

LEARNING MADE EASY



2nd Edition

Microsoft®

Excel® Power Pivot & Power Query

for
dummies®
A Wiley Brand



Import and manage multiple
data sources directly in Excel

Automate data cleanup and
transformation tasks

Create interactive multidimensional
reporting models

Michael Alexander



Microsoft[®] Excel[®] Power Pivot & Power Query

2nd Edition

by Michael Alexander

for
dummies[®]
A Wiley Brand

Microsoft® Excel® Power Pivot & Power Query For Dummies®, 2nd Edition

Published by: **John Wiley & Sons, Inc.**, 111 River Street, Hoboken, NJ 07030-5774, www.wiley.com

Copyright © 2022 by John Wiley & Sons, Inc., Hoboken, New Jersey

Published simultaneously in Canada

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States Copyright Act, without the prior written permission of the Publisher. Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, (201) 748-6011, fax (201) 748-6008, or online at <http://www.wiley.com/go/permissions>.

Trademarks: Wiley, For Dummies, the Dummies Man logo, Dummies.com, Making Everything Easier, and related trade dress are trademarks or registered trademarks of John Wiley & Sons, Inc., and may not be used without written permission. Microsoft and Excel are registered trademarks of Microsoft Corporation in the United States and other countries. All other trademarks are the property of their respective owners. John Wiley & Sons, Inc., is not associated with any product or vendor mentioned in this book.

LIMIT OF LIABILITY/DISCLAIMER OF WARRANTY: WHILE THE PUBLISHER AND AUTHORS HAVE USED THEIR BEST EFFORTS IN PREPARING THIS WORK, THEY MAKE NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE ACCURACY OR COMPLETENESS OF THE CONTENTS OF THIS WORK AND SPECIFICALLY DISCLAIM ALL WARRANTIES, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. NO WARRANTY MAY BE CREATED OR EXTENDED BY SALES REPRESENTATIVES, WRITTEN SALES MATERIALS OR PROMOTIONAL STATEMENTS FOR THIS WORK. THE FACT THAT AN ORGANIZATION, WEBSITE, OR PRODUCT IS REFERRED TO IN THIS WORK AS A CITATION AND/OR POTENTIAL SOURCE OF FURTHER INFORMATION DOES NOT MEAN THAT THE PUBLISHER AND AUTHORS ENDORSE THE INFORMATION OR SERVICES THE ORGANIZATION, WEBSITE, OR PRODUCT MAY PROVIDE OR RECOMMENDATIONS IT MAY MAKE. THIS WORK IS SOLD WITH THE UNDERSTANDING THAT THE PUBLISHER IS NOT ENGAGED IN RENDERING PROFESSIONAL SERVICES. THE ADVICE AND STRATEGIES CONTAINED HEREIN MAY NOT BE SUITABLE FOR YOUR SITUATION. YOU SHOULD CONSULT WITH A SPECIALIST WHERE APPROPRIATE. FURTHER, READERS SHOULD BE AWARE THAT WEBSITES LISTED IN THIS WORK MAY HAVE CHANGED OR DISAPPEARED BETWEEN WHEN THIS WORK WAS WRITTEN AND WHEN IT IS READ. NEITHER THE PUBLISHER NOR AUTHORS SHALL BE LIABLE FOR ANY LOSS OF PROFIT OR ANY OTHER COMMERCIAL DAMAGES, INCLUDING BUT NOT LIMITED TO SPECIAL, INCIDENTAL, CONSEQUENTIAL, OR OTHER DAMAGES.

For general information on our other products and services, please contact our Customer Care Department within the U.S. at 877-762-2974, outside the U.S. at 317-572-3993, or fax 317-572-4002. For technical support, please visit <https://hub.wiley.com/community/support/dummies>.

Wiley publishes in a variety of print and electronic formats and by print-on-demand. Some material included with standard print versions of this book may not be included in e-books or in print-on-demand. If this book refers to media such as a CD or DVD that is not included in the version you purchased, you may download this material at <http://booksupport.wiley.com>. For more information about Wiley products, visit www.wiley.com.

