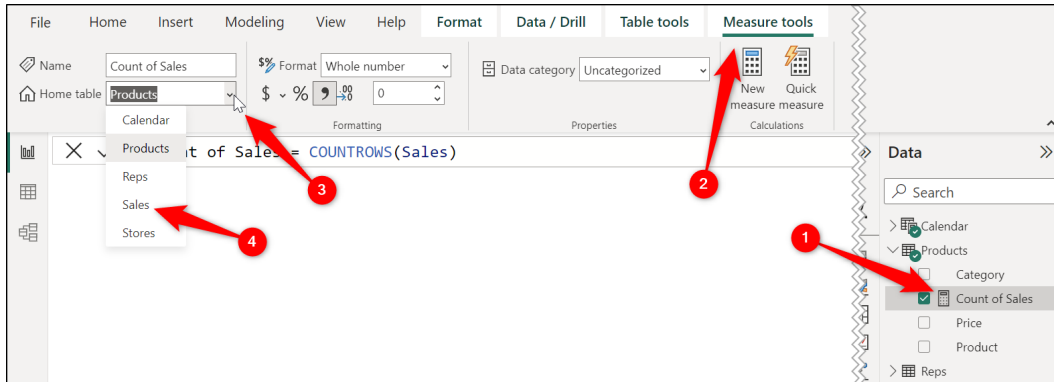


Figure 9.9: Moving a measure between tables

Creating folders within tables

Creating folders in tables is a great way to organize your measures. If you have many measures, this approach can be taken further, and you can create subfolders to organize your measures at a deeper level.

Let us create a folder in the **Sales** table named **Measures**. With this table, we can group them separately from the table fields. The folder will also allow us to collapse and expand the list of measures making it simple to hide and show measures at will.

Note: In addition to measures, the fields of a table can also be stored within folders for better organization.

1. Switch to the Model view.
2. In the **Data** pane, click on the measure that you want to store in a folder. Hold the *Ctrl* key and click any other measures you want to store in the folder.
3. In the **Properties** pane, type a name for the folder in the **Display folder** box and press *Enter*.

Figure 9.10 shows the **Measures** folder created in the **Sales** table. The **Count of Sales** and **Total Units Sold** measures were both selected prior to entering the **Display folder**, so they have both been moved to this folder:

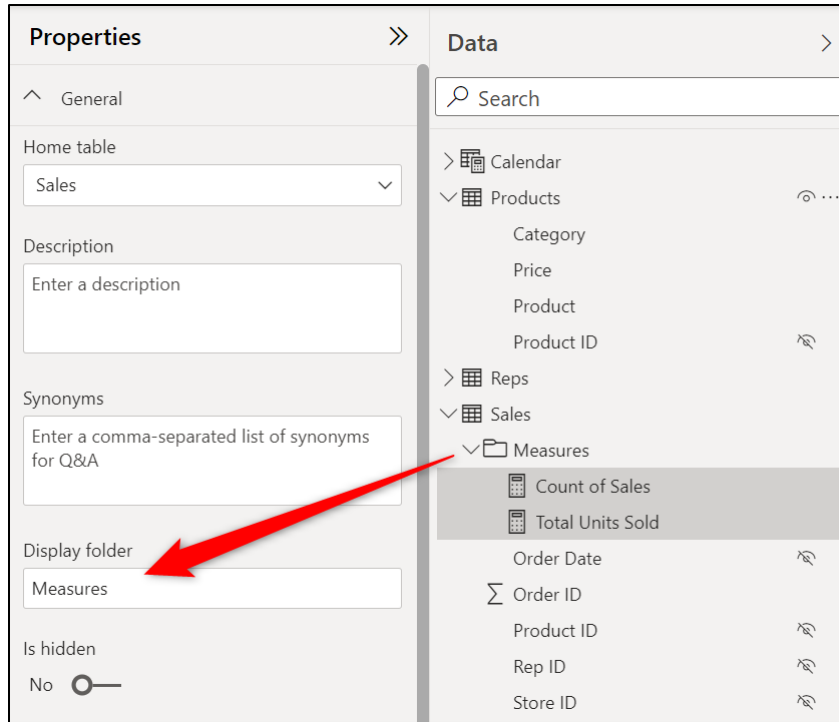


Figure 9.10: Creating a folder to organize measures within a table

To store measures in a subfolder within the **Measures** folder, repeat the steps as before and enter a backslash “\” between the name of the folder and subfolder when typing in the Display folder box.

In *figure 9.11*, a subfolder named **Key Metrics** has been created within the **Measures** folder.

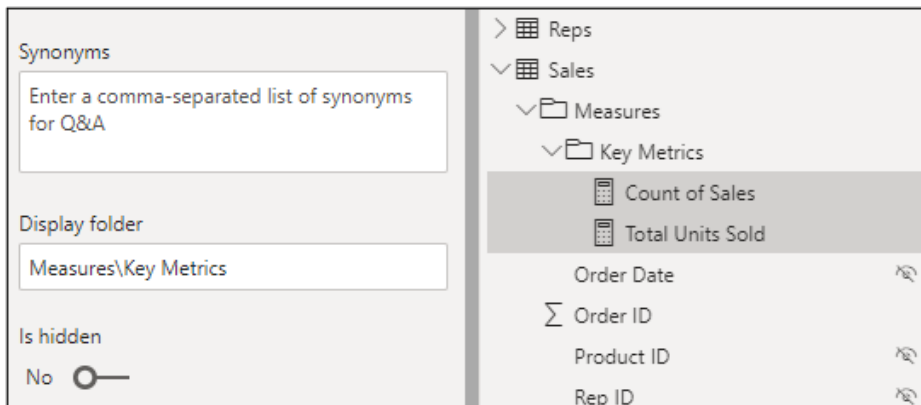


Figure 9.11: Creating a subfolder for your measures

Creating a measure table

To store the measures, we can create a separate calculated table rather than a folder inside a table.

Let us see how to create a measure table and move our existing measures into it.

1. From the Report view, click the **Enter data** button on the **Home** tab of the Ribbon (*figure 9.12*).

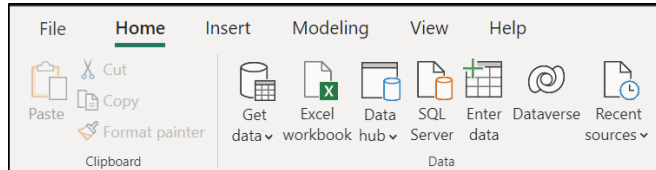


Figure 9.12: Enter data button on the Home tab of the Ribbon

2. In the **Create Table** window, type a name for your table in the **Name** box at the bottom of the window (*figure 9.13*), and double-click on the column header to rename the single column of our table (this column will be deleted shortly so renaming it is unnecessary and included for good practice of using tables only). Click **Load**.

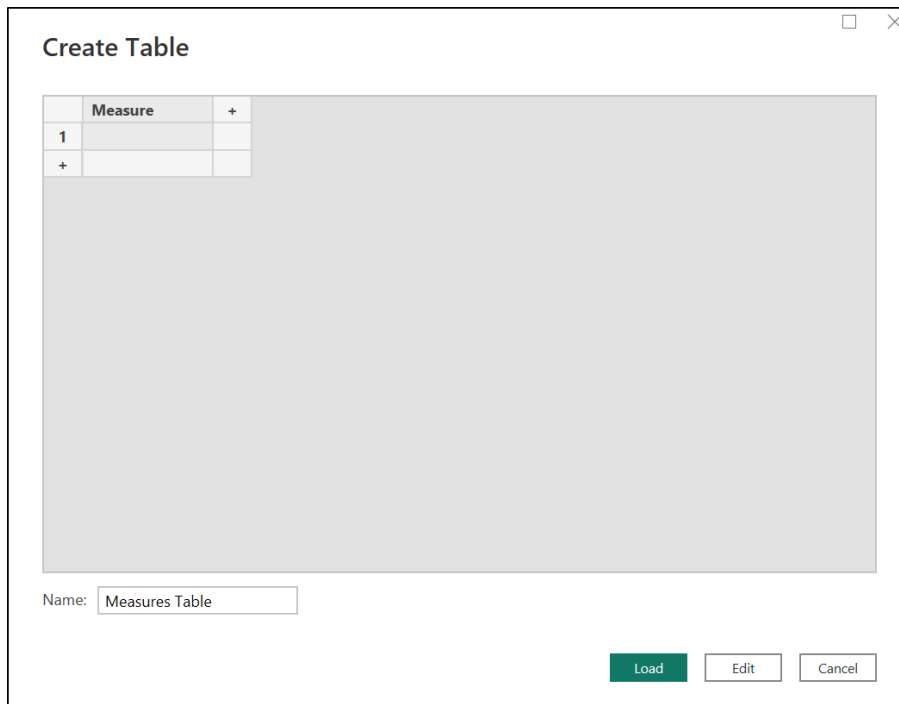


Figure 9.13: Assigning a table and column name in the Create Table window

- The new table appears in the **Data** pane with its single column (figure 9.14). Currently, this is a table like any other, but when you add measures to the table, Power BI automatically converts it to a measure table:

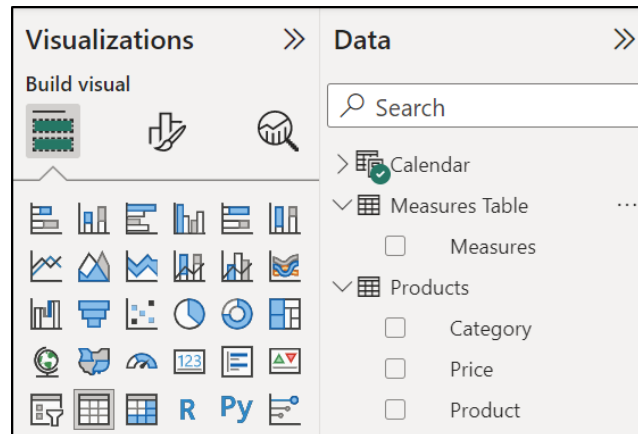


Figure 9.14: New measures table in the Data pane

- Select the **Count of Sales** measure and move it to the **Measures** Table by clicking the Home table list arrow on the **Measure** tools tab of the Ribbon and clicking **Measure Table**. Repeat for the **Total Units Sold** measure.
- Right-click on the **Measure** field and click **Delete from the model**. Click **Yes** to confirm that you want to delete the column.

Figure 9.15 shows both measures in the **Measures** Table and the **Measure** field removed. Notice the differences in the **Measures** Table when comparing the images in figures 9.14 and 9.15. In figure 9.15, the **Measures** Table appears at the top of the list above the **Calendar** table, and its icon has changed from a table icon to a calculator icon.

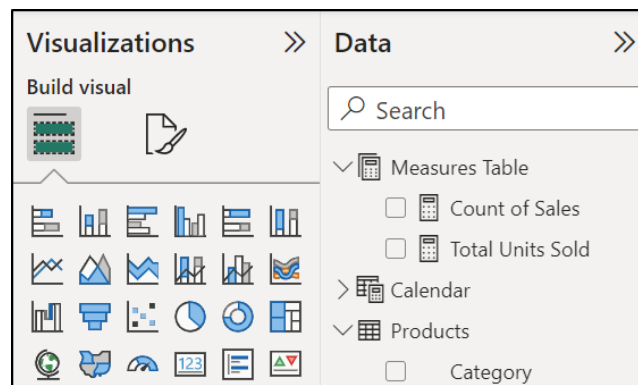


Figure 9.15: Existing measures moved to the measures table

Note: If your measures are stored within a folder in a table, as we created previously, then moving the measure also moves the folder. If you do not want the measure within that folder in the measuring table, switch to Model view and delete the text in the Display folder field that was used to create the folder.

From this point onward in the book, the measures that we create will be stored within folders that we create in the **Sales** table.

SUMX and other iterator functions

The next measure that we will create is for the total revenue from the sales.

Unfortunately, we do not have a total column containing the sales total for each row of the **Sales** table. So, this task is not as simple as using the **SUM** function on a single column.

However, what we do have in our data model, is a **Units Sold** field in the **Sales** table and a **Price** field in the **Products** table. To return the total revenue, we first need to multiply each value in the **Units Sold** field in **Sales** by the related product price from the **Price** column in **Products**. These row totals can then be summed.

Now, we could do this by inserting a calculated column that performs the units sold multiplied by the price calculation for each row of the **Sales** table. And then, create a measure with the **SUM** function to sum the values in this column.

Figure 9.16 shows the following formula used in a calculated column within the **Sales** table named **Total**. It returns each row total that can then be summed in a measure.

Total = Sales[Units Sold]*RELATED(Products[Price])

Order ID	Order Date	Product ID	Store ID	Units Sold	Rep ID	Total
19325	06 January 2019	R1006	2	10	SR1009	£66.00
19368	15 January 2019	R1008	2	10	SR1009	£33.00
19370	16 January 2019	R1002	2	12	SR1009	£28.80
19375	17 January 2019	R1009	2	15	SR1009	£37.50
19385	20 January 2019	R1001	2	8	SR1009	£9.60
19401	21 January 2019	R1010	2	30	SR1009	£84.00

Figure 9.16: Calculated column for the row totals

- The **RELATED** function is used in this formula. This brilliant function returns the related value from the **Price** column of the **Products** table for the product stated in the **Product ID** column in **Sales**. This function requires a relationship between the two tables.
- The syntax of the **RELATED** function is as follows. It only asks for the column that you want to return the related value from.

RELATED(ColumnName)

- This is cool, but we want to avoid the use of calculated columns so that we do not bloat the model too much and reduce calculation time. So, how do we get the total for each row in the **Sales** table without a calculated column?
- In DAX, there is a special group of functions known as iterator functions. These functions will perform a calculation on each row of a given table before they then perform some kind of aggregation. This group of functions includes **SUMX**, **COUNTAX**, **AVERAGEX**, **CONCATENATEX**, and more. Yes, they all have an X as the last letter of their name.

The one we need for this task is the **SUMX** function. It will perform a specified calculation for each row of a given table and then sum the resulting values. It is perfect for this task.

The syntax of **SUMX** is as follows. It requires the table to iterate down and the expression (calculation) that you want to perform on every row of that table.

SUMX(Table, Expression)

Let us use this function to return the total revenue and store the measure in the Key Metrics folder that we created earlier.

1. Click on the **Sales** table.
2. Click **Home | New measure**.
3. Type the following formula into the Formula bar.

Total Revenue = SUMX(Sales, Sales[Units Sold]*RELATED(Products [Price]))

4. On the **Measure tools** tab, select an appropriate format for the measure values from the Formatting group. In this example, a £ English currency was selected, and 0 decimal places were specified.
5. Move the measure into the **Key Metrics** subfolder. The easiest method is to switch to the Model view and click and drag the measure.

- Click on the table visual that we have been using to test the results of our measures so far and drag the **Total Revenue** measure into the Columns area of the visual to see its results.

Figure 9.17 shows the completed **Total Revenue** measure. The results of the measure and the formatting can be seen in the table visual.

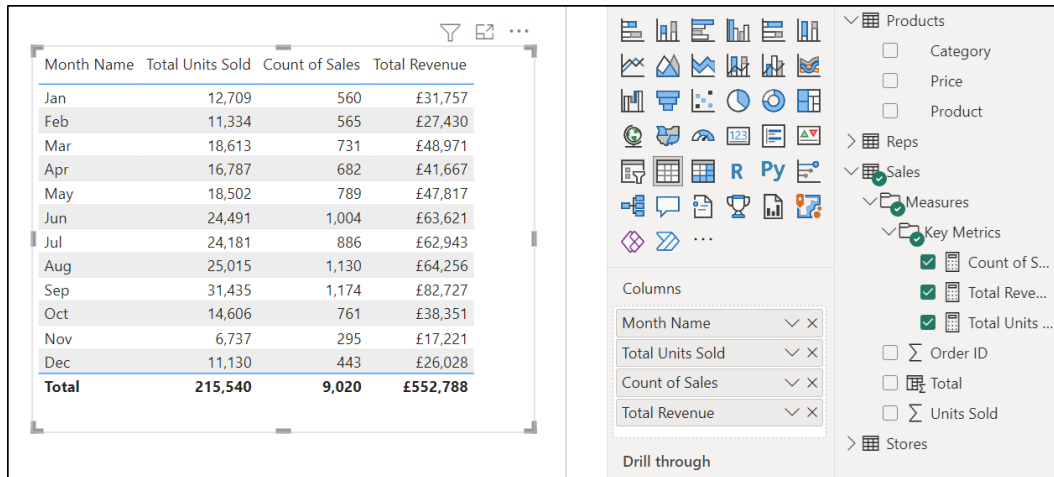


Figure 9.17: Total Revenue measure results are shown in the table

Let us now create a couple of measures that will use the **Calendar** table that we created in the previous chapter.

Returning the previous months revenue

The first of these measures will be to return the total revenue for the previous month. This measure will be used to help calculate the difference, and percentage difference, between the current month and the previous month.

This type of measure can be awkward when users are new to DAX, as you are working blind until the measure is used in a visual, and you might be thinking, “previous to which month?”. Currently, there is no context.

The context will be provided by a visual. In our report, a Slicer will be provided for the reader to specify a month. On choosing that month, our measures will return the total revenue for that month and the prior month.

Figure 9.18 shows a Slicer displaying the list of month names, and June is currently selected. A KPI visual is returning the total revenue for that month and comparing it to the previous month using the measure that we are about to create. We will create these visuals in *Chapter 10, Cards and Other Text Visuals*.

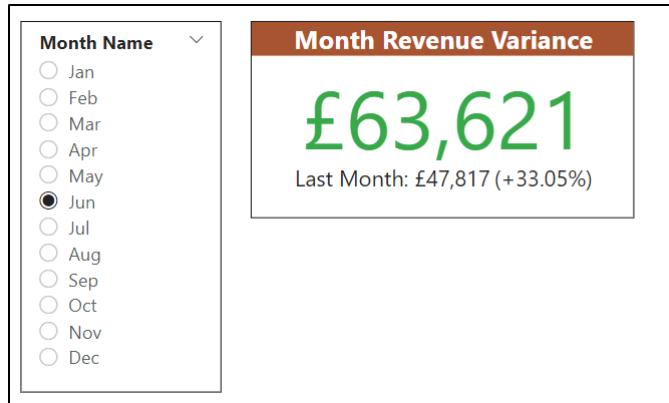


Figure 9.18: Month context provided by a Slicer visual

The CALCULATE function

To do this, we will apply a filter on our existing **Total Revenue** measure so that it returns the value for the month before the one in context.

The function that we will use to filter the **Total Revenue** measure will be **CALCULATE**. This will be one of only two examples of the **CALCULATE** function in this book, but as you start using more DAX, you will soon realize that **CALCULATE** is a very special function.

The **CALCULATE** function evaluates an expression in a context modified by a set of filters. So, in our example, the context of the **Total Revenue** expression is modified by a filter telling it to use the previous month.

The syntax for the **CALCULATE** function is as follows. We will only require one filter argument, but it can handle many.

```
CALCULATE(Expression, Filter1, [Filter2], ...)
```

The DATEADD function

To move the set of dates currently in context back one month, the **DATEADD** function will be used.

This function moves a set of dates by a specified interval. The dates can be shifted forward or backward, and the interval can be a year, quarter, month, or day. In our case, we want to shift the dates back by one month.

The syntax for the **DATEADD** function is as follows.

DATEADD(Dates, NumberOfIntervals, Interval)

Note: Alternative functions that could be used instead of DATEADD include PARALLELPERIOD and PREVIOUSMONTH.

Writing the measure

Let us now write the measure that will use **CALCULATE** and **DATEADD** to return the previous month's revenue.

1. Click on the **Sales** table in the **Data** pane and click **Home | New measure**.
2. Enter the following formula in the Formula bar (figure 9.19). Press *Shift + Enter* to start a new line in the Formula bar and the *Tab* and *Backspace* keys to add and remove indentation. This is done to make the formula more readable.

Revenue Previous Month =

CALCULATE([Total Revenue],DATEADD('Calendar'[Date],-1,MONTH))

The screenshot shows the Power BI interface. On the left, the Formula bar contains the following DAX formula:

```

1 Revenue Previous Month =
2     CALCULATE(
3         [Total Revenue],
4         DATEADD('Calendar'[Date], -1, MONTH)
5     )

```

On the right, the Data pane is visible, showing the 'Sales' table selected. The 'Measures' folder is expanded, and the newly created measure 'Revenue Previous Month' is listed at the bottom of the list.

Figure 9.19: Revenue Previous Month measure

The **Total Revenue** measure is used for the expression argument of **CALCULATE**. This demonstrates one of the advantages of explicit measures is that they can be reused. This is much better than entering the **SUMX** function for this argument. It is easier, more concise, is less calculations, and is more readable.

The **DATEADD** function is used for the **Filter1** argument of **CALCULATE**. It references the **Date** column of the **Calendar** table for the list of dates to be used. Remember, this column contains a complete and distinct list of dates. Then, **-1** is entered for the **NumberOfIntervals** argument to shift back one, and **MONTH** is chosen from the list provided for the Interval.

3. On the **Measure tools** tab, select an appropriate format for the measure from the Formatting group. In this example, The £ English currency was selected, and 0 decimal places were specified.

We will create a new subfolder within the existing **Measures** folder named **“Time Intelligence”**. We will use this folder to store this measure and the next three measures.

4. Switch to the Model view, click the **Revenue Previous Month** measure in the **Data** pane, and enter **Measures\Time Intelligence** into the **Display folder** box of the **Properties** pane (figure 9.20):

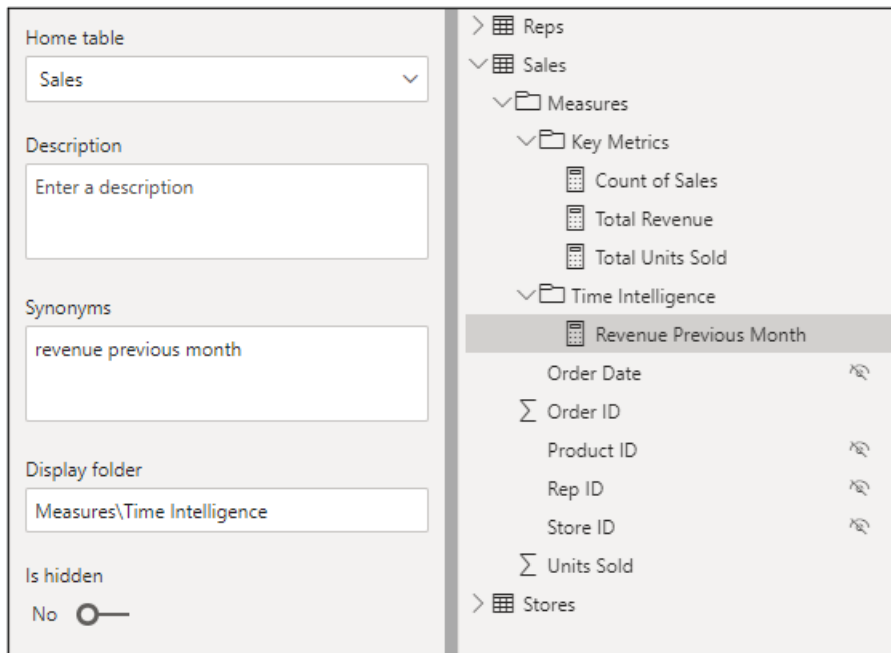


Figure 9.20: Storing the measure in a new Time Intelligence subfolder

5. Create a new table visual, or edit the existing one, to show only the **Month Name**, **Total Revenue**, and **Revenue Previous Month** fields and measures (figure 9.21).

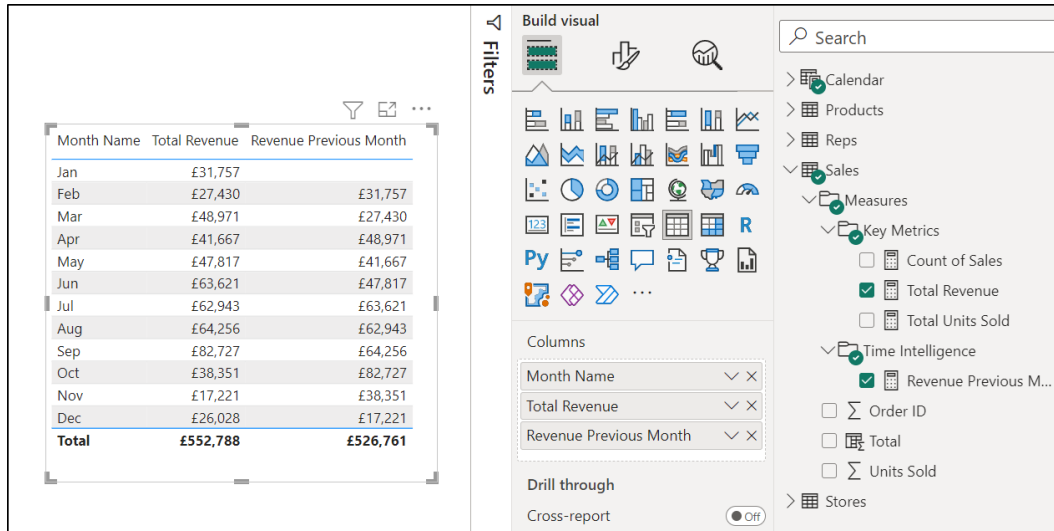


Figure 9.21: Testing the revenue from the previous month measure in a table

This may not be the most stunning visual for the finished report, but for now, it perfectly illustrates the **Revenue Previous Month** measure returning the correct results.

Revenue difference between the two months

Now that we have the previous month's revenue, we can create a measure for the revenue difference between the two months. The DAX formula will simply subtract the **Revenue Previous Month** value from the **Total Revenue** value.

1. Click on the **Sales** table in the **Data** pane and click **Home | New measure**.
2. Enter the following formula in the Formula bar.

Monthly Revenue Difference = [Total Revenue] - [Revenue Previous Month]

3. On the **Measure tools** tab, select an appropriate format for the measure from the Formatting group. The £ English currency was selected again with 0 decimal places specified.

4. Add the **Monthly Revenue Difference** measure to the table to test the results (figure 9.22).

Month Name	Total Revenue	Revenue Previous Month	Monthly Revenue Difference
Jan	£31,757		£31,757
Feb	£27,430	£31,757	-£4,327
Mar	£48,971	£27,430	£21,541
Apr	£41,667	£48,971	-£7,304
May	£47,817	£41,667	£6,150
Jun	£63,621	£47,817	£15,804
Jul	£62,943	£63,621	-£678
Aug	£64,256	£62,943	£1,314
Sep	£82,727	£64,256	£18,471
Oct	£38,351	£82,727	-£44,376
Nov	£17,221	£38,351	-£21,130
Dec	£26,028	£17,221	£8,807
Total	£552,788	£526,761	£26,028

Figure 9.22: The Monthly Revenue Difference formula added to the table

4. Switch to the Model view and move the **Monthly Revenue Difference** measure into the **Time Intelligence** subfolder.

Looking back at the results shown in figure 9.22, the results are correct; however, we do not really want to show the value for January. There is no prior month to January in this dataset, so showing the value is not appropriate, and a blank should be shown instead.

5. Let us edit the formula and add a conditional statement to show a blank if the **Revenue Previous Month** measure returns a blank.

Click on the **Monthly Revenue Difference** measure in the **Data** pane and edit the formula to the following (figure 9.23).

```

Monthly Revenue Difference =
    IF(
        ISBLANK([Revenue Previous Month]),
        BLANK(),
        [Total Revenue]-[Revenue Previous Month]
    )

```

The **IF** function is added for the conditional statement. The **ISBLANK** function is used to test if a blank is returned by the **Revenue Previous Month** measure,

and the **BLANK** function is used to return a blank if this is **TRUE**; otherwise, the difference is calculated.

```

1 Monthly Revenue Difference =
2     IF(
3         ISBLANK([Revenue Previous Month]),
4         BLANK(),
5         [Total Revenue] - [Revenue Previous Month]
6     )

```

Month Name	Total Revenue	Revenue Previous Month	Monthly Revenue Difference
Jan	£31,757		
Feb	£27,430	£31,757	−£4,327
Mar	£48,971	£27,430	£21,541
Apr	£41,667	£48,971	−£7,304
May	£47,817	£41,667	£6,150
Jun	£63,621	£47,817	£15,804
Jul	£62,943	£63,621	−£678
Aug	£64,256	£62,943	£1,314
Sep	£82,727	£64,256	£18,471
Oct	£38,351	£82,727	−£44,376
Nov	£17,221	£38,351	−£21,130
Dec	£26,028	£17,221	£8,807
Total	£552,788	£526,761	£26,028

Figure 9.23: Using the IF function to return blank instead of the value

Calculating the percentage revenue change

To calculate the percentage revenue change between the current month and the previous month, we need to divide the revenue difference by the previous month's revenue.

Now, we could perform a normal divide operation to achieve this, but the DAX library has a special **DIVIDE** function that we can use instead. Why use this? Well, it is referred to as the safe divide function as it provides an argument to perform an alternate response when dividing by 0.

The syntax for the **DIVIDE** function is as follows. The **AlternateResult** argument is optional.

DIVIDE(Numerator, Denominator, [AlternateResult])

Let us use **DIVIDE** to calculate the month-to-month percentage change.

1. Click on the **Sales** table in the **Data** pane and click **Home** | **New measure**.
2. Enter the following formula in the Formula bar.
% Monthly Revenue Difference =
DIVIDE([Monthly Revenue Difference],[Revenue Previous Month],BLANK())
3. On the **Measure tools** tab, click the button to display the values as a percentage and specify the values to 1 decimal place.
4. Add the **% Monthly Revenue Difference** measure to the table to test the results.

Figure 9.24 shows the completed **% Monthly Revenue Difference** measure added to the table. You can see the formula in the Formula bar and the formatting that was specified in the Ribbon. A blank is returned for January. Please refer to the following figure:

1 % Monthly Revenue Difference =
 2 DIVIDE([Monthly Revenue Difference],[Revenue Previous Month],BLANK())

Month Name	Total Revenue	Revenue Previous Month	Monthly Revenue Difference	% Monthly Revenue Difference
Jan	£31,757			
Feb	£27,430	£31,757	-£4,327	-13.6%
Mar	£48,971	£27,430	£21,541	78.5%
Apr	£41,667	£48,971	-£7,304	-14.9%
May	£47,817	£41,667	£6,150	14.8%
Jun	£63,621	£47,817	£15,804	33.1%
Jul	£62,943	£63,621	-£678	-1.1%
Aug	£64,256	£62,943	£1,314	2.1%
Sep	£82,727	£64,256	£18,471	28.7%
Oct	£38,351	£82,727	-£44,376	-53.6%
Nov	£17,221	£38,351	-£21,130	-55.1%
Dec	£26,028	£17,221	£8,807	51.1%
Total	£552,788	£526,761	£26,028	4.9%

Figure 9.24: % monthly revenue difference measure

5. Switch to the Model view and move the [% Monthly Revenue Difference] measure into the **Time Intelligence** subfolder.

Creating a year-to-date total

For our final measure, we will calculate the cumulative revenue for the year. For this, the **TOTALYTD** function will be used. This function returns the year-to-date value of a given expression.

The syntax for the TOTALYTD function is as follows:

TOTALYTD(Expression, Dates, [Filter], [YearEndDate])

There are two mandatory arguments. One for the expression to evaluate and another for the column of dates in the model.

There are optional arguments to specify a filter expression and a year-end date. The year-end date defaults to December 31 unless specified using a **month/day** string, such as **3/31** for March 31. We will not need either for this example.

Let us create the measure.

1. Click on the **Sales** table in the **Data** pane and click **Home | New measure**.
2. Enter the following formula in the Formula bar.
Revenue YTD = TOTALYTD([Total Revenue], 'Calendar'[Date])
3. On the **Measure tools** tab, specify the £ English currency with 0 decimal places.
4. Create a table visual and add the **Month Name**, **Total Revenue**, and **Revenue YTD** field and measures to test the results (figure 9.25).

Month Name	Total Revenue	Revenue YTD
Jan	£31,757	£31,757
Feb	£27,430	£59,187
Mar	£48,971	£108,158
Apr	£41,667	£149,825
May	£47,817	£197,642
Jun	£63,621	£261,263
Jul	£62,943	£324,206
Aug	£64,256	£388,462
Sep	£82,727	£471,189
Oct	£38,351	£509,540
Nov	£17,221	£526,761
Dec	£26,028	£552,788
Total	£552,788	£552,788

Figure 9.25: TOTALYTD function to show the cumulative revenue for the year

5. Switch to the Model view and move the **Revenue YTD** measure into the **Time Intelligence** subfolder.

That is all the DAX we will be covering in this book. *Figure 9.26* shows our completed set of measures in the Data pane.

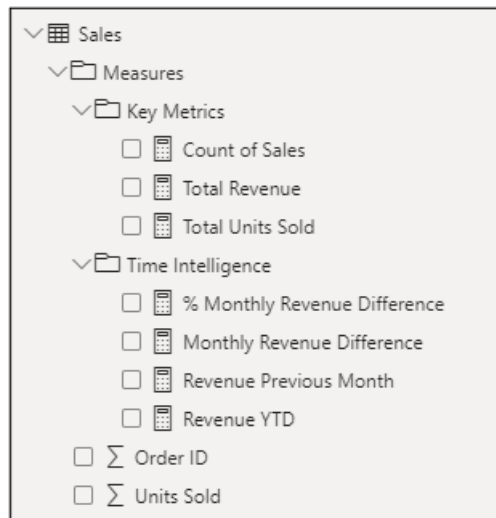


Figure 9.26: Completed set of measures

Conclusion

In this chapter, we learnt how to create DAX measures in Power BI, the difference between measures and calculated columns, and the advantages that measures offer over the implicit calculations that occur within a single visual.

We covered the two main methods for the effective organization of many measures—storing them in folders within the fact tables and creating a specific measure table for them.

A few different DAX functions were utilized in the chapter, including **SUMX**, **RELATED**, **DATEADD**, **CALCULATE**, and **TOTALYTD**.

In the next chapter, we will start creating the visuals for the reports. The upcoming chapter will focus on text-based visuals, including cards, KPI cards, tables, and the matrix.

Questions

Here are some questions to test what you have learnt in this chapter.

1. Which of the following are advantages to creating measures?
 - a. They only calculate once
 - b. They can be reused in other measures as well as visuals.
 - c. You can take advantage of the rich DAX language for more powerful calculations
 - d. All the above
2. Which function was used to apply a filter to modify the context of our Total Revenue measure?
 - a. CHANGEFILTER
 - b. FILTERREVENUE
 - c. CALCULATE
 - d. MODIFY
3. What is the correct path to creating a measure in Power BI?
 - a. Report view | select the table for the measure | Home | New measure
 - b. Right-click the table for the measure | New measure
 - c. Report view | select the table for the measure | Table tools | New measure
 - d. All the above
4. Which of the following is not a valid DAX function?
 - a. TOTALYTD
 - b. RELATEDITEM
 - c. SUMX
 - d. CALCULATE

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



CHAPTER 10

Cards and Other Text Visuals

Introduction

In this chapter, we begin looking at the visualizations available in Power BI to present your data effectively. This chapter focuses on text-based visuals, of which there are four that will be covered—the Card, KPI, Table, and Matrix.

For each visualization, we will see how to create them with a practical example of their use, learn the key formatting attributes, and a few tips to get the most from the visuals.

Structure

In this chapter, we will cover the following topics:

- How to create the four text-based visuals—Card, KPI, Table, and matrix.
- Key formatting attributes of interest for each visual.
- Few tips to get the best out of your text visuals.

Objectives

After reading this chapter, you will know how to use the most important text visuals in Power BI. You will also learn how to format these visuals along with some tips to take them beyond the standard visuals that others create.

The Card visual

Files: **sales-report-ch-10-start.pbix**

The Card is the simplest visual in Power BI but also one of the most used.

It is a humble box that is used to clearly show the more important metrics of your report page. For example, a single number that represents total sales, the number of new visitors, or the percentage of leads converted.

In the **sales-report-ch-10-start.pbix** file, we will insert two cards. One shows the total revenue, and the other shows the count of sales.

The following are the steps to create a card for total revenue:

1. Click the Card icon in the **Visualizations** pane.
2. Check the box for the **Total Revenue** measure in the **Data** pane or drag it into the **Fields** well in the **Visualizations** pane (*figure 10.1*).

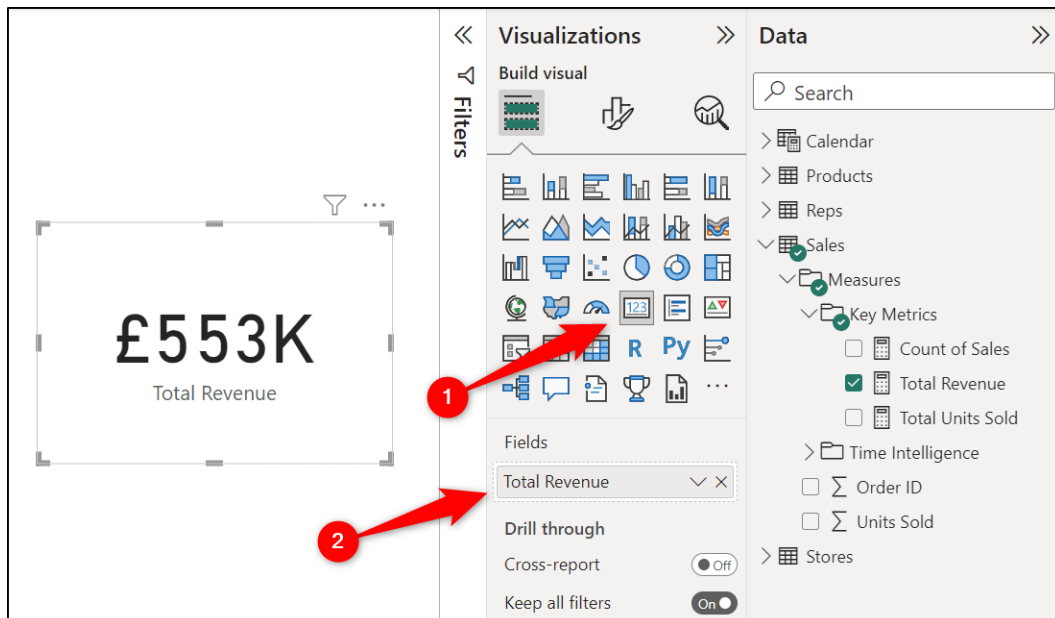


Figure 10.1: Inserting a card to show total revenue

The Card displays the result of the **Total Revenue** measure in a large black font with the measure name underneath. Let us look at improving the format of this Card.

To access the formatting options, follow the following steps:

1. Click on the Card visual on the page.
2. Click on the **Format your visual** button above the gallery of visualizations in the **Visualizations** pane (figure 10.2).

The gallery of **Visualizations** is replaced by the formatting options available for the selected visual. This same approach is used to format all visuals.

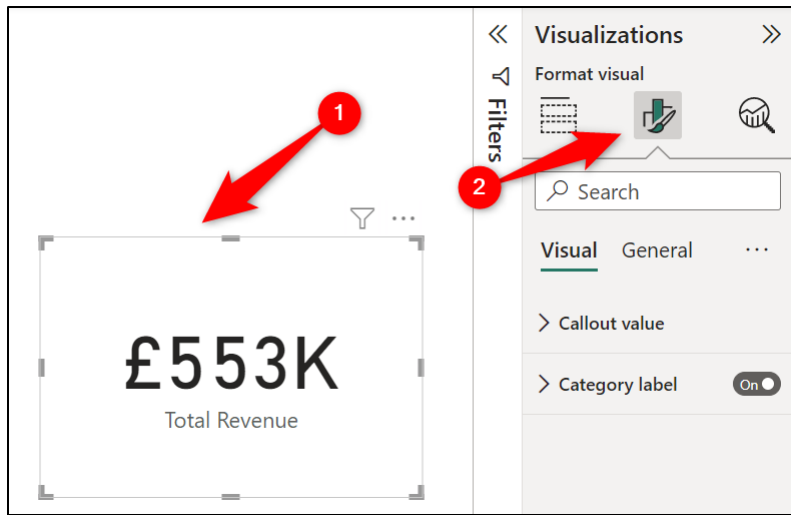


Figure 10.2: Accessing the formatting options for a visual

The **Visual** category lists the formatting options that are specific to the selected visual, and the **General** category lists the formatting options that are available for all visuals.

Formatting the callout value

The callout value is the large number or text displayed in the heart of the Card. You may wish to make changes to the font of the callout value or the units used to display the value.

For this Card, I will change the font from the default to the very nice Segoe UI Bold font. I will also reduce the font size to 29 and change it to a dark blue font color. In figure 10.2, the display units are automatically rounded to thousands. I will remove this to show more detail in the measurement result.

1. Click on the **Callout value** to expand the list of format options available.
2. Click the **Font** list and select the **Segoe UI Bold** font. Then, change the size in the next box by either entering the required value of **29** or by using the spin box arrows to reduce the size to **29**.
3. Click the **Color** list and choose a dark blue color.
4. Click the **Display units** list and select **None**. Notice the options to display values to thousands, millions, billions, or even trillions.

Figure 10.3 shows the result of these changes to the callout value:

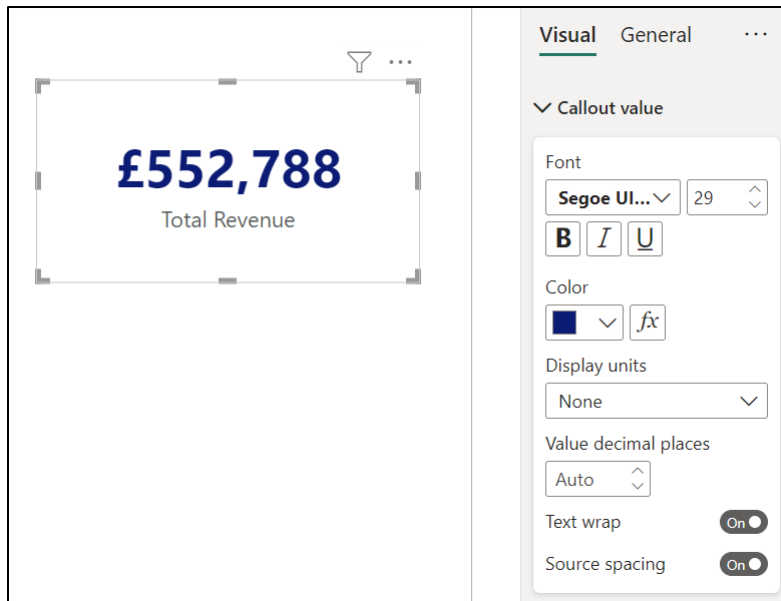


Figure 10.3: Formatting the callout value of a card visual

Category label versus title

The text shown below the callout value is the category label. This shows the name of the measure added to the Cards field. This is useful as it tells the audience what value they are looking at.

The category label has limited formatting options (*figure 10.4*), although these are often sufficient. In this example, the only formatting change applied is making the font bold.

1. Click on the **Callout value** to collapse the list of formatting options and free up some space.

2. Click **Category label** to expand the formatting options for this element.
3. Click on the **Bold** button.

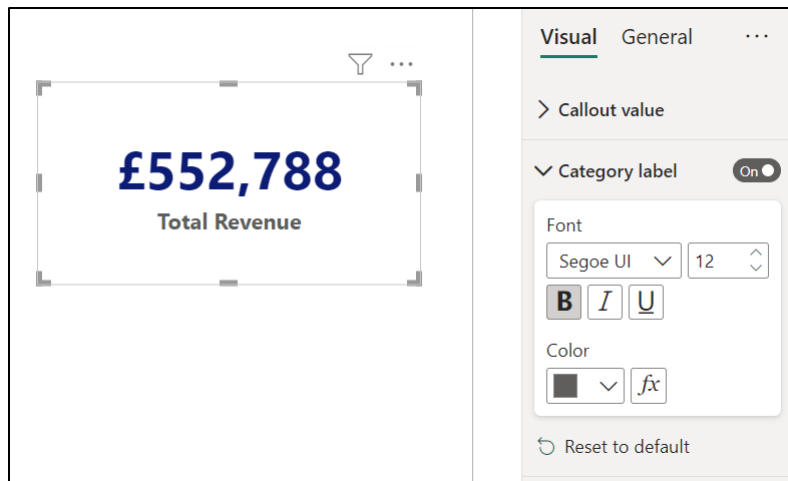


Figure 10.4: Formatting options for the category label

An alternative to the category label is to add a title. This is displayed above the callout value instead of below and offers additional formatting options (*figure 10.5*).

You can enter any title you wish. This is a nice advantage over the category label. You can also change elements such as the background color and text alignment, which you cannot do with the category label.

In *figure 10.5*, the title has been added along with an orange background color and white text color with bold style, and the text has been center aligned.

1. Click on the **General** category of formatting options.
2. Click on **Title** to expand the list of its formatting options, and click the *On/Off* slider button to turn the Title **On** and enable the options.
3. Click in the **Text** box and type **Total Revenue**.
4. Click on the **Font** list and select the **Segoe UI** font. This is consistent with the font of the callout value and is actually the default for the category label but not the title. Click the **Bold** button.
5. Click on the **Text color** list and select the white text color.
6. Click on the **Background color** list and select a dark orange color.
7. Click the **Center** button for the **Horizontal alignment**.

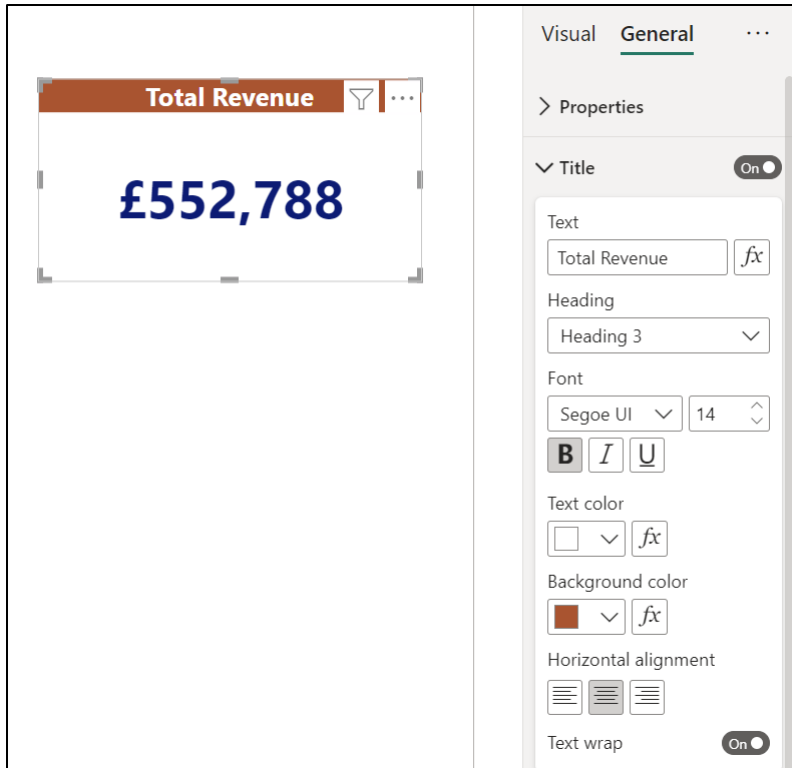


Figure 10.5: Formatting options for the title of a Card

Adding a border

The final formatting that will be applied to the Card is to add a light grey border (figure 10.6). It is subtle and will nicely define the edges of the visual.

1. Click on **Title** to collapse its list of formatting options.
2. Click on **Effects** to expand the options for this formatting element.
3. Click **Visual border** to expand the options available for the Card border, and click the *On/Off* slider to turn it **On**.
4. Click the **Color** list and select a light grey.

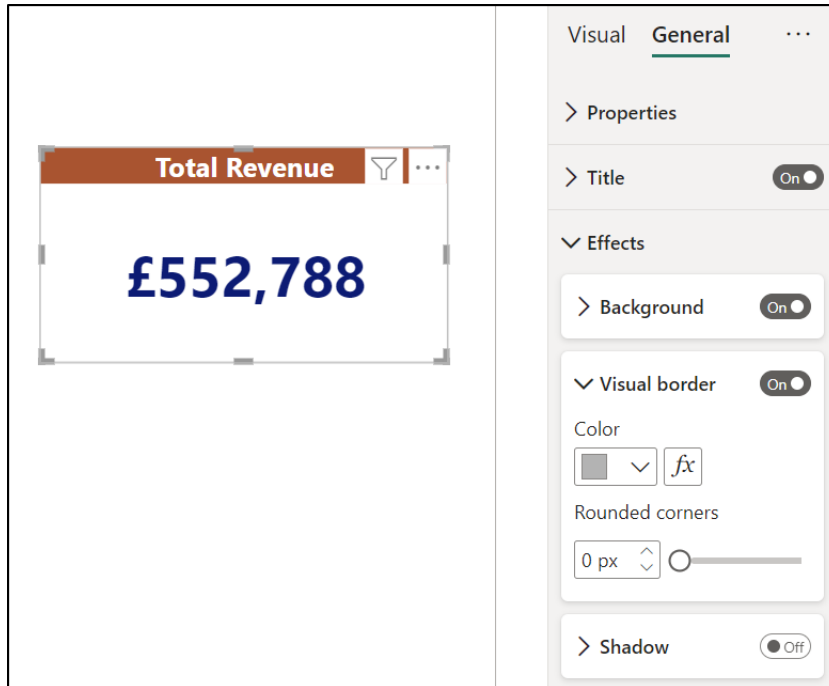


Figure 10.6: Adding a border to the Card visual

Format painter

We will now add a second Card to show the **Count of Sales** measure and use the Format painter button to quickly apply the same formatting to both visuals.

1. Click on a blank area of the report page to deselect the **Total Revenue** card visual.
2. Click the Card icon in the **Visualizations** pane.
3. Check the box for the **Count of Sales** measure in the **Data** pane to add it to the *Fields* well.
4. Click and drag both card visuals to the top left of the report page and align them accurately beside each other. Red dashed lines will appear to assist you with the alignment.

Figure 10.7 shows the second unformatted Card added to the report page.

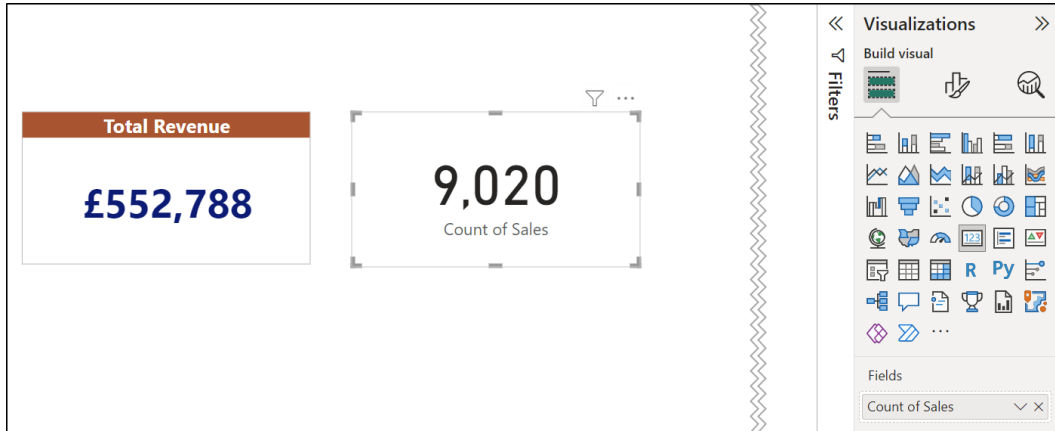


Figure 10.7: Second unformatted Card showing the Count of Sales measure

We will now copy the formatting from the first Card to the second using the Format painter command.

1. Click on the first Card visual.
2. Click **Home** | **Format painter** and then click on the second Card visual (figure 10.8).

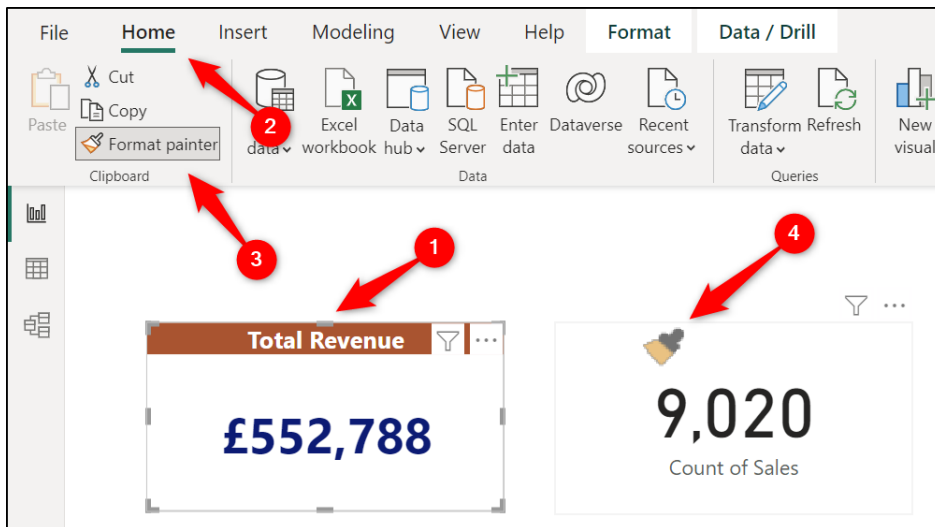


Figure 10.8: Using Format painter to copy formatting to another visual

Both cards now have the same formatting (figure 10.9). The border, fonts, colors, and the switch from the **Category label** to the **Title** have all been repeated on the second Card.

However, you will need to enter the text to be used for the title of the Card:

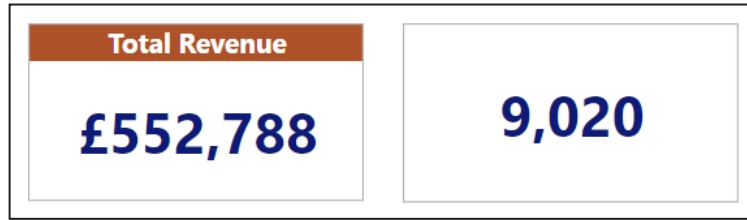


Figure 10.9: Formatted Card with a missing title

With the **Count of Sales** card selected;

1. Click on the **Format your visual** button above the gallery of visualizations in the **Visualizations** pane.
2. Click on the **General** category of formatting options.
3. Click on **Title** to expand the list of its formatting options, click in the Text box, and type “**Number of Sales**”.

Figure 10.10 shows the two completed cards.

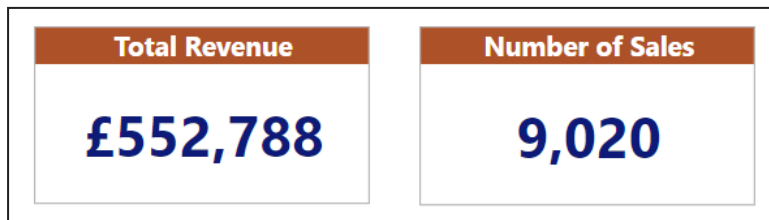


Figure 10.10: The completed cards for total revenue and number of sales

Note: You can also copy a visual on a page using the Copy and Paste buttons on the Home tab of the Ribbon, and then make the necessary edits.

The Format painter button is great for when the visual is already on the page, maybe created by someone else, and you need to quickly repeat formatting between visuals.

KPI

The **Key Performance Indicator (KPI)** visual presents the progress toward a specified goal or target. It presents the following:

- Easy recognition of whether you have achieved the target goal or not
- A measure of the distance that you are ahead or behind the target value.

The KPI requires a current value, a target value, and a trend. For our example, we will use the KPI to measure the sales value of a specified month against the sales value of the month previous to the one specified.

We will insert a KPI visual on a new page of our report that will be dedicated to monthly sales analysis.

1. Double-click on **Page 1** of the report, or right-click and click **Rename Page**, and type the name **Front Page**.
2. Click the **New page** button (plus icon beside the page tabs) to insert a new page named **Page 2**.
3. Double-click on **Page 2** and type the name **Monthly Sales Analysis** for the new page (figure 10.11).

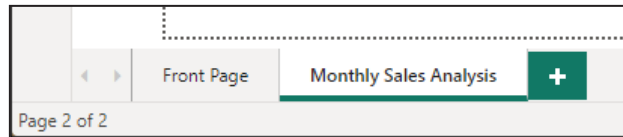


Figure 10.11: Pages of our report

4. Click the KPI card in the **Visualizations** pane (identified in figure 10.12).
5. Click and drag the **Total Revenue** measure into the *Value* well, the **Month Name** field from the **Calendar** table into the *Trend axis* well, and the **Revenue Previous Month** measure into the *Target* well (figure 10.12):

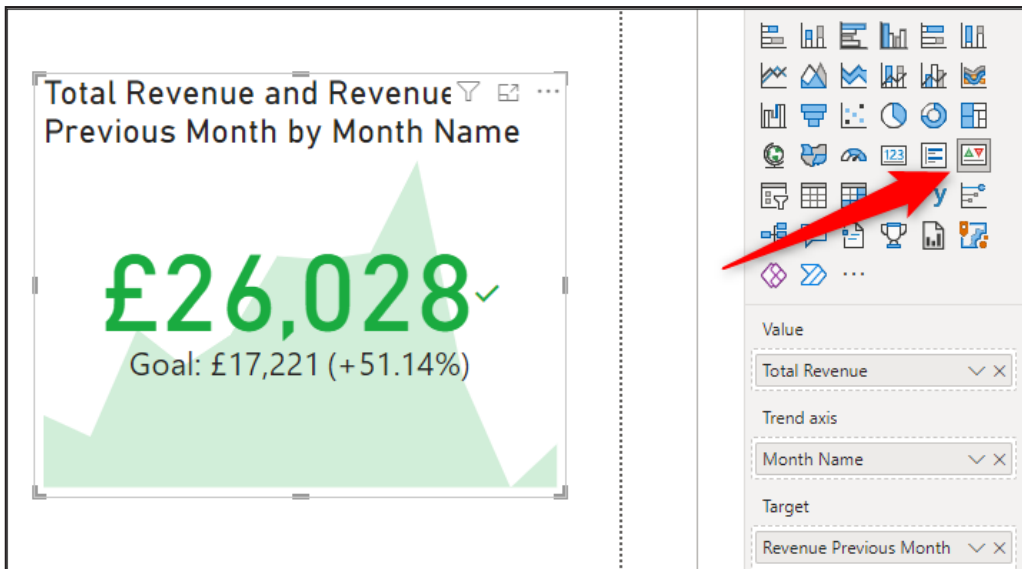


Figure 10.12: KPI visual showing the monthly sales trend and comparing the latest month's value against the previous month

In the KPI visual that is produced, you can see that we successfully outperformed the previous month's sales total. This is indicated by the green font, the green tick symbol, and the (+51.14%) variance.

You can also see an area chart in the background of the KPI showing the trend of sales totals for all 12 months of the year.

But which months do the sales total of £26,028 and a goal total of £17,221 refer to?

As shown in figure 10.13 with the Table visual, the KPI is using the latest month for the current value, and therefore, the previous month of November for the goal value of £17,221.

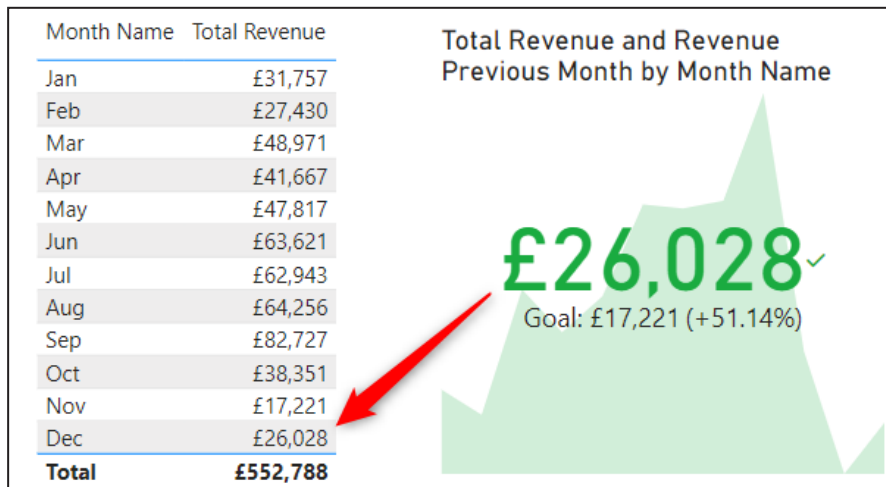


Figure 10.13: Table showing the months being used by the KPI visual

This could be exactly what you want. However, you would need to label the months being used so that it is easy for the reader to understand what they are looking at.

However, for this example, we want the reader to be able to choose the month to be used by the KPI.

So, before we begin formatting the KPI to our needs, we will provide a method for the reader to specify the month that they want to analyze sales. For this task, we will insert a Slicer.

Inserting a Slicer

A Slicer is a visual that enables a user to easily filter one or more visuals in a report. This will provide a simple way for the user to click a month name and have the KPI use that month as the context for its current value and the previous value.

Note: We will cover Slicers in more detail in *Chapter 14, Report Interactions, Filters, and Slicers*. This is just a quick example of using a Slicer to apply context for the KPI visual.

Click on a blank area of the report page to deselect the KPI visual.

1. Click the Slicer icon in the **Visualizations** pane.
2. Click and drag the **Month Name** field into the Field well of the Slicer (*figure 10.14*).

In *figure 10.14*, the Slicer and KPI visuals have been positioned and resized so that the Slicer sits neatly down the left side of the report page beside the KPI. The month of *Feb* has been specified in the Slicer, so the KPI now shows that total revenue decreased that month by 13.62% compared to the previous month of January.

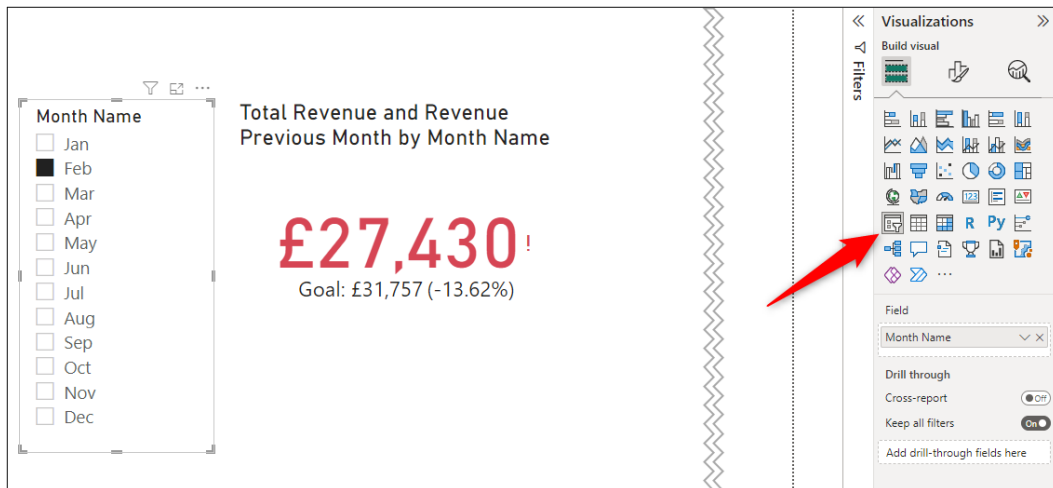


Figure 10.14: Adding a Slicer so the reader can specify a month easily

At the moment, it is possible for a user to specify multiple months by holding the *Ctrl* key down as they click month names in the Slicer. We do not want this behavior for our Slicer, so we will disable this functionality by specifying a single selection only.

1. With the Slicer visual selected, click the **Format your visual** button above the gallery of visuals.
2. Click the **Slicer settings** category to expand the formatting options.

3. Within the **Selection** sub-category, switch the **Single select** *On/Off* toggle to **On** (figure 10.15).

The checkboxes beside the month names in the Slicer are changed to option buttons.

Now a user can simply click a month name in the Slicer to switch, or filter, the KPI to that month.

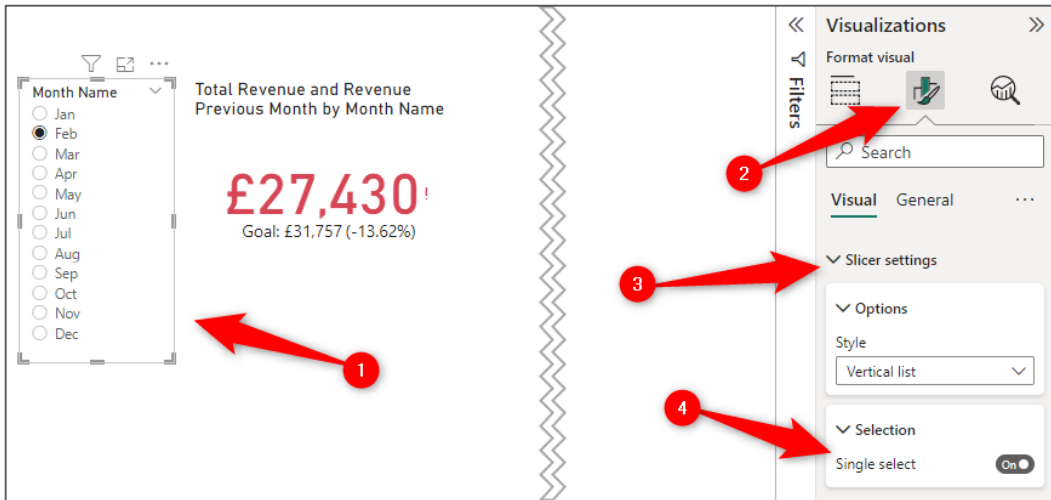


Figure 10.15: Specifying single select behavior for the Slicer

Let us now look at the formatting options for the KPI and make some modifications.

Removing the icon

The KPI shows a tick icon beside the callout value if the goal is achieved and an exclamation mark if it is not. The first formatting change we will make is to remove the icon from the KPI.

1. With the KPI visual selected, click the **Format your visual** button above the gallery of visualizations.
2. In the **Visual** category, click the *On/Off* slider button to turn **Off** the *Icons* (figure 10.16).

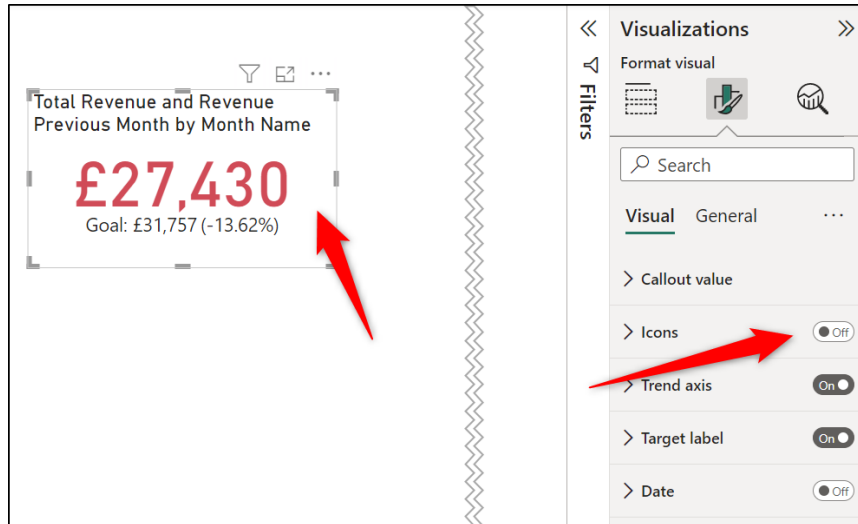


Figure 10.16: Removing the icon from a KPI

Trend axis options

There are some useful formatting options for the Trend Axis of a KPI (*figure 10.17*).

The default Direction is that High is good, and therefore, low is bad. In our example, this is true as we are presenting the total revenue. However, this option enables a change to Low is good if your KPI presents information such as the total number of refunds, complaints, or failed deliveries.

In *figure 10.17*, the month of August is selected in the Slicer to show the application of the Good color, due to the increase in sales between July and August.

You can change the good, neutral, and bad colors. The default is to use green for the Good Color, yellow for the Neutral Color, and red for the Bad Color. This can all be changed if required.

In this example, no changes have been made. Just an observation of the options available.

Note: The most common type of color blindness is the ability to notice the difference between the colors red and green.

The use of green, yellow, and red to present data is very commonplace and is often referred to as the RAG (red, amber, and green) status or traffic light system. However, it is useful to know your audience, and there is a case that more neutral colors such as blue and orange are better.

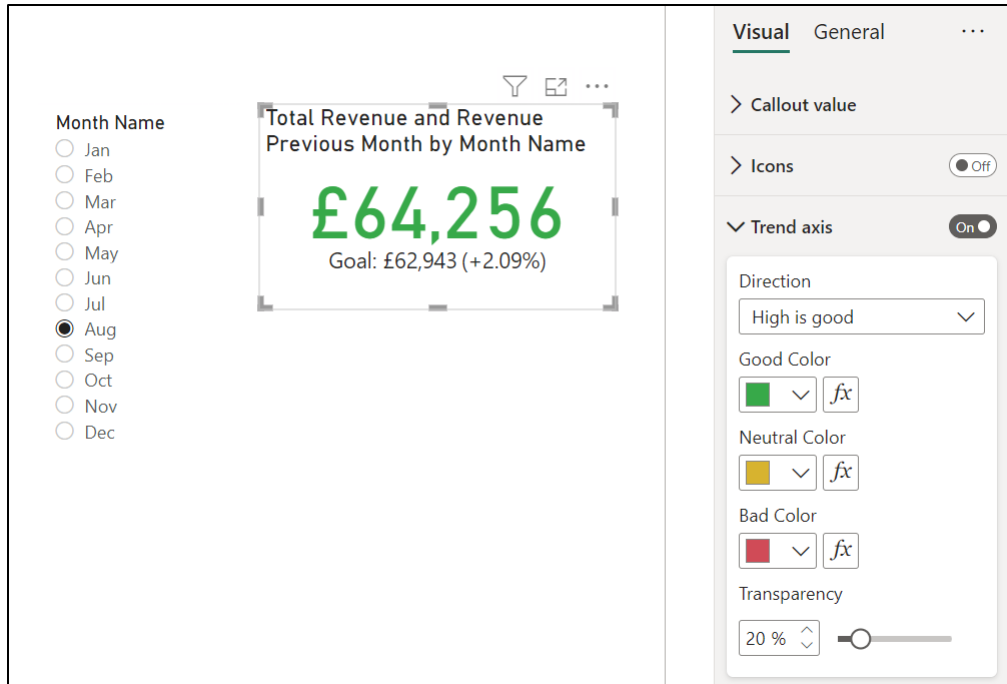


Figure 10.17: Formatting options for the Trend axis

Changing the target label

The default *Target label* is the text *Goal*. This is ok, but for our example, text such as *Last Month* would be more appropriate to the purpose of the KPI.

1. In the **Visual** section, click **Target label** to expand the formatting options.
2. Type **Last Month** into the Label box (figure 10.18).

There are options to change the font and other characteristics of the *Target label* text.

There is also an option to remove the *Distance to goal* (percentage displayed in brackets beside the target value) or change characteristics of the *Distance to goal*, such as whether to display it as a percent or value and if an increase in value is good or bad.

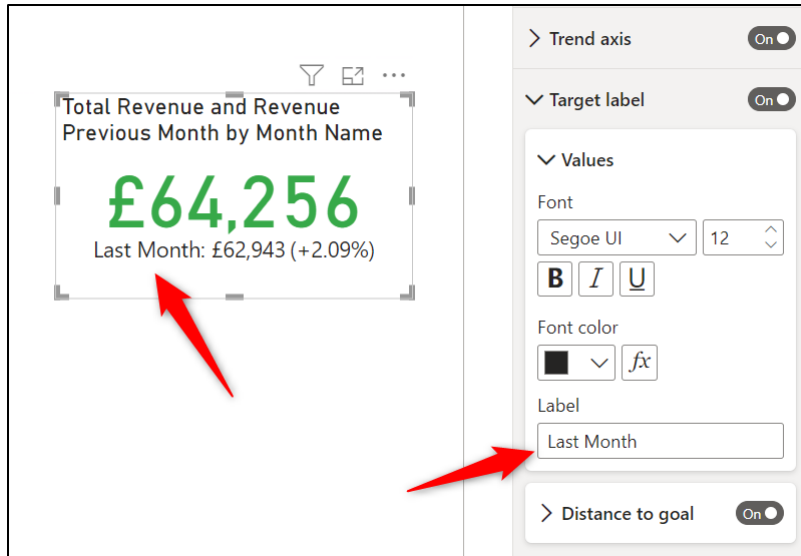


Figure 10.18: Changing the target label

Adding the title and border

To add the final polish to the Slicer and KPI visuals, we will add border and format aspects such as the font and title text to be consistent with the other visuals of our report.

Figure 10.19 shows the finished Slicer and KPI visuals.

To format the Slicer follow the following steps:

1. Click on the Slicer to select it.
2. Click the **Format your visual** button above the gallery of visualizations.
3. In the **Visual** section, click **Slicer header** to expand the options, select **Segoe UI** from the **Font** list, change the font size to **10**, and click the **Bold** button.
4. Click the **General** category header.
5. Click **Effects** to expand the options and click the *On/Off* slider button for the **Visual border** to turn it **On**. In the **Visual border** options, click the **Color** list and select a light grey.

The Slicer now uses the same Segoe UI font for its header and values as the other visuals of the report. It also contains the same light grey border.

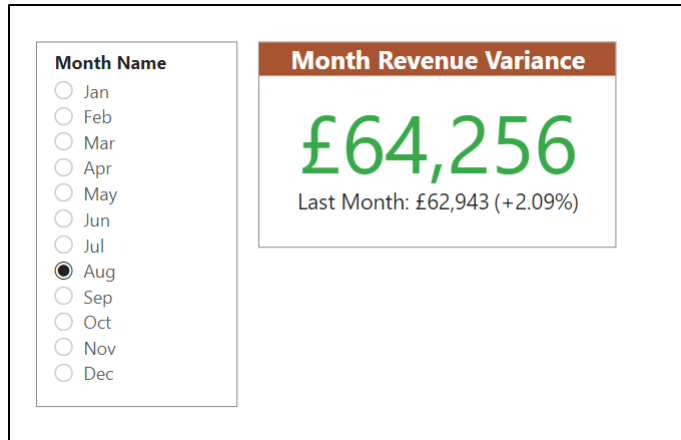


Figure 10.19: Completed Slicer and KPI visuals

Let us now format the KPI to also include the light grey border, Segoe UI font, and the same title format as the cards on the **Front Page** report page.

1. Click on the KPI visual and then click the **Format your visual** button.
2. In the **Visual** category, click **Callout value** to expand the options and select the **Segoe UI** font from the *Font* list.
3. Click the **General** category header.
4. Click **Effects** to expand the options and click the *On/Off* slider button for the **Visual border** to turn it **On**. In the **Visual border** options, click the *Color* list and choose light grey.
5. Click **Title** to expand the formatting options.
6. Click in the *Text* box and type **Month Revenue Variance**.
7. Click on the *Font* list and select the **Segoe UI** font and click the **Bold** button.
8. Click on the *Text color* list and select the white text color.
9. Click on the *Background color* list and select a dark orange color.
10. Click the **Center** button for the *Horizontal alignment*.

The two visuals are now completed and should appear as shown in *figure 10.19*.

Table visual

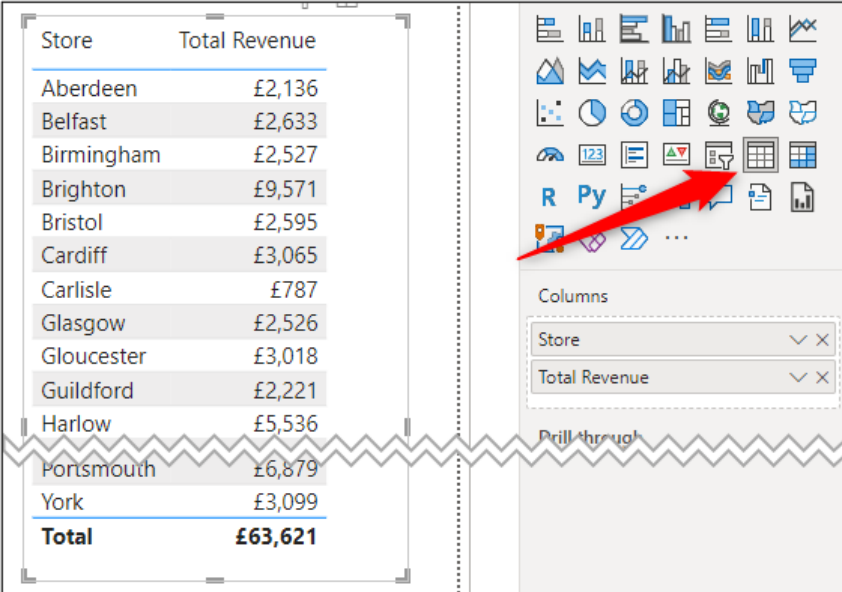
Using tables in Power BI reports is very common as users are familiar with tabular data from using spreadsheets.

Tables are probably overused and are not always the best visual to present data to compare values, view progress to a target, or measure variance.

However, they certainly do have their use cases, and we need to know how to create Table visuals in Power BI and what formatting options are at our disposal.

Let us insert a table visual to show the total revenue for each store.

1. Click on a blank part of the page to deselect all visuals.
2. Click on the Table visual in the **Visualizations** pane.
3. Click and drag the **Store** field and **Total Revenue** measure into the *Columns* well of the Table (figure 10.20).



Store	Total Revenue
Aberdeen	£2,136
Belfast	£2,633
Birmingham	£2,527
Brighton	£9,571
Bristol	£2,595
Cardiff	£3,065
Carlisle	£787
Glasgow	£2,526
Gloucester	£3,018
Guildford	£2,221
Harlow	£5,536
Portsmouth	£6,879
York	£3,099
Total	£63,621

Figure 10.20: Table showing revenue by store

Let us now format and change the order of the table values to meet our needs.

Formatting the grid and values

By default, the Table uses the Segoe UI font, which is one that we want. However, as standard, it also comes with a grey background color on the alternate rows and a thin blue border below the column headers and above the total row (please refer to figure 10.20).

