Microsoft

Collect, Combine, and Transform Data Using Power Query in Excel and Power BI

Gil Raviv

Sample files on the web

FREE SAMPLE CHAPTER

Microsoft

Collect, Combine, and Transform Data Using Power Query in Excel and Power BI

Gil Raviv

COLLECT, COMBINE, AND TRANSFORM DATA USING POWER QUERY IN EXCEL AND POWER BI

Published with the authorization of Microsoft Corporation by:
Pearson Education, Inc.
Copyright © 2019 by Gil Raviv

ISBN-13: 978-1-5093-0795-1
ISBN-10: 1-5093-0795-8

Library of Congress Control Number: 2018954693

01 18

PUBLISHER
Mark Taub

ACQUISITIONS EDITOR
Trina MacDonald

DEVELOPMENT EDITOR
Ellie Bru

MANAGING EDITOR
Sandra Schroeder

SENIOR PROJECT EDITOR
Tonya Simpson

COPY EDITOR
Kitty Wilson

INDEXER
Erika Millen

PROOFREADER
Abigail Manheim

TECHNICAL EDITOR
Justin DeVault

COVER DESIGNER
Twist Creative, Seattle

COMPOSITOR
codemantra

COVER IMAGE
Malosee Dolo/ShutterStock

Contents at a Glance

Contents

Figure Credits

Chapter 1, "Introduction to Power Query," Figures 1-1 through 1-9 courtesy of Microsoft Corporation.

Chapter 2, "Basic Data Preparation Challenges," Figures 2-1 through 2-16 courtesy of Microsoft Corporation.

Chapter 3, "Combining Data from Multiple Sources," Figures 3-1 through 3-8 courtesy of Microsoft Corporation.

Chapter 4, "Combining Mismatched Tables," Figures 4-1 through 4-11 courtesy of Microsoft Corporation.

Chapter 5, "Preserving Context," Figures 5-1 through 5-11 courtesy of Microsoft Corporation.

Chapter 6, "Unpivoting Tables," Figures 6-1 through 6-9 courtesy of Microsoft Corporation.

Chapter 7, "Advanced Unpivoting and Pivoting of Tables," Figures 7-1 through 7-5 courtesy of Microsoft Corporation.

Chapter 8, "Addressing Collaboration Challenges," Figures 8-1 through 8-10 courtesy of Microsoft Corporation.

Chapter 9, "Introduction to the Power Query M Formula Language," Figures 9-2 through 9-10 courtesy of Microsoft Corporation.

Chapter 10, "From Pitfalls to Robust Queries," Figures 10-2 through 10-9 courtesy of Microsoft Corporation.

Chapter 11, "Basic Text Analytics," Figures 11-1 through 11-14, Figures 11-16 and 11-17 courtesy of Microsoft Corporation.

Chapter 12, "Advanced Text Analytics: Extracting Meaning," Figures 12-3 through 12-8, Figures 12-10 through 12-14 courtesy of Microsoft Corporation.

Chapter 13, "Social Network Analytics," Figures 13-1 through 13-11 courtesy of Microsoft Corporation.

Chapter 14, "Final Project: Combining It All Together," Figures 14-1 through 14-3 courtesy of Microsoft Corporation.

Foreword

When we set out to build the original Power Query add-in for Excel, we had a simple yet ambitious mission: connecting to and transforming the world's data. Five years later, we've moved beyond the original Excel add-in with native integration into Excel, Power BI, Power Apps, and a growing set of products that need to extract and transform data. But our original mission remains largely unchanged. With the ever-increasing heterogeneity of data, in many ways, our mission feels even more ambitious and challenging than ever. Much of today's computing landscape is centered around data, but data isn't always where or how you need it—we continue to advance Power Query with the goal of bridging that gap between the raw and desired states of data.

Throughout the Power Query journey, the user community has played a critical role in shaping the product through suggestions and feedback. The community has also played a central role in developing valuable educational content. As one of the key drivers of Power Query's native integration into Excel 2016, Gil is well placed to provide valuable insights and tips for a variety of scenarios. Even after his tenure at Microsoft, Gil has remained an active and influential member of the Power Query community. Happy querying!

—Sid Jayadevan, Engineering Manager for Power Query, Microsoft Corporation

For readers not familiar with Power Query, it is an incredibly powerful and extensible engine that is the core of Microsoft BI tools. It enhances self-service business intelligence (BI) with an intuitive and consistent experience for discovering, combining, and refining data across a wide variety of sources. With data preparation typically touted as 80% of any BI solution, having a firm grasp of Power Query should be your first step in any sort of reporting or data discovery initiative. In addition to the core Power Query functionalities, Gil covers more advanced topics, such as how to use Power Query to automate data preparation and cleansing, how to connect to social networks to capture what your customers are saying about your business, how to use services like machine learning to do sentiment analysis, and how to use the M language to make practically any type of raw data a source of insights you glean value from. This book stands out in that it provides additional companion content with completed samples, data sources, and step-by-step tutorials.

Gil is a former member of the Excel team and the Microsoft Data Team. He directly contributed to the features and design of Power Query and has an amazing wealth of knowledge using Power Query and showing how it can make difficult data integration problems easy. That said, despite Power Query's inherently extensible and easy-to-use design, mastering it for enterprise scenarios can still be difficult. Luckily for the reader, as an avid community member, forum contributor, conference presenter, peer mentor, and Power BI MVP, Gil Raviv is a master at taking complex concepts and decomposing them into very easy-to-follow steps, setting the reader up for success and making this book a must have for any BI specialist, data systems owner, or businessperson who wants to get value out of the data around him.

—*Charles Sterling, Senior Program Manager,*
Microsoft Corporation

About the Author

Gil Raviv is a Microsoft MVP and a Power BI blogger at https://DataChant.com. As a Senior Program Manager on the Microsoft Excel Product team, Gil led the design and integration of Power Query as the next-generation Get Data and data-wrangling technology in Excel 2016, and he has been a devoted M practitioner ever since.

With 20 years of software development experience, and four U.S. patents in the fields of social networks, cyber security, and analytics, Gil has held a variety of innovative roles in cyber security and data analytics, and he has delivered a wide range of software products, from advanced threat detection enterprise systems to protection of kids on Facebook.

In his blog, DataChant.com, Gil has been chanting about Power BI and Power Query since he moved to his new home in the Chicago area in early 2016. As a Group Manager in Avanade's Analytics Practice, Gil is helping Fortune 500 clients create modern self-service analytics capability and solutions by leveraging Power BI and Azure.

You can contact Gil at gilra@datachant.com.

Acknowledgments

Writing this book is one of the scariest things I have willingly chosen to do, knowing I was going to journey into an uncharted land where only a few people have gone before and approach an ever-evolving technology that is relatively unfamiliar yet can drastically improve the professional lives of many users. How can I share the knowledge of this technology in a way that will enable you to harness its true essence and empower you to make a real impact on your business?

The writing of this book would not have been possible without the help and inspiration I received from many people.

First, I would like to thank my readers at DataChant.com. Your feedback and support made this endeavor possible. You have taught me the power of sharing.

Thank you to my wife and children, for being stranded at home with me for many days in late 2017 and the colder parts of 2018 to support my work. Thank you for your support. I hope you can also blame the winter in Chicago for staying with me so many weekends.

Special thanks to Trina MacDonald, my senior editor at Pearson. You reached out to me one day with an idea to write a book and have been supporting me all the way in publishing a completely different one. Good luck in your new journey.

Thank you to Justin DeVault, my first Six Sigma Master Black Belt client. As a technical editor, you combined your business savvy and technical prowess to review 14 chapters, 71 exercises, and 211 exercise files to ensure that the book can deliver on its promise. Without your insights, we could not have made it. You were the best person for this job.

To Microsoft Press, Pearson, Loretta Yates and the whole publishing team that contributed to the project, thank you! Thank you, Songlin Qiu, Ellie Bru, and Kitty Wilson for editing and proofreading and Tonya Simpson for orchestrating the production efforts; you have all magically transformed 14 chapters of Word documents into this book.

To my dear friend Yohai Nir, thank you for the rapport and guidance through the initial stages of the book.

Thank you to Luis Cabrera-Cordon, for reviewing Chapter 12. I hope that this chapter will help more business analysts use Microsoft Cognitive Services and gain new insights without the help of developers or data scientists.

To the amazing Program Managers Guy Hunkin, Miguel Llopis, Matt Masson, and Chuck Sterling: Thank you for the ongoing support and technical advice. Your work is truly inspirational.

Sid Jayadevan, Eli Schwarz, Vladik Branevich, and the brilliant people on the Redmond and Israeli development teams: It was a real pleasure working with you to deliver Power Query in Excel 2016.

To Yigal Edery, special thanks for accepting me into the ranks of the Microsoft Excel team and for challenging me to do more. I will never forget the night you pulled me over on the side of the road to share feedback and thank me.

Rob Collie, I wouldn't be here without you. You had welcomed me to PowerPivotPro.com as a guest blogger and a principal consultant, and you helped me make the leap into a brave new world.

Marco Russo, Ken Puls, Chris Webb, Matt Allington, and Reza Rad—My fellow Microsoft MVPs and Power BI bloggers—you are my role models, and I thank you for the inspiration and vast knowledge.

Since I joined the Avanade Analytics team in early 2017, I have learned so much from all of you at Avanade. Special thanks to Neelesh Raheja for your mentorship and leadership. You have truly expanded my horizons in the sea of analytics.

Finally, to my parents. Although I now live 6,208 miles away, I want to thank you. Dad, you had taught me how to crunch numbers and use formulas in Excel many years ago. And, Mom, your artistic talent is influencing my Power BI visuals every day.

—Gil Raviv

Introduction

Did you know that there is a data transformation technology inside Microsoft Excel, Power BI, and other products that allows you to work miracles on your data, avoid repetitive manual work, and save up to 80% of your time?

- Every time you copy/paste similar data to your workbook and manually clean it, you are wasting precious time, possibly unaware of the alternative way to do it better and faster.
- Every time you rely on others to get your data in the right shape and condition, you should know that there is an easier way to reshape your data once and enjoy an automation that works for you.
- Every time you need to make quick informed decisions but confront massive data cleansing challenges, know you can now easily address these challenges and gain unprecedented potential to reduce the time to insight.

Are you ready for the change? You are about to replace the maddening frustration of the repetitive manual data cleansing effort with sheer excitement and fun, and throughout this process, you may even improve your data quality and tap in to new insights.

Excel, Power BI, Analysis Services, and PowerApps share a game-changing data connectivity and transformation technology, Power Query, that empowers any person with basic Excel skills to perform and automate data importing, reshaping, and cleansing. With simple UI clicks and a unified user experience across wide variety of data sources and formats, you can resolve any data preparation challenge and become a master data wrangler.

In this book, you will tackle real data challenges and learn how to resolve them with Power Query. With more than 70 challenges and 200 exercise files in the companion content, you will import messy and disjointed tables and work your way through the creation of automated and well-structured datasets that are ready for analysis. Most of the techniques are simple to follow and can be easily reused in your own business context.

Who this book is for

This book was written to empower business users and report authors in Microsoft Excel and Power BI. The book is also relevant for SQL Server or Azure Analysis Services developers who wish to speed up their ETL development. Users who create apps using Microsoft PowerApps can also take advantage of this book to integrate complex datasets into their business logic.

Whether you are in charge of repetitive data preparation tasks in Excel or you develop Power BI reports for your corporation, this book is for you. Analysts, business intelligence specialists, and ETL developers can boost their productivity by learning the techniques in this book. As Power Query technology has become the primary data stack in Excel, and as Power BI adoption has been tremendously accelerating, this book will help you pave the way in your company and make a bigger impact.

The book was written to empower all Power Query users. Whether you are a new, moderate, or advanced user, you will find useful techniques that will help you move to the next level.

Assumptions

Prior knowledge of Excel or Power BI is expected. While any Excel user can benefit from this book, you would gain much more from it if you meet one of the following criteria. (Note that meeting a single criterion is sufficient.)

- You frequently copy and paste data into Excel from the same sources and often need to clean that data
- You build reports in Excel or Power BI that are connected to external sources, and wish to improve them
- You are familiar with PivotTables in Excel
- You are familiar with Power Pivot in Excel and wish to simplify your data models
- You are familiar with Power Query and want to move to the next level
- You develop business applications using PowerApps and need to connect to data sources with messy datasets
- You are a developer in Analysis Services and wish to speed up your ETL development

How this book is organized

The book is organized into 14 chapters that start from generic and simpler data challenges and move on to advanced and specific scenarios to master. It is packed with hands-on exercises and step-by-step solutions that provide the necessary techniques for mastering real-life data preparation challenges and serve as a long-term learning resource, no matter how many new features will be released in Power Query in the future.

In Chapter 1, “Introduction to Power Query,” you will be introduced to Power Query and gain the baseline knowledge to start the exercises that follow.

In Chapter 2, "Basic Data Preparation Challenges," you will learn how to tackle relatively basic common data challenges. If you carry out frequent data cleansing tasks at work, you will find this chapter extremely helpful. You will be introduced to the simplest techniques to automate your data cleansing duties, with simple mouse clicks and no software development skills. If you are new to Power Query, you will already start saving time by following the techniques in this chapter.

In Chapter 3, "Combining Data from Multiple Sources," you will learn how to combine disjointed datasets and append multiple tables in the Power Query Editor. You will learn how to append together multiple workbooks from a folder and combine multiple worksheets in a robust manner—so when new worksheets are added, a single refresh of the report will suffice to append the new data into your report.

In Chapter 4, "Combining Mismatched Tables," you will move to the next level and learn how to combine mismatched tables. In real-life scenarios your data is segmented and siloed, and often is not consistent in its format and structure. Learning how to normalize mismatched tables will enable you to gain new insights in strategic business scenarios.

In Chapter 5, "Preserving Context," you will learn how to extract and preserve external context in your tables and combine titles and other meta information, such as filenames and worksheet names, to enrich your appended tables.

In Chapter 6, "Unpivoting Tables," you will learn how to improve your table structure to utilize a better representation of the entities that the data represents. You will learn how the Unpivot transformation is a cornerstone in addressing badly designed tables, and harness the power of Unpivot to restructure your tables for better analysis. You will also learn how to address nested tables and why and how to ignore totals and subtotals from your source data.

In Chapter 7, "Advanced Unpivoting and Pivoting of Tables," you will continue the journey in Unpivot transformations and generalize a solution that will help you unpivot any summarized table, no matter how many levels of hierarchies you might have as rows and columns. Then, you will learn how to apply Pivot to handle multiline records. The techniques you learn in this chapter will enable you to perform a wide range of transformations and reshape overly structured datasets into a powerful and agile analytics platform.

As a report author, you will often share your reports with other authors in your team or company. In Chapter 8, "Addressing Collaboration Challenges," you will learn about basic collaboration challenges and how to resolve them using parameters and templates.

In Chapter 9, "Introduction to the Power Query M Formula Language," you will embark in a deep dive into M, the query language that can be used to customize your queries to achieve more, and reuse your transformation on a larger scale of challenges. In this chapter, you will learn the main building blocks of M—its syntax, operators, types, and a wide variety of built-in functions. If you are not an advanced user, you can skip this chapter and return later in your journey. Mastering M is not a prerequisite to becoming a master data wrangler, but the ability to modify the M formulas when needed can boost your powers significantly.

The user experience of the Power Query Editor in Excel and Power BI is extremely rewarding because it can turn your mundane, yet crucial, data preparation tasks into an automated refresh flow. Unfortunately, as you progress on your journey to master data wrangling, there are common mistakes you might be prone to making in the Power Query Editor, which will lead to the creation of vulnerable queries that will fail to refresh, or lead to incorrect results when the data changes. In Chapter 10, "From Pitfalls to Robust Queries," you will learn the common mistakes, or pitfalls, and how to avoid them by building robust queries that will not fail to refresh and will not lead to incorrect results.

In Chapter 11, "Basic Text Analytics," you will harness Power Query to gain fundamental insights into textual feeds. Many tables in your reports may already contain abundant textual columns that are often ignored in the analysis. You will learn how to apply common transformations to extract meaning from words, detect keywords, ignore common words (also known as stop words), and use Cartesian Product to apply complex text searches.

In Chapter 12, "Advanced Text Analytics: Extracting Meaning," you will progress from basic to advanced text analytics and learn how to apply language translation, sentiment analysis, and key phrase detection using Microsoft Cognitive Services. Using Power Query Web connector and a few basic M functions, you will be able to truly extract meaning from text and harness the power of artificial intelligence, without the help of data scientists or software developers.

In Chapter 13, "Social Network Analytics," you will learn how to analyze social network data and find how easy it is to connect to Facebook and gain insights into social activity and audience engagement on any brand, company, or product on Facebook. This exercise will also enable you to work on unstructured JSON datasets and practice Power Query on public datasets.

Finally, in Chapter 14, "Final Project: Combining It All Together," you will face the final challenge of the book and put all your knowledge to the test applying your new data-wrangling powers on a large-scale challenge. Apply the techniques from this book to combine dozens of worksheets from multiple workbooks, unpivot and pivot the data, and save Wide World Importers from a large-scale cyber-attack!

About the companion content

We have included this companion content to enrich your learning experience. You can download this book's companion content by following these instructions:

1. Register your book by going to www.microsoftpressstore.com and logging in or creating a new account.

2. On the Register a Product page, enter this book's ISBN (9781509307951), and click Submit.

3. Answer the challenge question as proof of book ownership.
4. On the Registered Products tab of your account page, click on the Access Bonus Content link to go to the page where your downloadable content is available.