Library of Congress Control Number: 2022930084

ISBN 978-1-119-84448-8 (pbk); ISBN 978-1-119-84449-5 (ebk); ISBN 978-1-119-84450-1 (ebk)

Contents at a Glance

Introduction	1
Part 1: Supercharged Reporting with Power Pivot	5
CHAPTER 1: Thinking Like a Database	7
CHAPTER 2: Introducing Power Pivot	17
CHAPTER 3: The Pivotal Pivot Table	29
CHAPTER 4: Using External Data with Power Pivot	63
CHAPTER 5: Working Directly with the Internal Data Model	89
CHAPTER 6: Adding Formulas to Power Pivot	103
CHAPTER 7: Diving into DAX	121
Part 2: Wrangling Data with Power Query	141
CHAPTER 8: Introducing Power Query	143
CHAPTER 9: Power Query Connection Types	159
CHAPTER 10: Transforming Your Way to Better Data	175
CHAPTER 11: Making Queries Work Together	207
CHAPTER 12: Extending Power Query with Custom Functions	225
Part 3: The Part of Tens	243
CHAPTER 13: Ten Ways to Improve Power Pivot Performance	245
CHAPTER 14: Ten Tips for Working with Power Query	253
Index	263

Table of Contents

INTRODUCTION	1
About This Book	2
Foolish Assumptions	3
Icons Used in This Book	3
Beyond the Book	4
Where to Go from Here	4
PART 1: SUPERCHARGED REPORTING WITH POWER PIVOT	5
CHAPTER 1: Thinking Like a Database	7
Exploring the Limits of Excel and How Databases Help	7
Scalability	8
Transparency of analytical processes	9
Separation of data and presentation	10
Getting to Know Database Terminology	11
Databases	11
Tables	11
Records, fields, and values	12
Queries	13
Understanding Relationships	13
CHAPTER 2: Introducing Power Pivot	17
Understanding the Power Pivot Internal Data Model	18
Linking Excel Tables to Power Pivot	20
Preparing Excel tables	21
Adding Excel Tables to the data model	22
Creating relationships between Power Pivot tables	24
Managing existing relationships	26
Using the Power Pivot data model in reporting	27
CHAPTER 3: The Pivotal Pivot Table	29
Introducing the Pivot Table	30
Defining the Four Areas of a Pivot Table	30
Values area	30
Row area	31
Column area	31
Filter area	32
Creating Your First Pivot Table	33
Changing and rearranging a pivot table	36
Adding a report filter	37
Keeping the pivot table fresh	38

Customizing Pivot Table Reports	40
Changing the pivot table layout.	40
Customizing field names	41
Applying numeric formats to data fields	42
Changing summary calculations	43
Suppressing subtotals	44
Showing and hiding data items	47
Hiding or showing items without data	49
Sorting the pivot table	51
Understanding Slicers.	52
Creating a Standard Slicer	54
Getting Fancy with Slicer Customizations	56
Size and placement	56
Data item columns	57
Miscellaneous slicer settings	58
Controlling Multiple Pivot Tables with One Slicer.	58
Creating a Timeline Slicer.	59
CHAPTER 4: Using External Data with Power Pivot	63
Loading Data from Relational Databases	64
Loading data from SQL Server.	64
Loading data from Microsoft Access databases	70
Loading data from other relational database systems	72
Loading Data from Flat Files	75
Loading data from external Excel files	76
Loading data from text files	78
Loading data from the Clipboard	81
Loading Data from Other Data Sources.	82
Refreshing and Managing External Data Connections	83
Manually refreshing Power Pivot data	83
Setting up automatic refreshing	84
Preventing Refresh All.	85
Editing the data connection	86
CHAPTER 5: Working Directly with the Internal Data Model	89
Directly Feeding the Internal Data Model	89
Managing Relationships in the Internal Data Model	95
Managing Queries and Connections	96
Creating a New Pivot Table Using the Internal Data Model	97
Filling the Internal Data Model with Multiple External Data Tables.	98
CHAPTER 6: Adding Formulas to Power Pivot.	103
Enhancing Power Pivot Data with Calculated Columns.	103
Creating your first calculated column	104
Formatting calculated columns.	105

	Referencing calculated columns in other calculations.	106
	Hiding calculated columns from end users.	107
	Utilizing DAX to Create Calculated Columns.	108
	Identifying DAX functions that are safe for calculated columns.	108
	Building DAX-driven calculated columns.	110
	Month sorting in Power Pivot–driven pivot tables.	112
	Referencing fields from other tables.	113
	Nesting functions.	115
	Understanding Calculated Measures.	116
	Creating a calculated measure.	116
	Editing and deleting calculated measures.	118
	Free Your Data with Cube Functions.	119
CHAPTER 7:	Diving into DAX.	121
	DAX Language Fundamentals.	121
	Using DAX operators.	125
	Applying conditional logic in DAX.	126
	Working with DAX aggregate functions.	128
	Exploring iterator functions and row context.	129
	Understanding Filter Context.	133
	Getting context transitions with the CALCULATE function.	135
	Adding flexibility with the FILTER function.	137
	PART 2: WRANGLING DATA WITH POWER QUERY.	141
CHAPTER 8:	Introducing Power Query.	143
	Power Query Basics.	144
	Starting the query.	144
	Understanding query steps.	150
	Refreshing Power Query data.	152
	Managing existing queries.	153
	Understanding Column-Level Actions.	155
	Understanding Table Actions.	157
CHAPTER 9:	Power Query Connection Types.	159
	Importing Data from Files.	160
	Getting data from Excel workbooks.	160
	Getting data from CSV and text files.	161
	Getting data from PDF files.	163
	Getting data from folders.	164
	Importing Data from Database Systems.	165
	A connection for every database type.	165
	Getting data from other data systems.	167
	Walk-through: Getting data from a database.	168