To remove or change the grey alternate background color, follow the following steps:

1. With the Table visual selected, click the **Format your visual** button.
2. In the **Visual** category, click **Values** to expand the options.
3. Select a different color from the **Alternate background color** list (figure 10.21). In this example, white has been chosen to remove the alternate shading:

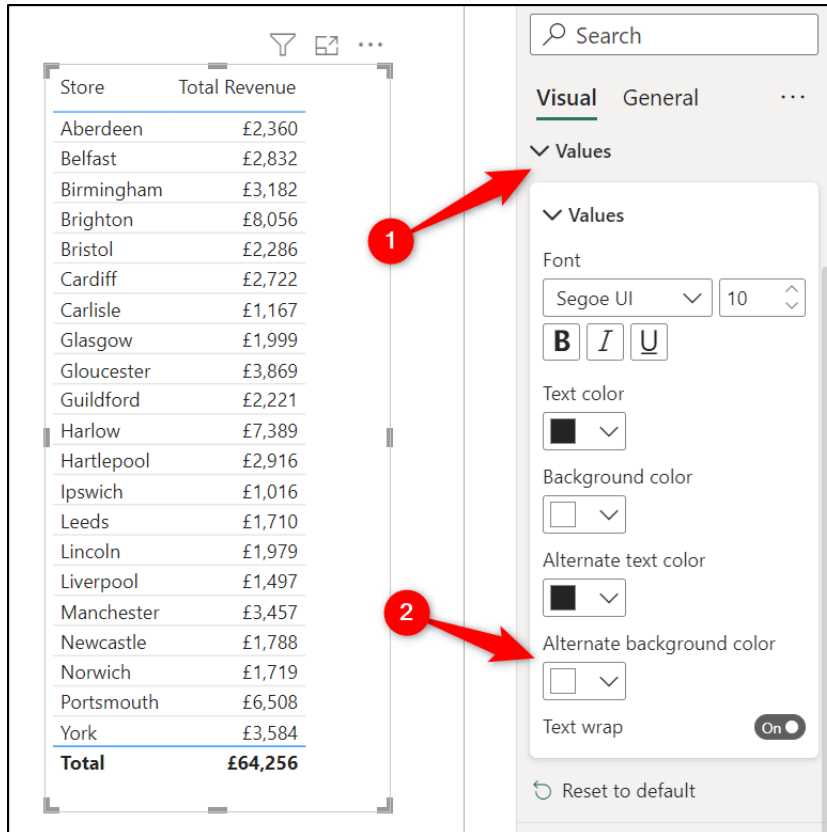


Figure 10.21: Removing the alternate background color

We will now change the thin blue border to dark orange to be consistent with the theme of the other visuals.

1. In the **Visual** category, click **Values** to collapse the options and click **Grid** to expand the options for that section.
2. Click **Border** to open the available options.
3. Click the **Color** list and change it to a dark orange (figure 10.22).

This changes the color of both borders. By clicking the *Section* list, you can specify different border formatting options for the different elements of the Table (headers, values, and totals).

There are other useful options in the *Grid* section, such as to add, remove, or change the formatting of the horizontal and vertical gridlines:

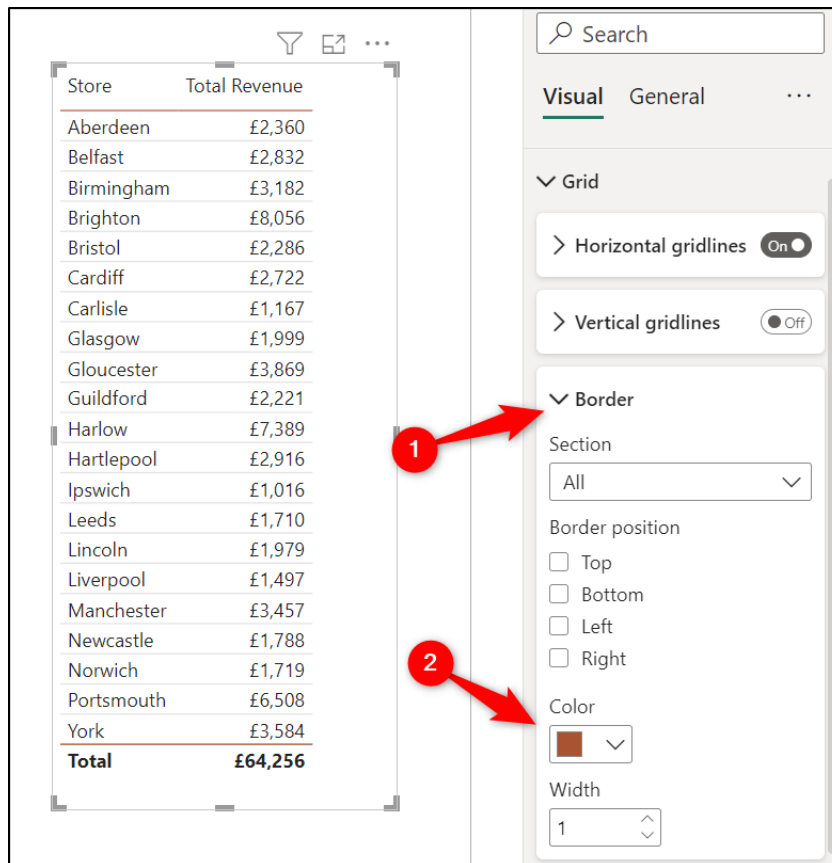


Figure 10.22: Changing the color of table grid borders

Formatting the column headers

We will now format the table column headers to be consistent with the titles of our Card and KPI visuals.

1. In the **Visual** category, click **Grid** to collapse the options and click **Column headers** to open the options for that section.
2. Click **Text**, if necessary, to expand its options.
3. Click the **Bold** button and change the **Text color** to white.

4. Change the **Background color** to the orange used in the other visuals (or whatever color you might be applying to your report visuals).

Store	Total Revenue
Aberdeen	£2,360
Belfast	£2,832
Birmingham	£3,182
Brighton	£8,056
Bristol	£2,286
Cardiff	£2,722
Carlisle	£1,167
Glasgow	£1,999
Gloucester	£3,869
Guildford	£2,221
Harlow	£7,389
Hartlepool	£2,916
Ipswich	£1,016
Leeds	£1,710
Lincoln	£1,979
Liverpool	£1,497
Manchester	£3,457
Newcastle	£1,788
Norwich	£1,719
Portsmouth	£6,508
York	£3,584
Total	£64,256

Figure 10.23: Formatting the column headers

Note: In Chapter 4, *Creating a Simple Power BI Report*, we saw an example of hiding the Totals in a Table visual.

Sorting table data

Sorting data in a table is simple. You just need to click the column header that you want to sort by until you get the order you require.

When you click the header of a column containing numeric data, it will sort descending first and then ascending on a second click. A column containing text data is the reverse.

Click the **Total Revenue** column header to sort the table data in descending order by the total revenue (*figure 10.24*). The table data is filtered by the month name Slicer on the page. The Table in *figure 10.24* shows the values for June:

Store	Total Revenue
Brighton	£9,571
Portsmouth	£6,879
Harlow	£5,536
Manchester	£3,301
York	£3,099
Hartlepool	£3,086
Cardiff	£3,065
Leeds	£1,716
Newcastle	£1,443
Ipswich	£1,226
Carlisle	£787
Total	£63,621

Figure 10.24: Table data ordered by total revenue in descending order

Sorting a table by multiple columns

Let us add a couple more columns to our Table.

1. With the Table visual selected, click the **Add data to your visual** button to switch from the **Format your visual** settings back to the *field wells*.
Click and drag the **Region** field from the **Stores** table into the *Columns* well between the **Store** and **Total Revenue** columns, as shown in *figure 10.25*.
2. Click and drag the **Count of Sales** measure from the **Sales** table into the *Columns* well as the last column of the Table (*figure 10.25*).

Store	Region	Total Revenue	Count of Sales
Brighton	South	£9,571	103
Portsmouth	South	£6,879	98
Harlow	East	£5,536	114
Manchester	North West	£3,301	56
York	North East	£3,099	50
Hartlepool	North East	£3,086	49
Cardiff	West	£3,065	44
Gloucester	West	£3,018	60
Belfast	West	£2,633	45
Bristol	West	£2,595	43
Birmingham	West	£2,527	41
Glasgow	North West	£2,526	39
Liverpool	North West	£2,249	23
Ipswich	East	£1,226	19
Carlisle	North West	£787	16
Total		£63,621	1,004

Figure 10.25: Region and Count of Sales fields added to the Table

All four columns are displayed in the order that they appear in the *Columns* well of the Table. You can drag the fields in the *Columns* well if you decide to change their order.

The table data is currently in descending order by the **Total Revenue** column. This is because we specified this previously.

Let us order the table data in ascending order by the **Region** column and then within each region, descending order by the **Total Revenue** column.

To sort a table by multiple columns:

1. Click on the **Region** column header to sort the Table in ascending order by region.
2. Press and hold the *Shift* key as you then click the **Total Revenue** column to add a second level sort of descending by total revenue (*figure 10.26*).

Store	Region	Total Revenue	Count of Sales
Harlow	East	£5,536	114
Lincoln	East	£2,168	34
Norwich	East	£1,741	33
Ipswich	East	£1,226	19
York	North East	£3,099	50
Hartlepool	North East	£3,086	49
Newcastle	North East	£1,443	28
Manchester	North West	£3,301	56
Guildford	South	£2,221	32
Cardiff	West	£3,065	44
Gloucester	West	£3,018	60
Belfast	West	£2,633	45
Bristol	West	£2,595	43
Birmingham	West	£2,527	41
Total		£63,621	1,004

Figure 10.26: Sort table by multiple columns

Note: Using Table visuals to create top N lists, such as Top 10 or Top 5 is an effective use of them. We will see an example of creating a top 10 list in *Chapter 14, Report Interactions, Filters, and Slicers*.

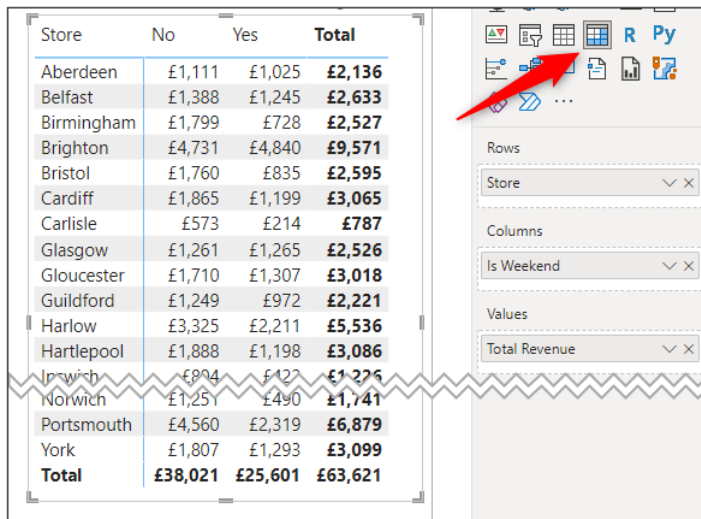
Matrix

The final text visual to cover in this chapter is the matrix.

A matrix provides the capacity to use labels in columns as well as rows, while a table only supports labels across rows. The matrix is like a PivotTable in Excel.

In this example, we will remove the Table visual and insert a matrix instead. This matrix will show the total revenue for each store split by the weekend and weekday.

1. Click on the Table visual (a good practice is to click the frame, or edge, of a visual when selecting it) and press *Delete* on the keyboard to remove it from the page.
2. Click the Matrix icon in the **Visualizations** pane (figure 10.27).
3. Click and drag the **Store** field from the **Stores** table into the *Rows* well of the matrix.
4. Click and drag the **Is Weekend** field from the **Calendar** table into the *Columns* well.
5. Click and drag the **Total Revenue** measure from the **Sales** table into the *Values* well (figure 10.27).



Store	No	Yes	Total
Aberdeen	£1,111	£1,025	£2,136
Belfast	£1,388	£1,245	£2,633
Birmingham	£1,799	£728	£2,527
Brighton	£4,731	£4,840	£9,571
Bristol	£1,760	£835	£2,595
Cardiff	£1,865	£1,199	£3,065
Carlisle	£573	£214	£787
Glasgow	£1,261	£1,265	£2,526
Gloucester	£1,710	£1,307	£3,018
Guildford	£1,249	£972	£2,221
Harlow	£3,325	£2,211	£5,536
Hartlepool	£1,888	£1,198	£3,086
Inverness	£804	£127	£931
Norwich	£1,251	£490	£1,741
Portsmouth	£4,560	£2,319	£6,879
York	£1,807	£1,293	£3,099
Total	£38,021	£25,601	£63,621

Figure 10.27: Matrix visual showing total revenue by store and weekday versus weekend

The default formatting of the matrix is similar to that of the Table visual. It has a bold font in the total column and row. Light grey background color for alternate rows. A thin blue border under the column labels and to the right of the row labels. And the font is Segoe UI.

Formatting the grid and values

Let us start by formatting the grid and values of the matrix. This is a similar task to the table formatting that we did previously. We will remove the grey background color on alternate rows, remove the blue border to the right of the row labels, and keep the border under the column labels but change it to a dark orange color (*figure 10.28*).

1. With the matrix selected, click the **Format your visual** button in the **Visualizations** pane.
2. In the **Visual** category, click **Values** to expand the options in that section.
3. Change the **Alternate background color** from grey to white.
4. Click **Values** to collapse its options and click **Grid** to expand the options in that section.
5. Click **Border** within the *Grid* section to open its options.
6. Click the *Color* list and change it to dark orange.
7. Click the *Section* list and select **Row headers**.
8. Uncheck the **Right** box to remove the border to the right of the row labels (*figure 10.28*).

Store	No	Yes	Total
Aberdeen	£1,111	£1,025	£2,136
Belfast	£1,388	£1,245	£2,633
Birmingham	£1,799	£728	£2,527
Brighton	£4,731	£4,840	£9,571
Bristol	£1,760	£835	£2,595
Cardiff	£1,865	£1,199	£3,065
Carlisle	£573	£214	£787
Glasgow	£1,261	£1,265	£2,526
Gloucester	£1,710	£1,307	£3,018
Guildford	£1,249	£972	£2,221
Harlow	£3,325	£2,211	£5,536
Hartlepool	£1,888	£1,198	£3,086
Ipswich	£804	£422	£1,226
Leeds	£1,064	£653	£1,716
Lincoln	£1,711	£457	£2,168
Liverpool	£1,001	£1,347	£2,349
Manchester	£2,267	£1,035	£3,301
Newcastle	£897	£546	£1,443
Norwich	£1,251	£490	£1,741
Portsmouth	£4,560	£2,319	£6,879
York	£1,807	£1,293	£3,099
Total	£38,021	£25,601	£63,621

Figure 10.28: Removing the row headers border from a matrix

Formatting the column headers

We will now apply the same formatting to the column headers of the matrix that we did to the Table visual (*figure 10.29*).

1. In the *Visual* category, click **Column headers** to expand the formatting options.
2. If necessary, click **Text** to open its options.
3. Click the *Bold* button and change the **Text color** to white.
4. Change the **Background color** to a dark orange (*figure 10.29*):

Store	No	Yes	Total
Aberdeen	£1,111	£1,025	£2,136
Belfast	£1,388	£1,245	£2,633
Birmingham	£1,799	£728	£2,527
Brighton	£4,731	£4,840	£9,571
Bristol	£1,760	£835	£2,595
Cardiff	£1,865	£1,199	£3,065
Carlisle	£573	£214	£787
Glasgow	£1,261	£1,265	£2,526
Gloucester	£1,710	£1,307	£3,018
Guildford	£1,249	£972	£2,221
Harlow	£3,325	£2,211	£5,536
Hartlepool	£1,888	£1,198	£3,086
Ipswich	£804	£422	£1,226
Leeds	£1,064	£653	£1,716
Lincoln	£1,711	£457	£2,168
Liverpool	£1,001	£1,347	£2,349
Manchester	£2,267	£1,035	£3,301
Newcastle	£897	£546	£1,443
Norwich	£1,251	£490	£1,741
Portsmouth	£4,560	£2,319	£6,879
York	£1,807	£1,293	£3,099
Total	£38,021	£25,601	£63,621

Figure 10.29: Formatting the column headers of a matrix

Tip: If you have a visual on the page selected at the time that you click a visual icon in the visualizations pane, it will convert the selected visual. In this example, converting the Table visual to a matrix could save time on formatting and other tasks compared to inserting a matrix from scratch.

Drill down actions for multiple row headers

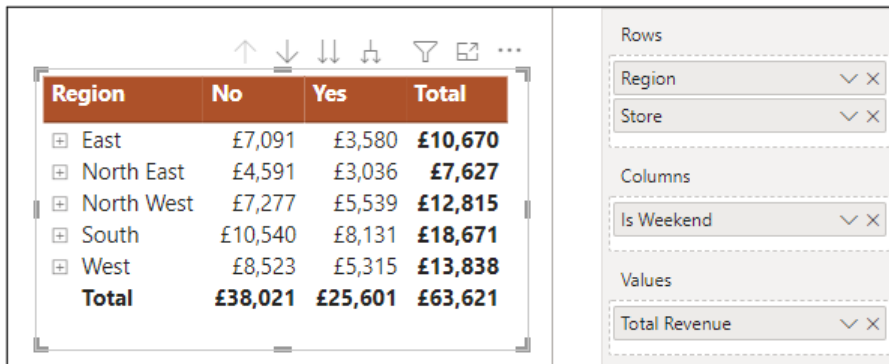
When using multiple row headers in a matrix, drill down actions are enabled. These actions provide different methods for a user to drill down or display the data at the different levels in the matrix.

Note: The drill down actions detailed here can also be performed on column headers in a matrix.

Let us add the **Region** field to the existing **Store** field in the row headers of the matrix.

1. Click the **Add data to your visual** button to switch from the **Format your visual** part of the **Visualizations** pane back to where we can add and remove fields.
2. Click and drag the **Region** field from the **Stores** table into the **Rows** well of the matrix and before the **Store** field (*figure 10.30*).

The two-row headers form a group, and this is collapsed to show the first level of data only, which is the region in this example (*figure 10.30*). The header icons above the matrix provide the functionality to drill down and expand the row headers:



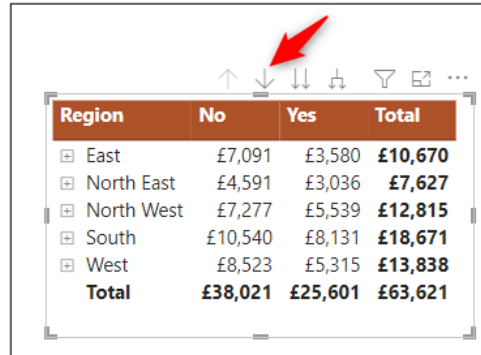
Region	No	Yes	Total
East	£7,091	£3,580	£10,670
North East	£4,591	£3,036	£7,627
North West	£7,277	£5,539	£12,815
South	£10,540	£8,131	£18,671
West	£8,523	£5,315	£13,838
Total	£38,021	£25,601	£63,621

Figure 10.30: Region and store fields in the Rows well of the matrix

Now, you can click the plus icons beside the region names to expand that group and see the data within, expand all the groups with one click, or use the drill down functionality to switch from viewing the first level of the hierarchy to the second level (the store in this example).

Let us explore these different options.

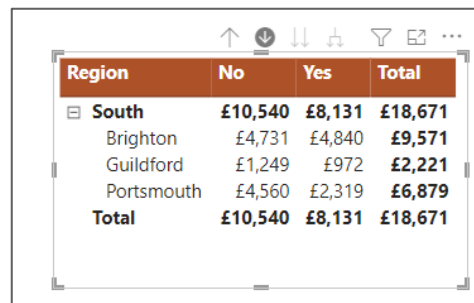
Click the drill down icon shown in *figure 10.31* to turn on the drill down function:



Region	No	Yes	Total
East	£7,091	£3,580	£10,670
North East	£4,591	£3,036	£7,627
North West	£7,277	£5,539	£12,815
South	£10,540	£8,131	£18,671
West	£8,523	£5,315	£13,838
Total	£38,021	£25,601	£63,621

Figure 10.31: Turn on the drill down functionality

With this functionality enabled, clicking a region name in the matrix will drill down to the store level of the hierarchy for that region only. Figure 10.32 shows the results of clicking the South region:

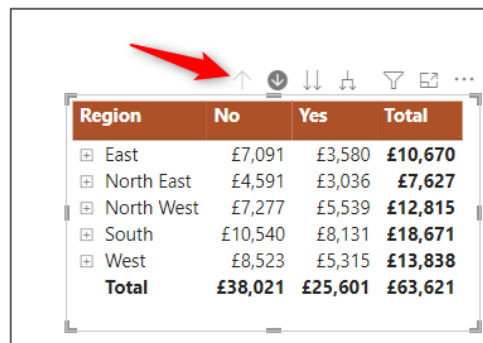


Region	No	Yes	Total
South	£10,540	£8,131	£18,671
Brighton	£4,731	£4,840	£9,571
Guildford	£1,249	£972	£2,221
Portsmouth	£4,560	£2,319	£6,879
Total	£10,540	£8,131	£18,671

Figure 10.32: Looking at sales from the South stores only

Notice the drill down arrow icon has changed to be enclosed in a black circle. This indicates to others that drill down behavior is enabled.

Click the up arrow in the header icons to back up to the next level up in the hierarchy. For this example, that is back to the regional sales (figure 10.33):

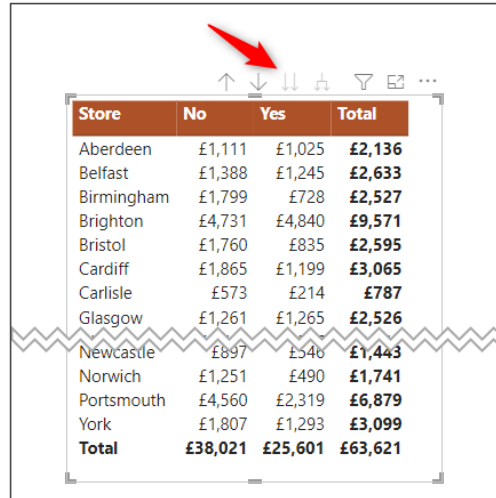


Region	No	Yes	Total
East	£7,091	£3,580	£10,670
North East	£4,591	£3,036	£7,627
North West	£7,277	£5,539	£12,815
South	£10,540	£8,131	£18,671
West	£8,523	£5,315	£13,838
Total	£38,021	£25,601	£63,621

Figure 10.33: Drill up arrow to back up to the next level in the hierarchy

To turn off the drill down functionality, simply click the drill down icon again.

Click the double arrow icon to go to the next level of the hierarchy for all regions (figure 10.34):



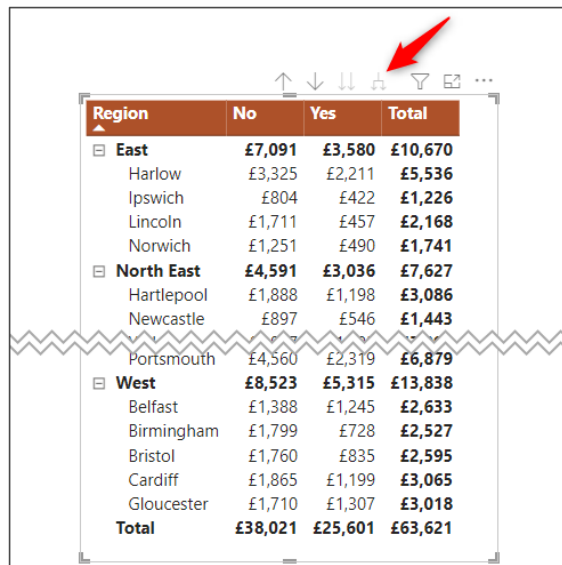
The screenshot shows a data table with a header row and a body of rows. The header row has columns: Store, No, Yes, and Total. The body rows list various cities and their corresponding values. A red arrow points to the double arrow icon in the header icons area.

Store	No	Yes	Total
Aberdeen	£1,111	£1,025	£2,136
Belfast	£1,388	£1,245	£2,633
Birmingham	£1,799	£728	£2,527
Brighton	£4,731	£4,840	£9,571
Bristol	£1,760	£835	£2,595
Cardiff	£1,865	£1,199	£3,065
Carlisle	£573	£214	£787
Glasgow	£1,261	£1,265	£2,526
Newcastle	£897	£546	£1,443
Norwich	£1,251	£490	£1,741
Portsmouth	£4,560	£2,319	£6,879
York	£1,807	£1,293	£3,099
Total	£38,021	£25,601	£63,621

Figure 10.34: Jump to the next level of the hierarchy

Click the up arrow in the header icons to back up to the next level up in the hierarchy again.

The final option is the pitchfork icon (figure 10.35). Click this icon to expand all labels at once.



The screenshot shows a data table with a header row and a body of rows. The header row has columns: Region, No, Yes, and Total. The body rows are grouped by region: East, North East, and West. A red arrow points to the pitchfork icon in the header icons area.

Region	No	Yes	Total
East	£7,091	£3,580	£10,670
Harlow	£3,325	£2,211	£5,536
Ipswich	£804	£422	£1,226
Lincoln	£1,711	£457	£2,168
Norwich	£1,251	£490	£1,741
North East	£4,591	£3,036	£7,627
Hartlepool	£1,888	£1,198	£3,086
Newcastle	£897	£546	£1,443
Portsmouth	£4,560	£2,319	£6,879
West	£8,523	£5,315	£13,838
Belfast	£1,388	£1,245	£2,633
Birmingham	£1,799	£728	£2,527
Bristol	£1,760	£835	£2,595
Cardiff	£1,865	£1,199	£3,065
Gloucester	£1,710	£1,307	£3,018
Total	£38,021	£25,601	£63,621

Figure 10.35: Expand all labels in the hierarchy

Click the up arrow in the header icons to back up to the next level up in the hierarchy again.

Note: The Header icons can be turned off to prevent users from accessing this functionality when reading the report if required. This setting is found in the General section of the formatting options. The header icons are only disabled in the reading view, so do not worry when you can still see them on the report page.

Conclusion

In this chapter, we learnt how to use four text-based visuals in Power BI—the Card, KPI, Table, and matrix. With each visual, we covered some useful formatting features available to them and some pro tips for their use.

In the upcoming chapter, we will look at chart visuals such as the column, line, donut, and gauge charts. We will explain the intricacies of each chart visual and explore the more important formatting options for each.

Questions

Here are some questions to test what you have learnt in this chapter.

1. What are the names of the two categories of formatting options available for all visuals?
 - a. Distinct and General
 - b. Visual and General
 - c. Visual and Basic
 - d. None of the above.
2. You can sort by multiple columns in a Table?
 - a. True
 - b. False
3. Which of the following are the names of fields for the KPI visual?
 - a. Trend axis
 - b. Value
 - c. Target
 - d. All of the above

4. With the Format Painter command, you can quickly copy the format of one visual to another.
 - a. True
 - b. False

5. With the KPI visual, you cannot change the good, bad, and neutral colors of the Trend axis text?
 - a. True
 - b. False

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



CHAPTER 11

Chart Visuals

Introduction

In this chapter, we focus on the chart visuals in Power BI. These include the column, line, combo, gauge, and treemap visuals.

For each visualization, we will see how to create them with a practical example of their use and learn key formatting attributes that can be changed to increase the effectiveness of the visual.

Structure

In this chapter, we will cover the following topics:

- Create popular chart visuals, including the column, line, donut, and treemap visuals.
- Adding data labels
- Key attributes and formatting options for each chart.

Objectives

After reading this chapter, you will know how to use the most important chart visuals in Power BI. You will also learn how to change key attributes and format options to increase the effectiveness of the visuals.

Column and bar charts

Files: **sales-report-ch-11-start.pbix**

Column and bar charts are two of the most used charts in reporting. They are used to easily compare the values of different categories.

These charts are very familiar to people, making them reliable for your audience to read and understand. They are also very functional and come with many settings to modify them to our needs in Power BI.

There are six different column and bar charts available as standard in Power BI. These are variations of the clustered and stacked column and bar charts. We will look at both the clustered and stacked column chart and understand their differences before then switching to a bar chart variation.

Clustered column chart

We will begin with a clustered column chart, quite possibly the most recognized chart around. Let us use this column chart to compare total revenue across the different regions.

1. Click on the **Clustered column chart** icon in the **Visualizations** pane (*figure 11.1*).
2. Click and drag the **Region** field from the **Stores** table into the *X-axis* well.
3. Click and drag the **Total Revenue** measure into the *Y-axis* well.
4. Move and resize the column chart to position it underneath the two card visuals and align it neatly, as shown in *figure 11.1*.

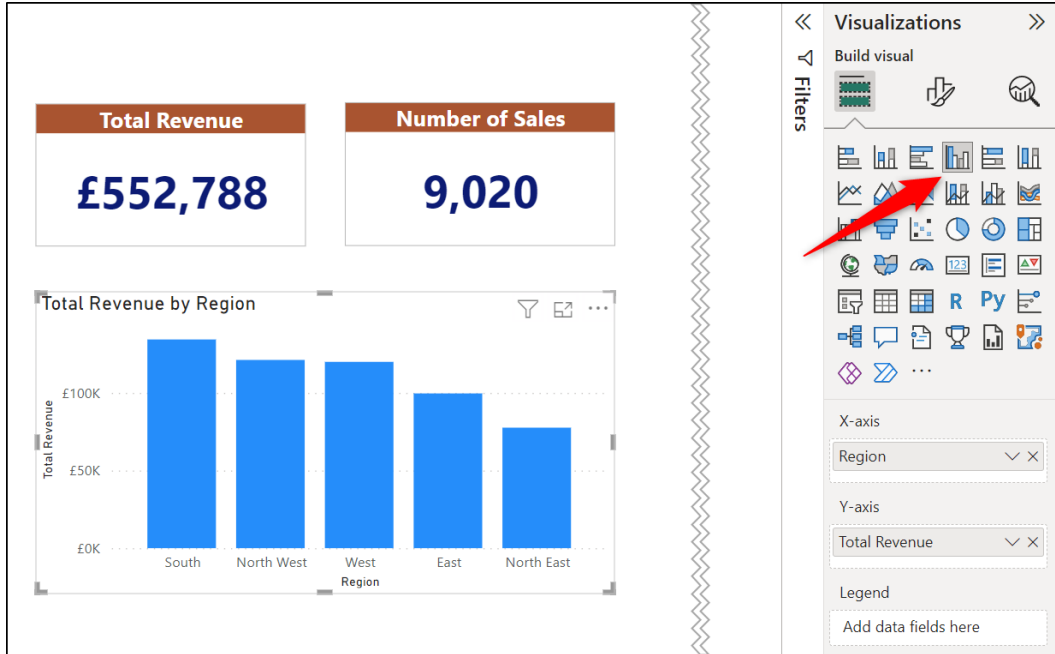


Figure 11.1: Column chart comparing total revenue across regions

Adding tooltips

By using the tooltips feature of Power BI visuals, you can make it easy for the reader to see further detail on the data points of a visual.

Tooltips are a feature of most Power BI visuals, including the column, pie, map, line, and gauge charts.

The tooltip is shown when you mouse over a data point. *Figure 11.2* shows a tooltip for the West data point of the column chart. This is the default tooltip and shows only the data being used by the column chart, that is, the region and total revenue:

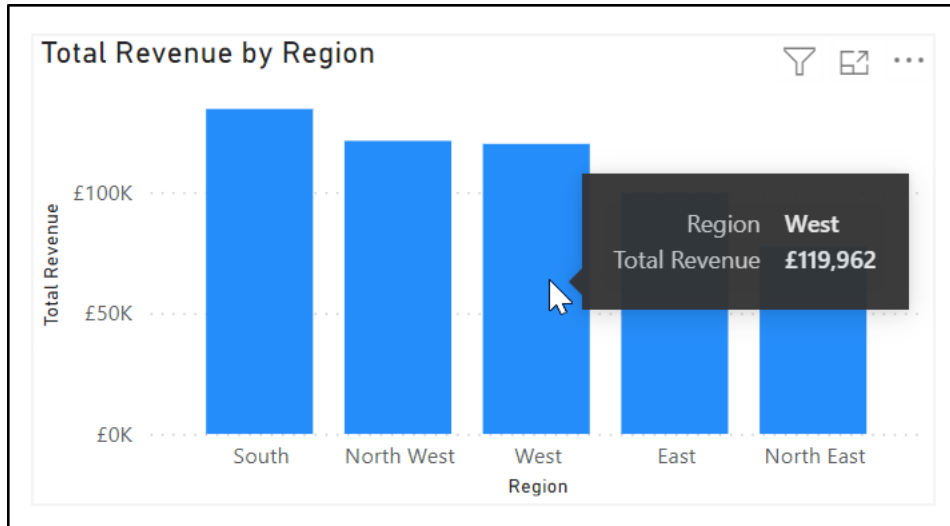


Figure 11.2: Tooltip for the West data point of the column chart

By adding fields to the *Tooltips* well of the visual, you can add additional data points that you think the reader would be interested in viewing.

1. Click the column chart to ensure that it is active.
2. Click and drag the **Count of Sales** and **Total Units Sold** measures to the *Tooltips* well as shown in figure 11.3:

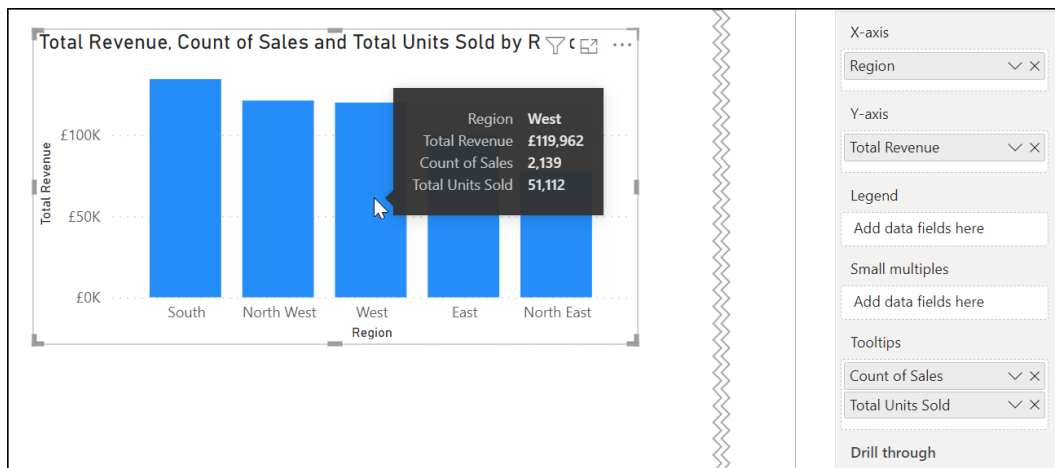


Figure 11.3: Custom tooltip showing additional data points

Now on mouse over, the results of the **Count of Sales** and **Total Units Sold** measures for that data point (West in *figure 11.3*) are also shown.

Tooltips are a great way to provide additional insight into the performance of a data point.

Note: In *Chapter 15, Enhancing your Power BI Reports*, we will see how you can create your own custom tooltip pages. This gives you unrivaled control over what the tooltip looks like, down to its size, background color, fonts, and so on. But most importantly, you can add whatever visuals and content that you want. Tooltip pages are a very powerful feature of Power BI.

Formatting the clustered column chart

We will now change a few formatting options to improve the chart from its default appearance.

The formatting changes we will make include the following:

- Changing the chart title. The default title is made up of the fields added to the wells of the clustered column chart. This can get unsightly when additional fields are added.
- Formatting the chart title to be consistent with the titles of the two card visuals above.
- Adding a light grey border.
- Removing the x-axis label. The “Region” label is not providing any, or much, benefit to this chart. Readers know that the South, Northwest, and West are regions.
- Swapping the y-axis for data labels. With only five columns, adding data labels as an alternative to the y-axis labels can be more effective.
- Change the color of the columns. We will not keep with the default blue.

Changing the chart title

The chart title currently being used is terrible, as you can see in *figure 11.4*. It is dynamically updated to show all information added to the wells of the visual. This includes the fields and measures that were added to the x- and y-axis and the two fields that were just added to the tooltips well.

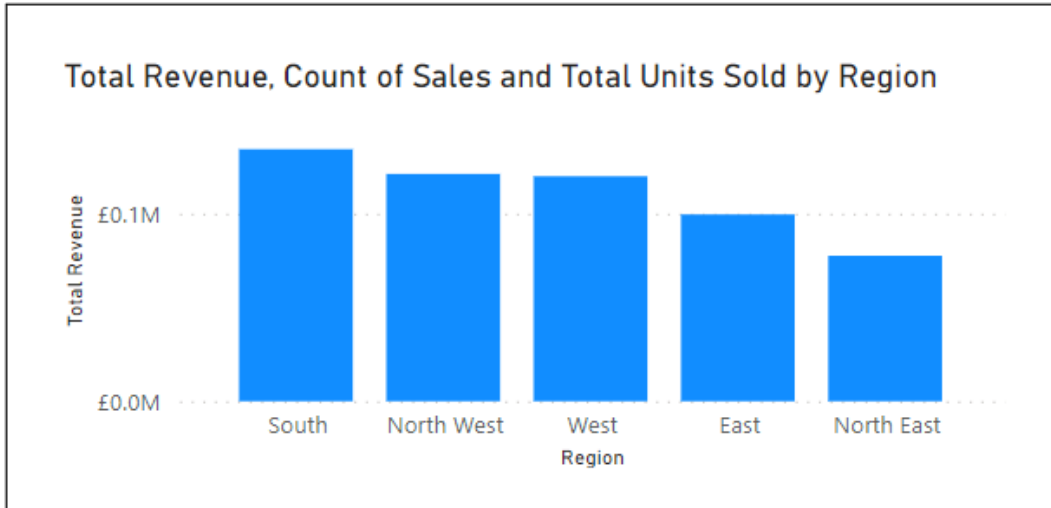


Figure 11.4: Default chart title listing information from all fields of the visual

We will edit the chart title text to exclude the mention of the measures added to tooltips and format the title to be consistent with the titles of the cards above and the KPI on the other report page (*figure 11.5*).

1. With the column chart selected, click the **Format your visual** button at the top of the *Visualizations* pane.
2. Click on the **General** category and click on **Title** to expand the list of its formatting options.
3. Click in the *Text* box and edit the text to read “**Total Revenue by Region**”.
4. Click on the *Font* list and select the **Segoe UI** font. Click the **Bold** button.
5. Click on the **Text color** list and select the white text color.
6. Click on the **Background color** list and select a dark orange color.
7. Click the **Center** button for the *Horizontal alignment* as shown in *figure 11.5*:

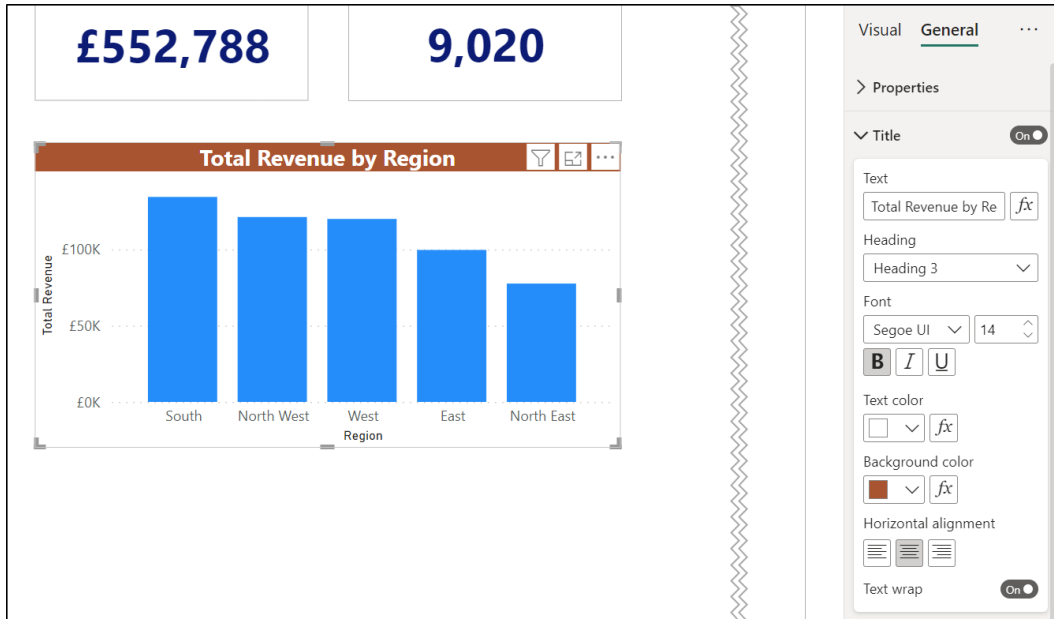


Figure 11.5: Column chart with the improved chart title

Note: Using DAX, you can construct more insightful titles for your visuals than is demonstrated in these simple examples. What does the reader want to know? You can create dynamic titles that present important details and answer questions the readers may be looking for.

Adding a border

Let us add a light grey border to the clustered column chart, just as we did to the two cards above.

1. Click on **Title** to collapse its list of formatting options, if necessary.
2. Click on **Effects** to expand the options for this formatting element.
3. Click **Visual border** to expand the options available for the chart's border, and click the *On/Off* slider to turn it **On**.
4. Click the **Color** list and select a light grey.

Changing the column colors

We will now change the column colors to something other than the default blue.

1. Click on the **Visual** category and click **Columns** to expand its formatting options.
2. Click the **default** list and choose a purple color (*figure 11.6*).

The color of all columns is changed.

Figure 11.6 shows the **Show all** slider set to **On**. This permits you to change the color of specific columns.

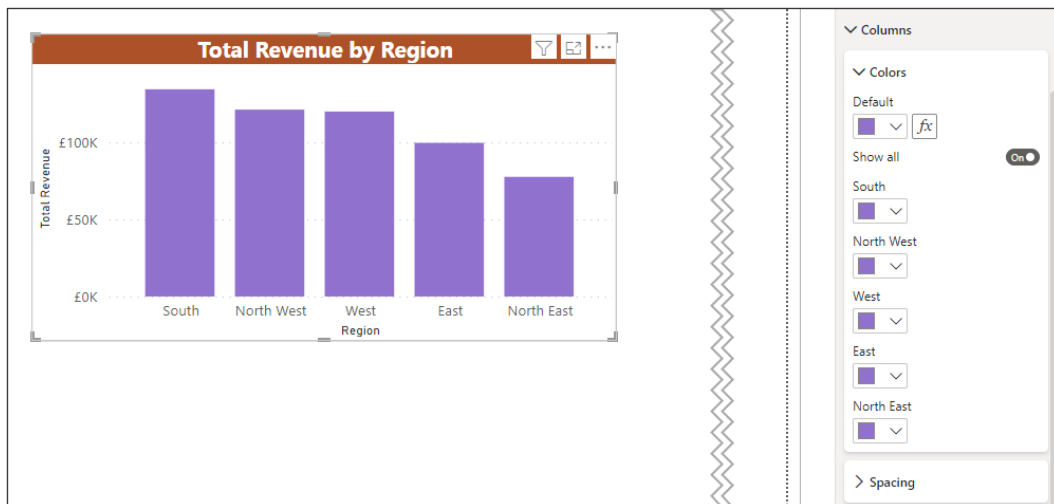


Figure 11.6: Changing the column colors

Note: The Conditional Formatting button (fx icon) beside the Default list is very useful. You can define rules that determine the color to be applied.

Many Power BI visuals permit the application of Conditional Formatting rules for font or area colors. These include the card, map, column chart, and table visuals.

Removing axis and axis titles

Let us now remove the y-axis from the chart and remove both the axis titles. Despite removing the y-axis, we still need to remove the y-axis title in the formatting options. Surprisingly, removing the axis does not automatically remove its title.

1. Click the *On/Off* slider for the *Y-axis* to turn it to **Off**.
2. Click the **Y-axis** to expand the formatting options and click the *On/Off* slider for the **Title** to turn it to **Off**.
3. Click the **X-axis** to expand the formatting options and click the *On/Off* slider for the **Title** to turn it to **Off**, as depicted in *figure 11.7*:

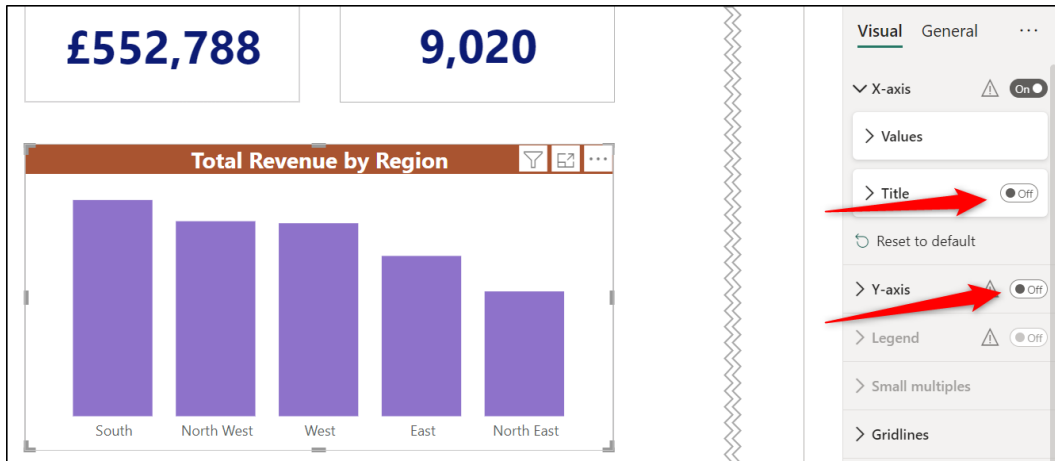


Figure 11.7: Removing the y-axis and both axis titles

Adding data labels

Data labels can now be added to display the values for each data point. When a column chart only has a few columns, such as the five in this example, data labels can be an effective alternative to the y-axis. If there are many columns, data labels can produce a noisy and cluttered chart.

1. Click the *On/Off* slider for the **Data labels** to turn it **On**, then click **Data labels** to expand its formatting options.
2. Click the **Position** list within the **Options** category and choose **Inside end** for the position of the labels, as shown in *figure 11.8*. The *Auto* option positions the labels outside the end.
3. Click **Values** to expand the options, and click the **Bold** button to emphasize the labels more.

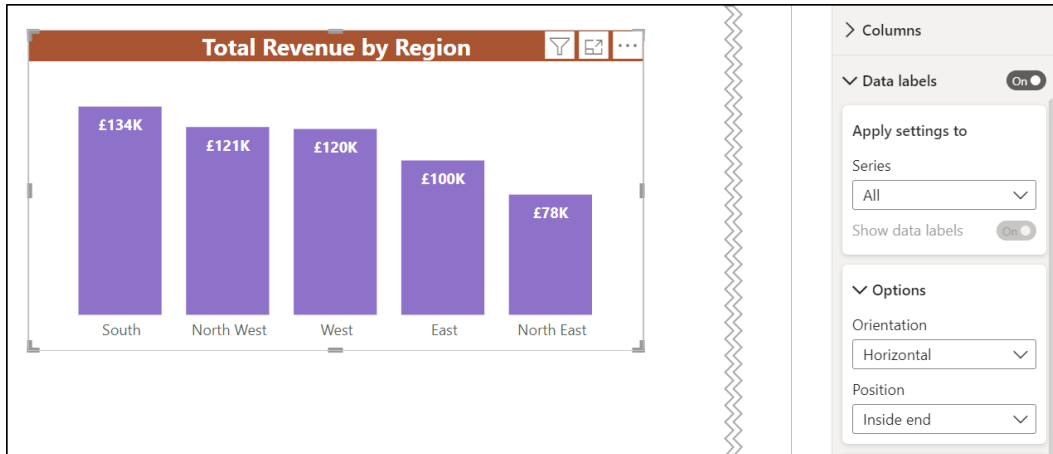


Figure 11.8: Adding data labels to the column chart

Stacked column chart

A stacked column chart provides the ability to compare parts to a whole in addition to comparing the values across different categories. For each category, the data series are stacked on top of one another in vertical columns.

For this example, we will add the **Category** column from the **Products** table so that we can see the contribution each category made to the total revenue for each region.

1. With the clustered column chart selected, click the **Add data to your visual** button to see the gallery of visualizations.
2. Click the **Stacked column chart** icon to convert the existing clustered column chart into a stacked column chart.
3. Click and drag the **Category** field from the **Products** table into the *Legend* well (figure 11.9).

Unfortunately, some of the data labels are not shown for the *Cakes* series, as the part of the column is too small to fit the label, as shown in figure 11.9.

Because of this, we will remove the data labels for each series and replace them with labels that show the total revenue value only for each category.

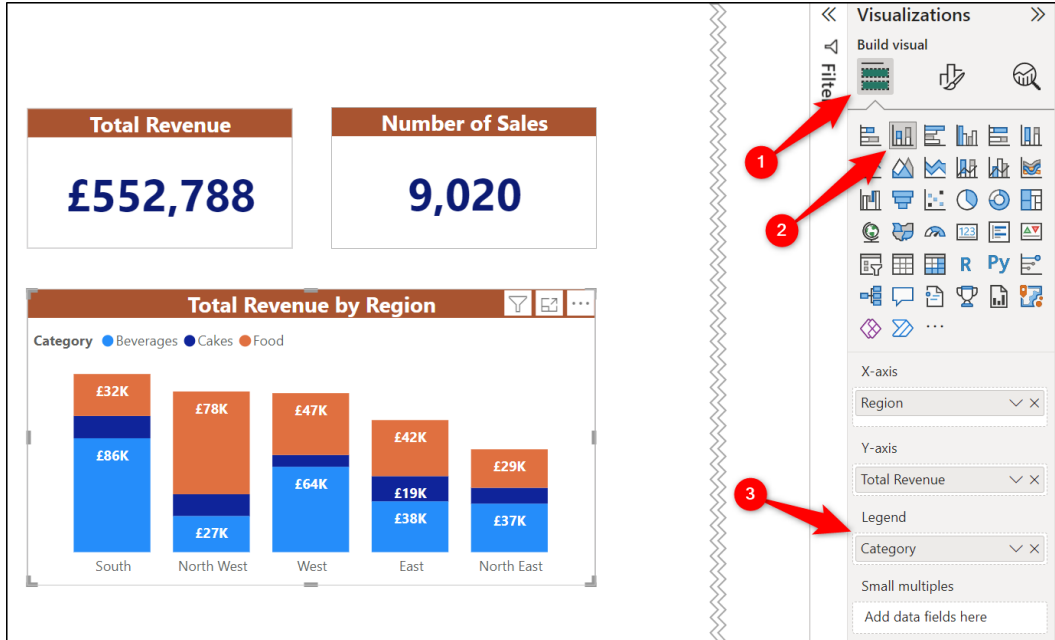


Figure 11.9: Switching to a stacked column chart

To add the total labels to a stacked column chart:

1. Click the **Format your visual** button to switch to the formatting options.
2. Click the *On/Off* slider for the **Data labels** to turn them **Off**.
3. Click the *On/Off* slider for the **Total labels** to turn them **On**, as shown in figure 11.10:

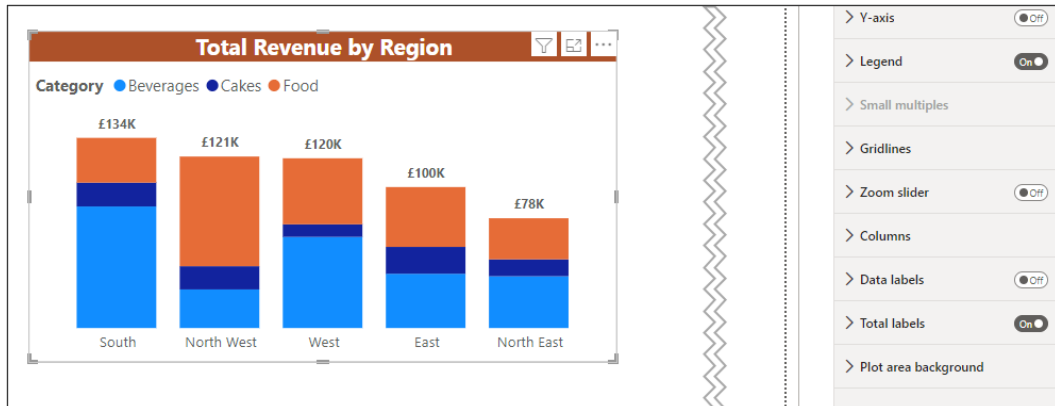


Figure 11.10: Adding total labels to a stacked column chart

Stacked column charts provide the added ability to compare the parts of a whole, even if data labels are removed. In *figure 11.10*, we can comfortably see that beverages made a large contribution to the revenue of the South, whereas for the Northwest, food sales made the most significant contribution.

Switching to a bar chart

Finally, let us convert the stacked column chart into a stacked bar chart. Bar charts can offer a better view of the data sometimes compared to column charts (*figure 11.11*). They are especially useful when the text labels in the category axis (y-axis of a bar chart) have many characters.

1. With the stacked column chart selected, click the **Add data to your visual** button to see the gallery of visualizations.
2. Click the **Stacked bar chart** icon to convert the existing stacked column chart into a stacked bar chart, as shown in *figure 11.11*:

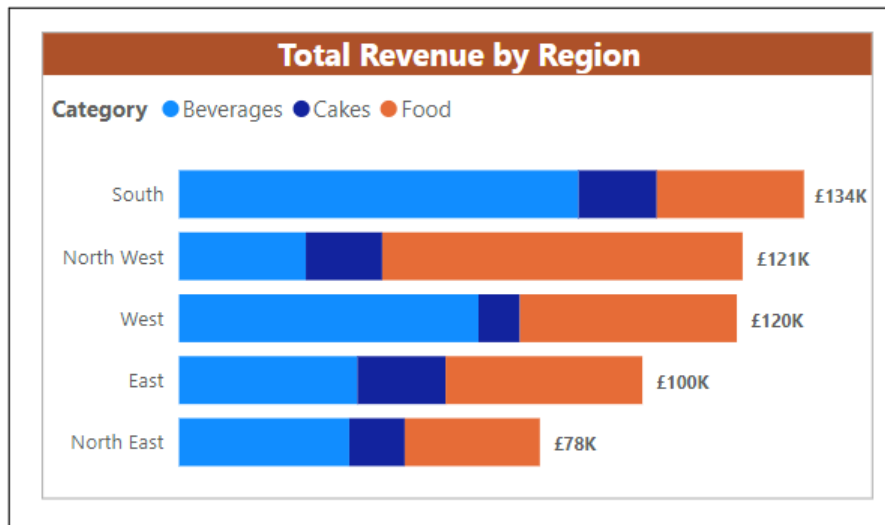


Figure 11.11: Stacked bar chart

Line and area charts

Line and area charts are used to show how a value changes over time. Area charts have advantages over line charts when comparing change among multiple data sets or over a long period of time.

Line chart

Let us create a line chart to show the change in total revenue over the 12 months of data we have in the model (*figure 11.12*).

1. Click on the **Line chart** icon in the **Visualizations** pane.
2. Click and drag the **Month Name** field from the **Calendar** table into the *X-axis* well.
3. Click and drag the **Total Revenue** measure into the *Y-axis* well.
4. Click and drag the **Monthly Revenue Difference** and **% Monthly Revenue Difference** measures into the *Tooltips* well, as shown in *figure 11.12*:

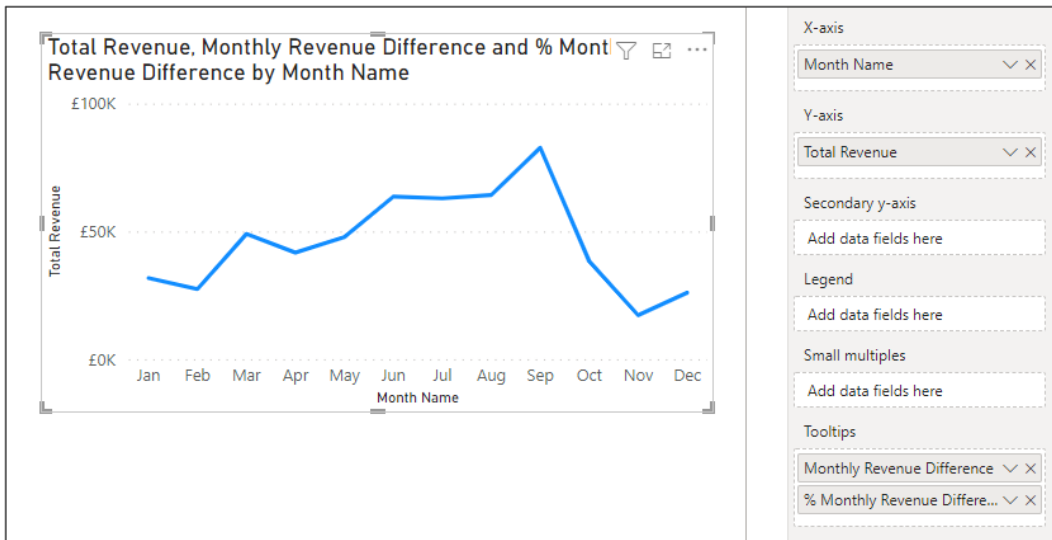


Figure 11.12: Line chart showing total revenue for 12 months

The default line chart shares similar formatting properties to that of the column chart.

- The chart title mentions all fields used by the chart, including the tooltip fields.
- Both axes display their titles.
- The default line color is blue.

Let us edit the chart title text to display “**Total Revenue over 12 Months**”, format the title to match how the title of the cards and stacked bar chart are formatted, and add a light grey visual border (*figure 11.13*).

The steps for these tasks will not be repeated here, as we have performed these tasks a few times now, both in the previous chapter and this one.

Figure 11.13 shows the improved chart title and border. The mouse arrow is positioned over the *Aug* data point to display the tooltip, including the **Monthly Revenue Difference** and **% Monthly Revenue Difference** measures. This provides additional insight into the revenue performance compared to the previous month:

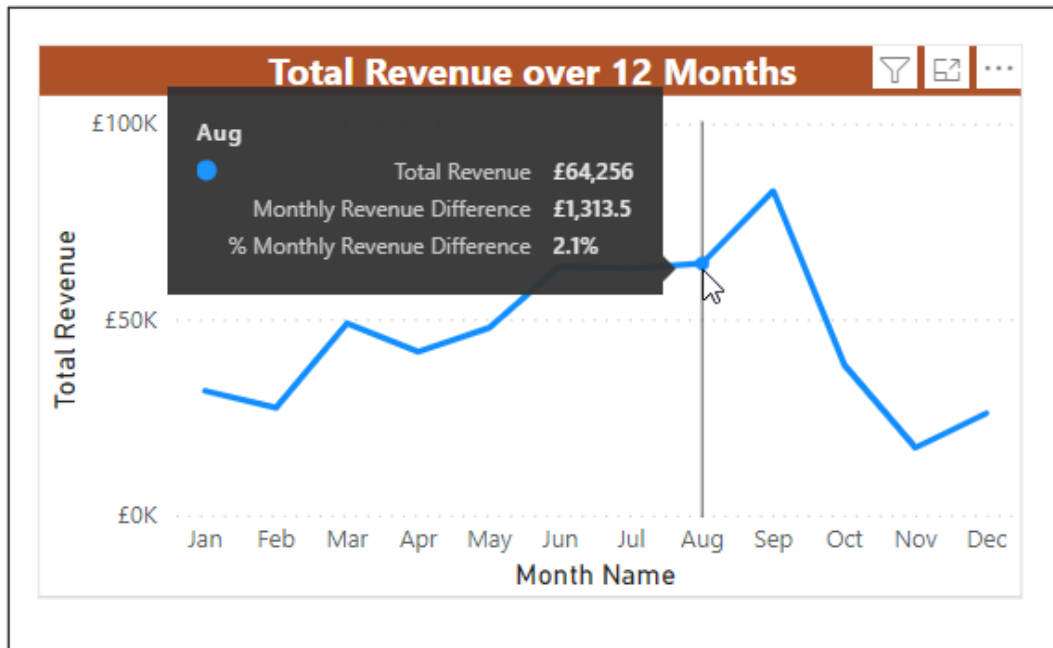


Figure 11.13: Tooltip showing value and percentage variance to the previous month

Let us now remove the axis titles and change the color of the line (figure 11.14).

1. From the **Format your visual** area of the **Visualizations** pane, click on the **Visual** category.
2. Click **X-axis** to expand the options and click the *On/Off* slider for the **Title** to turn it **Off**.
3. Click **Y-axis** to expand the options and click the *On/Off* slider for the **Title** to turn it **Off**.
4. Click **Lines** to expand its formatting options, click the **Total Revenue** list in the **Colors** section and choose a dark blue color (this color matches the callout value text color in the card visuals).

Figure 11.14 shows the completed line chart. Unfortunately, at the time of writing this book, there is no option to change the scale of the y-axis. A smaller increment than 50K would look better in this chart. Please refer to the following figure:

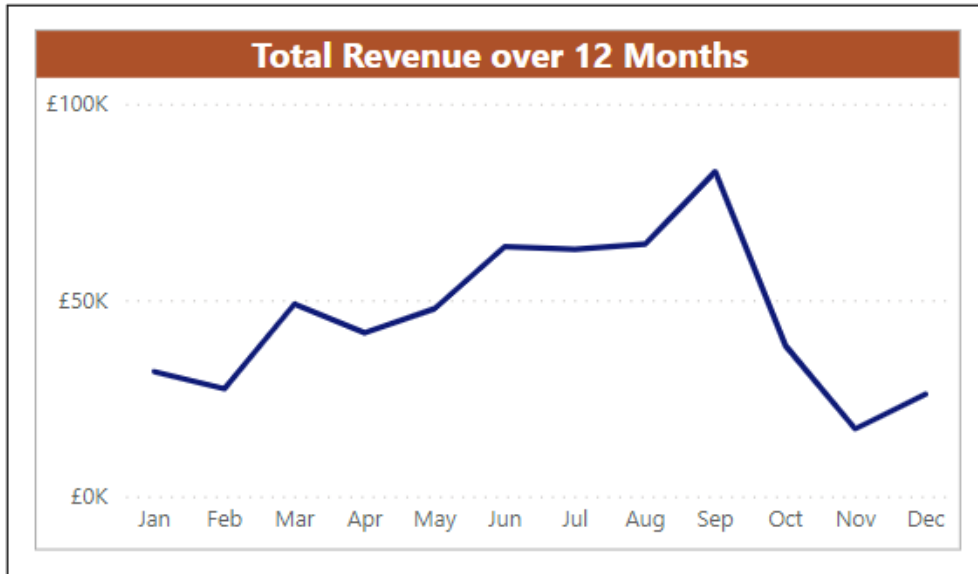


Figure 11.14: Line chart with axis titles removed and line color changed

Line chart with multiple data series

Line charts can handle multiple data series, although no more than 5 is generally recommended, or the chart can become cluttered and confusing.

For this example, we will add the **Category** column from the **Products** table to the legend as we did with the clustered column chart before. There are only three product categories (beverages, cakes, and food), so this should work quite well.

1. With the line chart selected, click the **Add data to your visual** button to see the gallery of visualizations and the field wells.
2. Click and drag the **Category** field from the **Products** table into the *Legend* well (figure 11.15).

With the values of the **Beverages** and **Food** series following a similar pattern, you can see how this chart would be confusing if we had more data series like these.

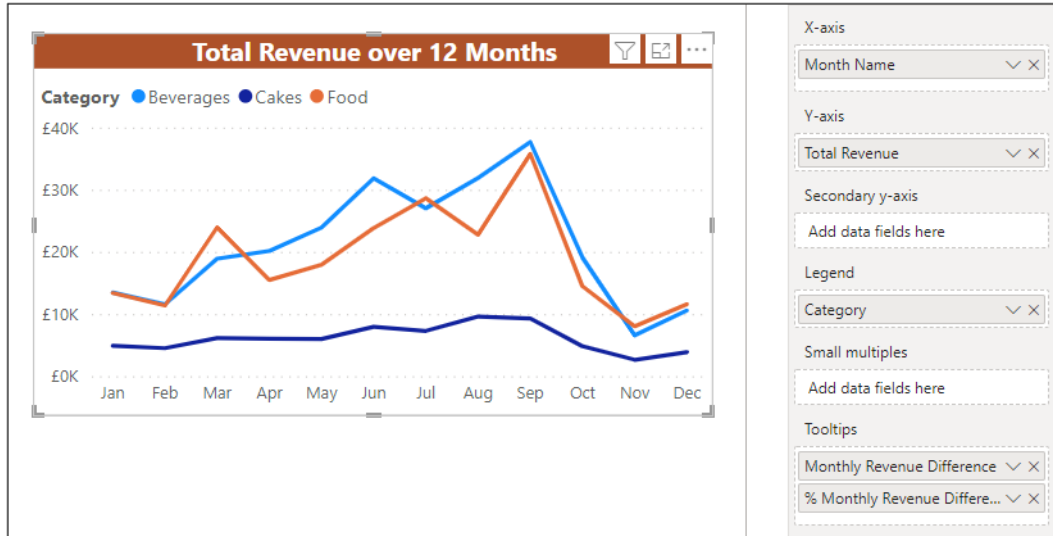


Figure 11.15: Line chart with multiple data series

A useful formatting option when handling multiple data series can be to add series labels instead of the legend. This offers an alternative way of distinguishing which line belongs to which data series.

Series labels can be positioned to the left (start) or to the right (end) of the lines.

In this example, because the start values and finish values of the *Beverages* and *Food* series are so close, only one label is shown in figure 11.16. This renders the option unhelpful and redundant. So, we will keep with the legend.

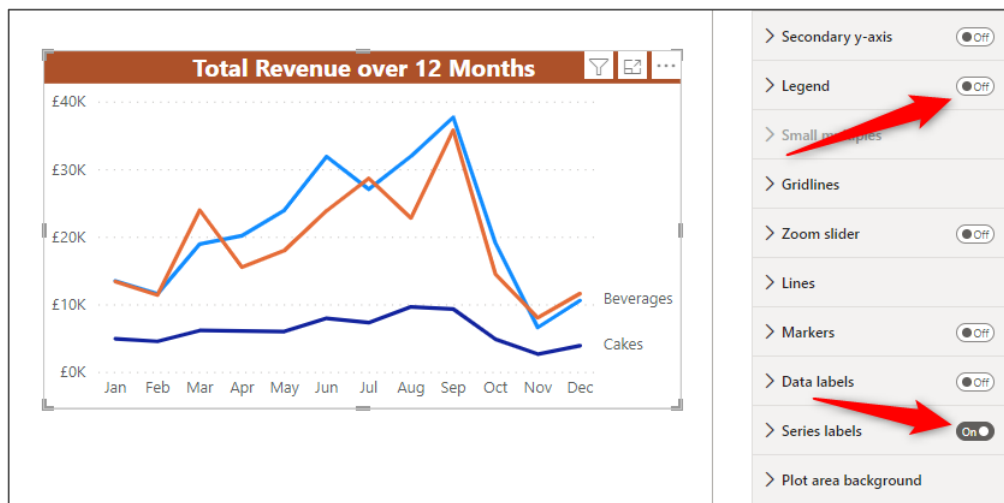


Figure 11.16: Swapping the legend for series labels

Stacked area chart

For the final line and area chart example, we will see a stacked area chart. This will show the contribution of each product category to the total revenue over the 12 months (*figure 11.17*).

This provides better clarity than the previous line chart, where the data series were often overlapping. It also returns us to see the total revenue over the 12 months in addition to category revenue, whereas the line chart with multiple series was focusing on the revenue of each category only.

With the line chart selected, click the **Stacked area chart** icon to convert the existing line chart into a stacked area chart, as shown in *figure 11.17*.

For this report, we will switch the chart back to a line chart and remove the **Category** column from the *Legend* well so that it uses a single total revenue data series.

It is good to explore and to be comfortable with the different chart types and their specific strengths and weaknesses.

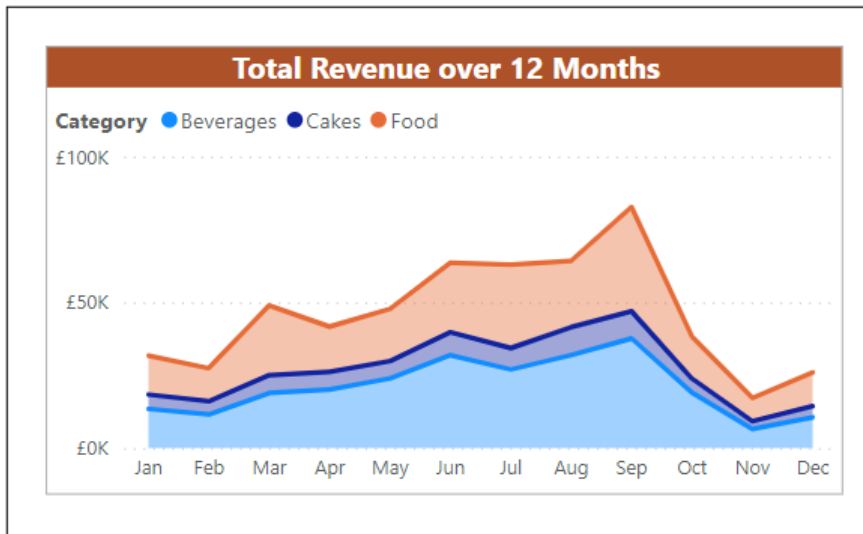


Figure 11.17: Stacked area chart

Combo charts

A combo chart, or combination chart, is a hybrid of two chart types, such as column and line or column and area.

Combo charts are used to display different types of information on the same chart. This could be to show actual values against budget values or two types of data that vary widely, such as sales volume and percentage of upsells.

In Power BI, the only combo charts available as standard are the line and stacked column chart or the line and clustered column chart.

In this example, we will add a line and clustered column chart to show the total revenue and count of sales for 12 months on a single chart (figure 11.18).

1. Click on the **Line and clustered column chart** icon in the **Visualizations** pane.
2. Click and drag the **Month Name** field from the **Calendar** table into the *X-axis* well.
3. Click and drag the **Total Revenue** measure into the *Column y-axis* well.
4. Click and drag the **Count of Sales** measure into the *Line y-axis* well as shown in figure 11.18:

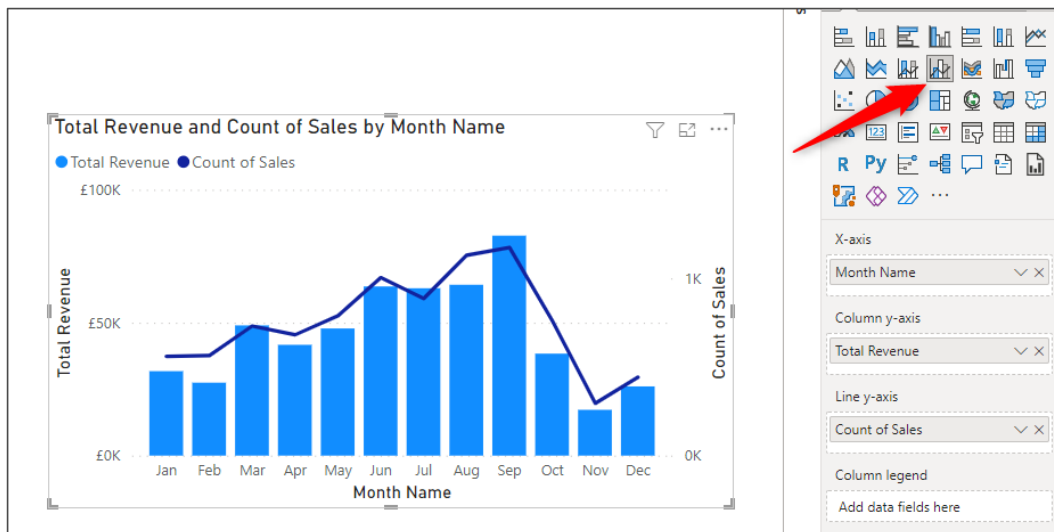


Figure 11.18: Combo chart showing total revenue and count of sales for 12 months

The **Total Revenue** and **Count of Sales** values are plotted on different axis as the values are very different. The primary axis on the left is used for the total revenue, and the secondary axis on the right is for the count of sales.

The formatting options available closely resemble those of the column, bar, and line charts discussed before, so we will not dive into those right now.

In *figure 11.19*, the chart title and border have been modified, and the x-axis title has been removed from the chart. The title for each of the y-axis has remained as it is useful to define them in this example.

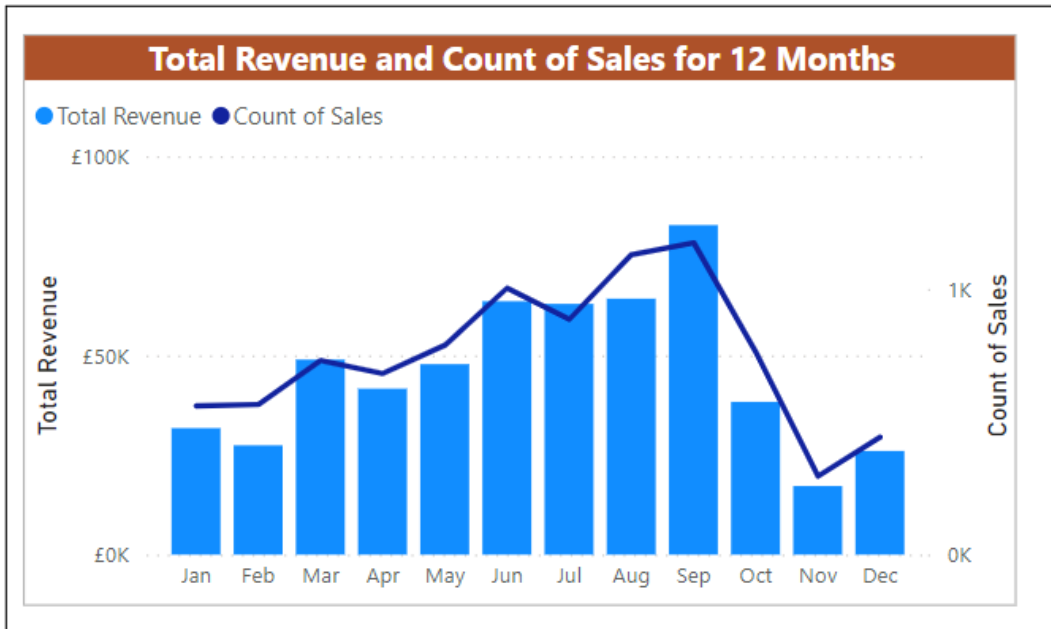


Figure 11.19: Improved line and clustered column combo chart

Let us see another example of a line and clustered column combo chart where the values share the same axis.

For this example, we will create a target value field and add it to the chart for the line y-axis. This will enable us to visualize the total revenue against the target. We will use a target value of 45,000 for this example.

Currently, we do not have a column with target values in our model, so we will create a new table that contains just a single column and a single value.

1. Click **Home** | **Enter data**
2. In the *Create Table* window (*figure 11.20*), type **Monthly Target** in the *Name* box for the name of the table.
3. Double-click on the column header and type “**Target**”.
4. Click in the first cell of the table and type “**45000**” for the performance target for all months.
5. Click **Load**.

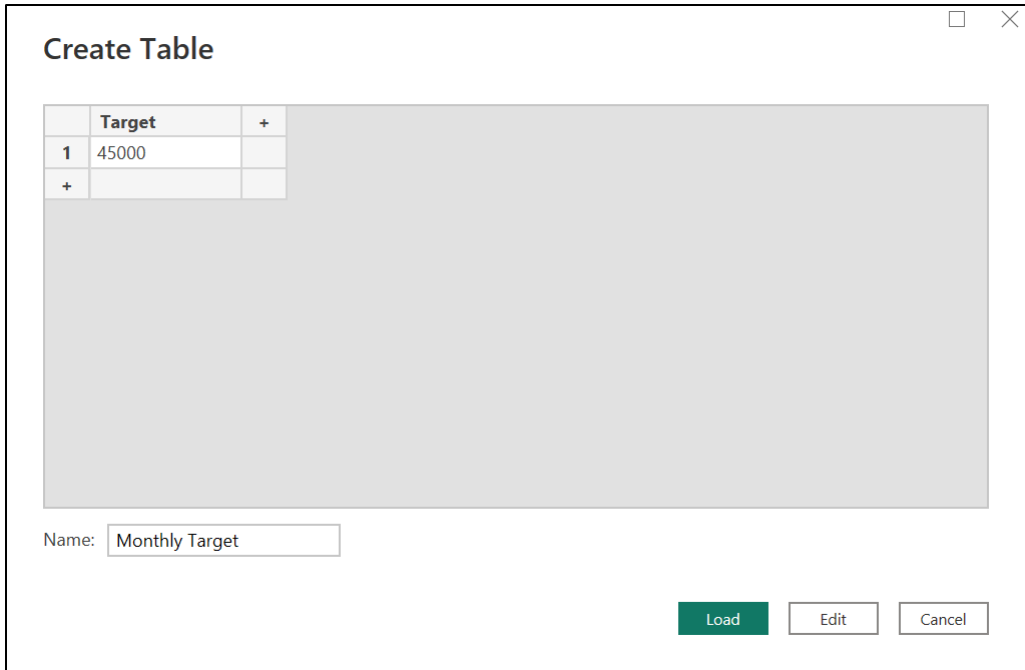


Figure 11.20: Creating a table for the target value

Using the same line and clustered column chart, we will change it to use the target value in the line y-axis instead of the count of sales.

1. Click on the line and clustered column chart to select it.
2. Click the X beside the **Count of Sales** measure in the *Line y-axis* well to remove it.
3. Click and drag the **Target** column from the **Monthly Target** table into the *Line y-axis* well.

Figure 11.21 shows the line and clustered column chart using the target value. The title of the chart has been updated to reflect the role of the chart, and the y-axis-title removed.

Notice that the total revenue and target values now share the same primary axis as the values are of the same type:

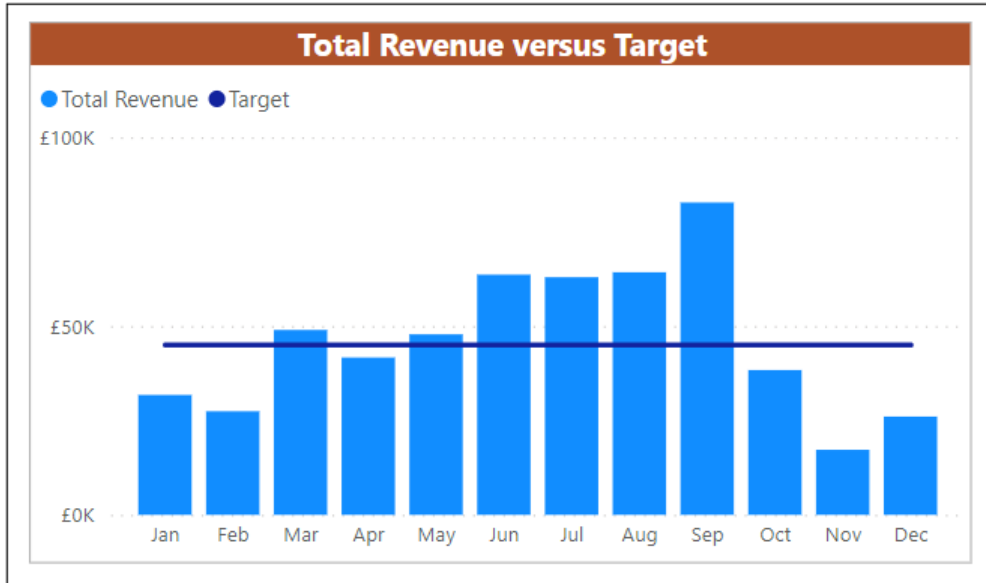


Figure 11.21: Total revenue versus target combo chart

Note: Line charts can also be used for these actuals v budget or actuals v KPI type charts. One line can represent the actuals, and the other line for the budget/target/KPI.

Pie and donut charts

Pie and donut charts are used to present parts-to-whole relationships. Power BI offers one type of pie and one type of donut to choose from.

In this example, we will insert a pie chart to present the split of total revenue during the week and at the weekend.