The companion content includes the following:

- Excel workbooks and CSV files that will be used as messy and badly formatted data sources for all the exercises in the book. No need to install any external database to complete the exercises.
- Solution workbooks and Power BI reports that include the necessary queries to resolve each of the data challenges.

The following table lists the practice files that are required to perform the exercises in this book.

Chapter	File(s)
Chapter 1: Introduction to Power Query	C01E01.xlsx C01E01 - Solution.xlsx C01E01 - Solution.pbix
Chapter 2: Basic Data Preparation Challenges	C02E01.xlsx C02E01 - Solution.xlsx C02E02.xlsx C02E02 - Solution - Part 1.xlsx C02E02 - Solution - Part 2.xlsx C02E02 - Solution - Part 3.xlsx C02E02 - Solution - Part 1.pbix C02E02 - Solution - Part 2.pbix C02E02 - Solution - Part 3.pbix C02E03 - Solution.xlsx C02E03 - Solution - Part 2.xlsx C02E03 - Solution.pbix C02E03 - Solution - Part 2.pbix C02E04.xlsx C02E04 - Solution.xlsx C02E04 - Solution.pbix C02E05.xlsx C02E05 - Solution.xlsx C02E05 - Solution.pbix C02E06.xlsx C02E06 - Solution.xlsx C02E06 - Solution.pbix

Chapter	File(s)
	C02E07.xlsx C02E07 - Solution.xlsx C02E07 - Solution.pbix C02E08.xlsx C02E08 - Solution.xlsx C02E08 - Solution.pbix
Chapter 3: Combining Data from Multiple Sources	C03E01 - Accessories.xlsx C03E01 - Bikes.xlsx C03E01 - Clothing.xlsx C03E01 - Components.xlsx C03E03 - Products.zip C03E03 - Solution.xlsx C03E03 - Solution.pbix C03E04 - Year per Worksheet.xlsx C03E04 - Solution 01.xlsx C03E04 - Solution 02.xlsx C03E04 - Solution 01.pbix C03E04 - Solution 02.pbix
Chapter 4: Combining Mismatched Tables	C04E01 - Accessories.xlsx C04E01 - Bikes.xlsx C04E02 - Products.zip C04E02 - Solution.xlsx C04E02 - Solution.pbix C04E03 - Products.zip C04E03 - Solution.xlsx C04E03 - Solution.pbix C04E04 - Products.zip C04E04 - Conversion Table.xlsx C04E04 - Solution - Transpose.xlsx C04E04 - Solution - Transpose.pbix C04E05 - Solution - Unpivot.xlsx C04E05 - Solution - Unpivot.pbix C04E06 - Solution - Transpose Headers.xlsx C04E06 - Solution - Transpose Headers.pbix C04E07 - Solution - M.xlsx C04E07 - Solution - M.pbix
Chapter 5: Preserving Context	C05E01 - Accessories.xlsx C05E01 - Bikes & Accessories.xlsx C05E01 - Bikes.xlsx C05E01 - Solution.xlsx C05E01 - Solution 2.xlsx C05E01 - Solution.pbix C05E01 - Solution 2.pbix

Chapter	File(s)
	C05E02 - Bikes.xlsx C05E02 - Solution.xlsx C05E02 - Solution.pbix C05E03 - Products.zip C05E03 - Solution.xlsx C05E03 - Solution.pbix C05E04 - Products.xlsx C05E04 - Solution.xlsx C05E04 - Solution.pbix C05E05 - Products.xlsx C05E05 - Solution.xlsx C05E05 - Solution.pbix C05E06 - Products.xlsx C05E06 - Jump Start.xlsx C05E06 - Jump Start.pbix C05E06 - Solution.xlsx C05E06 - Solution.pbix
Chapter 6: Unpivoting Tables	C06E01.xlsx C06E02.xlsx C06E03.xlsx C06E03 - Wrong Solution.pbix C06E03 - Solution.xlsx C06E03 - Solution.pbix C06E04.xlsx C06E04 - Solution.xlsx C06E04 - Solution.pbix C06E05.xlsx C06E05 - Solution.xlsx C06E05 - Solution.pbix C06E06.xlsx C06E06 - Solution.xlsx C06E06 - Solution.pbix
Chapter 7: Advanced Unpivoting and Pivoting of Tables	C07E01.xlsx C07E01 - Solution.xlsx C07E01 - Solution.pbix C07E02.xlsx C07E02.pbix C07E03 - Solution.xlsx C07E03 - Solution.pbix C07E04.xlsx C07E04 - Solution.xlsx C07E04 - Solution.pbix C07E05 - Solution.xlsx C07E05 - Solution.pbix

Chapter	File(s)
Chapter 8: Addressing Collaboration Challenges	C08E01.xlsx C08E01 - Alice.xlsx C08E01 - Alice.pbix C08E01 - Solution.xlsx C08E01 - Solution.pbix C08E02 - Solution.pbix C08E02 - Solution.pbit C08E02 - Solution 2.pbit C08E03 - Solution.xlsx C08E03 - Solution 2.xlsx C08E04 - Solution.xlsx C08E04 - Solution.pbix C08E05.xlsx C08E05.pbix C08E05 - Folder.zip C08E05 - Solution.xlsx C08E05 - Solution.pbix
Chapter 9: Introduction to the Power Query M Formula Language	C09E01 – Solution.xlsx C09E01 – Solution.pbix
Chapter 10: From Pitfalls to Robust Queries	C10E01.xlsx C10E01 - Solution.xlsx C10E02 - Solution.xlsx C10E02 - Solution.pbix C10E03 - Solution.xlsx C10E03 - Solution.pbix C10E04 - Solution.xlsx C10E04 - Solution.pbix C10E05.xlsx C10E05 - Solution.xlsx C10E05 - Solution.pbix C10E06.xlsx C10E06 - Solution.xlsx C10E06 - Solution.pbix C10E06-v2.xlsx
Chapter 11: Basic Text Analytics	Keywords.txt Stop Words.txt C11E01.xlsx C11E01 - Solution.xlsx C11E01 - Solution.pbix C11E02 - Solution.xlsx C11E02 - Refresh Comparison.xlsx C11E02 - Solution.pbix

Chapter	File(s)
	C11E03 - Solution.xlsx C11E04 - Solution.xlsx C11E04 - Solution.pbix C11E05 - Solution.xlsx C11E05 - Solution.pbix C11E06 - Solution.xlsx C11E06 - Solution.pbix C11E07 - Solution.pbix
Chapter 12: Advanced Text Analytics: Extracting Meaning	C12E01 - Solution.xlsx C12E01 - Solution.pbix C12E02.xlsx C12E02 - Solution.xlsx C12E02 - Solution.pbix C12E02 - Solution.pbit C12E03 - Solution.xlsx C12E03 - Solution.pbix C12E04.xlsx C12E04.pbix C12E04 - Solution.xlsx C12E04 - Solution.pbix C12E05 - Solution.pbix C12E06 - Solution.xlsx C12E06 - Solution.pbix
Chapter 13: Social Network Analytics	C13E01 - Solution.xlsx C13E01 - Solution.pbit C13E02 - Solution.xlsx C13E02 - Solution.pbit C13E03 - Solution.xltx C13E03 - Solution.pbit C13E04 - Solution.xlsx C13E04 - Solution.pbix C13E05 - Solution.xlsx C13E05 - Solution.pbix C13E06 - Solution.xlsx C13E06 - Solution.pbix
Chapter 14: Final Project: Combining It All Together	C14E01 - Goal.xlsx C14E01.zip C14E01 - Solution.xlsx C14E01 - Solution.pbix C14E02 - Compromised.xlsx C14E02 - Solution.xlsx C14E02 - Solution.pbix

System requirements

You need the following software and hardware to build and run the code samples for this book:

- Operating System: Windows 10, Windows 8, Windows 7, Windows Server 2008 R2, or Windows Server 2012
- Software: Office 365, Excel 2016 or later versions of Excel, Power BI Desktop, Excel 2013 with Power Query Add-In, or Excel 2010 with Power Query Add-In

This page intentionally left blank

CHAPTER 2

Basic Data Preparation Challenges

Before anything else, preparation is the key to success.

—Alexander Graham Bell

IN THIS CHAPTER, YOU WILL

- Learn how to split and extract valuable information from delimiter-separated values
- Learn how to enrich your data from lookup tables in a single table or by using relationships between fact tables and lookup tables
- Learn how Add Column from Examples can help you extract or compute meaningful data from columns and use it to explore new transformations
- Learn how to extract meaningful information, such as hyperlinks, from text columns
- Handle inconsistent date values from one or more locales
- Learn how to extract date or time elements from *Date* columns
- Split a table into a fact table and a lookup table by using the Reference and Remove Duplicates commands
- Learn how to avoid refresh failures when a lookup table contains duplicate values, even when you are sure you removed them
- Split delimiter-separated values into rows to define group/member associations

Working with messy datasets can consume a lot of time for a data analyst. In the past, analysts needed to tackle badly formatted data with painful manual cleansing efforts. Excel power users could also use advanced formulas and VBA to clean and prepare data. Those who knew programming languages such as Python and R could harness their power to aid in the cleansing effort. But in many cases, data analysts eventually abandoned their exploration of messy datasets, uncertain of the return on investment they would get from cleaning the badly formatted data.

In this chapter, you will learn a wide variety of techniques in the Power Query Editor in Excel and Power BI for cleaning badly formatted datasets. These techniques will enable you to prepare your data in a matter of minutes. If you are new to Power Query, this chapter is the key to achieving a lot as a data analyst. The simple techniques that you will learn here will significantly reduce your data preparation time and enable you to tackle bigger challenges more quickly.

You may need to prepare messy data for an ad hoc analysis, or you might need to periodically prepare and circulate reports for large audiences. This chapter introduces the most common and basic data preparation techniques that will save your time and enable you to automate and streamline your efforts to gain insights.

Extracting Meaning from Encoded Columns

One of the most common challenges in unprepared data is to extract meaning from columns that are badly formatted. If you have a sequence of delimiter-separated codes or values in a single column, you might want to split them into multiple columns.

Excel provides native formula functions for extracting values from badly formatted text. You can use a combination of the functions *LEFT*, *MID*, *RIGHT*, *FIND*, *SEARCH*, and *LEN* to extract any substring from text. But often, the logic to extract the information you seek requires complex formulas and is hard to maintain over time, requiring you to expand the formulas to new rows when your source data changes.

The Text to Columns wizard in the Data tab of Excel enables you to easily split a column into multiple columns. Unfortunately, if you need to load a new table and apply the same type of split, you need to repeat the steps or use macros and VBA. In this section you will learn how you can use Power Query to tackle this challenge. You will find that the Power Query Editor in Excel and Power BI is extremely helpful, especially if you need to repeat the same tasks often, work with large datasets, or maintain a report over the long term.

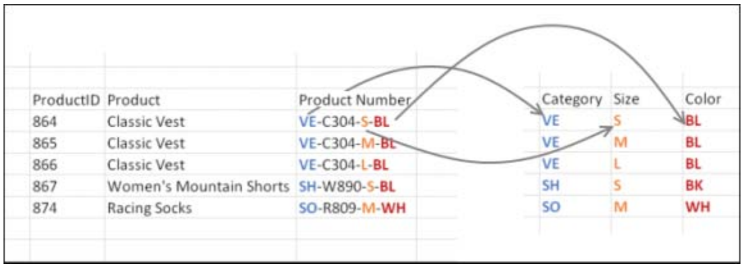
AdventureWorks Challenge

In this chapter, you will work with the product catalog of a fictitious bike manufacturer company, AdventureWorks Cycles, in the *AdventureWorks* database.

See Also AdventureWorks Cycles is a fictitious company invented by Microsoft to demonstrate the implementation of Microsoft applications in close-to-real-life business scenarios. The AdventureWorks database can be deployed on an Azure SQL Database server for learning purposes. You can create an Azure SQL Database server that contains AdventureWorks data by following the steps in the article https://docs.microsoft.com/en-us/azure/sql-database/sql-database-get-started-portal. You are not required to do this to follow the exercises in this book, but doing so will give you more data samples to practice Power Query.

Imagine that you are a new analyst in the fictitious company AdventureWorks. Your first task is to analyze the company's product catalog and find out how many products the company manufactures by category, product size, and color. The list of products is provided in an Excel workbook that is manually exported from a legacy data warehouse. You can download the workbook C02E01.xlsx from https://aka.ms/DataPwrBIPivot/downloads.

Unfortunately, the product catalog is missing the Category, Size, and Color columns. Instead, it contains a Product Number, containing dash-separated values that include the information on category, size, and color. Figure 2-1 shows the mapping between the Product Number column and the missing columns.



ProductID	Product	Product Number	Category	Size	Color
864	Classic Vest	VE-C304-S-BL	VE	S	BL
865	Classic Vest	VE-C304-M-BL	VE	M	BL
866	Classic Vest	VE-C304-L-BL	VE	L	BL
867	Women's Mountain Shorts	SH-W890-S-BL	SH	S	BK
874	Racing Socks	SO-R809-M-WH	SO	M	WH

FIGURE 2-1 The AdventureWorks product catalog has category, size, and color values encoded inside the product number.

Your predecessor, who was recently promoted and now leads the Business Intelligence team, had solved this challenge by using Excel formulas. In the next exercise, you will learn how he implemented the solution.

Exercise 2-1: The Old Way: Using Excel Formulas

Download the workbook C02E01 - Solution.xlsx from https://aka.ms/DataPwrBIPivot/downloads. This workbook, which was implemented by your predecessor, uses Excel formulas to extract the product category, color, and size. In this exercise, you can follow his notes and learn how he implemented the solution—without using Power Query.

Note Your goal in Exercise 2-1 is to review a typical solution in Excel, without using Power Query. You are not required to implement it or even fully understand it. As you will soon learn, the Power Query solution is simpler.

1. Open the workbook C02E01 - Solution.xlsx in Excel.
2. Review the first three worksheets: Products, Categories, and Colors. Products contains the main catalog table. In the Categories worksheet, you can see the mapping between category codes and values. For example, *VE* stands for *vests*, *SH* stands for *shorts*, *WB* stands for *bottles*, and *BC* stands for *cages*. In the Colors worksheet, you can see the mapping between color codes and values. For example, *BE* stands for *blue*, and *BK* stands for *black*.

   The fourth worksheet contains three PivotTables with the kind of analysis you would expect to do. All the PivotTables are fed from the data in the Products worksheet.
3. In the Products worksheet, select the cell F2. In the formula bar, notice that this cell contains the following formula:

   ```
   =LEFT(C2, 2)
   ```

The formula returns the two leftmost characters from column C, which contains the product number. These two characters represent the category code.

4. Select cell G2. In the formula bar, you can see the following formula:

   ```
   =RIGHT(C2, 2)
   ```

   The formula returns the two rightmost characters from column C, which contains the product number. These two characters represent the color code.

5. Select cell H2. In the formula bar you can see the following formula:

   ```
   =SUBSTITUTE(MID(C2, SEARCH("-", C2, 8), 3), "-", "")
   ```

   This formula returns the code that represents the product size. This is where things become difficult. The inner formula searches for the character "-" in the product number at cell C2, starting from character 8. It returns the location of the character. Then, the *MID* function returns a three-digit substring. The substring may contain a leading or trailing dash character. The *SUBSTITUTE* function trims away the dash character.

6. Select cell I2. In the formula bar you can see the following formula:

   ```
   =VLOOKUP(F2, Categories!A:B, 2, FALSE)
   ```

   This formula returns the category value from the Categories worksheet, whose product code matches the value in F2.

7. Select cell J2. In the formula bar you can see the following formula:

   ```
   =VLOOKUP(G2, Colors!A:B, 2, FALSE)
   ```

   This formula returns the color value from the Colors worksheet, whose color code matches the value in G2.

If you are an Excel power user, it will not be difficult for you to implement the same formulas just described on a new dataset, such as in C02E01.xlsx. If you are not a power user, you can copy and paste the dataset from C02E01.xlsx to the relevant columns in the Products worksheet of C02E01 - Old Solution.xlsx, and then copy down all the formulas in columns F to J.

But what do you do if you need to update this workbook on a weekly basis? As you get constant product updates, you will need to perform a repetitive sequence of copies and pastes, which may lead to human errors and incorrect calculations. In the next two exercises, you will learn a better way to achieve your goals—by using Power Query.

Exercise 2-2, Part 1: The New Way

In this exercise, you will use the Power Query Editor to create a single Products table. Maintaining this new solution will be much easier than with C02E01 - Old Solution. A single refresh of the workbook will suffice, and you will no longer have many formulas in the spreadsheet to control and protect.

You can follow this exercise in Excel, Power BI Desktop, or any other product that includes the Power Query Editor.

Download the workbook C02E02.xlsx from https://aka.ms/DataPwrBIPivot/downloads and save it in a new folder: *C:\Data\C02*. The workbook contains the *AdventureWorks* database Products, Categories, and Colors worksheets. You will start the exercise by importing the three tables to the Power Query Editor.

1. Start a new blank Excel workbook or a new Power BI Desktop report.
2. Follow these steps to import C02E02.xlsx to the Power Query Editor:
   a. **In Excel:** In the Data tab, select Get Data, From File, From Workbook.

> **Note** Due to variations in the ribbons in the older versions of Excel, the instructions throughout the book from now on will assume you use Excel 2016 or later. (As a result, all entry points will be through the Get Data drop-down in the Data tab.) You can still follow these steps in the Power Query add-in for Excel 2010 and 2013 as well—the differences will be minuscule, and will only be applicable to the initial steps in the Data/Power Query ribbons. The ribbon differences among the Power Query versions is described in Chapter 1, "Introduction to Power Query," in the section, "Where Can I Find Power Query?"