Managing Data Source Settings	170
Data Profiling with Power Query	171
Data Profiling options	172
Data Profiling quick actions	173
CHAPTER 10: Transforming Your Way to Better Data	175
Completing Common Transformation Tasks	176
Removing duplicate records	176
Filling in blank fields	178
Concatenating columns	179
Changing case	181
Finding and replacing specific text	181
Trimming and cleaning text	183
Extracting the left, right, and middle values	184
Splitting columns using character markers	187
Pivoting and unpivoting fields	189
Creating Custom Columns	193
Concatenating with a custom column	195
Understanding data type conversions	196
Spicing up custom columns with functions	197
Adding conditional logic to custom columns	199
Grouping and Aggregating Data	201
Working with Custom Data Types	203
CHAPTER 11: Making Queries Work Together	207
Reusing Query Steps	208
Understanding the Append Feature	211
Creating the needed base queries	212
Appending the data	213
Understanding the Merge Feature	216
Understanding Power Query joins	216
Merging queries	217
Understanding Fuzzy Match	221
CHAPTER 12: Extending Power Query with Custom Functions	225
Creating and Using a Basic Custom Function	225
Creating a Function to Merge Data from Multiple Excel Files	229
Creating Parameter Queries	236
Preparing for a parameter query	236
Creating the base query	238
Creating the parameter query	239

PART 3: THE PART OF TENS	243
CHAPTER 13: Ten Ways to Improve Power Pivot Performance	245
Limit the Number of Rows and Columns in Your Data Model Tables.....	246
Use Views Instead of Tables.....	246
Avoid Multi-Level Relationships.....	246
Let the Back-End Database Servers Do the Crunching.....	247
Beware of Columns with Many Unique Values.....	248
Limit the Number of Slicers in a Report.....	248
Create Slicers Only on Dimension Fields.....	249
Disable the Cross-Filter Behavior for Certain Slicers.....	250
Use Calculated Measures Instead of Calculated Columns.....	250
Upgrade to 64-Bit Excel.....	251
CHAPTER 14: Ten Tips for Working with Power Query	253
Getting Quick Information from the Queries & Connections Pane.....	253
Organizing Queries in Groups.....	254
Selecting Columns in Queries Faster.....	255
Renaming Query Steps.....	256
Quickly Creating Reference Tables.....	257
Viewing Query Dependencies.....	258
Setting a Default Load Behavior.....	259
Preventing Automatic Data Type Changes.....	259
Disabling Privacy Settings to Improve Performance.....	261
Disabling Relationship Detection.....	261
INDEX	263

Introduction

Over the past few years, the concept of self-service business intelligence (BI) has taken over the corporate world. Self-service BI is a form of business intelligence in which end users can independently generate their own reports, run their own queries, and conduct their own analyses, without the need to engage the IT department.

The demand for self-service BI is a direct result of several factors:

- » **More power users:** Organizations are realizing that no single enterprise reporting system or BI tool can accommodate all their users. Predefined reports and high-level dashboards may be sufficient for casual users, but a large portion of today's users are savvy enough to be considered power users. Power users have a greater understanding of data analysis and prefer to perform their own analysis, often within Excel.
- » **Changing analytical needs:** In the past, business intelligence primarily consisted of IT-managed dashboards showing historic data on an agreed-upon set of key performance metrics. Managers now demand more dynamic predictive analysis, the ability to perform data discovery iteratively, and the freedom to take the hard left and right turns on data presentation. These managers often turn to Excel to provide the needed analytics and visualization tools.
- » **Speed of BI:** Users are increasingly dissatisfied with the inability of IT to quickly deliver new reporting and metrics. Most traditional BI implementations fail specifically because the need for changes and answers to new questions overwhelmingly outpaces the IT department's ability to deliver them. As a result, users often find ways to work around the perceived IT bottleneck and ultimately build their own shadow BI (under the radar) solutions in Excel.

Recognizing the importance of the self-service BI revolution and the role Excel plays in it, Microsoft has made substantial investments in making Excel a player in the self-service BI arena by embedding both Power Pivot and Power Query directly into Excel.

You can integrate multiple data sources, define relationships between data sources, process analysis services cubes, and develop interactive dashboards that can be shared on the web. Indeed, the new Microsoft BI tools blur the line between Excel analysis and what is traditionally IT enterprise-level data management and reporting capabilities.