1. Click on the Pie chart icon in the **Visualizations** pane (figure 11.22).
2. Click and drag the **Is Weekend** field from the **Calendar** table into the *Legend* well.
3. Click and drag the **Total Revenue** measure into the *Values* well.
4. Click and drag the **Count of Sales** and **Total Units Sold** measures into the *Tooltips* well as shown in figure 11.22:

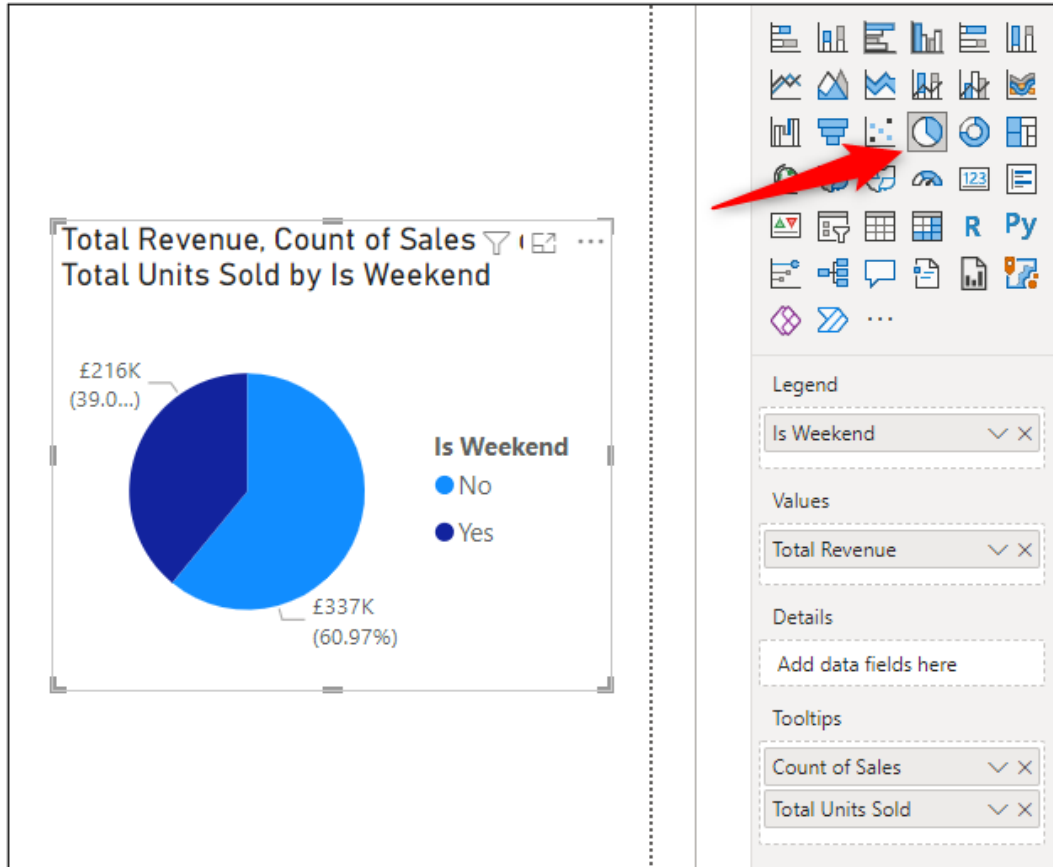


Figure 11.22: Pie chart showing total revenue split by the weekend and during the week

The usual improvements to the title and border will be made to the pie chart, as previously covered with the other chart types.

In addition to these changes, we will also:

- remove the legend
- change the color of the light blue slice to orange
- show the category (Yes or No) in the data label in addition to the total value and percentage.

To make these changes, follow the following steps:

1. Click on the **Visual** category of the formatting options.

2. Click on the *On/Off* slider for the **Legend** to turn it **Off**.
3. Click **Slices** to expand the formatting options, click the *No* list, and select an orange color.
4. Click **Detail labels** to expand the options; click the **Label contents** list within the **Options** category and select **All detail labels** (figure 11.23).
5. Click the **Values** sub-category of **Detail labels** to see the options, click the **Bold** button and type **0** in the *Percentage decimal places* box.

Figure 11.23 shows the completed pie chart. Removing the legend freed up some space, and with only two pie slices, there was plenty of room for detail labels:

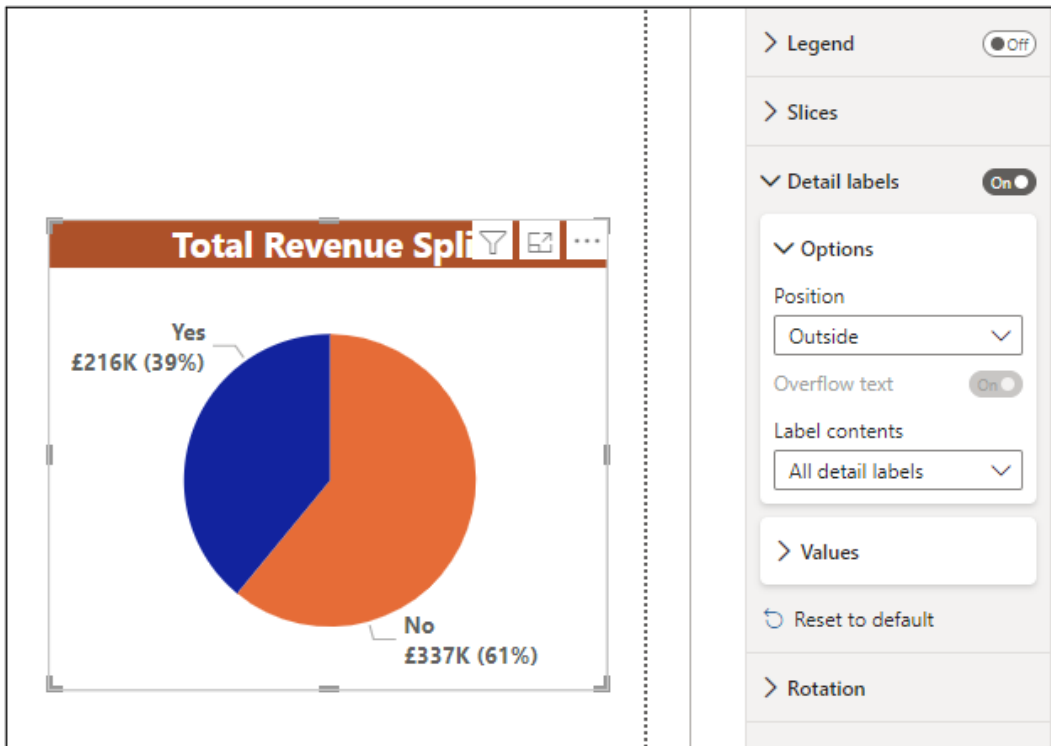


Figure 11.23: Pie chart showing all label details and with legend removed

This pie chart can easily be converted to a donut chart if preferred. Simply click the **Donut chart** icon in the **Visualizations** pane while the pie chart is selected, as shown in figure 11.24:

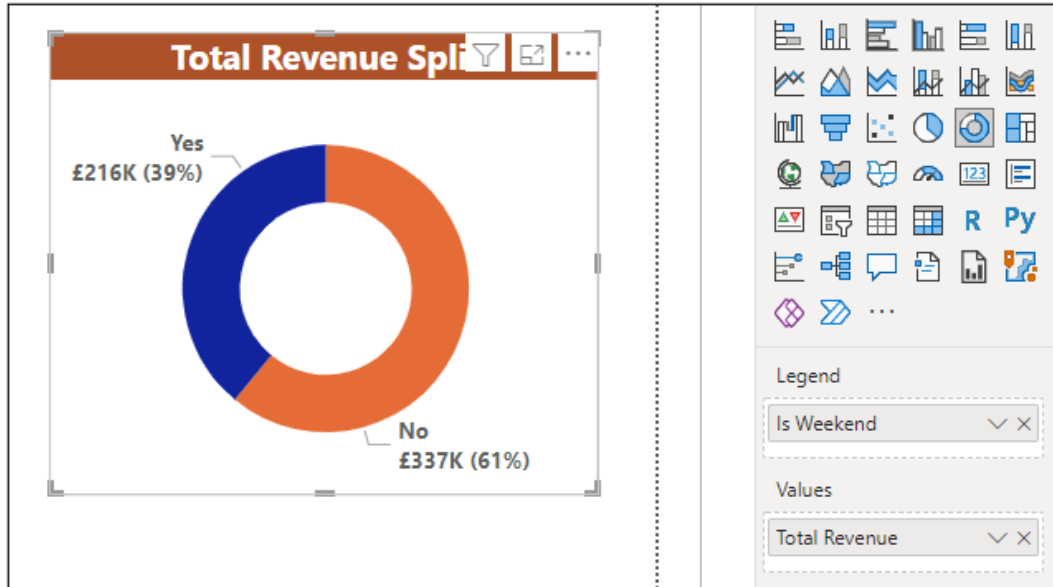


Figure 11.24: Donut chart showing total revenue split by the weekend and during the week

Gauge charts

Gauge charts are used to measure progress toward a goal. They also cater for another key performance indicator or target value to be included and measured against.

Progress toward the main goal is shown by a shaded bar moving toward the end of the arc. The indicator or target value is represented by a line on the gauge.

Let us create a gauge chart on the **Monthly Sales Analysis** report page to measure the progress of the total sales revenue for a given month and use the previous month's revenue as the target value.

1. Click the **Monthly Sales Analysis** page.
2. Click the Gauge icon in the **Visualizations** pane.
3. Click and drag the **Total Revenue** measure to the *Value* well.
4. Click and drag the **Revenue Previous Month** measure to the *Target value* well.
5. Click and drag the **Count of Sales** measure to the *Tooltips* well.

Figure 11.25 shows the gauge that is created from these fields. The values are spread evenly across the gauge, so the minimum value of the arc is 0, and the maximum value is automatically set as 98K because the revenue is 49K.

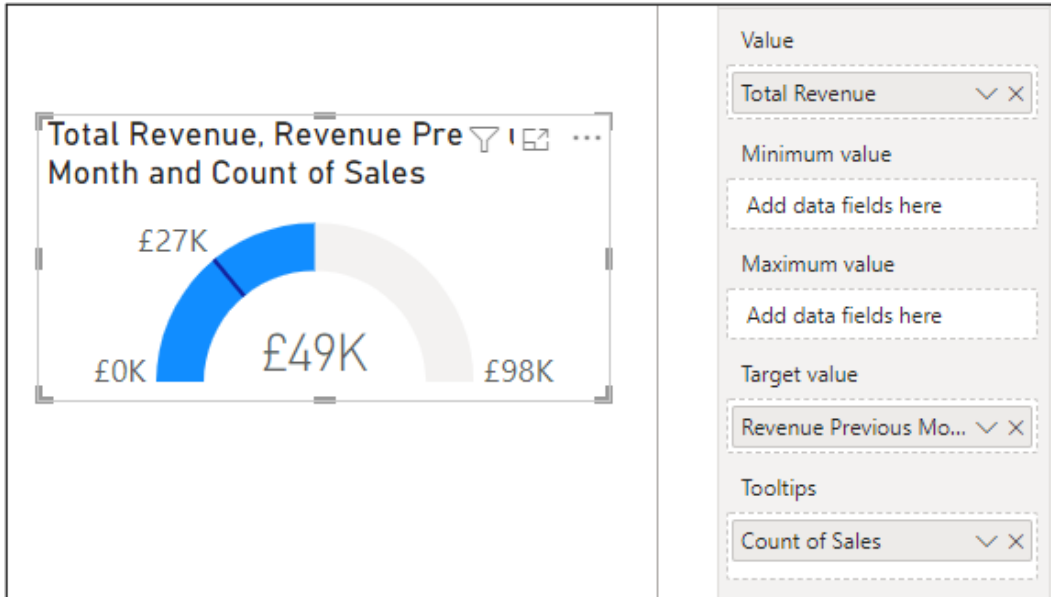


Figure 11.25: Gauge chart to measure revenue progress for a selected month

Remember, the visuals on the **Monthly Sales Analysis** page are currently all controlled by the list of month names Slicer visual. In *figure 11.26*, you can see that the month of March is currently selected:

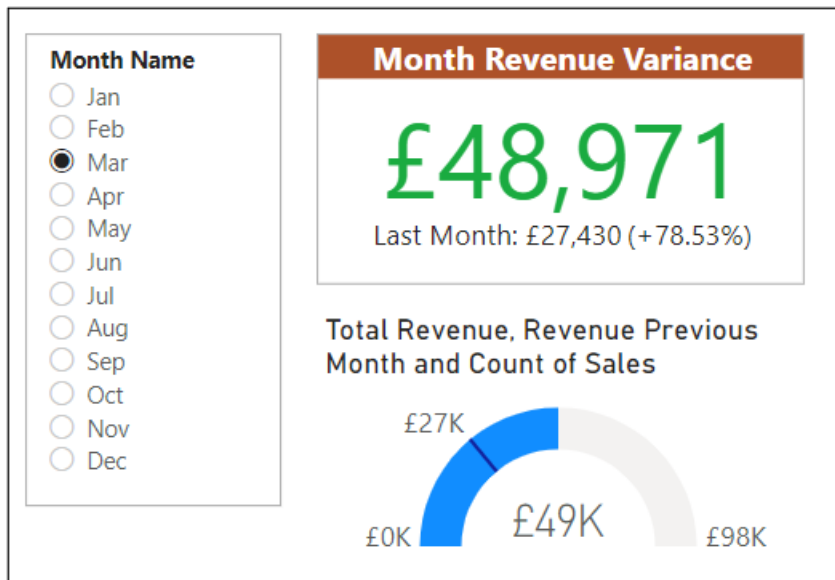


Figure 11.26: Month revenue specified by the selection in a slicer

We will now walk through multiple formatting changes to modify the gauge chart to our requirements. These will include changing the maximum value of the arc, removing the target label, and changing the color of the gauge bar shading.

Note: There are many variations of gauge charts available from the Power BI visuals marketplace. We will discuss this marketplace and how to access these visuals in *Chapter 13, Other Power BI Visualizations*.

General formatting

Let us now discuss the multiple formatting changes:

1. With the gauge chart selected, click the **Format your visual** button above the *Visualizations* gallery.
2. Click the **General** category of options.
3. Click **Effects**, then the *Visual border* sub-category to expand its options.
4. Click the *On/Off* slider for the *Visual border* to turn it **On**, and then click the **Color** list and set a light grey color.
5. Click **Effects** to collapse its formatting options and click **Title** to expand theirs.
6. Type **Revenue Progress** in the *Text* box.
7. Specify the *Font* as **Segoe UI** and click the **Bold** button.
8. Click **Center** for the *Horizontal alignment*.

Visual formatting

We will now go through the steps to do visual formatting:

1. Click the **Visual** category of options.
2. Click **Gauge axis** to expand the options and type **100000** in the *Max* box (*figure 11.27*).

You can use measures to specify the minimum and maximum values of the gauge axis. That offers a more dynamic approach than the one used in this simple example.

3. Click **Colors** and set an orange color for the *Fill color*.
4. Click **Data labels** and reduce the *Font* size to **11**.
5. Click the *On/Off* slider for the *Target label* to set it to **Off**.
6. Click **Callout value**, change the *Font* to **Segoe UI**, the *Color* to black, and the **Display units** to **None**.

Figure 11.27 shows the finished gauge chart. The formatting applied in these examples is a personal preference and depends on the information we are trying to convey.

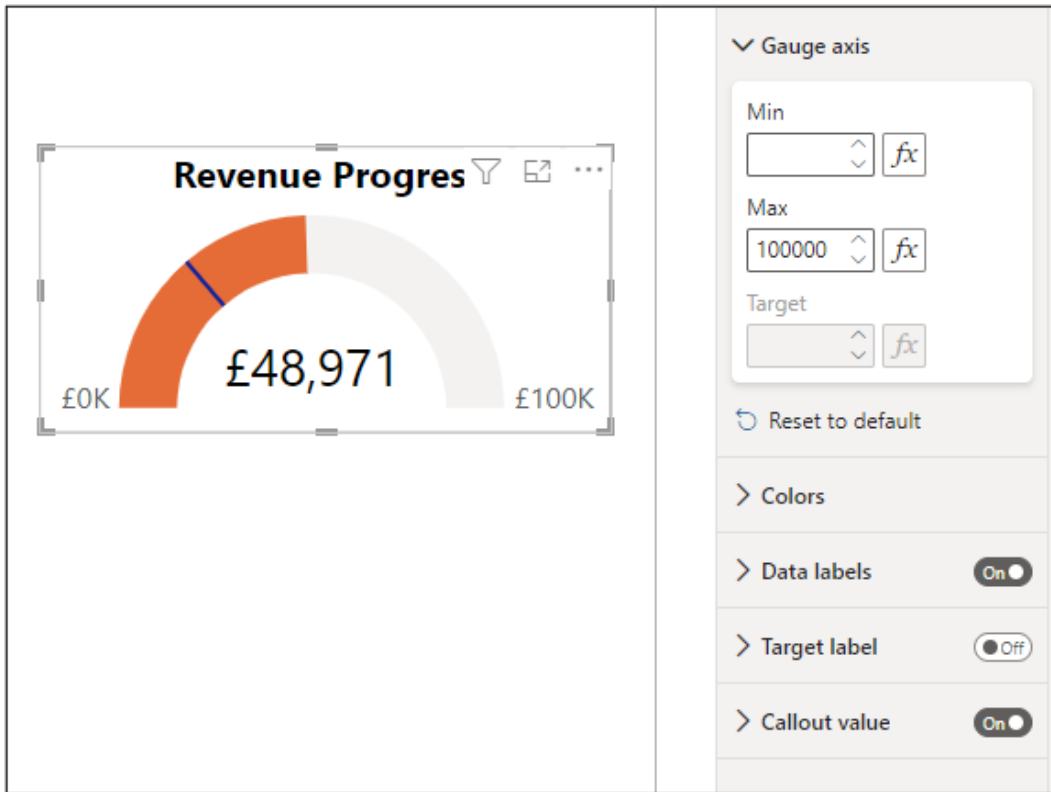


Figure 11.27: Setting the max value of the gauge chart axis

KPI versus gauge charts

Figure 11.28 shows the KPI visual and gauge chart side-by-side on the **Monthly Sales Analysis** page. It is not insinuated that it is valuable to use both visuals; they are just presented together to see the strengths each visual has over the other.

In this example, the gauge chart has been used as an alternative to the KPI visual. Like the KPI, the gauge chart visualizes revenue for the selected month against the previous months' revenue.

The KPI visual presents the percentage variance, which the gauge does not; however, the **% Monthly Revenue Difference** measure could have been used as a tooltip in the gauge for this insight.

In addition to seeing performance against last month's revenue, the gauge chart also presents the progress toward another value. In this example, the arbitrary value of 100,000 was used, but this value can be the result of a measure providing flexibility to its use, refer to the following figure.

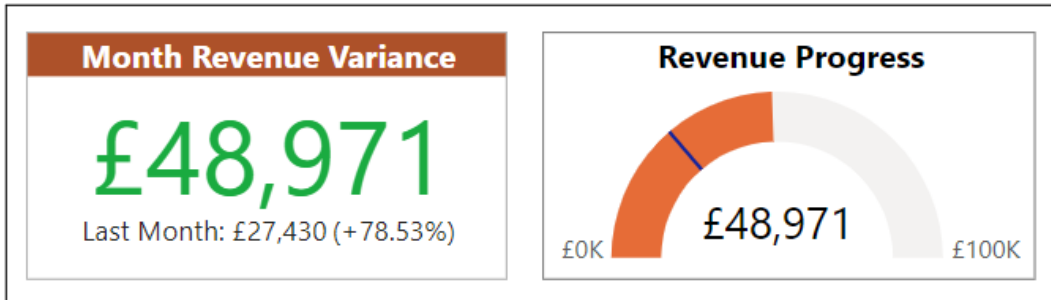


Figure 11.28: KPI or gauge chart

Of course, the **Target** value that was created earlier for the combo chart example could be used as the target value in the gauge chart.

Click and drag the **Target** column from the **Monthly Target** table into the *Target value* well. It will replace the **Revenue Previous Month** measure currently in the well, as shown in figure 11.29:

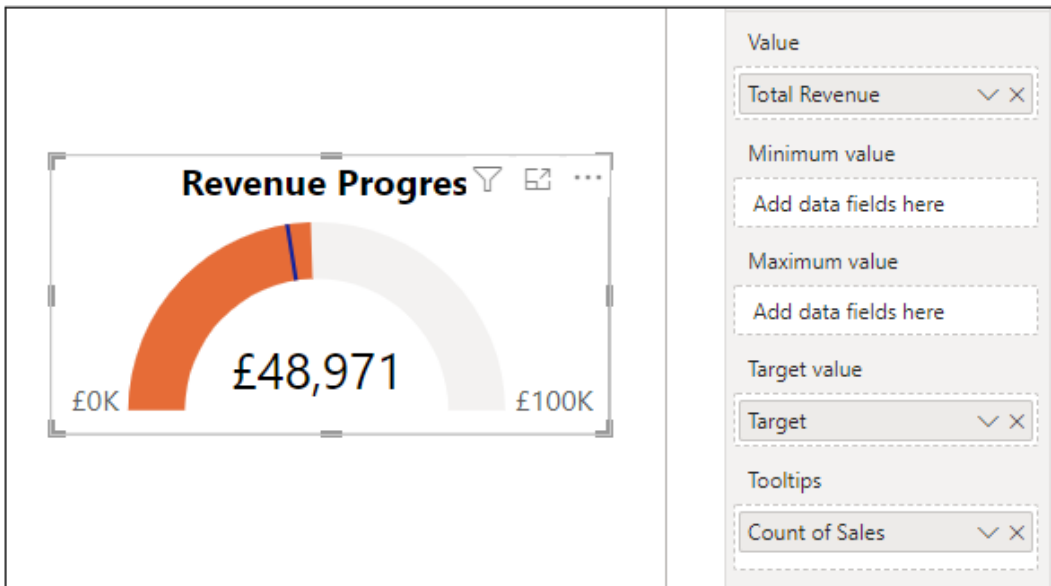


Figure 11.29: Gauge chart with target value

Treemap visual

The Treemap visual displays hierarchical data as a set of nested rectangles. The size of each rectangle is determined by the size of the value being measured. The rectangles are ordered from largest to smallest.

Treemaps can be used when you are working with many data points and offer an alternative to the column or bar chart that may struggle to present a large volume of data clearly.

They also present the distribution of the values across different categories showing part-to-whole relationships.

In this example, we will create a treemap to show the distribution of revenue across the different regions.

1. From the **Monthly Sales Analysis** page, click the **Treemap** icon in the **Visualizations** pane.
2. Click and drag the **Region** column from the **Stores** table to the **Category** well.
3. Click and drag the **Total Revenue** measure to the **Values** well.
4. Click and drag the **Count of Sales** measure to the **Tooltips** well.

Treemap depicts the distribution of revenue by region, as shown in *figure 11.30*:

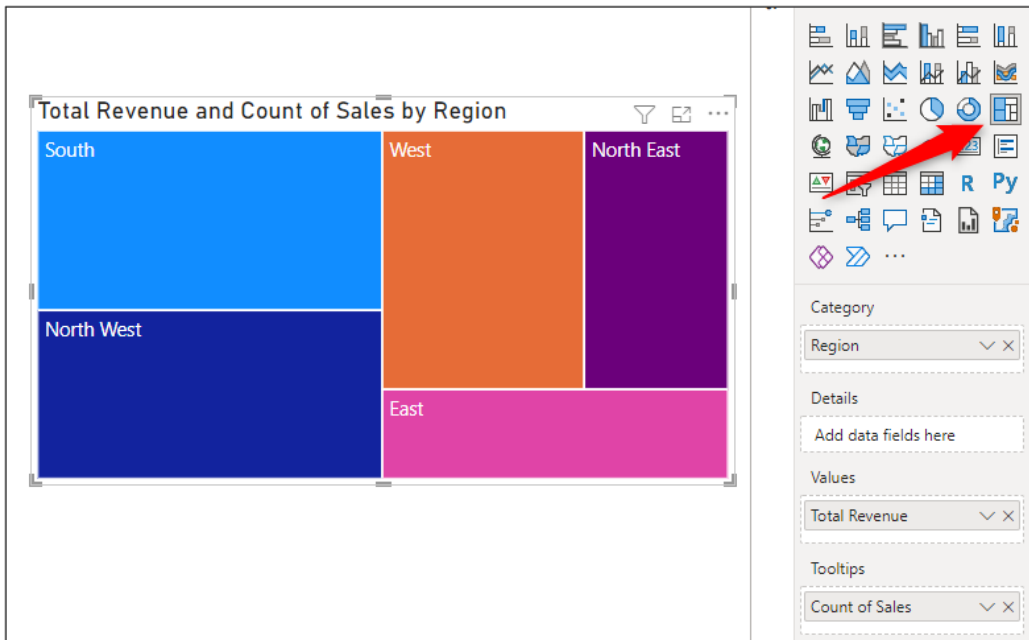


Figure 11.30: Treemap showing the distribution of revenue by region

The treemap values are being filtered by the month name slicer, so currently, we can see the results for March only, as can be seen in *figure 11.31*:

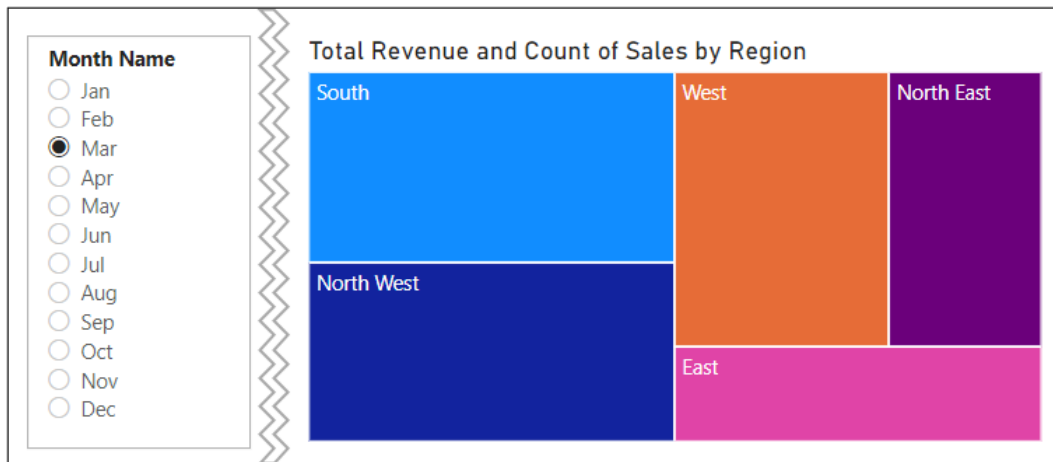


Figure 11.31: Treemap filtered by the month name slicer

Let us now add an additional field to the treemap to breakdown each region's revenue performance by product category.

Click and drag the **Category** column from the **Products** table into the *Details* well, as shown in *figure 11.32*. Each region (branch) is now split by the different product categories (leaves)

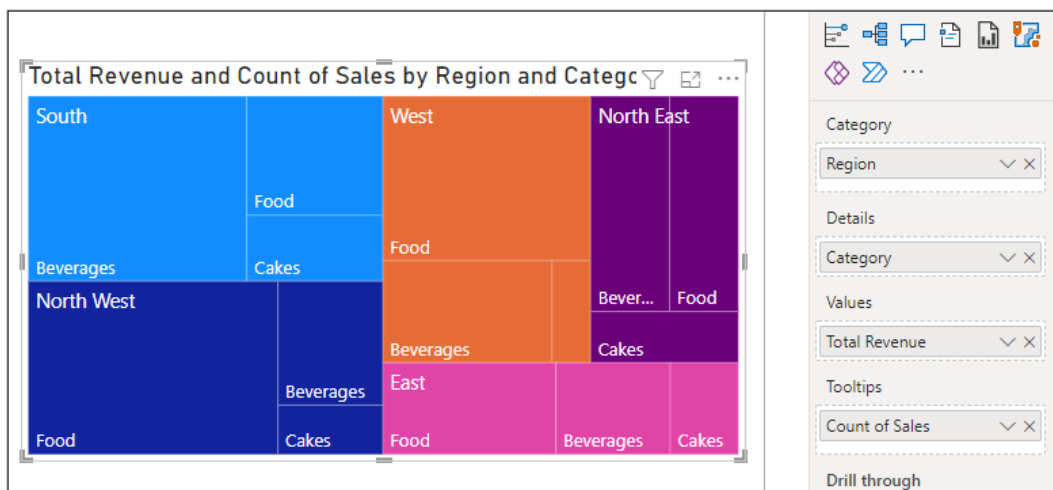


Figure 11.32: Adding details to the treemap

The formatting options for treemap visuals are limited to changing the colors of the branches, using a legend, and using category and data labels, as shown in *figure 11.33*:

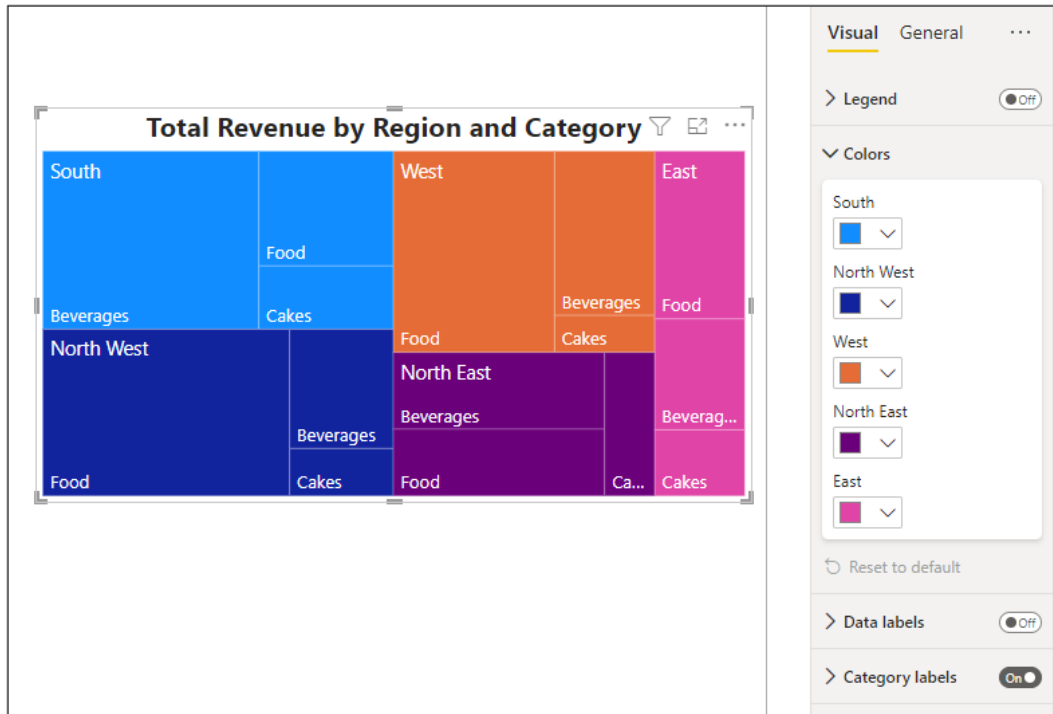


Figure 11.33: Limited formatting options for treemap visuals

The title of the treemap has been edited in *figure 11.33*, but no other changes have been made. There are no options to change the sort order or add percentage labels in treemaps.

Conclusion

In this chapter, we learnt some of the most important chart visuals in Power BI, including the column, line, combo, and donut.

We covered the purpose of each chart type and the specific strengths one may have over another. We modified key attributes and formatting options to increase the effectiveness of our charts.

In the upcoming chapter, we will look at the different map charts provided with Power BI.

Questions

Here are some questions to test what you have learnt in this chapter.

1. What is the primary purpose of a line chart?
 - a. Show the change in a value over time
 - b. Show the contribution of parts to a whole
 - c. Compare value across different categories
 - d. None of the above.
2. You can set the max value for the axis of a Gauge chart.
 - a. True
 - b. False
3. Which of the following charts provide the tooltips feature?
 - a. Pie chart
 - b. Gauge chart
 - c. Column chart
 - d. All of the above
4. Which of the following is true for treemap visuals?
 - a. You can change the color of the rectangles (branches)
 - b. You can add tooltips
 - c. You cannot change the sort order
 - d. All of the above

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



CHAPTER 12

Using Maps in Power BI Reports

Introduction

Maps provide a way to efficiently visualize geographic data such as countries, cities, and postcodes. These visuals are meaningful and beautiful.

Power BI has two main map visuals provided as standard. The Map and the filled Map. There is also a visual named the shape map, which is in preview and expected for release in the coming months.

In this chapter, we look at how to enable the use of the map visuals, insert a map into our report, and then learn a few tips to get the most out of Power BI maps.

Structure

In this chapter, we will cover the following topics:

- How to enable map visuals in Power BI.
- How to use the Map visual.
- Tips for mapping geographic data effectively.

Objectives

After reading this chapter, you will learn how to use the Map visual in Power BI and a few tips to overcome problems when plotting geographic data.

Enable map visuals in Power BI

The Map and filled Map visuals need to be enabled in Power BI before they can be used. If they are not enabled, you will see the error shown in *figure 12.1* when you attempt to create your first Map:

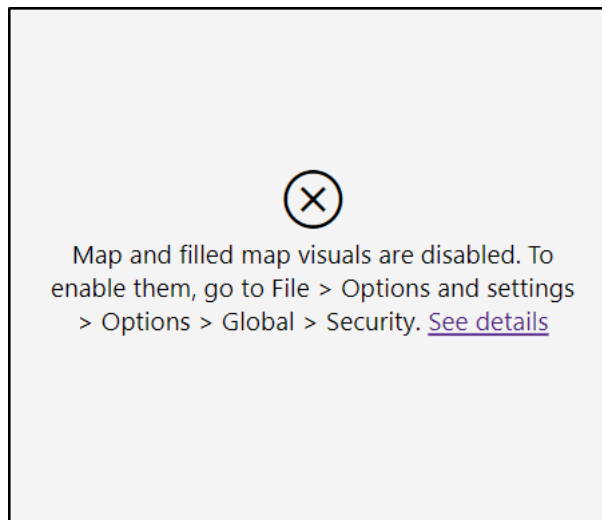


Figure 12.1: Map and filled map visuals disabled

To enable Maps follow the following steps:

- Click **File | Options and settings | Options**.
- Click the **Security** category from within the *Global* category of options.
- Click the **Use Map and Filled Map visuals** checkbox, as shown in *figure 12.2*.
- Click **OK**.

Note: We will focus on the Map visual in this chapter. The Filled Map uses different shades of color to present how values differ across regional areas.

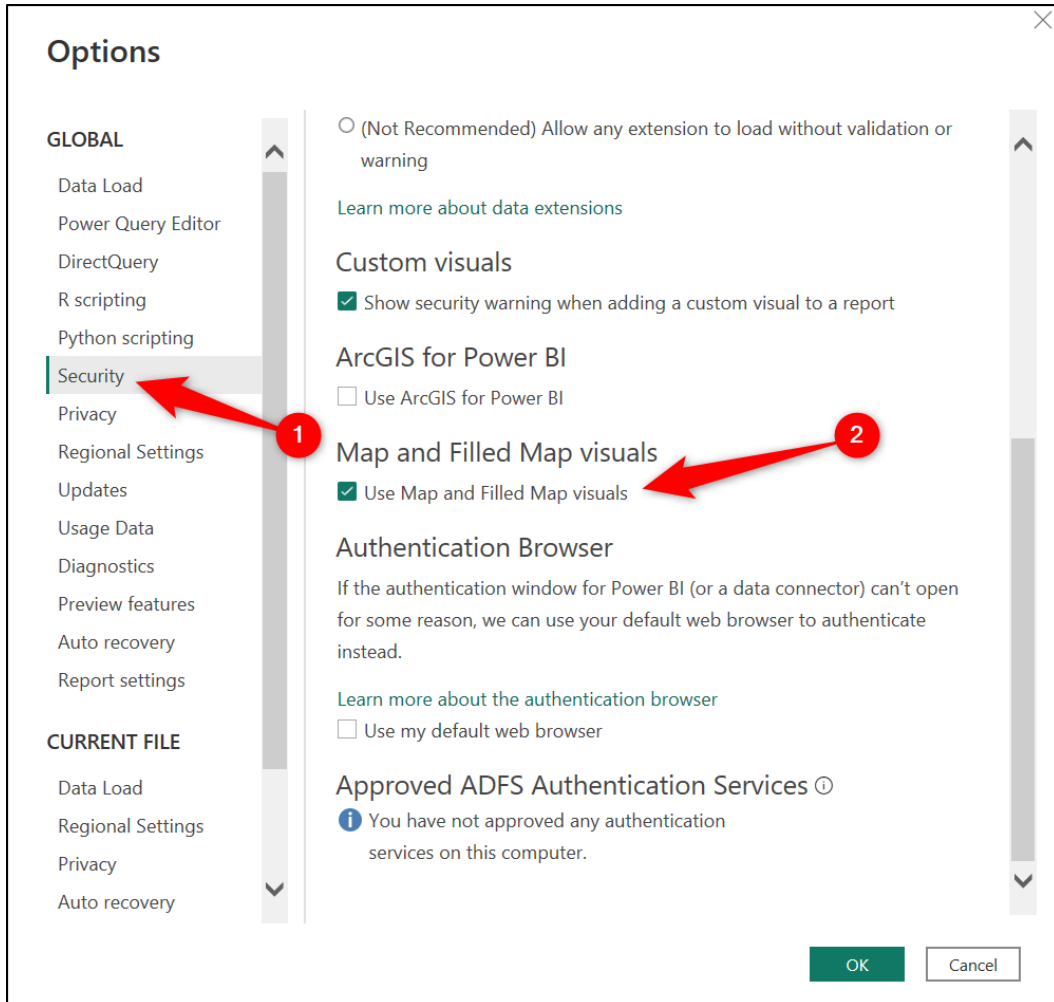


Figure 12.2: Enabling the use of map visuals in Power BI options

The Map visual

Files: **sales-report-ch-12-start.pbix** and **lat-and-long.xlsx**

The general Map visual in Power BI uses bubble markers to plot the geographic data on the Map. The size of the bubble is determined by the size of the value it is measuring.

In this example, we will use the Map visual to present the total revenue for all stores in the UK.

Let us first insert a new page named *Maps* to our report to focus on building the map chart away from the distraction of other visuals.

1. Click the **New page** button (plus icon beside the page tabs) to insert a new page named **Page 3**.
2. Double-click on **Page 3** and type the name “**Maps**” for the new page, as shown in *figure 12.3*:

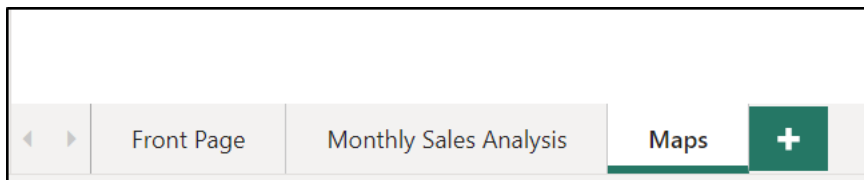


Figure 12.3: Maps page in the report

Using the Map visual

The Map visual uses the Bing Maps geocoding engine to effectively map the geographic fields (country, city, and postcode) of your data model.

This makes it very easy to create maps in Power BI. We cover some tips later in this chapter to assist the geocoding if there is any trouble plotting the geographic field correctly.

1. Click on the **Map** icon in the **Visualizations** pane.
2. Click and drag the **Postcode** field from the **Stores** table into the *Location* well, as depicted in *figure 12.4*.

The Map visual plots the locations on the Map with this single field. Each bubble represents the location of a postcode.

Note: There is actually a missing location on the Map. The Brighton store has failed to work. We will fix this later in the chapter by using the latitude and longitude of each location.

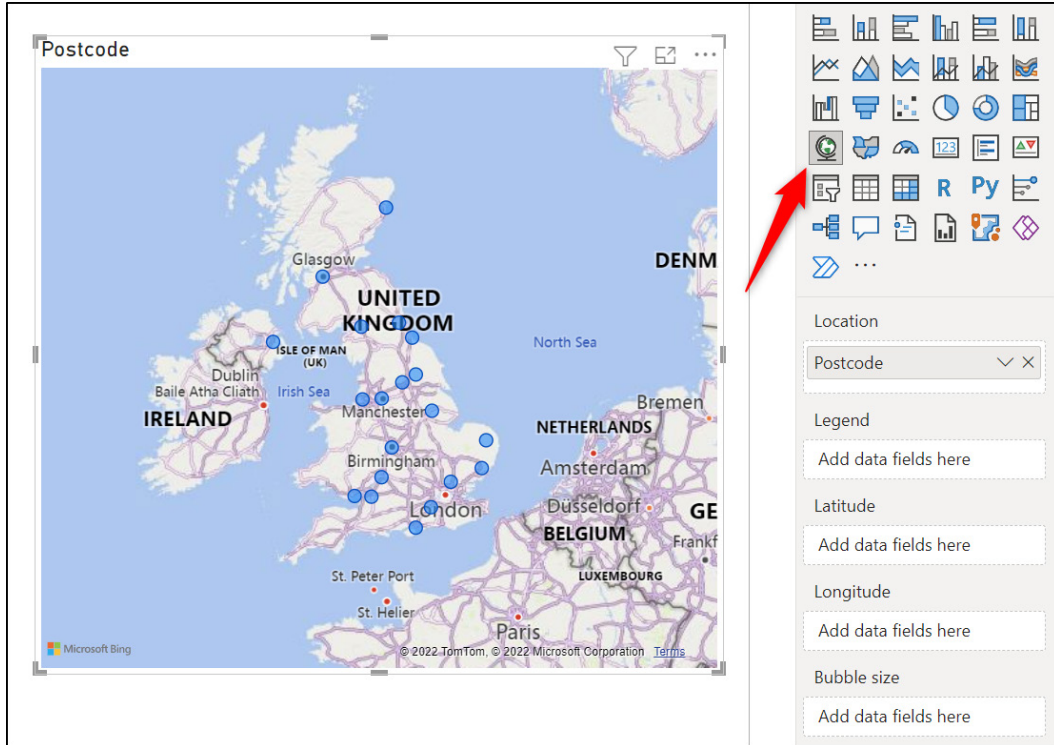


Figure 12.4: Map visual plotting UK postcode data

1. Click and drag the **Total Revenue** measure into the *Bubble size* well.
2. Click and drag the **Store** and **Manager** fields from the **Stores** table into the *Tooltips* well.

In *figure 12.5*, the size of the bubble markers is determined by the size of the total revenue value for the store at that postcode. The larger the bubble, the better the store's performance.

Figure 12.5 shows the tooltip for the Aberdeen store at postcode AB10 1AB. Using a simple mouse over, the reader can now see the total revenue, store name, and store manager name at that location.

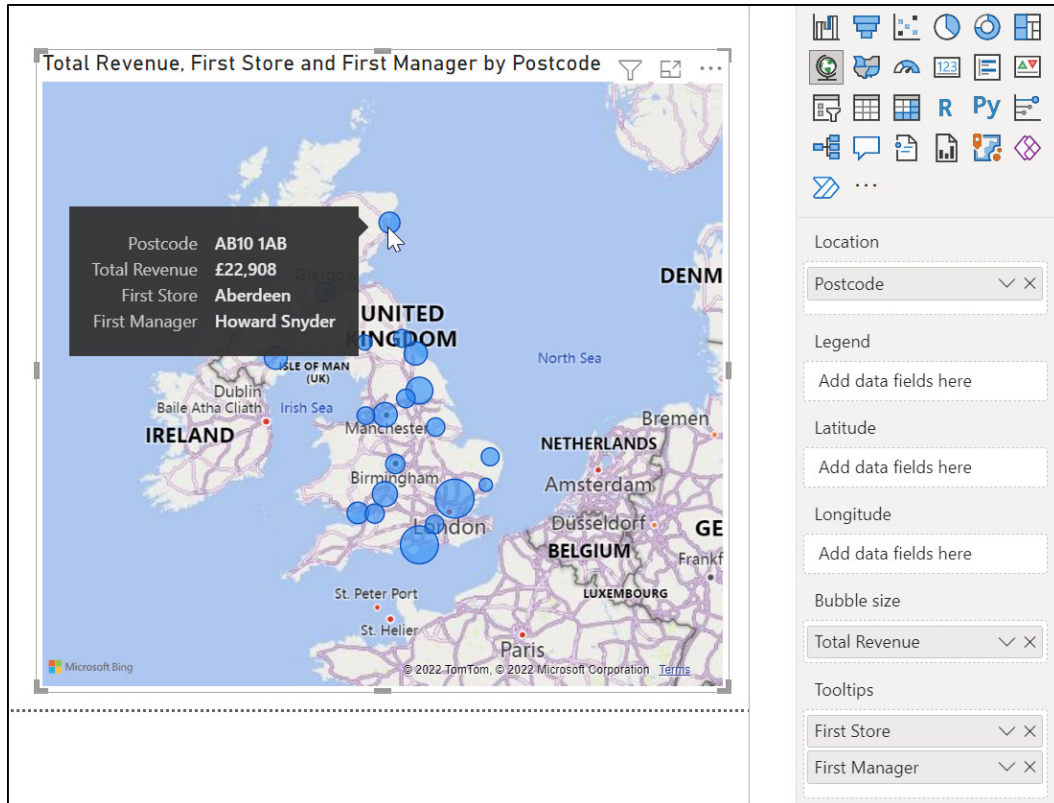


Figure 12.5: Adding fields for the Bubble size and Tooltips wells

The fields in the *Tooltips* well are displayed as *First Store* and *First Manager*. This is because they are text fields, and Power BI performs the default summarization of the first value with text fields. Other summarizations include the last value, count, and a distinct count.

There is only one store and one manager at any given postcode in this data set, so the initial names of *First Store* and *First Manager* are not useful.

Let's rename these fields to simply read **Store Name** and **Manager Name**, respectively.

1. Click the list arrow for the *First Store* field and click **Rename for this visual**, as shown in figure 12.6.

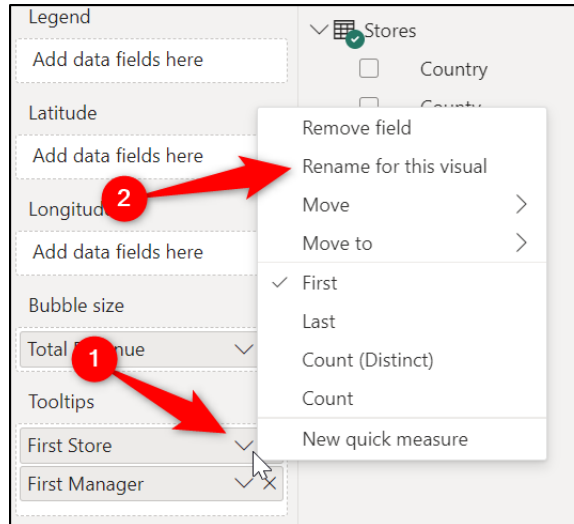


Figure 12.6: Renaming a field for a specific visual

2. Type **Store Name** as the name for the visual and press *Enter*.
3. Repeat these steps for the field labeled *First Manager* and name it “**Manager Name**”.

Figure 12.7 shows the renamed tooltips fields:

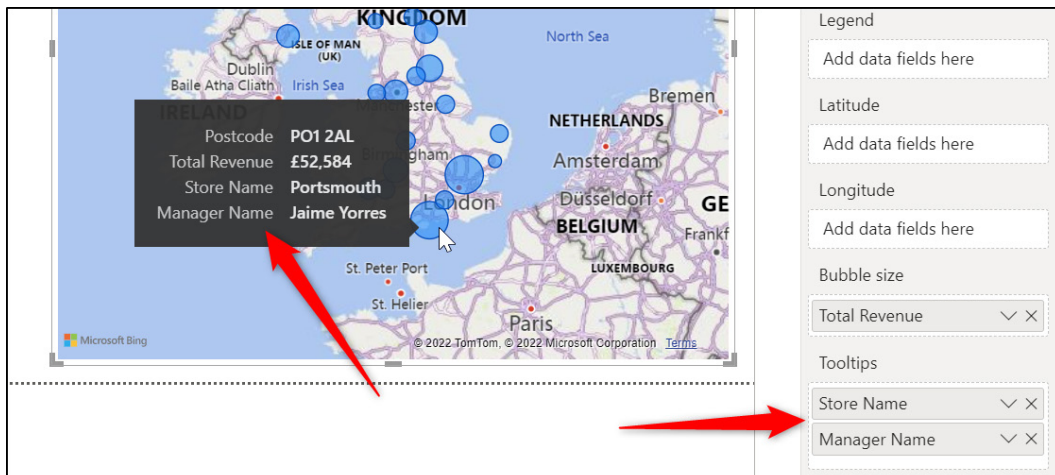


Figure 12.7: Tooltips fields renamed for the Map visual

Editing the map title

The map title includes all fields added to the *Location*, *Bubble size*, and *Tooltips* wells of the visual. This has resulted in a messy title. Let us improve it.

1. With the Map selected, click the **Format your visual** button at the top of the **Visualizations** pane.
2. Click the **General** category and click on **Title** to expand the list of its formatting options.
3. Click in the **Text** box and edit the text to read **Total Revenue by Store**.
4. Click on the **Font** list and select the **Segoe UI** font and size **16**. Click the **Bold** button.

Figure 12.8 shows the Map with the improved title:

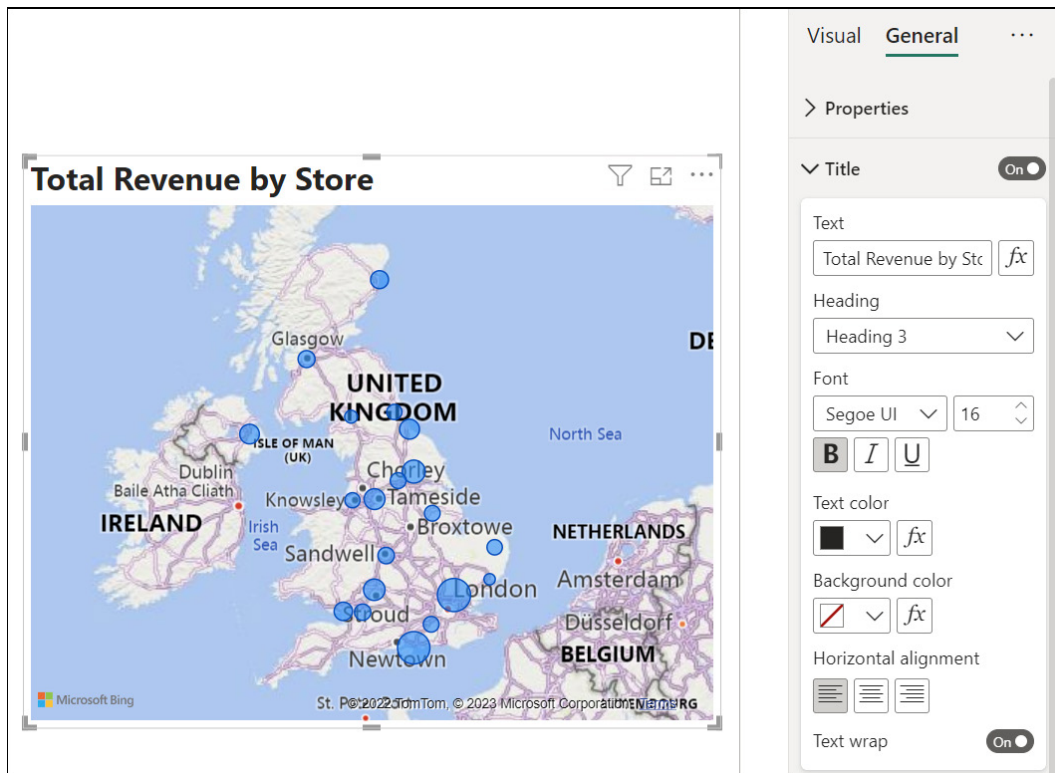


Figure 12.8: Map with improved title

Changing the map style

The default style applied to the Map is the *Road* style. In this style, features of the Map, such as the roads, are quite prominent.

Let us change the style to make the Map features more subtle and the bubbles more evident. Either the *Grayscale* or *Light* styles are great for this. In this example, the *Light* style will be applied.

1. Click the **Visual** category of the *Format your visual* tab in the **Visualizations** pane.
2. Click on **Map settings** and then **Style** to expand the formatting options for that section.
3. Click the **Style** list arrow and select **Light** from the style options as shown in figure 12.9.

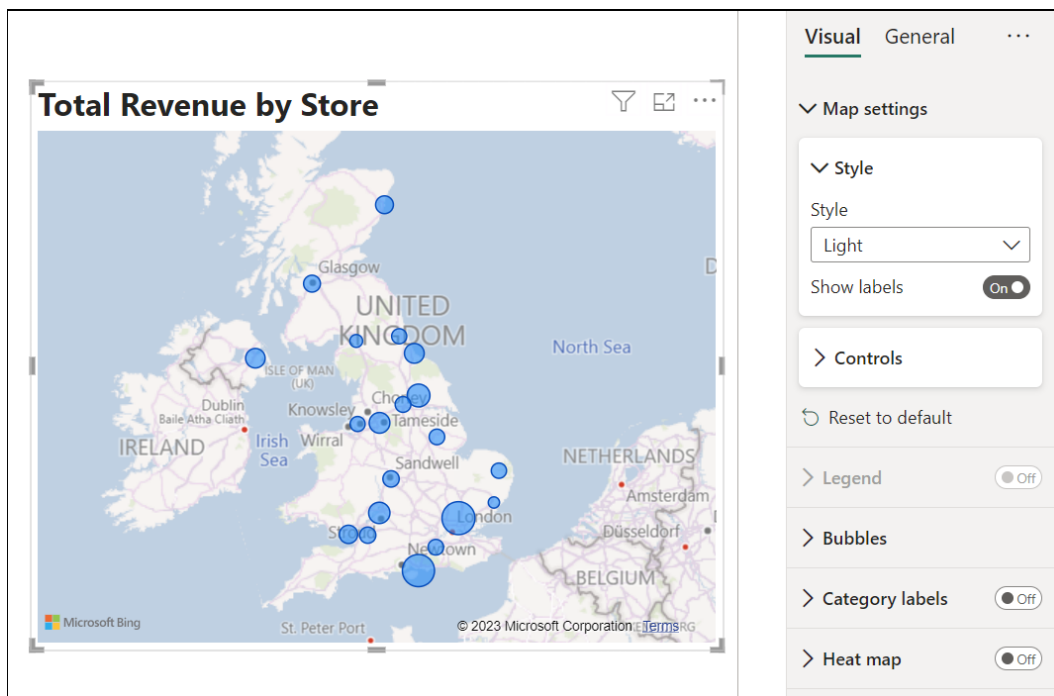


Figure 12.9: Changing the map style

The *Show labels* toggle switch turns off the labels of the countries, cities, seas, and so on. This is a nice option if the names of the areas are not required, as it reduces the “noise” on the Map. The focus should be on the values as much as possible.

Formatting the bubbles

Let us now format the bubbles to make them “pop” on the Map more.

1. Click on **Map settings** to collapse the options and click **Bubbles** to expand its formatting options.
2. In the **Colors** section, click the *Default* list arrow and select a color. In *figure 12.10*, a dark blue has been selected.
3. The bubble size can also be changed from its initial size set by the Map visual by using the spin arrows or the slider in the **Size** section.

Note: Conditional formatting rules can be applied to the bubbles using the fx button beside the color list. This feature would provide further insight into the performance of the data points on the Map.

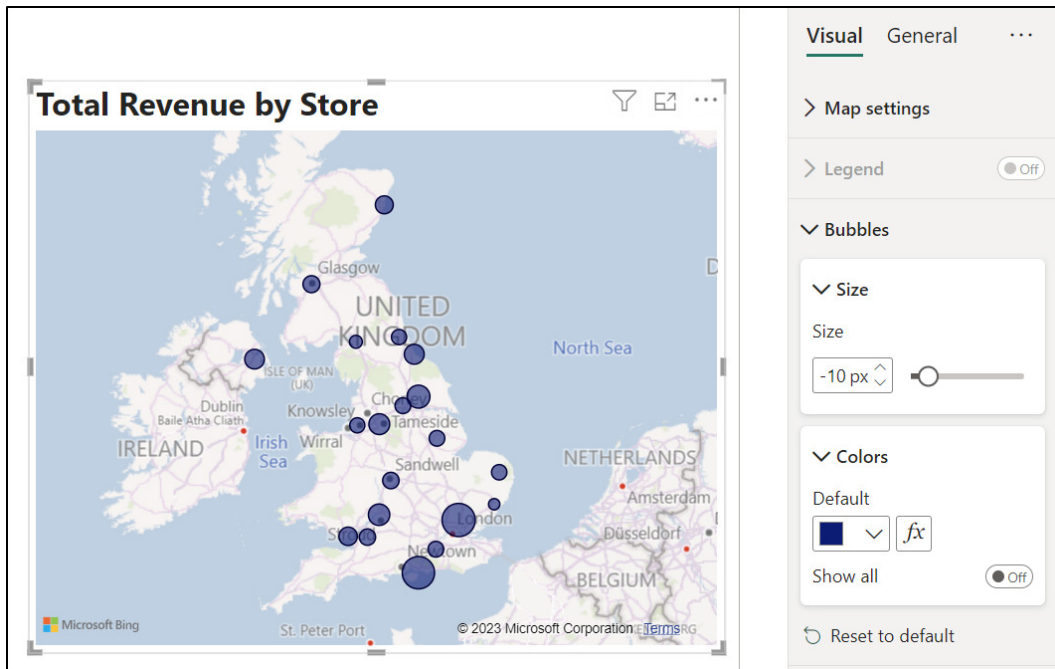


Figure 12.10: Formatting the bubble size and colors

Using a table to filter the map data

The auto-zoom functionality of the Map visual can be very effective when used appropriately. When the map data is filtered, and that filter reduces the data points to a smaller area on the Map, the Map automatically zooms into the area that contains the remaining data points.

This can work very well.

To take advantage of this functionality, let us add a table that lists the region names. This table can then be used to cross-filter the Map visual.

Note: Chapter 14, *Report Interactions, Filters, and Slicers* details different methods for filtering report visuals, including other visuals on the page, Slicers, and the Drill through the feature of Power BI.

1. Click on a blank area of the page to ensure that no visuals on the page are selected.
2. Click the **Table** icon in the **Visualizations** pane.
3. Click and drag the **Region** field from the **Stores** table into the *Columns* well.
4. Move, resize, and format the table as desired.

Figure 12.11 shows the table visual beside the Map. The table has had the alternate background color removed, the title edited to pronounce it more, and a border added. We will not go through those formatting steps now, as they have been covered previously in the book.

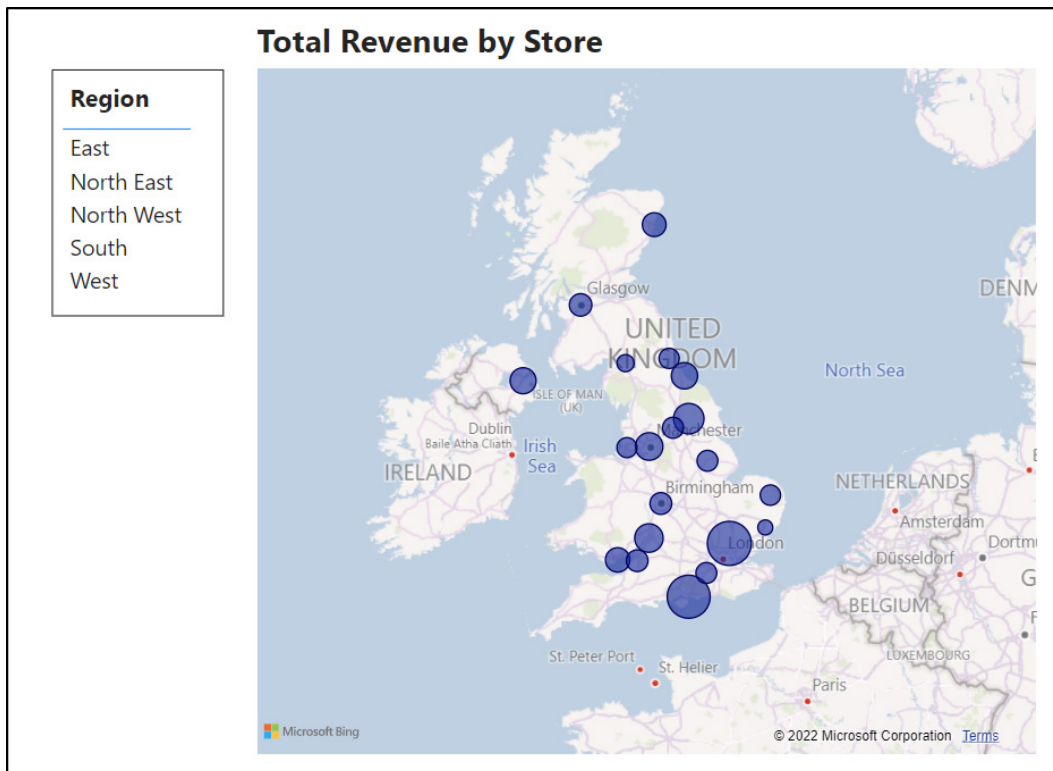


Figure 12.11: Table visual listing the regions to be used as a filter for the Map visual

Clicking a region name in the table visual cross-filters the map data. The Map automatically zooms into the specified region.

In *figure 12.12*, the *North East* has been selected in the table visual. The Map reacted and zoomed into the *North East* area of the UK.

Note: You can edit the behavior of a visual and how or if it cross-filters or cross-highlights other visuals on a report page. This is explained in *Chapter 14, Report Interactions, Filters, and Slicers*.

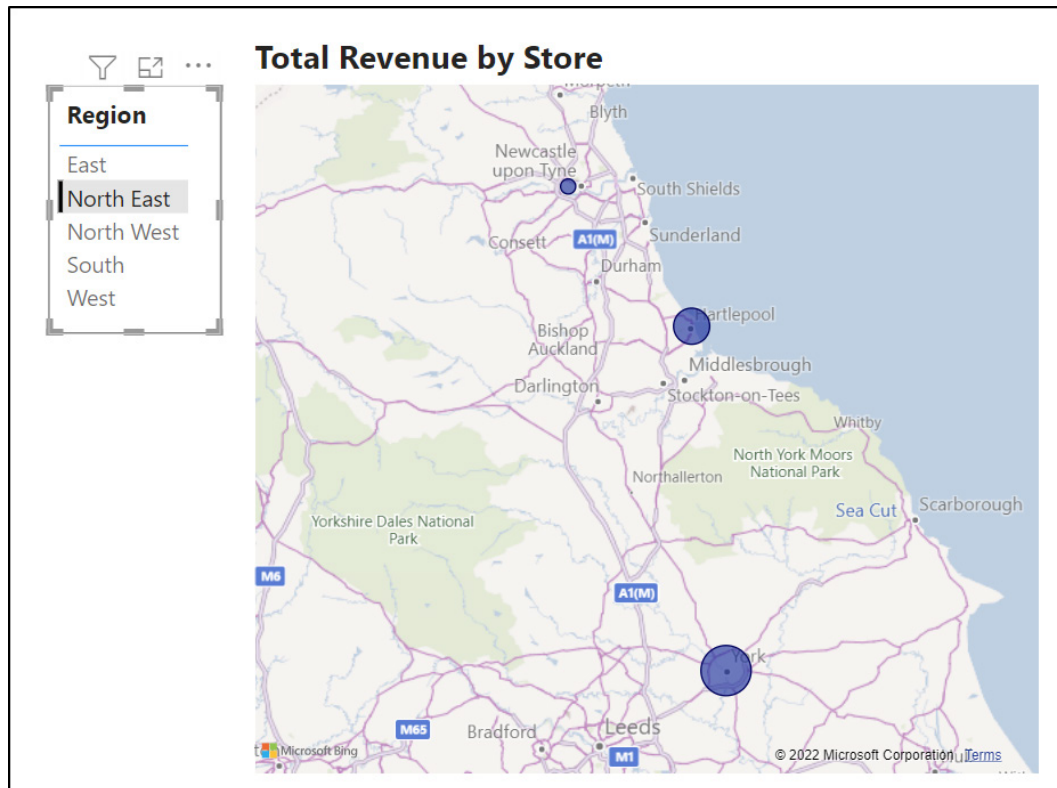


Figure 12.12: Map zoomed into the North East area of the UK

Disabling the auto-zoom feature

Another aspect of the auto-zoom feature of map visuals is that it zooms back to show all the data points when the Map leaves focus. For instance, a reader can zoom in on an area of the Map visual, but when they change the page in the report and return to see the Map, it will have automatically zoomed out to show all data points again. It will not retain the zoom level made by the reader.

We have seen the positive uses of this auto-zoom functionality when the Map is filtered to a specific area. But when a user is manually zooming into an area, the automatic return to show all data points may not be desired. So, let us see how to disable the auto zoom feature.

1. Click on the Map visual to select it.
2. Click the **Format your visual** button at the top of the **Visualizations** pane.
3. In the **Visual** category, click **Map settings | Controls** to expand the options in that section.
4. Click the **Auto zoom On/Off** slider to turn it **Off**, as shown in *figure 12.13*:

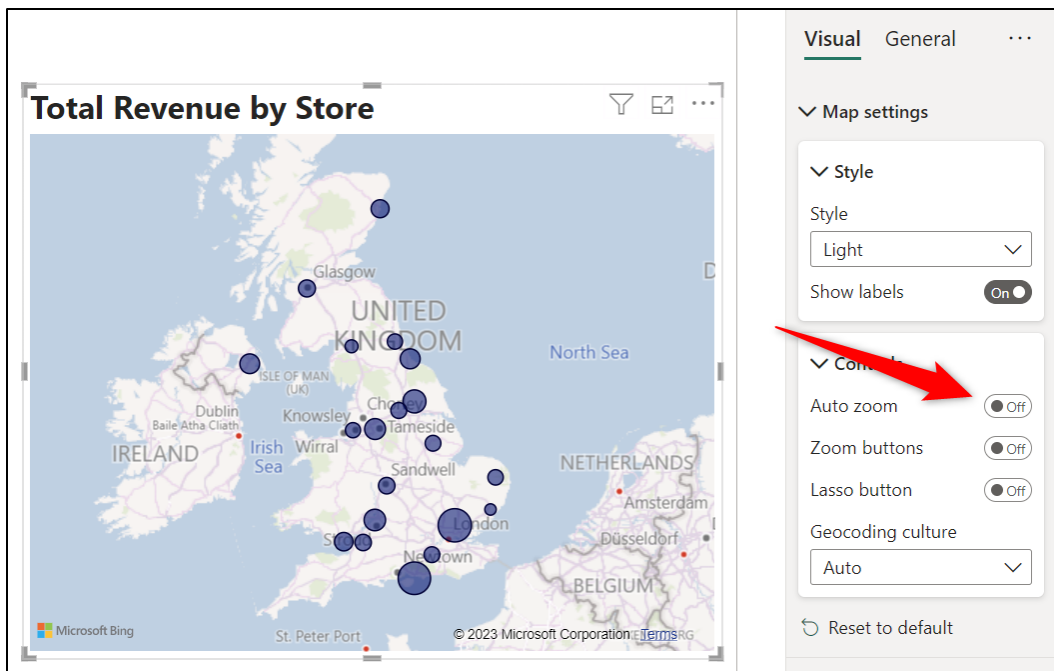


Figure 12.13: Disabling the auto-zoom feature of the Map

Tips for using the map visuals

If the Power BI map visuals struggle to return to the correct location, there are some tips to improve the likelihood of accurate results.

Categorize the geographic fields

When the Map visual is having trouble geocoding your data, you can help it by categorizing the geographic fields. This can increase the likelihood of correct geocoding.

Note: Using the latitude and longitude field wells provides the most accurate method of visualizing geographic data. If you have access to the dataset, including latitude and longitude values is encouraged.

Although the mapping is working very well with this dataset, let us see how we can categorize a field in Power BI Desktop. We will categorize the **Postcode** field.

1. In the Data view, click the **Stores** table in the **Data** pane to navigate to it.
2. Click the **Postcode** column.
3. Click **Column tools** and then click the **Data category** list arrow.
4. Choose the appropriate category from the list, shown in *figure 12.14*. In this example, it is the *Postal code*:

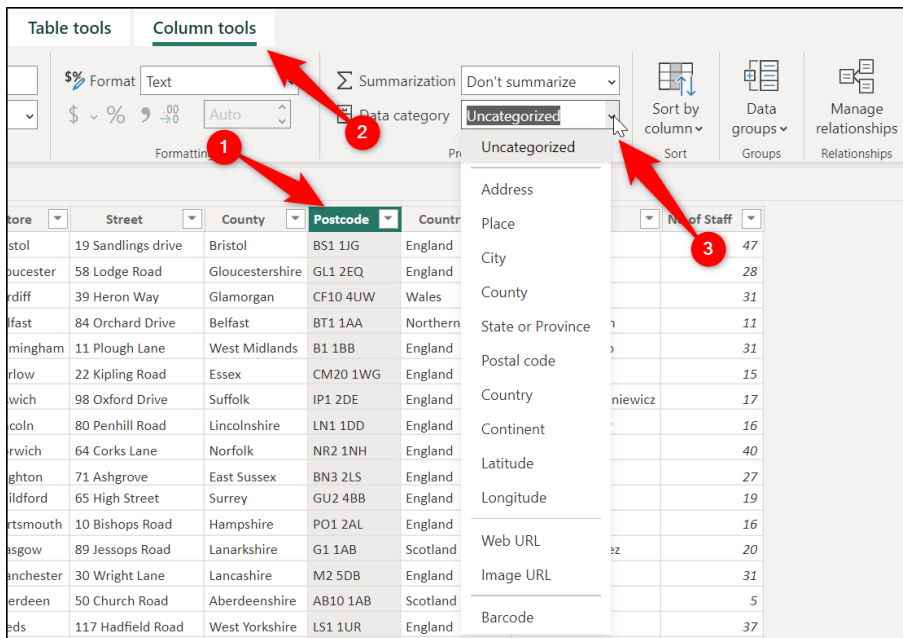


Figure 12.14: Categorizing a column in the Data view

A globe icon is displayed beside the column name in the *Data* pane of the Power BI views. This identifies that a location data category has been assigned. *Figure 12.15* shows the data category icon beside the **Postcode** field in the Report view:

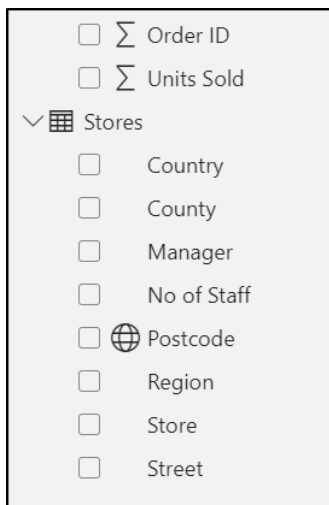


Figure 12.15: Data category icon beside field name in report view

Using latitude and longitude data

If you have access to the dataset, including the latitude and longitude data for each location would ensure quick and accurate results. This is the best thing you can do if it is a viable option.

You can find the latitude and longitude data for addresses using sources such as **latlong.net**.

The postcode data in this example is working very well, although there is an issue with the Brighton store not being recognized. Let us fix this by replacing the postcode on the Map with the latitude and longitude data for greater efficiency.

Currently, our dataset does not contain columns with latitude and longitude data, so we will first import an Excel file that contains this information and then perform a merge query to include the columns in the existing **Stores** table. This provides a nice excuse to recap on some of the skills learnt earlier in the book also.

1. Click **Home | Excel workbook**.
2. Locate and open the **lat-and-long.xlsx** workbook provided in this chapter's files.
3. Check the **Sheet1** checkbox and click **Transform Data**.
4. In the Power Query Editor window, rename the query "**LatLong**".
5. Select the **Stores** query in the *Queries* pane and click **Home | Merge Queries**. This will merge with the existing **Stores** query.

6. In the *Merge* window, as shown in *figure 12.16*, click the **Postcode** column in the **Stores** table and then click the **Postcode** column in the **LatLong** table. These are the two matching columns.
7. Ensure that **Left Outer** is selected as the *Join Kind* and click **OK**.

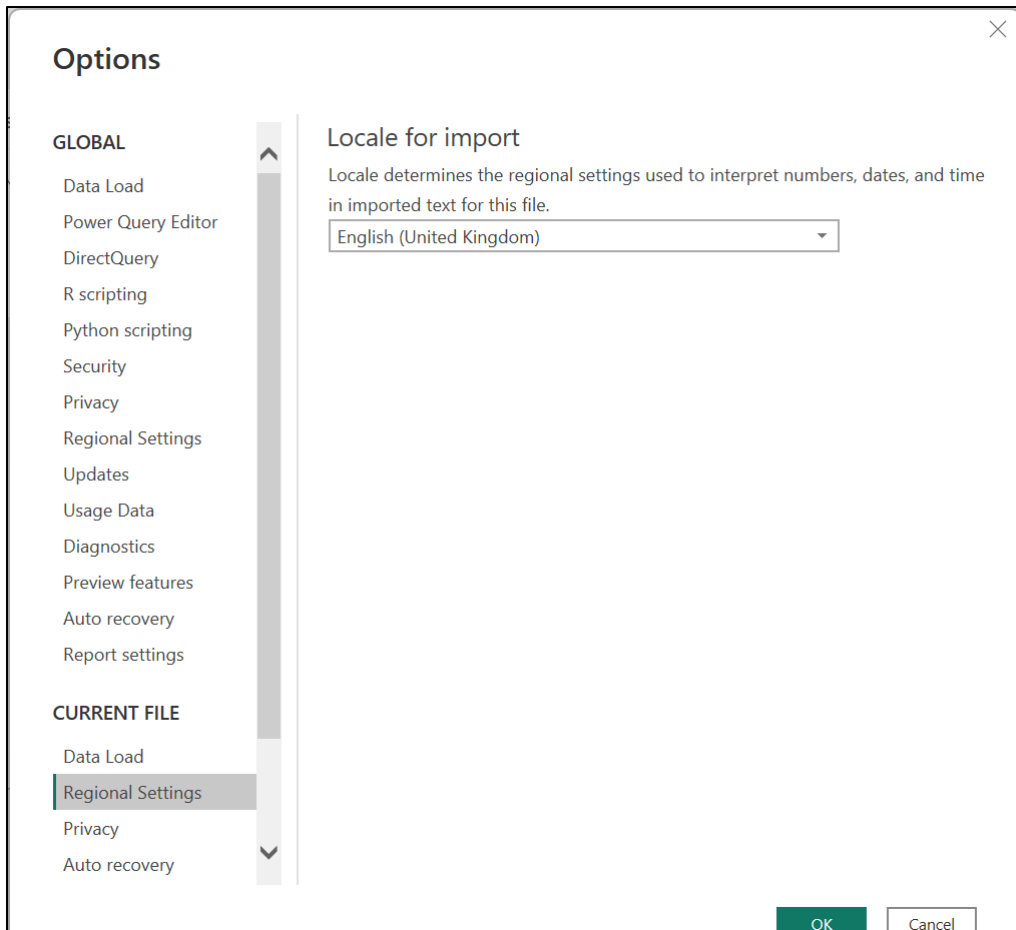


Figure 12.16: Merge query to merge the latitude and longitude columns

8. Click the double-arrow button next to the **LatLong** column name, as shown in *figure 12.17*.
9. Uncheck the **Postcode** box, as that column is already in the **Stores** table.
10. Uncheck the **Use original column name as prefix** box.
11. Click **OK**.

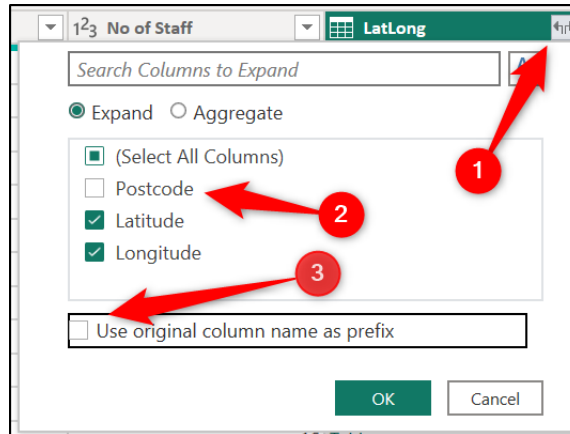


Figure 12.17: Specifying the columns to merge

Right-click the **LatLong** query in the *Queries* pane and click **Enable load** to disable the option, as shown in figure 12.18. With the **Latitude** and **Longitude** columns merged into the **Stores** table, there is no need to load this query/table into Power BI:

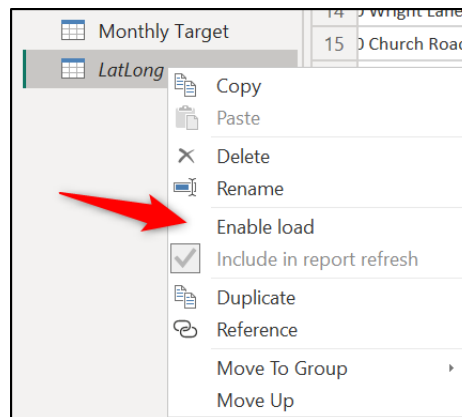


Figure 12.18: Disable the load of the query to Power BI

12. Click **Home** | **Close & Apply**.

With the latitude and longitude data now available in our dataset, there are a few recommended changes to be made to the columns in the data model.

First, the columns should be in a decimal format. Also, we will set them to not summarize when used in a visual and assign a data category, as we learnt in the previous tip.

13. Switch to the data view and click the **Stores** table in the *Data* pane to navigate to it.
14. Select the **Latitude** column, and from the *Column tools* tab, click the *Format* drop-down list arrow and click **Decimal number** as shown in *figure 12.19*.
15. Click the **Summarization** list arrow and select **Don't Summarize**.
16. Click the **Data category** list arrow and select **Latitude**.
17. Repeat Steps 14–16 for the **Longitude** column, ensuring to specify **Longitude** in Step 17.

The screenshot shows the Power BI Column tools ribbon with the following settings:

- Format:** Decimal number
- Summarization:** Don't summarize
- Data category:** Latitude

Below the ribbon, a table of store data is displayed with the following columns: Store, Street, County, Postcode, Country, Manager, No of Staff, Latitude, and Longitude. The Latitude and Longitude columns are highlighted in green, indicating they are selected.

Store	Street	County	Postcode	Country	Manager	No of Staff	Latitude	Longitude
stol	19 Sandlings drive	Bristol	BS1 1JG	England	Helen Bennett	47	51.45	-2.5935
oucester	58 Lodge Road	Gloucestershire	GL1 2EQ	England	Thomas Hardy	28	51.86	-2.24985
rdiff	39 Heron Way	Glamorgan	CF10 4UW	Wales	Yoshi Latimer	31	51.45	-0.03568
fast	84 Orchard Drive	Belfast	BT1 1AA	Northern Ireland	David Thompson	11	54.60	-5.92207
mingham	11 Plough Lane	West Midlands	B1 1BB	England	Antonio Moreno	31	51.53	-0.30637
rlow	22 Kipling Road	Essex	CM20 1WG	England	Annette Roulet	15	51.77	0.09258
wich	98 Oxford Drive	Suffolk	IP1 2DE	England	Zbyszek Piestrzeniewicz	17	52.05	1.14465
coln	80 Penhill Road	Lincolnshire	LN1 1DD	England	Renate Messner	16	53.23	-0.54202

Figure 12.19: Apply the decimal number format to the latitude and longitude columns

The **Latitude** and **Longitude** fields can now be used in the Map visual in place of the **Postcode**.

18. Switch to the report view and click on the Map visual to select it.
19. Click the X beside the **Postcode** field in the **Location** well to remove it.
20. Click and drag the **Latitude** and **Longitude** fields from the <Stores> table into their respective *Latitude* and *Longitude* wells, as shown in *figure 12.20*.