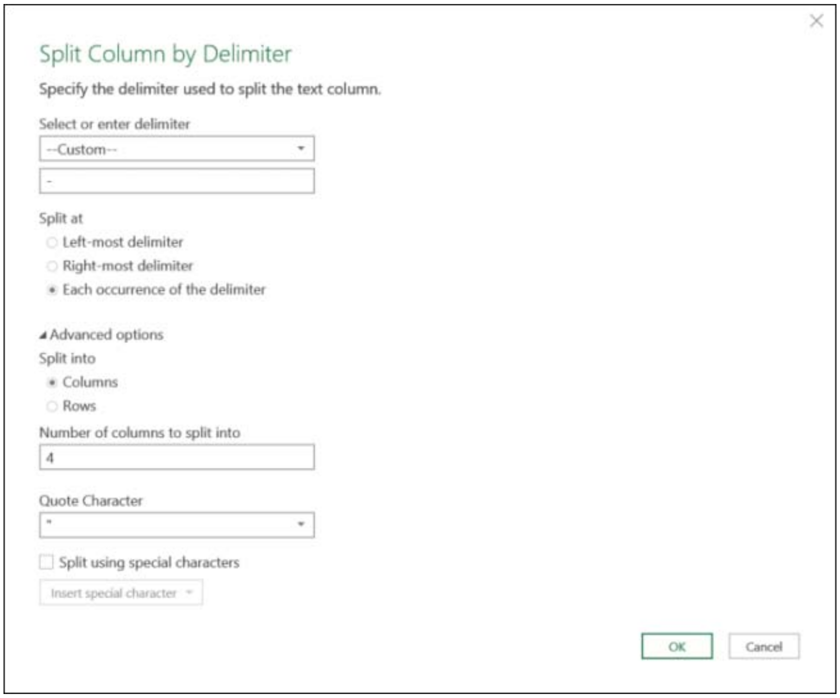
   In Power BI Desktop: In the Get Data drop-down menu, select Excel.

   b. Select the file C02E02.xlsx and select Import.
   c. In the Navigator dialog box that opens in Excel, select the Select Multiple Items check box. In either Excel or Power BI Desktop, select the worksheets Categories, Colors, and Products and select Edit. (As explained in Chapter 1, If you don't find the Edit button in the Navigator dialog box, select Transform Data, or the second button that is right to Load.)
3. In the Power Query Editor that opens, to the left you can see the Queries pane, which contains three queries—one for each of the worksheets you selected in the Navigator in step 2. By selecting each of the queries, you can see a preview of the data in the Preview pane. To the right, you can see the Query Settings pane and the Applied Steps pane, which shows you the sequence of transformation steps as you make changes to your query. Follow these steps to prepare the Categories and Colors queries:
   a. In the Queries pane, select the Categories query. In the Preview pane, you will notice that the headers are Column1 and Column2. The first row contains the column names: Category Code and Product Category Name. To promote the first row, on the Transform tab, select Use First Row As Headers.

> **Tip** If you prefer working with shortcut menus, you can click on the table icon in the top-left corner of the table in the Preview pane of the Power Query Editor and select Use First Row As Headers.

   b. In the Queries pane, select the Colors query, which requires the same preparation step as the Categories query. Its column names are Column1 and Column2. The first row contains the actual column names, Color Code and Color. To promote the first row, on Transform tab, select Use First Row As Headers.

4. In the Queries pane, select the Products query.
5. Select the Product Code column either by clicking its header or by selecting the Choose Columns drop-down menu on the Home tab and then selecting Go to Column. When the Go to Column dialog box opens, select the Product Code column in the list and click OK to close the dialog box.
6. On the Transform tab, select Split Column and then select By Delimiter. You can also right-click the header of the Product Code column and select Split Column from the shortcut menu, and then select By Delimiter.
7. In the Split Column by Delimiter dialog box that opens, by default, the correct settings are selected, and you could simply click OK to close the dialog box, but before you do, review the elements of the dialog (see Figure 2-2):


Split Column by Delimiter

Specify the delimiter used to split the text column.

Select or enter delimiter: --Custom--

-

Split at
- Left-most delimiter
- Right-most delimiter
- [✓ Each occurrence of the delimiter]

Advanced options

Split into
- [✓ Columns]
- Rows

Number of columns to split into: 4

Quote Character: "

Split using special characters

FIGURE 2-2 The Split Column by Delimiter dialog box is automatically set to split by the dash delimiter. More options can be found under Advanced Options. It is recommended that you review them and ensure that the default choices are correct.

- The Custom option is selected in the dialog box, and the dash character is specified as a delimiter. This selection is recognized by Power Query because all the values in the columns are dash-separated.
- Each Occurrence of the Delimiter is selected because there are multiple dash characters.
- If you expand the Advanced Options section, you can see that Power Query detected that Product Number values can be split into four columns.

Tip When you have a fixed number of delimiters, the split into columns is effective. When you have an unknown number of delimiters, you should consider splitting the column into rows. Later in this exercise, you will learn when splitting into rows can be helpful.

- You can see that the quote character is also set. Why do you need it? How will you split dash-separated values if one of the values contains a dash that should not be used as a delimiter? The owner of your data can use double quotes at the beginning and end of text that contains a dash. Power Query will not split the dash characters inside the quoted string. The only caveat is that when you have a double quote at the beginning of a value, but you are missing an ending double quote, the entire text starting from the double quote will be kept intact. To ignore quotes, select None as the quote character.

8. After you close the Split Column by Delimiter dialog box, you can see in the Preview pane that the Product Number column is split into four columns: Product Number.1, Product Number.2, Product Number.3, and Product Number.4. Rename them *Category Code, Short Product Number, Size,* and *Color*. To rename a column, double-click the column name and type a new name. Alternatively, you can select the column and then select Rename in the Transform tab. The column name is then selected, and you can enter the new column name.

   At this stage, you have extracted the size column, and you have separate codes for the product category and color. You will now learn two approaches for adding the actual category and color values instead of their codes. In Exercise 2-2, Part 2, you will learn how to merge the category and color values into the Products table, and in Exercise 2-2, Part 3, you will learn how to keep the three tables as fact and lookup tables.

9. To save the workbook or Power BI report follow these steps:

   In Excel:

   a. In the Home tab of the Power Query Editor, select the Close & Load drop-down menu and then select Close & Load To. The Import Data dialog box opens.
   b. Select Only Create Connection and ensure that Add This Data to the Data Model is unselected. (In the third part of this exercise, you will select this check box.)
   c. Save the workbook as C02E02 - Solution - Part 1.xlsx.

   In Power BI Desktop: Select Close & Apply and save the report as C02E02 - Solution - Part 1.pbix.

Exercise 2-2, Part 2: Merging Lookup Tables

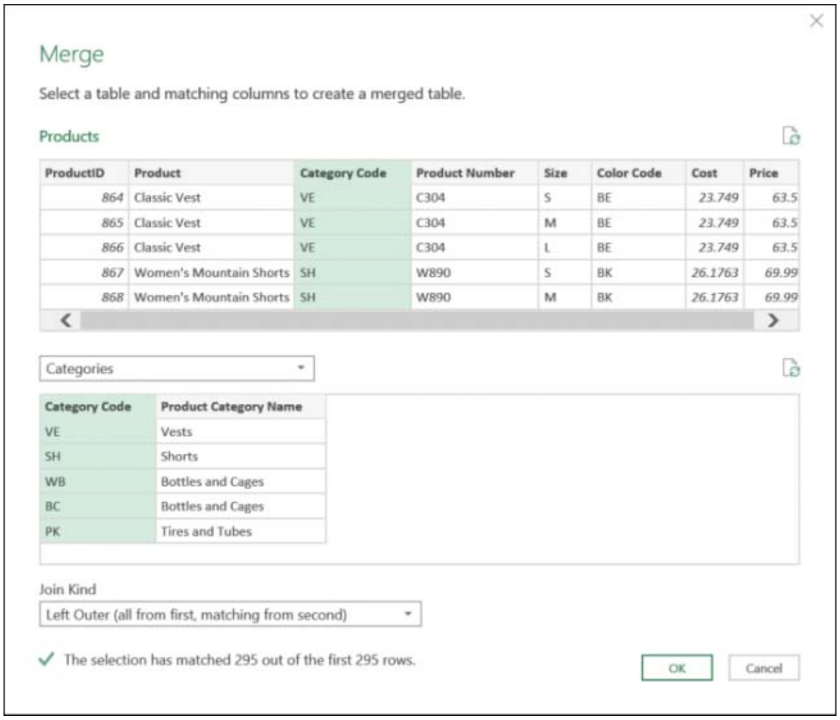
In this part of Exercise 2-2, you will merge the category and color values from the Categories and Colors queries into the Products query, according to their corresponding codes. Recall from Exercise 2-1 that mapping between the codes and the values can be done using *VLOOKUP* in Excel. In this exercise, you will learn an equivalent method using the Power Query Editor.

1. **In Excel:** Open the file C02E02 - Solution - Part 1.xlsx. In the Data tab, select Get Data and then select Launch Power Query Editor.

   In Power BI Desktop: Open the file C02E02 - Solution - Part 1.pbix and select Edit Queries from the Home tab.

2. In the Power Query Editor that opens, in the Queries pane, select the Products query. On the Home tab, select Merge Queries.

3. In the Merge dialog box that opens, you can merge between two tables according to matching values in specified columns. You will merge the category names from the Categories query into the Products query, according to the matching category code. Figure 2-3 illustrates the following steps you need to take in the Merge dialog box:



Merge

Select a table and matching columns to create a merged table.

Products

ProductID	Product	Category Code	Product Number	Size	Color Code	Cost	Price
864	Classic Vest	VE	C304	S	BE	23.749	63.5
865	Classic Vest	VE	C304	M	BE	23.749	63.5
866	Classic Vest	VE	C304	L	BE	23.749	63.5
867	Women's Mountain Shorts	SH	W890	S	BK	26.1763	69.99
868	Women's Mountain Shorts	SH	W890	M	BK	26.1763	69.99

Categories

Category Code	Product Category Name
VE	Vests
SH	Shorts
WB	Bottles and Cages
BC	Bottles and Cages
PK	Tires and Tubes

Join Kind

Left Outer (all from first, matching from second)

✓ The selection has matched 295 out of the first 295 rows.

FIGURE 2-3 You can merge categories into products by category code in the Merge dialog box.

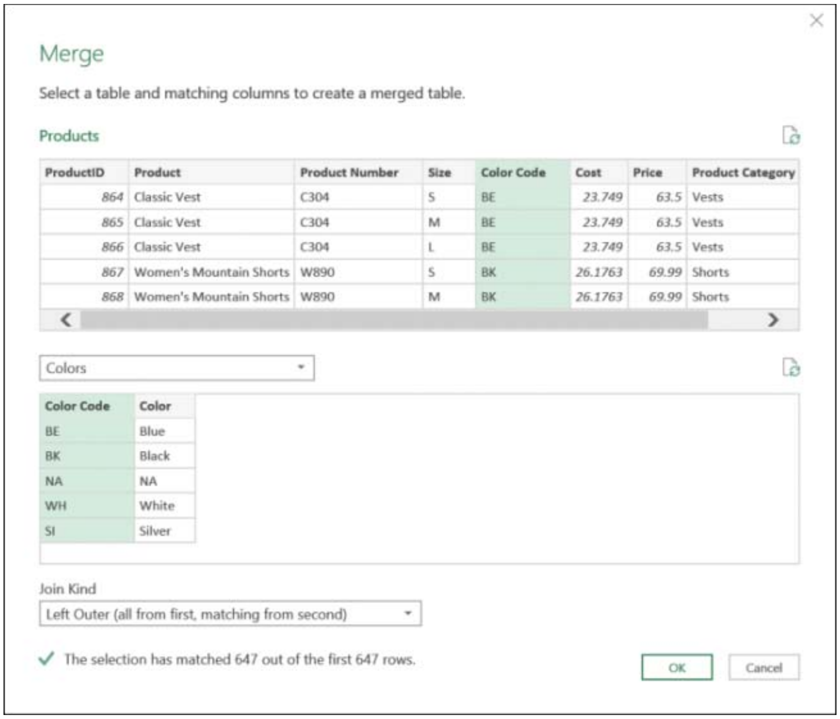
   a. Select the Category Code column in the Products table, and in the drop-down menu below the Products table, select Categories. Then select Category Code in the second table (refer to Figure 2-3).
   b. In the Join Kind drop-down, make sure Left Outer (All from First, Matching from Second) is selected, and click OK.

   In the Power Query Editor, you can see that a new Categories column is added as the last column, with Table objects as values.

4. Expand the Categories column (by clicking on the control at the right side of its header or by selecting the Categories column, and then selecting Expand in the Transform tab).
5. In the Expand pane, deselect Category Code and deselect Use Original Column Name As Prefix. Then click OK. The new Categories column is transformed to a new column called Product Category Name, with the matching category values in each row.
6. Now that you have the actual category name, remove the Category Code column by selecting it and pressing the Delete key.

Note Step 6 may seem a bit awkward for new users, who are accustomed to Excel formulas. You may be surprised to find that Product Category Name remains intact after the removal of Category Code. But keep in mind that the Power Query Editor does not work like Excel. In step 3, you relied on Category Code, as you merged categories into products by using the matching values in the Category Code column. However, in step 6 you removed the Category Code column. Whereas a spreadsheet can be perceived as a single layer, Power Query exposes you to a multi-layered flow of transformations, or states. Each state can be perceived as a transitory layer or a table, whose sole purpose is to serve as the groundwork for the next layer.

7. To merge the colors into the Products query, on the Home tab, select Merge Queries again. When the Merge dialog box opens, follow these steps (see Figure 2-4):
   a. Select the Color Code column in the Products table, and in the drop-down menu below the Products table, select Colors. Then select the Color Code column in the second table.
   b. In the Join Kind drop-down, ensure that Left Outer (All from First, Matching from Second) is selected and click OK.



Merge

Select a table and matching columns to create a merged table.

Products

ProductID	Product	Product Number	Size	Color Code	Cost	Price	Product Category
864	Classic Vest	C304	S	BE	23.749	63.5	Vests
865	Classic Vest	C304	M	BE	23.749	63.5	Vests
866	Classic Vest	C304	L	BE	23.749	63.5	Vests
867	Women's Mountain Shorts	W890	S	BK	26.1763	69.99	Shorts
868	Women's Mountain Shorts	W890	M	BK	26.1763	69.99	Shorts

Colors

Color Code	Color
BE	Blue
BK	Black
NA	NA
WH	White
SI	Silver

Join Kind

Left Outer (all from first, matching from second)

✓ The selection has matched 647 out of the first 647 rows.

FIGURE 2-4 You can merge colors into products by color code in the Merge dialog box.

In the Power Query Editor, a new Colors column is added as the last column, with Table objects as values. Let's pause for a moment and learn about the Table object and how to interact with it. In addition to the expand control located on the right side of the column, you have three common interactions with the Table objects:

- You can drill down to a single Table object by clicking on any of the table hyperlinks. When you do so, a new step is added to Applied Steps, and from here you could potentially continue preparing your data, which is based on the selected Table object. Try it now.

  Select any of the Table objects in the Colors column. The table in the Preview pane is transformed into a single-row table with the specific color code and color value that was merged in the cell that you selected. Delete this step from Applied Steps so you can get back to the Products table.

- You can also drill down to a single Table object as a new query. This can allow you to explore the data of a single table, while keeping the original table intact. To drill down to a single Table object as a new query, right-click any of the Table objects and select Add As New Query. A new query is created, with all the transformation steps in Products, including the new drill-down step, which is equivalent to the drill-down in the preceding example. If you tried it, you can now delete the new query.

- For temporary inspection of a specified table object, you can click on the space of any cell that contains a Table object. At the bottom of the Preview pane, you see a preview of the table. Now that you've learned a bit more about the Table objects, let's expand the Colors column.

> **Note** You will find that working with other structured objects in the Power Query Editor—such as Record, List, and Binary objects—is similar: You can drill down to single object, add the object as a new query, or preview its content by clicking anywhere in the cell's space.

8. Expand the Colors column (by clicking on the control at the right side of its header or by selecting the Colors column and then selecting Expand in the Transform tab).
9. In the Expand pane, deselect Color Code and deselect Use Original Column Name As Prefix. Then click OK. The new Colors column is transformed to Color, with the matching color values in each row.
10. Now that you have the actual color values, remove the Color Code column. The Products query is ready, and it's time to load it to the report.
11. **In Excel:** Follow these steps to load Products into the Data Model:
    - **a.** In the Home tab of the Power Query Editor, select Close & Load.
    - **b.** In Excel, on the Data tab, select Queries and Connections to open the Queries pane. Right-click Products and select Load To.
    - **c.** In the Import Data dialog box that opens, select Add This Data to the Data Model and click OK.
    - **d.** You can now create a PivotTable that uses the Data Model and analyzes the distribution of colors or categories. On the Insert tab, select PivotTable. The Create PivotTable dialog box opens. Ensure that the option Use This Workbook's Data Model is selected and click OK to close the dialog box.

    In Power BI Desktop: Follow these steps:
    - **a.** In the Queries pane, right-click Categories and deselect the Enable Load check box.
    - **b.** Right-click the Colors query and deselect the Enable Load check box.

      The last two steps allow you to treat specific queries as stepping-stones for other queries. Because you have extracted the category and color values from Categories and Colors, you can keep Products as the single table in your report. Deselecting Enable Load ensures that these queries are not loaded to the Power BI report. These steps are equivalent in Excel to step 9b, in Exercise 2-2, Part 1, where you selected Only Create Connection in the Import Data dialog box.
    - **c.** On the Home tab, select Close & Apply.