With these new tools in the Excel wheelhouse, it's becoming important for business analysts to expand their skill sets to new territory, including database management, query design, data integration, multidimensional reporting, and a host of other skills. Excel analysts have to expand their skill set knowledge base from the one-dimensional spreadsheets to relational databases, data integration, and multidimensional reporting.

That's where this book comes in. Here, you're introduced to the mysterious world of Power Pivot and Power Query. You find out how to leverage the rich set of tools and reporting capabilities to save time, automate data clean-up, and substantially enhance your data analysis and reporting capabilities.

About This Book

The goal of this book is to give you a solid overview of the self-service BI functionality offered by Power Pivot and Power Query. Each chapter guides you through practical techniques that enable you to

- » Extract data from databases and external files for use in Excel reporting
- » Scrape and import data from the web
- » Build automated processes to clean and transform data
- » Easily slice data into various views on the fly, gaining visibility from different perspectives
- » Analyze large amounts of data and report them in a meaningful way
- » Create powerful, interactive reporting mechanisms and dashboards

Within this book, you may note that some web addresses break across two lines of text. If you're reading this book in print and want to visit one of these web pages, simply key in the web address exactly as it's noted in the text, pretending as though the line break doesn't exist. If you're reading this as an e-book, you've got it easy — just click the web address to be taken directly to the web page.

Foolish Assumptions

Over the past few years, Microsoft has adopted an agile release cycle, allowing the company to release updates to Microsoft Office and the power BI tools practically monthly. This is great news for those who love seeing new features added to Power Pivot and Power Query. (It's not-so-great news if you're trying to document the features of these tools in a book.)

My assumption is that Microsoft will continue to add new bells and whistles to Power Pivot and Power Query at a rapid pace after publication of this book. So you may encounter new functionality not covered here.

The good news is that both Power Pivot and Power Query have stabilized and already have a broad feature set. So I'm also assuming that although changes will be made to these tools, they won't be so drastic as to turn this book into a door-stop. The core functionality covered in these chapters will remain relevant — even if the mechanics change a bit.

Icons Used in This Book

As you look in various places in this book, you see icons in the margins that indicate material of interest (or not, as the case may be). This section briefly describes each icon in this book.



TIP

Tips are beneficial because they help you save time or perform a task without having to do a lot of extra work. The tips in this book are time-saving techniques or pointers to resources that you should check out to get the maximum benefit from Excel.



WARNING

Try to avoid doing anything marked with a Warning icon, which (as you might expect) represents a danger of one sort or another.



TECHNICAL
STUFF

Whenever you see this icon, think *advanced* tip or technique. You might find these tidbits of useful information just too boring for words, or they could contain the solution you need to get a program running. Skip these bits of information whenever you like.



REMEMBER

If you get nothing else out of a particular chapter or section, remember the material marked by this icon. This text usually contains an essential process or a bit of information you ought to remember.



ON THE
WEB

Paragraphs marked with this icon reference the sample files for the book.

Beyond the Book

In addition to the book you have in your hands, you can access some extra content online. Check out the free Cheat Sheet for lists of Power Query text functions and Power Query date functions that are good to know. Just go to www.dummies.com and type **Excel Power Pivot & Power Query For Dummies Cheat Sheet** in the Search box.

If you want to follow along with the examples in this book, you can download the sample files at www.dummies.com/go/excelpowerpivotpowerqueryfd2e. The files are organized by chapter.

Where to Go from Here

It's time to start your self-service BI adventure! If you're primarily interested in Power Pivot, start with Chapter 1. If you want to dive right into Power Query, jump to Part 2, which begins at Chapter 8.

1 Supercharged Reporting with Power Pivot

IN THIS PART . . .

Think about data like a relational database.

Create your own Power Pivot data model.

Explore the workings of pivot tables.

Use external data with Power Pivot.

Manage the Power Pivot internal data model.

Create your own formulas in Power Pivot.

Delve deeper into the DAX formula language.

- » Examining traditional Excel limitations
- » Keeping up with database terminology
- » Looking into relationships

Chapter **1**

Thinking Like a Database

With the introduction of business intelligence (BI) tools such as Power Pivot and Power Query, it's becoming increasingly important for Excel analysts to understand core database principles. Unlike traditional Excel concepts, where the approach to developing solutions is relatively intuitive, you need to have a basic understanding of database terminology and architecture in order to get the most benefit from Power Pivot and Power Query. This chapter introduces you to a handful of fundamental concepts that you should know before taking on the rest of this book.

Exploring the Limits of Excel and How Databases Help

Years of consulting experience have brought this humble author face to face with managers, accountants, and analysts who all have had to accept this simple fact: Their analytical needs had outgrown Excel. They all faced fundamental challenges that stemmed from one or more of Excel's three problem areas: scalability, transparency of analytical processes, and separation of data and presentation.