The Brighton store is now shown on the Map. It is the large bubble on the south coast of the UK.

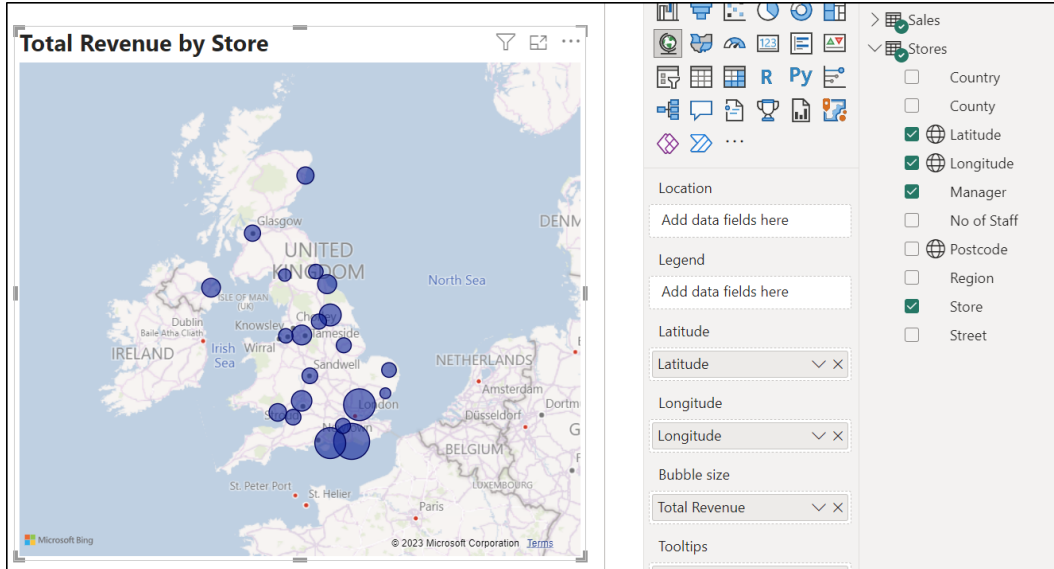


Figure 12.20: Using latitude and longitude in the Map visual

Using geographic hierarchies

Another tip to assist the Map in correctly geocoding location data is to add multiple hierarchical fields to the *Location* well. For example, the **County** field can be added along with the **Postcode**.

By adding multiple fields at different levels of geographic detail, a hierarchy is created. A reader can drill up and down through the different levels of the hierarchy in the visual.

These drill-up and drill-down techniques were covered when discussing the Matrix visual in *Chapter 10, Cards and Other Text Visuals*. We will not go through them again now, just know that it is possible. At this moment, we are only interested in improving the accuracy of the map geocoding.

1. Click on the Map to select the visual.
2. Click the **X** beside the **Latitude** and **Longitude** fields to remove them from their respective wells.
3. Click and drag the **County** and **Postcode** fields from the **Stores** table into the *Location* well in that order, as shown in *figure 12.21*.

This approach also solves the problem with the Brighton store. This is shown in *figure 12.21* by the arrow.

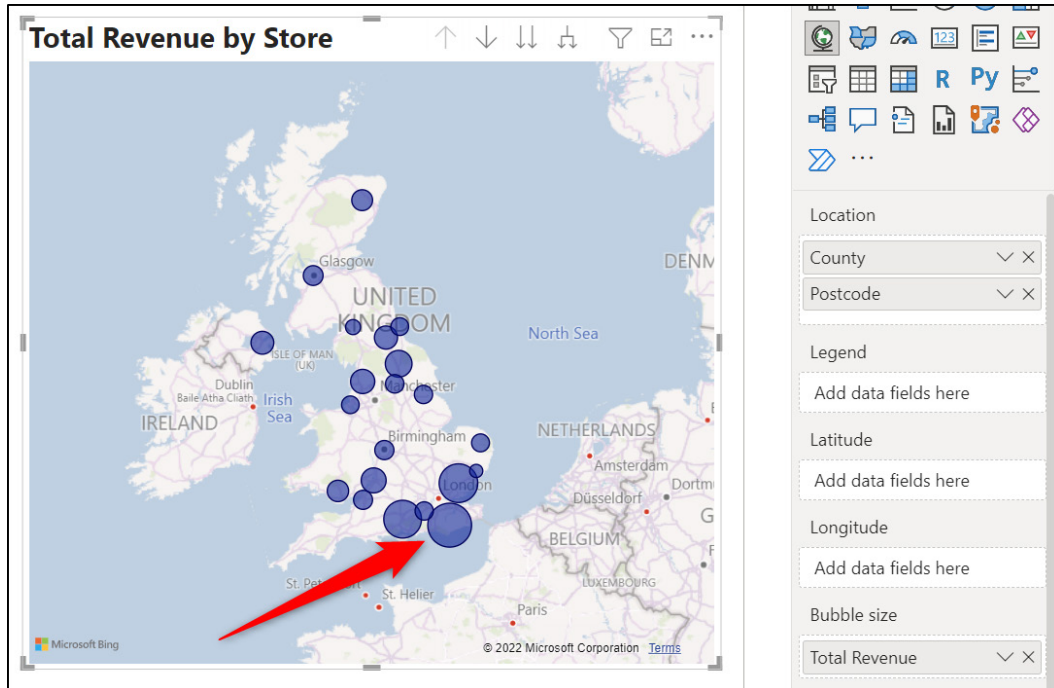


Figure 12.21: Using a hierarchy by adding two geographic fields in the Location well

The drill-up and drill-down buttons are displayed in the visual header. These could be removed, if required, from the **General | Header icons** section in the formatting options of the *Visualizations* pane.

Conclusion

In this chapter, we learnt how to use the Map visual in Power BI. We also covered some tips to geocode your fields effectively, including how to categorize your geographic fields and use latitude and longitude data.

In the upcoming chapter, we will look at the AI visuals in Power BI, such as the Q and A, decomposition tree, and key influencers visuals.

We then look at adding custom visuals to Power BI. There is an entire marketplace of visuals that extend beyond those provided by default in the **Visualizations** pane. We will add a custom visual from the marketplace and use it in our report.

Questions

Here are some questions to test what you have learnt in this chapter.

1. Using latitude and longitude values ensures the greatest chance of accuracy when plotting data on a map.
 - a. True
 - b. False
2. Which of the following statements is true of maps?
 - a. Maps use the Bing maps geocoding engine to correctly map location fields such as country, city, and state.
 - b. You can change the color of the bubbles on a map.
 - c. There are different map styles, including road, light, and grayscale.
 - d. All of the above
3. What are the correct steps to specify a data category for a column?
 - a. From report view, select the column > Modelling tab > Data category
 - b. From Data view, select the column > Column tools > Data category
 - c. From Data view, select the column > Home > Data category
 - d. None of the above
4. The auto zoom feature of a map can be disabled if required.
 - a. True
 - b. False

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



CHAPTER 13

Other Power BI Visualizations

Introduction

There are many visualizations in Power BI, and these visualizations are consistently changing and improving due to the fast-paced development that Microsoft has rolled out.

In this chapter, we will cover even more useful visualizations of what has been covered so far. These involve some of the AI visuals of Power BI, including the Q&A, Decomposition Tree, and Smart Narrative visuals.

We will also see how to add a custom visual from the AppSource marketplace in Power BI. You can find a rich array of visuals beyond those that are immediately available in the Visualizations pane of Power BI. In this chapter, we will see how to access these additional apps and add a visual from AppSource to our report.

With each visual, we will cover how and why to use them and apply some useful formatting options.

Structure

In this chapter, we will cover the following topics:

- Q&A visual
- Configuring the Q&A visual
- Decomposition tree
- Smart narrative
- Custom visuals in Power BI

Objectives

After reading this chapter, you will learn how to use the powerful AI visuals in Power BI to gain additional key insights into your data that you otherwise could not have seen.

You will also know how to access further visuals from the custom visual marketplace, known as Microsoft AppSource, and add one to your Power BI report. There is a wealth of visuals beyond those available as standard from the *Visualizations* pane.

The Q&A visual

Files: **sales-report-ch-13-start.pbix**

The Q&A visual enables a user to enter a natural language question to produce a visual. This provides a quick and effective way for a user to get answers from the data.

Q&A is a very powerful visual with a lot of scopes to the level of configuration and formatting that one can apply to it. This is great because you will want to refine the Q&A to work as best as possible with the questions that a user may ask and the terms that they may use.

Using the Q&A Visual

Let's see how to insert and use the Q&A, including making some configurations to optimize its use in our report.

Insert a new page to the sales report we have been working on during this book and name it "AI".

Click the **Q&A** icon in the **Visualizations** pane to insert the Q&A visual on the page (*figure 13.1*).

Tip: You can also insert the Q&A visual by simply double-clicking anywhere on a report page.

You can use the Q&A by either typing a question in the **Ask a question about your data** box or by clicking one of the suggested questions that follow, as shown in *figure 13.1*:

The Q&A is prompting to **Add synonyms now** to improve the question-and-answer experience. We can close this prompt and will see how to add synonyms later in this chapter, along with how to create your own suggested questions:

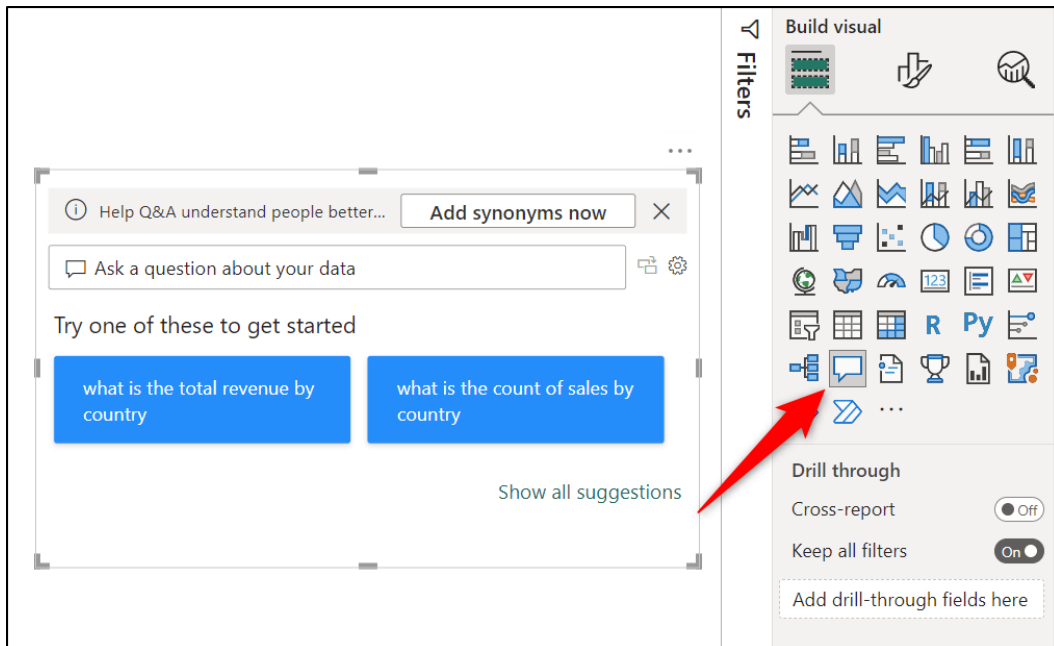


Figure 13.1: Inserting the Q&A visual

1. In *figure 13.2*, the suggested question “what is the total revenue by country” was clicked. It has produced a bar chart comparing the total revenue for each country.

If the visual is one you want to keep, click the **Turn this Q&A result into a standard visual** button to the right of the question box.

Let us now ask another question of the data. And this time, we will ask our own questions using the **Ask a question about your data** box.

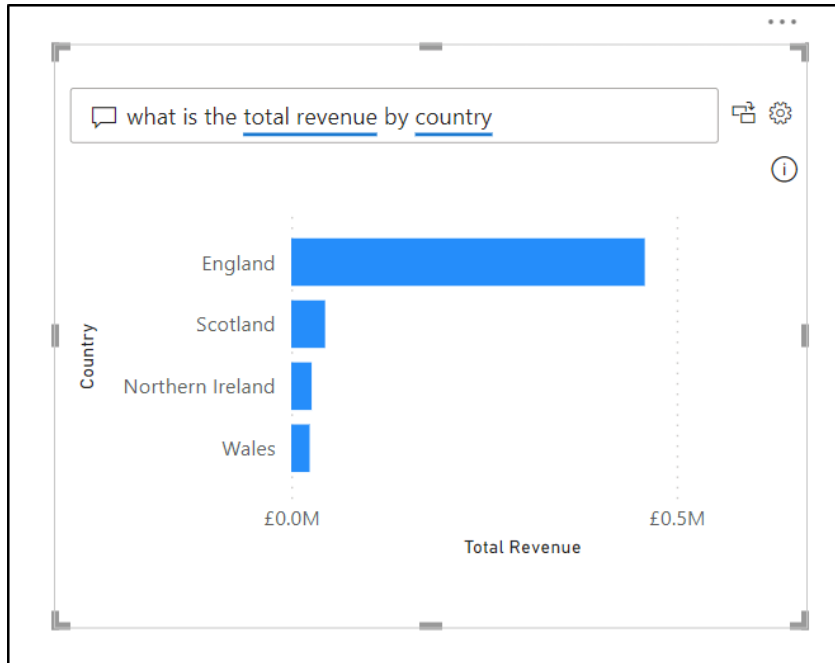


Figure 13.2: Bar chart showing total revenue by country

2. Click the speech icon on the left of the question box to select all the question text easily, and press *Delete*.
3. Start typing the question “**what are the top 5 products by total revenue**” into the *question box*. As you are typing, the autocomplete feature is suggesting terms and questions to you, as shown in figure 13.3:

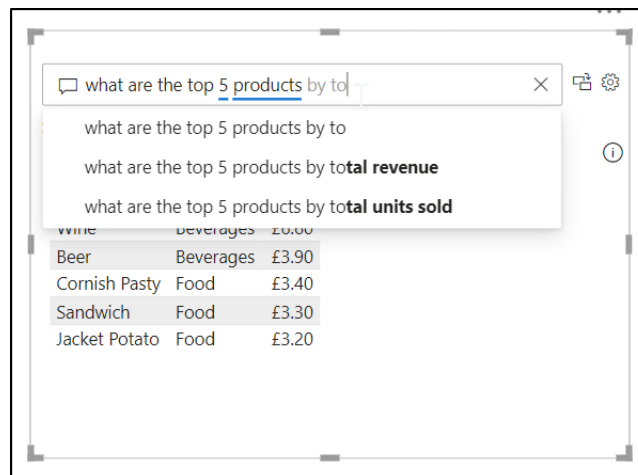


Figure 13.3: Autocomplete list appears as you type a question

4. Press *Enter* on completing the question. *Figure 13.4* shows a bar chart visual that has been used again, this time showing the Top 5 products by total revenue in descending order.

This is a great choice of visual and demonstrates how quickly you can gain effective insights into your data and even create visuals without creating them manually:

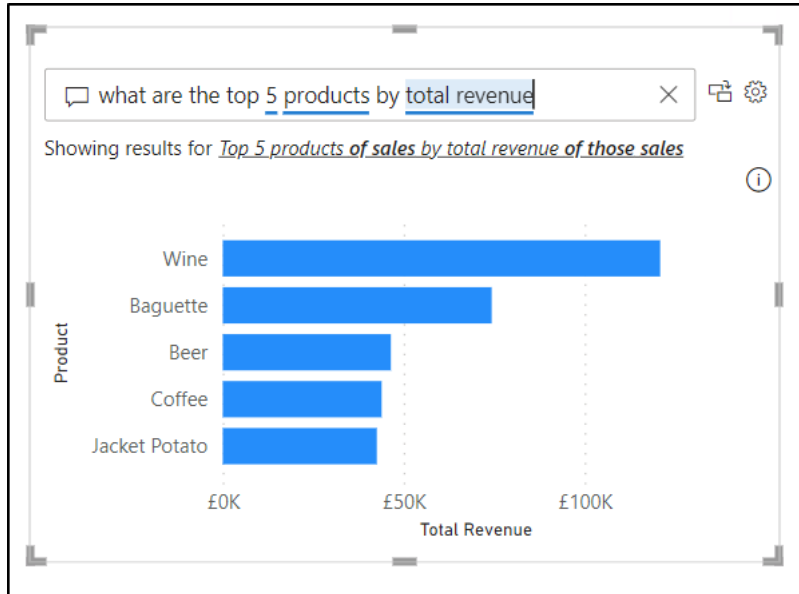


Figure 13.4: Bar chart showing the Top 5 products by total revenue

The visual will cross-filter or cross-highlight all other visuals on the page unless instructed otherwise. You do not need to convert it into a standard visual for this functionality to work.

In *figure 13.5*, the bar for the *Jacket Potato* product has been clicked. This has filtered the line chart (note the axis) that is on the same page.

This ability to quickly ask a question of our data, create a visual, and be able to interact with other visuals on a page in just a few seconds is fantastic.

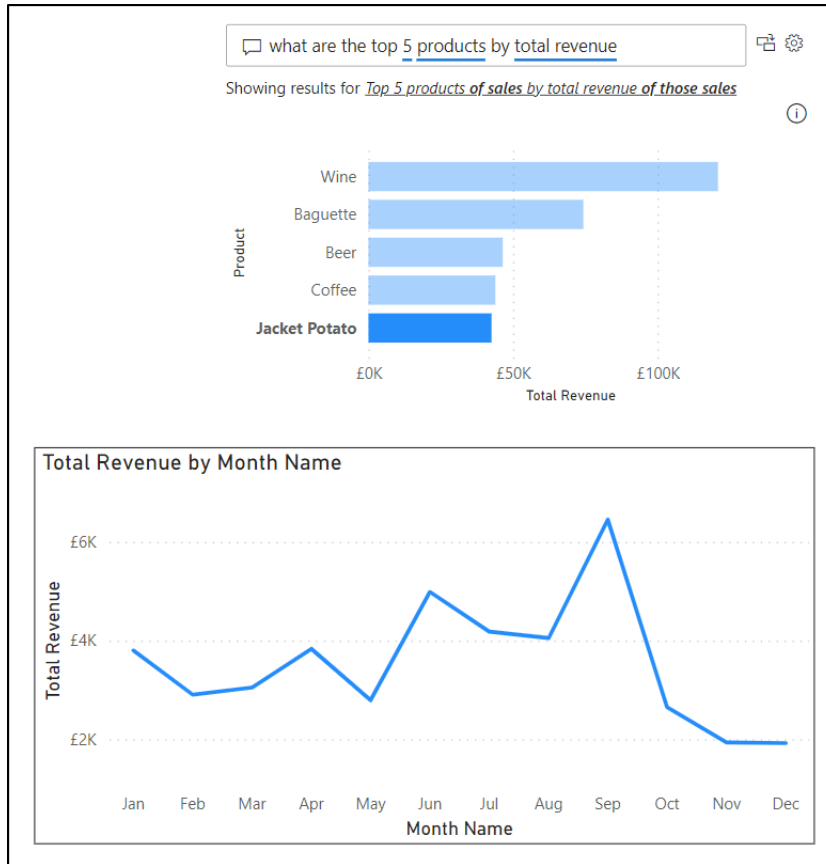


Figure 13.5: Q&A filtering another visual on a page

Let us convert the Q&A results into a standard visual to keep it by clicking the **Turn this Q&A result into a standard visual** button, as shown in figure 13.6:

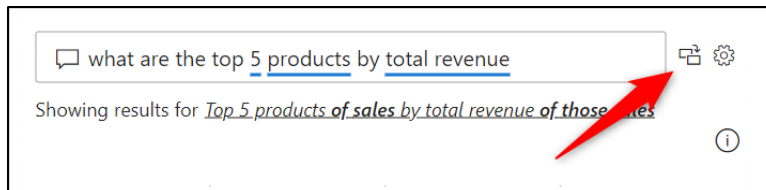


Figure 13.6: Convert the Q&A result into a standard visual to keep

The Q&A visual will also respect any other filters applied. To see an example of this, we will add a Q&A visual to the **Monthly Sales Analysis** page of the report. This page contains a month Slicer that was added in *Chapter 10, Cards and Other Text Visuals*.

We will use the Q&A visual to return the name of the best sales rep in respect of the revenue change from the previous month. We have a measure named “Monthly Revenue Difference” that was created in *Chapter 9, Adding DAX Measures* and calculates the sales reps revenue change. Let us ask the question, “who is the best rep by monthly revenue difference”.

Figure 13.7 shows the question entered in the question box. Notice the double red line under the word “best”. The double red underline identifies an unrecognized word.

The returned visual shows “Yuezhu Zhang” as the best sales rep. This is correct, but it has happened due to fortune and not because the question has worked. The Q&A is simply listing all reps in descending order. We need to define what the term “best” means or use a different term.

Note: We look at some aspects of optimizing the Q&A visual shortly.

In *figure 13.7*, we also see the autocomplete list helping us to construct the question again. It offers the question “who is the best rep by % monthly revenue difference” because we have a measure named % Monthly Revenue Difference in our model.

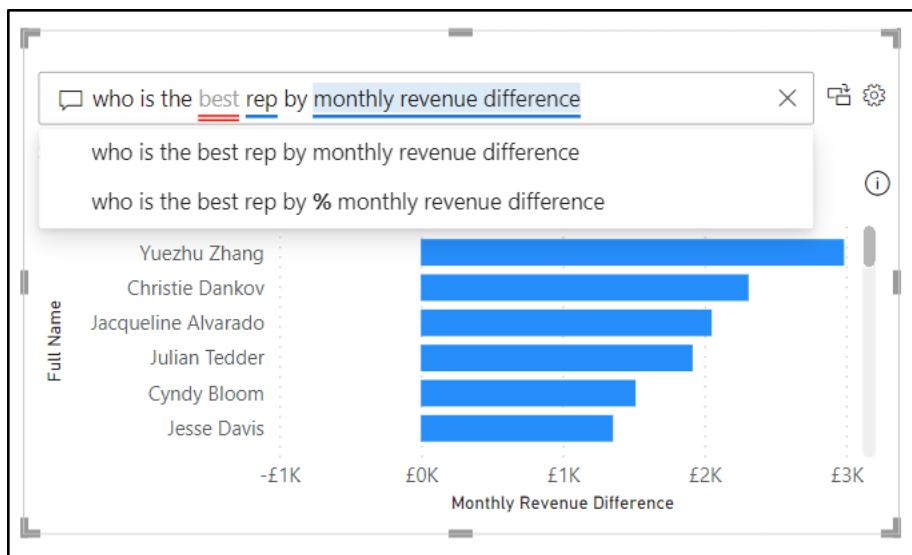


Figure 13.7: Unrecognized words and suggested questions

Let us change the term “best” to “top”. In *figure 13.8*, the change to the question has been made, and the double red underline has disappeared. The visual has also changed to show only one sales rep.

Figure 13.8 shows the Slicer that is filtering the Q&A visual to show data from March only. A table visual to show the top sales reps has been inserted too. The table is used only for the purpose of showing that the Q&A is returning the correct name and value:

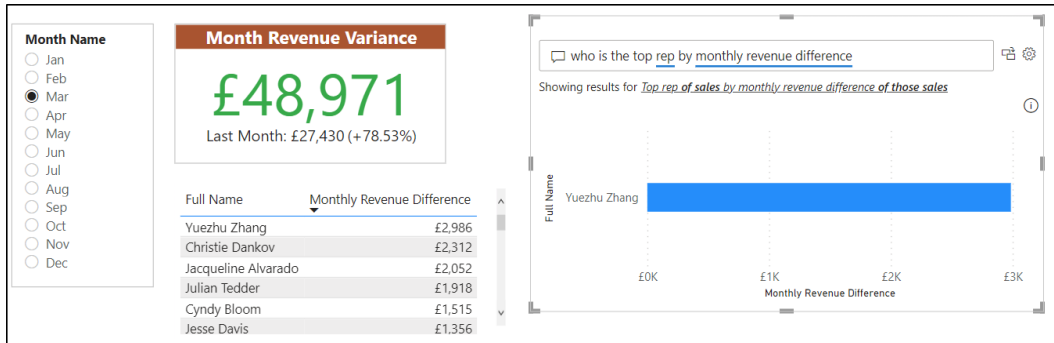


Figure 13.8: Q&A being filtered by a Slicer

If you want the results shown in a specific visual, this can be stated at the end of the question. In figure 13.9, the word “table” has been added to the end of the question to force the result to be shown in a table visual:

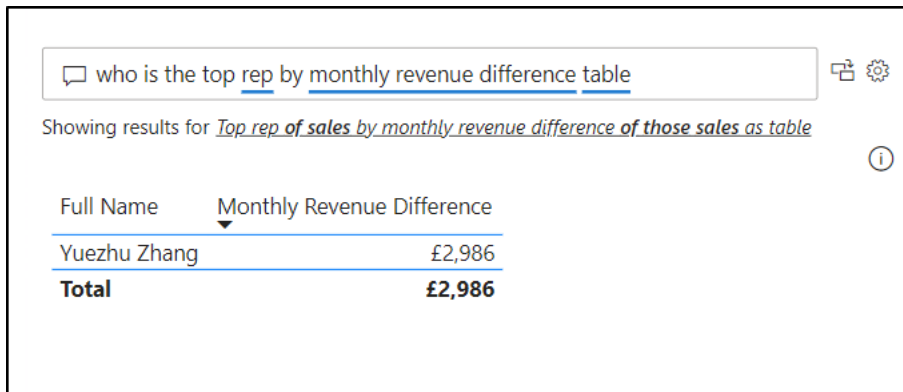


Figure 13.9: Specifying the result to be in a table

Formatting the Q&A

There are some interesting formatting elements that can be applied to the Q&A visual. These formatting options are only for the Q&A and not for the visual returned because of the question, for example, the bar chart, table, map and so on.

In figure 13.10, the *Visual* category for the formatting options is shown. The font of the question has been increased to size 14.

Note the options to change the color of the *Underline*. This is a useful formatting option. The red *Error color* is not easily visible by everyone, and the blue *Accepted color* clashes with other colors, such as the table header underline. Great to know that these can be altered.

General formatting changes such as border and background color can be made as with all visuals.

Who is the top rep by monthly revenue difference table

Showing results *Top rep of sales by monthly revenue difference of those sales as table* for

Full Name	Monthly Revenue Difference
Yuezhu Zhang	£2,985.8
Total	£2,985.8

Visual General ...

Question field

Text

Font: Segoe UI, 14

B I U

Color: [Black]

Background

Underline

Accepted color: [Blue]

Error color: [Red]

Warning color: [Orange]

Figure 13.10: Formatting the Q&A visual

Configuring the Q&A visual

There are many options available to configure the Q&A visual to your needs. These include reviewing questions that users have asked, adding your own suggested questions, and teaching the Q&A the meaning of different terms so that it may understand questions better. The options are quite extensive.

To access the tooling options, click the **gear icon** in the top right corner of the Q&A (figure 13.11):

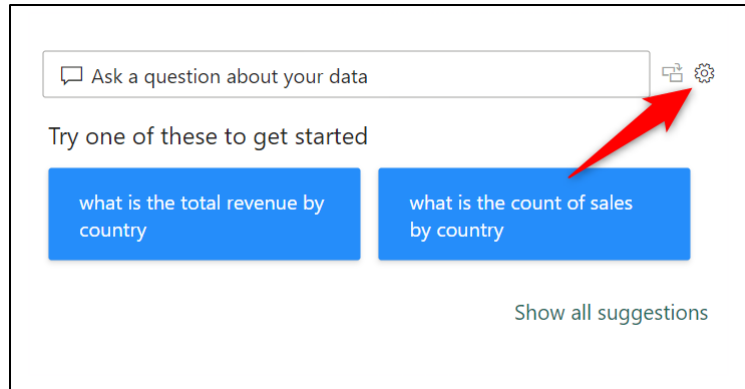


Figure 13.11: Access the setup options for the Q&A visual

The **Getting started** window opens (*figure 13.12*), providing access to the different configuration options:

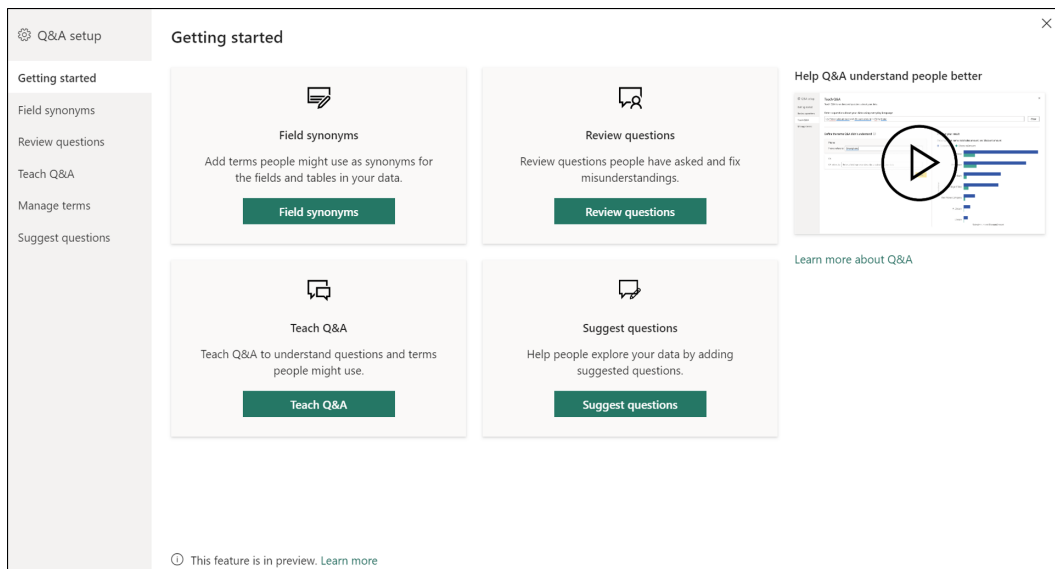


Figure 13.12: Getting started window with Q&A optimization options

Let us explore two of the different options available for optimizing the Q&A visual—adding synonyms and creating suggested questions.

Adding synonyms

You can add alternative names for the fields and tables of your model by adding synonyms. This is very useful. There may be multiple different terms that are used to refer to a table or field in an organization, and by adding synonyms, we are training

the Q&A to recognize them.

Let us add synonyms for the **Reps** table.

1. Click the **Field synonyms** button in the **Getting started** window or from the menu down the left side.
2. All tables of the model are shown. Click the **Reps** table to expand the **Terms** and **Suggested terms** for the table and fields of the table (*figure 13.13*).
3. Click the **representative** suggested term to add it as a term for the **Reps** table. If **representative** is not a suggested term, pick another that you think is a good alternative.
4. Click the **Add** button, type “**salesperson**”, and press *Enter*.

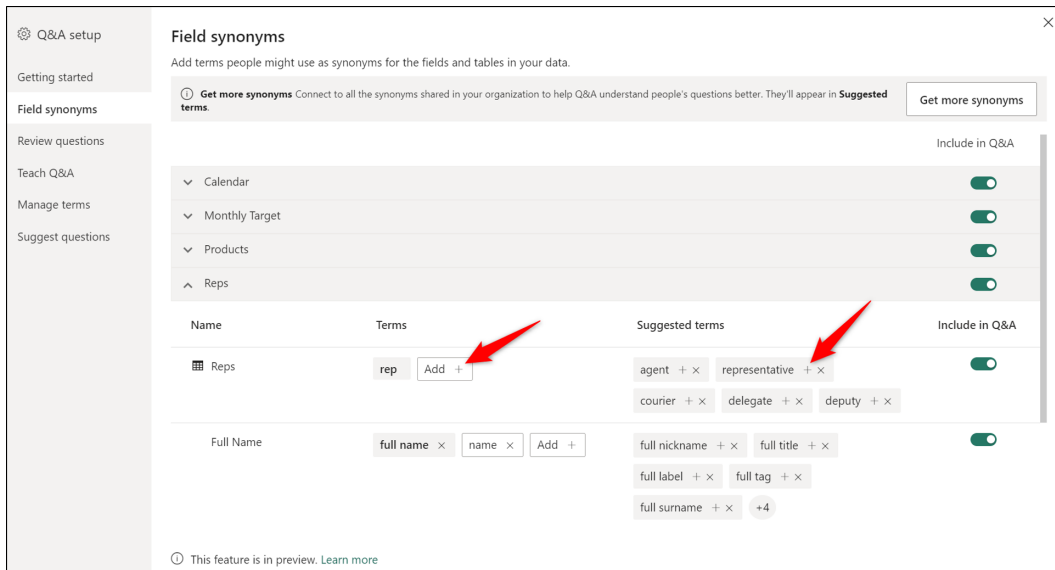


Figure 13.13: Field synonyms area of the Getting started window

5. *Figure 13.14* shows the **representative** and **salesperson** terms added for the **Reps** table. The **Suggested terms** list is revised each time a term is added. Click **salesclerk** to add that term.



Figure 13.14: Added terms and revised suggested terms

6. Click the **X** to close the **Getting started** window.

Synonyms can also be added and changed in the model view, though it is not as user-friendly, and you will not be offered suggested terms.

In *figure 13.15*, the **Synonyms** box in the **Properties** pane shows the terms that were added for the **Reps** table. You can click on any table, field or measure in the model view and add terms to its *Synonyms* box. Each term is separated by a comma.

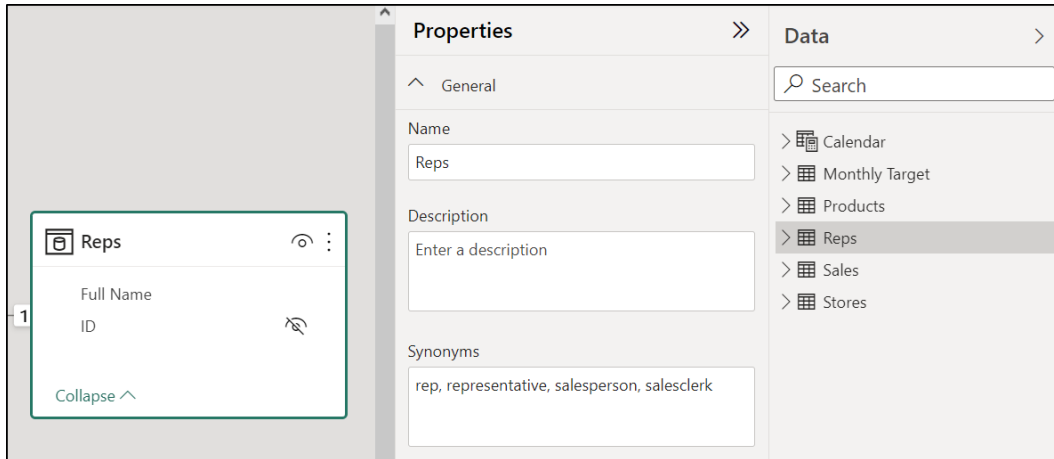


Figure 13.15: Synonyms shown in the Properties pane of the model view

Figure 13.16 shows the new **Salesperson** term being used in a question. The blue underline indicates that *salesperson* is an accepted term:

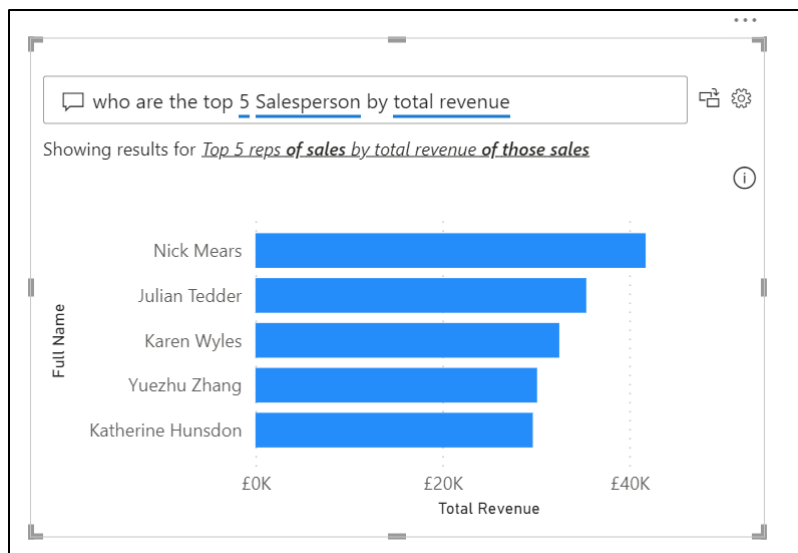


Figure 13.16: Using the new “salesperson” term in a question

Creating suggested questions

Let us now see how to create your own suggested questions. These suggested questions appear directly below the question box when using the Q&A.

The question that we added earlier for the Top 5 products by total revenue could be added as a suggested question for anyone using the Q&A. Users can then click this question instead of worrying about the correct terms to use.

1. Click the **gear icon** in the top right corner of the Q&A to open the *Getting started* window.
2. Click **Suggest questions** in the menu on the left.
3. Type the question “*what are the top 5 products by total revenue*” in the question box provided and click **Add**.
4. Add any other questions you want. *Figure 13.17* shows three questions that have been added and a preview of the result for the first question in the list.
5. Click **Save**.

You can reorder the list of questions by simply clicking and dragging them into the desired order.

The screenshot shows the 'Suggest questions' window in Power BI. On the left is a sidebar with navigation options: Q&A setup, Getting started, Field synonyms, Review questions, Teach Q&A, Manage terms, and Suggest questions. The main area is titled 'Suggest questions' and contains a text input field with the question 'what are the top 5 products by total revenue' and an 'Add' button. Below the input field, it shows 'Showing results for Top 5 products of sales by total revenue of those sales'. Underneath, there is a section 'Reorder your suggested questions' with three items: 'what are the top 5 products by total revenue', 'total revenue by region and category', and 'total revenue by region and category stacked bar'. At the bottom right is a 'Save' button. On the right side of the window, there is a 'Preview your result' section showing a horizontal bar chart for the question 'what are the top 5 products by total revenue'. The chart lists products and their total revenue in £000.

Product	Total Revenue (£000)
Wine	~105
Baguette	~75
Beer	~55
Coffee	~50
Jacket Potato	~45

Figure 13.17: Adding suggested questions to the Q&A

Figure 13.18 shows the suggested questions that were added to the Q&A. A user can generate a visual by simply clicking the suggested question they like.

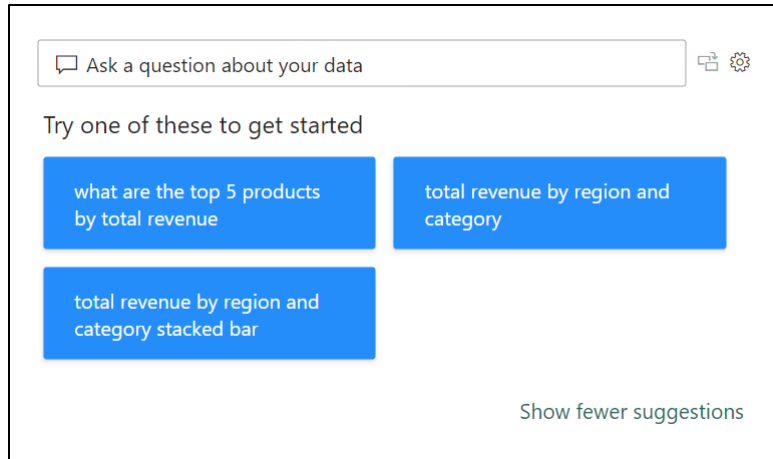


Figure 13.18: Suggested questions listed in the Q&A

Decomposition tree

The decomposition tree visual is used to drill down into aggregation and explain it by different dimensions. For example, to drill down into the total revenue and break it down by region and store.

The user can follow the thread (branches of the tree) through each dimension and understand the contribution each value of the dimension has made to the aggregated value.

The decomposition tree is great for ad-hoc analysis, as a user can specify the dimension to drill into and easily switch between the different values of that dimension with the click of a button.

1. Click on the **Decomposition tree** icon in the **Visualizations** pane.
2. Click and drag the **Total Revenue** measure into the *Analyze* well.
3. Click and drag the **Category**, **Product**, **Region**, and **Store** fields into the *Explain by* well, as shown in figure 13.19.
4. Click the **+** icon beside the aggregated value to choose the dimension to drill into.

By adding the four different fields to the *Explain by* well, we have the option to drill into any of these dimensions. The decomposition tree also offers us the *High value* and *Low value* AI splits, indicated by the light bulb icon:

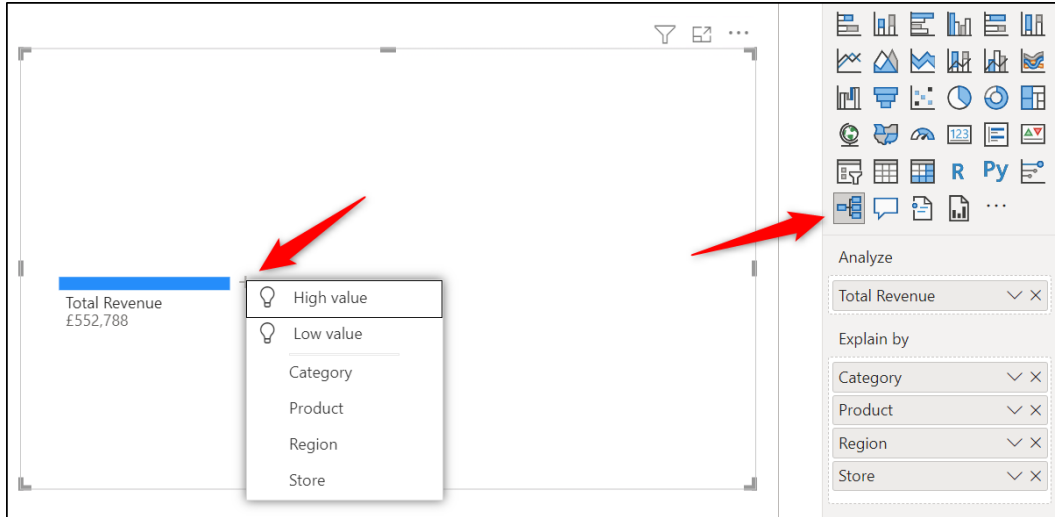


Figure 13.19: Decomposition tree used to explain total revenue

In figure 13.20, the **Region** and **Store** dimensions have been used to break down the total revenue. The **Store** dimension was selected from the + icon beside the **West** value. You can see the **West** is formatted in bold, and there are connector lines joining each store value to the **West**:

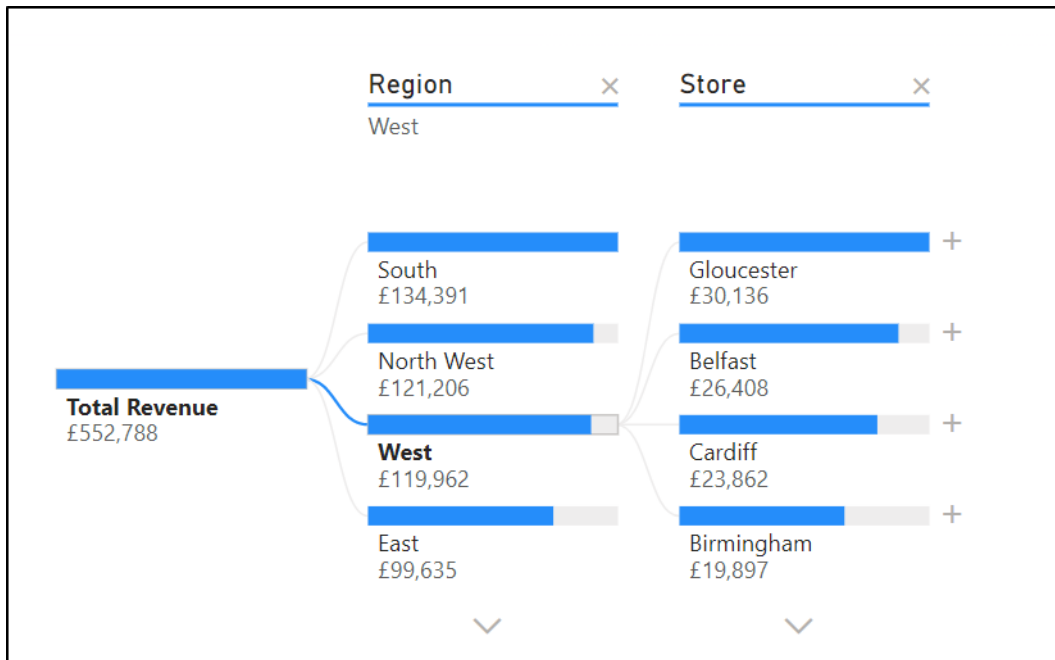


Figure 13.20: Total revenue explained by region and store

Clicking a different region will switch the analysis to that region, and the decomposition tree will list the stores in order of highest value first.

Clicking any of the stores will cross-filter or cross-highlight other visuals on the same page of the report (we will cover how to disable this in the upcoming chapter).

When formatting the decomposition tree visual, you can change the color of the bars and connector lines. Let us see how to do this.

1. In the *Visual* category of the formatting options, Click **Tree | Connectors** and change the color of the *Selected line* to a dark blue.
2. Click **Bars | Colors** and change the color *Positive bar* to a medium-strength orange, *Negative bar* to a dark orange, and the *Bar background* to a light blue.

Figure 13.21 shows the decomposition tree with the applied formatting changes:

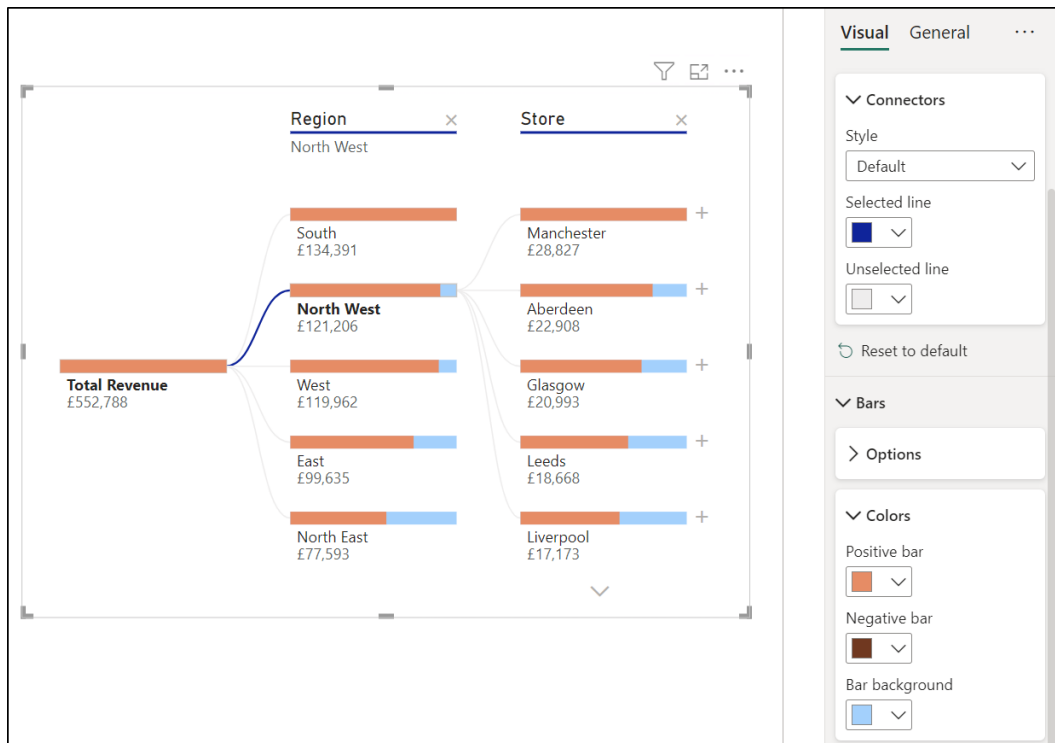


Figure 13.21: Formatting the connectors and bars of the decomposition tree

You can also make formatting changes to aspects of the decomposition tree, including the category labels and values. And as with all visuals, there are general options, including title, border, and header icons.

Smart narrative

The smart narrative visual is used to help tell the story of a specific visual or all visuals on a page. It is an incredible AI visual that considers the position, type, and data used by the different visuals and produces a narrative of the key insights.

It is used in a reverse way to the Q&A discussed earlier. Instead of asking a question (though this can be done) using natural language, you start with the visual.

Creating the smart narrative

Let us see how to create the smart narrative on a single visual and then one that considers all visuals on the page.

1. Click the **Front Page** tab to navigate to this page of the report.
2. Right-click on the stacked bar chart visual that shows total revenue by the product category and region and click **Summarize** (figure 13.22):

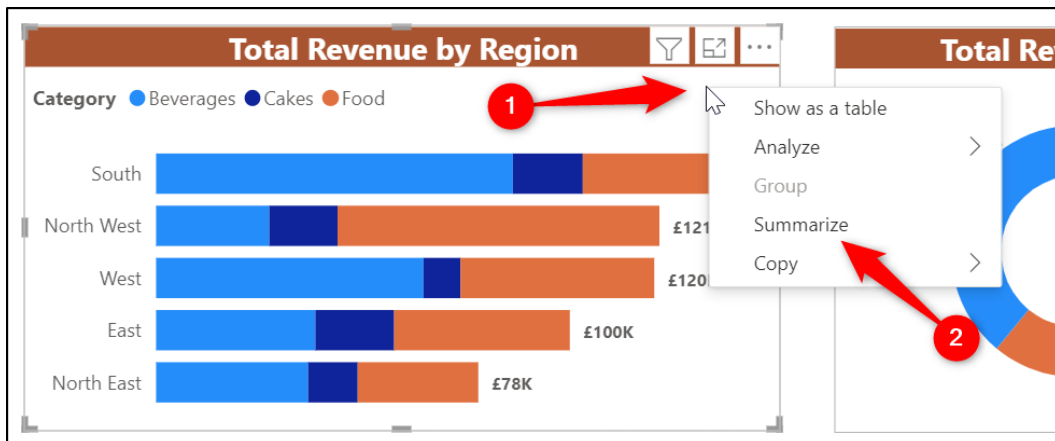


Figure 13.22: Summarize for a single visual

Figure 13.23 shows the smart narrative that is produced for this visual. All text with a blue underline is dynamic and will update when data is refreshed or filtered:

South in Category made up 15.54% of Total Revenue.

Beverages had the highest average Total Revenue at 50,520.18, followed by Food at 45,429.82 and Cakes at 14,607.62.

Figure 13.23: Smart narrative text for the stacked bar chart

The smart narrative updates, just like any other visual, when filters are applied, that affect it. In *figure 13.24*, the *Yes* slice of the **Total Revenue Split** pie chart is clicked. This cross-filters the smart narrative visual, and the narrative is updated to reflect this:

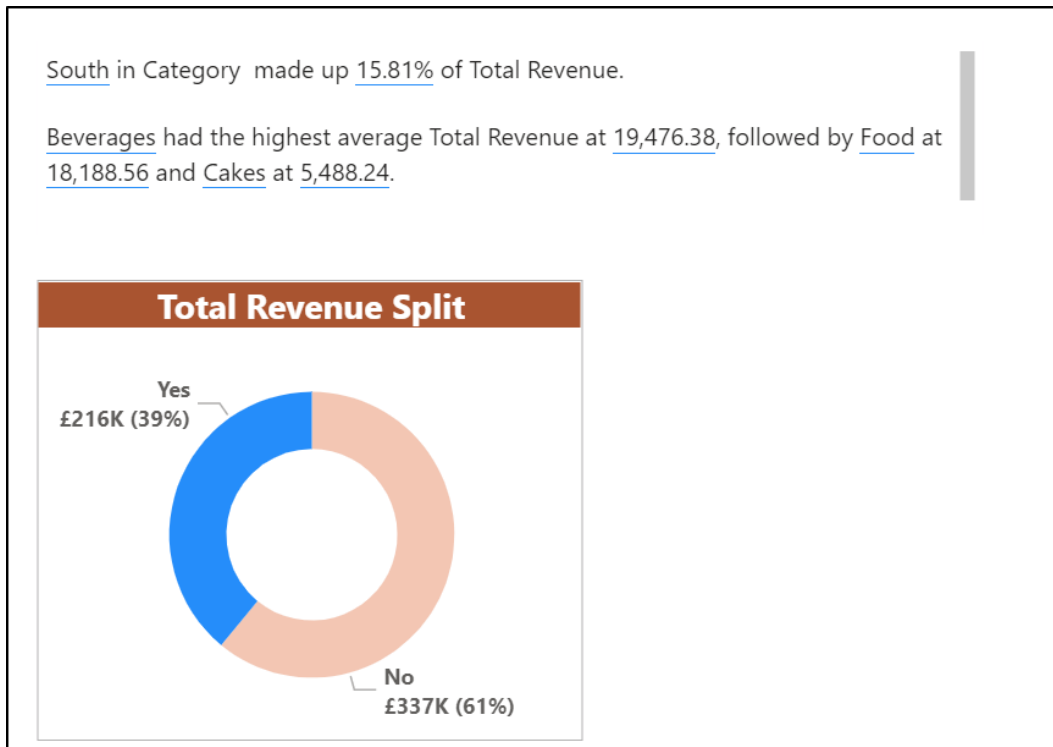


Figure 13.24: Smart narrative updating when filters are applied

To create a smart narrative for all visuals on a page, follow the following steps:

1. Ensure that none of the **Front Page** visuals are selected by clicking a blank area of the page.
2. Click the **Smart narrative** icon in the **Visualizations** pane (*figure 13.25*).
3. Move and resize the visual as necessary.

Figure 13.25 shows the smart narrative that is produced. It details the key insights from all visuals. You can see the narrative here that refers to the line chart, the column chart with the target line, and the last two paragraphs are a repeat of the previous single visual smart narrative.

At £82,727, Sep had the highest Total Revenue and was 380.39% higher than Nov, which had the lowest Total Revenue at £17,221.

Total Revenue and total Target are negatively correlated with each other.

Total Revenue and Target diverged the most when the Month Name was Sep, when Total Revenue were £37,727 higher than Target.

Across all 12 Month Name, Total Revenue ranged from £17,221 to £82,727.

South in Category made up 15.54% of Total Revenue.

Beverages had the highest average Total Revenue at 50,520.18, followed by Food at 45,429.82 and Cakes at 14,607.62.

The visualization pane on the right shows various chart types and a 'Smart Narrative' icon highlighted with a red arrow. Below the visualization pane, there are sections for 'Values', 'Drill through', 'Cross-report' (set to Off), and 'Keep all filters' (set to On).

Figure 13.25: Smart narrative from all visuals on the page

You can delete any text in the smart narrative in the same manner that you would delete the text of any text box. This allows you to refine the visual and to only show the insights that you are interested in.

In *figure 13.26*, the second paragraph has been deleted:

At £82,727, Sep had the highest Total Revenue and was 380.39% higher than Nov, which had the lowest Total Revenue at £17,221.

Total Revenue and Target diverged the most when the Month Name was Sep, when Total Revenue were £37,727 higher than Target.

Across all 12 Month Name, Total Revenue ranged from £17,221 to £82,727.

South in Category made up 15.54% of Total Revenue.

Beverages had the highest average Total Revenue at 50,520.18, followed by Food at 45,429.82 and Cakes at 14,607.62.

Figure 13.26: Second paragraph removed from the smart narrative

Adding and formatting dynamic values

All the narrative so far has been automatically produced by the smart narrative visual, but you can also ask your own questions, as we did with the Q&A, to generate the narrative that you want.

For this example, we will delete the penultimate paragraph that begins with “*South in Category ...*” and create our own version of that paragraph:

1. Click on the smart narrative visual and delete the penultimate paragraph.
2. Type the text “The”.
3. Click the **Value** button in the window that appears (figure 13.27).
4. Type the question “*which region has the highest total revenue*” into the **How would you calculate this value** box. Click **Save**:

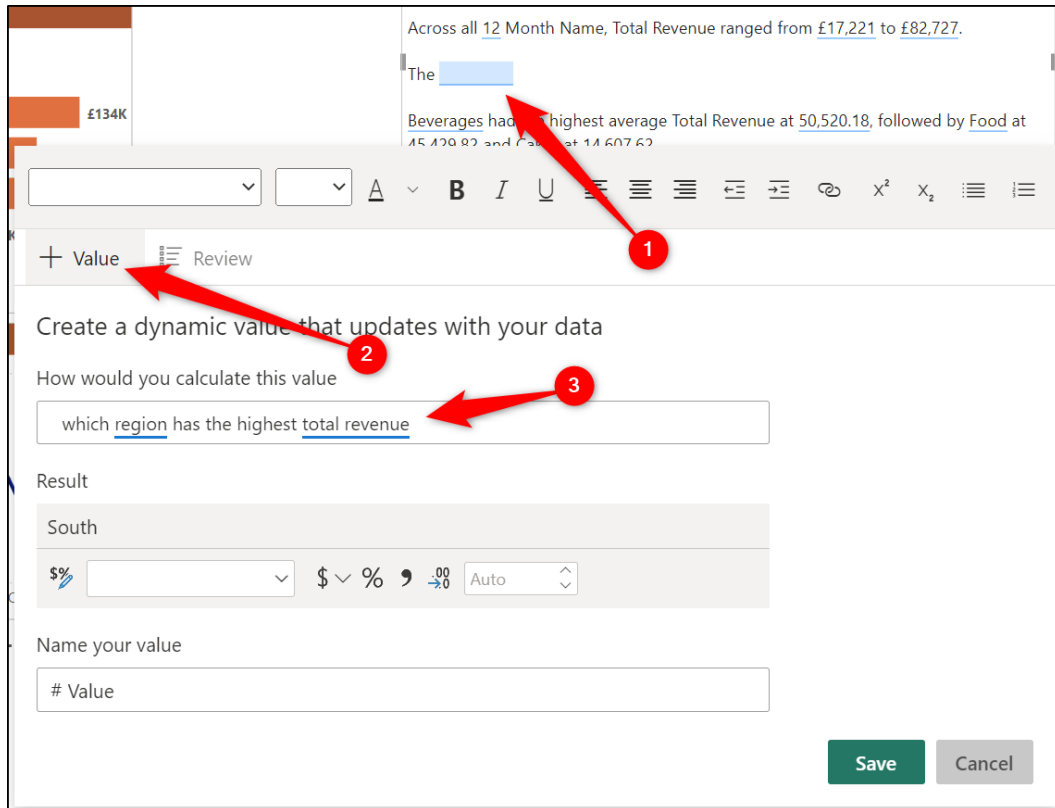


Figure 13.27: Creating your own dynamic text narrative

The name for the *South* region is added. This value is dynamic, and as data changes, the region name may change.

5. Continue typing with the text “has the highest total revenue at”.
6. Click the **Value** button to create another dynamic value.
7. Enter the question, “what is the total revenue for the top region by total revenue”. In *figure 13.28*, you can see a preview of the result of the question and its format too.
8. Click **Save**.

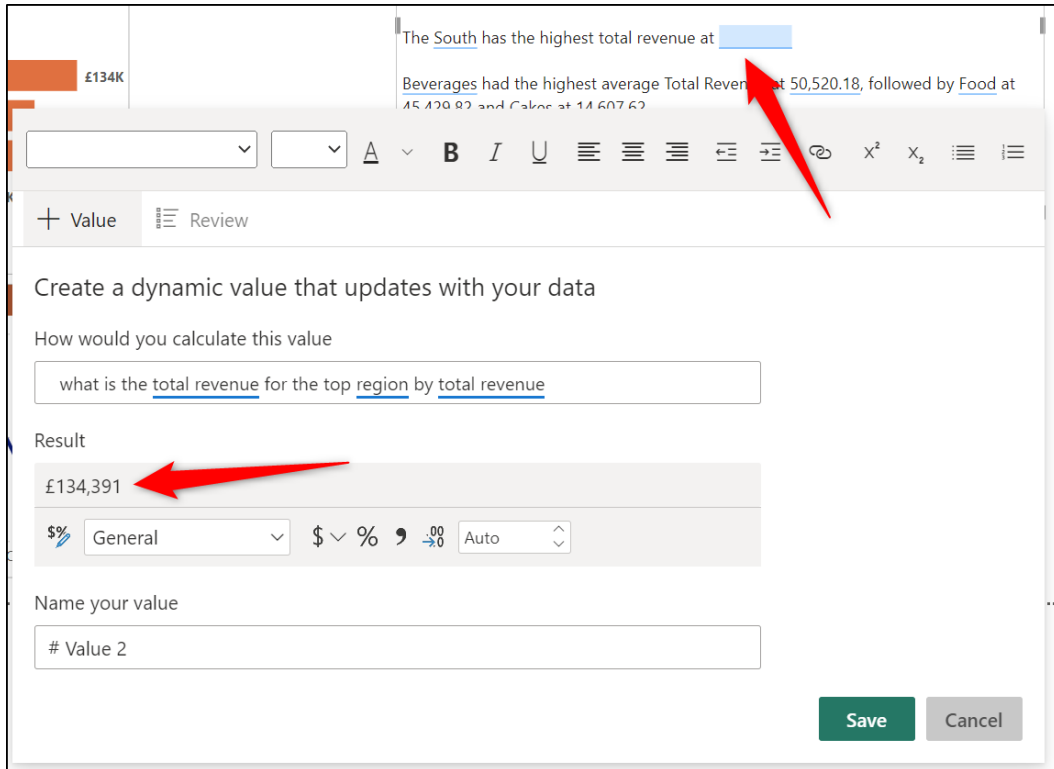


Figure 13.28: Adding the dynamic value for the top regions’ total revenue

Figure 13.29 shows the newly created paragraph in the smart narrative. The dynamic values that we added are blue underlined.

At £82,727, Sep had the highest Total Revenue and was 380.39% higher than Nov, which had the lowest Total Revenue at £17,221.

Total Revenue and Target diverged the most when the Month Name was Sep, when Total Revenue were £37,727 higher than Target.

Across all 12 Month Name, Total Revenue ranged from £17,221 to £82,727.

The South has the highest total revenue at £134,391.

Beverages had the highest average Total Revenue at 50,520.18, followed by Food at 45,429.82 and Cakes at 14,607.62.

Figure 13.29: Smart narrative with the newly created paragraph

In *figure 13.29*, you can see that the dynamic values in the last paragraph are not formatted in British Pounds Sterling like all other total revenue values in the smart narrative. Let us fix this inconsistency.

1. Click on the value for which you want to change the format.
2. In the window that appears, specify the formatting that you want to apply. In *figure 13.30*, the *Currency* format for *£ English (United Kingdom)* has been applied, and *0* decimal places have been specified.
3. Click **Save**.

The screenshot shows a formatting dialog box with a rich text editor toolbar at the top. Below the toolbar, there are tabs for '+ Value' and 'Review'. The main content area contains the following text:

This value was automatically generated while summarizing the report

Average Total Revenue for Category with max average Total Revenue

Result

£ 50,520

The value is currently displayed as '£ 50,520'. Below this, there is a formatting section with a currency symbol icon, a dropdown menu set to 'Currency', a '\$' symbol, a '%' symbol, a '0' in a box, and a decimal places icon. At the bottom, there is a text input field labeled 'Name your value' containing '# V5 (4)'. Two buttons, 'Save' and 'Cancel', are located at the bottom right.

Figure 13.30: Formatting dynamic values in the smart narrative

Figure 13.31 shows the completed smart narrative visual. All values in the final paragraph have the currency format applied:

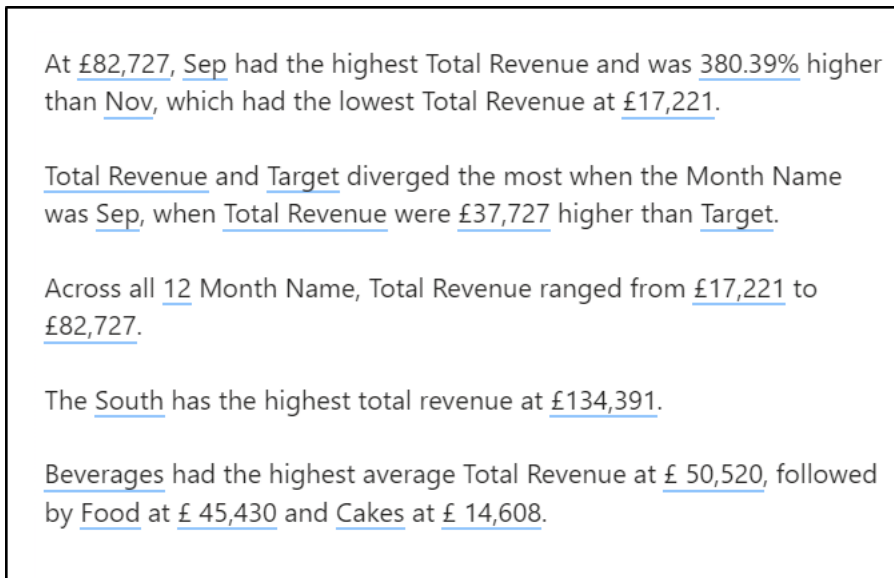


Figure 13.31: The completed smart narrative visual

Custom visuals in Power BI

In addition to the core visuals available in the Visualizations pane, you can also download visuals from Microsoft AppSource. The AppSource marketplace contains many more visuals for you to explore and utilize.

Adding a custom visual from AppSource

There are two main approaches to accessing the AppSource marketplace to download a visual.

One approach is to click the **Get more visuals** ellipsis icon at the end of the gallery of visuals in the **Visualizations** pane and click **Get more visuals** from the menu (figure 13.32):

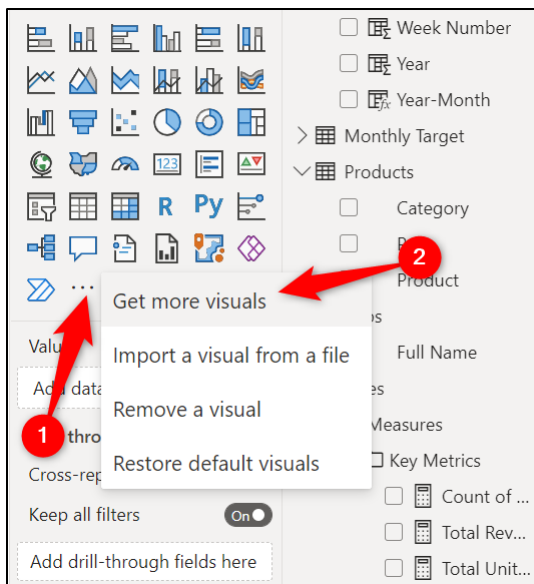


Figure 13.32: Get more visuals from the Visualizations pane

A second approach is to click **Insert | More visuals | From AppSource** (figure 13.33).

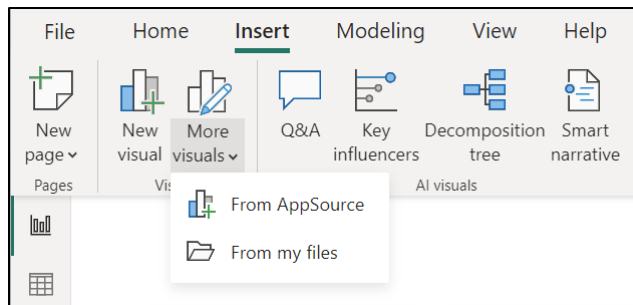


Figure 13.33: Get more visuals from the Insert tab

The Power BI visuals window appears and presents all visuals available for you to explore and find the one that fits your requirements (figure 13.34).

There is a *Filter by* list that can be used to filter for a visual by category, such as *KPI* or *Change Over Time*. There is also a *Search* bar that can be used to find a visual using keywords.

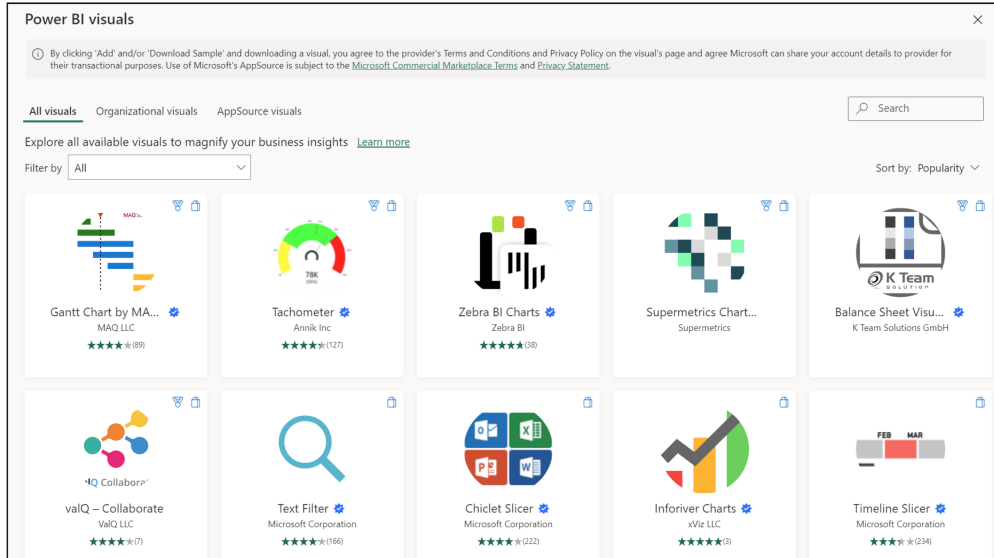


Figure 13.34: The Microsoft AppSource marketplace has many visuals

Click on a visual to view further information about it, including a description, images, key information such as when the app was last updated and a link for sample instructions (figure 13.35).

Click the **Add** button to import the custom visual into Power BI.

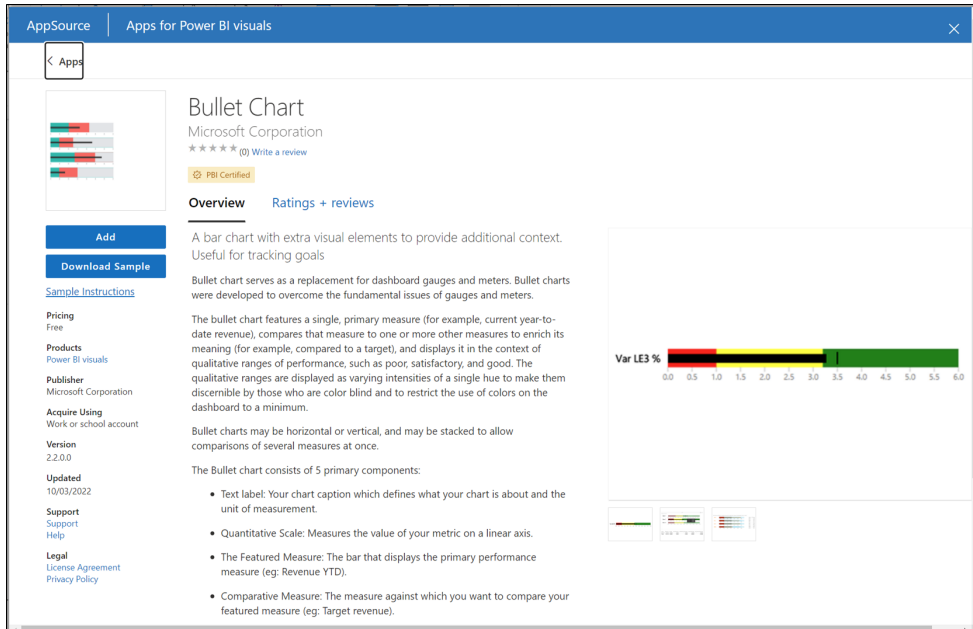


Figure 13.35: App information when adding to Power BI

The *Import successful* message appears to confirm that the custom visual has been added, as shown in *figure 13.36*. Click **OK**:



Figure 13.36: Import successful message

The added custom visual can be seen below the gallery of visuals in the **Visualizations** pane, as shown in *figure 13.37*. The visual can be used in the same manner as any of the core visuals of Power BI. Read the app instructions for specifics on its use.

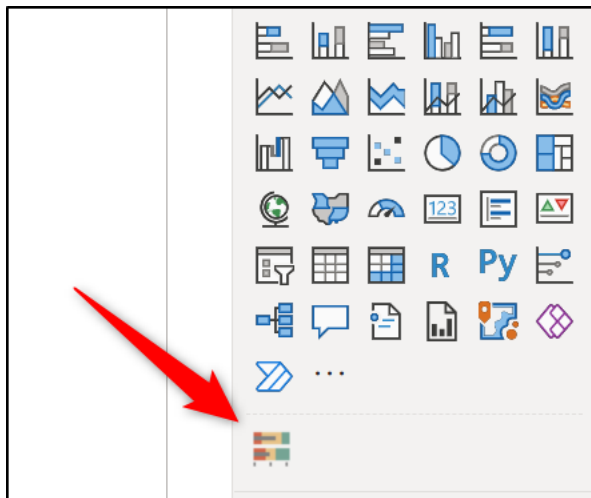


Figure 13.37: Custom visual added to the Visualizations pane

You can pin a custom visual so that it appears in the gallery of core visualizations instead of below. Simply right-click on the custom visual and click **Pin to visualizations pane** (*figure 13.38*):

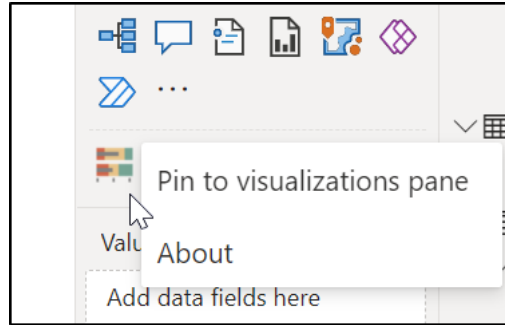


Figure 13.38: Pin a visual to the Visualizations pane

Removing a custom visual

If a custom visual is no longer required, it can be removed. Removing the visual removes all instances of it in the report.

Click the **Get more visuals** ellipsis icon at the end of the gallery of visuals in the **Visualizations** pane and click **Remove a visual**, as shown in *figure 13.39*:

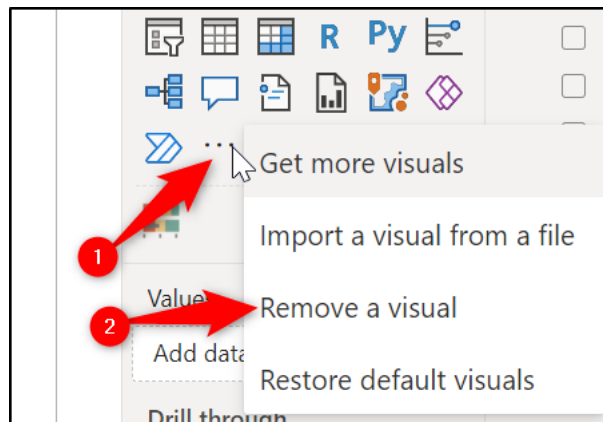


Figure 13.39: Remove a custom visual in Power BI

The **Select visuals to remove** window appears as shown in *figure 13.40*. All custom visuals are listed, and it states the number of instances that each visual is used in the report. This is referred to as the associated tiles.

Click on the visual to be removed and click **Remove**.

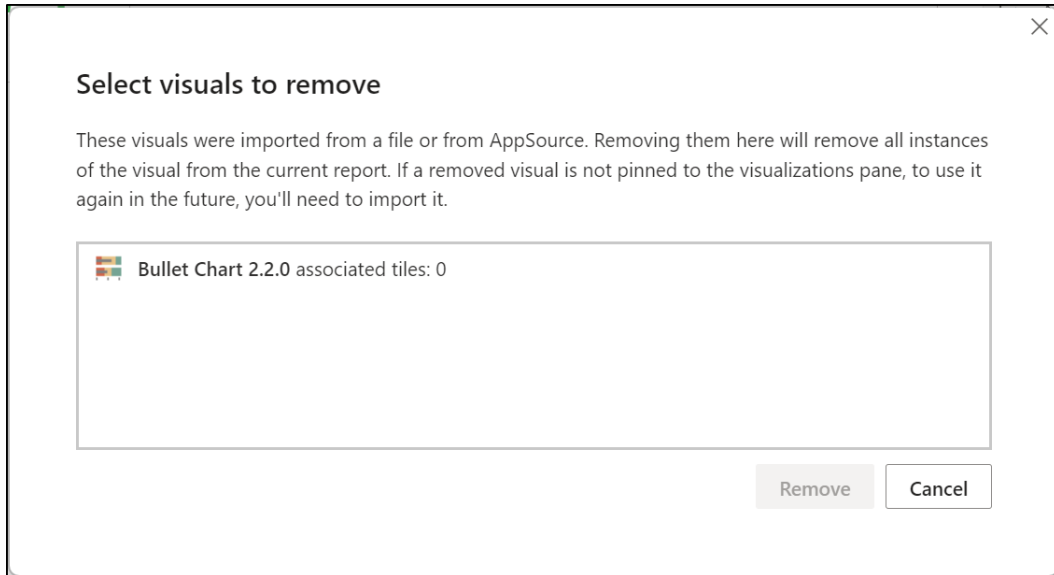


Figure 13.40: Selecting the visuals to remove

Conclusion

In this chapter, we learnt how to use AI visuals in Power BI. The first visual we looked at was the Q&A visual. This powerful visual allows users to find key insights by asking natural language questions of the data. We learnt how the Q&A can be configured for optimal use in a report.

We then learnt how to use the decomposition tree and smart narrative visuals. Two more fantastic AI visuals that really assist the user at finding insightful information with minimum effort.

In the upcoming chapter, we will look at some of the different options available to set report interactions and filters.

We start with the ability to cross-filter and cross-highlight visuals from another visual on a report page. We then dive into Slicers, possibly the most popular of the on-page filtering options. We will then explore the Filter pane, and finally, the drill through feature of Power BI is covered.

Questions

Here are some questions to test what you have learnt in this chapter.

1. Which of the following methods can be used to insert the Q&A visual?
 - a. Double-click a blank area of the page.
 - b. Click Insert > Q&A.
 - c. Click the Q&A icon in the Visualizations pane.
 - d. All of the above
2. Which of the following are features that can be used to optimize the Q&A visual?
 - a. Create suggested questions.
 - b. Add synonyms.
 - c. Review questions that users have asked.
 - d. All of the above
3. You can remove a custom visual from a report if it is no longer required.
 - a. True
 - b. False
4. The smart narrative can be created for a single visual or for all visuals on a page.
 - a. True
 - b. False

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



CHAPTER 14

Report Interactions, Filters, and Slicers

Introduction

There are many ways for a user to interact with and filter the visuals of a Power BI report. We have already seen small examples of filtering visuals at different times in this book. Let us get into more detail now on the more commonly available filtering features.

In this chapter, we will cover four different filter options—visual interactions, slicers, the Filters pane, and Drill through.

We will start with the interactions between visuals and how you can specify them. Then, we will learn how visual filters or highlights data in other visuals on the page. Then we will move onto slicers, the most common option for on-page filtering of visuals. We will cover different settings to customize the slicers as per our requirements.

Next is the Filters pane. This is a powerful option and is great for setting filters that will be engaged less frequently. Finally, we will see how to set up a Drill through the filter and modify it as per our needs.

Structure

In this chapter, we will cover the following topics:

- How to edit visual interactions on a page.
- Using slicers to filter page visuals.
- Filtering individual visuals and page visuals with the Filters pane.
- Creating a top *N* list using the Filters pane.
- Filtering visuals on all pages with slicers and the Filters pane.
- Setting up a Drill through the page and applying it to visuals.

