

K2's NEXT GENERATION EXCEL REPORTING

CONTINUING PROFESSIONAL EDUCATION SERIES



To access the demonstration files used in this seminar, please visit <https://cpe.cx/download> and use the exact code **23s-nge** to access the materials. Please note that the code is case-sensitive.

K2 ENTERPRISES
WWW.K2E.COM

K2E CANADA INC.
WWW.K2E.CA

Copyright © 2025, by K2 Enterprises and K2E Canada Inc.

All rights reserved. No part of this publication may be reproduced or transmitted in any form without the express written consent of:

K2 Enterprises
1905 W. Thomas St., Suite D #363
Hammond, LA 70401-2901
985.542.9390

K2E Canada Inc.
174 Brewer Court
Burlington, Ontario, L7L4G4
289-427-5890

You may e-mail requests to reproduce or transmit to info@k2e.com or caroline@k2e.ca. You may also obtain additional information on the web at www.k2e.com or www.k2e.ca.

The use of trade names and trademarks in these materials does not convey endorsement of these vendors or products. All trade names and trademarks used in these materials are the property of their respective owners. Any abbreviations used herein are solely for the reader's convenience and do not constitute any intent to compromise trademarks.

The statements in these materials about specific companies or specific hardware and software products should in no way be misconstrued as statements of fact but rather are opinions of the authors and K2 Enterprises and K2 Canada Inc. We take great care in providing views that we think will be useful in helping our participants make informed decisions. K2 Enterprises and K2 Canada Inc. reserve the right to independently evaluate companies' merits and the hardware and software products referenced in our seminars and conference presentations.

K2 Enterprises and K2 Canada Inc. make no representations or warranty regarding the contents of these materials and disclaim any implied warranties of merchantability or fitness for any particular purpose. The contents of these materials are subject to change without notice.

CONTRIBUTORS: The shareholders, associates, and staff of K2 Enterprises and K2 Canada Inc. contributed to producing these materials.

Table of Contents

Introduction	IV
Three Rules of this Seminar	IV
Course Development	IV
About K2 Enterprises	IV
Our Mission Statement	V
Our Instructional Team	V
The K2 Seminar Curriculum	V
On-Site Training	VII
Web-Based Training.....	VIII
K2 Internet Sites.....	VIII
Chapter 1 – What Is Next Generation Excel Reporting?.....	1
Learning Objectives.....	1
Reporting Challenges with Excel.....	1
Five Features for Better Excel-based Reporting	3
Data Models.....	5
Power Query	6
Power Pivot	6
Data Analysis Expressions.....	7
Slicers and Timelines.....	7
How the Five Tools Work Together	8
Is This Always the Best Approach?	9
Chapter 2 – Tables and Data Models – Cornerstones of Modern Reporting Practices.....	11
Learning Objectives.....	11
Excel Table Fundamentals	11
Four Advantages of Working with Tables.....	12
Creating Tables	12
Re-sizing Tables Manually	18
What Is a Data Model?	19
Advantages of the Data Model	21
Disadvantages of the Data Model	22
Creating Simple Data Models from Excel Tables	22

Data Model Relationships	27
Simple Relationships in Data Models	28
A Summary Example of Simple Data Models	29
Chapter 3 – Accessing and Transforming Data with Power Query.....	35
Learning Objectives.....	35
Introducing Power Query.....	35
Building and Adding to Data Models with Power Query.....	37
Querying Data from SQL Server	39
Editing a Query.....	42
Linking Data from the Web Using Power Query.....	44
Adding a Simple Transformation to a Query	47
Examples of Useful Power Query Transformations.....	48
Pivoting Data with Power Query.....	48
Unpivoting Data	51
Splitting Data into Multiple Columns with Power Query	53
Transforming Data by Adding Columns	55
Using Power Query to Group Data by Fiscal Periods.....	57
Creating a Transformation to Transpose Data	59
Combining Excel Worksheets with a Power Query Transformation	60
Querying All Data from a Folder	62
Preparing Consolidated Financial Statements with Power Query	64
Chapter 4 – Managing Your Data Models with Power Pivot	67
Learning Objectives.....	67
Introducing Power Pivot	67
Why Power Pivot?.....	67
Creating PivotTables from Power Pivot-based Data Models.....	68
Using Power Pivot to Create a PivotTable from Multiple Data Sources.....	75
Managing Data and Relationships in a Data Model	79
Using Perspectives to Control the Amount of Data Available	82
Chapter 5 – Building DAX Functions And Cube Formulas.....	87
Learning Objectives.....	87
Building Formulas with Data Analysis Expressions	87
Creating Simple Calculated Columns with DAX	89

Creating Measures in a PivotTable Using DAX.....	92
Creating Key Performance Indicators in Power Pivot.....	94
A Deeper Dive into DAX	96
Calendar Tables – The Key to Period Reporting and Time Intelligence.....	100
Automatic Creation of Calendar Tables.....	102
Using CUBE Functions to Summarize Transactions from Data Models.....	103
Excel’s Seven CUBE Functions.....	104
Chapter 6 – Slicer and Timeline Filters for Improved End-User Experiences.....	107
Learning Objectives.....	107
Easy (and Advanced) Filtering with Slicers and Timelines.....	107
Adding and Working with Slicers	107
Adding and Working with Timelines.....	108
Advanced Functionality Associated with Slicers and Timelines	109

Introduction

Three Rules of this Seminar

1. **There Are No Dumb Questions.** Please ask questions if you need further explanation. We welcome your questions and will do our best to provide accurate and meaningful answers.
2. **Obtain the Information for Which You Came.** If you have a specific interest in a topic, please let us know. If your topic is not part of the course materials, we will do our best to accommodate your request. Your instructor is anxious to meet your needs.
3. **Share and Share-Alike.** Your instructor is here to share all possible information related to the course. We hope that you will be willing to do the same. If you have helpful tips or experiences, please share them with the other participants and us.