Scalability

Scalability is the ability of an application to develop flexibly to meet growth and complexity requirements. In the context of this chapter, scalability refers to Excel's ability to handle ever-increasing volumes of data.

Imagine that you're working in a small company and using Excel to analyze its daily transactions. As time goes on, you build a robust process complete with all the formulas, pivot tables, and macros you need in order to analyze the data that is stored in your neatly maintained worksheet.

As the amount of data grows, you will first notice performance issues. The spreadsheet will become slow to load and then slow to calculate. Why does this happen? It has to do with the way Excel handles memory. When an Excel file is loaded, the entire file is loaded into RAM. Excel does this to allow for quick data processing and access. The drawback to this behavior is that every time the data in your spreadsheet changes, Excel has to reload the entire document into RAM. The net result in a large spreadsheet is that it takes a great deal of RAM to process even the smallest change. Eventually, every action you take in the gigantic worksheet is preceded by an excruciating wait.

Your pivot tables will require bigger pivot caches, almost doubling the Excel workbook's file size. Eventually, the workbook will become too big to distribute easily. You may even consider breaking down the workbook into smaller workbooks (possibly one for each region). This causes you to duplicate your work.

In time, you may eventually reach the 1,048,576-row limit of the worksheet. What happens then? Do you start a new worksheet? How do you analyze two datasets on two different worksheets as one entity? Are your formulas still good? Will you have to write new macros?

These are all issues that need to be addressed.

Of course, you will also encounter the Excel power customers, who will find various clever ways to work around these limitations. In the end, though, these methods will always be simply workarounds. Eventually, even these power customers will begin to think less about the most effective way to perform and present analysis of their data and more about how to make data "fit" into Excel without breaking their formulas and functions. Excel is flexible enough that a proficient customer can make most things fit just fine. However, when customers think only in terms of Excel, they're undoubtedly limiting themselves, albeit in an incredibly functional way.

In addition, these capacity limitations often force Excel customers to have the data prepared for them. That is, someone else extracts large chunks of data from

a large database and then aggregates and shapes the data for use in Excel. Should the serious analyst always be dependent on someone else for their data needs? What if an analyst could be given the tools to access vast quantities of data without being reliant on others to provide data? Could that analyst be more valuable to the organization? Could that analyst focus on the accuracy of the analysis and the quality of the presentation instead of routine Excel data maintenance?

A relational database system (such as Access or SQL Server) is a logical next step for the analyst who faces an ever-increasing data pool. Database systems don't usually have performance implications with large amounts of stored data, and are built to address large volumes of data. An analyst can then handle larger datasets without requiring the data to be summarized or prepared to fit into Excel. Also, if a process ever becomes more crucial to the organization and needs to be tracked in a more enterprise-acceptable environment, it will be easier to upgrade and scale up if that process is already in a relational database system.

Transparency of analytical processes

One of Excel's most attractive features is its flexibility. Each individual cell can contain text, a number, a formula, or practically anything else the customer defines. Indeed, this is one of the fundamental reasons that Excel is an effective tool for data analysis. Customers can use named ranges, formulas, and macros to create an intricate system of interlocking calculations, linked cells, and formatted summaries that work together to create a final analysis.

So what is the problem? The problem is that there is no transparency of analytical processes. It is extremely difficult to determine what is actually going on in a spreadsheet. Anyone who has had to work with a spreadsheet created by someone else knows all too well the frustration that comes with deciphering the various gyrations of calculations and links being used to perform analysis. Small spreadsheets that are performing modest analysis are painful to decipher, and large, elaborate, multi-worksheet workbooks are virtually impossible to decode, often leaving you to start from scratch.

Compared to Excel, database systems might seem rigid, strict, and unwavering in their rules. However, all this rigidity comes with a benefit.

Because only certain actions are allowable, you can more easily come to understand what is being done within structured database objects such as queries or stored procedures. If a dataset is being edited, a number is being calculated, or any portion of the dataset is being affected as part of an analytical process, you can readily see that action by reviewing the query syntax or the stored procedure code. Indeed, in a relational database system, you never encounter hidden formulas, hidden cells, or dead named ranges.

Separation of data and presentation

Data should be separate from presentation; you don't want the data to become too tied into any particular way of presenting it. For example, when you receive an invoice from a company, you don't assume that the financial data on that invoice is the true source of your data. It is *a presentation of your data*. It can be presented to you in other manners and styles on charts or on websites, but such representations are never the actual source of the data.

What exactly does this concept have to do with Excel? People who perform data analysis with Excel tend, more often than not, to fuse the data, the analysis, and the presentation. For example, you often see an Excel workbook that has 12 worksheets, each representing a month. On each worksheet, data for that month is listed along with formulas, pivot tables, and summaries. What happens when you're asked to provide a summary by quarter? Do you add more formulas and worksheets to consolidate the data on each of the month worksheets? The fundamental problem in this scenario is that the worksheets actually represent data values that are fused into the presentation of the analysis.