Objectives

After reading this chapter, you will learn how to use different filter options available in Power BI. These are visual interactions, slicers, the Filters pane, and the Drill through feature. You will learn when the best time is to apply each one and understand the different qualities that each filter offers.

Editing visual interactions

Files: `sales-report-ch-14-start.pbix`

All visuals that have the capacity to do so interact with all other visuals on a page in Power BI by default. The default interaction, if possible, is to highlight the data in the other visual; otherwise, the data is filtered. This functionality is also known as cross-highlighting and cross-filtering.

Note: In *Chapter 3, Getting Started with Power BI Desktop*, we covered the options to change the default settings and disable cross-highlighting and cross-filtering for all visuals. We also saw an option to change the default interaction to cross-filtering.

Figure 14.1 shows a few of the visuals on the **Front Page** page of the sales report. No filters are currently applied.

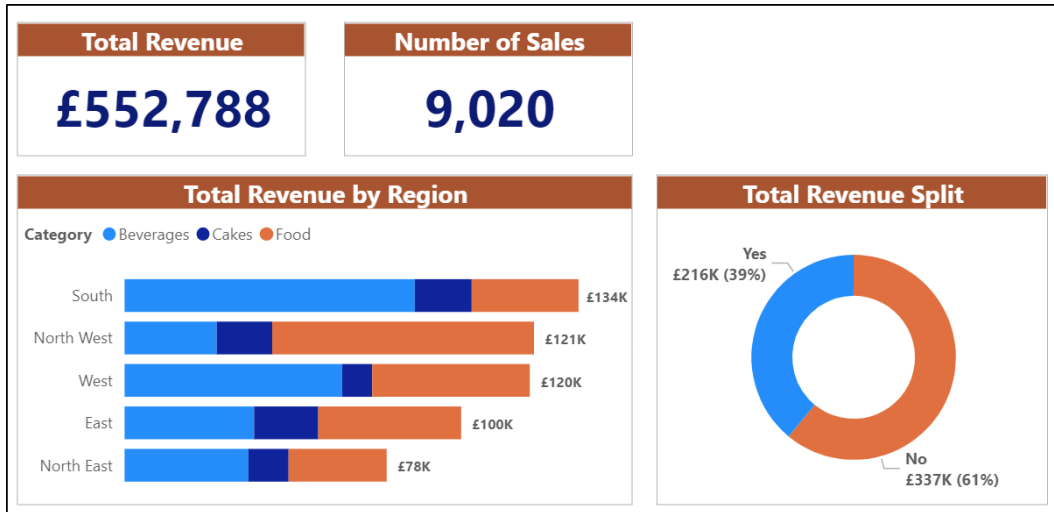


Figure 14.1: Visuals on the report without filters applied

When a data point in a visual is clicked unless specified otherwise, it will interact with all other visuals on the page.

In figure 14.2, the data point in the stacked bar chart representing the food category in the south region has been clicked. This has filtered the data in two card visuals and highlighted the data in the donut chart.

This is evident when you compare the results in figure 14.2 to figure 14.1. The callout values in the card visuals and the data labels in the donut chart are different.

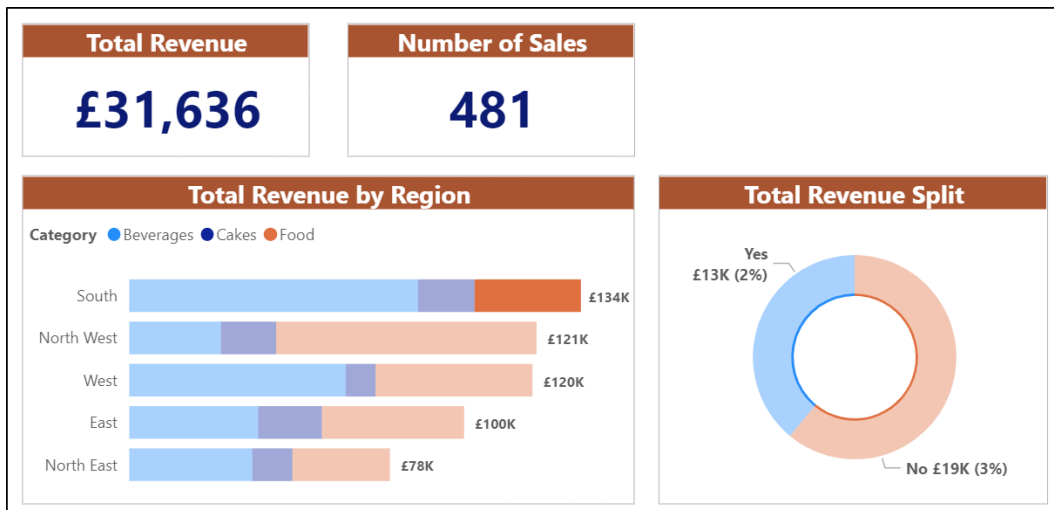


Figure 14.2: Visual data highlighted and filtered for the food category and south region

To edit the interactions from the stacked bar chart:

1. Click on the stacked bar chart to select it.
2. Click **Format** | **Edit interactions** (as shown in *figure 14.3*).

The *Edit interactions* button is lit to indicate that it is enabled, and icons are displayed in the top right corner of all other page visuals.

The card visuals only have two icons as they only have the options for *Filter* and *None*. While the donut has three icons—*Filter*, *Highlight* and *None* (as shown in *figure 14.3*):

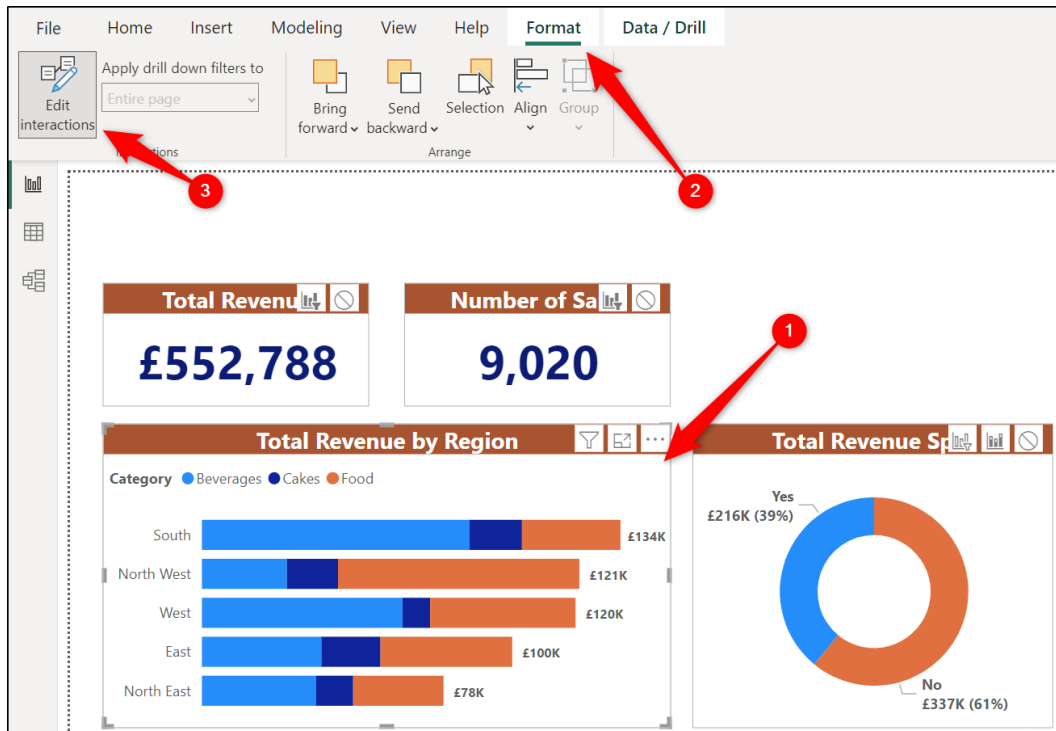


Figure 14.3: Edit interactions enabled on visuals from the stacked bar chart

Let us disable the interaction to the two card visuals by using the *None* option and change the *Highlight* interaction on the donut chart to *Filter*.

1. Click the **None** icon in the top right corner of each of the card visuals.
2. Click the **Filter** icon in the top right corner of the donut chart (*figure 14.4*).

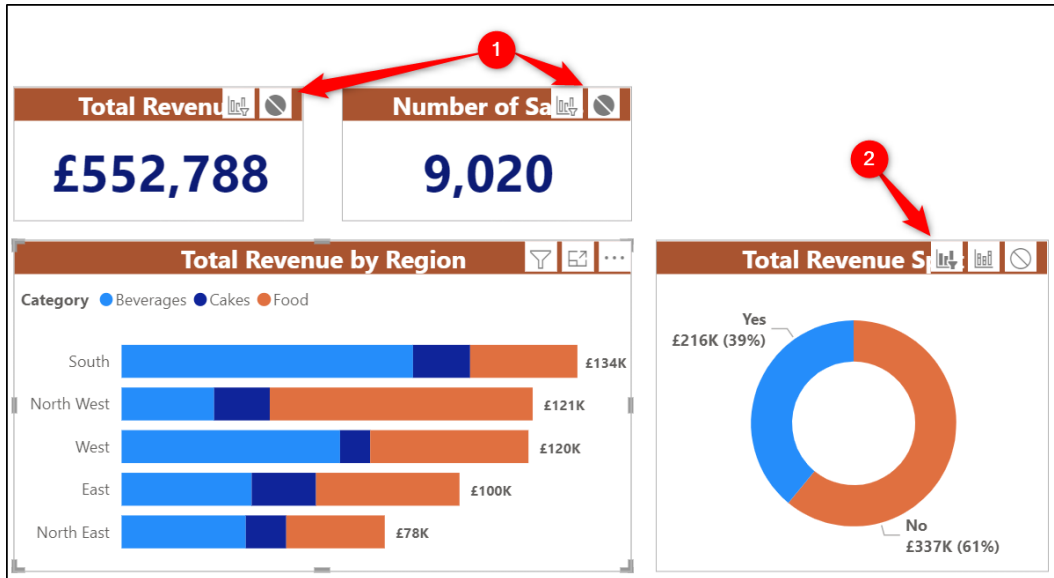


Figure 14.4 Interactions changed for the stacked bar chart

Now, when the data point in the stacked bar chart representing the food category in the south region is clicked, only the donut chart is filtered. The two cards are not affected (as shown in figure 14.5).

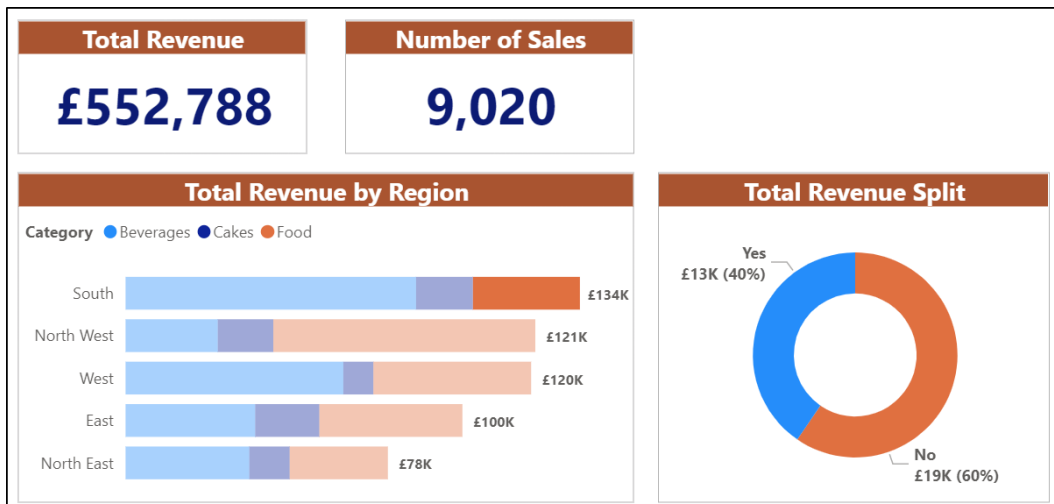


Figure 14.5: Only the donut chart is filtered

You can proceed to edit the interactions from other visuals on the page. Simply click another visual and then change the cross-highlight and cross-filter options on the other visuals.

Not all visuals can cross-highlight and cross-filter; for example, the two card visuals and the smart narrative visual on this page do not have that capacity.

The *Edit interactions* button can be switched off when you have finished editing the interactions of the visuals in the report. However, it could also be left on to save having to switch it on again the next time you need to edit interactions.

It is important to remember this default behavior of Power BI visuals and to get it right when building reports. Reports are used to communicate data. If users misread a report, it can lead to bad decisions. It is important that a user understands if and how data has been filtered and how they can use this powerful feature effectively.

Let us see an example where not setting visual interactions correctly can cause other visuals to break entirely.

We will copy the line chart from the **Front Page** page to the **Monthly Sales Analysis** page of the report. Unfortunately, the line chart will not be operational.

Figure 14.6 shows the broken line chart on the **Monthly Sales Analysis** page. This is caused by the slicer. The slicer is filtering the line chart data to a specific month resulting in a single data point being shown. Not good!

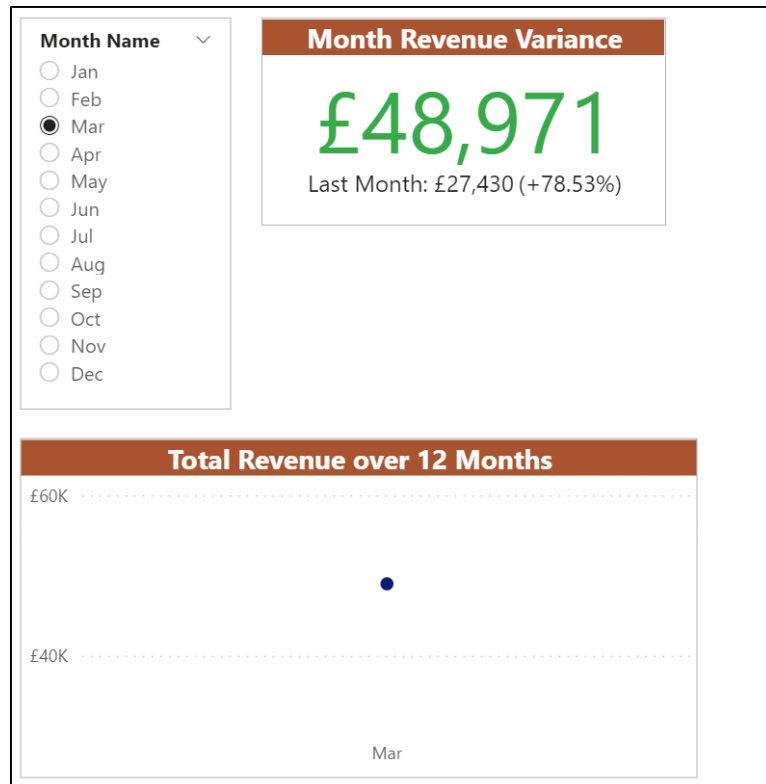


Figure 14.6: Broken line chart due to filter from the slicer

Let us disable the interaction from the slicer to the line chart.

1. Click on the slicer to select the visual.
2. Click **Format** | **Edit interactions**, if necessary (you may still have it enabled from the previous example).
3. Click the **None** icon in the top right corner of the line chart (*figure 14.7*).

The line chart now correctly shows the total revenue for all 12 months:



Figure 14.7: Disabling the interaction from slicer to line chart

However, the problems are not all solved yet. If a user clicks a data point in the line chart, the KPI visual breaks (*figure 14.8*). This is because both the slicer and line chart are applying separate month filters. We need to disable the filter from the line chart to the KPI.

Note: There is also a treemap visual on the page that breaks as a consequence of the line chart cross-filtering its data.

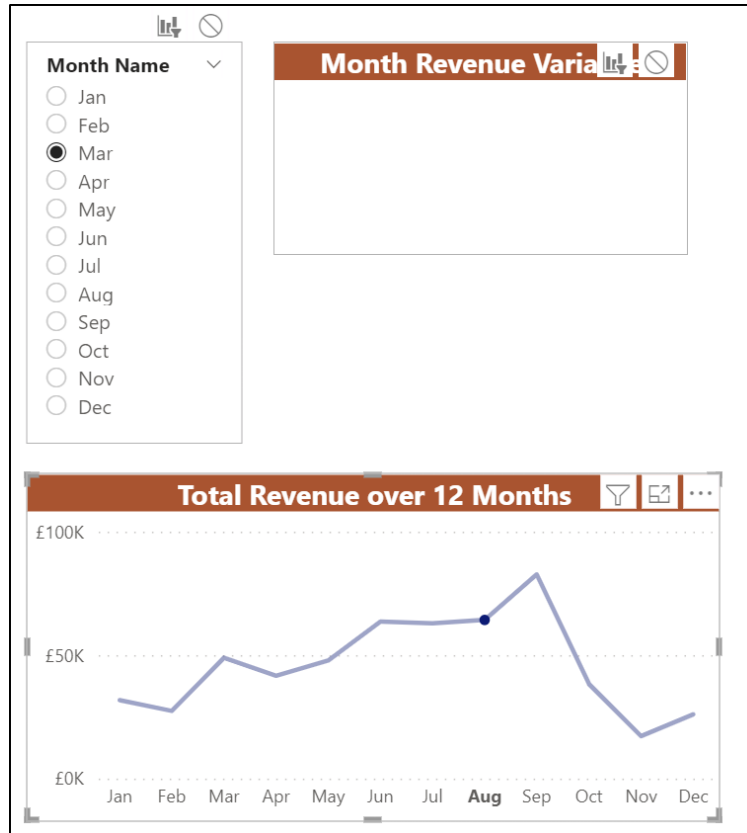


Figure 14.8: KPI visual broken by the Aug filter coming from the line chart

Following are the steps to disable the line chart filter:

1. Click the line chart visual to select it.
2. Click the **None** icon for the KPI (as shown in *figure 14.9*).

So, setting visual interactions correctly is imperative for an effective report. In the first example, the interactions from the stacked bar chart were not desirable and only dangerous if the user did not understand what they are looking at and how it can be used. In the second example, not having the interactions right caused visuals to break. Please refer to the following figure:

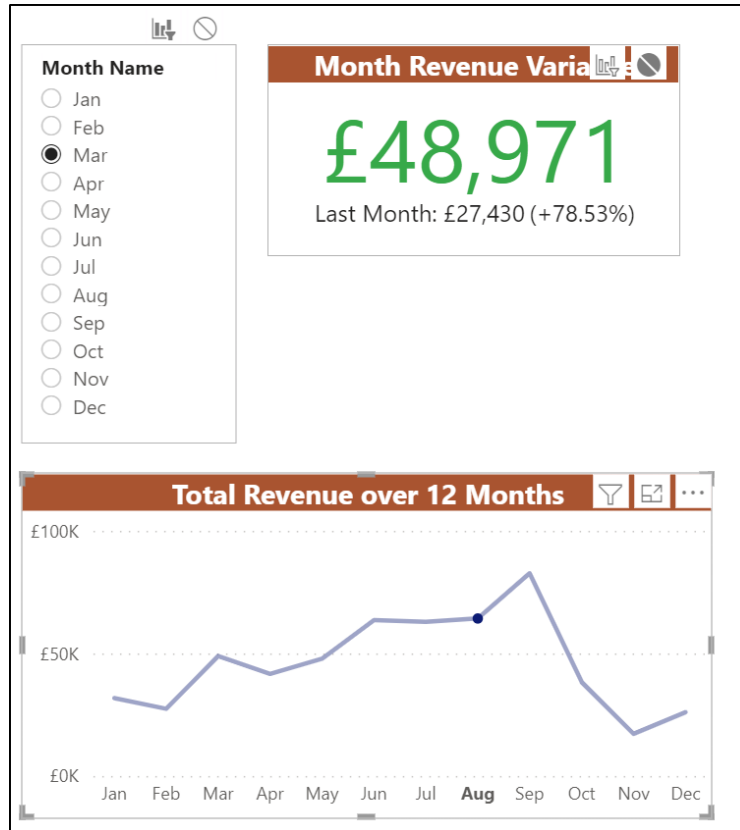


Figure 14.9: KPI is working now; the line chart filter is disabled

Slicers

Slicers are probably the most common option to provide filters for users to apply as they choose. In contrast to the visual interactions, slicers are easy to use and clear for a report user to identify and see which filters are applied.

By default, slicers filter all visuals on a page, though this can be changed by editing the interactions as we did in the last example.

We will see two slicer examples—one to filter dates and another to filter text values. We will customize each slicer to fit the needs of the report.

Using date fields in a slicer

For the first slicer, we want the ability to filter the visuals on the **Front Page** page for specific date ranges. So, we will insert a slicer that uses the **Date** field from the **Calendar** table.

1. Switch to the **Front Page** page and click a blank area on the canvas to ensure that no visuals are selected.
2. Click the **Slicer** icon in the **Visualizations** pane (figure 14.10).
3. Click and drag the **Date** field from the **Calendar** table into the **Field** well of the slicer.

Figure 14.10 shows the slicer added to the **Front Page** page. When using date fields, the default slicer type is a slider that allows both ends of the date range to be adjusted. This slicer type is known as *Between*. We will see how this could be changed shortly. Please refer to the following figure:

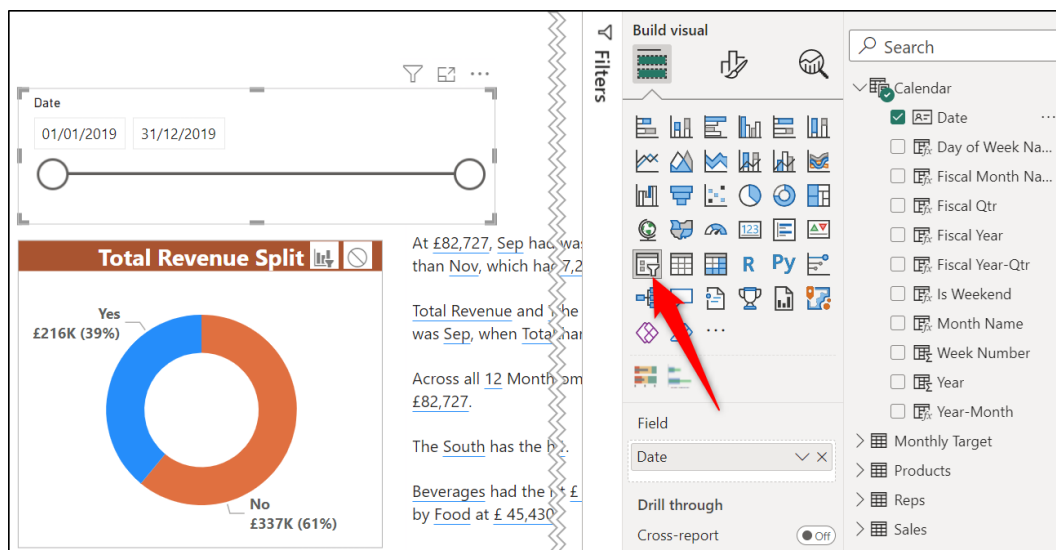


Figure 14.10: Inserting a slicer that uses a date field

There are the following three main methods to adjust a date at either end of the date range:

1. Click and drag the circle on the end of the slider.
2. Click in the start or end date box and change the date using the calendar picker (as shown in figure 14.11).
3. Type a date into the start and end boxes provided.

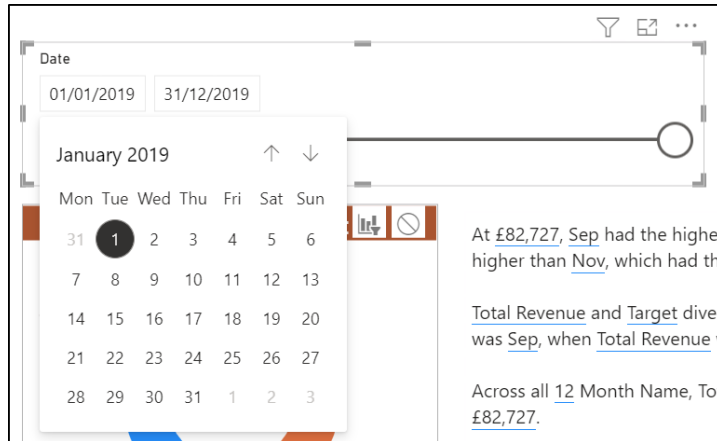


Figure 14.11: Changing the start date using the calendar picker method

The slider provides the fastest method of adjusting the date, but it is not accurate for when picking a specific date of interest.

When clicking on the start or end date boxes, the calendar picker immediately appears. If you prefer to type the date, click away from the calendar to hide it, and then you can edit the date in the box.

In figure 14.12, a filter is applied to show data only for the date range of July 1, 2019 to December 31, 2019.

To quickly clear a filter from the slicer, click the **Clear selections** icon (eraser image) in the top right corner (as shown in figure 14.12). It only appears when your mouse is positioned in the slicer header. Please refer to the following figure:

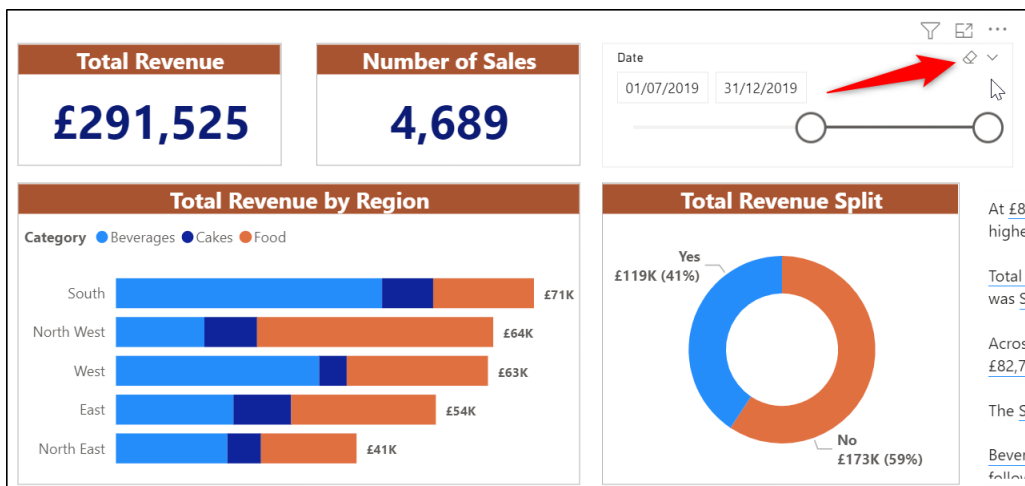


Figure 14.12: Filtering all visuals for a six-month date range

In addition to the *Between* slicer type, there are seven more types of slicer that can be used when working with date fields.

To change the slicer type follow the following steps:

1. With the slicer selected, click the **Format your visual** button to switch to the formatting options.
2. In the **Visual** section, click **Slicer settings** to expand the options in that category.
3. Click the **Style** list in the *Options* section and choose the one you want from the list provided (as shown in *figure 14.13*):

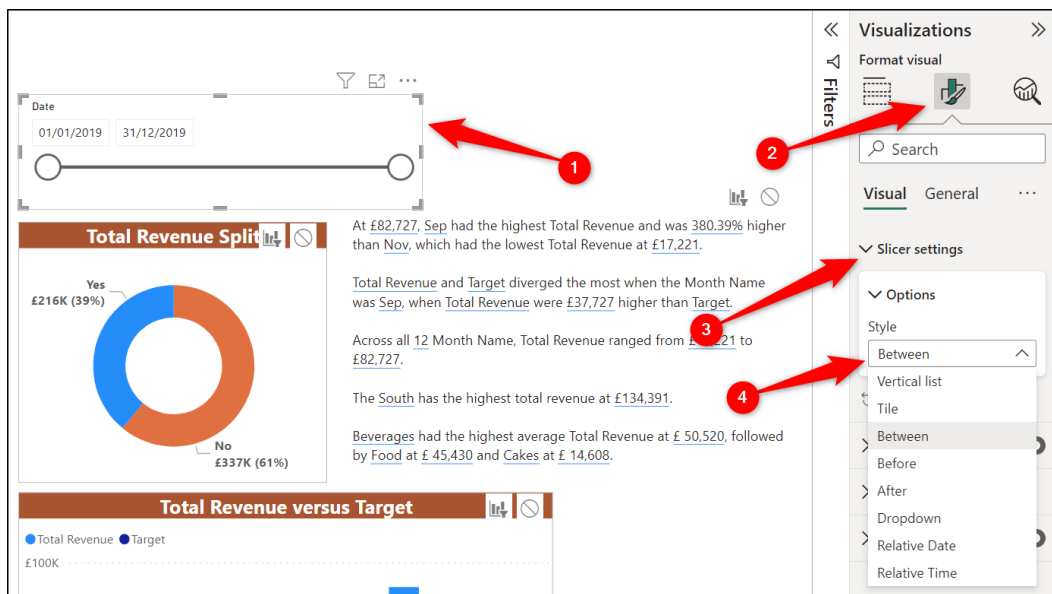


Figure 14.13: Types of slicer when using date fields

The **Vertical list** and **Dropdown** options are self-explanatory, and we will see them shortly when using text fields in a slicer. Let us look at a couple of more interesting-sounding slicer types.

Figure 14.14 shows the *After* slicer type being used to filter the visuals for data after September 2, 2019 (the first Monday of that month). The start date can be adjusted, but the end date is fixed on the last date in the range. Please refer to the following figure:

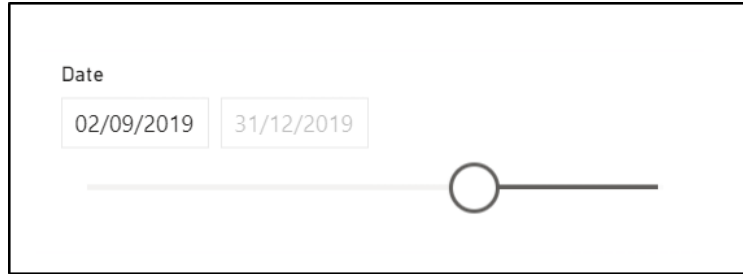


Figure 14.14: After type slicer

As you can imagine, the *Before* slicer type is the reverse. The end date can be adjusted, and the start date is fixed to the first date in the range.

Figure 14.15 shows the *Relative Date* slicer type being used to filter visuals for the last six weeks data only. October 23, 2022 shown in the slicer is today's date:

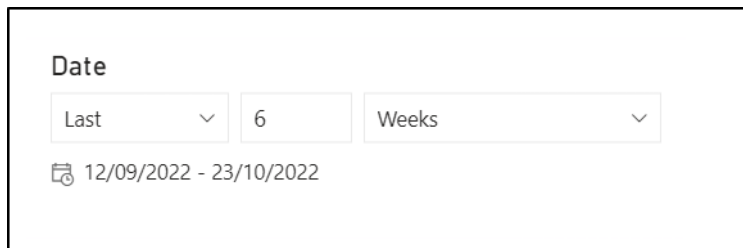


Figure 14.15: Filtering visuals for the last six weeks data

In our report, this shows no data as we are working with sales data from 2019 only. However, working with relative dates is very useful when producing live (or close-to-live) reports.

You can set filters for the *Last* x interval of time or the *Next* x amount of time. The intervals available include days, weeks, months, and years. And you can choose from the interval ending on today's date, as in figure 14.15, or use the calendar. For example, the last one month would be the date range from September 24, 2022 to October 23, 2022, and the last calendar month would be September 1, 2022 to September 30, 2022.

Let us now see some of the formatting options available for slicer.

We will remove the slicer header as it is not required. The slicer header shows the header text and the *Clear selections* button. The *Clear selections* button is useful and a great reason to keep the slicer header, but in this example, it is easy to drag the slider ends.

We will also change the color of the slider to match the dark blue color used elsewhere on the page.

1. Click on the slicer and click the **Format your visual** button in the **Visualizations** pane.
2. In the **Visual** section, Click the *On/Off* slider for the **Slicer header** to **Off**.
3. Click on the **Slider** section to expand the options and choose a dark blue color from the *Color* list (as shown in *figure 14.16*).

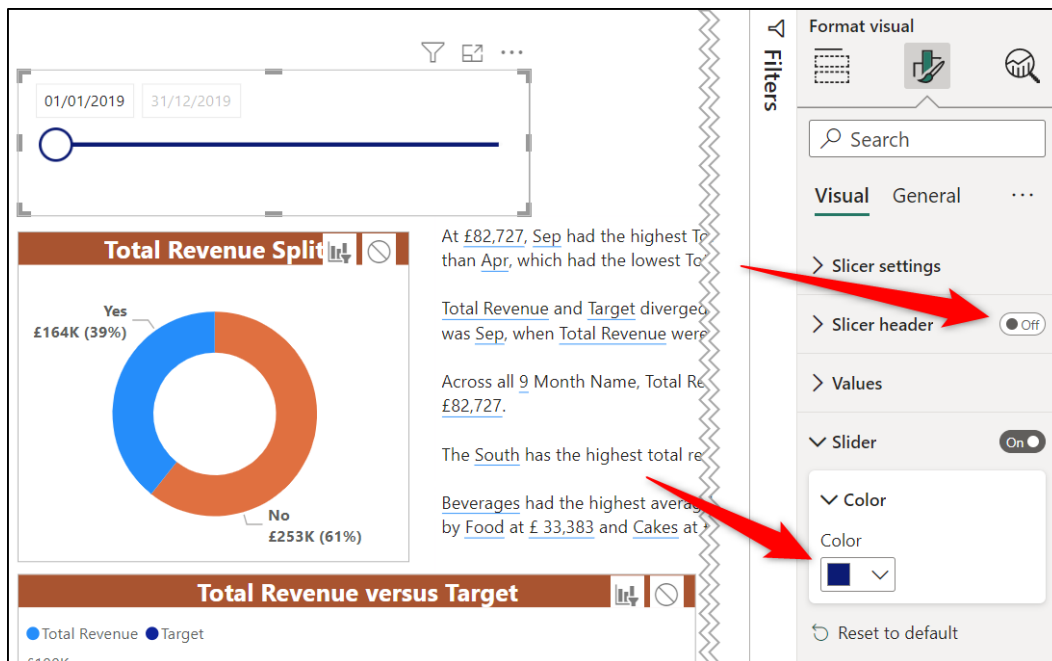


Figure 14.16: Formatting the slider and removing the slicer header

The usual *General* formatting options are available just like with any other visual. A visual border has been added to the slicer.

Using text fields in a slicer

Let us now look at using text values in a slicer. Switch to the **Regional Analysis** page of the report, and for this example, we will add a slicer to filter all visuals on the page by a specified region.

1. Click on the **Slicer** icon in the **Visualizations** pane.
2. Click and drag the **Region** field from the **Stores** table into the *Field* well.

When using text values, the default slicer type is a vertical list. The user can select multiple items in the list by holding the *Ctrl* key as they click. This is shown in *figure 14.17*.

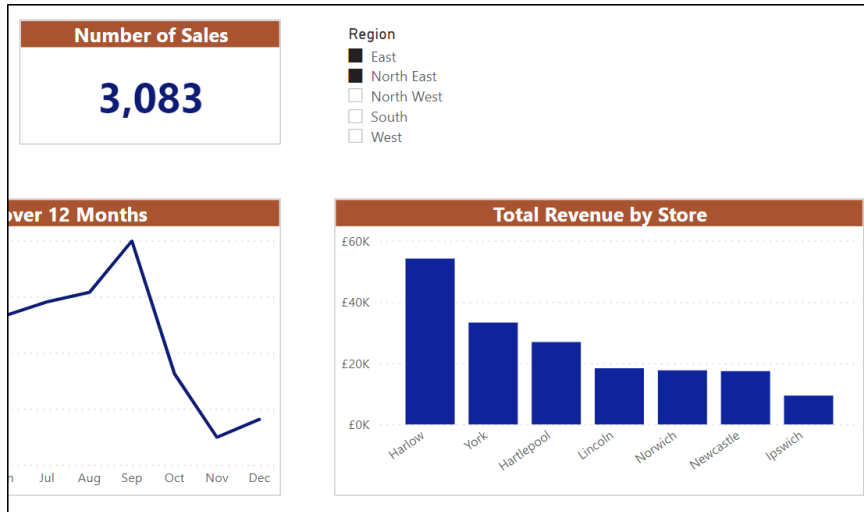


Figure 14.17: Vertical list slicer with multiple items selected

When there are only five items, a vertical list works nice. There are, however, three slicer types when working with text values—*Vertical list*, *Tile*, and *Dropdown*.

To switch to a dropdown slicer follow the following steps:

1. With the slicer selected, click the **Format your visual** button in the *Visualizations* pane to switch to the formatting options.
2. In the *Visual* section, click **Slicer settings** to expand the options in that category.
3. Click the **Style** list in the **Options** section and click **Dropdown** (as shown in figure 14.18).

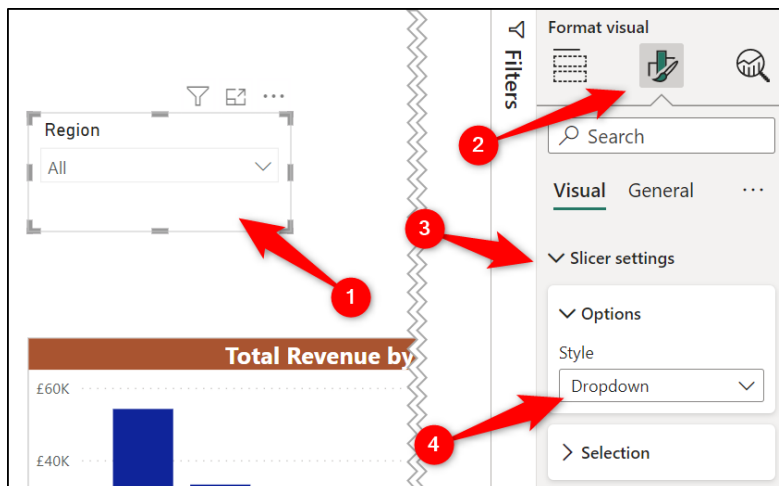


Figure 14.18: Changing the type of slicer

The dropdown slicer also allows multiple selections by default on holding the *Ctrl* key. Because of this, when you click an item in a dropdown slicer, the list remains expanded. You will need to click elsewhere to collapse the dropdown list.

The dropdown slicer displays the text *Multiple selections*, which does not tell the user which items have been selected (figure 14.19). So, the vertical list type works better when using multiple selections.

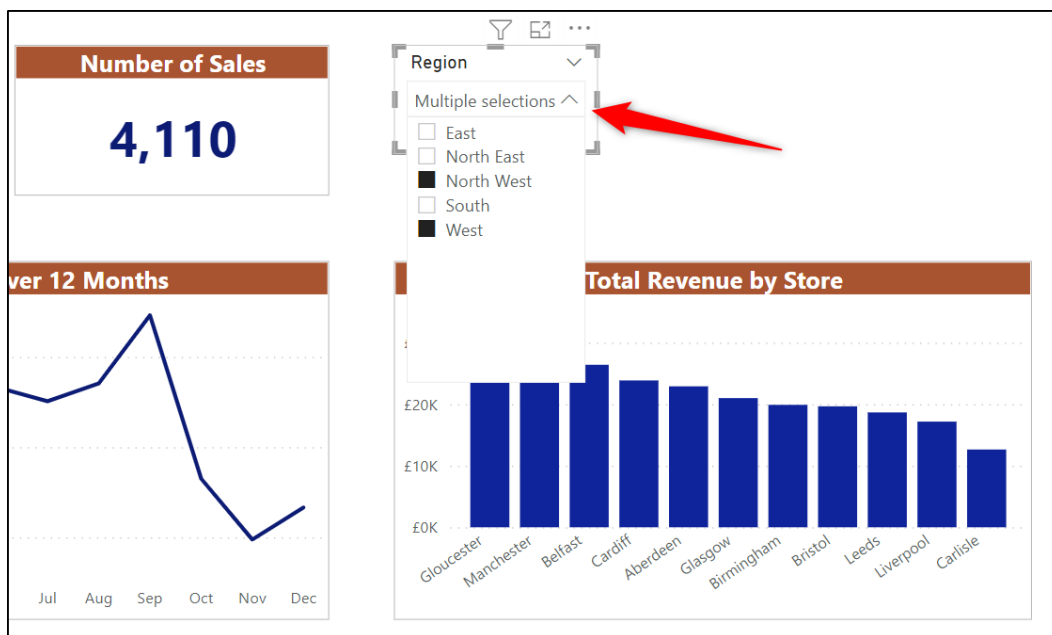


Figure 14.19: Multiple selections in a dropdown slicer

You can change the slicer settings to make the slicer work the way that you want. This includes disabling the requirement of the *Ctrl* key to make multiple selections and making a slicer single select only.

In this example, we will change the slicer from dropdown back to the list and then turn on the single select option.

1. From the formatting options in the **Visualizations** pane, click the **Vertical list** option in the **Style** list of the **Slicer settings** category.
2. Click the **Selection** sub-category to expand the formatting options.
3. Click the **Single select** *On/Off* toggle to **On** (figure 14.20).

The slicer now only allows the single selection of a region. The *Ctrl* key cannot be used to select multiples, and there is no *Clear selections* icon to show all regions. There must be one, and only one, region selected at any given time.

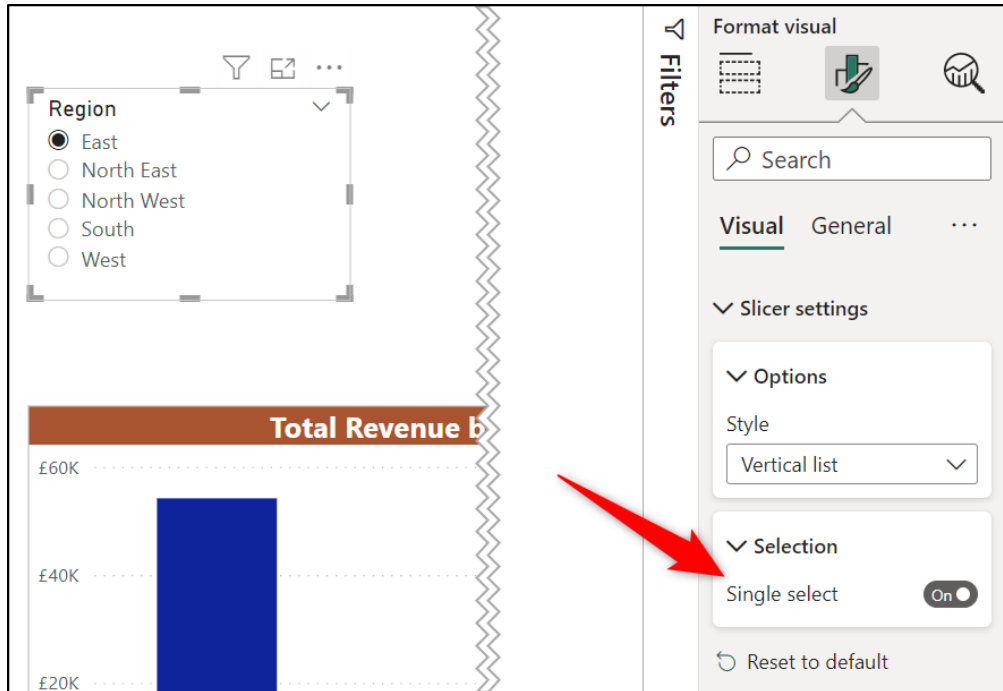


Figure 14.20: Single select vertical list slicer

Finally, the slicer can be changed from a vertical list to a tile slicer. When displayed as a tile, the slicer items are shown as large buttons in a horizontal orientation instead of a vertical list of checkboxes or option buttons.

Let us make this change. We will also remove the slicer header that currently displays “**Region**” and change the color of the slicer buttons to orange (or whichever color you want).

1. From the same **Slicer settings** category of the formatting options, click the **Options** sub-category if necessary, and change the *Style* to **Tile** (figure 14.21).
2. Click the **Slicer header On/Off** toggle to **Off**.

3. Click the **Values** category, then **Background**, click the **Color** list and choose a color. You can also change the font, color, and style of the values.

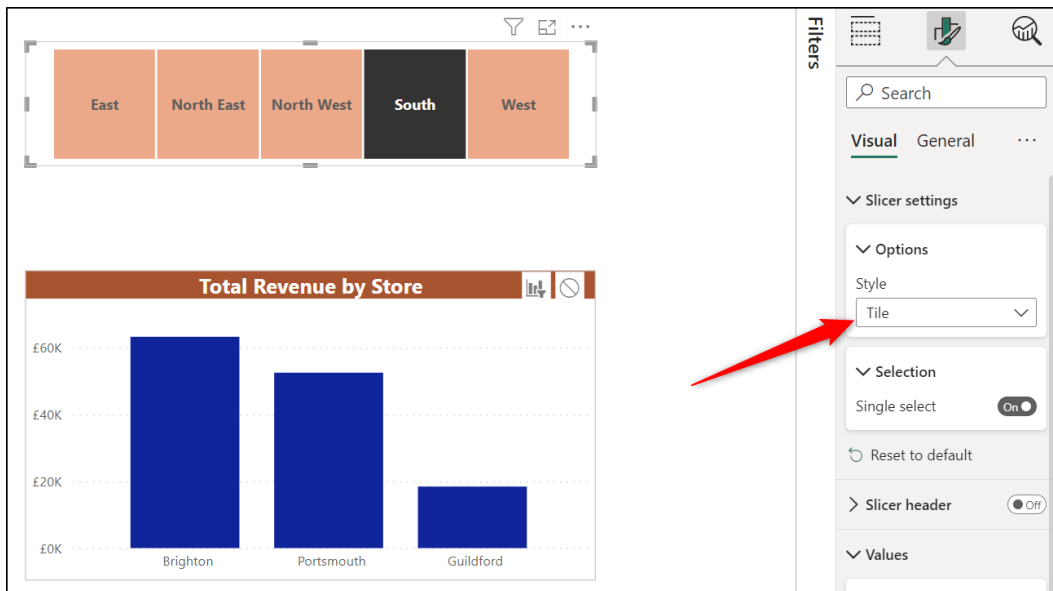


Figure 14.21: Tile slicer with the South region selected

Filtering visuals on other pages

By default, slicers filter all visuals on the page. We saw in the *Editing Visual Interactions* section of this chapter how to disable the interaction from a slicer to specific visuals. However, you can also extend the reach of a slicer to filter visuals on other pages of a report.

For this example, let us enable the date slicer on the **Front Page** page to filter the visuals on the **Maps** page also.

1. From the report view, click on the date slicer.
2. Click **View | Sync slicers** to open the **Sync slicers** pane on the right of the window.
3. Check the sync box for the **Front Page** and **Maps** pages (*figure 14.22*).

When the sync checkbox is checked, the visible box is also checked in the *Sync slicers* pane. This ensures that the slicer also appears on the **Maps** page, making it easy for the user to know that the visuals are filtered by date.

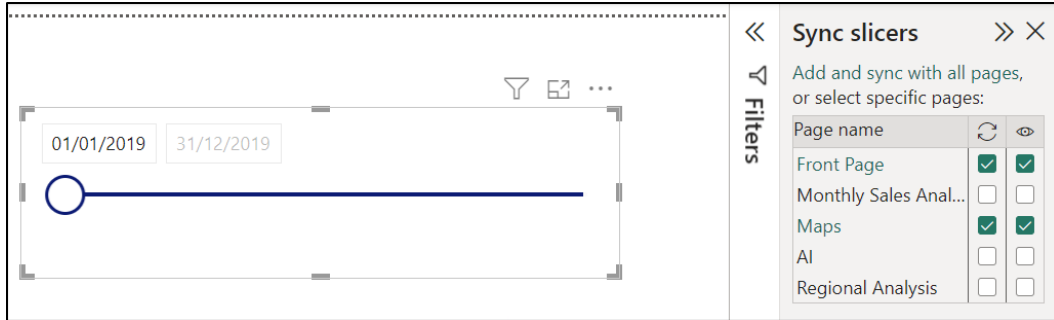


Figure 14.22: Sync slicers on specific pages

You could choose not to make the slicer visible on the **Maps** page too. The visuals on the **Maps** page would still be filtered, and there would just be no slicer to illustrate this clearly.

One option to see what filters may be applied to any visual on a page is to position the pointer arrow over the **Filters on Visual** button in the top right of a visual. *Figure 14.23* shows this feature is used on the map chart of the **Maps** page. It informs us that there is a filter affecting the visual to only show values before 22/11/2019. Please refer to the following figure:

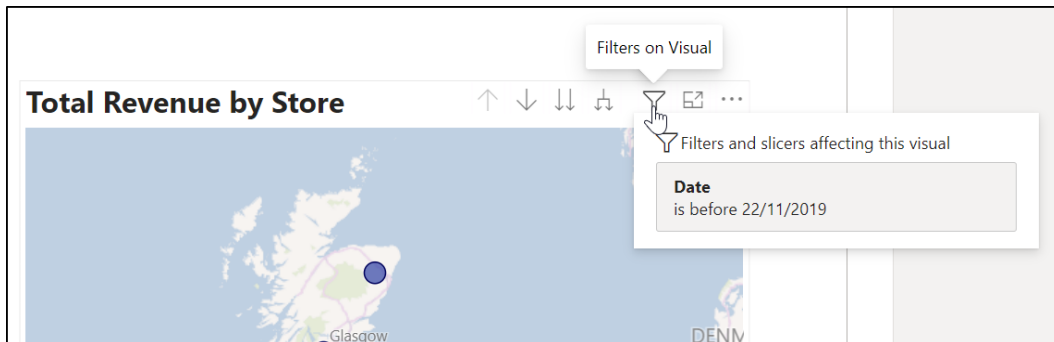


Figure 14.23: Identifying the filters affecting a visual

The Filters Pane

Another method to filter specific visuals, all visuals on a page, or even all visuals on all pages of a report, is the Filters pane.

The Filters pane is tucked away on the right side of the window. It is collapsed by default so that it does not distract users from the report.

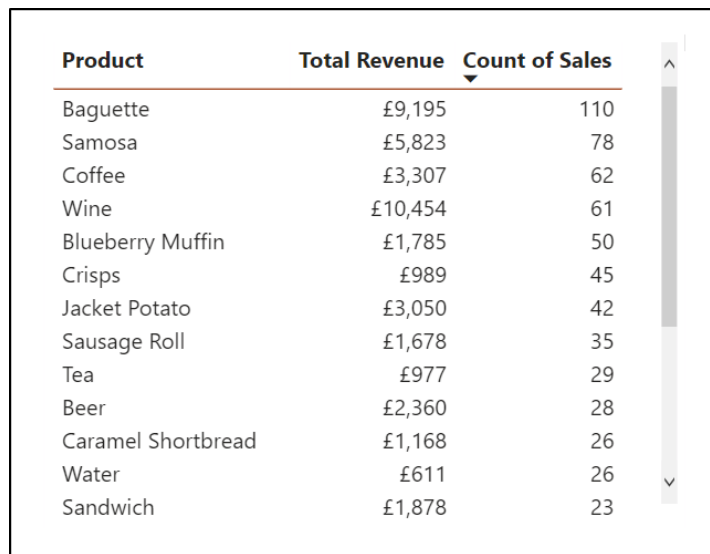
Because it is not prominent on a report page, in the way that a slicer is, for example, this powerful filter is best used when setting filters that are not engaged with

regularly by a user. A filter that, when set, is probably not changed. While a slicer is fantastic for regular user interaction.

Setting a Top N filter

Let us use the Filters pane to set a Top N filter on a specific visual.

Figure 14.24 shows a table visual that has been added to the **Monthly Sales Analysis** page. It lists the products along with their total revenue and count of sales. The products are ordered in descending order by the count of sales column. Please refer to the following figure:



Product	Total Revenue	Count of Sales
Baguette	£9,195	110
Samosa	£5,823	78
Coffee	£3,307	62
Wine	£10,454	61
Blueberry Muffin	£1,785	50
Crisps	£989	45
Jacket Potato	£3,050	42
Sausage Roll	£1,678	35
Tea	£977	29
Beer	£2,360	28
Caramel Shortbread	£1,168	26
Water	£611	26
Sandwich	£1,878	23

Figure 14.24: Table of products ordered by the count of sales

We will use the Filters pane to filter this table to show the Top 10 products only.

1. Click on the table visual.
2. In the *Filters on this visual* section of the **Filters** pane, click the small arrow to expand the **Product** filters card (figure 14.25).
3. Click the **Filter type** list arrow and click **Top N**.

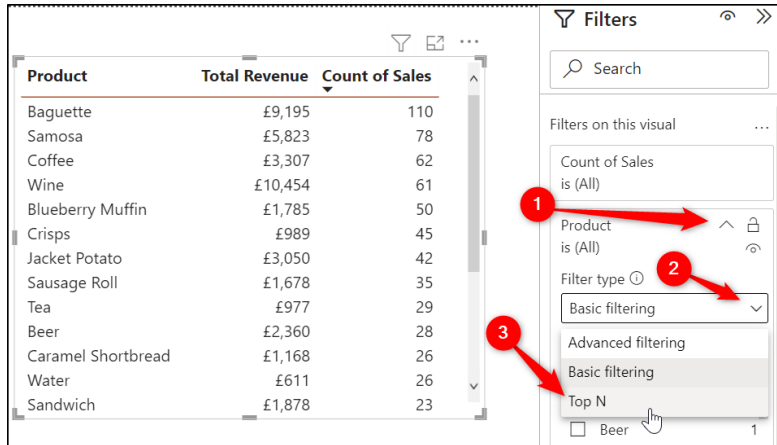


Figure 14.25: Setting a Top N filter for a specific visual

4. Type “10” in the **Show items** field (figure 14.26).
5. Click and drag the **Total Revenue** measure into the *By value* field to specify this as the criteria to filter the products by.
6. Click the **Apply filter** link.

Figure 14.26 shows the completed table showing the Top 10 products by total revenue. The products are still in descending order by the count of sales.

The **Filters** pane could be collapsed at this point so that it does not take up valuable screen space. It can be shown only when needed.

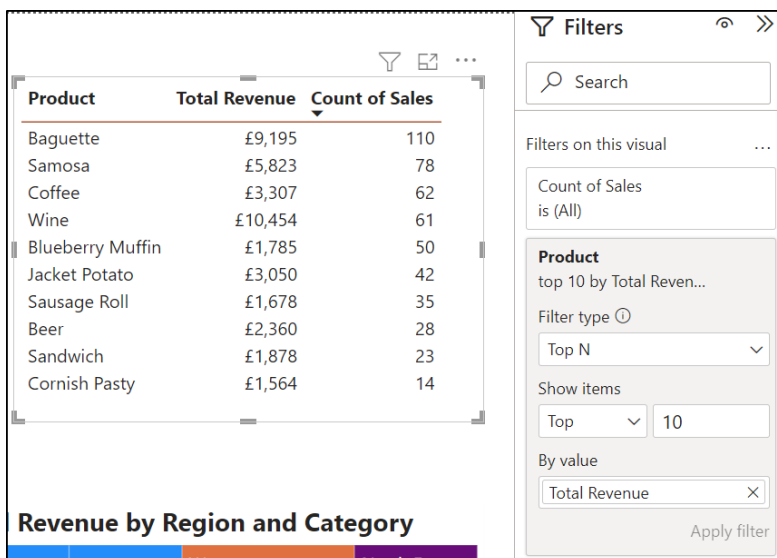


Figure 14.26: Top 10 products by total revenue using the Filters pane

The Filters pane can also be hidden from readers of the report. To do this, click the eye icon in the header of the Filters pane (as shown in *figure 14.27*).

The Filters pane will not be shown in the report when in the reading view of the Power BI service. This prevents users with permission to read reports only from interacting with the filter:

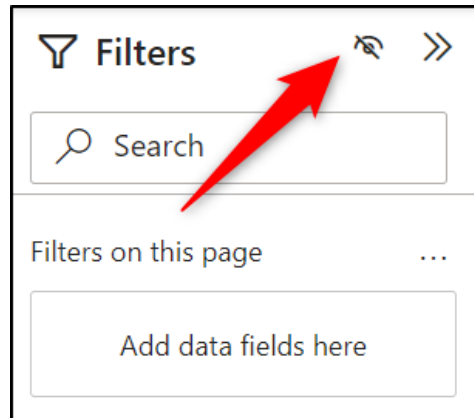


Figure 14.27: Hide the filters pane from readers

Filter all visuals on a page or all pages

The Filters pane can also filter all the visuals on a page or visuals on all pages.

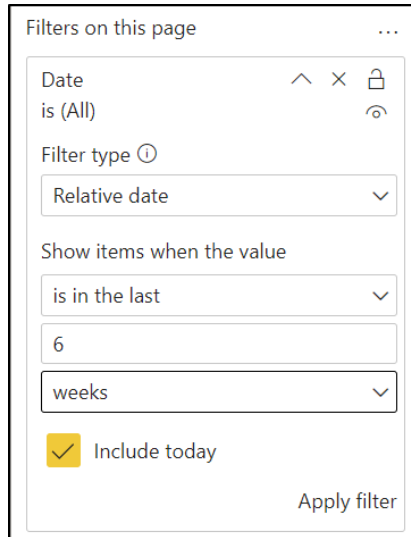
For this example, we will filter all visuals on a page to show data for the last six weeks only.

Earlier on in this chapter, we saw how a slicer can be used with date fields to create a timeline to filter date ranges. We also saw that the slicer could be changed to filter for relative dates, such as the last six weeks or the last thirteen months.

This is great! But if you are setting a filter for something such as the last six weeks, it is probably not common for users to interact and change that filter criteria.

So, using the Filters pane offers an alternative approach to filter all visuals on a page, all pages, or specific visuals in this way.

1. From the page where you want to apply the filters, click, and drag the **Date** column into the **Filters on this page** well of the Filters pane.
2. Click the **Filter type** list and click **Relative date**.
3. In the **Show items when the value** section, choose **is in the last**, type “6” in the box for the number of periods and then select **weeks** from the interval list (as shown in *figure 14.28*).



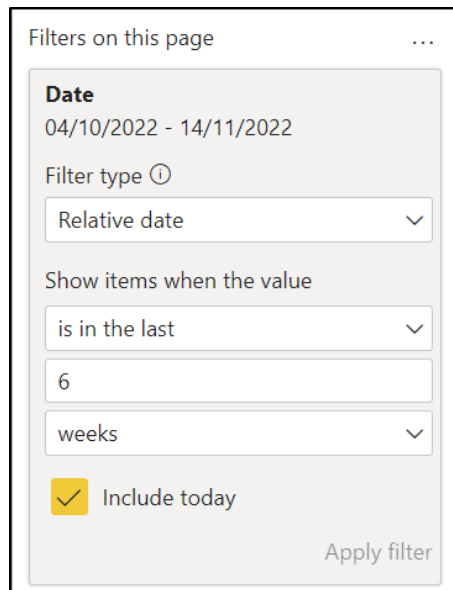
The screenshot shows a dialog box titled "Filters on this page" with a three-dot menu icon in the top right corner. Inside the dialog, the filter is configured as follows:

- Field: Date
- Value: is (All)
- Filter type: Relative date
- Show items when the value: is in the last
- Duration: 6 weeks
- Include today:
- Apply filter button

Figure 14.28: Creating a relative date filter for all visuals on a page

- Specify if you want to include today's date in the filter by checking the **Include today** box.
- Click **Apply filter**.

Figure 14.29 shows the filter applied. The date range used in the filter is shown at the top of the well.



The screenshot shows the same dialog box as Figure 14.28, but with the filter applied. The date range is displayed at the top of the filter well:

- Date: 04/10/2022 - 14/11/2022
- Filter type: Relative date
- Show items when the value: is in the last
- Duration: 6 weeks
- Include today:
- Apply filter button

Figure 14.29: Filter is applied, and the used date range is shown

To clear the filter criteria for the **Date** column, click the **Clear filter** button (eraser icon). To remove the filter completely, click the **Remove filter** button (x icon).

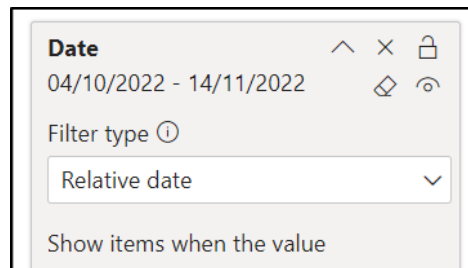


Figure 14.30: Clearing or removing a filter

Using drill through

The drill through filter in Power BI provides a simple method for a user to dive into further detail on a specific data point. The user is taken from the data point in a visual to an entire page of visuals to give more insight.

To set up drill through, first, a destination page is created with the required visuals, just like any other report page. A drill through field is then specified for that page.

Now, from any other page in the report where that field is used in a visual, the user can right-click a data point and drill through to the destination page. The destination page is filtered for the selected data point.

Note: Tooltip pages are covered in *Chapter 15, Enhancing your Power BI Reports*, and these offer an alternative approach to the drill through technique. They are smaller and present less data than an entire report page, but they prevent the need to hop around different pages for further analysis.

Figure 14.31 shows a drill through being performed from a stacked bar chart. Two different drill through pages are offered to the user—**Regional Analysis** and **Product Sales**.

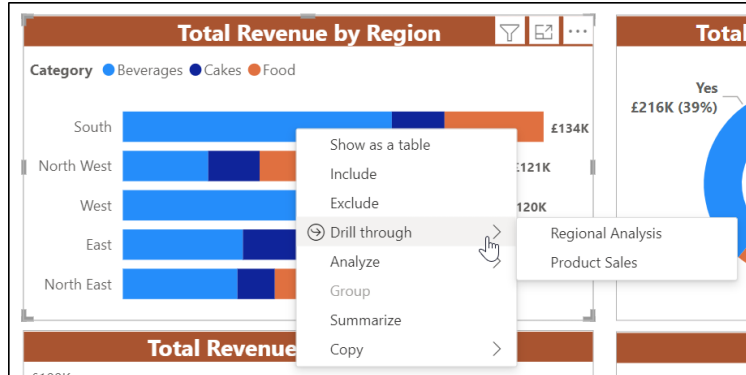


Figure 14.31: Drill through filter from a stacked bar chart

Creating a drill through

In this example, we will setup a drill through on the **Regional Analysis** page. The slicer that was created on that page earlier in this chapter has been removed, and a drill through will be created instead.

The **Region** column in the **Stores** table will be used as the drill through field. So, on any report page and from any visual using the **Region** field and with the capacity for drill through, the user can drill through to the **Regional Analysis** page.

1. Navigate to the **Regional Analysis** page.
2. Click and drag the **Region** field into the **Add drill-through fields here** well in the **Drill through** section of the **Visualizations** pane (as shown in *figure 14.32*).

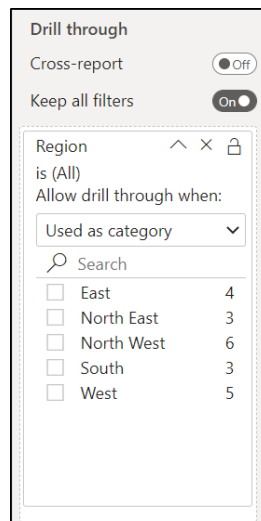


Figure 14.32: Region field used for drill through

It appears as a filter in the *Drill through* section (figure 14.32), but we are not going to filter this field in the manner that we did in the Filters pane previously. Our interest is in the capacity to drill through that we now have from other visuals.

You will also notice a back arrow icon appear in the top left corner of the **Regional Analysis** page (figure 14.33). This arrow button returns the user to the previous page that the drill through took them from. In Power BI Desktop, you need to hold Ctrl and click the button, but when the report is published to the Power BI service, the button can simply be clicked.

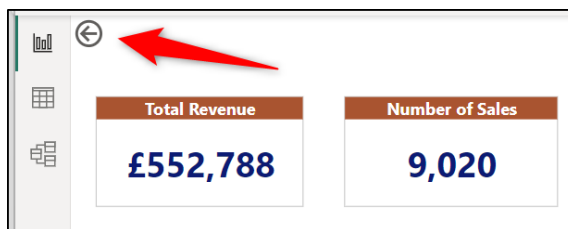


Figure 14.33: Button to return to the previous page

Performing the drill through

Other visuals in the report that uses the **Region** field can now be used to drill through to the **Regional Analysis** page.

Let us use the stacked bar chart on the **Front Page** page as the source of a drill through.

1. Navigate to the **Front Page** page.
2. Right-click on one of the data points, position the mouse over the **drill through** option and click **Regional Analysis**.

In figure 14.34, the data point for the Food sales in the North West region has been selected:

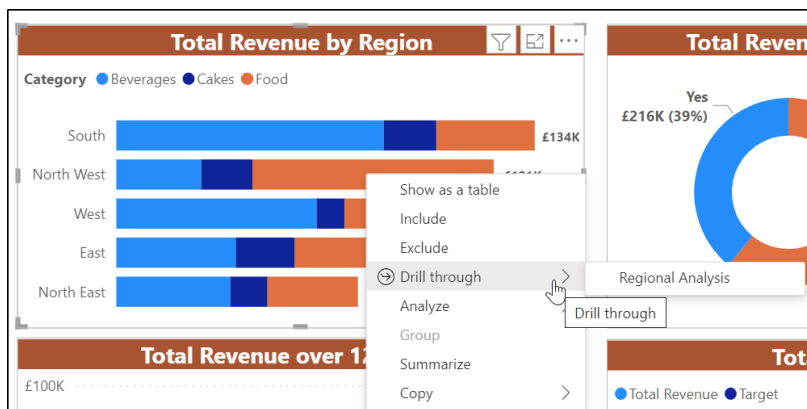


Figure 14.34: Drill through from the North West and food data point

The user is taken to the **Regional Analysis** page, and the visuals are filtered for the selected data point.

Note: You can create multiple drill through pages for a specific field. When a user right-clicks on a data point, all the drill through pages for that field are listed.

Figure 14.35 shows the **Drill through** section of the **Visualizations** pane on the **Regional Analysis** page. You can see that the visuals have been filtered by the category of food as well because the data point on the stacked bar chart was for North West and food.

This may come as a surprise because we only added the **Region** field to the *Drill through* area.

This has occurred because the **Keep all filters** option in the **Drill through** section is set to *On* (figure 14.35). So, any other filters that have been applied, such as the stacked bar charts internal filter, but also any filters set by slicers or the Filters pane, are retained on drill through.

Click the *On/Off* toggle for the **Keep all filters** to **Off** to stop other filters from carrying through. So, if you follow the drill through from a data point in the stacked bar chart again, only the region is filtered.

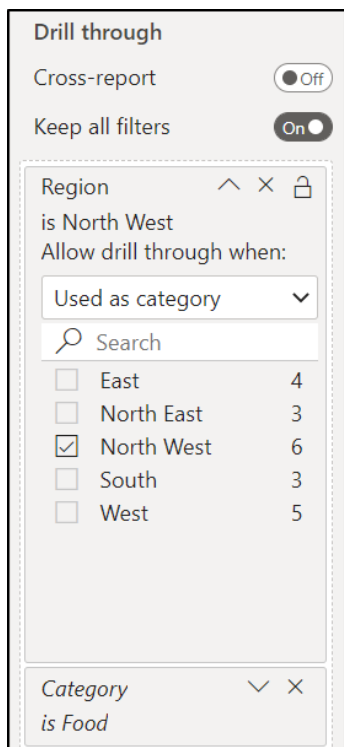


Figure 14.35: Category filter is also applied

One of the struggles with using the drill through technique is the user's awareness of the option to drill through. And when taken to the destination page, knowing which filters have been applied.

When a user positions their mouse over a data point with the capacity for drill through, the tooltip directs them to this ability (*figure 14.36*).

However, it is still not that obvious to all users that this option exists when using a report. Therefore, the use of chart titles or other labels to inform a user that they can drill through is recommended.

As mentioned, another struggle is that it is not always clear which filters have been applied on the destination page. Especially if the **Keep all filters** option is set to on, which it is by default.

So, let us look at creating a simple label on the destination page to help the users know which region the page visuals are filtered by.

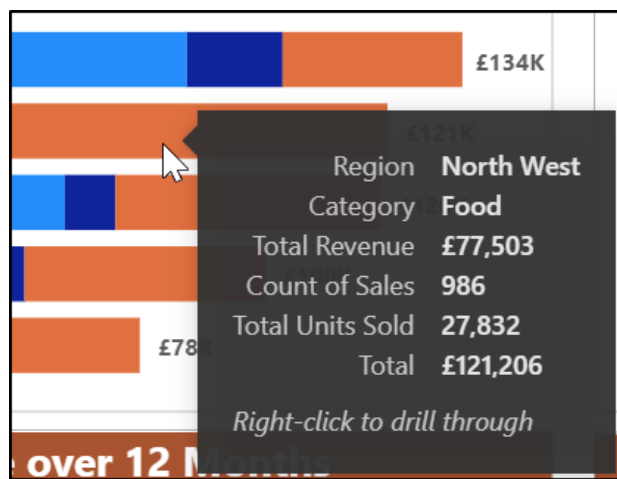


Figure 14.36: Right-click to drill through message on the visual tooltip

Adding a page title

In this example, we will create a simple page title that tells the user which region has been filtered via the drill through.

To do this, we will write a simple DAX measure and then use it in a text box on the page.

In cases when the **Keep all filters** setting in the **Drill through** section of the **Visualizations** pane is set to On and more complex filter criteria is applied, more advanced DAX techniques will be required. That is beyond the scope of what we will cover in this chapter.

Let us first create the measure by using the following steps:

1. Right-click on the **Sales** table and click **New measure**.
2. Type the following formula into the formula bar.

Selected Region = SELECTEDVALUE(Stores[Region]) & “ region”

This formula creates a measure named **Selected Region**. The **SELECTEDVALUE** function is used to return the selected value from the **Region** column of the **Stores** table. The string “ region” is then appended to the selected value.

Tip: Once created, the [Selected Region] measure can be moved to the Measures folder in the Sales table. This was covered in Chapter 9, Adding DAX Measures.

Let us now add this to a text box on the **Regional Analysis** page.

1. Navigate to the **Regional Analysis** page.
2. Click **Insert | Text box**.
3. Click and drag the text box to position it where you want on the page.
4. In the **Format** pane of the text box, click the *On/Off* toggle for the **Title** section to **On** and expand its options.
5. Click the **Conditional formatting** icon (*figure 14.37*) beside the **Text** box.

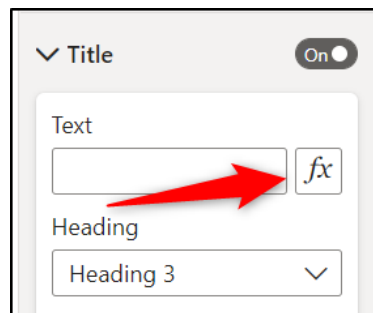


Figure 14.37: Conditional formatting icon for the text box title

6. In the **Title text – Title** window (*figure 14.38*), keep the **Format style** as **Field value** and select the **Selected Region** measure from the **What field should we base this on?** list.

7. Click **OK**.

Figure 14.38: Using a measure for the text box title

8. Format the title text how you want. For example, change the font, increase the size, and change its color.

Figure 14.39 shows a simple text box completed. The North West region has been used for the drill through criteria. Please refer to the following figure:

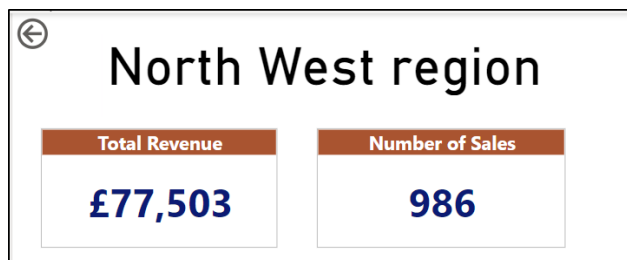


Figure 14.39: Text box showing the region selected at drill through

Conclusion

In this chapter, we learnt how to use the different filter options available for your Power BI reports. These were the visual interactions, slicers, Filters pane, and the Drill through feature. We saw how to apply each filter and customize each one to suit our requirements.

In the upcoming chapter, we will take our Power BI reports further using features such as bookmarks and tooltips. These fantastic features add further interactivity to our reports and richness to our visuals.

We will also add objects such as images and buttons to our report page. These will be used to add identity, improve page navigation, and make the report more accessible. Finally, we will look at themes and how they can be used to simplify the formatting of our reports and ensure a consistent look and feel.

Questions

Here are some questions to test what you have learnt in this chapter.

- 1. What are the correct steps to edit interactions from a visual?**
 - a. Click on the visual and click Insert > Interactions.
 - b. Click on the visual and click Format > Edit Interactions.
 - c. Click on the visual and click Modeling > Visual Interactions.
 - d. Click on the visual and click View > Edit Interactions
- 2. Which of the following are formatting options for a slicer that uses a text field?**
 - a. Show or hide the slicer header.
 - b. Specify single select only for the slicer options.
 - c. Change the alignment from vertical to horizontal and vice versa.
 - d. All of the above
- 3. Cross-highlighting and cross-filtering between visuals can be disabled by default.**
 - a. True
 - b. False
- 4. The Filters pane can be hidden on the report when in reading view.**
 - a. True
 - b. False

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



CHAPTER 15

Enhancing Your Power BI Reports

Introduction

In this chapter, we will make some final enhancements to our report before we publish it to the Power BI service and share it with others.

We will first look at inserting text boxes, images, and buttons into our report. These elements can be used to annotate key aspects of our report, add branding, divert the reader's eyes to what is important, and add interactivity. We will then add a background so that the visuals pop on the page.

Next, we look at creating tooltip pages. This is a useful method for enabling readers to gain further insights from a visual on a simple mouse over a data point. These custom tooltips can be designed how we want and are very powerful.

Finally, we will cover bookmarks to save the current state of a report page so that it can be quickly returned. With bookmarks, you can create useful interactions on your report page so that a reader can apply filters or switch visuals at the click of a button.

Structure

In this chapter, we will cover the following topics:

- How to insert text boxes, images, and buttons into a report.

- Assign different actions to buttons and images, including page navigation and viewing a bookmarked page.
- Format the background of a report page.
- How to create tooltip pages so a user can gain further insights from a visual quickly and easily.
- How to use bookmarks to add further interactivity to a report.

Objectives

After reading this chapter, you will learn how to enhance your report pages using text boxes, images, and buttons. These elements are used to provide easier navigation, extra interactivity, and a more accessible report.

You will also know how to create custom tooltip pages and use bookmarks. These two features are fantastic for taking your report further. Tooltips enable users to dive into more detail on a data point quickly, and bookmarks allow users to personalize a report by switching visuals or changing filters at the click of a button.

Adding text boxes, images, and buttons

Files: **sales-report-ch-15-start.pbix**

In this section, we will cover elements that are familiar to all reading this book—text boxes, images, and buttons. These elements are very useful for adding to the user experience, or UX, with our reports.

Adding text boxes

Reports can be made more accessible by adding text boxes for effective labeling and annotation of specific functionality, such as a drill through feature or tooltip information. Aspects that a reader may not be aware of without it being stated somewhere.

We used a text box toward the end of *Chapter 14, Report Interactions, Filters, and Slicers*, to display a dynamic page title that showed the used filter was applied by a drill through to a page.

In this example, we will add a static page title to the **Front Page** page of our report.

1. Navigate to the **Front Page** page of the report.
2. Click **Insert** | **Text box**.
3. Type the text “**Sales Report**”.

4. Click on a blank area of the page to deselect the text box, and then click and drag it into position at the top of the report (as shown in *figure 15.1*). The text box is initially positioned in the bottom right corner of the page and can be challenging to move. Drag the text box using the frame of the text box.
5. Select the text in the text box and use the toolbar to apply some formatting. In *figure 15.1*, the text has been changed to size 32 and bold.

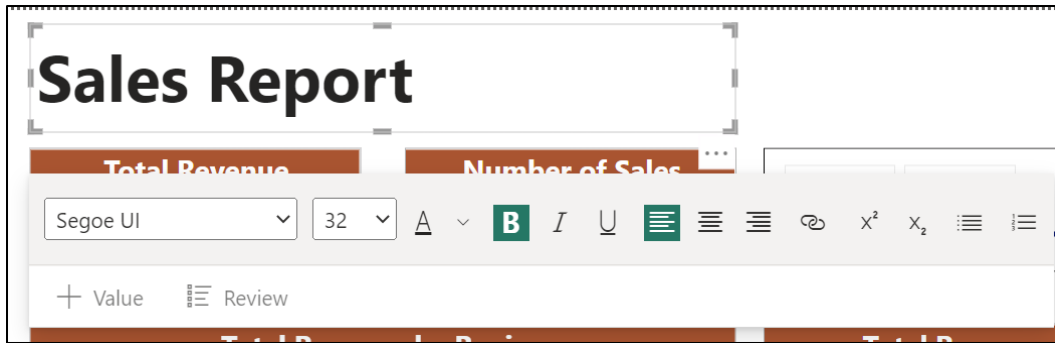


Figure 15.1: Formatting a text box in a Power BI report

The *Value* and *Review* buttons in the text box toolbar allow you to add and manage dynamic values in the text box. We saw an example of this when the smart narrative visual was discussed in *Chapter 13, Other Power BI Visualizations*. Using these buttons, you can create dynamic values in text boxes without the need for complex DAX statements.

There is so much potential in the different features of Power BI that extend beyond the scope of this book. It is recommended to spend time exploring all the features that are covered and continue your journey to learn more.

Adding images

Static images can be added to your reports for custom branding or to be used as a clickable object to navigate to another page, change bookmark, or be taken to a Web URL.

For this example, we will add an image to represent the company logo. We will then add an action to the logo so that when clicked, it takes users back to the first page of the report.

1. Navigate to the **Front Page** page of the report.
2. Click **Insert | Image**.
3. Locate the **Coffee House.png** image in the *Chapter-15-images* folder. Select it and click **Open**.

- The image is placed on the page where there is some space. Move the page title text box inserted previously to make space, drag the image into the top left corner beside it, and resize as appropriate (as shown in *figure 15.2*):

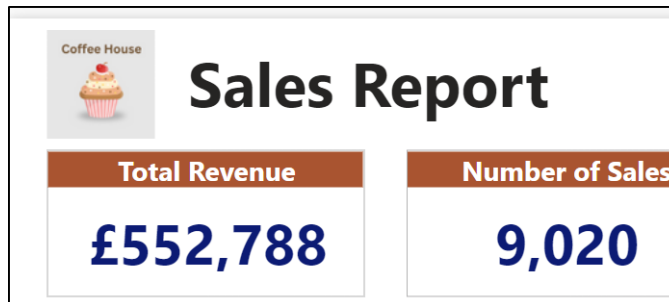


Figure 15.2: Logo image on the first page of the report

The image has a white background surrounding it. We will change this to match the background color in the image.

- With the image selected, in the **Format** pane, click the **General** section.
- Expand the **Effects** options, click the **Color** list, and select the *White, 10% Darker* color. This is the same color as the image background and uses the hex code #E6E6E6.
- Change the **Transparency** to 0% (as shown in *figure 15.3*).