The Products table is now loaded and ready, with categories, colors, and sizes. To review the solution, including an example of a PivotTable with the distribution of products by color, download the workbook C02E02 - Solution - Part 2.xlsx from https://aka.ms/DataPwrBIPivot/downloads. To review the Power BI solution, download the report C02E02 - Solution - Part 2.pbix.

Exercise 2-2, Part 3: Fact and Lookup Tables

In Part 2 of Exercise 2-2, you merged the category and color values from the Categories and Colors queries into the Products query. While this approach enables you to have a single table with the necessary information for your report, there is a better way to do it, using relationships.

Note When you import multiple tables, relationships between those tables may be necessary to accurately calculate results and display the correct information in your reports. The Excel Data Model (also known as Power Pivot) and Power BI Desktop make creating those relationships easy. To learn more about relationships in Power BI, see https://docs.microsoft.com/en-us/power-bi/desktop-create-and-manage-relationships.

In Part 3 of Exercise 2-2, you will start where you left off in Part 1 and load the three queries as separate tables in the Data Model.

1. **In Excel:** Open the file C02E02 - Solution - Part 1.xlsx. In the next steps you will load all the queries into the Data Model in Excel, which will enable you to create PivotTables and PivotCharts from multiple tables. On the Data tab, select Queries & Connections.
    a. In the Queries & Connections pane that opens, right-click the Categories query and select Load To.
    b. In the Import Data dialog box that opens, select Add This Data to the Data Model and close the dialog box.
    c. Right-click the Colors query and select Load To.
    d. In the Import Data dialog box that opens, select Add This Data to the Data Model and close the dialog box.
    e. Right-click the Products query and select Load To.
    f. In the Import Data dialog box that opens, select Add This Data to the Data Model and close the dialog box.

        At this stage, you have the three tables loaded to the Data Model. It's time to create the relationship between these tables.
    g. On the Data tab, select Manage Data Model. The Power Pivot for Excel window opens.
    h. On the Home tab, select the Diagram view.

    In Power BI Desktop: Open the file C02E02 - Solution - Part 1.pbix and select the Relationships view from the left-side menu.

2. Drag and drop Category Code from the Categories table to Category Code in the Products table.

3. Drag and drop Color Code from the Colors table to Color Code in the Products table.

   Your Data Model now consists of a fact table with the product list and two lookup tables with the supplementary information on categories and colors. As shown in Figure 2-5, there are one-to-many relationships between Category Code in the Categories table and Category Code in the Products table, as well as between Color Code in the Colors table and Color Code in the Products table.

See Also The topic of modeling is not the focus of this book. To learn more about Power Pivot, Data Models, and relationships in Excel and Power BI, see *Analyzing Data with Power BI and Power Pivot for Excel* by Alberto Ferrari and Marco Russo (Microsoft Press).

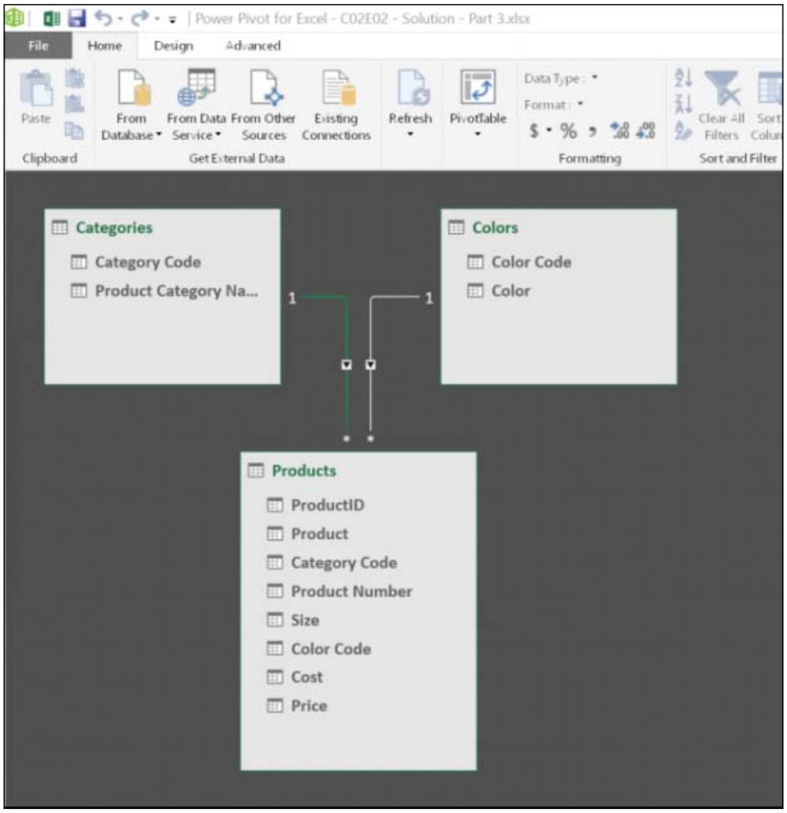


FIGURE 2-5 The Diagram view shows the Data Model relationships in Excel between Categories and Products and between Colors and Products.

4. After you have established the relationships between the tables, hide the columns Category Code and Color Code in the tables to ensure that these fields will no longer be selected in PivotTables, PivotCharts, and Power BI visualizations. By hiding these fields, you help report consumers enjoy a cleaner report. In other scenarios where you have multiple fact tables connected to a lookup table by the same column, hiding the column in the fact tables can lead to a more extendable report, in which a single column in the lookup table can be used in a slicer or a visual to filter data coming from the multiple fact tables.

   In Excel, right-click Color Code in the Colors table and select Hide from Client Tools. Repeat this step on Color Code in the Products table and Category Code in the Categories and Products tables.

   In Power BI, right-click Color Code in the Colors table and select Hide in Report View. Repeat this step on Color Code in the Products table and Category Code in the Categories and Products tables.

You can now create PivotTables, PivotCharts, and visualizations as demonstrated in the solution files C02E02 - Solution - Part 3.xlsx and C02E02 - Solution - Part 3.pbix, which are available at https://aka.ms/DataPwrBIPivot/downloads.

Tip While the solution in Exercise 2-2, Part 2 simplifies your report, as it leaves you with a single table, in real-life reports that have multiple tables, the solution in Exercise 2-2, Part 3 will provide more opportunities for insight, as your model will become more complex. For example, having a single Categories lookup table connected to multiple fact tables with Product Category codes will allow you to compare between numerical columns across the different fact tables, using the shared categories.

Using Column from Examples

Often, the first command you see on the left side of the ribbon can reveal the importance of a feature. That is why, for example, you can find PivotTable as the first command in the Insert tab of Excel. In the Power Query Editor, you can find one of the most important capabilities as the first command in Add Column tab. Column from Examples is a powerful feature that enables you to extract meaning from existing columns into a new column, without any preliminary knowledge of the different transformations available in the Power Query Editor.

By using Column from Examples, you can add new columns of data in the Power Query Editor by simply providing one or more sample values for your new column. When you provide these examples, Power Query tries to deduce the calculation needed to generate the values in the new column. This capability can be used as a shortcut to extract new meaning from data. This is a very powerful feature, especially for new users because it means you are not required to explore for the necessary transformation in the ribbons or memorize the M formulas to extract the meaningful data into the new column. If you simply provide a few examples in the new column, Power Query tries to do the work for you.

Exercise 2-3 provides a quick demonstration of Column from Examples with the dataset from Exercise 2-2.

Exercise 2-3, Part 1: Introducing Column from Examples

In Exercise 2-2, you learned how to split the product code into four elements. But imagine that you just need to extract the product size from the code. In this exercise you will learn how to do this by using Column from Examples.

If you didn't follow Exercise 2-2, download the workbook C02E02.xlsx from https://aka.ms/DataPwrBIPivot/downloads and save it in the folder *C:\Data\C02*.

1. Start a new blank Excel workbook or a new Power BI Desktop report.
2. Follow these steps to import C02E02.xlsx to the Power Query Editor:
    a. **In Excel:** On the Data tab, select Get Data, From File, From Workbook.
    In Power BI Desktop: In the Get Data drop-down menu, select Excel.
    b. Select the file C02E02.xlsx and select Import.
    c. In the Navigator dialog box that opens, select Products and then select Edit.
3. In the Power Query Editor, you can now see in the Preview pane that the Product Number column contains four dash-separated codes (for example, VE-C304-S-BE). The challenge is to extract the third value, which reflects the size of the product.

    Select the Product Number column, and on the Add Column tab, select the Column from Examples drop-down menu, where you see two options:

    - From All Columns
    - From Selection

    In this exercise, you need to extract the size of the product from Product Number, so select From Selection.

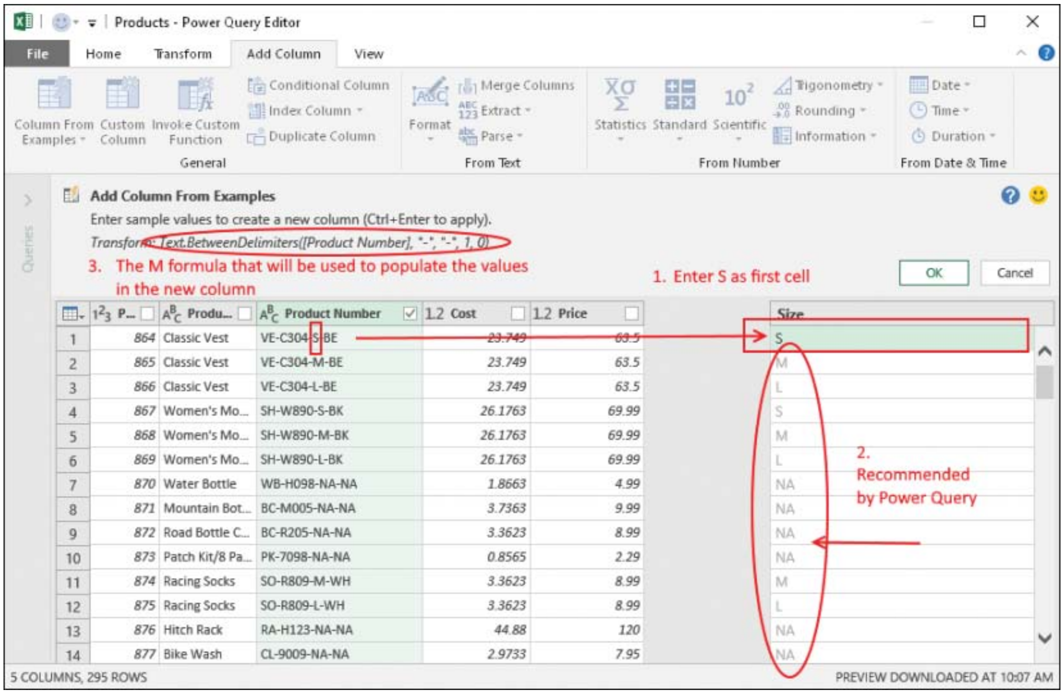
Tip In many cases, if you know in advance which columns should contribute to the new column, selecting the relevant input columns and then selecting From Selection in the Column from Examples drop-down menu will improve the chances that Power Query will provide a useful recommendation.

The Power Query Editor now enters a new state. The Preview pane is pushed down, and a new section is shown on top of the Preview pane, with the message Enter Sample Values to Create a New Column. In the bottom-right part of this new section are OK and Cancel buttons to exit from this special state.

On the right side of the Preview pane is an area with a new empty column. This is where you can enter your examples. Before you do so, rename Column1 to *Size* by double-clicking the header of the new column.

4. Double-click the first blank cell of the Size column in the Preview pane. A drop-down menu shows a few recommended examples that you can select from to create the new column. The values in the drop-down menu can provide some ideas of transformation that can be used to populate the new column.

5. In the first blank cell, enter *S*, which represents the bolded size value in the product number VE-C304-**S**-BE. Press Enter, and Power Query populates the new Size column with all the suggested values, as illustrated in Figure 2-6. Press Ctrl+Enter or click OK to create the column.



Enter sample values to create a new column (Ctrl+Enter to apply).

Transform: Text.BetweenDelimiters([Product Number], "-", "-", 1, 0)

	P...	Produ...	Product Number	Cost	Price	Size
1	864	Classic Vest	VE-C304-S-BE	23.749	63.5	S
2	865	Classic Vest	VE-C304-M-BE	23.749	63.5	M
3	866	Classic Vest	VE-C304-L-BE	23.749	63.5	L
4	867	Women's Mo...	SH-W890-S-BK	26.1763	69.99	S
5	868	Women's Mo...	SH-W890-M-BK	26.1763	69.99	M
6	869	Women's Mo...	SH-W890-L-BK	26.1763	69.99	L
7	870	Water Bottle	WB-H098-NA-NA	1.8663	4.99	NA
8	871	Mountain Bot...	BC-M005-NA-NA	3.7363	9.99	NA
9	872	Road Bottle C...	BC-R205-NA-NA	3.3623	8.99	NA
10	873	Patch Kit/8 Pa...	PK-7098-NA-NA	0.8565	2.29	NA
11	874	Racing Socks	SO-R809-M-WH	3.3623	8.99	M
12	875	Racing Socks	SO-R809-L-WH	3.3623	8.99	L
13	876	Hitch Rack	RA-H123-NA-NA	44.88	120	NA
14	877	Bike Wash	CL-9009-NA-NA	2.9733	7.95	NA

5 COLUMNS, 295 ROWS

PREVIEW DOWNLOADED AT 10:07 AM

FIGURE 2-6 The Power Query Editor in Add Column from Examples mode enables you to easily extract the size code from Product Number column.

6. You can now see the new Size column, calculated as expected. Load the data to your workbook or to a Power BI report and save.

You can download the solution files C02E03 - Solution.xlsx and C02E03 - Solution.pbix from https://aka.ms/DataPwrBIPivot/downloads.

Practical Use of Column from Examples

Column from Examples is a very useful tool to discover new transformations that are available in the user interface and in M. Let's demonstrate how a new transformation can be discovered, using the output of Column from Examples. In step 6 of Exercise 2-3, Part 1, you can see that in the Applied Steps pane of the Power Query Editor, the last step, Inserted Text Between Delimiters, contains a settings control (in the shape of a cog) at the right side of the step. By clicking the settings control you can open the Text Between Delimiters dialog box. This dialog box allows you to extract text between delimiters. Now, when you know that such a dialog box exists, you can explore the Power Query Editor to find which ribbon command can trigger it, outside the Column from Examples flow. Exploring the Transform tab reveals the command Text Between Delimiters inside the Extract drop-down menu.

For intermediate Power Query users, a great way to improve knowledge of M is to use Column from Examples and review the suggested code that is shown in the top pane (highlighted in Figure 2-6). You may find some useful functions, such as *Text.BetweenDelimiters*, which is used in Exercise 2-3, Part 1.

Column from Examples can help you achieve a wide variety of transformations on text, dates, times, and numbers. You can even use it to create bucketing/ranges and conditional columns, as you will see in Exercise 2-3, Part 2.

See Also To find the latest list of the functions supported by Column from Examples, go to https://docs.microsoft.com/en-us/power-bi/desktop-add-column-from-example.

Exercise 2-3, Part 2: Converting Size to Buckets/Ranges

In this part of Exercise 2-3, you will use Column from Examples to group numeric size values into buckets. You can download the solution files C02E03 - Solution.xlsx and C02E03 - Solution.pbix from https://aka.ms/DataPwrBIPivot/downloads to follow the steps of this exercise.

1. Open C02E03 - Solution.xlsx or C02E03 - Solution.pbix and launch the Power Query Editor. To launch the Power Query Editor in Excel, go to the Data tab, select Get Data, and then select Launch Power Query Editor. To launch the Power Query Editor in Power BI Desktop, select Edit Queries on the Home tab.

   You can see that the Size column contains a combination of alphabetic and numeric size values. In the next two steps you will learn how to ignore the textual values and focus only on numeric values in the Size column by creating a new column with the numeric representation for sizes. There are several ways to achieve this goal. In step 2, you will do it using the error-handling features in Power Query. In step 3, you will do it using Column from Examples.

2. To extract all the numeric sizes from the Size column by using error handling, follow these steps:

   a. Select the Size column. On the Add Column tab, select Duplicate Column.

   b. Rename the new column *Size - Numbers* and change its type to Whole Number by selecting the ABC control in the header and Whole Number in the drop-down menu.

c. You now see Error values in all the cells that contained textual size codes (S, M, L, X, and NA), but you want to work on numbers only, so you need to replace the errors with nulls. To do this, select the Size - Numbers column, and on the Transform tab, select the Replace Values drop-down menu and then select Replace Errors. An easier way to find the Replace Errors transformation is by right-clicking on the column header and finding the transformation in the shortcut menu.

d. In the Replace Errors dialog box that opens, enter *null* in the Value box and click OK to close the dialog box.

3. To extract the numeric size values by using Column from Examples, instead of replacing errors with null, follow these steps:

   a. If you applied step 2, delete the last four steps that were created in Applied Steps. Your last step should now be Inserted Text Between Delimiter.

   b. Select the Size column. On the Add Column tab, select the Column from Examples drop-down menu and then select From Selection.

   c. When the Add Column from Examples pane opens, showing the new column, rename it *Size - Numbers*.

   d. Double-click the first blank cell in the Size - Numbers column. You can see that the value in the Size column is S. Because you are not interested in textual size values, enter *null* in the first blank cell and press Enter.

   e. When you see that the second and third values in the Size column are M and L, enter *null* in both the second and third cells of Size - Numbers and press Enter.

   f. Move to the cell in row 7. You can see that the value in the Size column is NA. Enter null in the seventh cell of the Size - Numbers column and press Enter.

   g. Move to row 21. You can see that the value in the Size column is X. Enter *null* in the corresponding cell of the Size - Numbers column and press Enter.

   h. Now, in row 22, you see the value 60. Enter *60* in the corresponding cell of the Size - Numbers column and press Enter. Power Query populates the new Size column with all the suggested values. You can now press Ctrl+Enter to create the new Size - Numbers column.