The point being made here is that data should not be tied to a particular presentation, no matter how apparently logical or useful it may be. However, in Excel, it happens all the time.

In addition, as discussed earlier in this chapter, because all manners and phases of analysis can be done directly within a spreadsheet, Excel cannot effectively provide adequate transparency to the analysis. Each cell has the potential to hold formulas, be hidden, and contain links to other cells. In Excel, this blurs the line between analysis and data, which makes it difficult to determine exactly what is going on in a spreadsheet. Moreover, it takes a great deal of effort in the way of manual maintenance to ensure that edits and unforeseen changes don't affect previous analyses.

Relational database systems inherently separate analytical components into tables, queries, and reports. By separating these elements, databases make data less sensitive to changes and create a data analysis environment in which you can easily respond to new requests for analysis without destroying previous analyses.

You may find that you manipulate Excel's functionalities to approximate this database behavior. If so, you must consider that if you're using Excel's functionality to make it behave like a database application, perhaps the real thing just might have something to offer. Utilizing databases for data storage and analytical needs would enhance overall data analysis and would allow Excel power customers to focus on the presentation in their spreadsheets.

In these days of big data, customers demand more, not less, complex data analysis. Excel analysts will need to add tools to their repertoires to avoid being simply “spreadsheet mechanics.” Excel can be stretched to do just about anything, but maintaining such creative solutions can be a tedious manual task. You can be sure that the sexy aspect of data analysis does not lie in the routine data management within Excel; rather, it lies in leveraging BI Tools such as providing clients with the best solution for any situation.

Getting to Know Database Terminology

The terms *database*, *table*, *record*, *field*, and *value* indicate a hierarchy from largest to smallest. These same terms are used with virtually all database systems, so you should learn them well.

Databases

Generally, the word *database* is a computer term for a collection of information concerning a certain topic or business application. A database helps you organize this related information in a logical fashion for easy access and retrieval. Certain older database systems used the term *database* to describe individual tables. The current use of *database* applies to all elements of a database system.

Databases aren’t only for computers. Manual databases are sometimes referred to as manual filing systems or manual database systems. These filing systems usually consist of people, papers, folders, and filing cabinets — paper is the key to a manual database system. In a real-life manual database system, you probably have in-baskets and out-baskets and some type of formal filing method. You access information manually by opening a file cabinet, removing a file folder, and finding the correct piece of paper. Customers fill out paper forms for input, perhaps by using a keyboard to input information that is printed on forms. You find information by manually sorting the papers or by copying information from many papers to another piece of paper (or even into an Excel spreadsheet). You may use a spreadsheet or calculator to analyze the data or display it in new and interesting ways.

Tables

A database stores information in a carefully defined structure known as a table. A *table* is just a container for raw information (called *data*), similar to a folder in a manual filing system. Each table in a database contains information about a single

entity, such as a person or product, and the data in the table is organized into rows and columns. A relational database system stores data in related tables. For example, a table containing employee data (names and addresses) may be related to a table containing payroll information (pay date, pay amount, and check number).

To use database wording, a table is an object. As you design and work with databases, it's important to see each table as a unique entity and to see how each table relates to the other objects in the database.

In most database systems, you can view the contents of a table in a spreadsheet-like form called a *datasheet*, composed of rows and columns (known as *records* and *fields*, respectively — see the following section). Although a datasheet and a spreadsheet are superficially similar, a datasheet is quite a different type of object. You typically cannot make changes or add calculations directly within a table. Your interaction with tables will primarily come in the form of queries or views — see the later section “Queries”.

Records, fields, and values

A database table is divided into rows (called *records*) and columns (called *fields*), with the first row (the heading on top of each column) containing the names of the fields in the database.

Each row is a single record containing fields that are related to that record. In a manual system, the rows are individual forms (sheets of paper), and the fields are equivalent to the blank areas on a printed form that you fill in.

Each column is a field that includes many properties specifying the type of data contained within the field and how the database should handle the field's data. These properties include the name of the field (Company) and the type of data in the field (Text). A field may include other properties as well. For example, the Address field's Size property tells the database the maximum number of characters allowed for the address.

At the intersection of a record and a field is a *value* — the actual data element. For example, in a field named Company, a company name entered into that field would represent one data value.



REMEMBER

When working with Microsoft Access, the term *field* is used to refer to an attribute stored in a record. In many other database systems, including SQL Server, *column* is the expression you hear most often in place of *field* — field and column mean the same thing. The exact terminology that's used relies somewhat on the context of the database system underlying the table containing the record.