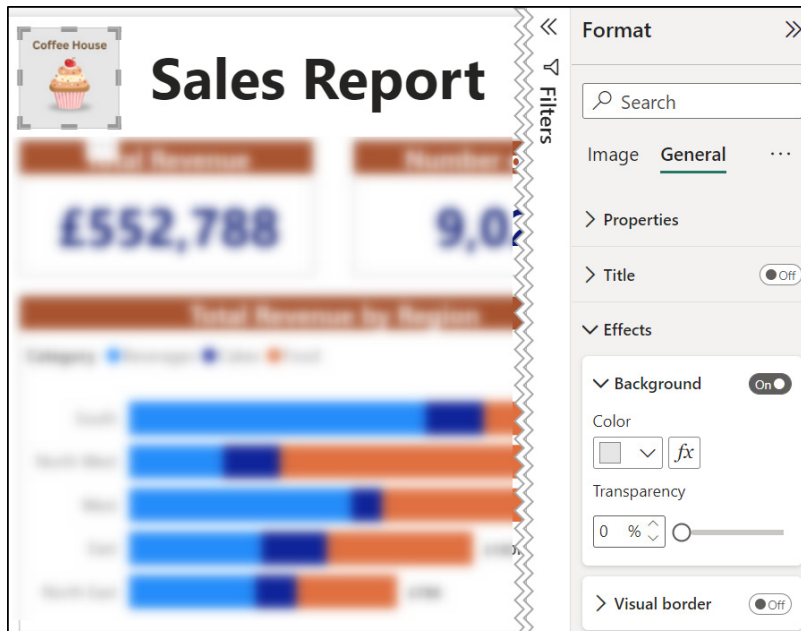


Figure 15.3: Changing the background for an image

We will now copy this image and the page title text box to the same position on each page of the report. The text in the text box will be changed to represent the page title, and we will apply an action to the image so that it will take the user back to the **Front Page** page when clicked.

1. Select the image and text box on the page and click **Home | Copy**.
2. Navigate to the **Monthly Sales Analysis** page and click **Home | Paste**.
3. Change the text in the text box to match the page title (“**Monthly Sales Analysis**” in this case).

Note: You may need to move some visuals and resize the text box to fit them on the different pages.

Figure 15.4 shows the logo image and page title for the **Monthly Sales Analysis** page.



Figure 15.4: Logo image and page title on the Monthly Sales Analysis page

4. With the image selected, in the *Format* pane, click the **Action On/Off** slider to **On** and expand the options.
5. Click the *Type* list and select **Page navigation** (as shown in figure 15.5).

- Click the **Destination** list and select **Front Page**.

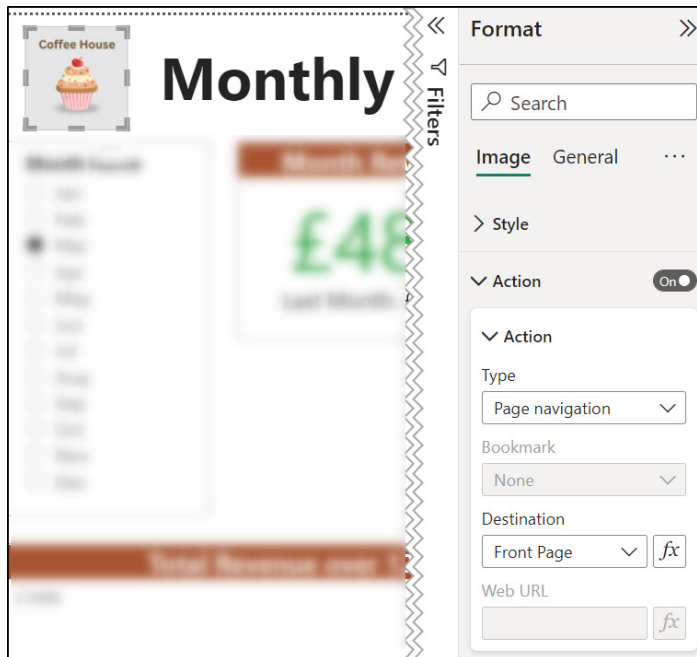


Figure 15.5: Applying an action to navigate to a specific page

- Repeat these steps for each page of the report.

Tip: We will discuss adding a background to report pages shortly. You can design a background using software such as Canva that includes images and other shapes. This background will save time and ensure consistency versus copying images and shapes across the pages of a report.

Adding buttons

The ability to add actions to images and shapes is great and provides an alternative to using buttons for this purpose. However, let us give buttons their due respect and discuss their use to aid page navigation.

We will see buttons used later in this chapter to enable a user to easily switch between bookmarks. Buttons can also be used to trigger drill through filters and navigate to Web URLs, but we will not cover these examples in the book.

Changing the drill through back button

For the first example, we will replace the default back button that is inserted when creating a Drill through filter with our own back button. Instead of the default back arrow icon (as shown in *figure 15.6*), we will have a button that simply displays the text “Back”. Please refer to the following figure:

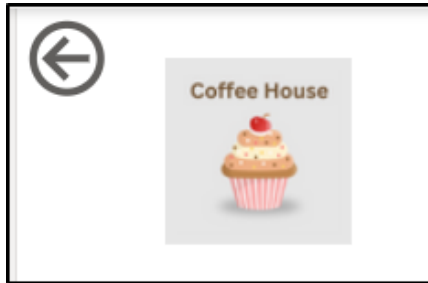


Figure 15.6: Default Back arrow button

1. Navigate to the **Regional Analysis** page, select the default back button, and press *Delete* on the keyboard to remove it.
2. Click **Insert | Buttons** (as shown in *figure 15.7*).
3. Click **Blank**.
4. Position the button in the top right corner of the page and align it with the logo image and text box.
5. With the image selected, in the **Format** pane, click the **Action On/Off** slider to **On**. The default action is *Back*, so no other changes are necessary here:

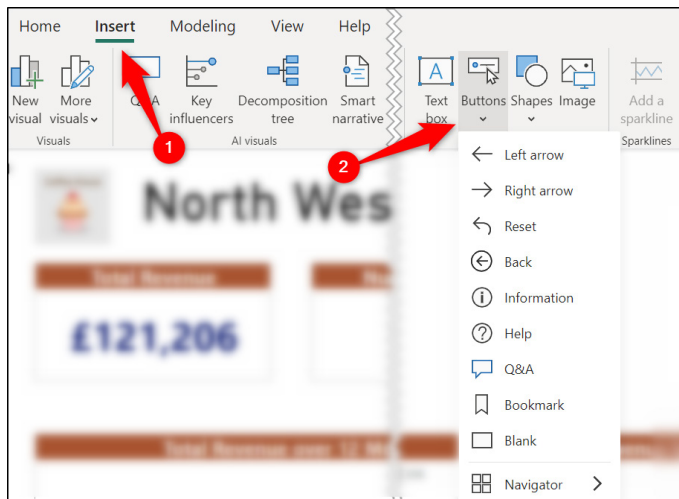


Figure 15.7: Insert buttons in Power BI Desktop

We will now go through the steps to change the button appearance and add the “Back” text.

6. With the button selected in the **Format** pane, expand the *Shape* section and add rounded corners to the rectangle button. In this example, the *Rounded corners* option is set to **6px**. Notice the option here to change the button shape if required.
7. Expand the *Style* section and click the **Text On/Off** slider to **On**. Type “Back” in the *Text* box. Click the **Bold** button, change the *Font size* to **12**, and set the *Font color* to **Black**.
8. Expand the *Fill* section, change the *Color* to **Black**, and the *Transparency* to **0%**.
9. Click the **Border On/Off** slider to **Off**.

Figure 15.8 shows the back button in its natural state on the report page. For the final formatting change, we will change the fill color to a dark blue when the button is in an *On hover* state.

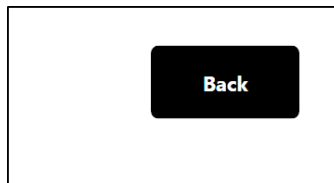


Figure 15.8: Back button in the natural state

10. In the *Style* section of the *Format* pane, click the *State* list and click **On hover**.
11. In the *Fill* options, click the **Color** list and select the dark blue that matches the one used in other areas of the report.

Figure 15.9 shows the back button in the *On hover* state.

Note the message to *Ctrl + click* to follow the link. This is because we are in Power BI Desktop, the authoring tool of Power BI. When the report is published to the service, this will be a single click without the need to hold *Ctrl*:

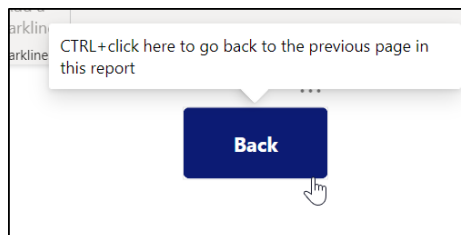


Figure 15.9: Back button in *On hover* state

Perform the drill through filter from the **Home Page** and use the new back button to see it functioning correctly.

Page navigator buttons

For the next example, we will add the *Page navigator* buttons to provide an “on page” method for a reader to easily navigate the pages of the report.

1. Navigate to the **Front Page** page of the report.
2. Click **Insert** | **Buttons** | **Navigator** | **Page navigator**.
3. Move and resize the buttons to position them in the top right corner of the page and align them with the logo image and page title (as shown in *figure 15.10*):

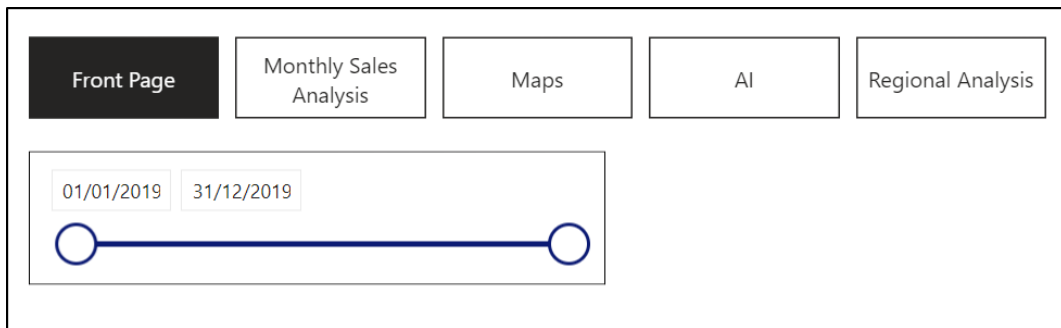


Figure 15.10: Page navigator buttons on the report page

4. Follow the steps performed in the previous example to make the shape corners rounded at 6px, the text bold and white, fill color black, and the *Hover* fill color dark blue.
5. In the *Style* section, click the *State* list and change it to **Selected**. These settings change the appearance of the button for the page you are currently viewing. Change the text color to black and the fill color to white.

Figure 15.11 shows all three defined button states. The **Maps** button shows the *Hover* state, and the **Front Page** button shows the *Selected* state. The remaining buttons are in their *Default* state.

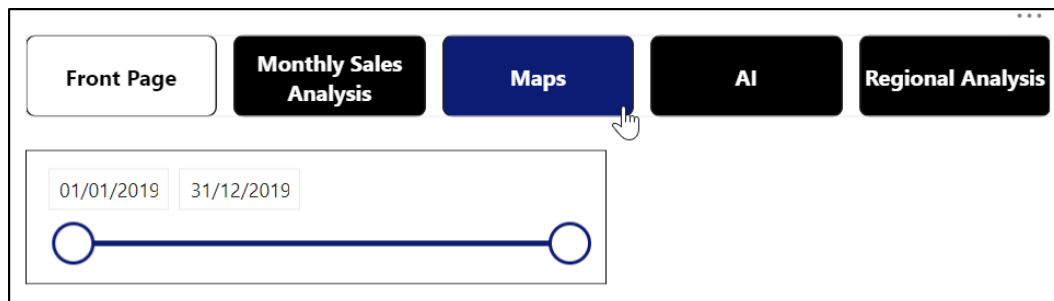


Figure 15.11: Page navigator buttons with the three defined button states displayed

We do not want the **Regional Analysis** page tab to be provided as a button in the page navigator. Let us take the necessary steps to hide it.

6. Right-click on the **Regional Analysis** page tab at the bottom of the report and click **Hide Page** (as shown in figure 15.12).

An eye icon with a strike-through line is shown for hidden pages in Power BI Desktop (in the Power BI service, this page will not be visible).

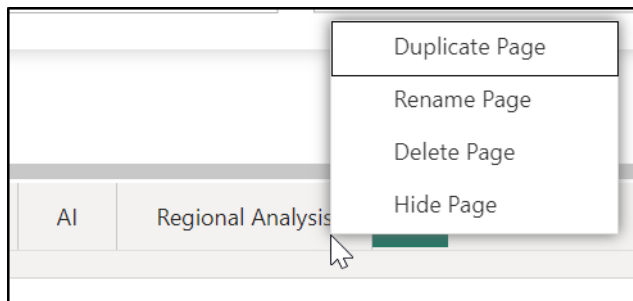


Figure 15.12: Hiding a page in Power BI Desktop

7. Select the *Page navigator* buttons, and in the **Format** pane, expand the **Pages** section and click the *On/Off* slider button for the **Show hidden pages** option to **Off** (as shown in figure 15.13).

The **Regional Analysis** page is removed from the page navigator.

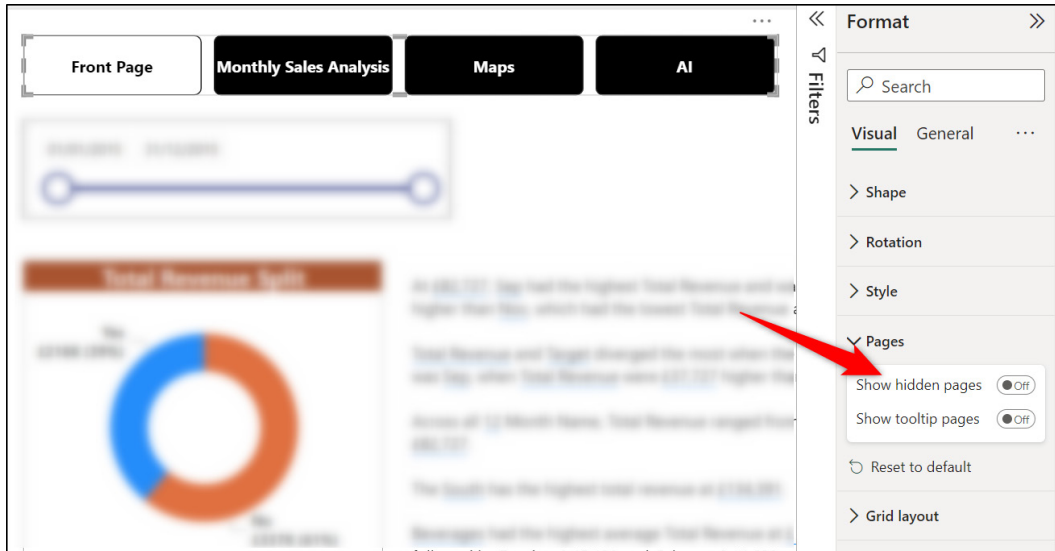


Figure 15.13: Preventing hidden pages from showing in the page navigator

- Copy and paste the page navigator buttons to all pages of the report except the **Regional Analysis** page.

Adding a background

Let us now add a background color to our report pages instead of the standard white background. This will emphasize the visuals more on the page.

For this example, we will apply a simple light grey background with the hex value #E6E6E6. This is known as white, 10% darker in the color palette of Power BI.

Note: It would make sense to make these changes early in a report so that you can duplicate pages and do not need to apply the background to every existing page. Not all content in the book is discussed in the order that you may apply them when creating your reports.

- Navigate to the **Front Page** page and click a blank area of the page to ensure that no visuals are selected.
- Click the **Format your report page** button (the same button used to access the formatting options for selected visuals).
- Click **Canvas background** to expand the formatting options.
- Click the **Color** list and click the *White, 10% darker* color.

- Set the **Transparency** setting to **0%**.

Figure 15.14 shows the canvas background settings applied to the **Front Page** page.

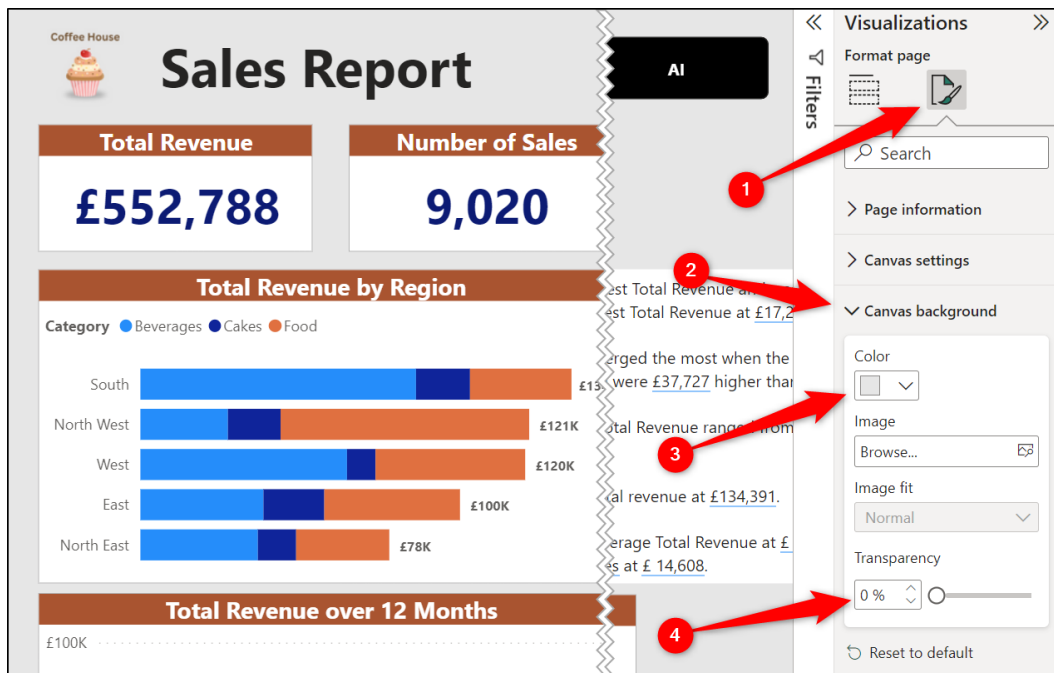


Figure 15.14: Canvas background settings applied to a report page

You may want to change the background color of some of the visuals to match the background. The most notable ones are the background of the text box used for the page title and the slicer. So, let us look at changing these.

- Click on the text box to select it.
- In the **Format** pane, expand the **Effects** section.
- Click the **Color** list in the **Background** formatting section and choose *White, 10% darker* color.
- Repeat the steps for the slicer but remember that the **Effects** section is found in the *General* category when formatting the visual (as shown in figure 15.15).

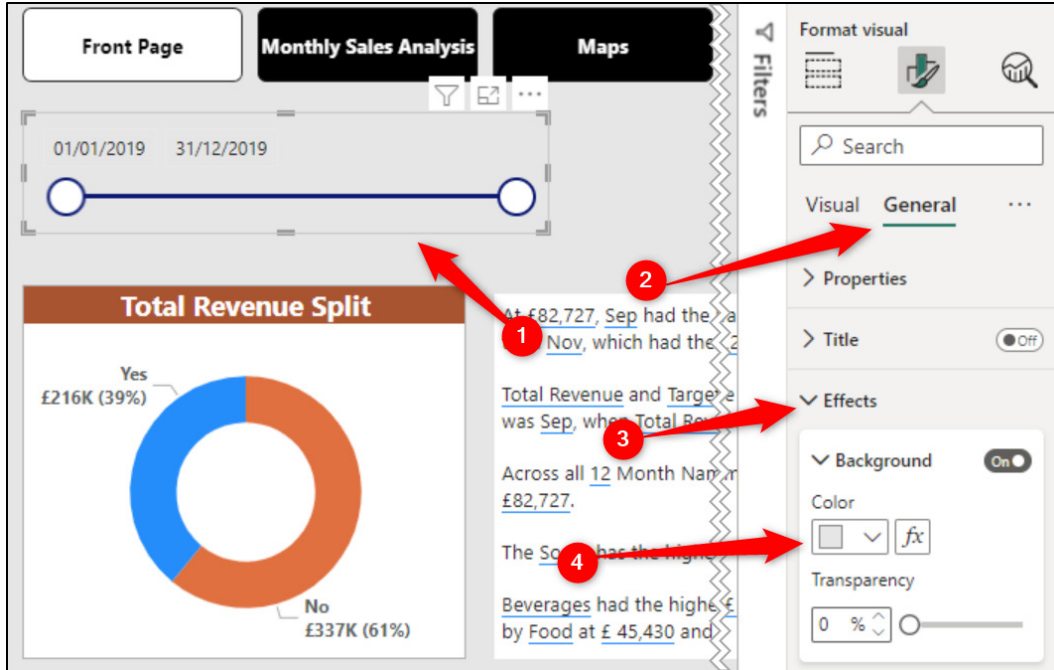


Figure 15.15: Changing the background of the slicer

Changing the background color of the slicer to match the background of the page is effective because it provides a visual indicator to the reader that the slicer effects all visuals on the page.

Repeat the steps to format the background of each page of the report with the *White, 10% darker* color. Then format the background of any visual that you feel requires the change and to the text box.

Custom tooltip pages

Creating custom tooltip pages is a brilliant way to provide deeper insight into the specific data points of a visual. When hovering over the data point in a chart, table, or matrix, a small page will appear showing a further breakdown of that data.

We have covered the default tooltip pages that are automatically enabled for visuals at different times already in this book. *Figure 15.16* shows the tooltip pages that were customized for the map visual in *Chapter 12, Using Maps in Power BI Reports*.

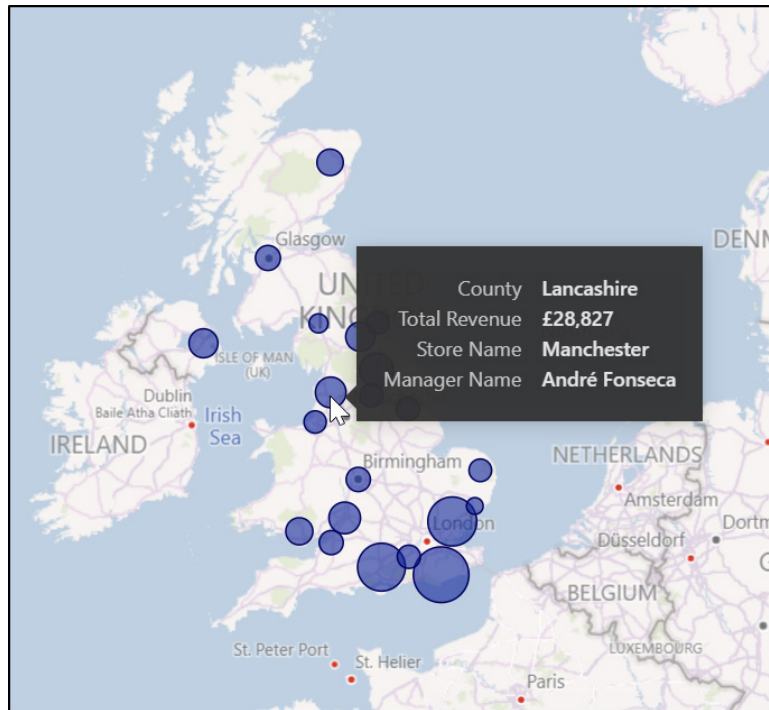


Figure 15.16: Default tooltip page customized for the map visual

Additional fields can be added to the automatically provided tooltip pages, as shown by the **Store Name** and **Manager Name** fields being added to the tooltip page for the map. This provides you with the ability to display data beyond what the visual is showing superficially.

However, you cannot customize aspects such as the background color, font, or alignment. And most importantly, you cannot add other charts, tables, or images.

By creating custom tooltip pages, you are free to design the page how you like, including its size, and add any visuals, images, or text boxes that you want.

For this example, we will create a tooltip page for use with the line chart visuals on the **Front Page** and **Monthly Sales Analysis** pages.

We will add two card visuals to the tooltip, one to display the total revenue that month and another to display the percent revenue change from the previous month. We will also add a column chart to display the top five selling products that month.

Creating a custom tooltip page

To create a custom tooltip page, you must first insert a new page and then allow it to be used as a tooltip page.

1. Click the **New page** button.
2. Double-click on the new page and type “**TT Month**” for its name.

Tip: The “TT” in its name stands for tooltip. This is not necessary, but it is a naming convention that I like to use to identify tooltip pages from other pages easily.

3. Right-click on the page and click **Hide Page**. This will prevent the page from being accessed in a conventional way and only as a tooltip.

Figure 15.17 shows the new hidden tooltip page:

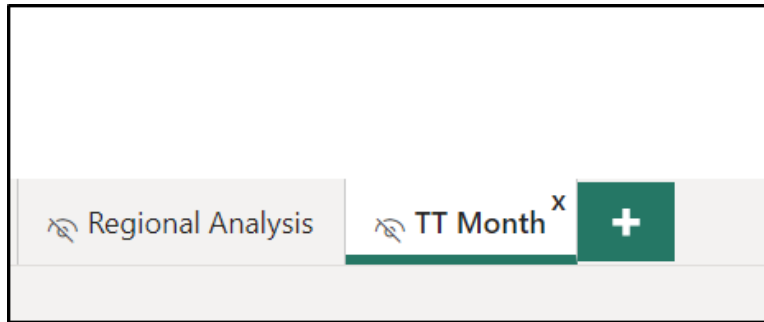


Figure 15.17: New hidden page for the tooltip

4. Click on the **Format your report page** button, as shown in figure 15.18.
5. Click **Page information** to expand the options and click the **Allow use as tooltip** *On/Off* slider to **On**.

It is now a tooltip page and has decreased in size significantly, as shown in *figure 15.18*. This is because it appears when the cursor is hovered over a data point and obscures other content on the report page.

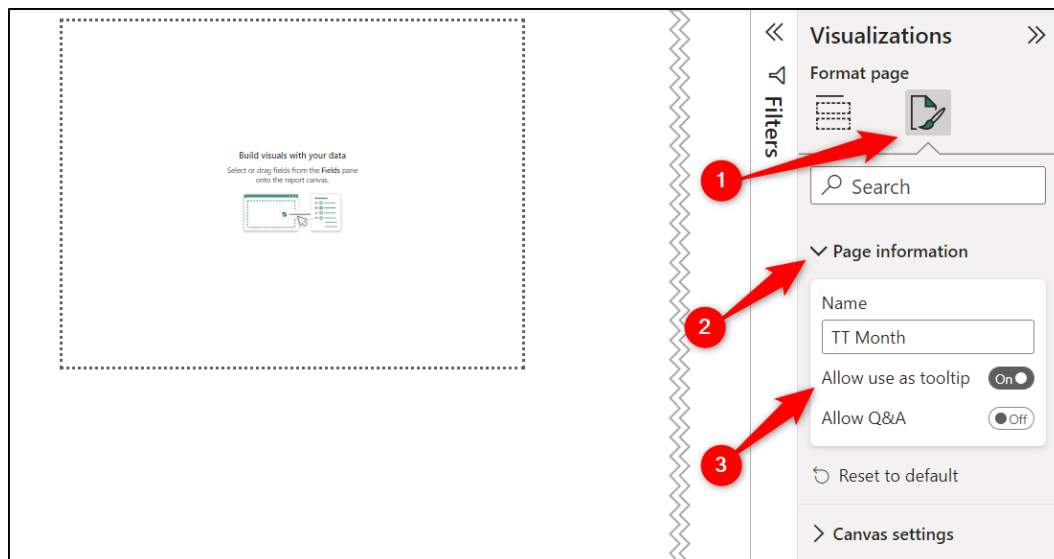


Figure 15.18: Enabling the use of a page as a tooltip page

You can resize the page and make it larger if you prefer or even smaller. In this example, we will keep with the classic tooltip size. However, to access these options, click **Canvas settings** from the formatting options of the **Visualizations** pane, change the **Type** from **Tooltip** to **Custom**, and enter the preferred **Height** and **Width** dimensions (refer to *figure 15.19*):

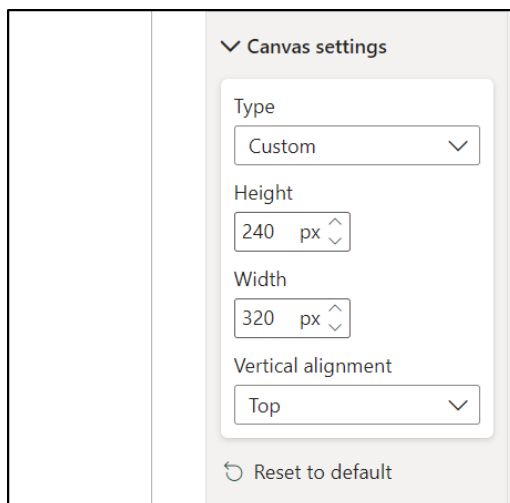


Figure 15.19: Customizing the tooltip page size

Next, we will change the background color of the page. This is something that we just discussed in relation to normal report pages.

6. Click **Canvas background** to expand the options, click **Color** and choose a light yellow, hex reference *f0e199*, and change the *Transparency* to **0%**.

Before we start to add visuals and other content to the tooltip page, you may want to consider changing the page scale. Because the tooltip page size is small, it can be difficult to click parts of a visual and make the necessary changes.

7. Click **View | Page view**, and then click **Fit to page** (as shown in *figure 15.20*):

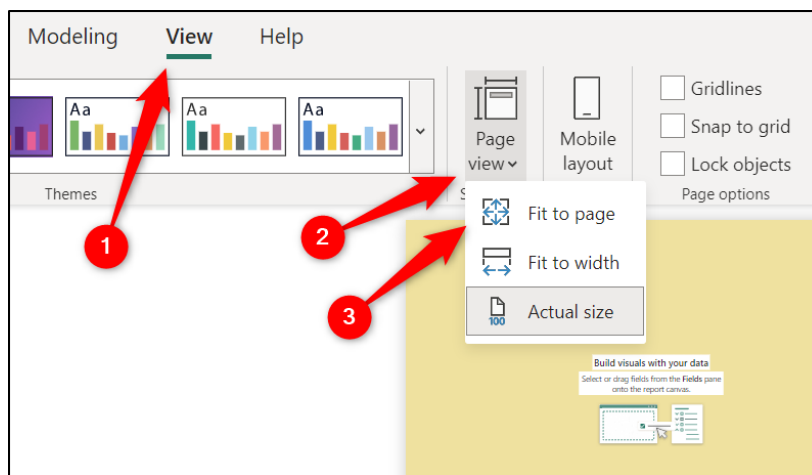


Figure 15.20: Scaling the tooltip to “fit to page”

Adding visuals to the custom tooltip page

We will now add the visuals to the page in the same manner as we have covered in the last few chapters of the book. There will be some adjustments needed to resize visuals, decrease fonts, and remove anything that does not benefit the visual to cater to the smaller page size.

Some of the formatting steps are written in shorthand, as they have been covered in detail in the relevant chapters.

First, let us add the two card visuals to display the total revenue and percent revenue change from the previous month.

1. Click the **Add data to your visual** button to switch from the *Format page* view of the *Visualizations* pane to the *Build visual* view.

2. Click the **Card** visual to insert it on the page. It appears very large due to the increased scale of the page. Resize it to something suitable.
3. Click and drag the **Total Revenue** measure to the *Fields* well.
To quickly apply the same formatting as was used previously on other cards in the report, we will use the Format Painter tool.
4. Navigate to the **Front Page** page, click on a card visual on the page, click **Home** | **Format Painter**, navigate back to the **TT Month** page, and click on the card.
5. Click the **Format your visual** button to see the formatting options for the card.
6. Click **General** | **Title**, and type “**Total Revenue**” in the *Text* field. Change the *Font* size to **10**.
7. Click **Visual** | **Callout value** and change the *Font* size to **22**.
8. Copy and paste the card to duplicate it and position the second card beside it.
9. Remove the **Total Revenue** measure from the *Fields* well and replace it with the [% Monthly Revenue Difference] measure.

Figure 15.21 shows the two completed card visuals on the page.

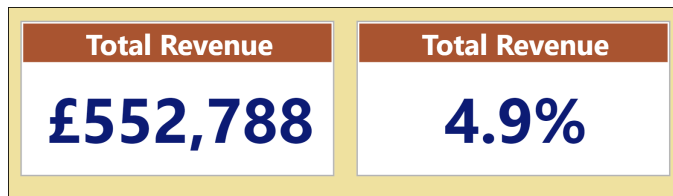


Figure 15.21: The two card visuals

Let us now add the column chart to present the top five selling products.

1. Click the **Clustered column chart** visual icon in the *Visualizations* pane to insert it on the page. Move and resize it appropriately.
2. Click and drag the **Product** field into the *X-axis* well and then click and drag the **Total Revenue** measure into the *Y-axis* well.

Let us set the top five filters using the **Filters** pane before we format the chart.

3. Expand the *Filters* pane if necessary. In the *Filters on this visual* section, expand the **Product** filters and select **Top N** from the *Filter type* list (refer to figure 15.22).
4. Under *Show items*, select **Top** in the list and type “**5**” in the box beside it.

- Click and drag the **Total Revenue** measure into the **By value** box and click **Apply filter**.

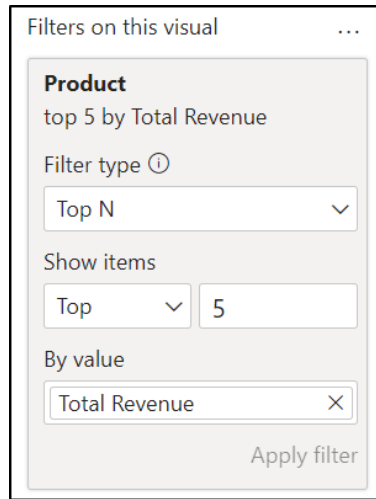


Figure 15.22: Top five products by total revenue filter

We will now apply a few formatting changes. We will remove unnecessary chart elements, add data labels, and reduce the font of the text on the x-axis.

- Click the **General** category in the formatting options of the *Visualizations* tab and click the *On/Off* slider for the **Title** to set it to **Off**.
- Click **Visual**, expand the **Y-axis** options, and click the *On/Off* slider for **Title** to **Off** and the *On/Off* slider for the **Y-axis** itself to **Off**.
- Expand the **X-axis** options and set the *Font* size to **8**.
- Click the *On/Off* slider for **Data Labels** to turn them **On**.

Figure 15.23 shows the completed clustered column chart.

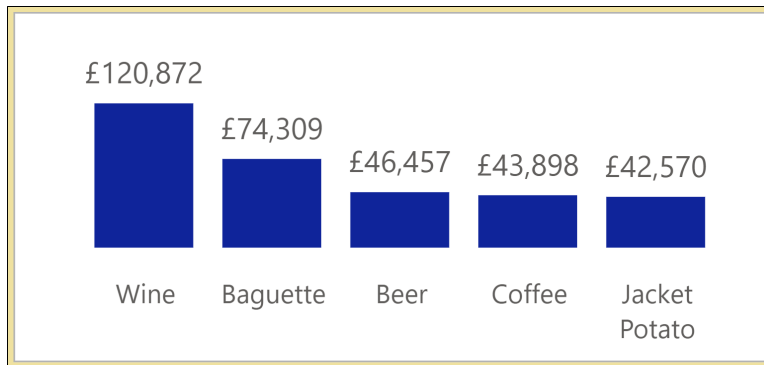


Figure 15.23: Column chart showing the top five products by total revenue

Adding the tooltip page to a visual

Now that we have a tooltip page, it is time to apply it to visuals in the report. We will add it to the line chart visuals on both the **Front Page** and **Monthly Sales Analysis** pages.

Note: You can add the new tooltip page to any visual that you want. This tooltip page has been designed to work off a data point that is filtered by month. So, it would only make sense for use on a visual that displays values filtered by month.

However, a more generic tooltip page could be created and added to any other visual in the report, including the matrix, table, or any chart.

1. Navigate to the **Front Page** page.
2. Select the line chart visual.
3. Click the **Format your visual** button in the **Visualizations** pane to switch to the *Format visual* settings.
4. Click **General** and then click on **Tooltips** to expand the options in that section.
5. Click on the **Page** list and select the **TT Month** tooltip page, as shown in *figure 15.24*:

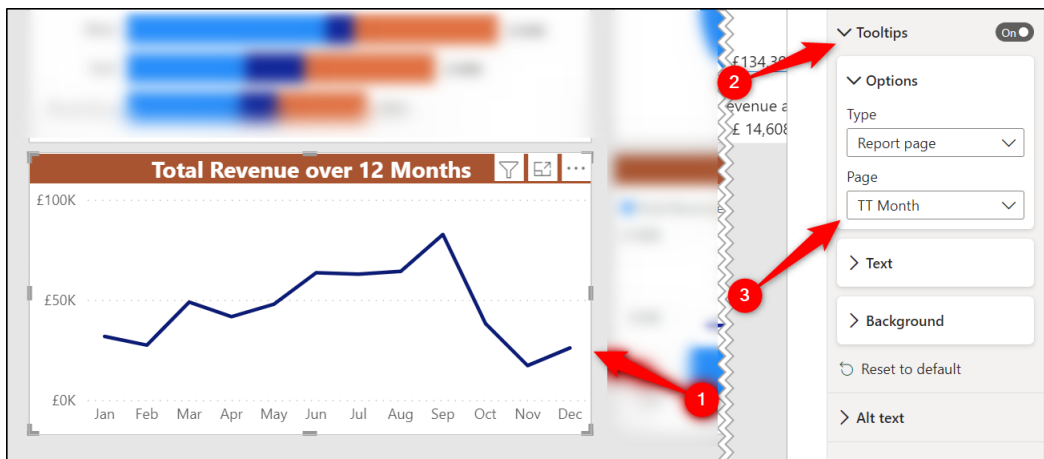


Figure 15.24: Adding a custom tooltip page to a visual

Figure 15.25 shows the tooltip being displayed for the August data point in the line chart. Additional insight into the August sales from a simple hover over:

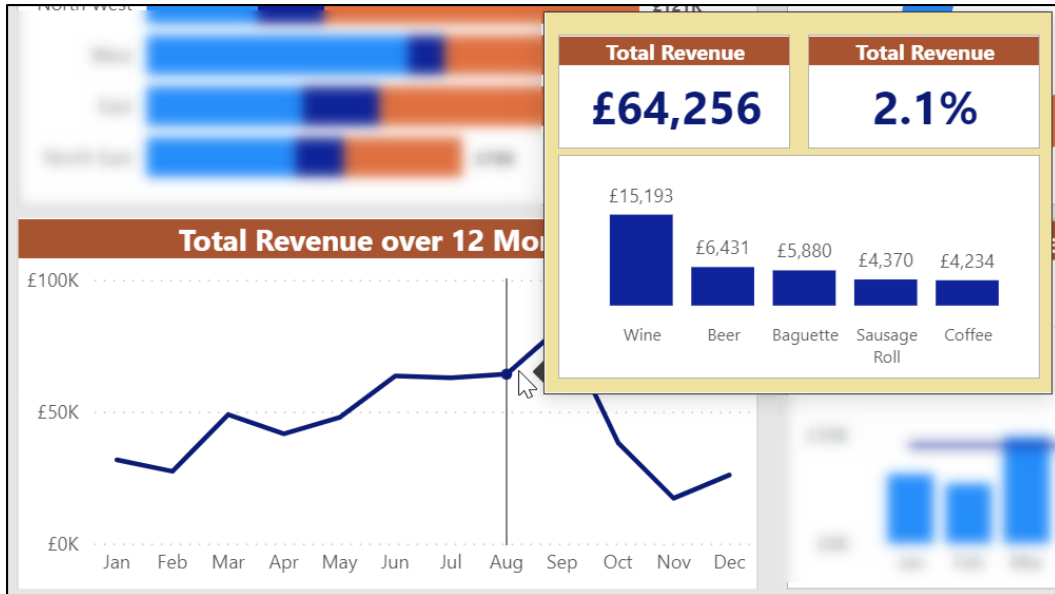


Figure 15.25: Tooltip shown for the August data point

- Repeat Steps 1–4 for the line chart on the **Monthly Sales Analysis** page.

Using bookmarks

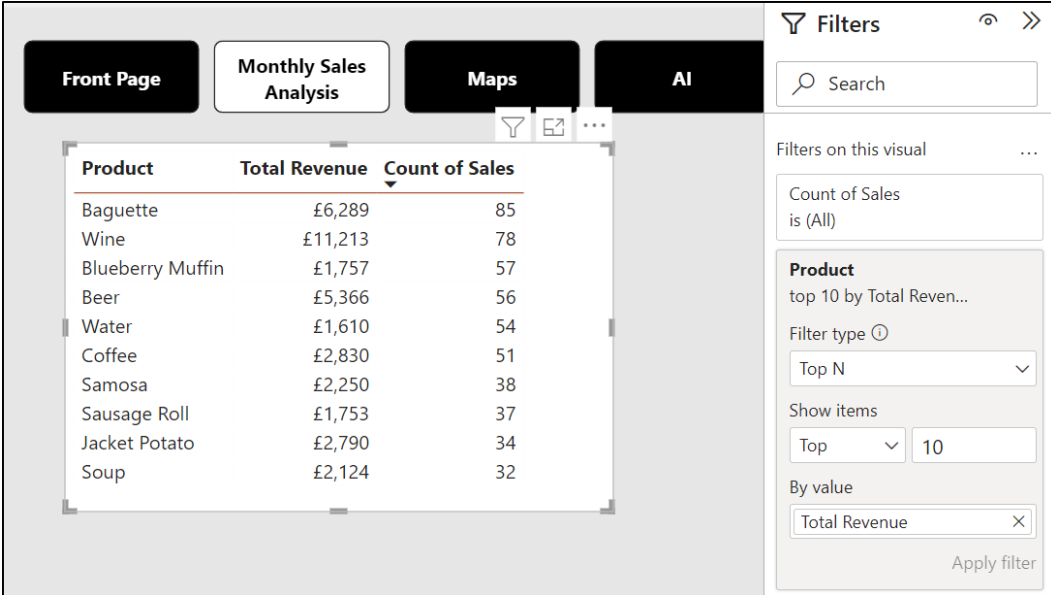
Bookmarks allow you to save the current view of a report page so that you can return to it with the click of a button. Bookmarks can be awkward at times to create, but they are very flexible and powerful.

The view saved by a bookmark includes all applied filters, slicers, sort orders, and cross-highlights between visuals. You can also decide which visuals and other on-page elements are visible in that view.

Although bookmarks save the state of all filters and visuals on a page by default, a bookmark can be changed to apply to only selected visuals. You can also specify a bookmark to ignore data changes such as filters, drill downs, and sort orders and focus on the page display only, or vice-versa.

For this example, we will create two bookmarks that save the state of the table visual only on the **Monthly Sales Analysis** page.

The table currently shows the top 10 products by total revenue, as shown in *figure 15.26*. The products are listed in descending order by the count of sales, shown by the arrow in the column header:



Product	Total Revenue	Count of Sales
Baguette	£6,289	85
Wine	£11,213	78
Blueberry Muffin	£1,757	57
Beer	£5,366	56
Water	£1,610	54
Coffee	£2,830	51
Samosa	£2,250	38
Sausage Roll	£1,753	37
Jacket Potato	£2,790	34
Soup	£2,124	32

Figure 15.26: Table showing top 10 products by total revenue and ordered by the count of sales

We will create a bookmark to save the visual when filtered for the *Top 10* products and another bookmark to save the visual when filtered for the *Bottom 10* products. Both bookmarks will also save the table with the products listed in descending order by the *Total Revenue* column.

The bookmarks will only save the state of the selected visual and ignore other filters on the page.

Buttons will then be inserted to trigger each bookmark and provide the user with a simple method to switch between the filters without having to do it manually in the Filters pane.

Note: When using bookmarks to save the display of a page, the Selection pane is used to show and hide the different visuals and other elements on the page.

Creating bookmarks

Before we create any bookmarks, we need to open the *Bookmarks* pane. To do this, click **View** | **Bookmarks** (*figure 15.27*):

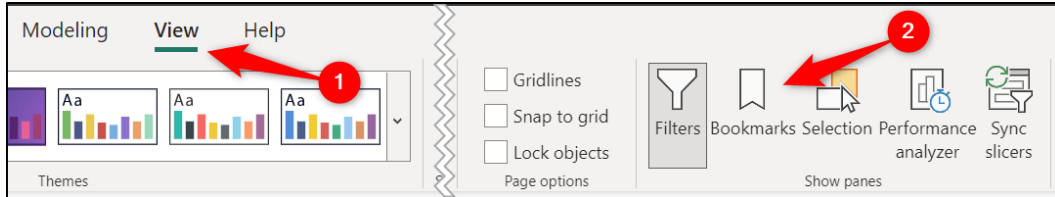


Figure 15.27: Opening the Bookmarks pane

The *Bookmarks* pane opens and is displayed between the **Filters** and **Visualizations** panes by default. There is currently nothing to see there as we have no bookmarks, so let us jump in and create the first one.

The first bookmark is for the *Top 10* products, and that filter is already set. However, we do need to order the products in descending order by total revenue before we add the bookmark.

1. Click on the **Total Revenue** column header to order the products descending by that column.
2. Ensure that the table visual is still selected and click the **Add** button in the **Bookmarks** pane (as shown in figure 15.28).

A bookmark named *Bookmark 1* is created. This bookmark has saved the current view of the entire page respecting all filters, cross-highlights, sort orders, and visuals that are displayed.

3. Click the **More options** ellipsis dots to the right of the bookmark name and click **Rename**, as shown in figure 15.28.
4. Type “**Top 10 Products**” for the bookmark name and press *Enter*.

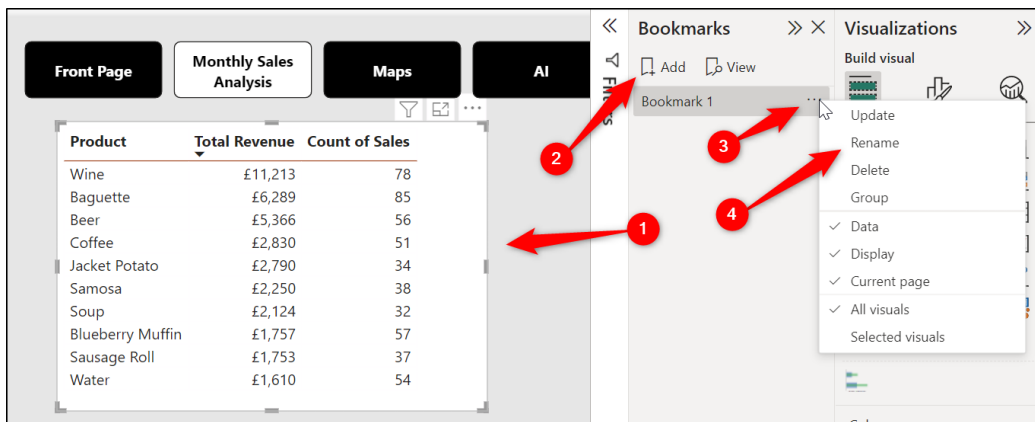


Figure 15.28: Adding and renaming a bookmark

We will now apply the bookmark to the selected table visual only and not all visuals on the page, which is the default setting.

5. Click the **More options** ellipsis dots and click **Selected visuals**, as shown in *figure 15.29*.

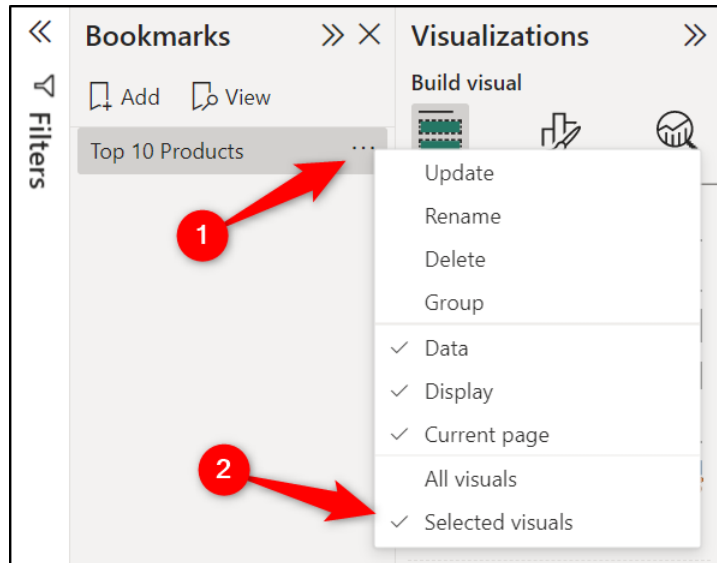


Figure 15.29: Applying the bookmark to selected visuals only

That is the first bookmark created. Let us now change the filter applied to the table and add the second bookmark.

Note: You can edit an existing bookmark by making the necessary filter and displaying changes on the page, and clicking More options | Update.

6. Ensure that the table visual is selected and expand the **Filters** pane.
7. Click the **Show items** list and select **Bottom**.
8. Click **Apply filter**.

Figure 15.30 shows the bottom 10 products filter applied to the table visual using the *Filters* pane.

Product	Total Revenue	Count of Sales
Tea	£1,484	47
Cornish Pasty	£1,442	20
Caramel Shortbread	£1,331	29
Sandwich	£1,294	15
Flapjack	£999	31
Croissant	£760	21
Crisps	£758	36
Orange Juice	£703	29
Hot Chocolate	£691	21
Chocolate Chip Muffin	£374	18

The screenshot shows a Power BI report interface with a navigation bar at the top containing 'Front Page', 'Monthly Sales Analysis', 'Maps', and 'AI'. A table visual is displayed in the center, showing a list of products with their total revenue and count of sales. The table is filtered to show the bottom 10 products by total revenue. On the right side, the 'Filters' pane is open, showing the filter applied: 'Product bottom 10 by Total Revenue'. The filter type is set to 'Top N' and the number of items is set to 10. The filter is applied to the 'Total Revenue' field.

Figure 15.30: Filtering the table to show the bottom 10 products

Let us now add the second bookmark to save the view of this table.

9. In the **Bookmarks** pane, click **Add**.
10. Click the **More options** ellipsis dots beside the bookmark name and click **Rename**. Type “**Bottom 10 Products**” for the bookmark name and press **Enter**.
11. Click the **More options** ellipsis dots again and click **Selected visuals**.

With both bookmarks created, you can test their functionality by clicking their name in the *Bookmarks* pane. When you click the bookmark, it will change the table to the relevant saved view.

Notice that the slicer on the page that contains the list of months and filters all visuals on the page is ignored by the bookmarks. This is because we specified that the bookmark is applied to the table visual only; therefore, it did not save the filter applied by the slicer.

You can check this by clicking a different month in the slicer than the one that was active when you added the bookmark and then clicking a bookmark name. The slicer selection is not changed. This proves that the bookmark did not save the slicer selection as part of the saved view.

Inserting buttons to run the bookmarks

Clicking a bookmark name in the *Bookmarks* pane is fine to test their functionality, but it is not how we want readers of the report to switch between the bookmarks.

We will insert two buttons on the page and then assign them the action to apply the relevant bookmark.

1. Click **Insert** | **Buttons** | **Blank** (there is a *Bookmark* button, but any button, picture, or shape can be used to trigger a bookmark.)
2. Move the button beside the table and resize it as required.
3. In the **Format** pane, click **Style** and click the **Text On/Off** slider to turn it **On**.
4. Type “**Top 10**” in the *Text* box.
5. Format the button with the fill, text, and border settings of your choosing.
6. Copy the button and change the text to “**Bottom 10**”.

Now, we will apply the actions to trigger the bookmarks.

7. Select the *Top 10* button, and click the **Action On/Off** slider to turn it **On**.
8. Click the *Type* list and select **Bookmark**.
9. Click the **Bookmark** list and select *Top 10 Products*.

Figure 15.31 shows the Top 10 Products bookmark action being assigned to the first button.

10. Repeat Steps 7–9 for the *Bottom 10* button but select the **Bottom 10 Products** in the *Bookmark* list.

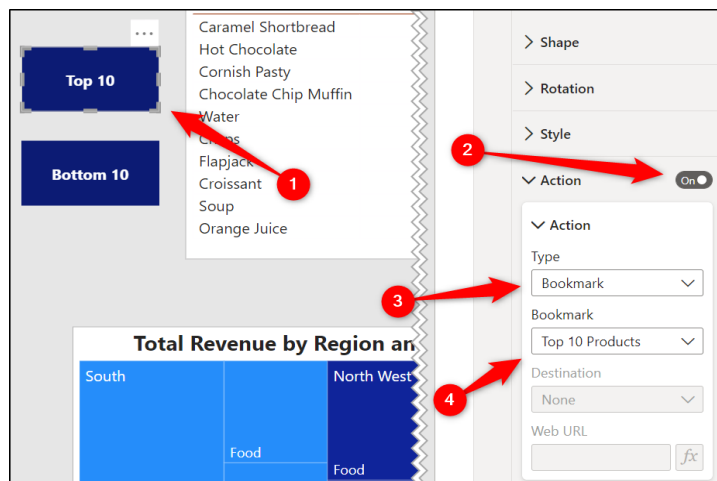


Figure 15.31: Setting a bookmark action for a button

The buttons can now be used to switch between the two views of the table. Remember, while in Power BI Desktop, you need to press *Ctrl + click* to follow a link.

Save the report as **sales-report.pbix**, ready for publishing to the Power BI service in the next chapter.

Conclusion

In this chapter, we learnt how to use text boxes, images, and buttons to enhance the user experience with our reports. These elements help to highlight key functionality to a reader and aid navigation.

We then learnt how to create custom tooltip pages to provide additional insights on a data point quickly. This effective technique is simple to set up and allows you to design the tooltips that you need for your visuals.

Finally, bookmarks were covered. Another fantastic feature to improve the interactivity of a report and to enable a reader to change the filters or switch visuals with the click of a button.

In the upcoming chapter, we will publish the report to the Power BI service, look at how to create new reports from existing datasets and see how to share our reports with others.

Questions

Here are some questions to test what you have learnt in this chapter.

1. Which of the following is **not** an action that can be assigned to an image or button?
 - a. Drill through.
 - b. Back.
 - c. Next.
 - d. Bookmark.
2. Which of the following are found in the *More options* list for bookmarks?
 - a. Update.
 - b. Display.
 - c. Selected visuals.
 - d. All of the above

3. **An image can be used for the *Canvas background*.**
 - a. True
 - b. False

4. **A custom tooltip page can be applied to more than one visual in a report.**
 - a. True
 - b. False

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



CHAPTER 16

Publishing and Sharing Your Reports

Introduction

It is now time to publish the report and share it with others. This is simple to do in Power BI, and there are many options for sharing and collaborating on reports. We will cover the primary method to share reports in this chapter and learn more in *Chapter 18, Associated Tools with Power BI*.

We begin the chapter by seeing how to publish the report to the Power BI service, discuss what was published, and how to access and view that report.

Then, we explore how to create new reports from existing datasets that were published to the service. We see how to do this from within the Power BI service and from Power BI Desktop.

Finally, we share the report with others in your organization using a link that can be distributed via e-mail or some other medium. We also cover managing the permissions that others have to a report.

Structure

In this chapter, we will cover the following topics:

- Publishing a report to the Power BI service

- Accessing the Report
- How to create a new report from an existing dataset
- Sharing a report with others and managing their permissions

Objectives

After reading this chapter, you will know how to publish a report to the Power BI service, access the report, and share it with others.

You will understand how to navigate the Power BI service and know the differences between a report, a dataset, and a workspace.

You can share your Power BI report by sending a link to others and managing their permissions.

Publishing your Report

Files: **sales-report.pbix**

The report is created, and it is time to publish it to the Power BI service. Once published, it can be distributed and collaborated on with others. It will also be available to other apps such as Excel and PowerPoint (we will cover this in detail in *Chapter 18, Associated Tools with Power BI*).

When publishing to the service, you will be prompted to publish to a workspace. This is a repository on the service to save your reports, datasets, dashboards, and Excel workbooks. You can think of a workspace as a folder, just like you get on a computer to save your documents and images.

1. Click **Home** | **Publish** in Power BI Desktop, as shown in *figure 16.1*:

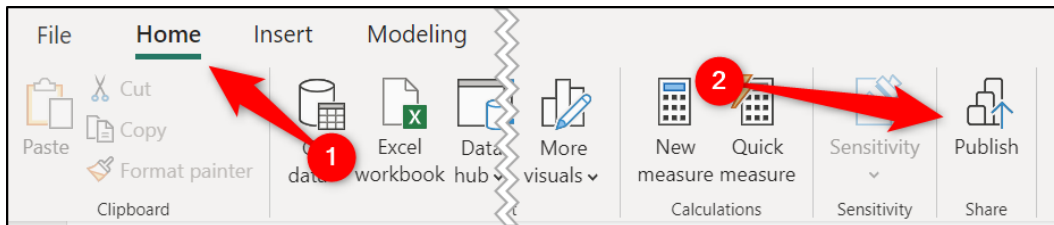


Figure 16.1: Publish a report to the Power BI service

2. If the report has not been saved recently, you will be prompted to save changes before publishing. Click **Save** (as shown in *figure 16.2*).

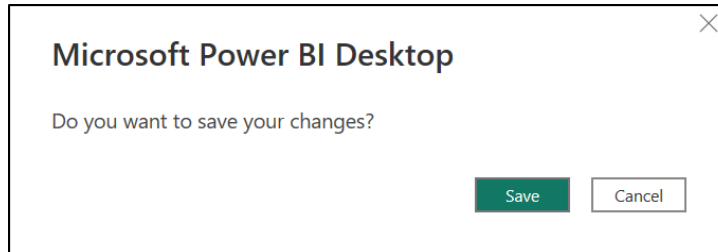


Figure 16.2: Prompt to save changes to your report

3. In the *Publish to Power BI* window (refer to figure 16.3), the workspaces that you have access to are listed. Click the workspace you want to publish to and click **Select**.

In this example, I am publishing to the *Management Team* workspace. In the upcoming chapter, we will discuss how to create workspaces and manage how others access them (known as permissions). Please refer to the following figure:

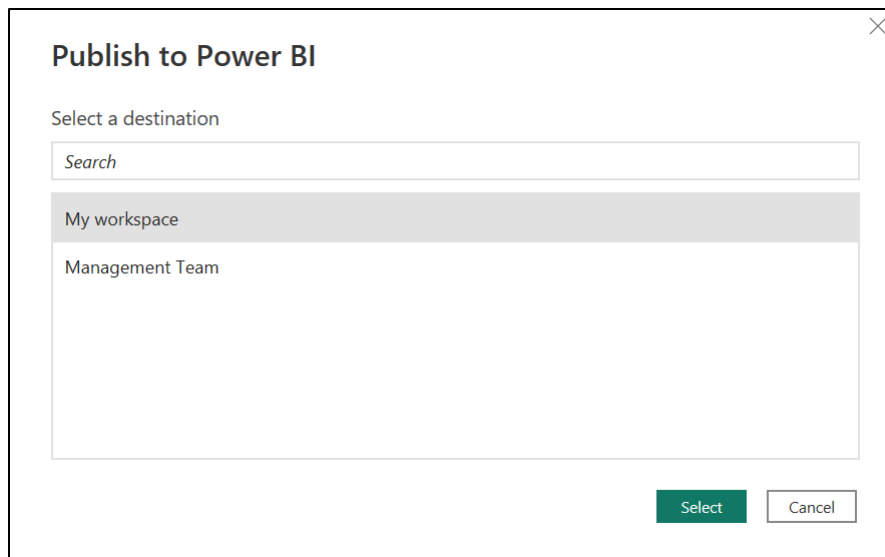


Figure 16.3: Choose a workspace to publish to

4. A message confirming that the publish was successful is shown when the publish is complete (as shown in figure 16.4).

The window also provides links to open the published report in the service and to get quick insights. Click the **Open 'sales-report.pbix' in Power BI** link, or if you do not want to see the report online just yet, click **Got it**.

Next, we will cover how to access the report in the Power BI service. Please refer to the following figure:

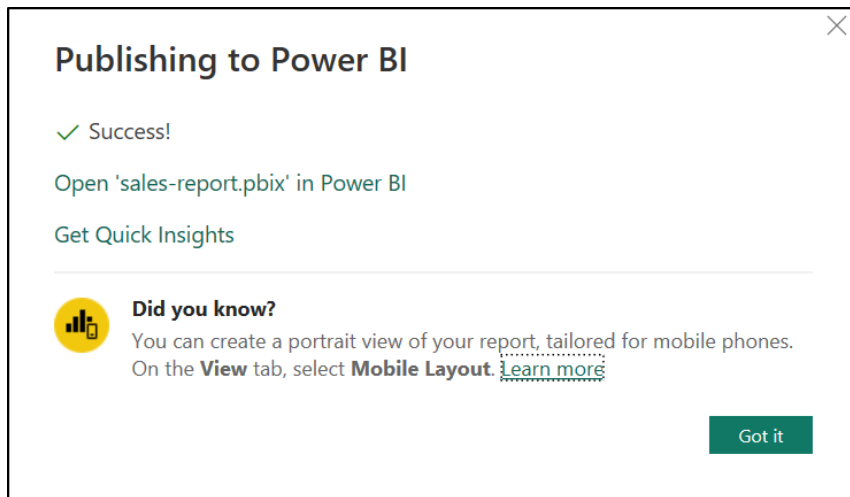


Figure 16.4: Success in publishing report

Accessing the report on the service

With the report now live in the Power BI service, you will want to know how you can access it. Let us have an orientation to the service and see how to find the reports before we look at sharing reports later in this chapter.

Note: We will only cover using the Power BI service with a Web browser in this book, but you can also access your workspaces and reports easily using the Power BI app for mobile. Simply download the app, log in, and navigate using the menu at the bottom of the screen.

1. Open a Web browser and type “**powerbi.com**” into the address bar.

You are taken to the home page of the Power BI service (as shown in *figure 16.5*).

The recommended links at the top of the page include frequently accessed areas of the service and tutorials on aspects of Power BI.

Following the recommended links is a list of the workspaces, reports, dashboards, apps, and datasets that you have access to. You can filter the list for the most recently used, those marked as a favorite, or search using keywords.

The menu on the left allows you to navigate the service and access your workspaces, apps, or create new reports from published datasets.

2. Click **Workspaces** in the menu on the left (refer to *figure 16.5*):

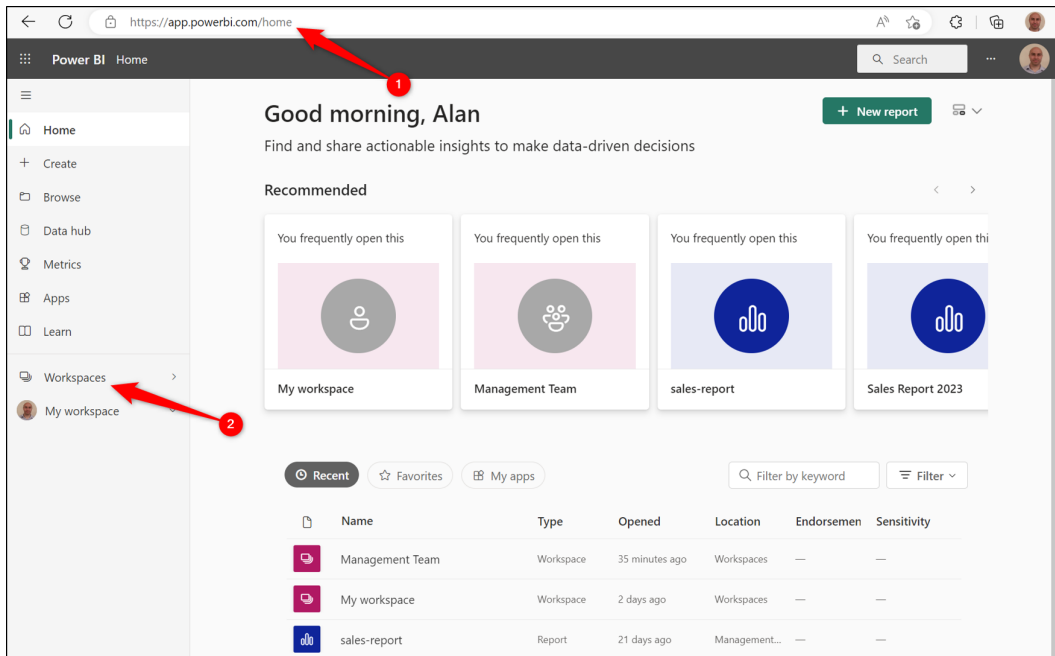


Figure 16.5: Homepage of the Power BI service in a Web browser

3. A list of workspaces is shown. Click on the workspace where the report was published (as shown in *figure 16.6*). In this example, that is my *Management Team* workspace.

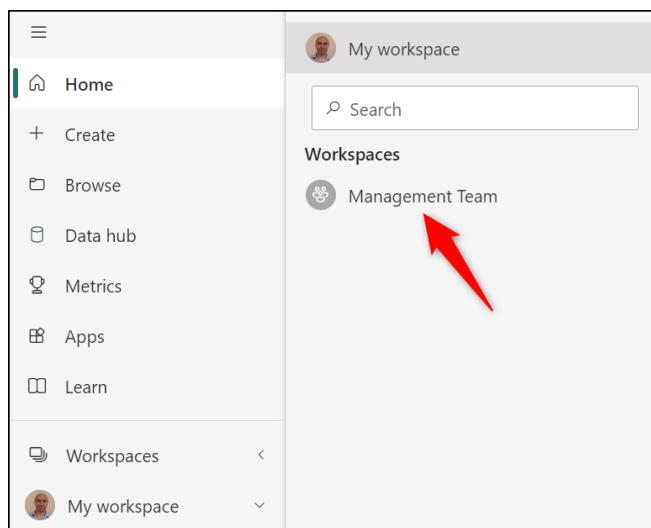


Figure 16.6: Accessing the Management Team workspace

4. Within the workspace, click **sales-report** to open it (as shown in *figure 16.7*).

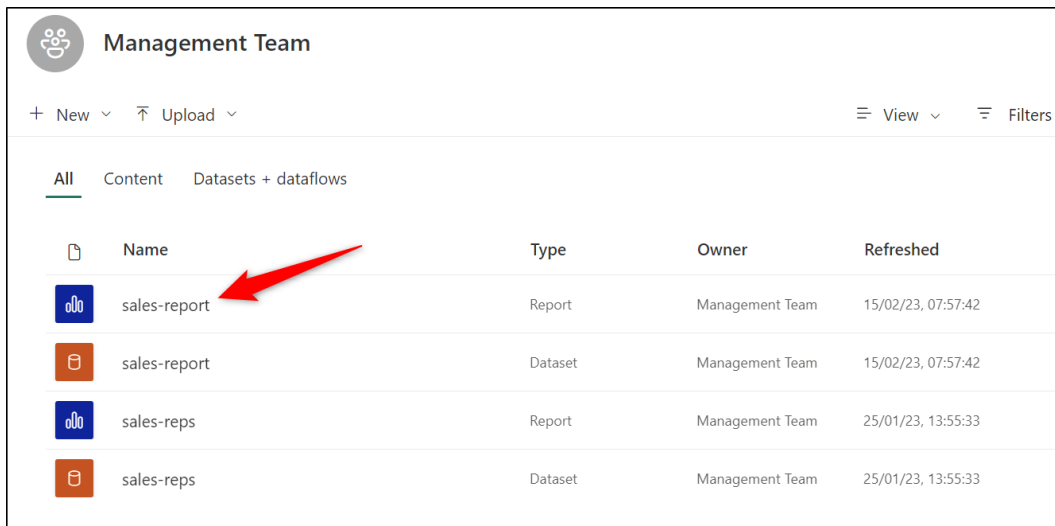


Figure 16.7: Sales report in the Management Team workspace

Figure 16.8 shows the *sales report* in the Power BI service. Notice the following areas of the page:

- A link in the top left to return to the workspace where the report is stored.
- The date that the report was last updated in the center of the title bar at the top of the window.
- A list of the report pages on the left of the page currently being viewed.
- The *Filters* pane on the right.
- Buttons to export, share, edit, and collaborate in a toolbar at the top of the report page.
- The Power BI service menu down the left of the window to jump to a different workspace, app, and more.

The report is fully interactive, and all the links, bookmarks, visual interactions, slicers, tooltips, and so on covered throughout this book function as expected.

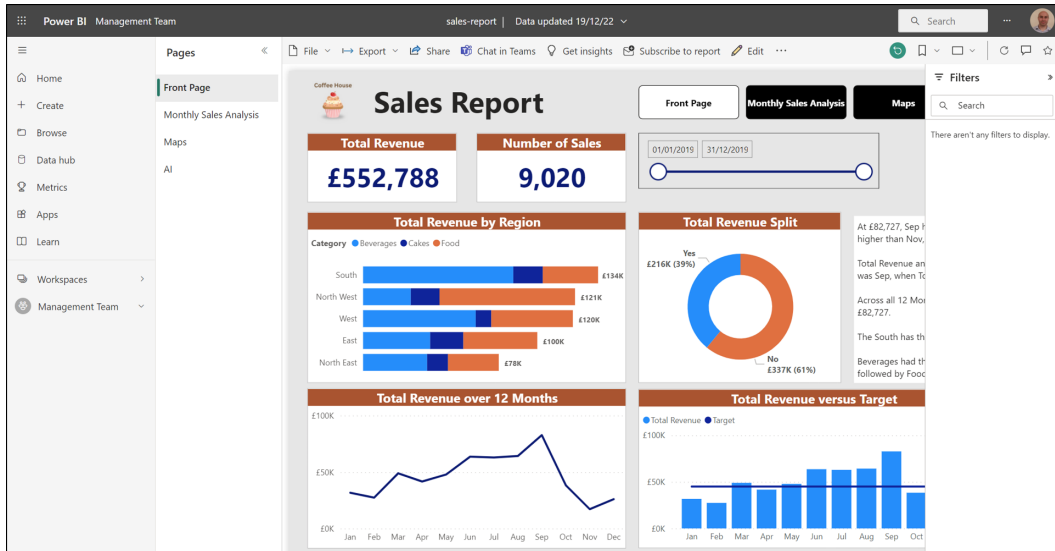


Figure 16.8: Sales report in the Power BI service

Datasets and reports

When you publish a report to the Power BI service, it is not only the report that is published but also the dataset.

Figure 16.9 shows the report and dataset listed in the workspace. These are shown in the *All* category, but you can also see a separate *Datasets + dataflows* category.

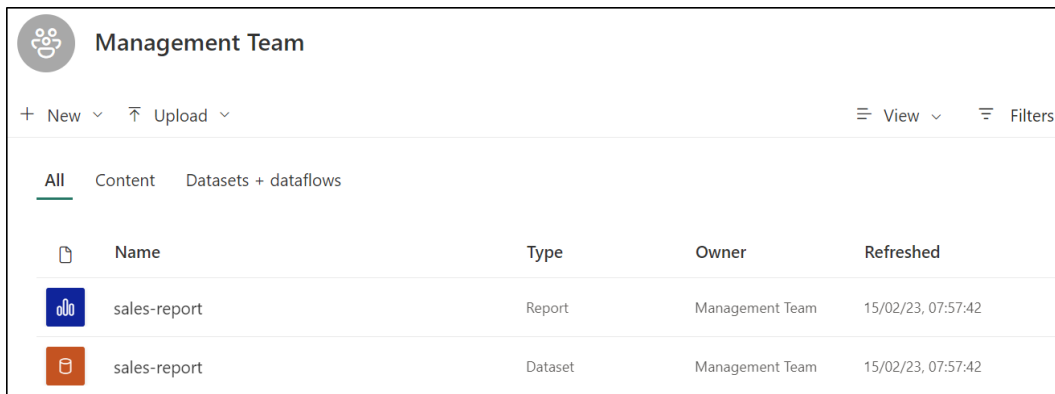


Figure 16.9: Report and dataset published to the workspace

The dataset is the collection of data that was built into a cohesive data model earlier in this book. It consists of the data source connections, the query transformations, the relationships, and DAX measures.

Essentially, the dataset is everything except the visuals and filters built in the Power BI Desktop report view.

This dataset can be used by other Power BI reports and even exported into Excel and used my PivotTables to analyze and summarize the data.

Let us have a look at how you can create new reports in Power BI using an existing dataset that has been published to the service.

Creating a report from an existing dataset

We will look at how to create reports using an existing dataset from within the Power BI service and from Power BI Desktop.

Both methods are quick and straightforward and establish a live connection to the entire dataset. You will have access to all the tables, fields, and measures that you have permission to use.

From the Power BI Service

Following are the steps to create a report from the Power BI Service:

Note: There are multiple options for creating a new report. These include clicking the **New report** button in the top right corner of the window, or the **Create** button in the menu down the left of the window. You could also create a new report from the workspace where you are planning to save the new report.

1. Click **Create** in the menu down the left (as shown in *figure 16.10*).

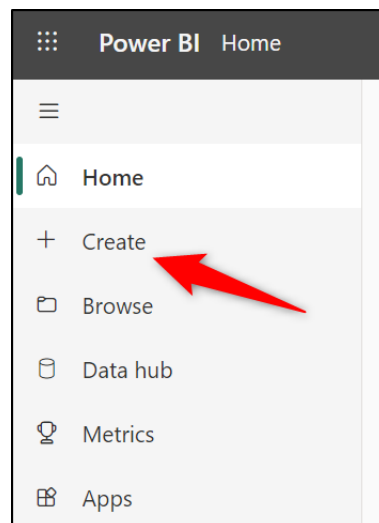


Figure 16.10: Creating a new report in the Power BI service

2. Click **Pick a published dataset** (refer to *figure 16.11*).

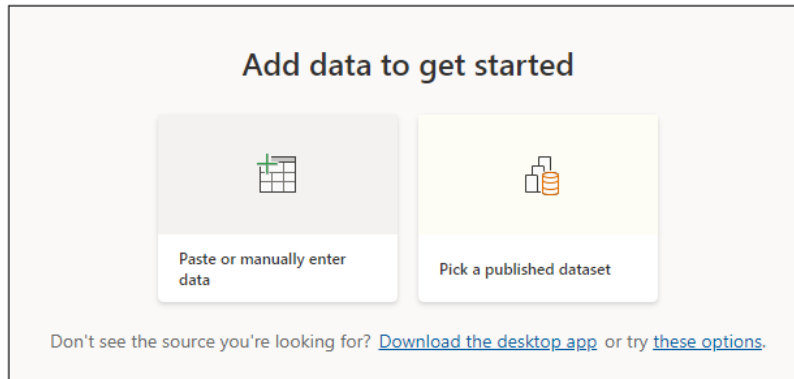


Figure 16.11: Choosing the data to get started with the report

3. A window appears listing all the datasets for which you have access (as shown in *figure 16.12*). Select the dataset to use for the new report, click the list arrow beside the *Auto-create report* button, and click **Create a blank report**.

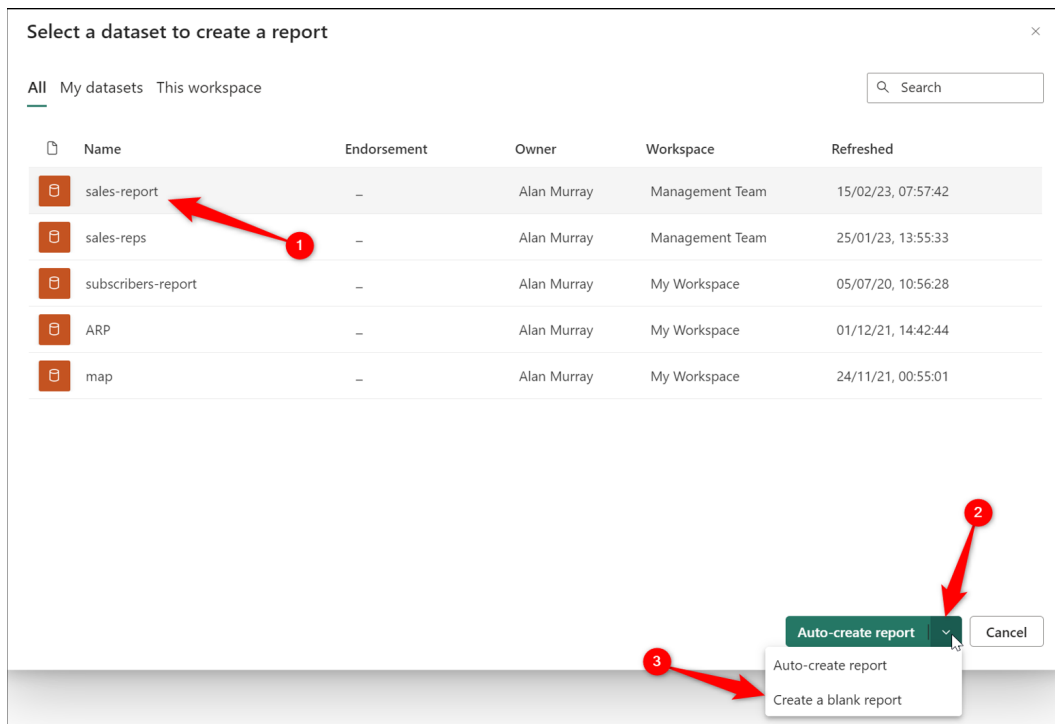


Figure 16.12: Selecting the dataset to use for the report

4. A blank report is created, and you will see the **Filters**, **Visualizations**, and **Fields** panes to the right of the report pages (as shown in *figure 16.13*).

The toolbar above the report provides buttons to insert text boxes, shapes, buttons, and images. The *Edit interactions* option is on the toolbar so that you can specify how visuals cross-filter and cross-highlight each other. And there is a *View* menu, enabling you to scale the page and show panes such as the bookmarks and selection panes.

The experience is like that of creating reports in Power BI Desktop.

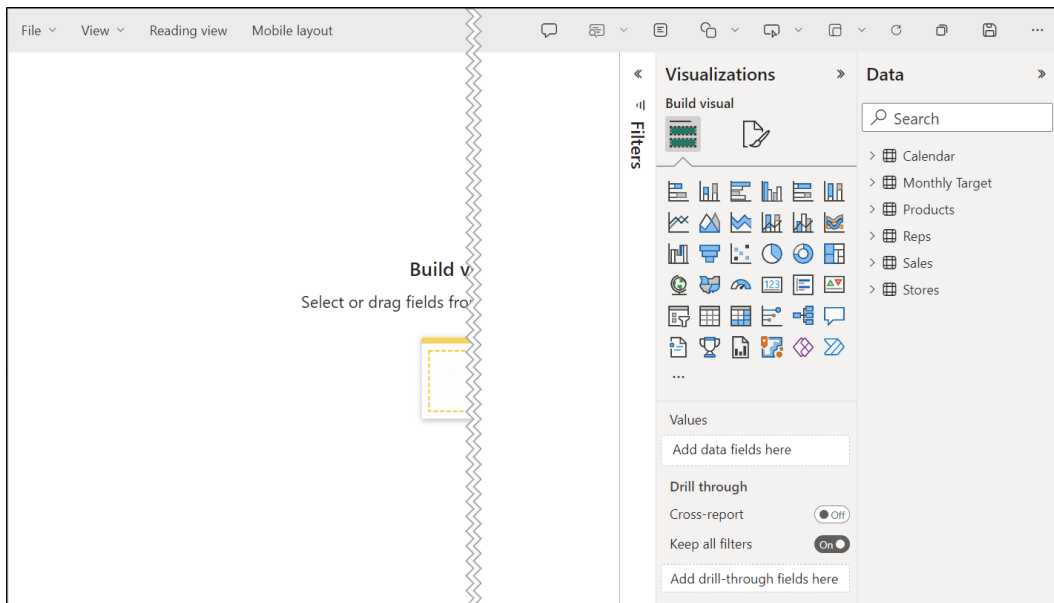


Figure 16.13: Report creating experience in the PBI service

- To save the report, click **File** | **Save** from the toolbar (as shown in figure 16.14). The **Save as** option can be used to save a copy of the report in another workspace.