   In Applied Steps, you can now see the new Added Conditional Column step. Select the settings icon or double-click this step, and the Add Conditional Column dialog box opens. This dialog box allows you to create a column based on conditions in other columns. You could also have reached this point, and created a new column for the numeric size values, by applying a conditional column instead of by using Column from Examples. Close the Add Conditional Column dialog box. You will learn about this dialog box In Exercise 2-4.

4. Change the type of the Size - Numbers column to Whole Number by selecting the ABC control in the header of the column in the Preview pane and Whole Number in the drop-down menu.

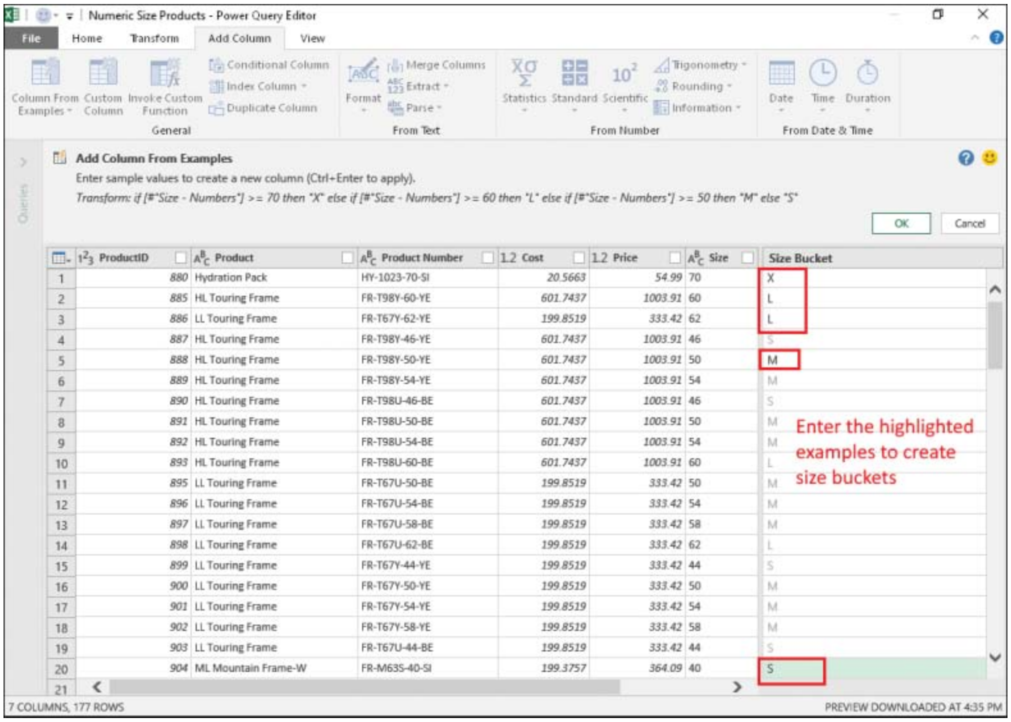
In the next part of this exercise, you will create a separate table for the products with numeric sizes and will classify them into four size buckets (S = Small, M = Medium, L = Large, and X = Extra Large) by using Column from Example.

5. In the Queries pane, right-click Products and select Reference. Rename the new query *Numeric-Size Products*. To rename the query, right-click the query in the Queries pane and select Rename in the shortcut menu, or rename the query in the Query Settings pane, under Properties, in the Name box.

Tip Using Reference is a very useful technique for branching from a source table and creating a new one, starting from the last step of the referenced query.

6. In the new query, remove all the products whose Size - Numbers value is null. To do so, click the filter control in the Size - Numbers column and select Remove Empty.

7. Say that you want to convert the numeric sizes into the buckets of X, L, M, and S as follows: Numbers equal to or greater than 70 will be assigned to the X bucket. Numbers equal to or greater than 60 will be assigned to L. Numbers equal to or greater than 50 will be assigned to M, and numbers equal to or greater than 40 will be assigned to S. Here are the steps to achieve this by using Column from Examples (see Figure 2-7 to ensure you follow the steps correctly):



Add Column From Examples

Enter sample values to create a new column (Ctrl+Enter to apply).

Transform: if [#"Size - Numbers"] >= 70 then "X" else if [#"Size - Numbers"] >= 60 then "L" else if [#"Size - Numbers"] >= 50 then "M" else "S"

	ProductID	Product	Product Number	Cost	Price	Size	Size Bucket
1	880	Hydration Pack	HY-1023-70-SI	20.5663	54.99	70	X
2	885	HL Touring Frame	FR-T98Y-60-YE	601.7437	1003.91	60	L
3	886	LL Touring Frame	FR-T67Y-62-YE	199.8519	333.42	62	L
4	887	HL Touring Frame	FR-T98Y-46-YE	601.7437	1003.91	46	S
5	888	HL Touring Frame	FR-T98Y-50-YE	601.7437	1003.91	50	M
6	889	HL Touring Frame	FR-T98Y-54-YE	601.7437	1003.91	54	M
7	890	HL Touring Frame	FR-T98U-46-BE	601.7437	1003.91	46	S
8	891	HL Touring Frame	FR-T98U-50-BE	601.7437	1003.91	50	M
9	892	HL Touring Frame	FR-T98U-54-BE	601.7437	1003.91	54	M
10	893	HL Touring Frame	FR-T98U-60-BE	601.7437	1003.91	60	L
11	895	LL Touring Frame	FR-T67U-50-BE	199.8519	333.42	50	M
12	896	LL Touring Frame	FR-T67U-54-BE	199.8519	333.42	54	M
13	897	LL Touring Frame	FR-T67U-58-BE	199.8519	333.42	58	M
14	898	LL Touring Frame	FR-T67U-62-BE	199.8519	333.42	62	L
15	899	LL Touring Frame	FR-T67Y-44-YE	199.8519	333.42	44	S
16	900	LL Touring Frame	FR-T67Y-50-YE	199.8519	333.42	50	M
17	901	LL Touring Frame	FR-T67Y-54-YE	199.8519	333.42	54	M
18	902	LL Touring Frame	FR-T67Y-58-YE	199.8519	333.42	58	M
19	903	LL Touring Frame	FR-T67U-44-BE	199.8519	333.42	44	S
20	904	ML Mountain Frame-W	FR-M63S-40-SI	199.3757	364.09	40	S
21							

7 COLUMNS, 177 ROWS

PREVIEW DOWNLOADED AT 4:35 PM

FIGURE 2-7 You can use Add Column from Examples to create size buckets.

   a. Select the Size - Numbers column, and in the Add Column tab, select the Column from Examples drop-down menu and then select From Selection.
   b. Rename the new column *Size Bucket*.
   c. Enter *X* in row 1 (because Size - Numbers is 70).
   d. Enter *L* in rows 2 and 3 (because the Size - Numbers values are 60 and 62, respectively).
   e. Enter *M* in row 5 (because Size - Numbers is 50).
   f. Enter S in row 20 (because Size - Numbers is 40).
   g. Press Ctrl+Enter to create the new Size - Bucket column. The Size - Bucket column now contains the required bucket names for the relevant ranges.

8. For learning purposes, double-click the Added Conditional Column step in the Applied Steps pane. The Add Conditional Column dialog box opens. Review the conditions that defined the ranges for each bucket, and close the dialog box.
9. Review the M formula in the formula bar for a quick glimpse into the M syntax, which is discussed in more detail throughout the book, specifically in Chapter 9, "Introduction to the Power Query M Formula Language."

You can download the solution files C02E03 - Solution - Part 2.xlsx and C02E03 - Solution - Part 2.pbix from https://aka.ms/DataPwrBIPivot/downloads.

Extracting Information from Text Columns

In the preceding exercises, you extracted meaningful information from delimiter-separated codes by using the Split Column by Delimiter transformation. In the next exercise, you will tackle a common challenge: how to extract meaningful data from unstructured textual data. While this challenge can be simple if your data is relatively consistent, you need a wider arsenal of techniques to address inconsistent data.

Exercise 2-4: Extracting Hyperlinks from Messages

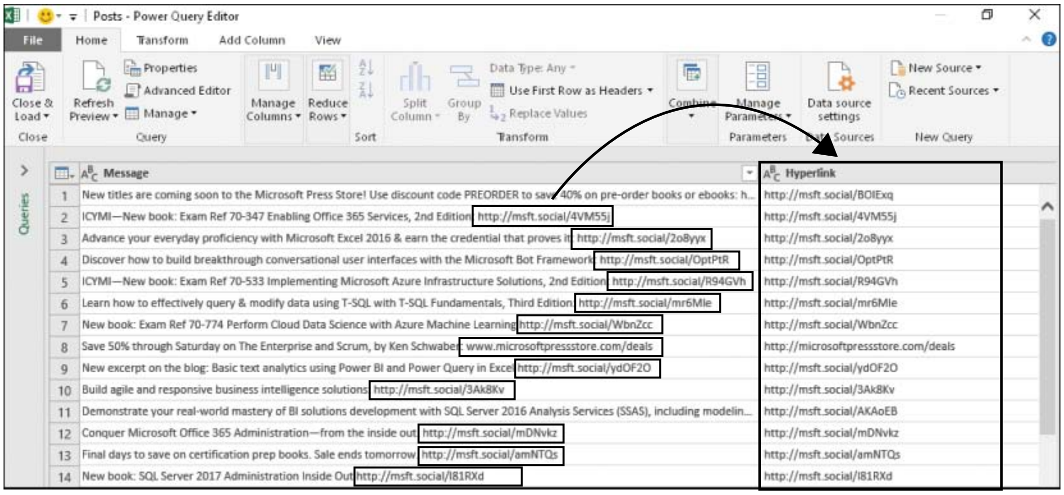
In this exercise you will work on a workbook that contains imported Facebook posts from the official Microsoft Press Facebook page. Specifically, you will extract the hyperlinks from posts that were shared by Microsoft Press. While some hyperlinks are easily extracted, as they start with the prefix "http://", there are plenty of edge cases that cannot be easily addressed.

Before you start, download the workbook C02E04.xlsx from https://aka.ms/DataPwrBIPivot/downloads and save it in *C:\Data\C02*.

Note While some elements in this exercise may be a bit advanced, requiring lightweight manipulation of formulas that are not explained in detail in this chapter, you will still find these elements extremely useful, and you will be able to apply them to your own data even if you do not fully understand them yet. Chapter 9 explains M in more detail, and it will help you feel more comfortable with the type of changes you encounter here.

1. Start a new blank Excel workbook or a new Power BI Desktop report.
2. Follow these steps to import C02E04.xlsx to the Power Query Editor:
   a. **In Excel:** On the Data tab, select Get Data, From File, From Workbook.
      In Power BI Desktop: In the Get Data drop-down menu, select Excel.
   b. Select the file C02E04.xlsx and select Import.
   c. In the Navigator dialog box that opens, select Sheet1 and then click Edit.

   The Power Query Editor opens, and you see that the Message column contains free text. Before you extract the hyperlinks from the Message column, to achieve your goal, as illustrated in Figure 2-8, you first need to duplicate the column to keep the original message intact, so select the Message column, and from the Add Column tab, select Duplicate Column.



	Message	Hyperlink
1	New titles are coming soon to the Microsoft Press Store! Use discount code PREORDER to save 40% on pre-order books or ebooks: h...	http://msft.social/BOIExq
2	ICYMI—New book: Exam Ref 70-347 Enabling Office 365 Services, 2nd Edition http://msft.social/4VM55j	http://msft.social/4VM55j
3	Advance your everyday proficiency with Microsoft Excel 2016 & earn the credential that proves it http://msft.social/2oByyx	http://msft.social/2oByyx
4	Discover how to build breakthrough conversational user interfaces with the Microsoft Bot Framework http://msft.social/OptPtR	http://msft.social/OptPtR
5	ICYMI—New book: Exam Ref 70-533 Implementing Microsoft Azure Infrastructure Solutions, 2nd Edition http://msft.social/R94GVh	http://msft.social/R94GVh
6	Learn how to effectively query & modify data using T-SQL with T-SQL Fundamentals, Third Edition http://msft.social/mr6Mle	http://msft.social/mr6Mle
7	New book: Exam Ref 70-774 Perform Cloud Data Science with Azure Machine Learning http://msft.social/WbnZcc	http://msft.social/WbnZcc
8	Save 50% through Saturday on The Enterprise and Scrum, by Ken Schwaber www.microsoftpressstore.com/deals	http://microsoftpressstore.com/deals
9	New excerpt on the blog: Basic text analytics using Power BI and Power Query in Excel http://msft.social/ydOF2O	http://msft.social/ydOF2O
10	Build agile and responsive business intelligence solutions http://msft.social/3Ak8Kv	http://msft.social/3Ak8Kv
11	Demonstrate your real-world mastery of BI solutions development with SQL Server 2016 Analysis Services (SSAS), including modelin...	http://msft.social/AKAoEB
12	Conquer Microsoft Office 365 Administration—from the inside out http://msft.social/mDNvkz	http://msft.social/mDNvkz
13	Final days to save on certification prep books. Sale ends tomorrow http://msft.social/amNTQs	http://msft.social/amNTQs
14	New book: SQL Server 2017 Administration Inside Out http://msft.social/I81RXd	http://msft.social/I81RXd

FIGURE 2-8 You can extract hyperlinks from Microsoft Press Facebook posts.

3. Rename the new column *Hyperlink*.
4. To extract the hyperlink from the new column, you can use the hyperlink prefix “http://” as a delimiter and split the column, so select the Hyperlink column, and on the Transform tab, select Split Column and then select By Delimiter.

5. In the Split Column by Delimiter dialog box that opens, do the following:

   a. Enter *"http://"* in the text box below Custom.

   b. Select the Left-Most Delimiter option at Split At radio buttons and click OK to close the dialog box.

   You can now see that the Hyperlink column has been split into two columns: Hyperlink.1 and Hyperlink.2. But how can you extract hyperlinks that start with "https://" or perhaps just "www."?

6. In Applied Steps, select the step Split Column by Delimiter.

7. Ensure that your formula bar in the Power Query Editor is active. If you don't see it, go to the View tab and select the Formula Bar check box. Now, you can see the following formula, which was generated in step 7:

```
= Table.SplitColumn(#"Renamed Columns", "Hyperlink",
Splitter.SplitTextByEachDelimiter({"http://"},
QuoteStyle.Csv, false), {"Hyperlink.1", "Hyperlink.2"})
```

8. To split the Hyperlink column by additional delimiters, you can change the function name from *Splitter.SplitTextBy**Each**Delimiter* to *Splitter.SplitTextBy**Any**Delimiter*. Now, you can add the new delimiters "https://" and "www.". You can see in the formula above that "https://" is wrapped in double quotes inside curly brackets. Curly brackets represent lists in the M language of Power Query. You can now add multiple items to the list by adding double-quoted comma-separated delimiters, like this: *{"http://", "https://", "www."}*.

   Here is the complete modified formula:

```
= Table.SplitColumn(#"Renamed Columns", "Hyperlink",
Splitter.SplitTextByAnyDelimiter({"http://", "https://", "www."},
QuoteStyle.Csv, false), {"Hyperlink.1", "Hyperlink.2"})
```

9. Thanks to the addition of "www." as a delimiter, you can now see in the Preview pane that in line 8 a new hyperlink was detected in the Hyperlink.2 column: microsoftpressstore.com/deals.

10. Go back to the last step in Applied Steps.

11. To audit the results in Hyperlink.2, in row 15, notice that the original message contains additional text following the hyperlink. To remove the trailing text from the hyperlink, you can assume that the hyperlinks end with a space character and split Hyperlink.2. To do it, select the Hyperlink.2 column. On the Transform tab, select Split Column and then select By Delimiter. The Split Column by Delimiter dialog box opens.

    a. Select Space from the Select or Enter Delimiter drop-down.

    b. Select the Left-Most Delimiter option for Split At radio buttons and click OK to close the dialog box.

12. Remove the Hyperlink.2.2 column.

    To continue auditing the results in Hyperlink.2, scroll down until you find the first null value, in row 29. A null value means that you couldn't detect any hyperlink in the message. But if you take a closer look at the message in the Hyperlink.1 column, you can see that it contains the bolded hyperlink, as shown in the line *Discover Windows containers in this free ebook. Download here:* **`aka.ms/containersebook`**.

The next challenge is to extract the hyperlinks whose domain name is *aka.ms* and include them in the Hyperlink column. This challenge is rather complex because after you split the Message column by the delimiter *aka.ms*, the domain name will be removed from the split results. When you applied "www." in step 8, "www." was removed from the hyperlink, but that was a safe manipulation. You were taking the risk that the resulted hyperlink will work well without the "www." prefix because it is common to omit this prefix from hyperlinks (e.g., www.microsoft.com and microsoft.com lead to the same website).

You need to find a way to split by aka.ms, and you must find a way to return the domain name, but only for rows that contain that domain. You will soon learn how to do it by using a conditional column. But first, you need to add aka.ms to the list of delimiters.