Queries

Most relational database systems allow the creation of queries (sometimes called views). A query extracts information from the tables in the database; a query selects and defines a group of records that fulfill a certain condition. Most database outputs are based on queries that combine, filter, or sort data before it's displayed. Queries are often called from other database objects, such as stored procedures, macros, or code modules. In addition to extracting data from tables, queries can be used to change, add, or delete database records.

An example of a query is when a person at the sales office tells the database, "Show me all customers, in alphabetical order by name, who are located in Massachusetts and who made a purchase over the past six months." Or "Show me all customers who bought Chevrolet car models within the past six months, and display them sorted by customer name and then by sale date."

Rather than ask the question using English words, a person uses a special syntax, such as Structured Query Language (or SQL), to communicate to the database what the query will need to do.

Understanding Relationships

After you understand the basic terminology of databases, it's time to focus on one of their more useful features: A *relationship* is the mechanism by which separate tables are related to each other. You can think of a relationship as a kind of VLOOKUP, in which you relate the data in one data range to the data in another data range using an index or a unique identifier. In databases, relationships do the same thing, but without the hassle of writing formulas.

Relationships are important because most of the data you work with fits into a multidimensional hierarchy of sorts. For example, you may have a table showing customers who buy products. These customers require invoices that have invoice numbers. Those invoices have multiple lines of transactions listing what they bought. A hierarchy exists there.

Now, in the one-dimensional spreadsheet world, this data typically would be stored in a flat table, like the one shown in Figure 1-1.

Because customers have more than one invoice, the customer information (in this example, CustomerID and CustomerName) has to be repeated. This causes a problem when that data needs to be updated.

FIGURE 1-1:
Data is stored
in an Excel
spreadsheet
using a flat-table
format.

	A	B	C	D	E	F
1	CustomerID	CustomerName	InvoiceNumber	InvoiceDate	Quantity	UnitPrice
2	BAKERSEM0001	Baker's Emporium Inc.	ORDST1025	5/8/2005	1	19.95
3	BAKERSEM0001	Baker's Emporium Inc.	ORDST1025	5/8/2005	5	1759.95
4	BAKERSEM0001	Baker's Emporium Inc.	ORDST1025	5/8/2005	4	9.95
5	BAKERSEM0001	Baker's Emporium Inc.	STDINV2251	4/12/2007	4	9.95
6	AARONFIT0001	Aaron Fitz Electrical	ORDST1026	5/8/2005	5	9.95
7	AARONFIT0001	Aaron Fitz Electrical	ORDST1026	5/8/2005	3	1759.95
8	AARONFIT0001	Aaron Fitz Electrical	ORDST1026	5/8/2005	2	79.95
9	AARONFIT0001	Aaron Fitz Electrical	STDINV2252	4/12/2007	3	1759.95
10	AARONFIT0001	Aaron Fitz Electrical	STDINV2252	4/12/2007	5	9.95
11	METROPOL0001	Metropolitan Fiber Systems	ORD1002	5/7/2004	1	9.95
12	AARONFIT0001	Aaron Fitz Electrical	INV1024	2/10/2004	1	119.95
13	AARONFIT0001	Aaron Fitz Electrical	INV1025	2/15/2004	1	109.95
14	LECLERC0001	LeClerc & Associates	ORDPH1005	5/10/2004	2	189.95
15	MAGNIFIC0001	Magnificent Office Images	ORD1000	5/8/2004	1	359.95
16	HOLLINGC0001	Holling Communications Inc.	ORD1001	5/10/2004	2	59.95
17	MAHLERST0001	Mabler State University	ORDST1008	5/10/2004	1	5999.95

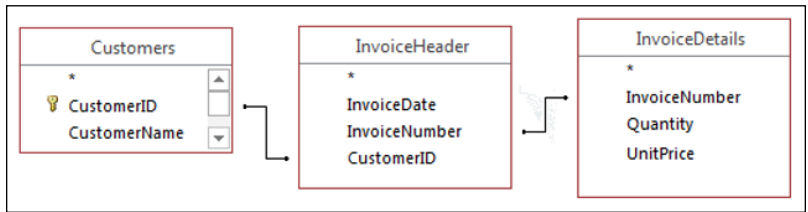
For example, imagine that the name of the company Aaron Fitz Electrical changes to Fitz and Sons Electrical. Looking at Figure 1-1, you see that multiple rows contain the old name. You would have to ensure that every row containing the old company name is updated to reflect the change. Any rows you miss will not correctly map back to the right customer.

Wouldn't it be more logical and efficient to record the name and information of the customer only one time? Then, rather than have to write the same customer information repeatedly, you could simply have some form of customer reference number.

This is the idea behind relationships. You can separate customers from invoices, placing each in their own tables. Then you can use a unique identifier (such as CustomerID) to relate them together.