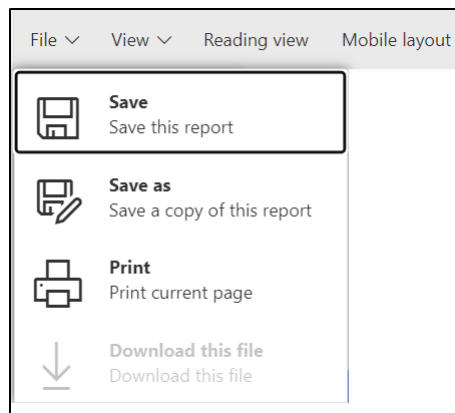


Figure 16.14: Saving the report in the PBI service

- In the *Save your report* window, type a name for the report and select a destination workspace (as shown in *figure 16.15*). Click **Save**.

The destination workspace can be any workspace you have access to. The dataset and report do not need to exist in the same workspace. The dataset is not copied. Only the report is saved and connected to the live dataset.

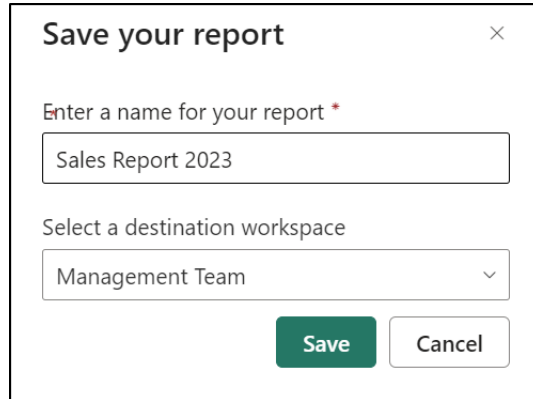


Figure 16.15: Entering a name for the report and a destination workspace

From Power BI Desktop

Following are the steps to create a report from the Power BI Desktop:

- Click **Home** | **Get data** | **Power BI datasets** (as shown in *figure 16.16*) or click **Home** | **Data hub**.

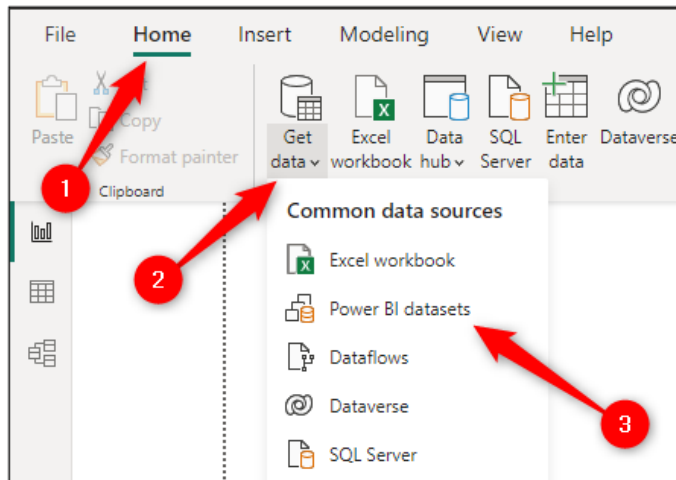


Figure 16.16: Using an existing Power BI dataset in Power BI Desktop

2. In the *Data hub* window, select the dataset that you want to use and click **Connect** (as shown in *figure 16.17*).

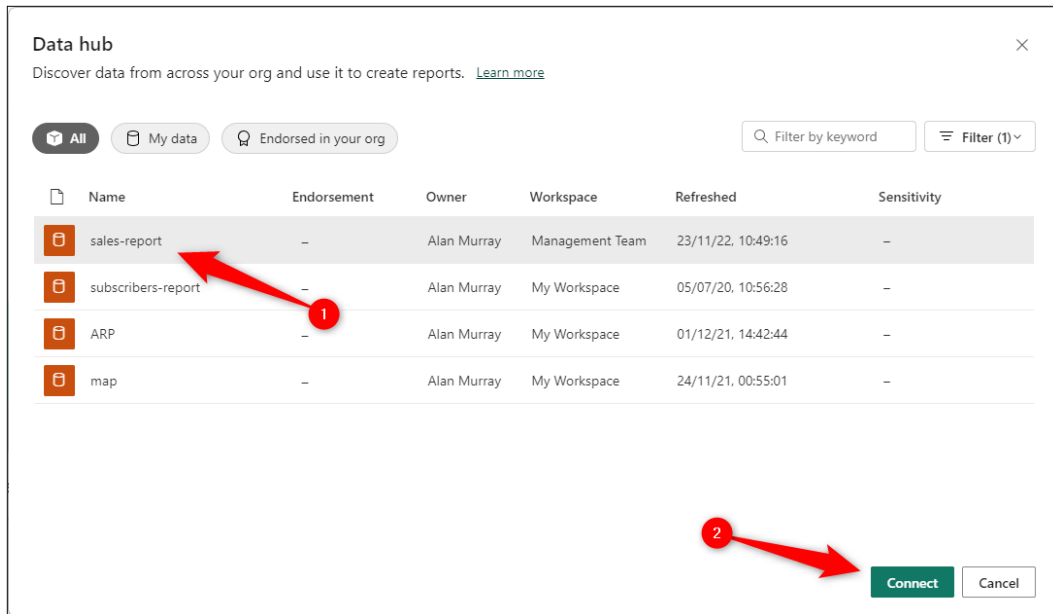


Figure 16.17: Data hub showing all datasets

A blank report is created (*figure 16.18*), and you can add visuals, filters, bookmarks, and so on, as discussed throughout this book.

A live connection to the data is created. You do not have access to the data, but you can create new measures. You will notice that there is no *Data* view and that some functionality is also missing from the Ribbon.

When the report is created, it can be published in the same workspace as the dataset or in a different workspace.

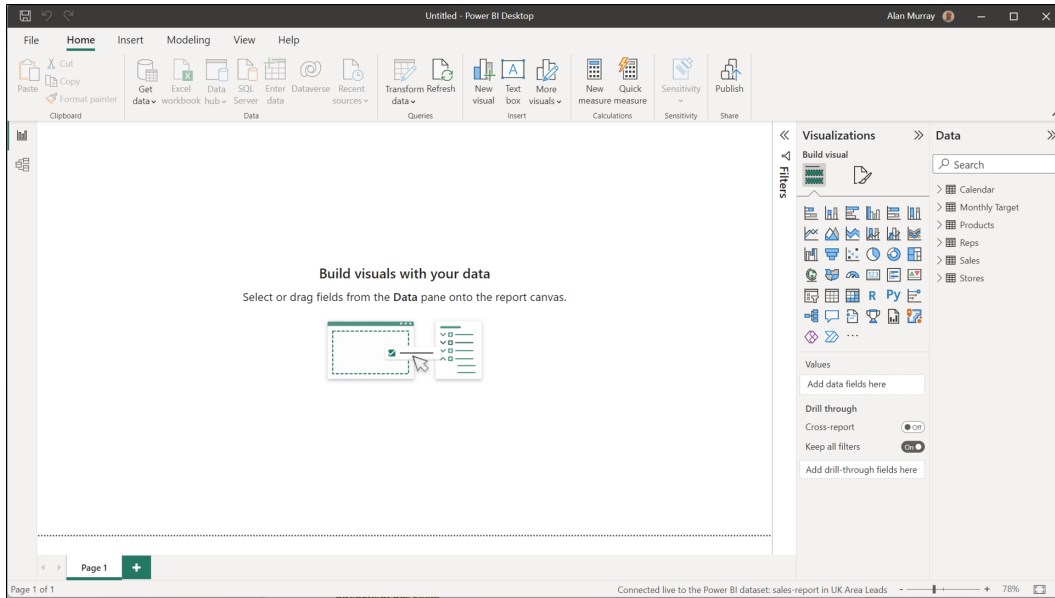


Figure 16.18: Blank report connected to a live Power BI dataset

Share a link to the report

You can easily share Power BI reports with others within and outside your organization in the Power BI service.

When you share the report with others, they can interact with the report fully but cannot edit it. They will also have access to the dataset and can reshare the report with others unless you take the necessary steps to prevent them.

Note: To allow others to edit a shared report, you can assign a workspace role that permits this. We discuss workspaces and workspace roles in *Chapter 17, Datasets, Dashboards, and Reports*

1. From within the report in the Power BI service, click the **Share** button on the report toolbar (as shown in *figure 16.19*).

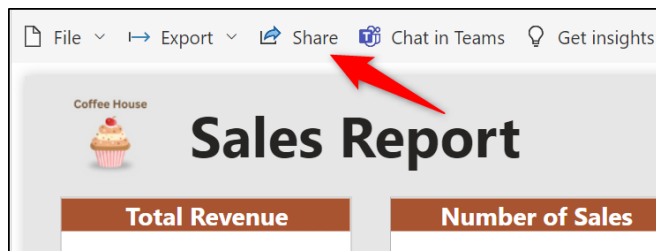


Figure 16.19: Sharing a report in the Power BI service

2. In the *Send link* window (as shown in *figure 16.20*), it defaults to sharing with people in your organization. To continue with this option, enter the e-mail addresses of the recipients with whom you want to share the report.
3. Add a message, though this is optional.
4. Uncheck the **Include my changes** box. This creates a shared view so that the recipients will see your changes, such as filter selections and drilling. It expires in 180 days.

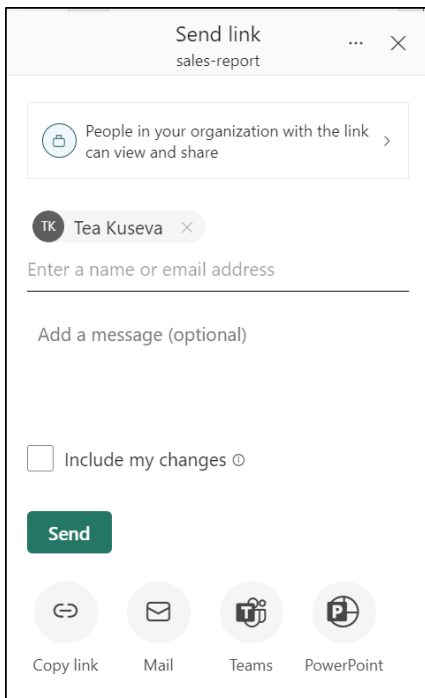


Figure 16.20: Send link window to enter recipients and choose the sharing method

5. Click **Send** or use one of the options below to **Copy link** or use **Teams**.

Figure 16.21 shows the confirmation message you receive when clicking the **Send** button to share the report. The other methods will take you to a different window to share the copied link or enter recipients in Teams, and so on:

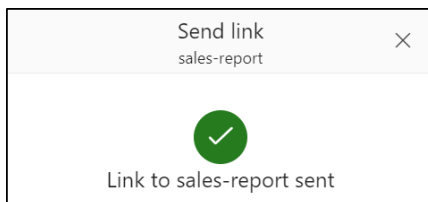


Figure 16.21: confirmation that the link was successfully sent

To change how the report is shared, from the **Send link** window (refer to *figure 16.20*), click the **People in your organization with the link can view and share**.

Three options are shown for you to specify who you want the link to work for (as shown in *figure 16.22*):

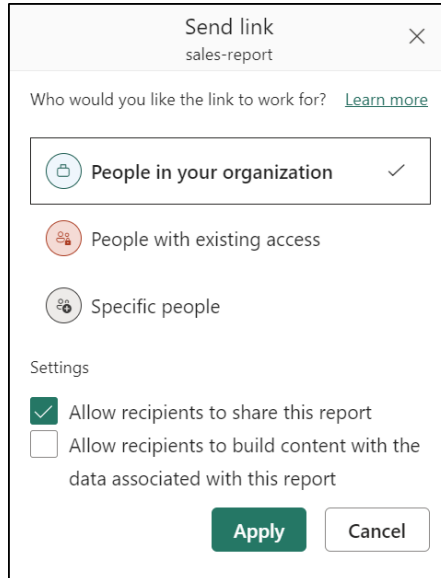


Figure 16.22: Choose how to share the report and other settings

- **People in your organization:** This link provides access to view and share the report by default but does not work for external users of the organization.
- **People with existing access:** This option generates a URL to share with users who already have access to the report. It does not grant access.
- **Specific people:** This option allows specific people and groups access to the report. It can be shared with external users to the organization to grant access, but they would need to be added as guests to the organization's Azure Active Directory.

Specify whether you are happy with recipients being able to share the report or building content with data associated with the report and click **Apply**.

Managing report permissions

You can manage users' access and any links that have been created at any time.

1. Navigate to the workspace where the report resides.

- Click the **More options** ellipsis to the right of the report name and click **Manage permissions** (as shown in *figure 16.23*).

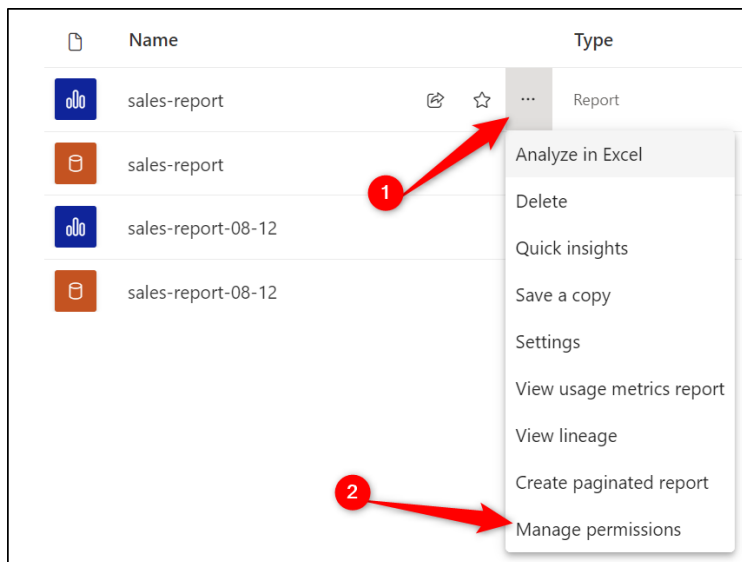


Figure 16.23: Manage the permissions for a report

Figure 16.24 shows the manage permission window showing the list of links that have been generated and shared with others.

You can manage the access that the link provides, copy the link, or delete it via the ellipsis dots beside the link URL.

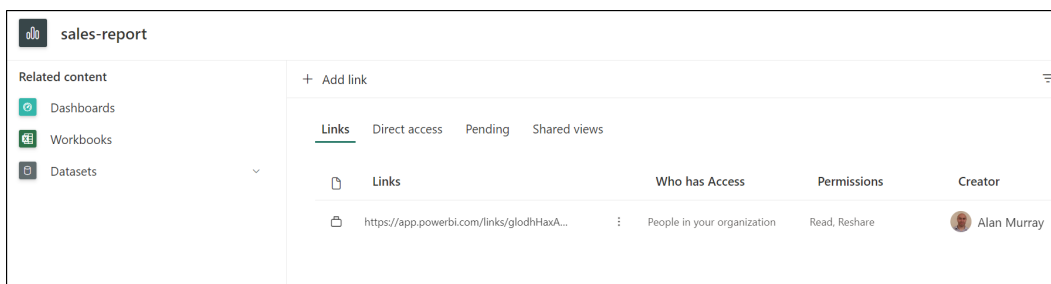


Figure 16.24: List of links that have been created.

You can also add links to grant access for others using the **Add link** button at the top of the window. This would take you to the same **Send link** window that was discussed previously.

There are also menu items for managing direct access that people have, viewing access requests, and managing shared views (created when the **Include my changes** option is selected in the **Send link** window).

Conclusion

In this chapter, we learnt how to publish a report to the Power BI service and how to navigate the service to view your reports and create new reports from existing datasets.

We then learnt how to share reports with others both inside and outside of your organization. Then manage the permission that others have to access your reports.

We will explore workspaces, datasets, and dashboards in the upcoming chapter. We start with learning how to create new workspaces and manage permissions to them.

Next, you will learn how to refresh a dataset from the Power BI service. In addition to manually refreshing datasets, you can also set up scheduled refreshes. As part of this process, you will also learn about installing and establishing data gateways.

We finally dive into dashboards. You will understand the differences between a report and a dashboard, learn how to create dashboards, and share them with others.

Questions

Here are some questions to test what you have learnt in this chapter.

1. **When publishing a report from Power BI Desktop, what is published?**
 - a. Dataset and a dashboard.
 - b. Report, dataset, and dashboard.
 - c. Report and the dataset.
 - d. None of the above

2. **What is a Power BI workspace?**
 - a. A repository on the service to save your reports, datasets, dashboards, and Excel workbooks.
 - b. A report that we can pin visuals from different reports.
 - c. A temporary shared view available for only 180 days.
 - d. It is an alternative name for an Excel workbook when published to the service.

3. **You can create new reports from existing datasets that have been published to the service.**
 - a. True
 - b. False

4. **When you share the link to a report, the recipients can view and interact with the report but cannot edit it.**
 - a. True
 - b. False

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



CHAPTER 17

Datasets, Dashboards, and Reports

Introduction

In this chapter, we will dive into more details on some of the key features of the Power BI service —datasets, dashboards, and reports. We will also cover the important topic of how to refresh your data and your reports.

We start with workspaces and see how to create workspaces in the Power BI service and allow others access to them. By setting permissions, you can determine the features that others have in that workspace; for example, are they allowed to add reports?

Next, we see how to refresh the datasets and reports. We will install a gateway and cover setting up scheduled refreshes in the Power BI service.

Finally, we visit dashboards. What is a dashboard? And how does it differ from a report? We will learn how to create dashboards and share them with others in the organization.

Structure

In this chapter, we will cover the following topics:

- How to create a workspace in the Power BI service and allow access to others

- Why you may need a gateway and how to install one
- Refreshing your data and setting scheduled refreshes
- The difference between dashboards and reports
- How to create a dashboard, pin visuals, and share it with others

Objectives

After reading this chapter, you will know how to create workspaces and dashboards in the Power BI service and how to allow specific access to others in your organization.

You will also learn how to refresh the data from the service, including scheduled refreshes, and how to install and set up a gateway, if required.

Workspaces

Workspaces in Power BI are areas where your colleagues and you can save, share, and collaborate on reports, dashboards, datasets, and Excel workbooks.

Let us look at how you can create new workspaces and then share and manage others' access to them.

Creating a new workspace

We will create a new workspace named *UK Area Leads*. We will then manage access to this workspace and copy the sales report to it.

1. Click **Workspaces** in the Power BI service menu down the left of the window.
2. A sub-menu pops out, listing all workspaces that you are a member of. Click **Create a workspace** at the bottom of the sub-menu (as shown in *figure 17.1*):

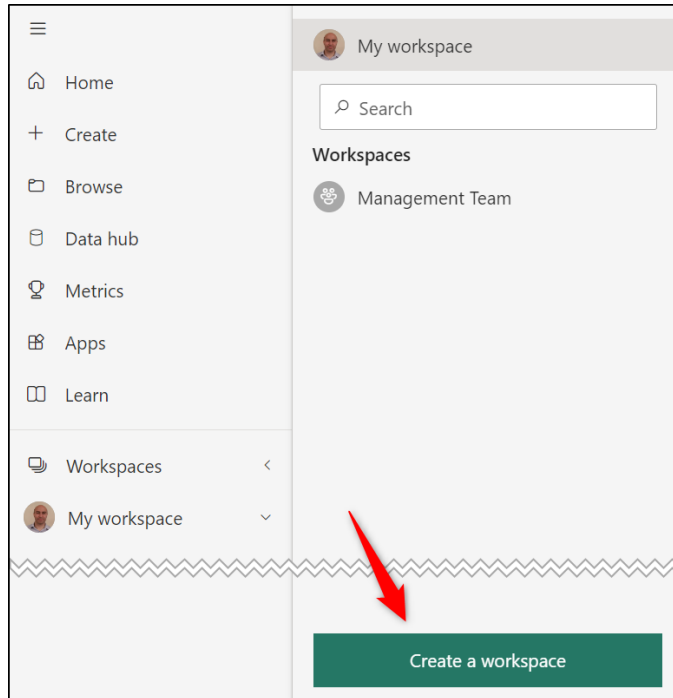


Figure 17.1: Creating a new workspace


3. Type **UK Area Leads** as the **Workspace name** and enter a **Description**, although the description is optional (refer to *figure 17.2*).

Note: You can give access to other users within the **Advanced** section of the **Create a workspace** pane. However, we will discuss how to manage access after creating the workspace.

4. Click **Save**, as shown in the following figure:

Create a workspace

Workspace image

 Upload
Delete

Workspace name *

UK Area Leads

Available

Description

This workspace is for the UK area leads to see the reports, dashboards and data related to the product sales in the UK.

[Learn more about workspace settings](#)

Advanced ▾

Save Cancel

Figure 17.2: Enter a workspace name and description

The workspace is created and can be found in the list of workspaces that you are a member of.

Let us see how you can now manage others' access to this workspace.

Managing access to a workspace

You can allow others access to a workspace, making it easy to share and collaborate on multiple reports, dashboards, and datasets.

A workspace role is assigned to each user or group of users that you assign access. The different workspace roles are Admin, Member, Contributor, and Viewer.

- **Admin:** An admin can do everything, including updating and deleting the workspace and adding and removing user's access to the workspace.
- **Member:** Lots of capabilities, including creating and editing content and copying reports. A Member can also add users and change permissions but only for Contributors and Viewers (permissions lower than the Member role).

- **Contributor:** As the name suggests, a contributor can publish reports to the workspace, create, edit, delete, and copy reports, but they cannot edit permissions for other users.
- **Viewer:** The viewer role means that users can only view and interact with workspace items.

Note: We discussed how to share a specific report with others in the previous chapter. Power BI offers multiple methods for sharing to cater to different scenarios. By sharing a workspace, you can manage access to multiple reports and dashboards for groups of users quickly and easily.

To manage access to a workspace, follow the following steps:

1. Click **Workspaces** in the Power BI service menu and then click the workspace name that you want to manage access.
2. Click the **Access** button in the workspace menu (as shown in *figure 17.3*):

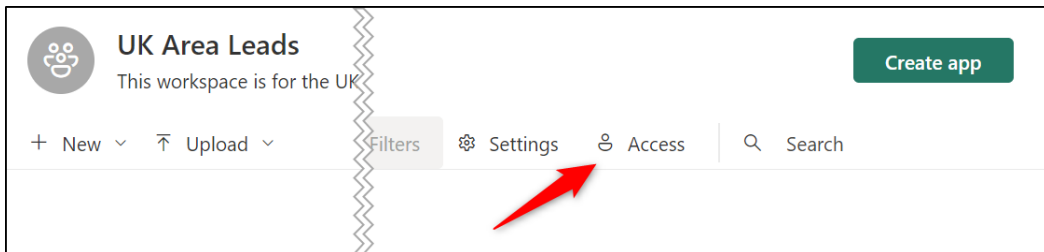


Figure 17.3: Manage access to a workspace

3. An **Access** panel appears. Enter the e-mail address of the user or group you want to add, and then select the workspace role to assign from the list provided.

Figure 17.4 shows a user named Tea Kuseva, that has been added as a Contributor, and a group named UK South, which is in the process of being added with Viewer-only permission.

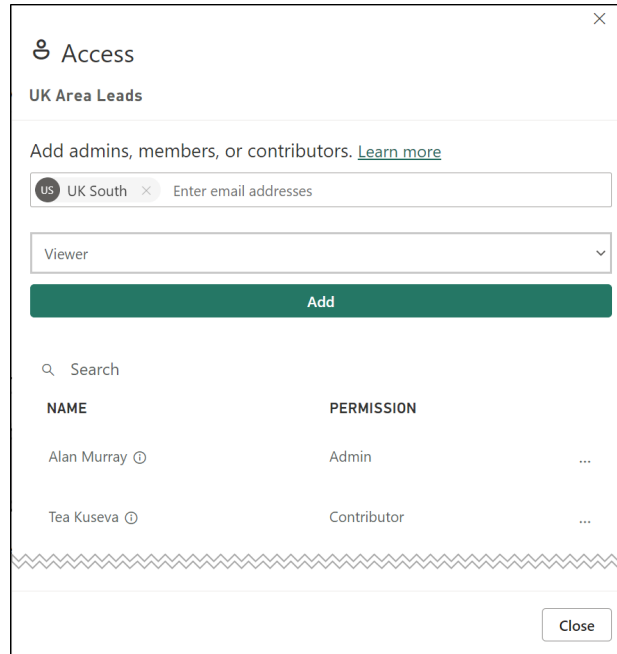
4. Click **Close**.

Figure 17.4: Assigning users and groups access to a workspace

You can change access to a workspace at any time. This can include adding new users and groups, removing the access of existing users and groups, or changing the workspace role of existing members.

To edit the permission of existing users and groups, follow the following steps:

1. Re-open the **Access** panel by clicking **Access** on the **Workspace** page.
2. Click the ellipsis to the right of the user or group that you want to edit access for (as shown in *figure 17.5*).
3. Click the workspace role that you want to assign or click **Remove**:

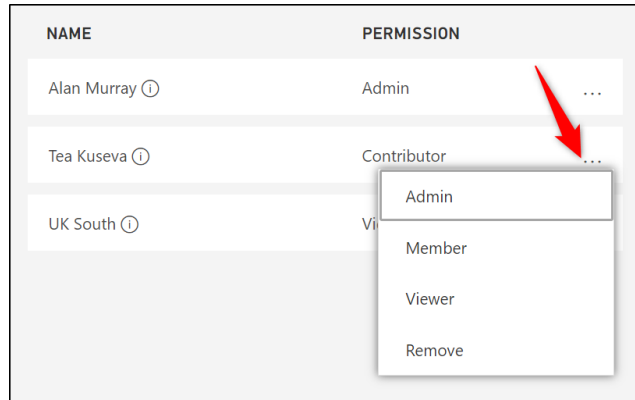


Figure 17.5: Changing the access for a user

Copying a report to another workspace

You can save a copy of existing reports to other workspaces that you are a member of. You need any workspace role above Viewer permission to be able to perform this action.

For this example, we will save a copy of the sales report that was published to the *Management Team* workspace into the newly created *UK Area Leads* workspace.

1. Navigate to the workspace where the report is stored (*Management Team* in my example).
2. Click the ellipsis to the right of the report name and click **Save a copy** (as shown in *figure 17.6*).

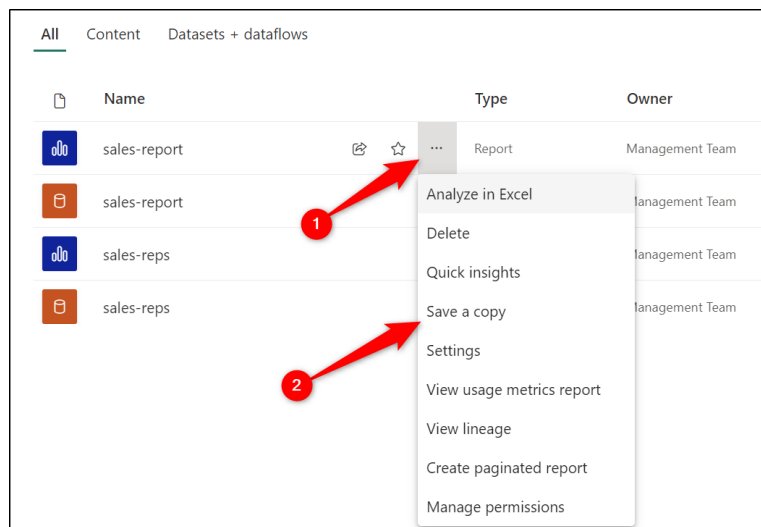


Figure 17.6: Saving a copy of an existing report

- The **Save a copy of this report** panel appears. Click on the workspace that you want to save the copy to and click **Save** (as shown in *figure 17.7*):

Figure 17.7: Saving a copy of a report to the UK Area Leads workspace

A message appears confirming that the save was successful (as shown in *figure 17.8*). Click the **Go to report** button to be taken to the report, or you can find it by navigating to the workspace.

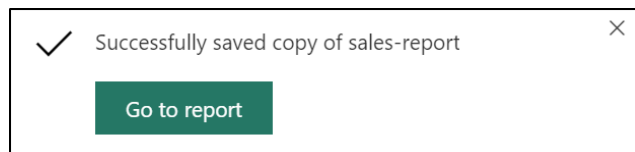


Figure 17.8: Successfully saved copy of a report

The existing report could be deleted from the other workspace (*Management Team* in my example) if required. Reports and datasets can be stored in separate workspaces with no problem. They will still be connected.

To delete a report, click the ellipsis to the right of the report name and click **Delete** (as shown in *figure 17.9*):

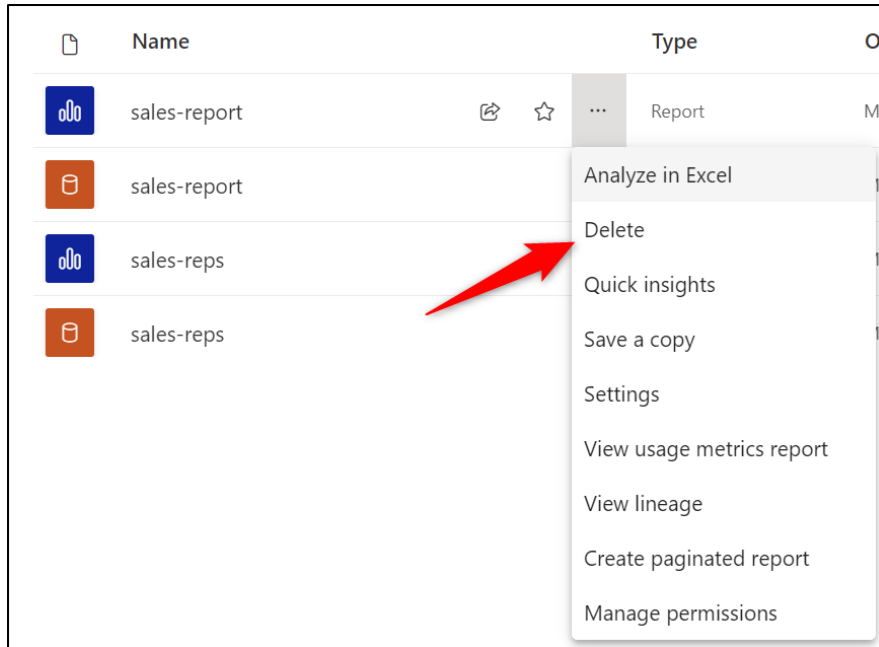


Figure 17.9: Deleting a report from a workspace

Refreshing datasets and reports

Once a report is published to the cloud (Power BI service), you will want to refresh it when the data changes. Data sources can be refreshed manually, or times can be specified for automatic refreshes.

The report itself will refresh with the browser or app, so it is not necessary to refresh visuals manually. However, there is a button on the toolbar of the report to refresh the visuals and sync it with the data if the data sources update while the report is open.

Power BI data gateways

A Power BI data gateway is required to refresh on-premises data sources (data that is not in the cloud) from the Power BI service. The gateway provides a fast and secure connection between on-premises data and Microsoft cloud services.

A gateway is not required to refresh data from cloud services, such as files on SharePoint or in an Azure database.

Note: A data gateway is also required for refreshing data that is connected using the Data | Web connector, as we used in Chapter 4, *Creating a Simple Power BI Report* to import the IMDB top 250 film data.

There are three types of Power BI gateway—standard, personal, and virtual network.

- **Standard (enterprise) mode gateway:** This gateway allows multiple users to connect on-premises data sources to many Microsoft cloud services, including Power BI, Power Automate, Power Apps, and Azure Analysis Services. It is normally installed on a server (especially in enterprise scenarios), and this is often a task that the IT department will control. This gateway can have multiple admins that manage and monitor the gateway.
- **Personal mode gateway:** This allows one user only to connect on-premises data sources to the Power BI service, and the gateway cannot be shared with others. It also only works for Power BI and not with other Microsoft cloud services such as Power Automate and Power Apps. Therefore, this option is great when you only work with Power BI and do not need to share data sources with others.
- **Virtual network gateway:** With this gateway, virtual networks are used to securely connect multiple users to data sources. No installation is required (unlike the other two) because it is a Microsoft-managed service.

In our example, we have used a few files that are stored on a local machine or network, so a gateway is required to refresh them from the Power BI service.

We will install the standard (enterprise) gateway on a local machine and then add the on-premises data sources that were used in the sales report example. Ensure that the machine in which you install the gateway has access to the data sources.

Note: You can install both the standard and personal mode gateways on the same machine.

Installing the standard mode gateway

Let us download the gateway from the Power BI service and then proceed through the steps to install and set up the gateway for use.

1. From the Power BI service, click the **Settings** button (ellipsis icon), point to the **Download** option, and click **Data Gateway** from the sub-menu (as shown in *figure 17.10*).

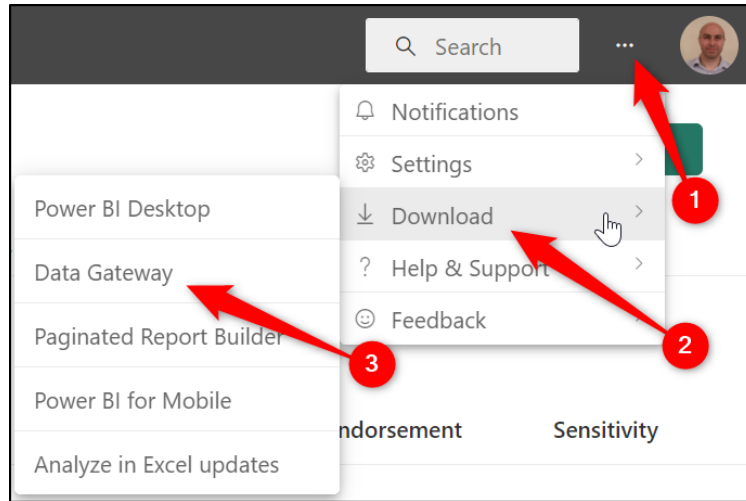


Figure 17.10: Downloading a data gateway from the Power BI service

A browser opens, and you are taken to the following URL—<https://powerbi.microsoft.com/en-us/gateway/>. Two buttons are shown at the top of the page providing the options to download the standard mode gateway or the personal mode gateway (as shown in figure 17.11).

2. Click the **Download standard mode** button to begin downloading the executable file for that gateway.

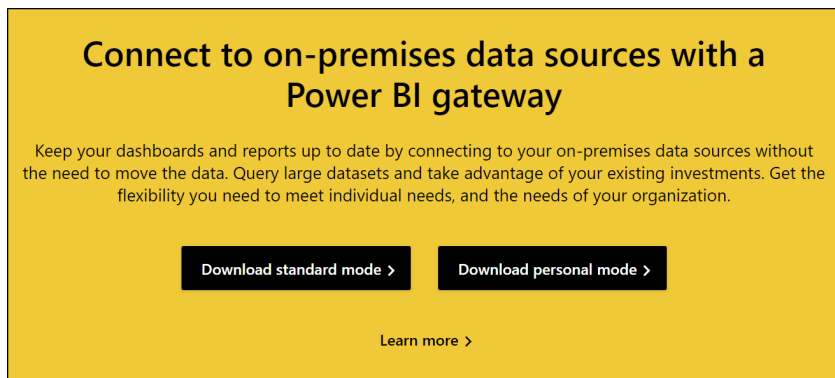


Figure 17.11: Buttons to download the different data gateways

3. Open the **GatewayInstall.exe** file that was downloaded to begin the installation.

- Review the minimum requirements for the installation and the path to install, and accept the terms of use and privacy statement. Click **Install** (figure 17.12):

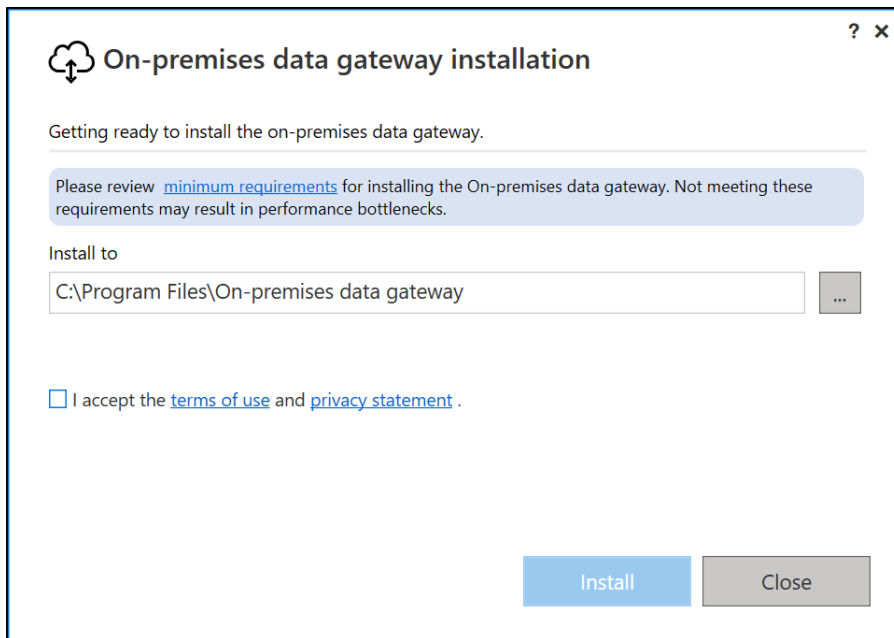


Figure 17.12: Begin the installation of the gateway

- After installation, you need to sign in to register the gateway (as shown in figure 17.13). Be sure to use an e-mail address associated with Power BI. Enter the e-mail address and click **Sign in**.

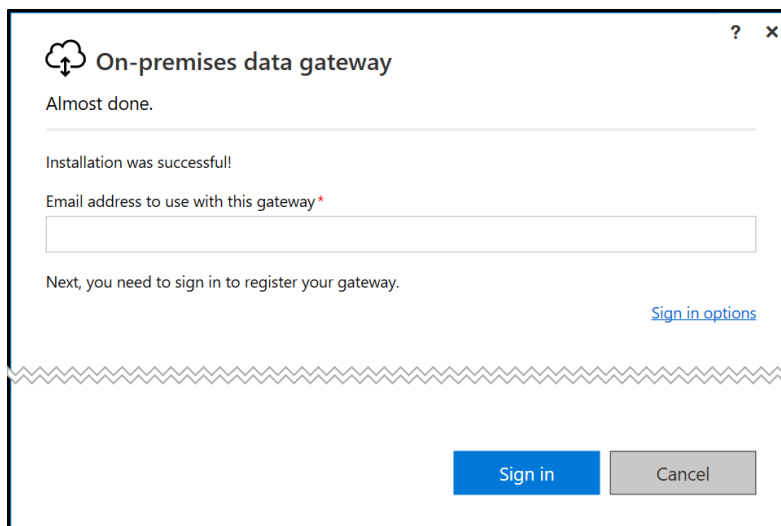


Figure 17.13: Sign in to register the gateway

6. Select **Register a new gateway on this computer** and click **Next** (as shown in *figure 17.14*).

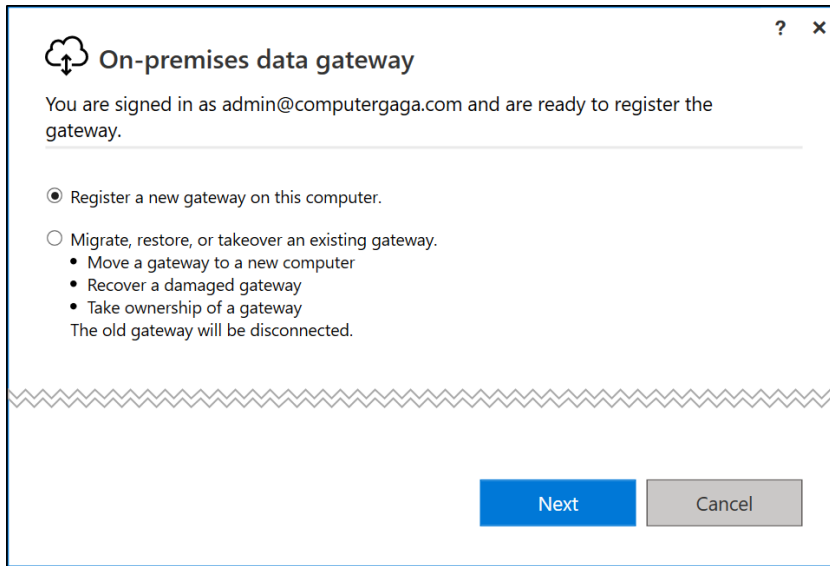


Figure 17.14: Register a new gateway on the computer

7. Once signed in, it is now time to enter a name for the new gateway and a recovery key (*figure 17.15*). The recovery key is needed to migrate, restore, or when creating a gateway cluster. Click **Configure**.

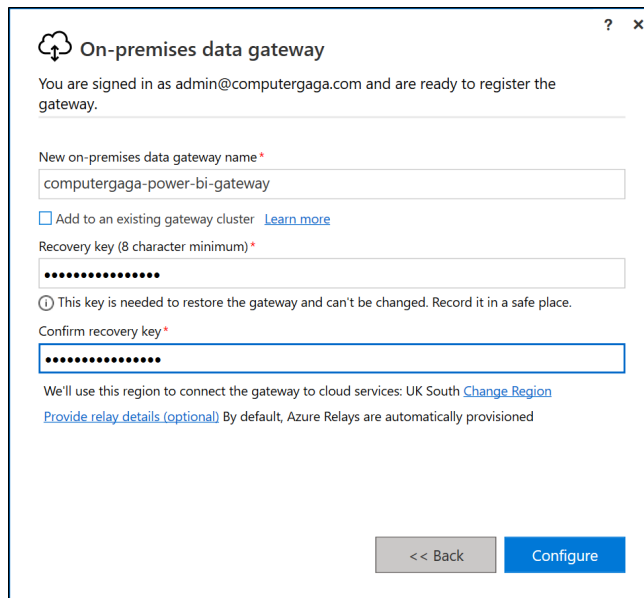


Figure 17.15: Entering a gateway name and recovery key

8. The gateway is now online and ready to be used (*figure 17.16*). The regions used for connections are shown. Power BI uses the default environment, the region where your tenant is located, but this can be changed if you want to specify an alternative local Azure region. Click **Close**:

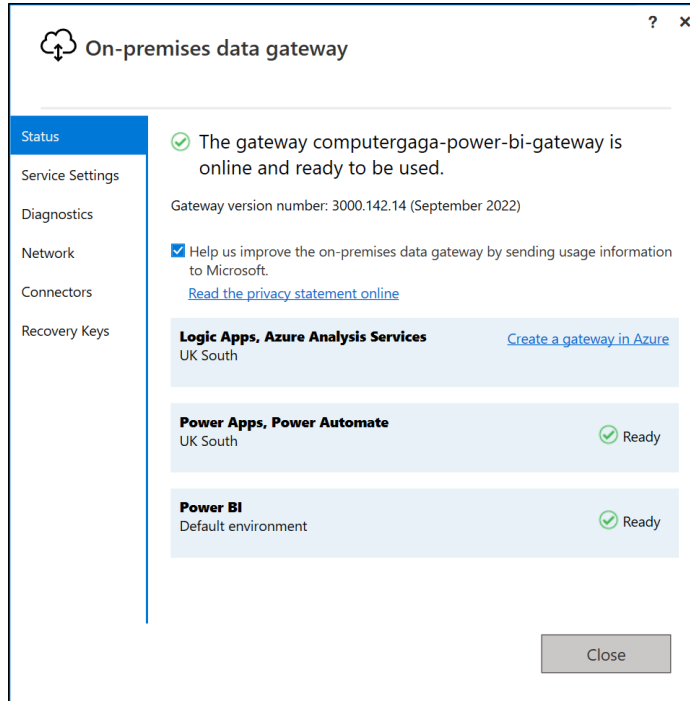


Figure 17.16: Gateway is online and ready

To check the status of the gateway or to access it at a later date to edit settings, click the **Settings** button (ellipsis icon), point to **Settings**, and click **Manage connections and gateways** (as shown in *figure 17.17*).

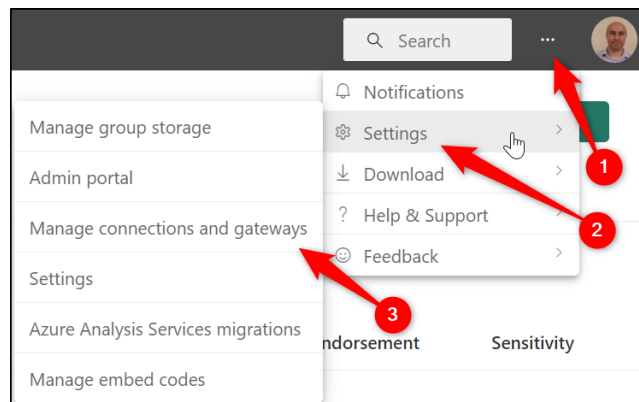


Figure 17.17: Manage gateways in the Power BI service

A list of all gateways is shown with information such as their users and status. By creating the gateway, you are immediately assigned admin rights. From this area, you can add data sources, schedule refreshes, and modify other aspects of the on-premises data gateways.

Data (preview)

Data sources **On-premises data gateways** Virtual network data gateways

The data gateway acts as a bridge, providing quick and secure data transfer between on-premises data and Power BI, Microsoft Flow, Logic Apps, and PowerApps. [Learn more in this overview.](#)

Name ↑	Contact info	Users	Status	Gateways
computergaga-office	admin@computergaga.com	Alan	☺	1
computergaga-power-bi-gateway	admin@computergaga.com	Alan	☺	1

Figure 17.18: List of on-premises data gateways

Adding data sources to a gateway

With the data gateway installed and online, the data sources that need refreshing will need to be added to the gateway.

This is a two-step process; first, the data source is added to the Power BI service, and credentials are added so that we have access to refresh it. Second, the data source used in a dataset is mapped to the data source that was added in the first step.

You can add data sources through the dataset settings or from the gateway management screen. For this example, we will go via the dataset.

1. Navigate to the workspace where the dataset is located (in my example, this is the *Management Team* workspace).

2. Click the ellipsis to the right of the dataset name and click **Settings**, as shown in figure 17.19:

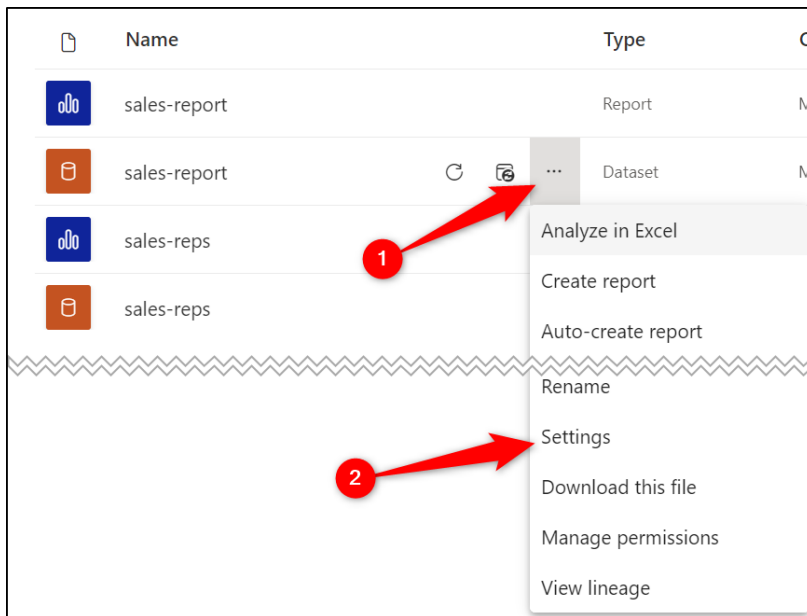


Figure 17.19: Accessing the dataset settings

3. Within the dataset settings, click the **Gateway connection** section to expand the options.
4. Click the **Actions** arrow (if necessary) for the data gateway that you want to configure (figure 17.20).

All data sources that are included in the dataset are listed. None of the data sources has been added yet, so an **Add to gateway** link is shown to the right of each source.

Each of the data sources needs to be added, except for the **product.xlsx** file stored on SharePoint. Adding to the gateway is optional for a cloud-based file.

- Click the **Add to gateway** link for the data source to be added, as shown in *figure 17.20*:

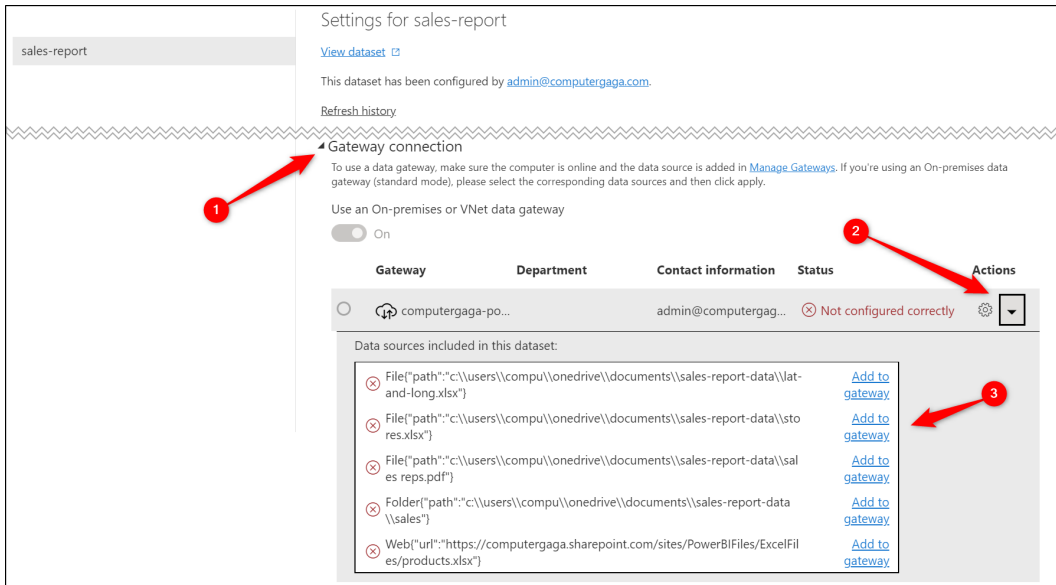


Figure 17.20: Dataset data sources that need adding to the gateway

- In the **New data source** window that appears (as shown in *figure 17.21*), the fields for the **Gateway cluster name**, **Data source type**, and **Full path** should be automatically completed. This is a benefit of adding the data source via the dataset instead of via the gateway (as shown in *figure 17.23*).
- Type a name for the data source in the **Data source name** field.
- Enter the **Authentication** details for the data source.
- Specify the **Privacy level**.

10. Click **Create**.

Figure 17.21: Adding a new data source to a gateway

11. Repeat Steps 5–10 for each data source to be added to the gateway.

You can view the data sources and their status in the gateway management screen (as shown in *figure 17.22*).

1. Click the **Settings** button (ellipsis icon) in the top right corner of the Power BI service.
2. Click **Settings | Manage connections and gateways**.

Figure 17.22 shows all data sources added to the **computergaga-power-bi-gateway** cluster. This includes the Web data source that we used in *Chapter 4, Creating a Simple*

Power BI Report, though you can see that the data source is added to a different gateway cluster:

Data (preview)

Data sources On-premises data gateways Virtual network data gateways

Power BI data sources for DirectQuery and Import datasets and dataflows, via cloud and the data gateway. [Learn more about supported data sources.](#)

Name ↑	Data source type	Users	Status	Gateway cluster name
Lat and Long	File	Alan	⊕	computergaga-power-bi-gateway
Reps	File	Alan	⊕	computergaga-power-bi-gateway
Sales	Folder	Alan	⊕	computergaga-power-bi-gateway
Stores	File	Alan	⊕	computergaga-power-bi-gateway
top-films	Web	Alan	⊕	computergaga-office
{"url":"https://computergaga.sharep...	Web	Alan	⊕	

Figure 17.22: All data sources have been added

Data sources can also be added through the gateway management screen (*figure 17.23*). However, you start with a blank *New data source* window, so going via the dataset is easier.

Click **New** at the top of the screen (as shown in *figure 17.23*). The *New data source* window appears for the data source details and credentials for authentication to be added.

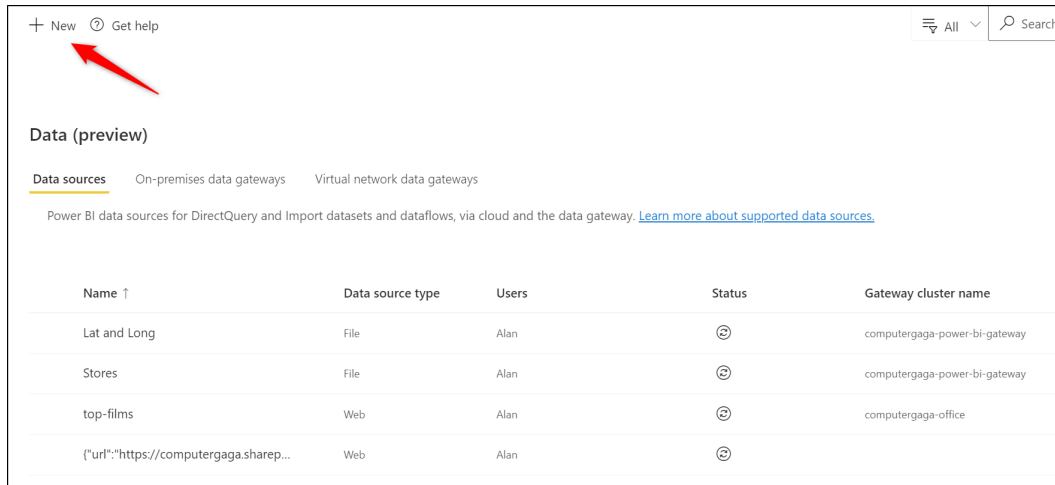


Figure 17.23: Adding a new data source from the gateway management screen

Now, we need to map each data source in the dataset to the data source that we added in the gateway. If further datasets are created that use an existing data source, there is no need to add it again; you simply map each instance of a data source to the one added to the gateway.

From the **Gateway connection** section of the dataset settings window, the **Add to gateway** link beside each data source has been replaced by a **Maps to** drop-down list (figure 17.24).

For each data source, click the **Maps to** drop-down list and select the corresponding gateway data source (as shown in figure 17.24).

The gateway status is shown as running with a green tick:

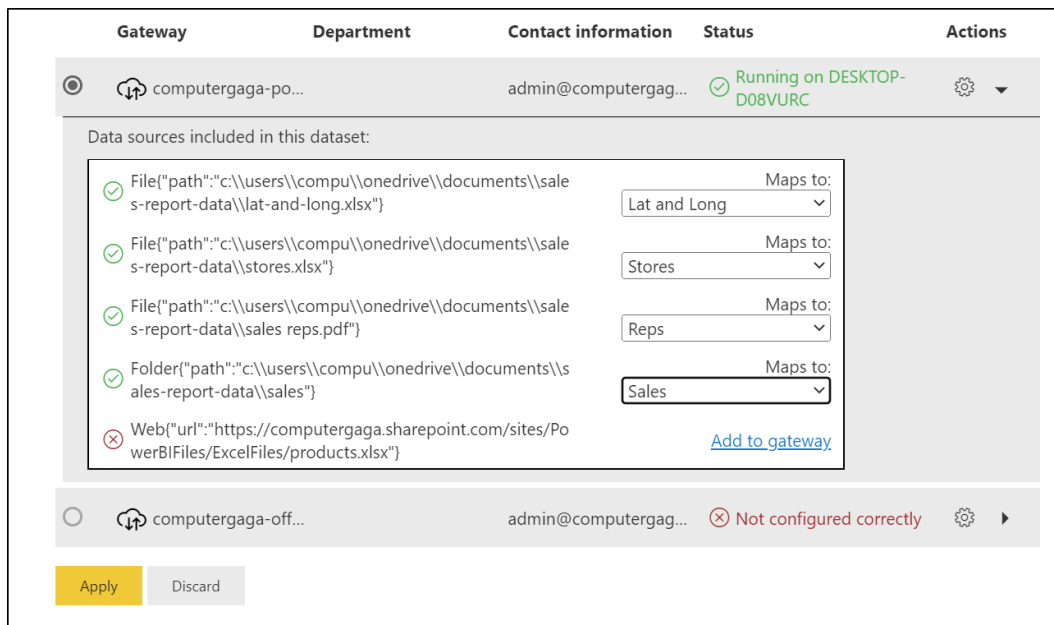


Figure 17.24: Mapping the data sources of a dataset

Although the **products.xlsx** file that is stored on SharePoint does not need to be added to the gateway, you could proceed with this step.

Otherwise, the only requirement is to enter credentials so that the file can be accessed and refreshed. Click the **Data source credentials** section with the dataset settings window to expand it.

An **Edit credentials** link is provided beside the **Web** source (as shown in figure 17.25). Click the **Edit credentials** link to configure the authentication details for this cloud-based file:

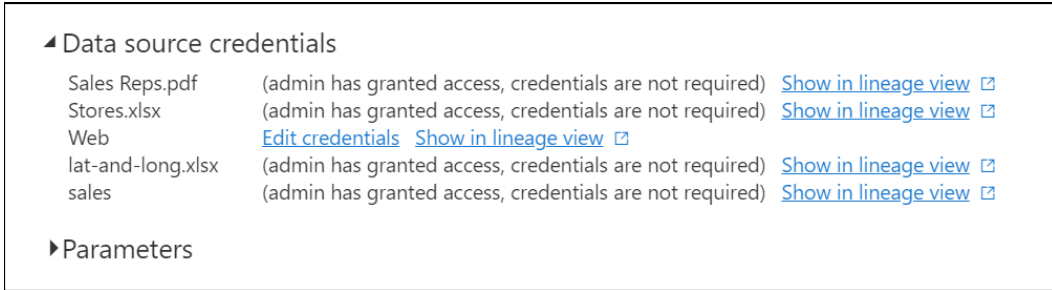


Figure 17.25: Checking the data source credentials

In the *Configure sales-report* (or whatever the dataset may be named) window (as shown in figure 17.26), specify the **Authentication method**, enter the **User name** and **Password** (if required), and specify the **Privacy level setting for this data source**. Click **Sign in**.

×
Configure sales-report
 url

 Authentication method

 User name

 Password

 Privacy level setting for this data source

 Skip test connection

Figure 17.26: Authenticating a cloud data source

Refreshing your data

With the data sources added, mapped, and authenticated, it is simple to refresh the dataset.

Navigate to the workspace where the dataset is stored and click the **Refresh now** button to the right of the dataset name (as shown in *figure 17.27*).

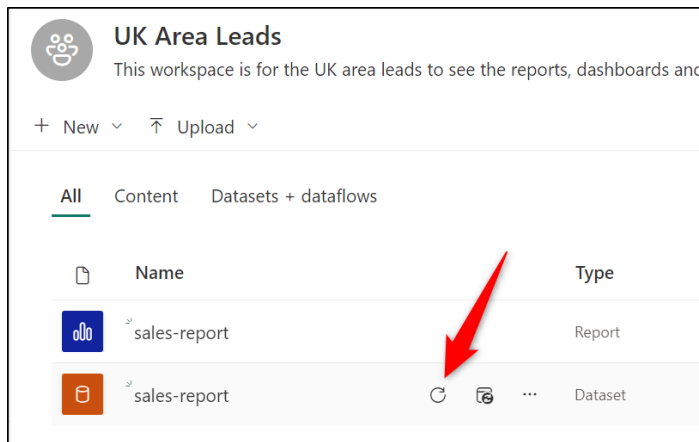


Figure 17.27: Refreshing a dataset

Scheduling data refreshes

You can schedule refreshes for a dataset so that the dataset refreshes on the times specified, and you do not need to be around to manually refresh the dataset.

For the dataset to refresh, the computer where the data gateway is installed and running will need to be switched on. Therefore, it is advised to install the data gateway on a server.

1. Navigate to the workspace where the dataset is located.
2. Click the **Schedule refresh** button to the right of the dataset name (as shown in *figure 17.28*):

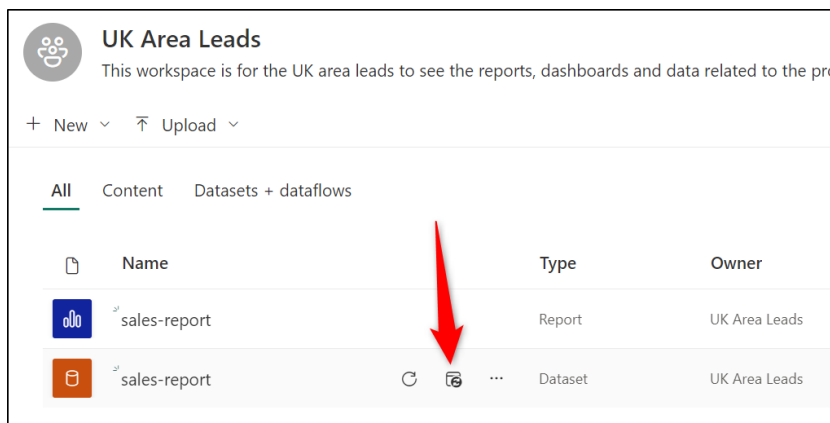


Figure 17.28: Accessing schedule refreshes for a dataset

3. You are taken to the dataset settings page. In the **Scheduled refresh** section, click the *On/Off* slider to switch scheduled refresh **On** (figure 17.29).
4. Set the **Refresh frequency** from the list provided. You can choose from **Daily** or **Weekly**.
5. Set the **Time zone** and click the **Add another time** link. Set the time for the automatic refresh and click the **Add another time** link again if you want to schedule further refreshes (as shown in figure 17.29)
6. By default, refresh failure notifications are sent to the dataset owner only. You can add additional or different contacts to receive these notifications if required.
7. Click **Apply**.

Note: If the dataset resides in a premium capacity or Premium Per User workspace, you can schedule up to 48 refreshes per 24 hours. If you have a Pro license, you are limited to eight scheduled refreshes per 24 hours. However, this is often enough.

⏏ Scheduled refresh

Keep your data up to date

Configure a data refresh schedule to import data from the data source into the dataset. [Learn more](#)

On

Refresh frequency

Daily ▾

Time zone

(UTC) Dublin, Edinburgh, Lisbon, Lon ▾

Time

5 ▾ 00 ▾ AM ▾ ×

7 ▾ 00 ▾ PM ▾ ×

[Add another time](#)

Send refresh failure notifications to

Dataset owner

These contacts:

Enter email addresses

Figure 17.29: Setting a daily refresh schedule for a dataset

Dashboards

There is often confusion initially with the terms report and dashboard. They are different, but you will encounter many people who inadvertently refer to a report as a dashboard.

A dashboard is a single page that contains visuals selected from one or more reports. The visuals on a dashboard are known as tiles. And the act of adding a visual from a report to a dashboard is known as pinning.

So, a dashboard can be thought of as the highlights from one or more reports. A report uses data from an underlying dataset, but a dashboard is strictly the visuals.

Dashboards are only available in the Power BI service. You cannot create or view dashboards within the Power BI Desktop.

Now that we know what a dashboard is and how it differs from a report let us create a dashboard and pin some visuals from an existing report.

Creating a dashboard

You can create a new dashboard directly from pinning a report visual or from a workspace. Let us take the workspace approach.

1. Navigate to the workspace where you want the dashboard to reside. For this example, I am using the UK Area Leads workspace that we created at the beginning of the chapter.
2. Click **New | Dashboard**, as shown in *figure 17.30*:

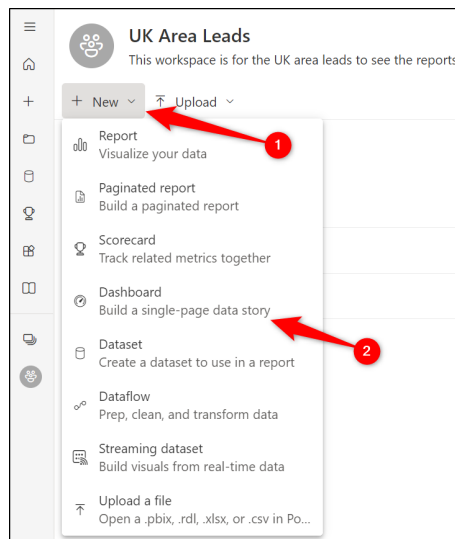


Figure 17.30: Creating a dashboard from a workspace

3. In the *Create dashboard* window, type the name for the new dashboard in the *Dashboard name* field and click **Create**, as shown in *figure 17.31*:

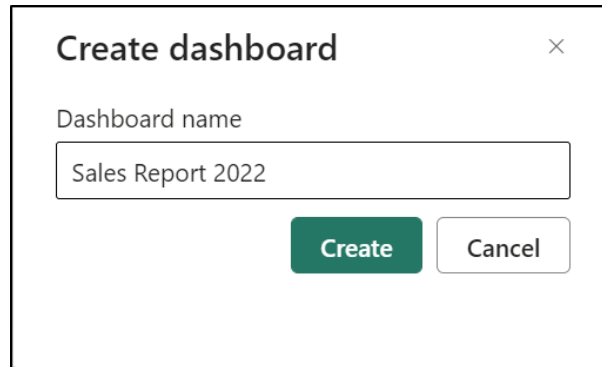


Figure 17.31: Naming the new dashboard

You are taken directly to the newly created dashboard. The dashboard is currently empty, so it does not look particularly interesting. Let us pin a visual.

Pinning visuals to a dashboard

Adding visuals to a dashboard is known as pinning, and it is simple to do. You can pin any visual from a report you have access to.

1. Open the report that contains the visual that you want to pin.
2. Position your cursor over the top right of the visual until the visual header appears, and click the **Pin visual** button, as shown in *figure 17.32*:



Figure 17.32: Pin a visual from a report

3. In the *Pin to dashboard* window, select **Existing dashboard**, and then from the *Select existing dashboard* list, choose the *Sales Report 2022* dashboard that was just created, as shown in *figure 17.33*.

Note: By choosing **New dashboard** in the **Pin to dashboard** window, we could have created the dashboard and pinned the visual in one step. However, we wanted to demonstrate creating new dashboards from a workspace as that approach offers other new items, such as dataflows, paginated reports, and scorecards that you can explore.

4. Click **Pin**.

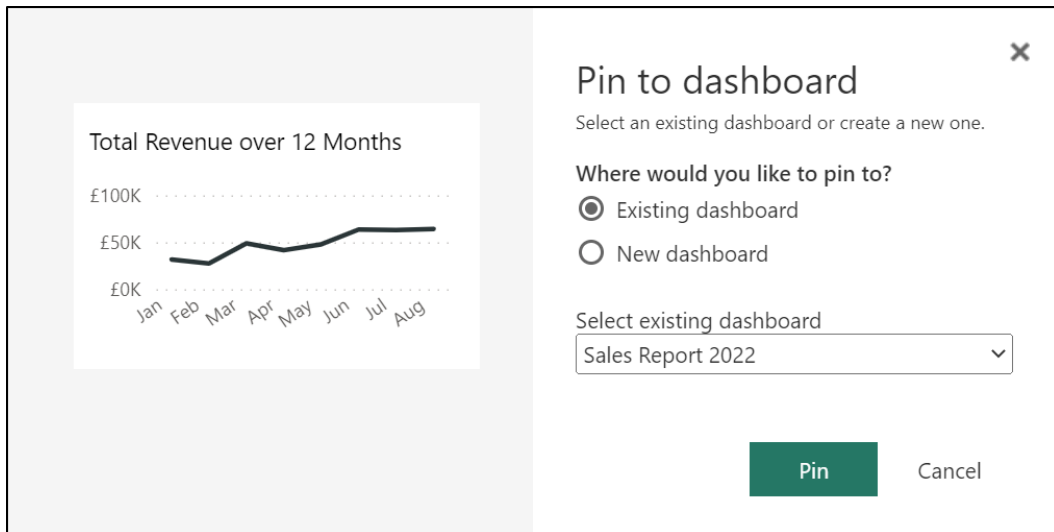


Figure 17.33: Pinning a visual to an existing dashboard

The visual is pinned as a tile to the dashboard, and the *Pinned to dashboard* message appears temporarily in the top right corner of the window.

Click **Go to dashboard** to jump straight to the dashboard or stay on the report to pin further visuals.

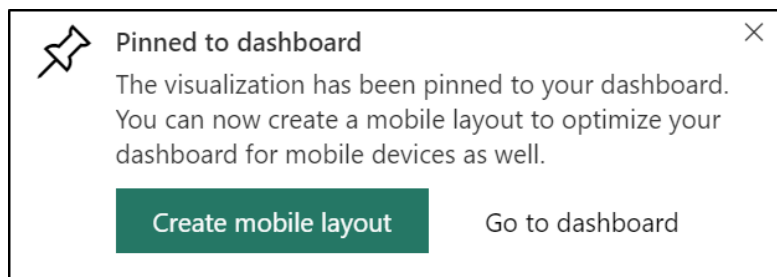


Figure 17.34: Pinned to dashboard message

If you missed the **Pinned to dashboard** message, you could open the dashboard by navigating to the workspace where it resides and clicking on it there.

Figure 17.35 shows the dashboard with a single pinned tile. Note the options along the toolbar to share, comment, edit, and more. There is also an **Ask a question about your data** prompt. Please refer to the following figure:



Figure 17.35: The dashboard with a single pinned tile

Note: In addition to pinning visuals from reports to a dashboard, you can also pin a tile from another dashboard, pin a tile from Excel, or pin an entire report page. You can also add images, text boxes, and videos.

Sharing a dashboard

You can share a dashboard with others offering you the chance to share a single page of selected visuals for a specific audience.

1. Navigate to the workspace where the dashboard resides.

- Click the ellipsis dots to the right of the dashboard name and click **Manage permissions**, as shown in *figure 17.36*:

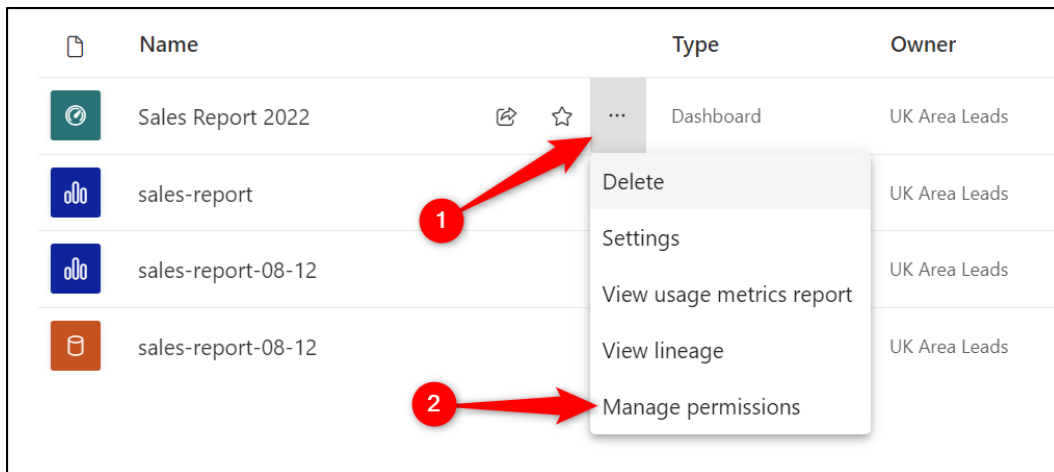


Figure 17.36: Managing the permissions for a dashboard

You are taken to a window showing the people and groups who have access to the dashboard. Those shown in *figure 17.37* have workspace roles automatically, giving them access to the dashboard within it.

- Click the **Add user** button above the list to grant access to another person or group:

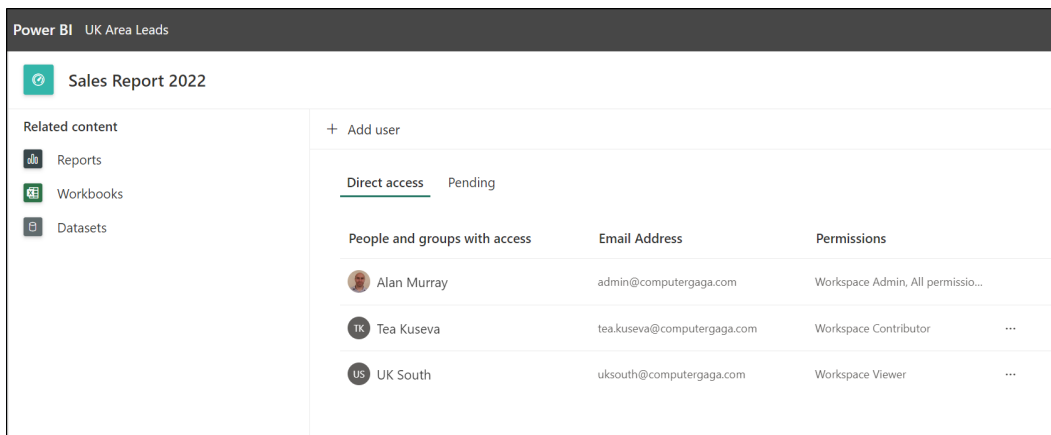
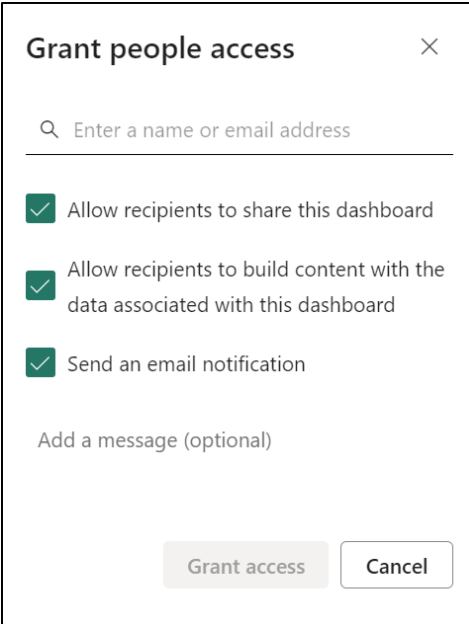


Figure 17.37: List of people and groups with access to the dashboard

- In the **Grant people access** window (*figure 17.38*), enter the name or e-mail address of the person or group you want to grant access.

5. Specify whether you want recipients to be able to share the dashboard with others or to build content with the associated data.
6. Enter a message with any welcome information you would like to share with the recipients.
7. Click **Grant access**, as shown in *figure 17.38*:



Grant people access ×

🔍 Enter a name or email address

Allow recipients to share this dashboard

Allow recipients to build content with the data associated with this dashboard

Send an email notification

Add a message (optional)

Grant access Cancel

Figure 17.38: Sharing the dashboard with others

The dashboard is shared with the people or groups, and they will receive an e-mail notification informing them that you have granted access (unless the box in *figure 17.38* is unchecked).

Conclusion

In this chapter, we learnt how to use workspaces and dashboards in the Power BI service. We saw how to create them, share them with others in the organization, and control their access.

We also learned how to refresh the data from the service, install and set up a gateway, and how set up scheduled refreshes.

In the upcoming chapter, we will see how Power BI works with other associated tools such as PowerPoint, Excel, and Teams. It is very likely that a Power BI user has some involvement in some or all three of these associated tools.

Questions

Here are some questions to test what you have learned in this chapter.

1. **Which of the following is not a workspace role?**
 - a. Contributor.
 - b. Master.
 - c. Viewer.
 - d. Admin.

2. **You can schedule up to 48 refreshes per 24 hours with a Premium Per User account.**
 - a. True.
 - b. False.

3. **Which of the following statements are true of dashboards?**
 - a. They are not available in Power BI Desktop.
 - b. They contain only one page.
 - c. You can pin visuals from one or more reports.
 - d. All of the above.

4. **You can copy a report to other workspaces.**
 - a. True
 - b. False

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



CHAPTER 18

Power BI and Other Apps

Introduction

This chapter looks at using Power BI with other Microsoft apps. For this chapter, we will focus on using Power BI with Excel, PowerPoint, and Microsoft Teams.

It is very commonplace for users of Power BI to work with Excel, also. Thankfully, it is easy to transfer your data models between Excel and Power BI and vice versa. We will cover a few methods to create reports on both platforms from existing datasets.

We will also look at how Power BI can interact with PowerPoint and with Microsoft Teams. These tools are popular methods for presenting and collaborating on reports.

Structure

In this chapter, we will cover the following topics:

- How to analyze a Power BI dataset in Excel.
- How to connect to live datasets from Excel.
- How to create Power BI reports from Excel data models.
- How to export Power BI reports into PowerPoint.
- How to share and collaborate on Power BI reports in Microsoft Teams.

Objectives

After reading this chapter, you will know how Power BI can be used with Microsoft Excel, PowerPoint, and Teams. In this connected world, Power BI does not work alone but within an ecosystem of applications, and it is important that you can use them together.

You will learn how to create reports in Microsoft Excel from existing Power BI datasets and how to create Power BI reports from existing Excel data models. You will know how to export Power BI reports into PowerPoint to be shared or presented to others live and understand how to distribute and collaborate on Power BI reports in Microsoft Teams.

Power BI and Excel

Power BI and Excel are closely integrated. There are many options for exchanging data between the two products. These can include exporting data from visuals or data model tables in the Power BI service to Excel and using featured tables in Power BI (known as data types in Excel).

We will not cover all the possible options in this chapter but will focus instead on how to create reports in Excel from Power BI datasets and create Power BI reports from existing Excel data models (known as Power Pivot in Excel).

Creating Excel Reports from Power BI datasets

To create reports in Excel from Power BI datasets, you can use the Analyze in Excel feature in the Power BI service or connect to the Power BI dataset from Excel. Let us explore both options.

Using Analyze in Excel

With Analyze in Excel, the Power BI dataset is sent to Excel so that it can be analyzed using PivotTables, Slicers, and other Excel features.

1. Open the report in the Power BI service that you want to analyze in Excel.
2. Click **Export | Analyze in Excel** (as shown in *figure 18.1*).

Although this action is being performed from the report, it is the underlying dataset only that is sent to Excel.

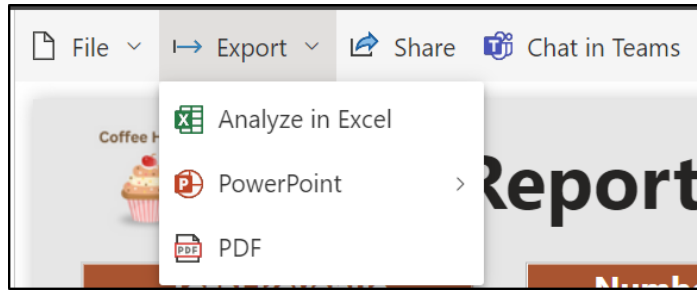


Figure 18.1: Analyze a Power BI data model in Excel

You can also access **Analyze in Excel** from the dataset or from the **More options** ellipsis dots beside the report or dataset from a workspace (as shown in figure 18.2).