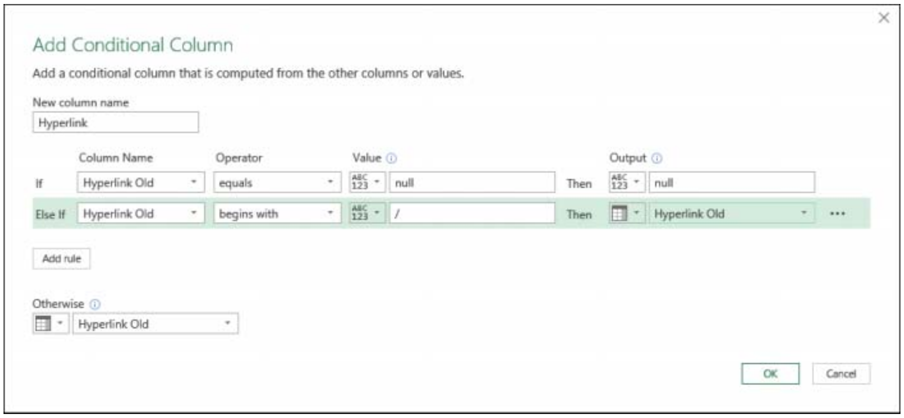
13. In Applied Steps, select again the first Split Column by Delimiter step, which you modified in step 8, and add aka.ms to the list of delimiters. Here is the modified formula:

```
= Table.SplitColumn(#"Renamed Columns", "Hyperlink",
Splitter.SplitTextByAnyDelimiter({"http://", "https://", "www.", "aka.ms"},
QuoteStyle.Csv, false), {"Hyperlink.1", "Hyperlink.2"})
```

    You can now see in row 29 that the hyperlink is /containersebook. In the next steps, you will add a conditional column that will return the missing domain name aka.ms if the values in the Hyperlink column start with /.

14. In Applied Steps, go back to the last step. Remove the Hyperlink.1 column, and rename Hyperlink.2.1 to *Hyperlink Old*.

15. On the Add Column tab, select Conditional Column. When the Add Conditional Column dialog box opens, follow these steps (see Figure 2-9):


Add Conditional Column

Add a conditional column that is computed from the other columns or values.

New column name: Hyperlink

	Column Name	Operator	Value		Output
If	Hyperlink Old	equals	null	Then	null
Else If	Hyperlink Old	begins with	/	Then	Hyperlink Old

Otherwise: Hyperlink Old

FIGURE 2-9 You can add a conditional column as a preliminary step in adding back the domain name aka.ms. Note that the conditions in the dialog box are used as a stepping-stone to the final conditions.

Note The use of the Add Conditional Column dialog box will help you set the stage for the final conditions. The user interface is limited, and you are using it as a shortcut to generate part of the required conditions. The final touch will require a lightweight modification of the formula in the formula bar in step 16.

a. Enter *Hyperlink* in the New Column Name box.
b. Select Hyperlink Old from the Column Name drop-down for If.
c. Select Equals from the Operator drop-down and enter *null* in both the Value and Output text boxes.
d. Click the Add Rule button to add a new condition line.
e. Select Hyperlink Old from the Column Name drop-down for Else If.
f. Select Begins With as the operator in the second line.
g. Ensure that Enter a Value is selected in the ABC123 drop-down menu in the second line, under Value, and enter / in the Value box.
h. In the drop-down menu after the Then label for Else If, select Select a Column and then select Hyperlink Old.
i. In the drop-down menu below the Otherwise label, select Select a Column and then select Hyperlink Old.
j. Ensure that your settings match those in Figure 2-9 and click OK to close the dialog box.

16. Now look at the formula bar. It's time to modify the M formula to add the prefix aka.ms if a hyperlink starts with /. Here is the original formula, following step 15:

```
= Table.AddColumn(#"Renamed Columns1", "Custom", each if [Hyperlink Old] = null
then null else if Text.StartsWith([Hyperlink Old], "/") then [Hyperlink Old] else
[Hyperlink Old])
```

To fix it, you can concatenate aka.ms and [Hyperlink Old]. Much as you would do in an Excel formula, you can add aka.ms as a prefix in the Hyperlink Old column by using the *&* operator:

```
"aka.ms" & [Hyperlink Old]
```

Here is the modified formula, including the change, in bold:

```
= Table.AddColumn(#"Renamed Columns1", "Custom", each if [Hyperlink Old] = null
then null else if Text.StartsWith([Hyperlink Old], "/") then
"aka.ms" & [Hyperlink Old] else [Hyperlink Old])
```

Note In step 15, the first null condition was added to prevent *Text.StartsWith* from being applied on a null value in [Hyperlink Old], which would result in an error.

17. Remove the Hyperlink Old column.

You are almost done, but there are few more surprises waiting for you. Continue auditing the results in the Hyperlink column by scrolling down to row 149. The Hyperlink cell is blank, but looking at the original Message column, you can see the following hyperlink inside the message: *https://www.microsoftpressstore.com/Ignite*.

Why is the Hyperlink cell blank in this case? When you split this message in step 14, both "https://" and "www." participated in the split as delimiters. As a result, the split included three separated values, but only the first two were loaded. The next steps help you resolve this problem.

18. In Applied Steps, select the first Split Column by Delimiter step. Change the bolded section in the formula below from *{"Hyperlink.1", "Hyperlink.2"}* to *3*:

```
= Table.SplitColumn(#"Renamed Columns", "Hyperlink",
Splitter.SplitTextByAnyDelimiter({"http://", "https://", "www.", "aka.ms"},
QuoteStyle.Csv, false), {"Hyperlink.1", "Hyperlink.2"})
```

Here is the complete modified formula:

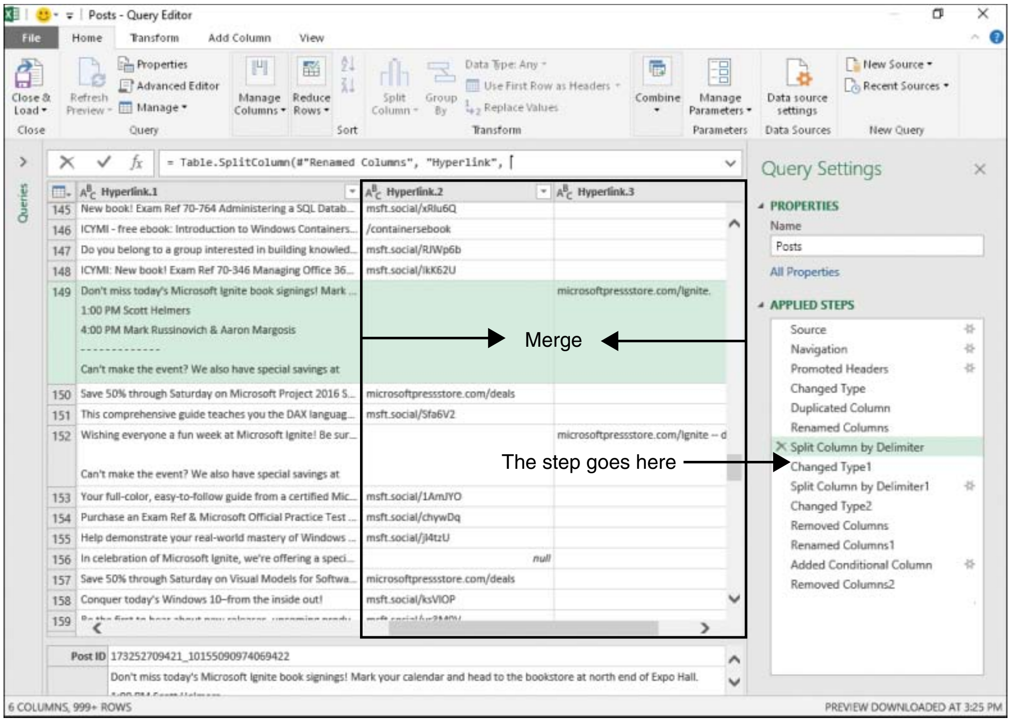
```
= Table.SplitColumn(#"Renamed Columns", "Hyperlink",
Splitter.SplitTextByAnyDelimiter({"http://", "https://", "www.", "aka.ms"},
QuoteStyle.Csv, false), 3)
```

See Also Here you use the argument 3 instead of the list of column names *{"Hyperlink.1", "Hyperlink.2", Hyperlink.3"}*. The results are the same. The *Table.SplitColumn* function accepts either the column names or the number of new columns to use in the split. The fact is, you miss crucial data when you use only the default split; this is discussed in more detail in Chapter 10, "From Pitfalls to Robust Queries."

Now, as shown in Figure 2-10, you have the hyperlink values either in Hyperlink.2 (in most of the rows) or in Hyperlink.3 (in row 149). If you merge the two columns, you can retrieve the missing hyperlinks and fix the query.

19. While the Split Column by Delimiter step is still selected in Applied Steps, insert the following steps into the transformation sequence:

    a. Select the two columns Hyperlink.2 and Hyperlink.3.
    b. On the Transform tab, select Merge Columns.
    c. When the Insert Step dialog box opens, warning you that this step will be inserted between the existing steps and may break your subsequent steps, select Insert.
    d. In the Merge Columns dialog box that opens, keep the default settings and click OK to close the dialog box.
    e. Rename the Merged column *Hyperlink.2*.
    f. When the Insert Step dialog box opens again, select Insert.



```
= Table.SplitColumn(#"Renamed Columns", "Hyperlink",
```

	Hyperlink.1	Hyperlink.2	Hyperlink.3
145	New book! Exam Ref 70-764 Administering a SQL Datab...	msft.social/xRlu6Q	
146	ICYMI - free ebook: Introduction to Windows Containers...	/containersebook	
147	Do you belong to a group interested in building knowled...	msft.social/RJWp6b	
148	ICYMI: New book! Exam Ref 70-346 Managing Office 36...	msft.social/IkK62U	
149	Don't miss today's Microsoft Ignite book signings! Mark ... 1:00 PM Scott Helmers 4:00 PM Mark Russinovich & Aaron Margosis ------------- Can't make the event? We also have special savings at		microsoftpressstore.com/Ignite.
150	Save 50% through Saturday on Microsoft Project 2016 S...	microsoftpressstore.com/deals	
151	This comprehensive guide teaches you the DAX languag...	msft.social/Sfa6V2	
152	Wishing everyone a fun week at Microsoft Ignite! Be sur... Can't make the event? We also have special savings at		microsoftpressstore.com/Ignite -- d
153	Your full-color, easy-to-follow guide from a certified Mic...	msft.social/1AmJYO	
154	Purchase an Exam Ref & Microsoft Official Practice Test ...	msft.social/chywDq	
155	Help demonstrate your real-world mastery of Windows ...	msft.social/jl4tzU	
156	In celebration of Microsoft Ignite, we're offering a speci...	*null*	
157	Save 50% through Saturday on Visual Models for Softwa...	microsoftpressstore.com/deals	
158	Conquer today's Windows 10–from the inside out!	msft.social/ksVlOP	

Post ID 173252709421_10155090974069422
Don't miss today's Microsoft Ignite book signings! Mark your calendar and head to the bookstore at north end of Expo Hall.

6 COLUMNS, 999+ ROWS PREVIEW DOWNLOADED AT 3:25 PM

Name: Posts

Applied Steps:
- Source
- Navigation
- Promoted Headers
- Changed Type
- Duplicated Column
- Renamed Columns
- Split Column by Delimiter
- Changed Type1
- Split Column by Delimiter1
- Changed Type2
- Removed Columns
- Renamed Columns1
- Added Conditional Column
- Removed Columns2

FIGURE 2-10 You can merge the Hyperlink.2 column with the Hyperlink.3 column to extract hyperlinks that start with https://www.

20. You can now select the last step in Applied Steps to ensure that the preceding steps work as expected.

There are still two cases that have not been handled correctly. First, in row 149 you can see that the hyperlinks don't end with a space but with other punctuation, such as a dot. As a result, in step 11, when you applied space as a delimiter, you were not able to clean the trailing message from the hyperlink value. To fix it, you should trim the punctuation from all hyperlinks.

21. Select the Hyperlink column, and on the Transform tab, select Format and then select Trim. Alternatively, right-click the Hyperlink column, and in the shortcut menu, select Transform and then select Trim.

By default, Trim removes whitespace from the beginning and end of text values. You can manipulate it to also trim the trailing punctuation. This requires some changes in the formula bar. You can see that the formula includes the bolded element *Text.Trim*:

```
= Table.TransformColumns(#"Removed Columns1",{{"Hyperlink", Text.Trim, type text}})
```

The M function *Text.Trim* accepts a list of text items as its second argument to trim leading and trailing text other than spaces. In M, the punctuation list—dot, comma, and closing parenthesis—can be defined with curly brackets and comma-separated double-quoted values, as shown here:

```
{ ".", ",", ")" }
```

To feed the list as the second argument to *Text.Trim*, you should also feed its first argument, which is the actual text in the Hyperlink column. To get the actual text, you can use the combination of the keyword *each* and the underscore character. (Chapter 9 explains this in more details.) Copy this modified formula to the formula bar to trim the punctuations from the Hyperlink column:

```
= Table.TransformColumns(#"Removed Columns1",{{"Hyperlink", each Text.Trim(_,
{".",",",")"}), type text}})
```

22. Notice the final issue in row 174, where you can see that the hyperlink ends with a new line, followed by more text. When you applied the second Split Text By Delimiter using a space, you didn't extract the hyperlink correctly. To fix it, select the second Split Text By Delimiter in Applied Steps.

23. In the formula bar, notice the following code:

```
= Table.SplitColumn(#"Changed Type1", "Hyperlink.2",
Splitter.SplitTextByEachDelimiter({" "}, QuoteStyle.Csv, false),
{"Hyperlink.2.1", "Hyperlink.2.2"})
```

Change it to the following (where changes are highlighted in bold):

```
= Table.SplitColumn(#"Changed Type1", "Hyperlink.2",
Splitter.SplitTextByAnyDelimiter({" ", "#(lf)"}, QuoteStyle.Csv, false),
{"Hyperlink.2.1", "Hyperlink.2.2"})
```

The value "*#(lf)*" describes the special line-feed character. In the preceding formula, you used the advanced split function *Splitter.SplitTextByAnyDelimiter* instead of *Splitter.SplitTextByDelimiter* to split by both spaces and line feeds.

24. Finally, to include the prefix "http://" in all the Hyperlink values, in Applied Steps, go back to the last step. Select the Hyperlink column, and on the Transform tab, select Format and then select Add Prefix. When the Prefix dialog box opens, enter *"http://"* in the Value box and close the dialog box.

25. In the preceding step, you added the prefix "http://" in all the rows, even where there are no URLs. But there are legitimate cases in which the Hyperlink column should be empty (for example, row 113). To remove "http://" from cells without hyperlinks, follow these steps:

    a. Select the Hyperlink column. Then, on the Home tab, select Replace Values.

    b. When the Replace Values dialog box opens, enter *"http://"* in the Value to Find box.

    c. Leave the Replace With box empty.

    d. Expand Advanced Options and select the Match Entire Cell Contents check box. (This step is crucial. Without it, Power Query removes "http://" from all the values.)

    e. Click OK to close the dialog box and select Close & Load in Excel or Close & Apply in Power BI Desktop to load the messages and their extracted hyperlinks to the report.

By now, you have resolved most of the challenges in this dataset and extracted the required hyperlinks. You can download the solution files C02E04 - Solution.xlsx and C02E04 - Solution.pbix from https://aka.ms/DataPwrBIPivot/downloads.

See Also There are still a few hyperlinks in the Message column. These hyperlinks do not start with “www.”, “http”, “https”, or “aka.ms”. Can you find them? In Chapter 11, “Basic Text Analytics,” you will learn advanced techniques for detecting keywords in messages. These techniques will help you detect a wider list of domain names and extract the hyperlinks.

Handling Dates

One of the most common data preparation challenges is dealing with data types. While text columns are easy to handle, numbers and dates can make even the simplest datasets daunting for analysis. In this section you will learn how to handle dates and times. You will start with a common challenge—converting text to dates—and then move on to more challenging cases involving invalid dates. At the end of this section you will learn how to extract specific date or time elements from date/time values.

When you load a table with dates or date/time values, Power Query converts the relevant columns to their correct date/time format. In Exercise 2-4, you imported a dataset with a Date/Time column. Power Query automatically converted the *Date* column to *Date/Time*. Exercise 2-5 shows how Power Query handles a column with mixed date formats.

Exercise 2-5: Handling Multiple Date Formats

Download the workbook C02E05.xlsx from https://aka.ms/DataPwrBIPivot/downloads and save it in *C:\Data\C02*. This workbook contains the AdventureWorks product catalog, with the release date of each product in the last column. Imagine that different data entry teams had typed the products’ release dates in five different formats:

- 7/1/2018
- 2018-07-01
- 7.1.2018
- Jul 1, 2018
- 1 July, 2018

For the sake of simplicity, assume that all the teams use the English/United States Locale in their regional settings, so the first date, 7/1/2018, is July 1 (not January 7).

When you try to work with these different formats and apply date calculations, you see that Excel does not recognize the dates in all the cells, and for some cells, it returns a *#VALUE!* error.

1. Open a copy of the workbook C02E05.xlsx. (Be sure to use a copy rather than the original workbook.)

2. In cell H2, add the following formula:

   ```
   = G2 + 1
   ```

   This formula should increment the release date by one day, but the result is 43283. This is because Excel stores dates as sequential serial numbers, so that they can be used in calculations. By default, January 1, 1900, is serial number 1, and July 1, 2018, is serial number 43282 (indicating the number of days since 1/1/1900).

3. To view the dates correctly, change the format of column H to Date. After you make this change, copy and paste the formula from step 2 into the next four rows: H3, H4, H5, and H6.

   Cells H4 and H6 return a *#VALUE!* error because Excel cannot find a way to convert them to dates or numbers to increment them by one.

4. It's time to learn how Power Query can help in this scenario, so open a new workbook or launch Power BI Desktop. Because your input data is based on English (United States) regional settings, you need to ensure that you use this locale in this exercise.