Figure 1-2 illustrates how this data would look in a relational database. The data would be split into three separate tables: Customers, InvoiceHeader, and Invoice-Details. Each table would then be related using unique identifiers (CustomerID and InvoiceNumber, in this case).

FIGURE 1-2:
Databases use
relationships to
store data in
unique tables and
simply relate
these tables to
each other.



The Customers table would contain a unique record for each customer. That way, if you need to change a customer's name, you would need to make the change in only that record. Of course, in real life, the Customers table would include other attributes, such as customer address, customer phone number, and customer start date. Any of these other attributes could also be easily stored and managed in the Customers table.

The most common relationship type is a *one-to-many* relationship. That is, for each record in one table, one record can be matched to many records in a separate table. For example, an invoice header table is related to an invoice detail table. The invoice header table has a unique identifier: Invoice Number. The invoice detail will use the Invoice Number for every record representing a detail of that particular invoice.

Another kind of relationship type is the *one-to-one* relationship: For each record in one table, one and only one matching record is in a different table. Data from different tables in a one-to-one relationship can technically be combined into a single table.

Finally, in a *many-to-many* relationship, records in both tables can have any number of matching records in the other table. For instance, a database at a bank may have a table of the various types of loans (home loan, car loan, and so on) and a table of customers. A customer can have many types of loans. Meanwhile, each type of loan can be granted to many customers.

If your head is spinning from all this database talk, don't worry. You don't need to be an expert database modeler to use Power Pivot. But it's important to understand these concepts. The better you understand how data is stored and managed in databases, the more effectively you'll leverage Power Pivot for reporting.

- » Getting to know the Internal Data Model
- » Activating the Power Pivot add-in
- » Linking to Excel data
- » Managing relationships

Chapter 2

Introducing Power Pivot

Over the past decade or so, corporate managers, eager to turn impossible amounts of data into useful information, drove the business intelligence (BI) industry to innovate new ways of synthesizing data into meaningful insights. During this period, organizations spent lots of time and money implementing big enterprise reporting systems to help keep up with the hunger for data analytics and dashboards.

Recognizing the importance of the BI revolution and the place that Excel holds within it, Microsoft proceeded to make substantial investments in improving Excel's BI capabilities. It specifically focused on Excel's *self-service* BI capabilities and its ability to better manage and analyze information from the increasing number of available data sources.

The key product of that endeavor was essentially Power Pivot (introduced in Excel 2010 as an add-in). With Power Pivot came the ability to set up relationships between large, disparate data sources. For the first time, Excel analysts were able to add a relational view to their reporting without the use of problematic functions such as VLOOKUPS. The ability to merge data sources with hundreds of thousands of rows into one analytical engine within Excel was groundbreaking.

With the release of Excel 2016, Microsoft incorporated Power Pivot directly into Excel. The powerful capabilities of Power Pivot are available out of the box!

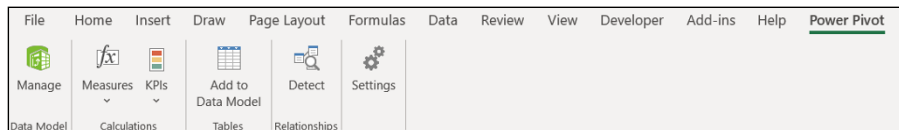
In this chapter, you get an overview of those capabilities by exploring the key features, benefits, and capabilities of Power Pivot.

Understanding the Power Pivot Internal Data Model

At its core, Power Pivot is essentially a SQL Server Analysis Services engine made available by way of an in-memory process that runs directly within Excel. Its technical name is the xVelocity analytics engine. However, in Excel, it's referred to as the Internal Data Model.

Every Excel workbook contains an *Internal Data Model*, a single instance of the Power Pivot in-memory engine. The most effective way to interact with the Internal Data Model is to use the Power Pivot Ribbon interface (see Figure 2-1).

FIGURE 2-1:
The Power Pivot
Ribbon interface.



The Power Pivot Ribbon interface exposes the full set of functionalities you don't get with the standard Excel Data tab. Here are a few examples of functionality available with the Power Pivot interface:

- » You can browse, edit, filter, and apply custom sorting to data.
- » You can create custom calculated columns that apply to all rows in the data import.
- » You can define a default number format to use when the field appears in a pivot table.
- » You can easily configure relationships via the handy Graphical Diagram view.
- » You can choose to prevent certain fields from appearing in the PivotTable Field List.

As with everything else in Excel, the Internal Data Model does have limitations. Most Excel users will not likely hit these limitations, because Power Pivot's compression algorithm is typically able to shrink imported data to about one-tenth its original size. For example, a 100MB text file would take up only approximately 10MB in the Internal Data Model.

Nevertheless, it's important to understand the maximum and configurable limits for Power Pivot Data Models. Table 2-1 highlights them.