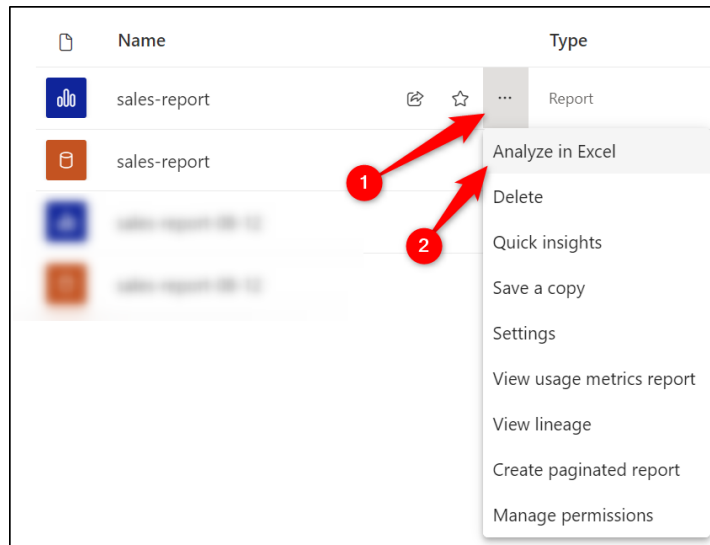


Figure 18.2: Analyze in Excel from a workspace

An Excel file is created and saved to your OneDrive for Business. If you do not have OneDrive for Business, *Analyze in Excel* will download the Excel file to your local *Downloads* folder.

A message appears in the top right of the window to inform you when the Excel file is ready (as shown in figure 18.3). Click the **Open in Excel for the web** button to

immediately open it in a Web browser and prevent having to navigate through your OneDrive for Business to open it.

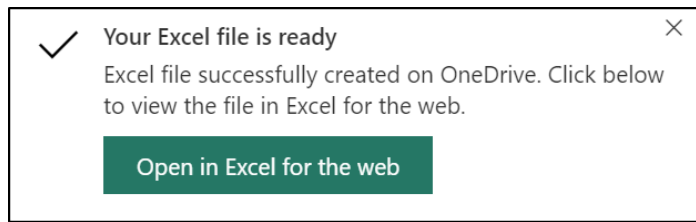


Figure 18.3: Your Excel file is ready to prompt

A message may appear warning you of queries that have been created by the *Analyze in Excel* feature (as shown in figure 18.4). Click on **Yes** to enable the queries and allow you to start building your Excel report.

An OLAP connection has been created to the Power BI dataset. This is great as this connection can be refreshed, ensuring that our Excel report uses the latest version of the data.

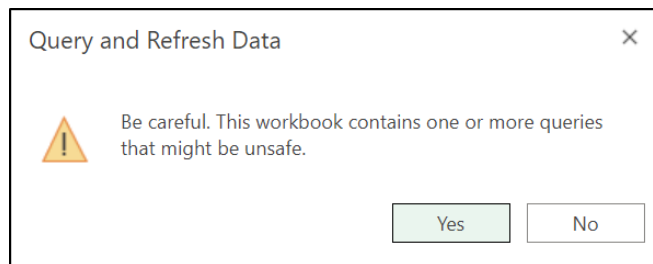


Figure 18.4: Warning about queries that have been created

Figure 18.5 shows a blank PivotTable and the **PivotTable Fields** pane listing all the tables and fields from the Power BI dataset. Notice that the measures have been moved to their own folder above the tables:

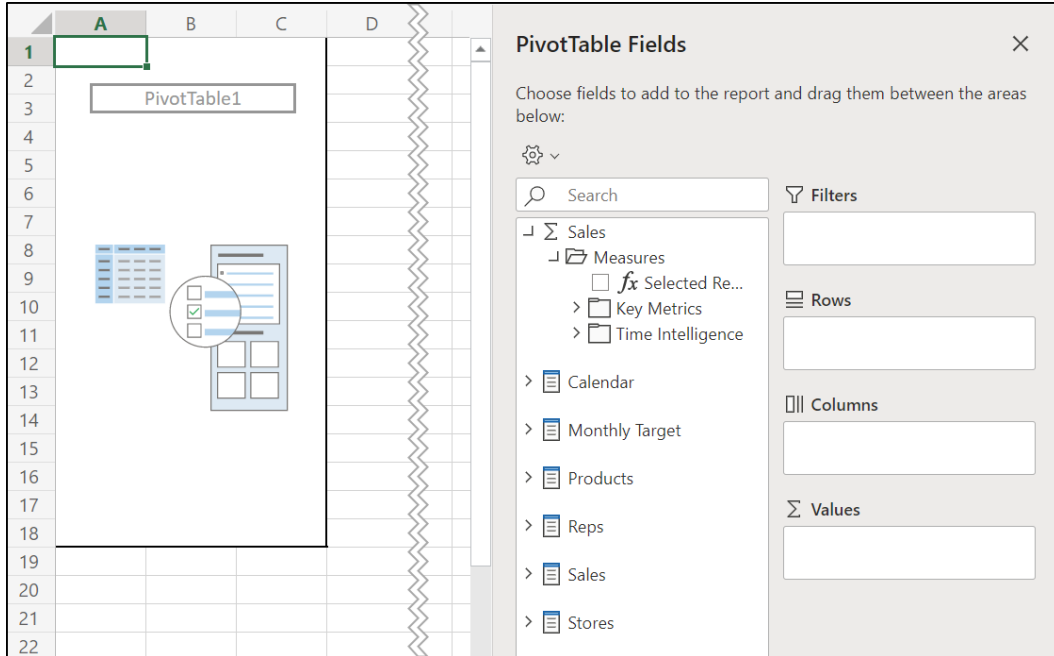


Figure 18.5: PivotTable using the Power BI dataset

From here, you can build your PivotTables, Slicers, PivotCharts, and more from the dataset just like you would with data from a worksheet.

To refresh the connection to the Power BI dataset, click on **Data | Refresh All** (figure 18.6).

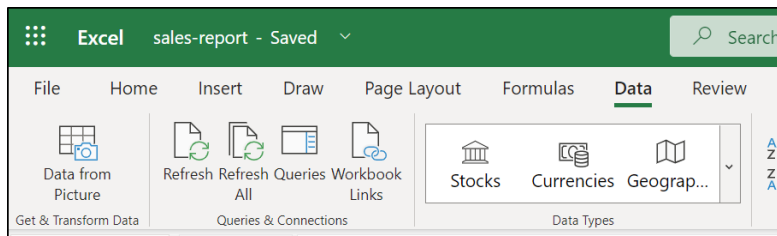


Figure 18.6: Refreshing queries in Excel for the Web

Excel for the Web is experiencing tremendous growth and is almost on a par with the Excel Desktop version in its richness of features. However, there are still commands in Excel Desktop that have not made it to the Web version yet.

Click on **Editing | Open in Desktop App** to switch to the desktop version (as shown in figure 18.7).

You may be prompted to enable content and confirm that you trust the document as it is stored in the cloud.

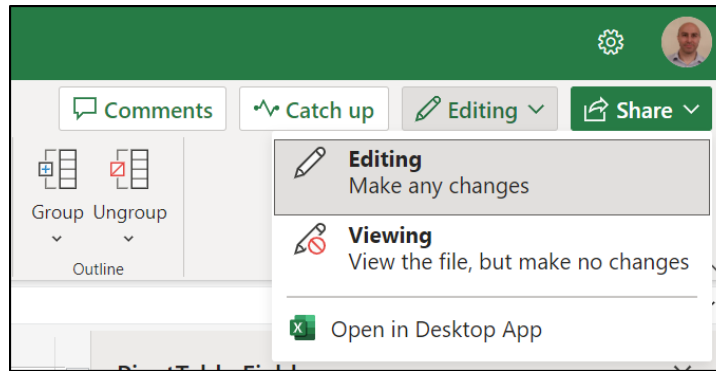


Figure 18.7: Opening the Excel file in the Excel desktop app

Connecting to Power BI datasets from Excel

You can connect to live Power BI datasets from Excel without needing to navigate to the report or dataset in the Power BI service first.

1. Click on **Data** | **Get Data** | **From Power Platform** | **From Power BI** (as shown in figure 18.8).

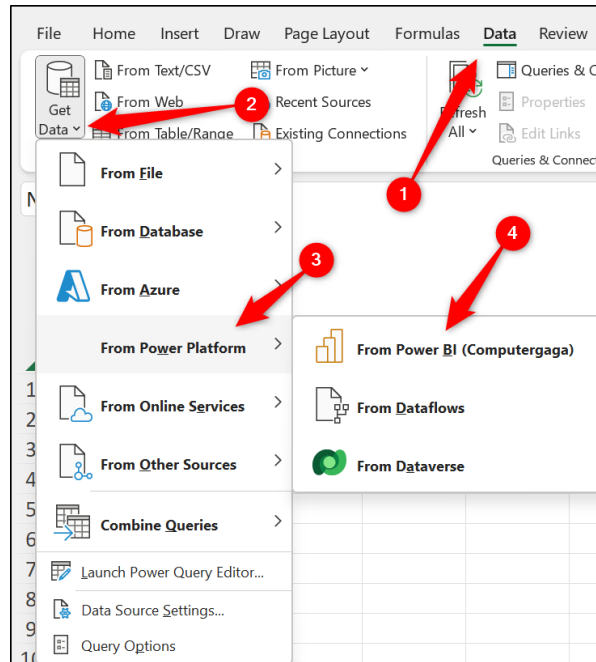


Figure 18.8: Get data from Power BI in Excel

The **Power BI Datasets** pane appears on the right of the window, as shown in *figure 18.9*.

Details such as the workspace where the dataset resides, when it was last refreshed, what tables are included, and what reports are using the dataset are stated for each Power BI dataset.

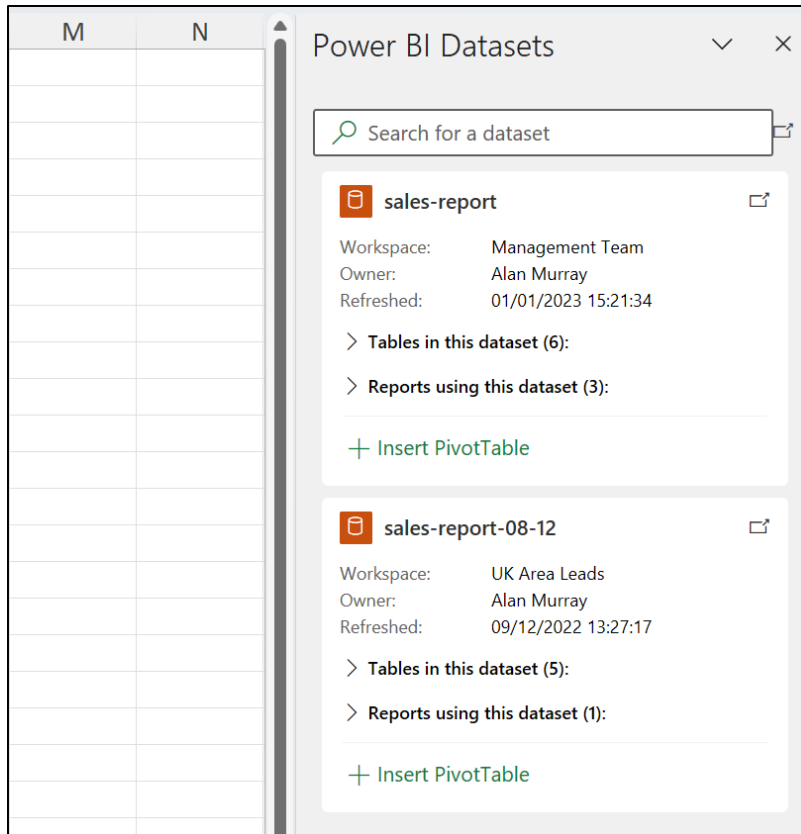


Figure 18.9: Listing of Power BI datasets for use in Excel

2. Click on the **Insert PivotTable** button for the dataset that you want to use in Excel.

The PivotTable is inserted into the worksheet, and the *PivotTable Fields* pane lists all tables and measures from the Power BI dataset (as shown in *figure 18.10*).

- Click and drag the fields and measures into the PivotTable areas to create your PivotTable as you usually would.

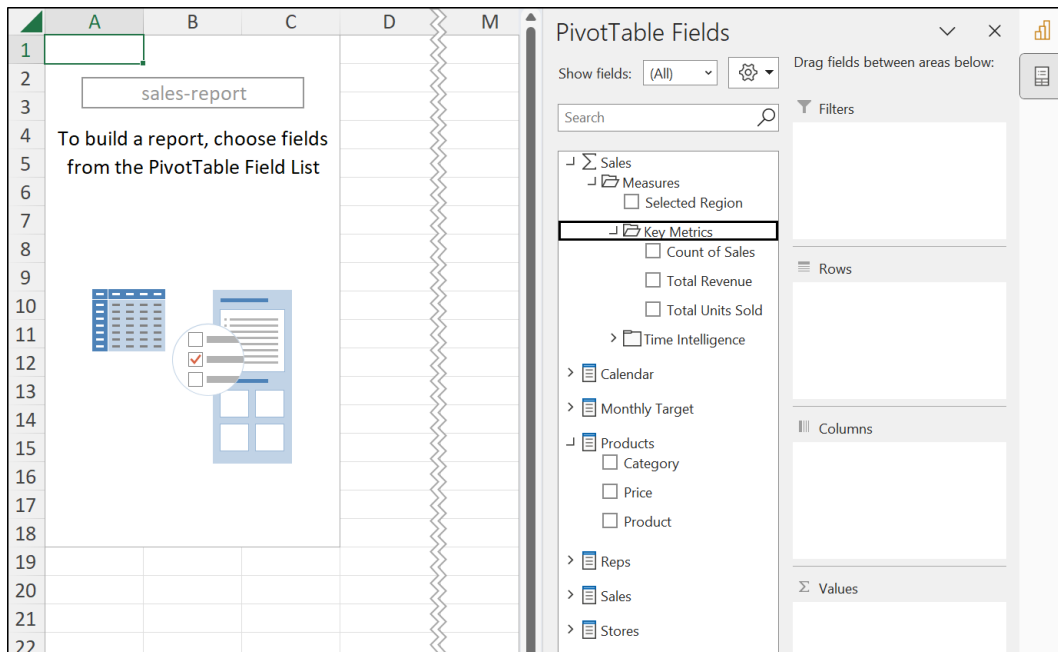


Figure 18.10: PivotTable with all tables and measures from the Power BI dataset

This is a live connection to the dataset in the Power BI service and can be refreshed to include any changes to the live dataset.

This connection can be found in the **Queries & Connections** pane in Excel, as shown in *figure 18.11*, by clicking on **Data | Queries & Connections**.

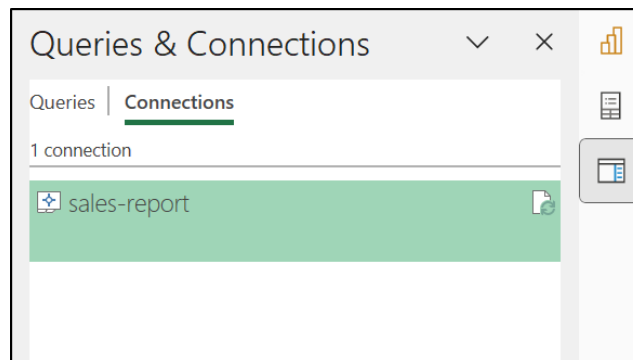


Figure 18.11: Live connection to the sales-report dataset

Right-click on the connection to refresh or delete it.

You can also modify the connection properties to set the connection to automatically refresh every x number of minutes or whenever the Excel workbook is opened.

Access the properties by selecting the connection and clicking on **Data | Properties** or by right-clicking the connection and clicking on **Properties**.

In the **Connection Properties** window (*figure 18.12*), the connection can be modified to **Refresh every x minutes** or to **Refresh data when opening the file**. You can also see the date and time of when it was last refreshed:

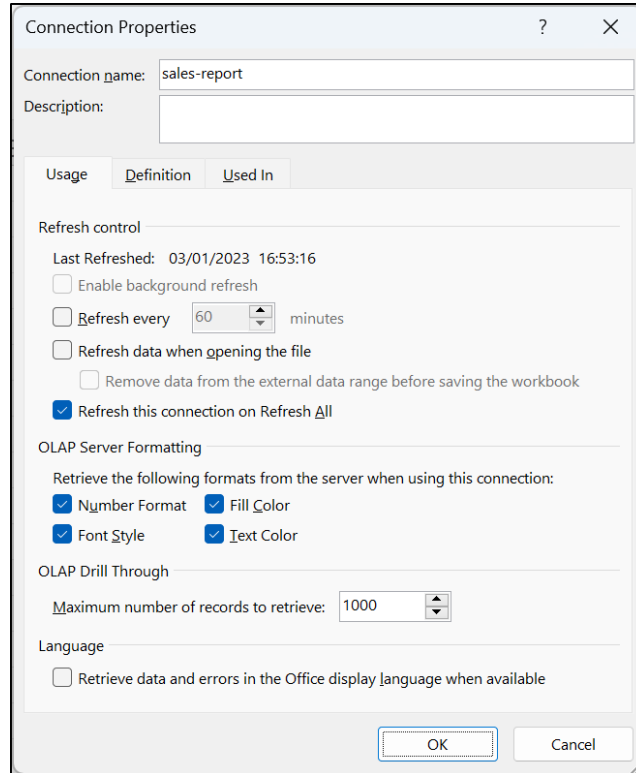


Figure 18.12: Connection properties window for the Power BI dataset

Import Power Pivot models into Power BI

You can import existing Excel data models, known as Power Pivot models, into Power BI to harness the greater visualizations and interactivity that Power BI offers over Excel.

You can import from Excel using Power BI Desktop or through the Power BI service. Let us look at both approaches.

From Power BI Desktop

Files: **excel-data-model.xlsx**

Importing data models from Excel using Power BI Desktop is quick and simple and offers the opportunity to add DAX measures and queries and make other refinements before it is published to the Power BI service.

Data is always imported into a new Power BI file (**.pbix** file).

1. Open Power BI Desktop.
2. Click on **File | Import | Power Query, Power Pivot, Power View** (as shown in *figure 18.13*).

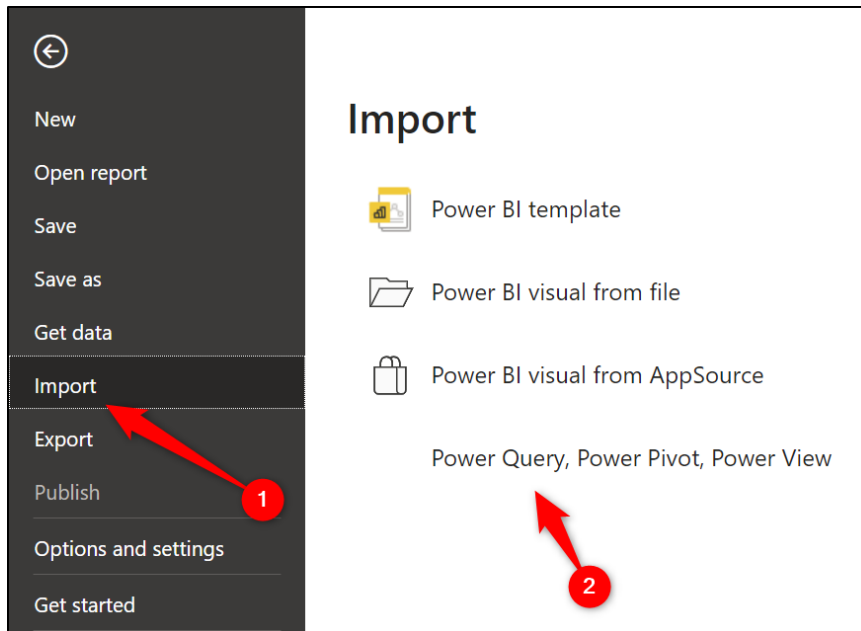


Figure 18.13: Importing Power Pivot data from Excel

3. You will be prompted to save the current file as a new one is being created. Click on **Don't save** (unless you have a file open that requires saving).
4. Navigate to and open the **excel-data-model.xlsx** file available in *Chapter 18, Files* folder.
5. The **Import Excel workbook contents** window opens and informs you that Power BI does not work directly with Excel but can import the data model and that a new Power BI file will be created (as shown in *figure 18.14*). Click on **Start**:

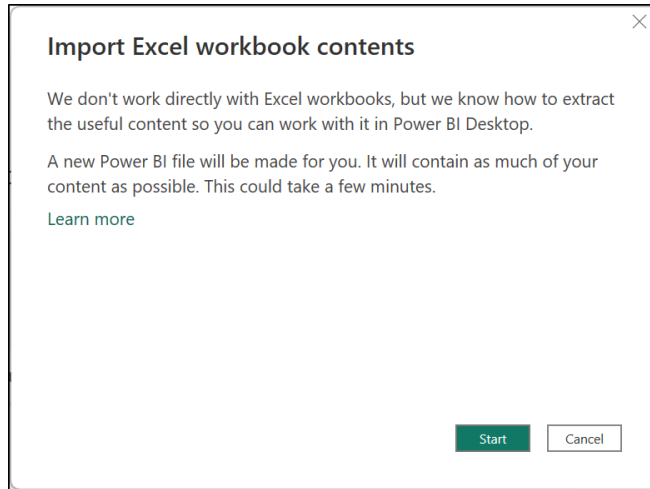


Figure 18.14: Import Excel workbook contents window

- The import process begins, and when complete, the **Import Excel workbook contents** window displays **Migration completed** and lists the queries that were successfully imported (as shown in *figure 18.15*). Click **Close**.

Note: You may see a message asking if you want to link to the files or import them into the report. If you receive this, choose the appropriate option.

The `excel-data-model.xlsx` workbook used in this example uses some of the same connections that our `sales-report.pbix` file does. It includes the `products.xlsx`, `stores.xlsx`, `sales reps.pdf`, and `sales` folder connections. You can direct the links to the same files in your directory for this process to work more efficiently.

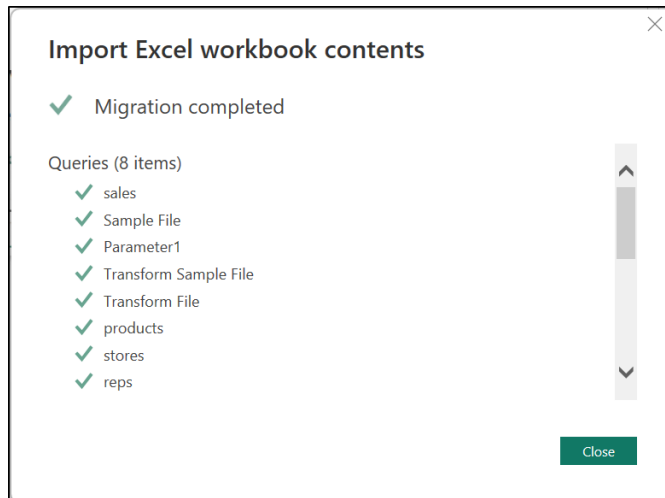


Figure 18.15: Migration of Excel contents completed

The tables, fields, and measures are shown in the *Data* pane on the right. You can work further on the data model or begin creating a report, just as we have throughout this book.

When ready, click on **Home** | **Publish** to publish the dataset (and report) to the Power BI service.

From the Power BI service

You can also import Excel data directly from the Power BI service. This approach assumes that the data model is ready for use and requires nothing to be added or changed.

Once live in the service, Power BI and Excel can connect to the dataset to create reports and perform quick analysis.

1. Navigate to the workspace where you want to import the dataset from Excel (**Management Team** has been used for this example).
2. From the workspace, click on **New** | **Dataset** (as shown in *figure 18.16*).

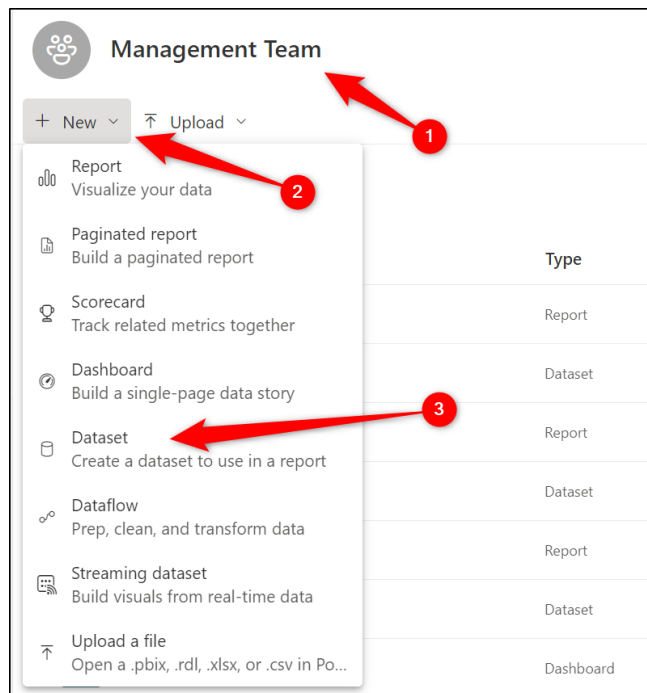


Figure 18.16: Creating a new dataset in the Management Team workspace

3. Click on the **Excel** button in the **Add Data to get started** screen, as shown in *figure 18.17*:

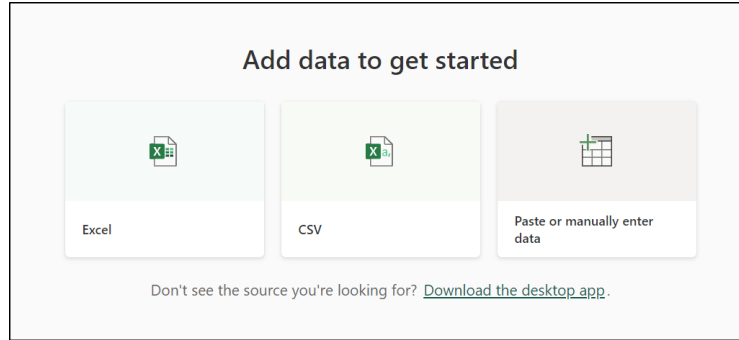


Figure 18.17: Get data from Excel in the Power BI service

- In the **Select a file** window (figure 18.18), navigate to the **excel-data-model.xlsx** file. You are initially taken to your OneDrive files and folders. Click the **Browse this device** button to access locally stored files.

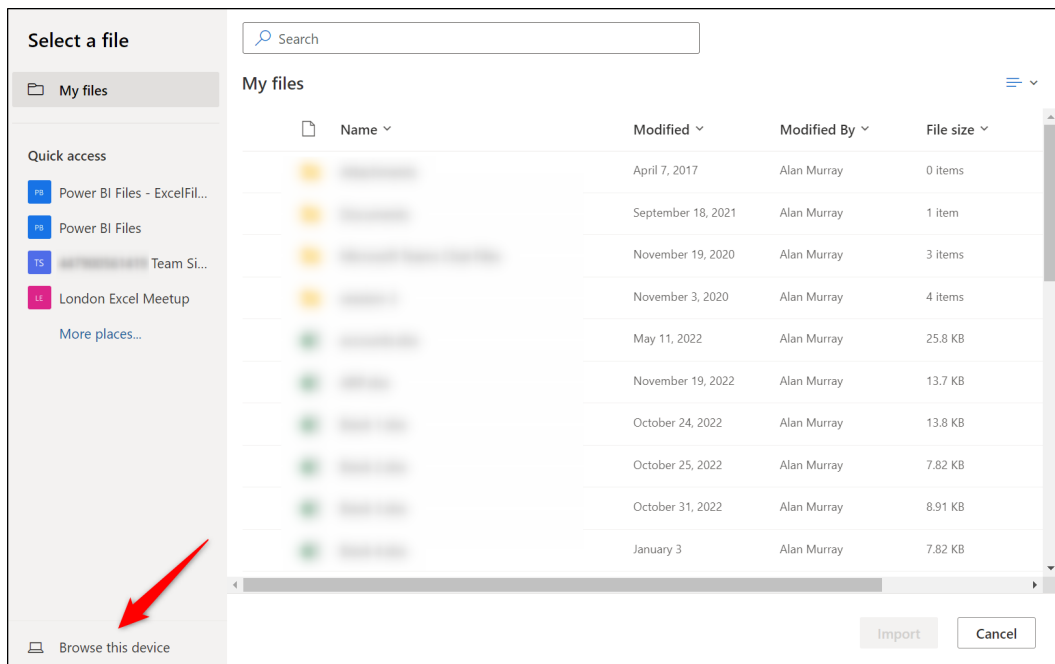


Figure 18.18: Select the Excel workbook

Note: The upload of an Excel workbook to the Power BI service approach is not covered in this chapter. It is yet another way that Excel and Power BI work together. You can work directly with the Excel file in Excel for the Web and pin parts of the worksheets to dashboards.

The Excel data model is published and is now available like any other dataset to be used in future Power BI reports (and Excel too).

Figure 18.19 shows the **excel-data-model** dataset in the **Management Team** workspace. The Excel file name has been used for the dataset name, but it can be renamed if required. Click on the **More options** ellipsis beside the dataset name and click **Rename** to do so:

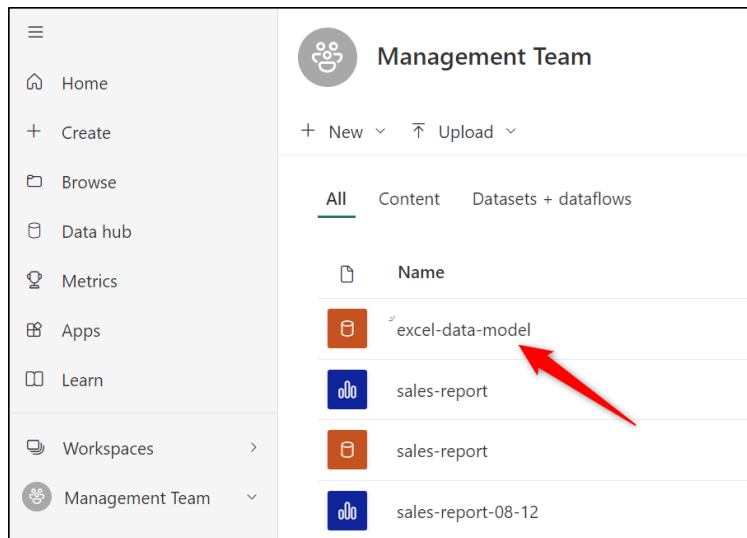


Figure 18.19: Excel data model successfully published to the Management Team workspace

Export to PowerPoint

You can export your Power BI reports to PowerPoint in just a few clicks. This offers an alternative method to share your Power BI reports with advantages including the following:

- Users do not need access to the Power BI service to view the report.
- Users do not need knowledge of how to use Power BI.
- Users do not have access to the data.

PowerPoint is a program that is familiar to many people and provides a simple and secure way of sharing with others.

Using the export to PowerPoint functionality, Power BI reports can also be seamlessly integrated into a PowerPoint slide deck for a presentation or to be published on the Web.

A Power BI report can be exported as a static image or as a live report allowing users to interact with the filters and the visuals.

Figure 18.20 shows both methods of exporting to PowerPoint available in the **Export | PowerPoint** menu from a Power BI report.

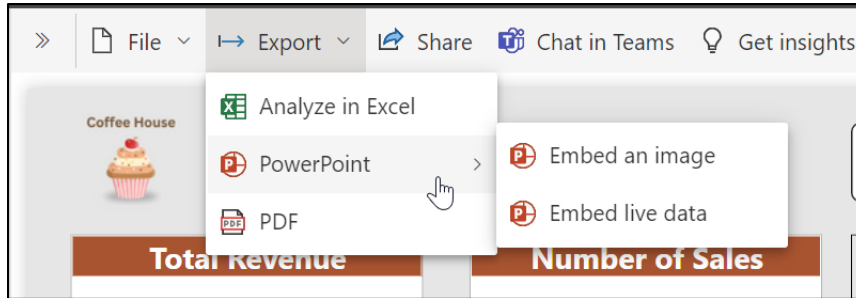


Figure 18.20: Exporting a Power BI report to PowerPoint

Embed an image

Let us look at exporting a Power BI report to PowerPoint as a static image first.

This option is great for keeping control over what a user sees. You can specify which filters are applied and whether to export the current page only or all report pages. Users will have no functionality making this a simple but reliable and secure method of sharing reports.

1. Open the report that you want to share.
2. Click on **Export | PowerPoint | Embed an image**.

In the *Export* window that appears (as shown in figure 18.21), you can specify whether to use the current values (different filters and slicers may be applied) or the default values without any filters applied.

By default, all report pages except the hidden pages are exported, but you can check or uncheck the boxes as per your requirements.

3. In this example, let us check the **Only export current page** box and leave the other settings to export **Current Values** and **Exclude hidden report tabs**. Click on **Export**.

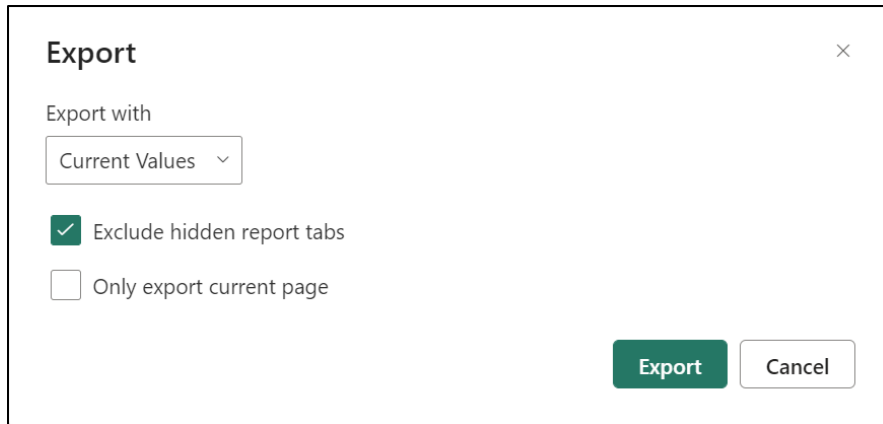


Figure 18.21: Export as an image to PowerPoint window

4. The image(s) are exported into a new PowerPoint presentation, and it is downloaded to your default downloads folder.

Figure 18.22 shows the **Front Page** page of our sales report exported into PowerPoint. A cover slide is generated automatically, providing a link to the report in the Power BI service and a date-time stamp for when the data was last refreshed and when it was downloaded.

If the **Only export current page** box was left unchecked, all report pages, except the hidden pages, would be exported as images onto separate slides.

Individual slides can be copied into existing PowerPoint presentations using a simple copy and paste or the *Reuse slides* feature of PowerPoint.

Each image is also a link back to the report on the Power BI service. In the Slide Show view of PowerPoint, you can click an image to be taken directly to the report if you have access to the report on that account.

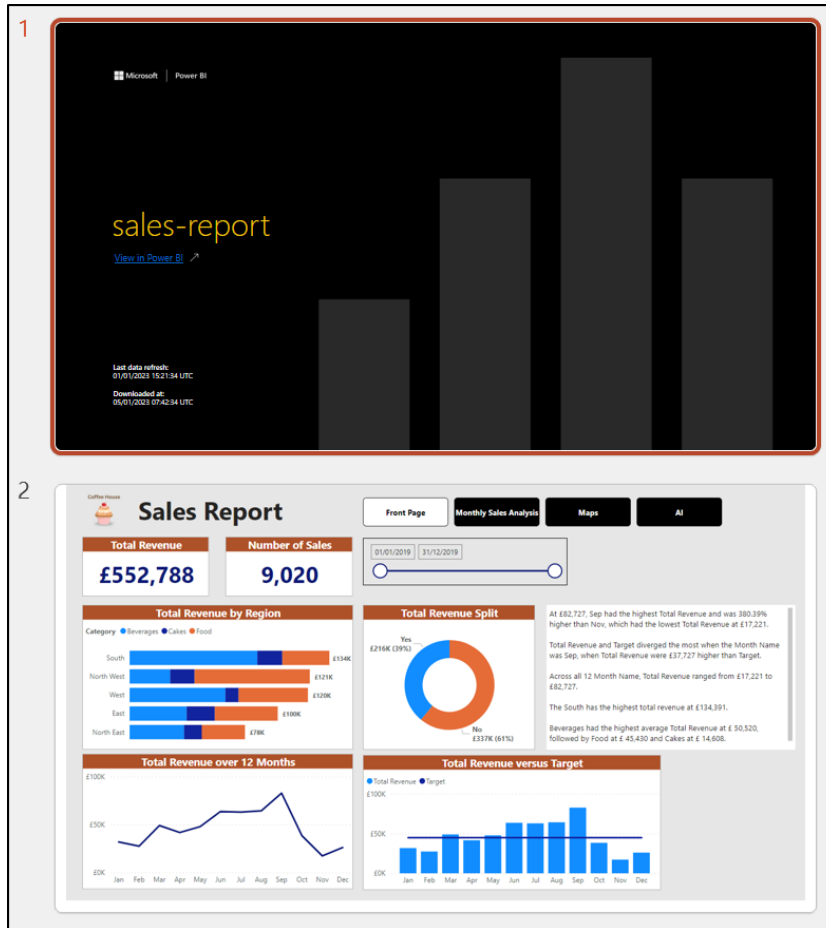


Figure 18.22: Report pages exported into a new PowerPoint presentation

Embed live data

When you export the live data to PowerPoint, you will be able to interact with the report in a PowerPoint slide show, just as you do in the Power BI service. This means that you and others can apply filters, cross-highlight visuals, copy visuals, use bookmarks, and more within PowerPoint.

1. Open the report that you want to share.
2. Click on **Export** | **PowerPoint** | **Embed live data**.

Note: When you export live data for the first time, you may be requested to install the Power BI add-in in PowerPoint.

- In the **Embed live data in PowerPoint** window (as shown in *figure 18.23*), you can click **Copy** to copy the **Report page link** and paste it into the Power BI app within PowerPoint or click on **Open in PowerPoint** to open a new PowerPoint presentation with the live data embedded:

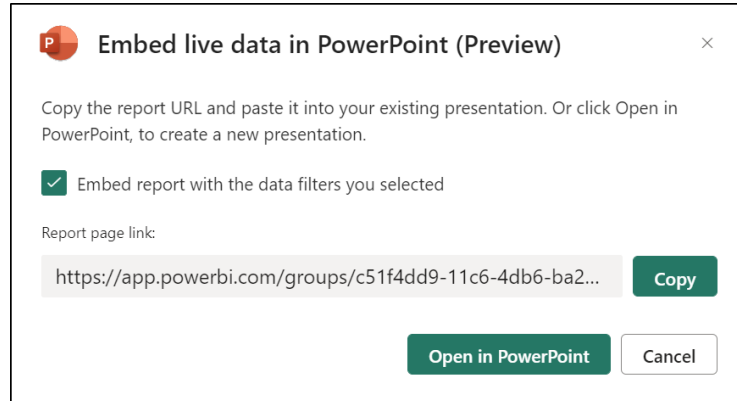


Figure 18.23: The Embed live data in the PowerPoint window

If you click on **Open in PowerPoint**, a new PowerPoint presentation is generated with a slide containing your report.

Figure 18.24 shows the report embedded as live data. Notice the **Filters** pane available on the right, the date-time stamp of when the data was last refreshed in the bottom left, and buttons in the bottom right with functionality such as refreshing the dataset.



Figure 18.24: Live data embedded into a PowerPoint presentation

If you click on **Copy** in the **Embed live data in PowerPoint** window (as shown in *figure 18.23*), the report link is copied to the clipboard, ready for pasting into the Power BI app of PowerPoint.

Within an existing PowerPoint presentation, click on **Insert | Power BI** to insert the live data, as shown in *figure 18.25*:

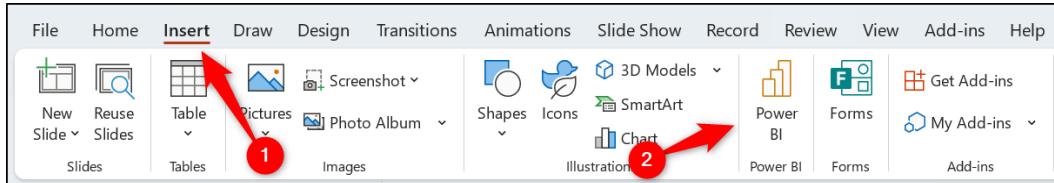


Figure 18.25: Inserting live Power BI data in PowerPoint

Paste the report link into the box provided and click **Insert** (as shown in *figure 18.26*). The report is embedded into the presentation, as shown in *figure 18.24*.

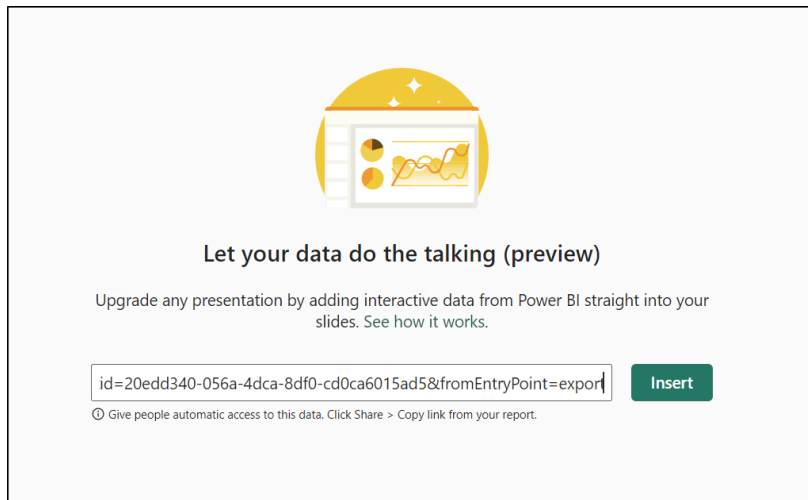


Figure 18.26: Pasting the URL into the Power BI app

Publishing to teams

Power BI can integrate with Microsoft Teams in many ways. This allows organizations to easily interact, share and collaborate on Power BI content. These methods include the following:

- Using the Power BI app for Microsoft Teams. This integrates the Power BI service experience within Teams, so users do not need to open a separate app or browser window for Power BI.

- Embed Power BI reports in Teams channels and chats. This enables easy collaboration on a report within the appropriate channel or meeting.
- Chat in Teams. This is a button found on the toolbar above a report or dashboard in the Power BI service that initiates a chat about that report with specific users.

Let us discuss a couple of these methods.

Power BI App in Microsoft teams

The Power BI app can be installed in Microsoft Teams so that you can perform typical Power BI service tasks, such as consuming, editing, and sharing reports and dashboards, without leaving Teams.

The Power BI app for Teams does not provide 100% of the features found within the Power BI service, but it contains most features.

Click on the **Apps** button on the Teams menu and locate the Power BI app to install it (or open it if it is already installed.)

Figure 18.27 shows the sales report open within Teams using the Power BI app. The Power BI app is pinned to the Teams menu on the left.

Notice the functionality to edit, export, share, and chat in Teams along the top menu, just like you see within the Power BI service. Please refer to the following figure:

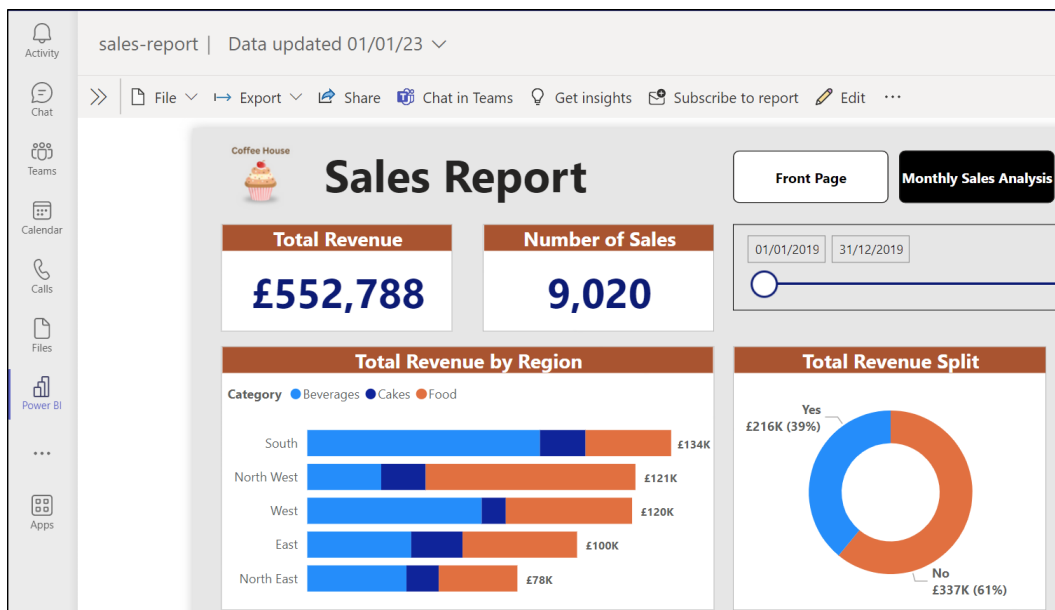


Figure 18.27: Sales report within the Power BI app for Teams

Embed reports in Teams channels and chats

Embedding reports in Microsoft Teams channels and chats make it simple to bring Power BI reports into the correct area for discussion.

It helps to keep the conversation in one place and prevent it from being dispersed across different channels such as Outlook, the Power BI service, and Slack. With so many platforms for sharing content and conversing with colleagues, comments can become lost.

Let us see how a report can be embedded into a Teams chat.

1. Open the Teams chat where you want to embed the report and click on the **Add a tab** icon, as shown in *figure 18.28*.

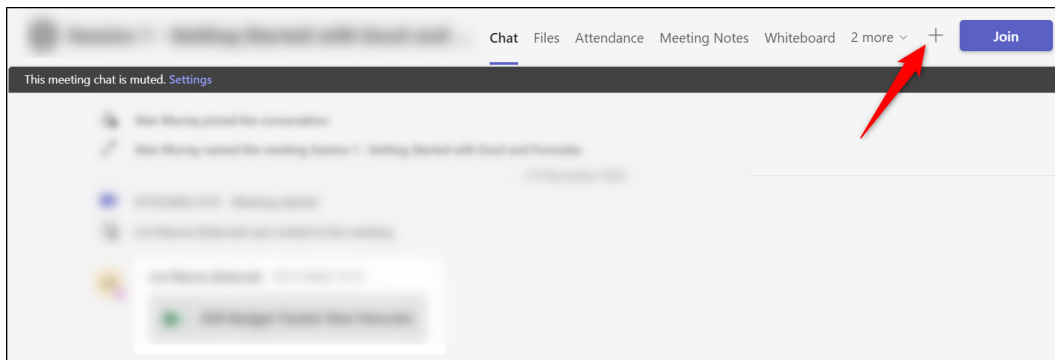


Figure 18.28: Adding a tab to a Teams chat

2. The **Add a tab** window lists popular apps to be integrated as a tab. Locate and click on **Power BI**.

- In the **Power BI** window (as shown in *figure 18.29*), select the report that you want to share and type a meaningful name into the **Tab name** box. Click on **Save**.

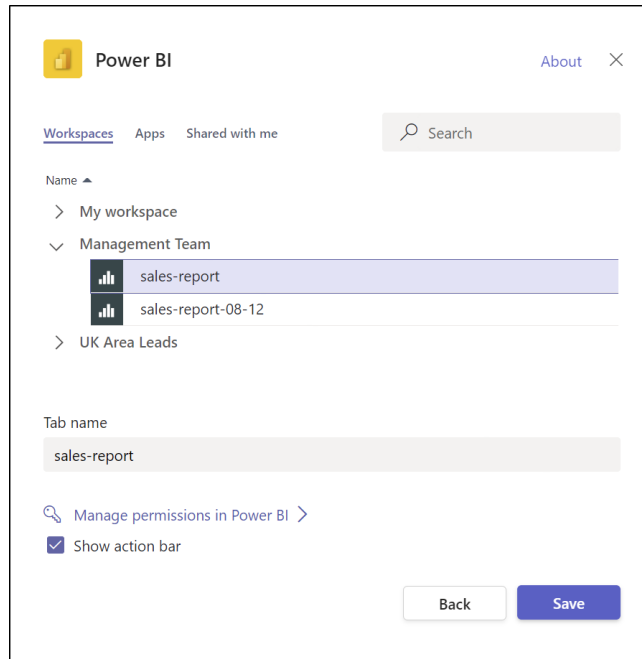


Figure 18.29: Select the report to be shared and name the tab

Figure 18.30 shows the sales report embedded in a Teams chat area. The report is fully interactive, and members of the chat can apply filters and click buttons to navigate and trigger bookmarks.

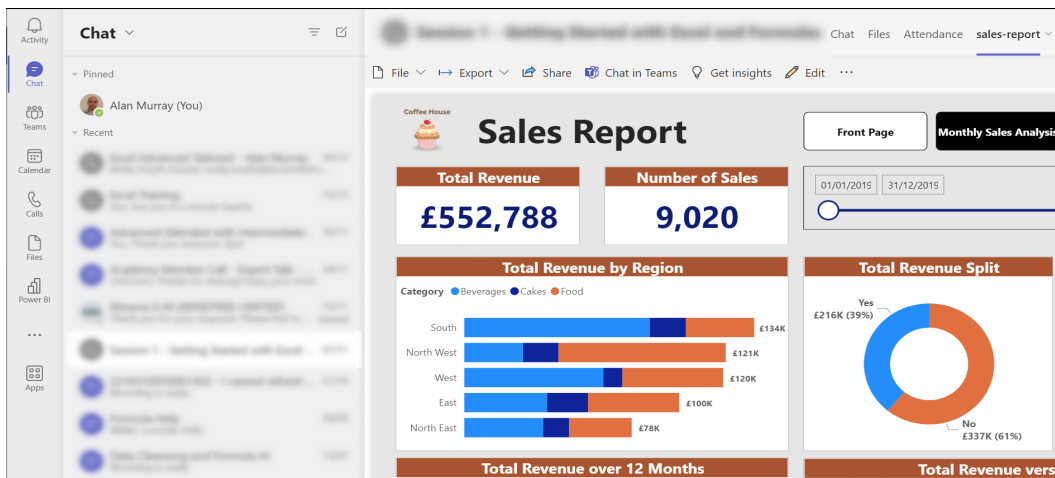


Figure 18.30: Sales report embedded into a Teams chat

Conclusion

In this chapter, we learned how to use Power BI with Microsoft Excel, PowerPoint, and Teams.

You learned how a Power BI dataset could be brought into Excel for analysis and how Excel data models could be imported into Power BI. We looked at two different methods for exporting Power BI reports into PowerPoint and how Power BI reports could be shared with others using Microsoft Teams.

In the upcoming chapter, you will learn what skills, experience and character traits recruiters are looking for when hiring a business analyst or other roles that require the use of Power BI.

We will discuss different types of interview questions you may be asked, certifications that you can earn to prove your knowledge, and how you can gain experience even when you are applying for your first Power BI role.

Questions

Here are some questions to test what you have learnt in this chapter.

1. **Which of the following statements is true when exporting a Power BI report to PowerPoint as an image?**
 - a. You can export all pages of the report or the current page only.
 - b. By default, hidden report pages are excluded from the exported images.
 - c. You can apply filters on a page and export the report page with those values.
 - d. All of the above
2. **Which process is correct to import an Excel data model using Power BI Desktop?**
 - a. File | Import | Power Query, Power Pivot, Power View.
 - b. Insert | Power Pivot.
 - c. Home | Import | Excel.
 - d. Insert | Excel data model.

3. **When exporting a Power BI dataset to Excel using Analyze in Excel, the DAX measures are moved to their own folder at the top of the PivotTable Fields list.**
 - a. True
 - b. False

4. **You cannot interact with a Power BI report that is embedded into a Teams chat to apply filters and click buttons.**
 - a. True
 - b. False

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



CHAPTER 19

Interview Questions, Certifications, and Resources

Introduction

This is the final chapter of the book. You have learned how to use Power Query, create data models, write DAX, create awesome visualizations, and share your reports. Now, it is time to talk about impressing employers and landing that job.

In this chapter, we discuss the types of typical interview questions that you may be asked when applying for BI analysts or similar roles. Also, what experiences, certifications, and proof of knowledge may you be asked to demonstrate?

Finally, we will see opportunities for further learning and development and places where you can practice your skills and gain experience.

Structure

In this chapter, we will cover the following topics:

- Types of interview questions and skills tested.
- Microsoft certifications, and are they worth it?
- How to gain experience and a portfolio of work.
- Resources for further learning and development.

Objectives

After reading this chapter, you will know what to expect from recruiters and be better prepared to prove your skills and competence in Power BI.

You will also know some good resources to continue your learning journey and strengthen the skills that you learned throughout this book.

Interview questions

When you are preparing for your first interview to land that analyst job, you will want to prepare the best that you can, and you are probably thinking, “what will they ask me?”.

Of course, nobody knows exactly what someone may ask you in an interview, but from speakers to recruitment agencies and those who recruit within companies, we have a good idea of what you can expect, and therefore, how to prepare.

An interviewer is likely to ask questions about the company for which you are seeking employment, and about you personally, but in this chapter, we will focus on questions about your Power BI knowledge and experience only.

There are three types of questions you are likely to be quizzed on in terms of Power BI: your knowledge, your experience, and your passion.

Knowledge

You are likely to be asked questions that test your knowledge of the subject. You may even be asked to demonstrate some of these skills within a Power BI report.

It is difficult for a recruiter to get a strong understanding of your knowledge just by asking a few questions. So, these questions will likely be to test your core competence in important Power BI topics. Here are some example questions.

- What is the difference between a measure and a calculated column?
- What is DAX?
- What is the difference between a workspace, a report, and a dashboard?
- How many active relationships can you have between two tables in a data model?
- What is the difference between merge queries and table relationships?
- What is Power Query?
- What is a many-to-many relationship?

These types of questions test core competencies that you would expect a Power BI analyst to answer easily. They also give the recruiter a good idea of best practices. Asking about calculated columns, many-to-many relationships, and merge queries will not only test your understanding of what they are but when they should be used.

Be sure to provide complete answers to these questions. Only a few sentences will suffice, do not ramble too much. If asked about the difference between two features or terms, such as merge queries and table relationships, be sure to detail scenarios when you may use one of the other and why.

Some questions will require a straight answer, such as what is Power Query? But for others, such as “when would you use M instead of DAX”, it is more to test your understanding. There may not be a perfect answer, but if you explain a scenario that proves your understanding of their applications, that is, what they want.

Experience

Power BI is still relatively fresh. It was introduced around 2015, and it took a few years before it started to become what it is today. Because of this, most people looking for work have a handful of years of experience at best.

If you are applying for your first job, do not worry. In speaking to recruiters, depending on the role, experience is not as important as proof of competence and passion. Now, that is not true for all, but it is something that I heard numerous times.

There are many ways that you can gain experience in Power BI before your first job. One method is to take part in challenges and contests that are posted by training companies, enthusiasts on LinkedIn, and popular YouTubers. You can take part in these challenges and contests to both practice your skills and gain experience that you can show to an employer.

Search for “Power BI” on LinkedIn, and you will find leaders in the industry and passionate people from the community who are regularly sharing tips and different contests that they are participating in.

Another method is to compete for jobs on freelance sites such as **upwork.com**. There are many freelance sites that you can sign up for to get work, get paid, but most importantly, get experience and build a portfolio of work that can be shown to employers.

You can show recruiters the reports and models that you have created and the “real work” experience that you have gained.

Passion

The Power BI community is extremely passionate. Microsoft releases a monthly update for Power BI, and this is eagerly awaited by the community to read about what has changed or new features that have appeared, and often give their opinion on it.

Demonstrating that you are at the “cutting edge” of Power BI, that you follow the Microsoft blog, participate in the community, and endeavor to learn more is a huge plus for many recruiters.

With Power BI changing and evolving so regularly, recruiters are looking for people that have their “finger on the pulse” and are keeping up with the changes in Power BI. You do not necessarily need to be an expert in everything, but you are able to show a willingness to stay informed.

The strong link that Microsoft has with the community is shown by the existence of the *Help* tab in Power BI Desktop with buttons for the *Power BI blog* and *Community*. In the Power BI service, there is a *Learn* button in the menu on the left that provides access to the blog and community too.

Recruiters will be impressed to hear that you are directly involved in the community. Answering questions in the forum, commenting on the blog, and submitting and reacting to ideas. You can show a recruiter your profile to prove your involvement.

This demonstrates your passion and confirms that you are knowledgeable, helpful, and hard-working. Who would not want to see those attributes in someone they are thinking of employing?

Certifications

Depending on the role that you are applying for, having a degree in something such as Data Science or Mathematics can certainly help. It shows a level of education and commitment.

Many people use Power BI in their role, and having such a qualification is often not necessary. Experience, passion, and dedication often are more valuable.

Microsoft has its own Power BI certification known as *PL-300: Microsoft Power BI Data Analyst*. This used to be named the *DA-100: Analyzing Data with Microsoft Power BI*, but that was retired on March 31, 2022, and replaced by the PL-300 exam.

This exam tests your skills across four areas:

- Preparing the data
- Modeling the data

- Visualizing and analyzing data
- Deploying and maintaining assets.

To pass the exam, you must show proficiency in all areas of Power BI, including Power Query, writing DAX expressions, and working in the Power BI service.

To prepare for the exam, you can take the lessons provided in the structured learning path at learn.microsoft.com. Of course, these are skills that you can learn elsewhere, including the content of this book. However, this is the official learning path, so you should be confident that you will not be tested on something that is not covered in a lesson.

The exam currently costs \$165 to sit and can be taken in person at a test center or at home.

On speaking to recruiters, this certification is nice to have. It demonstrates your knowledge and devotion to learning. However, it is certainly not a requirement and not a priority in what recruiters are looking for.

Useful resources

There are many great resources to continue your Power BI learning and re-enforce what you learned in this book. The following is a collection of some of the best resources to learn Power BI.

Websites

The best website to keep up to date on the latest Power BI developments is the Microsoft Power BI blog (<https://powerbi.microsoft.com/en-us/blog/>). Here you will find the latest announcements from Microsoft and the monthly feature summary as soon as it comes out.

You can subscribe to the Microsoft Power BI blog by e-mail to ensure that you hear announcements as soon as they happen.

For the latest articles, community posts, and support, you want the community website (https://community.powerbi.com/t5/Community-Blog/bg-p/community_blog). This website is full of interesting links and articles, and if you need some support from the community, this is the place to be.

Other fantastic websites include Chris Webb's BI blog (<https://blog.crossjoin.co.uk/>). Chris works for the Microsoft Power BI CAT team and regularly posts in-depth articles about whatever he finds interesting at the time.

The SQLBI blog (<https://www.sqlbi.com/articles/>) is the best resource for DAX. Marco Russo and Alberto Ferrari are well-known as the tip of the spear when it comes to DAX skills.

YouTube channels

YouTube is a great resource for learning, and some of the best Power BI channels include:

The **Microsoft Power BI official channel** for videos on the monthly feature update, customer success stories, and other Power BI events.

Guy in a Cube for helpful tips and lessons in Power BI to be successful as a business analyst. Adam Saxton and Patrick LeBlanc are both Microsoft employees and post videos regularly. They offer a fun and simple explanation for even more difficult topics.

Podcasts

The best Power BI podcast is Explicit Measures (<https://powerbi.tips/explicit-measures-power-bi-podcast/>), hosted by Mike Carlo, Seth Bauer, and Tommy Puglia. You can find it in all the places where you would usually find a podcast.

They deliver two podcasts a week and discuss the latest developments in Power BI, and then have a detailed discussion on the designated topic of that session.

All three are data platform Microsoft MVPs and are directly involved in building Power BI reports for clients. So, they offer a “real world” perspective and opinions on the application of different Power BI tools and the best practice.

Meetups and conferences

Another great resource in the community is the number of events that occur around the world. There are events every month. Some of them in-person, and some held virtually. Some events are paid for, and others are free.

Microsoft has its own events that are held annually, including Microsoft Ignite and Dashboard in a Day. You can see the upcoming power platform events at their website (<https://events.microsoft.com/en-us/powerplatform>).

There are also the Power BI User Groups (<https://www.pbusergroup.com/home>). There are over 270 PUG groups around the world hosting in-person and virtual meetups on Power BI topics so that you can learn, share, and network with other Power BI people. These groups are free to attend.

Conclusion

In this chapter, we learned some tips on how to prepare to land your first Power BI job. You learned different methods in which you can gain experience and proof of the skills that you have attained through reading this book and elsewhere.

You also learned some of the best resources for continuing your Power BI education and keeping up with the consistent progress of Power BI and other apps in Microsoft's data platform.

Questions

Here are some questions to test what you have learned in this chapter.

- 1. Which of the following methods can be used to demonstrate your knowledge and expertise before you have “on-the-job” experience?**
 - a. Answering questions in Power BI forums.
 - b. Completing challenges and participating in Power BI report contests.
 - c. Complete small jobs on freelance sites.
 - d. All of the above.
- 2. What is the name of the Power BI certification offered by Microsoft?**
 - a. DA-100.
 - b. PBI-1000.
 - c. PL-300.
 - d. Microsoft does not offer a Power BI data analyst certification.
- 3. You can access the Power BI community and blog easily from both Power BI Desktop and the Power BI service.**
 - a. True
 - b. False
- 4. How many Power BI User Groups are there around the world?**
 - a. 0.
 - b. 5–10.
 - c. 100–150.
 - d. More than 270.

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



Multiple Choice Questions Answers

The following are the answers to the multiple choice questions at the end of each chapter.

Answers

Chapter 2

1. **d:** All of the above. Dashboards, reports, and workspaces are all found in the Power BI service.
2. **b:** False. You do not need a Power BI account to use Power BI Desktop, but you will need one to publish reports.
3. DAX stands for Data Analysis Expressions.
4. **c:** Power Query is used to get, transform, and shape data ready for analysis.

Chapter 3

1. **b:** Visuals are created in the Report view of Power BI Desktop.
2. **a:** The Model view is best for seeing the relationships between the tables of the data model.

3. The extension of a Power BI file is .pbix
4. **a:** True. Hidden data tables are created for all table columns of a date data type.

Chapter 4

1. **d:** Column From Examples, Split Column, and Choose Columns are all transformation tools available in Power Query.
2. **a:** True. You can disable the automatic Changed Type step of Power Query.
3. **b:** False. Specify a data type and formatting values are not the same. A data type specifies how the value is stored, and its type will determine which operations can be performed on it. Formatting is the presentation of the value and is done in Power BI, not Power Query.
4. **c:** The Trim transformation is used to remove excess spaces.
5. **a:** True. You can change the order of values in a column chart.
6. **b:** Click Home > Transform Data to edit existing queries in the Power Query Editor.

Chapter 5

1. **d:** DirectQuery, Import, and Dual are all types of storage modes in Power BI.
2. **a:** True. Power Query is case-sensitive.
3. **a:** Click on the column > Transform > Format.
4. **d:** The Web connector is the correct connector to use when importing data from a file on OneDrive or SharePoint. This ensures that you connection to the cloud version and not the local version of the file.

Chapter 6

1. **d:** The Days in Month, End of Month, and Age calculations are all found in the date calculations list in Power Query.
2. **b:** Complete is not a valid join type when performing merge queries. Inner, Left Outer, and Left Anti are all different join types.
3. **a:** The correct path to unpivot columns is to click on the columns > Transform > Unpivot Columns.

4. **d:** Left Outer is the join type to be used to pull across values for all matching items in the first table.
5. **d:** Standard, Rounding, and Statistics are all calculation groups on the Transform tab in Power Query.

Chapter 7

1. **d:** Many-to-one, one-to-one, and many-to-many are all types of cardinality found in Power BI. Many-to-one is the most utilized.
2. **a:** Power BI auto-detects table relationships by default.
3. **c:** You would not hide a field from the report view if you planned to use it in a report visual.
4. **b:** Preventing a query from loading to Power BI does not prevent it from being refreshed with the dataset.
5. **a:** You can format table columns in all three Power BI views.

Chapter 8

1. **d:** Avoiding bloated models due to the hidden date table and duplicate fields and the ability to create only the date attribute fields that you need are all reasons to create a date table.
2. **b:** The FORMAT function was used for the month name and day of week name fields.
3. **d:** Data view > click on the column > Column tools > Sort by column.
4. **d:** CALENDARAUTO, QUARTER, and IF are all DAX functions in Power BI.

Chapter 9

1. **d:** All of the above. Advantages to writing explicit measures include that they only calculate once, can be reused in other measures, and you can utilize the rich DAX language for powerful calculations.
2. **c:** The CALCULATE function was used to modify the filter context for the total revenue measure.
3. **d:** All of the above are methods to create a new measure in Power BI.
4. **b:** RELATEDITEM is not a valid DAX function.

Chapter 10

1. **b:** The names of the two formatting categories for all visuals are Visual and General.
2. **a:** True. You can sort the values in a table visual by multiple columns.
3. **d:** Value, Trend axis, and Target are all fields for the KPI visual.
4. **a:** True. You can use the Format Painter command to quickly copy formatting between visuals.
5. **b:** False. You can change the good, bad, and neutral colors of the Trend axis text in a KPI visual.

Chapter 11

1. **a:** The primary purpose of a line chart is to show the change in a value over time.
2. **a:** True. You can set the max value for the axis of a Gauge chart.
3. **d:** The pie, gauge, and column charts all provide a tooltip feature.
4. **d:** All of the above. You cannot change the sort order of values in a treemap visual, but you can change the colors of the branches and add tooltip values.

Chapter 12

1. **a:** True. Using latitude and longitude values ensures the greatest chance of accuracy when plotting data on a map.
2. **d:** All of the above. With the map visual, the Bing maps geocoding engine is used to correctly map location fields, you can change the color of the bubbles, and you can change the map style.
3. **b:** From Data view, select the column > Column tools > Data category.
4. **a:** True. The auto zoom feature of a map can be disabled if required.

Chapter 13

1. **d:** All of the above. The Q&A visual can be inserted by double-clicking on the page canvas, clicking Insert > Q&A, and clicking the Q&A icon in the Visualizations pane.

2. **d:** All of the above. Creating suggested questions, adding synonyms, and reviewing questions that have been asked are all methods to optimize the Q&A visual further.
3. **a:** True. You can remove a custom visual from a report when it is no longer required.
4. **a:** True. The smart narrative visual can be created for all visuals on a page or for a specific visual only.

Chapter 14

1. **b:** Click on the visual and click Format > Edit Interactions.
2. **d:** All of the above. Hiding the slicer header, specifying the single select option, and changing the alignment of the slicer items are all formatting options for a slicer using a text field.
3. **a:** True. Cross-highlighting and cross-filtering between visuals can be disabled by default.
4. **a:** True. The Filters pane can be hidden on the report when in reading view.

Chapter 15

1. **c:** Next is not an action that can be assigned to an image or button.
2. **d:** All of the above. Update, Display, and Selected visuals are all found in the More options list for bookmarks.
3. **a:** True. An image can be used for the Canvas background.
4. **a:** True. A custom tooltip page can be applied to more than one visual in a report.

Chapter 16

1. **c:** The report and dataset are published when you publish the report from the Power BI Desktop.
2. **a:** A repository on the service to save your reports, datasets, dashboards, and Excel workbooks.
3. **a:** True. You can create new reports from existing datasets that have been published to the Power BI service.
4. **a:** True. When you share the report with others, they can interact with the report fully but cannot edit it.

Chapter 17

1. **b:** The workspace roles that you can assign are admin, member, contributor, and viewer.
2. **a:** True. You can schedule up to 48 refreshes per 24 hours with a PPU license or if the dataset resides on a premium capacity. On a Pro license, you are limited to 8 refreshes per 24 hours.
3. **d:** Dashboards are not a feature available in Power BI Desktop, they consist of only one page, and you can pin visuals from one or more reports that have been published to the service.
4. **a:** True. You can copy reports to other workspaces.

Chapter 18

1. **d:** All of the above. When exporting a Power BI report to PowerPoint as an image, you can specify to export the current page only, or all pages and hidden report pages are excluded unless specified, and you can export the report pages with the currently filtered values.
2. **a:** File | Import | Power Query, Power Pivot, Power View.
3. **a:** True. When exporting a Power BI dataset to Excel using Analyze in Excel, the DAX measures are moved to their own folder at the top of the PivotTable Fields list.
4. **b:** False. The report is fully interactive, and members of the chat can apply filters and click buttons to navigate and trigger bookmarks.

Chapter 19

1. **d:** All of the above. Answering questions in forums, participating in challenges and Power BI dashboard contests, and completing small freelance jobs are great ways to improve your Power BI skills, experience, and build a portfolio of work.
2. **c:** The Power BI certification offered by Microsoft is the PL-300.
3. **a:** True. You can access the Power BI community and blog easily from both Power BI Desktop and the Power BI service.
4. **d:** There are over 270 Power BI User Groups around the world hosting in-person and virtual meetups.

Index

A

Access Analytic 7

Analyze

using, in Excel 462-466

B

bookmarks 405, 406

buttons, inserting 410, 411

creating 406-409

C

CALCULATE function 226

calculations, Power Query 137, 138

columns, unpivoting 151-154

conditional columns, inserting 148-151

data type, defining 144

date calculations 145-148

mathematical calculations 138-140

M code, adding 141

M code, editing 141

rounding method, changing 142, 143

values, rounding 140, 141

cardinality 160

types 161

Card visual 238, 239

border, adding 242

callout value, formatting 239, 240

category label, versus title 240, 241

format painter 243-245

certifications 488, 489

chart visuals

column and bar charts 270

combo chart 285-288

gauge charts 292-294

linear and area charts 280

- pie and donut charts 289-291
 - Treemap visual 297-299
- clustered column chart
 - axis, removing 276
 - axis titles, removing 276, 277
 - border, adding 275
 - chart title, changing 273, 274
 - column colors, changing 276
 - formatting 273
 - using 270
- column and bar charts
 - bar chart, switching to 280
 - clustered column chart 270
 - data labels, adding 277
 - stacked column chart 278-280
 - tooltips, adding 271-273
- column chart
 - creating 59-62
- combo chart 285-288
- COUNTROWS function 216, 217
- custom tooltip pages 397, 398
 - adding, to visual 404, 405
 - creating 399-401
 - visuals, adding 401-403
- custom visual 345
 - adding, from AppSource 345-348
 - removing 349
- D**
- dashboard 454
 - creating 454, 455
 - sharing 457-459
 - visuals, pinning 455-457
- Data Analysis Expressions (DAX) 183,

- 184
- CALENDARAUOTO function 184-188
- CALENDAR function 184-190
- data import, from Excel workbook 81-83
 - columns, renaming 87
 - columns, splitting 85, 86
 - data types, checking 87
 - rows, removing 83-85
- data import, from multiple files in folder 95-97
 - columns, renaming 105
 - data, combining into one table 98-101
 - data types, checking 105
 - text case, changing 104
 - unwanted characters, removing 101-103
- data import, from OneDrive/SharePoint files 105-110
 - columns, renaming 112, 113
 - data types 112
 - tables, combining 110-112
- data import, from PDF files 87-89
 - columns, merging 93, 94
 - columns, renaming 95
 - data types, checking 95
 - queries, appending 89-92
 - table headers, promoting 92
- data load options, Power BI Desktop
 - background data 36
 - time intelligence 36
 - type detection 35
- data model

- fields, deleting 175, 176
- query loading, disabling 166-168
- table columns, formatting 169-172
- table fields, hiding from
 - Report view 172-174
- table relationships, creating 161-166
- datasets
 - refreshing 439
- data source settings
 - global permissions, changing 116, 117
 - in current file 114, 115
 - managing 114
 - source, changing in Power Query Editor 117-119
- data transformations, in Power Query 47, 48
 - column data types, changing 54-57
 - columns, removing 48, 49
 - columns, renaming 54
 - columns, splitting by delimiter 50-52
 - excess spaces, removing with trim 53, 54
 - negative values, converting to positive values 52, 53
 - steps, renaming 57, 58
- DATEADD function 226, 227
- date calculations, Power Query
 - age calculating 145-148
- date columns
 - creating 190, 191
 - data table, creating with one formula 200
 - day of week name 193, 194
 - fiscal months 197, 198
 - fiscal quarters 196, 197
 - fiscal years 195, 196
 - month 191, 192
 - weekend date, checking 195
 - week number of year 198, 199
 - year 191
 - year and month 193
- date data type
 - establishing 132
 - ISO 8601 dates, converting 135-137
 - local data type, using 132-135
- date table
 - columns, sorting by another column 201, 202
 - fields, hiding 205
 - purpose 180-182
 - relationships, creating 203
 - table, marking as 204, 205
- DAX measure 210, 211
 - implicit, versus explicit measures 211-216
 - moving 218
 - organizing 217
 - percentage revenue change, calculating 231, 232
 - previous months revenue, returning 225
 - revenue difference between two months, calculating 229-231
- SUMX function 223, 224
- table, creating 221-223
- tables, creating within tables 219, 220
- writing 227-229
- year-to-date total, creating 232-234

decomposition tree visual 336-338

drill through filter 376

 creating 377, 378

 page title, adding 380-382

 performing 378-380

E

Excel report

 Analyze, using 462-466

 creating, from Power BI datasets 462

 Power BI datasets, connecting to 466-469

experience 487

F

Filters pane 371

 Top N filter, setting 372-374

 visuals, filtering 374-376

G

Gartner Magic Quadrant 2022 2, 3

gauge charts 292-294

 general formatting 294

 KPI, versus gauge charts 295, 296

 visual formatting 294, 295

Goldmeier, Jordan 8

H

Hopkins, Wyn 6, 7

I

Import mode

 versus, DirectQuery mode 80, 81

interview questions 486

J

Join Kinds 122, 123

Full Outer 124

Inner 124

Left Anti 124

Left Outer 123

Right Anti 124

Right Outer 124

K

Key Performance Indicator (KPI)
 visual 245-247

 icon, removing 249

 Slicer, inserting 247-249

 target label, changing 251

 title and border, adding 252, 253

 trend axis options 250

knowledge

 testing 486

L

linear and area charts 280

 line chart 281-283

 line chart, with multiple data
 series 283, 284

 stacked area chart 285

M

Map visuals 301

 auto-zoom feature, disabling 312, 313

 bubbles, formatting 310

 enabling, in Power BI 302

 example 303, 304

 geographic fields, categorizing 314

 geographic hierarchies, using 319, 320

 latitude and longitude data, using
 315-318

 map style, changing 309

map title, editing 308
 table, for filtering map data 310-312
 tips 313
 using 304-307

matrix 260
 column headers, formatting 262
 drill down actions, for multiple
 row headers 263-266
 grid and values, formatting 261

Microsoft Teams
 Power BI app 480
 Power BI report, embedding in
 channels and chats 481, 482

model relationships
 managing 164-166

P

personal mode gateway 440
 pie and donut charts 289-291

Power BI 1, 11
 components 12
 integrating, with Excel 462
 jobs at LinkedIn UK 3
 jobs at Naukri 4
 jobs at Nigel Frank International 5
 jobs on Upwork 6
 professionals opinions 6
 publishing, to Teams 479, 480
 queries, applying 58, 59
 skills 3

Power BI account
 creating 24, 25

Power BI data gateway 439
 data refreshes, scheduling 452, 453

data, refreshing 451, 452
 data sources, adding 445-451
 personal mode gateway 440
 standard (enterprise) mode gateway
 440
 virtual network gateway 440

Power BI datasets
 connecting, from Excel 466-469

Power BI Desktop 13
 Data Analysis Expressions
 (DAX) 15
 data load options 35
 data modeling 14
 Data view 31, 32
 downloading, from
 Microsoft store 20, 21
 downloading, from website 21-23
 file, saving 41, 42
 installing 19
 Model view 33, 34
 options 34, 35
 power query 13, 14
 preview features 37
 query reduction 38, 39
 regional settings 38
 Report settings 40
 reports, publishing 16
 Report view 29-31
 settings 34
 visualizations 15
 welcome screen 28, 29

Power BI file
 creating 78

Power BI mobile 19

Power BI report 439

- accessing, on service 416-418
- column chart, creating 59-62
- column chart visual, modifying 65, 66
- column, using from examples 63-65
- creating 43
- creating, from existing dataset 420
- creating, from Power BI Desktop 423, 424
- creating, from Power BI Service 420-423
- data, getting from web 44-46
- datasets, using 419, 420
- data, transforming in Power Query 47, 48
- existing queries, editing 62
- interactions, between visuals 74, 75
- link, sharing 425-427
- permissions, managing 427, 428
- publishing 414-416
- table visual, creating 68-73
- X-axis order, changing 67, 68

Power BI report, enhancing

- background, adding 395-397
- bookmarks, using 405, 406
- buttons, adding 390
- custom tooltip pages 397, 398
- drill through back button, changing 391, 392
- images, adding 387-390
- page navigator buttons 393-395
- text boxes, adding 386, 387

Power BI report export

- image embedding 475, 476

- live data embedding 477-479
- to PowerPoint 474, 475

Power BI service 16, 17

- apps 19
- dashboard 18, 19
- dataset 18
- report 18
- workspaces 17

Power Pivot models

- importing, from Power BI Desktop 470-472
- importing, into Power BI 469

Power Pivot service

- importing, from Power BI Desktop 472-474

Power Query

- calculations 137
- data, getting with 79

Prescott, Nathalie 7**Q****Q&A visual** 324

- configuring 331, 332
- formatting 330, 331
- suggested questions, creating 335
- synonyms, adding 332-334
- using 324-330

queries

- disabling, from loading 166-168
- merging 122

query merging examples

- Left Anti Join 129-131
- Left Outer Join 124-129

R

Report view

- table fields, hiding from 172-174

resources

- conferences 490
- podcasts 490
- websites 489
- YouTube channels 490

S

slicers 361

- date fields, using 361-366
- text fields, using 366-370
- visuals, filtering on other pages 370, 371

smart narrative visual 339

- creating 339-341
- dynamic values, adding 342
- dynamic values, formatting 342-345

stacked area chart 285

stacked column chart 278, 279

- switching, to bar chart 280

standard (enterprise) mode gateway 440

- installing 440-444

SUMX function 223, 224

T

table columns

- formatting 169-172

table fields

- hiding, from Report view 172-174

table relationships

- creating 161-163
- purpose 158-160

Table visual 253, 254

- column headers, formatting 256, 257
- grid and values, formatting 254-256
- table data, sorting 257
- table, sorting by multiple columns 258, 259

Treemap visual 297-299

V

virtual network gateway 440

visual interactions

- editing 354-360

visualizations 323

- custom visual 345
- decomposition tree visual 336-38
- Q&A visual 324
- smart narrative visual 339

W

workspaces 432

- access, managing 434-436
- creating 432-434
- report, copying 437, 438

Power BI for Jobseekers

DESCRIPTION

Power BI is a powerful Business Intelligence tool developed by Microsoft that helps you connect, analyze, and visualize data from a wide range of sources. If you are looking to gain a solid understanding of Power BI, then this book is for you.

This book starts by covering the fundamentals of using the tool. It then teaches you how to import data from various sources, use DAX calculations, and take advantage of many visualization and filtering features in Power BI to create visually appealing and informative reports. Finally, the book covers different ways to share your Power BI reports and dashboards with others.

By learning Power BI, you can stay ahead of the competition and gain a competitive edge in the field of data analysis and visualization.

KEY FEATURES

- Learn how to transform, shape, and model your data in Power BI.
- Create rich, interactive and stunning reports using Power BI.
- Understand what recruiters are looking for and how to get started with a career in business analytics.

WHAT YOU WILL LEARN

- Learn how to use Power BI to connect to multiple data sources.
- Create tables, columns, and measures in Power BI using DAX.
- Explore ways to enhance your Power BI reports.
- Use Power BI with Excel, PowerPoint, and Microsoft Teams.
- Publish, share, collaborate, and update your Power BI reports.

WHO THIS BOOK IS FOR

This comprehensive book caters to professionals who want to pursue a career in data analysis. It is also designed for beginners and Excel users who wish to enhance their data analysis and reporting skills beyond conventional methods.



BPB PUBLICATIONS

www.bpbonline.com

ISBN 978-93-5551-814-9



9 789355 518149