   In Excel: While you are still in the Power Query Editor, select File. Select Options and Settings, and then select Query Options. Under Current Workbook, select Regional Settings and then select English (United States) from the Locale drop-down.

   In Power BI Desktop: Select File, Options and Settings, Options. Under Current File, select Regional Settings and then select English (United States) from the Locale drop-down.

5. Click OK to close the Options dialog box.

6. Import the workbook C02E05.xlsx into the Power Query Editor:

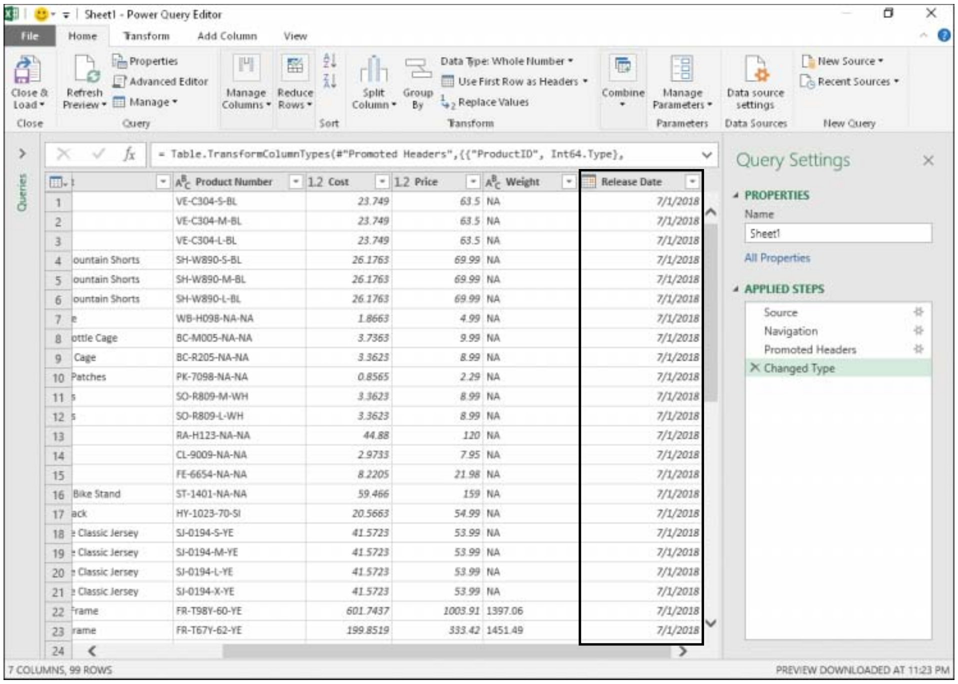
   a. **In Excel:** On the Data tab, select Get Data, From File, From Workbook.

      In Power BI Desktop: In the Get Data drop-down menu, select Excel.

   b. Select the file C02E05.xlsx and select Import.

   c. When the Navigator dialog box opens, select Sheet1 and then select Edit.

7. When the Power Query Editor opens, notice that all the date values in the Release Date column are detected correctly, as shown in Figure 2-11.



```
= Table.TransformColumnTypes(#"Promoted Headers",{{"ProductID", Int64.Type},
```

		Product Number	Cost	Price	Weight	Release Date
1		VE-C304-S-BL	23.749	63.5	NA	7/1/2018
2		VE-C304-M-BL	23.749	63.5	NA	7/1/2018
3		VE-C304-L-BL	23.749	63.5	NA	7/1/2018
4	ountain Shorts	SH-W890-S-BL	26.1763	69.99	NA	7/1/2018
5	ountain Shorts	SH-W890-M-BL	26.1763	69.99	NA	7/1/2018
6	ountain Shorts	SH-W890-L-BL	26.1763	69.99	NA	7/1/2018
7	e	WB-H098-NA-NA	1.8663	4.99	NA	7/1/2018
8	ottle Cage	BC-M005-NA-NA	3.7363	9.99	NA	7/1/2018
9	Cage	BC-R205-NA-NA	3.3623	8.99	NA	7/1/2018
10	Patches	PK-7098-NA-NA	0.8565	2.29	NA	7/1/2018
11	s	SO-R809-M-WH	3.3623	8.99	NA	7/1/2018
12	s	SO-R809-L-WH	3.3623	8.99	NA	7/1/2018
13		RA-H123-NA-NA	44.88	120	NA	7/1/2018
14		CL-9009-NA-NA	2.9733	7.95	NA	7/1/2018
15		FE-6654-NA-NA	8.2205	21.98	NA	7/1/2018
16	Bike Stand	ST-1401-NA-NA	59.466	159	NA	7/1/2018
17	ack	HY-1023-70-SI	20.5663	54.99	NA	7/1/2018
18	e Classic Jersey	SJ-0194-S-YE	41.5723	53.99	NA	7/1/2018
19	e Classic Jersey	SJ-0194-M-YE	41.5723	53.99	NA	7/1/2018
20	e Classic Jersey	SJ-0194-L-YE	41.5723	53.99	NA	7/1/2018
21	e Classic Jersey	SJ-0194-X-YE	41.5723	53.99	NA	7/1/2018
22	Frame	FR-T98Y-60-YE	601.7437	1003.91	1397.06	7/1/2018
23	rame	FR-T67Y-62-YE	199.8519	333.42	1451.49	7/1/2018
24						

7 COLUMNS, 99 ROWS

FIGURE 2-11 Power Query automatically changes the column type from *Text* to *Date* and correctly converts the different date formats.

8. Close the Power Query Editor and load the data to your report.

> **Tip** If you import dates of a different locale than the one defined for your Microsoft Windows operating system, you can define the specific locale in the Query Options dialog box, as shown in step 4. After you refresh the preview in the Power Query Editor, you see that the dates are recognized correctly.

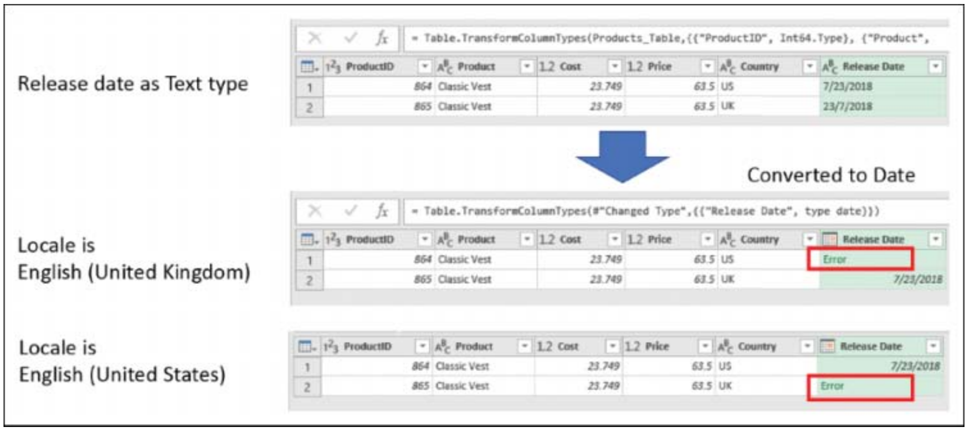
Exercise 2-6: Handling Dates with Two Locales

In Exercise 2-5, you had multiple date formats, but you assumed that all dates were used by the same locale. In this exercise, you will learn how to handle cases where your dates are for multiple locales, and the months and days values should be swapped for some of the rows.

Download the workbook C02E06.xlsx from https://aka.ms/DataPwrBIPivot/downloads and save it in *C:\Data\C02*. For simplicity, the workbook contains only two rows from the AdventureWorks product catalog—each row for a product with a different country and release date formatted by the specific country's locale:

- Country: United States, Release Date: 7/23/2018
- Country: United Kingdom, Release Date: 23/7/2018

As you can see in Figure 2-12, the Release Date column, when changed to the *Date* type, returns an error value for the UK row if your locale in the Power Query Options dialog box is set to English (United States); it returns an error value for the US row if your locale is English (United Kingdom) in that dialog box.


Release date as Text type
Converted to Date
Locale is English (United Kingdom)
Locale is English (United States)

```
= Table.TransformColumnTypes(Products_Table,{{"ProductID", Int64.Type}, {"Product",
```

	ProductID	Product	Cost	Price	Country	Release Date
1	864	Classic Vest	23.749	63.5	US	7/23/2018
2	865	Classic Vest	23.749	63.5	UK	23/7/2018

```
= Table.TransformColumnTypes(#"Changed Type",{{"Release Date", type date}})
```

	ProductID	Product	Cost	Price	Country	Release Date
1	864	Classic Vest	23.749	63.5	US	Error
2	865	Classic Vest	23.749	63.5	UK	7/23/2018

	ProductID	Product	Cost	Price	Country	Release Date
1	864	Classic Vest	23.749	63.5	US	7/23/2018
2	865	Classic Vest	23.749	63.5	UK	Error

FIGURE 2-12 Converting the Release Date column from *Text* to *Date* yields different errors, depending on the workbook's or file's locale.

To resolve this issue, you cannot rely on the locale in the Options dialog box. Instead, you can use the Split Column by Delimiter and then apply a conditional column, as described in this exercise.

1. Start a new blank Excel workbook or a new Power BI Desktop report.
2. Import C02E06.xlsx to the Power Query Editor:
    a. **In Excel:** On the Data tab, select Get Data, From File, From Workbook.
       In Power BI Desktop: In the Get Data drop-down menu, select Excel.
    b. Select the file C02E06.xlsx and select Import.
    c. In the Navigator dialog box that opens, select Products, and then select Edit.
3. Select the Release Date column. On the Transform tab, select Split Column and then select By Delimiter. The Split Column by Delimiter dialog box that opens detects the forward slash (/) as the delimiter. Click OK to close the dialog box.

   You can now see that Release Date is split into three columns: Release Date.1, Release Date.2, and Release Date.3. When Country is UK, Release Date.1 is the day, and Release Date.2 is the month. When Country is US, Release Date.1 is the month, and Release Date.2 is the day. In the next step, you will learn how to create a new column with the correct date, using an M formula and a custom column. Then, in steps 6–10, you will learn an alternative way that doesn't require you to know M but that involves more interactions with the user interface of the Power Query Editor.

4. On the Add Column tab, select Custom Column. The Custom Column dialog box opens.

   a. Enter *Date* in the New Column Name box.

   b. Enter the following formula in the Custom Column Formula box:

   ```
   if [Country] = "US" then
       #date([Release Date.3], [Release Date.1], [Release Date.2])
   else
       #date([Release Date.3], [Release Date.2], [Release Date.1])
   ```

   Then click OK to close the dialog box.

   c. Change the type of the Date column to *Date*.

   d. Remove the columns Release Date.1, Release Date.2, and Release Date.3.

Note In steps 6–10, you will learn the alternative way to create the date. Before you proceed, you should duplicate the current query so you can keep both versions.

5. To duplicate the Products query, right-click Products in the Queries pane and select Duplicate. A new query, Products (2), is created. Select it and in Applied Steps, delete the last three steps (including Added Custom).

6. Using the Ctrl key, select the three split date columns in this order: Release Date.1, Release Date.2, and Release Date.3. Now, on the Add Column tab, select Merge Columns. The Merge Columns dialog box opens.

   a. Select --Custom as the separator and then enter a forward slash character (/) in the box below.

   b. Enter *US Date* in the New Column Name box and click OK to close the dialog box.

7. Now, create a date column for the UK locale. Using the Ctrl key, select the three split date columns in this order: Release Date.2, Release Date.1, and Release Date.3. Now, on the Add Column tab, select Merge Columns. The Merge Columns dialog box opens.

   a. Select --Custom as the separator and then enter forward slash character (/) in the box below.

   b. Enter *UK Date* in the New Column Name box and click OK to close the dialog box.

   You now have the columns US Date and UK Date with different date formats. It's time to select which date format to choose for each row.

8. On the Add Column tab, select Conditional Column. The Add Conditional Column dialog box opens. Follow these steps to set the dialog as shown in Figure 2-13:

   a. Enter *Date* in the New Column Name box.

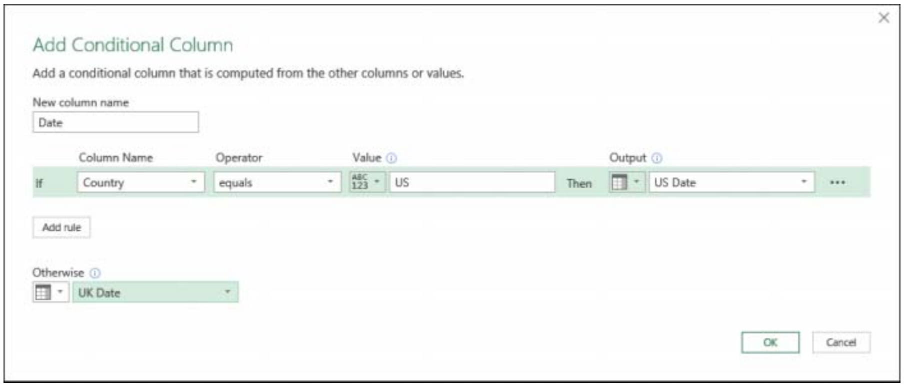
   b. Select Country from the Column Name drop-down.

   c. Select Equals as the operator and enter *US* in the box under Value.

d. In the drop-down menu below Output, select Select a Column and then select US Date.

e. In the drop-down menu below the Otherwise label, select Select a Column and then select UK Date.

f. Ensure that your settings match the screenshot in Figure 2-13 and click OK to close the Add Conditional Column dialog box.


Add Conditional Column

Add a conditional column that is computed from the other columns or values.

New column name: Date

	Column Name	Operator	Value		Output
If	Country	equals	US	Then	US Date

Otherwise: UK Date

FIGURE 2-13 You can add a conditional column to select the correct date by country.

9. Change the type of the Date column to Date.

10. Remove the columns Release Date.1, Release Date.2, Release Date.3, US Date, and UK Date.

11. Load the query to your Excel workbook or Power BI report. The dates are displayed according to your Windows Locale settings, and you do not have any errors.

You can download the solution files C02E06 - Solution.xlsx and C02E06 - Solution.pbix from https://aka.ms/DataPwrBIPivot/downloads.

Extracting Date and Time Elements

Now you have Date or DateTime columns in your queries, and the format is consistent. How can you extract meaningful elements from such values? The Power Query Editor includes a wide variety of functions, as well as transformations in the Transform and Add Column tabs, to enable you to extract years, quarters, months, days, and many other calculated elements in order to enrich your dataset with time intelligence.

Here are some of the useful transformation steps, which are available on the Transform and Add Column tabs, under the Date drop-down menus when you select a Date or Date/Time column:

- **Age:** You assign this transformation on a selected Date or Date/Time column to calculate the elapsed time since the specific date. This is implemented by subtracting the current Date/Time from the specific value.

- **Date Only:** Extracts the date portion from the selected Date/Time column. Combined with the transformation Time Only, you can transform a Date/Time column into two separate columns, of types Date and Time, which simplifies your model and reduces the size of your report and memory consumption.
- **Year:** Extracts the year portion from a Date or Date/Time column.
- **Start of Year/End of Year:** Extracts the first or last day of the year as a Date value.
- **Month:** Extracts the month portion from a Date or Date/Time column.
- **Start of Month/End of Month:** Extracts the first or last day of the month as a Date value.
- **Days in Month:** Calculates the number of days in the current month of the selected Date column.
- **Name of Month:** Returns the name of the month.
- **Quarter of Year:** Returns the quarter of the current year.
- **Start of Quarter/End of Quarter:** Extracts the first or last day of the quarter as a Date value.
- **Week of Year/Week of Month:** Calculates the number of the week in the year or month.
- **Start of Week/End of Week:** Extracts the first or last day of the week as a Date value.
- **Day:** Extracts the day portion from a Date or Date/Time column.
- **Start of Day/End of Day:** Extracts the first or last date and time of the day from a Date/Time column.
- **Day of Week/Day of Year:** Calculates the number of the current day in the week or year.
- **Name of Day:** Returns the name of the day (for example, Sunday, Monday).

When you select multiple Date or Date/Time columns, you can also apply Subtract Days to calculate the elapsed time between the two dates or compute the earliest or latest dates.

Preparing the Model

Data preparation is key to the success of data analysis. To set the stage for effective analysis, you often need to split tables into multiple tables to ensure that you have a single flat table for facts or transactions (for example, Sales Order) and supplementary tables to support your facts (for example, Products, Clients).

In Exercise 2-2, Part 3, you learned how to create relationships between fact tables and lookup tables. While the modeling elements in Power Pivot and Power BI are not the focus of this book, the capability to shape your tables to meet your needs is indeed the focus. In the next three chapters, you will learn how to combine multiple tables to simplify your model. In Chapter 6, "Unpivoting Tables," you will learn how to break down complex table structures to better support your analysis.

In this section, you will learn the imperative preparation steps that allow you to split an aggregated table into multiple tables, to build star schemas, and to control the granularity of your entities.

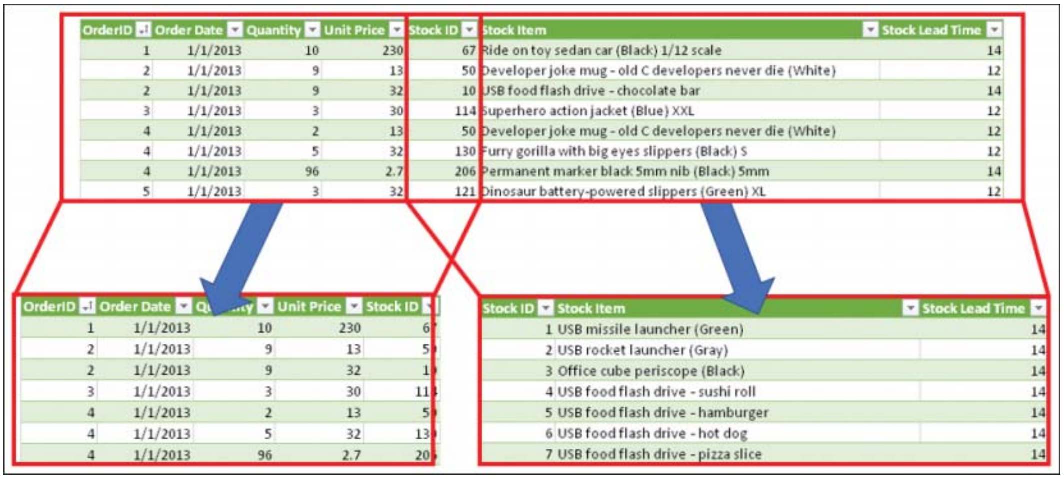
Exercise 2-7: Splitting Data into Lookup Tables and Fact Tables

In this exercise, you will import sample data from another fictitious company: Wide World Importers. This dataset, which is richer and newer than the AdventureWorks dataset, is also provided by Microsoft for learning purposes. To learn more about this data sample, go to https://blogs.technet.microsoft.com/dataplatforminsider/2016/06/09/wideworldimporters-the-new-sql-server-sample-database/.

The workbook C02E07.xslx which can be downloaded from https://aka.ms/DataPwrBIPivot/downloads, summarizes Wide World Importers orders. The goal of this exercise is to show you how easy it is to use Power Query to split a table into a fact table and a lookup table by using the Reference and Remove Duplicate options.

1. Download the workbook C02E07.xslx and save it in the folder *C:\Data\C02\*.
2. Open a new blank Excel workbook or a new Power BI Desktop report and import the workbook C02E07.xslx to the Power Query Editor.
3. In the Navigator dialog, select the Sales_Order table and then select Edit.
4. Rename the Sales_Order query *Sales Order - Base*.

   Your goal here is to split the original table into two tables, with the correct granularity levels, as shown in Figure 2-14. One of the tables (the bottom left table in Figure 2-14) is for the orders and their stock item identifiers (the fact table), and the second (the bottom right table in Figure 2-14) is for the stock items (the lookup table).



OrderID	Order Date	Quantity	Unit Price	Stock ID	Stock Item	Stock Lead Time
1	1/1/2013	10	230	67	Ride on toy sedan car (Black) 1/12 scale	14
2	1/1/2013	9	13	50	Developer joke mug - old C developers never die (White)	12
2	1/1/2013	9	32	10	USB food flash drive - chocolate bar	14
3	1/1/2013	3	30	114	Superhero action jacket (Blue) XXL	12
4	1/1/2013	2	13	50	Developer joke mug - old C developers never die (White)	12
4	1/1/2013	5	32	130	Furry gorilla with big eyes slippers (Black) S	12
4	1/1/2013	96	2.7	206	Permanent marker black 5mm nib (Black) 5mm	14
5	1/1/2013	3	32	121	Dinosaur battery-powered slippers (Green) XL	12

OrderID	Order Date	Qu[illegible]ity	Unit Price	Stock ID
1	1/1/2013	10	230	6
2	1/1/2013	9	13	5
2	1/1/2013	9	32	1
3	1/1/2013	3	30	11
4	1/1/2013	2	13	5
4	1/1/2013	5	32	13
4	1/1/2013	96	2.7	20

Stock ID	Stock Item	Stock Lead Time
1	USB missile launcher (Green)	14
2	USB rocket launcher (Gray)	14
3	Office cube periscope (Black)	14
4	USB food flash drive - sushi roll	14
5	USB food flash drive - hamburger	14
6	USB food flash drive - hot dog	14
7	USB food flash drive - pizza slice	14

FIGURE 2-14 You can split the Sales Order table into a fact table and a lookup table.

5. Right-click Sales Order - Base and select Reference. Rename the new query *Stock Items*. (To rename the query, right-click the query in the Queries pane and select Rename in the shortcut menu, or rename the query in the Query Settings pane, under Properties, in the Name box.)

6. Select the Stock Items query, and in Home tab, select Choose Columns.

7. In the Choose Columns dialog box that opens, deselect all columns and then select the columns Stock ID, Stock Item, and Stock Lead Time. Click OK to close the dialog box.

8. Select the Stock ID column, and on Home tab, select Remove Rows and then Remove Duplicates.

   You now have a lookup table for the stock items, with unique stock items on each row.

Note If you have multiple Stock ID values in the stock items table, you will not be able to create the relationship in step 13. This exercise deliberately avoids two crucial steps that are explained at the end of this exercise. Without these steps, Remove Duplicates by itself will not be sufficient to ensure that your lookup table has unique Stock ID values, and, as a result, the refresh will fail.

9. To create a new fact table for the orders, in the Queries pane, right-click on the Sales Order - Base query and select Reference.

10. Rename the new query *Sales Orders*. (To rename the query, right-click the query in the Queries pane and select Rename in the shortcut menu, or rename it in the Query Settings pane, under Properties, in the Name box.)

11. Select the Sales Orders query, and on the Home tab, select Choose Columns. The Choose Columns dialog box opens. Deselect the columns Stock Item and Stock Lead Time. Click OK to close the dialog box.

12. **In Excel:** On the Home tab, select Close & Load To to avoid loading the base query into your worksheet or Data Model. Then, in the Queries and Connections pane, select Sales Orders and Stock Items and load them into the Data Model, using the Load To option, as you did in Exercise 2-2, Part 2 step 11.

    In Power BI Desktop: Disable the loading of Sales Order - Base, as you did in Exercise 2-2, Part 2 step 11.

13. Create a relationship between the two tables through the Stock ID columns.

When a Relationship Fails

When you create lookup tables in Power Query, as you did in Exercise 2-7, and you use text columns as keys in the relationship, two additional steps must be performed before you remove the duplicate keys in step 8 to ensure that your report does not fail to refresh.

In Exercise 2-7, in step 8 you removed the duplicate Stock ID values from the lookup table. Unfortunately, Power Query was designed differently than the Data Model in Excel and Power BI. While Power

Query is always case sensitive, the Data Model is not. As a result, if your Stock ID value, *“Stock_1”*, will include a lowercase or uppercase variant (e.g. *“STOCK_1”*) or will include a trailing space (e.g. *“Stock_1 “*), Power Query will keep the two instances as unique values, but the Data Model will consider them duplicates and will issue the following error:

```
Column 'Stock ID' in Table 'Stock Items' contains a duplicate value 'Stock_1' and this is
not allowed for columns on the one side of a many-to-one relationship or for columns that
are used as the primary key of a table.
```

To experience this error, open the workbook C02E07.xslx, and modify two of the Stock ID values. Change one of the values to uppercase, and add a space character at the end of another value. Then save the workbook and try to refresh your Exercise 2-7 solution workbook or Power BI report.

To fix this issue, apply the following steps on the Sales Orders - Base query:

1. Launch the Power Query Editor and select the Sales Orders - Base query.
2. Select the Stock ID column, and in the Transform tab, select Format and then select Lowercase or Uppercase or Capitalize Each Word. Any of these three options will ensure that the keys are normalized, so variations of lowercase/uppercase values will be removed as duplicate values by Power Query.
3. Select the Stock ID column, and on the Transform tab, select Format and then Trim.
4. Refresh the workbook or report, and the error does not occur again.

Referencing and removing duplicates to create lookup tables are common steps. Many users who apply the Remove Duplicates step are not aware of the potential refresh failures that could occur for a dataset that contains lowercase/uppercase variations or trailing spaces. Oftentimes, the errors even appear sooner, as the users try to build the relationship. Handling the case-sensitive values and trimming the spaces before the removal of duplicates will resolve these issues.

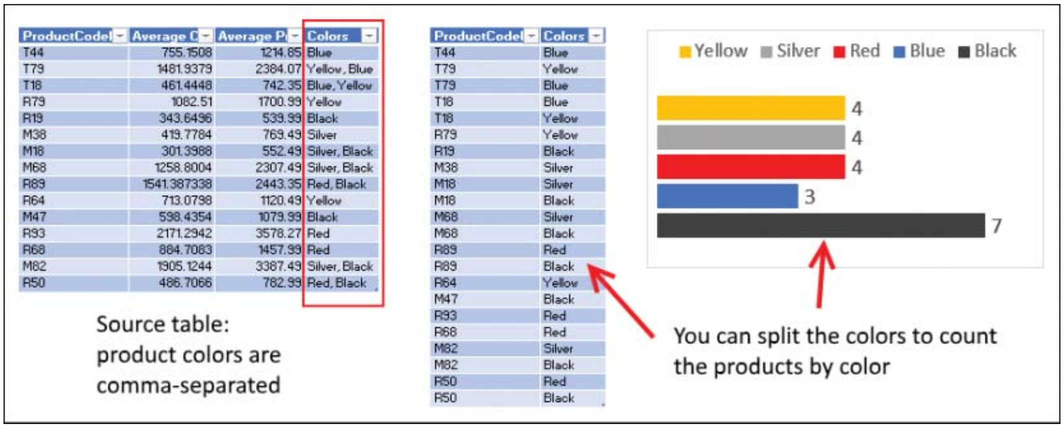
Download the solution files C02E07 - Solution.xlsx and C02E07 - Solution.pbix from https://aka.ms/DataPwrBIPivot/downloads.

See Also The pitfall of failing to apply these steps to prevent relationship refresh failures, is described at my blog at https://datachant.com/2017/02/14/power-bi-pitfall-10/. In Chapter 10, you will learn about more common pitfalls, or common errors that expose your reports to serious failures.

Exercise 2-8: Splitting Delimiter-Separated Values into Rows

In the preceding section, you learned how to split tables into fact and lookup tables. In this exercise, you will learn about a special case of fact and lookup tables. Imagine that your fact table acts as a mapping table that associates between members’ entities and their group unique identifiers. The lookup table in this scenario describes the group entities.

In this exercise, you will again use the AdventureWorks product table. Each product can be shipped in one or more colors. The associated colors are represented in comma-separated values, as illustrated in Figure 2-15, in the table to the left. In this exercise, you will learn how to split the products table into two tables: one for the entities (in this case, the products entities, without the colors) and the other one for the associations between groups and their members (in this case, product codes and colors).



ProductCodeI	Average C	Average P	Colors
T44	755.1508	1214.85	Blue
T79	1481.9379	2384.07	Yellow, Blue
T18	461.4448	742.35	Blue, Yellow
R79	1082.51	1700.99	Yellow
R19	343.6496	539.99	Black
M38	419.7784	769.49	Silver
M18	301.3988	552.49	Silver, Black
M68	1258.8004	2307.49	Silver, Black
R89	1541.387338	2443.35	Red, Black
R64	713.0798	1120.49	Yellow
M47	598.4354	1079.99	Black
R93	2171.2942	3578.27	Red
R68	884.7083	1457.99	Red
M82	1905.1244	3387.49	Silver, Black
R50	486.7066	782.99	Red, Black

ProductCodeI	Colors
T44	Blue
T79	Yellow
T79	Blue
T18	Blue
T18	Yellow
R79	Yellow
R19	Black
M38	Silver
M18	Silver
M18	Black
M68	Silver
M68	Black
R89	Red
R89	Black
R64	Yellow
M47	Black
R93	Red
R68	Red
M82	Silver
M82	Black
R50	Red
R50	Black

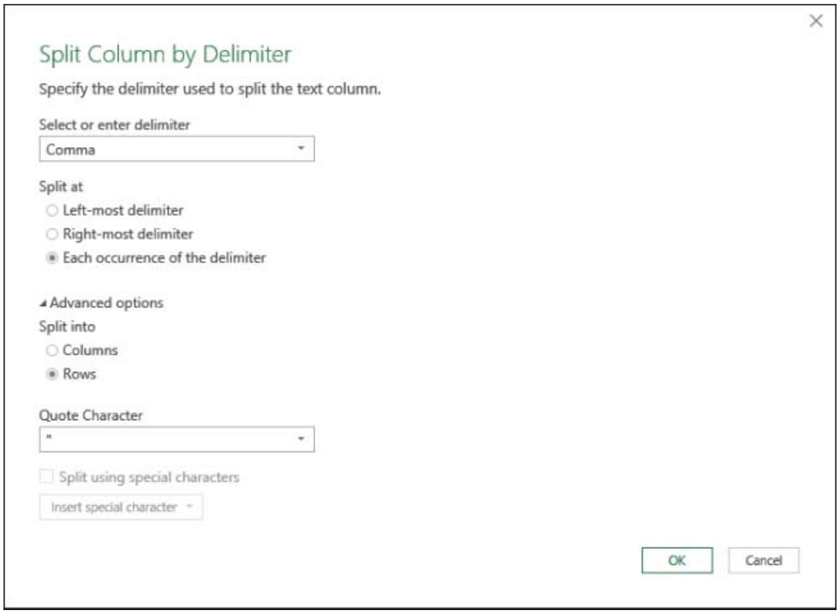
Source table: product colors are comma-separated
Yellow
Silver
Red
Blue
Black
4
4
4
3
7
You can split the colors to count the products by color

FIGURE 2-15 You can split the comma-separated AdventureWorks product colors column to find how many products are released by color.

For this exercise, you will use the sample workbook C02E08.xlsx, which can be downloaded from https://aka.ms/DataPwrBIPivot/downloads. The sample workbook summarizes AdventureWorks product codes by average cost, average price, and comma-separated colors. To create a report that shows the number of products of each color, as shown in Figure 2-15, do the following.

1. Download the workbook C02E08.xslx and save it in the folder *C:\Data\C02\*.
2. Open a new blank Excel workbook or a new Power BI Desktop report and import the workbook C02E08.xslx from *C:\Data\C02\* to the Power Query Editor.
3. In the Navigator dialog, select Products and then select Edit.
4. In the Queries pane, right-click Products and select Reference. Your goal is to create a new table with a mapping between the product codes and the colors.
5. Rename the new query *Products and Colors*. (To rename the query, you can right-click the query in the Queries pane and select Rename in the shortcut menu, or you can rename the query in the Query Settings pane, under Properties, in the Name box.)
6. With the new query selected, on the Home tab, select Choose Columns.
7. In the Choose Columns dialog box that opens, select ProductCodeNew and Colors and click OK to close the dialog box.
8. Select the Colors column, and on the Transform tab, select Split Column and then select By Delimiter.

9. In the Split Column by Delimiter dialog box that opens, note that the comma delimiter is selected by default, along with the option Each Occurrence of the Delimiter (see Figure 2-16). Expand the Advanced Options section and switch the Split Into option from Columns to Rows. Click OK to close the dialog box.



Split Column by Delimiter

Specify the delimiter used to split the text column.

Select or enter delimiter: Comma

Split at
- ○ Left-most delimiter
- ○ Right-most delimiter
- ● Each occurrence of the delimiter

Advanced options

Split into
- ○ Columns
- ● Rows

Quote Character: "

☐ Split using special characters

FIGURE 2-16 In the Split Column by Delimiter dialog box, don't forget to select Split Into, and then select Rows under Advanced Options.

10. In the Colors and Products query, select the Colors column. On the Transform tab, select Format and then select Trim. This step removes the leading spaces from the color values.

You can now load the two tables into the Data Model in Excel or Power BI Desktop and create a relationship between ProductCodeNew in Products and ProductCodeNew in Products and Colors. You can download and review the solution files C02E08 - Solution.xlsx and C02E08 - Solution.pbix from https://aka.ms/DataPwrBIPivot/downloads.

Tip To minimize the size of your report, you should remove the Colors column from the final Products table. However, if you try to do it now, you will get errors, as the Products query is referenced by Products and Colors. To resolve this issue and make it possible to remove Colors from Products, you need to create a new reference from Products and remove the Colors column from the referencing query. Then ensure that the original Products query is used only as a base query and will not get loaded to the Data Model or report. (In Exercise 2-2, Part 2, you learned how to disable the loading.)

Summary

Data preparation can be extremely time-consuming. But with the rich transformation capabilities available in the Power Query Editor, you can clean and prepare any dataset for analysis. This chapter introduced you to common data challenges and the ways Power Query provides to address them.

In Exercise 2-2 you learned how Power Query can extract codes and how to associate meaning with these codes, without the complex and difficult-to-maintain formulas presented in Exercise 2-1.

In Exercise 2-3, you were introduced to a very useful capability to extract meaning from columns, using examples. You saw that Column from Examples enables you to extract text, date, time, and numbers and offers a variety of transformations, based on the output examples you provide. You saw how this feature can be used to extract text between delimiters, apply a conditional column, and create buckets from ranges of numbers.

In Exercise 2-4, you learned how to parse hyperlinks from textual feed and apply different transformation techniques to achieve your goals. One of the most important lessons from this exercise is that you should audit your queries as often as possible and look for edge cases.

A common and highly time-consuming challenge involves handling inconsistent dates in your datasets. In Exercises 2-5 and 2-6, you learned how simple it is address inconsistent dates in Power Query.

Finally, in Exercises 2-7 and 2-8, you learned how to split a table into a fact table and a lookup table by using the Reference and Remove Duplicates options. You also learned how to avoid refresh errors by using Lowercase and Trim transformations, as well as how to split comma-separated values into rows to build group/members tables.

Even though you are still close to the beginning of this book, based on the exercises you have completed so far, you can probably reduce your data cleansing efforts by thousands of hours. In the next chapter, you will learn how to combine multiple tables and further reduce your time to insight.

Index