Course Development

The shareholders, associates, and staff of K2 Enterprises and K2E Canada Inc. developed this course and the related course materials. We aim to deliver quality educational practices that help you better use and understand the tools and concepts discussed in this session.

We designed our course presentation style for busy professionals. Through numerous short case studies and feature reviews, we strive to deliver concise information in a context that is easy to understand and absorb.

To maximize the use of your time and the benefits of attending this course, we have worked hard to filter out trivial and obscure features. Our goal is not to cover every keystroke and menu option but to provide a fast-paced course focusing on state-of-the-art information technology and what it can do for you and your staff.

About K2 Enterprises

Over more than thirty-one years, our team members have collectively delivered more than 42,000 national and international technology-focused seminars, webinars, and conferences. To help us stay on top of technology, our team members read numerous technology trade magazines, attend many technology conferences, and talk with multiple technology companies yearly to obtain the latest information to include in our courses. Also, we experience everything we teach and recommend to our audiences. Because our shareholders and associates reside in different states and provinces, using technology efficiently and appropriately is necessary for our business's success and survival.

Our Mission Statement

K2 aims to produce and deliver the highest quality technology seminars and conferences to business professionals. We work cooperatively with professional organizations such as state CPA societies, associations of Chartered Professional Accountants, and vendors of technology products. We make every effort to maintain high integrity, family values, and friendship among all involved.

Our Instructional Team

For a complete listing of instructors and bios, please visit:

K2 Enterprises

[Instructors - K2 Enterprises](#)

K2E Canada Inc.

[Instructors - K2E Canada Inc](#)

The K2 Seminar Curriculum

Two-day Seminar

- K2's Excel Boot Camp

Full-day Seminars

- K2's Accountant's Guide to QuickBooks Online
- K2's Advanced Excel
- K2's Business Intelligence, Featuring Microsoft's Power BI Tools
- K2's Case Studies In Fraud And Technology Controls
- K2's Excel Best Practices
- K2's Excel Essentials For Staff Accountants
- K2's Excel PivotTables For Accountants
- K2's Excel Tips, Tricks, And Techniques For Accountants
- K2's Microsoft 365/Office 365 – All The Things You Need To Know
- K2's Next Generation Excel Reporting
- K2's Paperless Office
- K2's QuickBooks For Accountants
- K2's Small Business Internal Controls, Security, And Fraud Prevention And Detection
- K2's Technology For CPAs – Don't Get Left Behind

Half-day Seminars

- K2's Advanced QuickBooks Tips And Techniques
- K2's Artificial Intelligence For Accounting And Financial Professionals
- K2's Best Word, Outlook, And PowerPoint Features
- K2's Better Productivity Through Artificial Intelligence and Automation Tools

- K2's Case Studies In Fraud And Technology Controls
- K2's Data Analytics For Accountants And Auditors
- K2's Ethics And Technology
- K2's Excel Charting And Visualizations
- K2's Implementing Internal Controls In QuickBooks Environments
- K2's Improving Productivity With Microsoft 365/Office 365 Cloud Applications
- K2's Mastering Advanced Excel Functions
- K2's Microsoft Teams
- K2's Securing Your Data: Practical Tools For Protecting Information
- K2's Technology Update
- K2's Top PDF Features You Should Know
- K2's Working Remotely – The New Normal

Besides the seminars listed above, K2 also produces numerous one and two-day technology conferences annually. You can find a complete listing of our course offerings, including dates, locations, and course descriptions, on our website at:

K2 Enterprises

[Technology Conference - K2 Enterprises](#)



K2E Canada Inc.

[Technology Conference - K2E Canada Inc.](#)



On-Site Training

Working cooperatively with state CPA organizations and associations of Chartered Professional Accountants, we are pleased to offer all our seminars and conference sessions as on-site training opportunities. On-site training provides companies of all sizes with numerous advantages, including the following.

- **Customization.** K2 can customize the content of all on-site training events to meet your organization's specific needs. For example, we can combine selected parts of existing courses into a focused learning experience to meet your team's needs. Likewise, we can develop custom content from the ground up and deliver it to your team members, ensuring we meet your training objectives.
- **Cost savings.** For organizations of all sizes, on-site training can provide significant cost savings compared to public training venues. In addition to potentially reduced registration fees, these cost savings materialize in the form of reduced travel costs and reducing the amount of time team members spend away from the office.
- **Convenience.** Because you select your on-site training event's date, time, and location, on-site training is more convenient for you and your team than traditional training options. Many organizations schedule on-site training with staff meetings, retreats, and customer/client events to leverage the value of all participants' time.

Customization, cost savings, and convenience are the three C's of "training to go." For more information on how you and your organization can experience these benefits or schedule an on-site training event, please visit: [K2 Enterprises](#)

K2 Enterprises
[On-site Training - K2 Enterprises](#)

caroline@k2e.com



K2E Canada Inc.
[Seminars - K2E Canada Inc](#)

caroline@k2e.ca



Web-Based Training

K2 offers web-based CPE and CPD to accounting and business professionals throughout the United States and Canada. K2's award-winning instructors deliver two-, four-, and eight-hour webinars on numerous technology-focused topics.

Additionally, K2 provides on-demand training. On-demand courses offer you the advantage of receiving top-quality training at a time and location that fits your busy schedule. You can start a class today and complete it in the future. For more information, please visit:

For more information on either platform, please visit:

K2 Enterprises
[Webinars - K2 Enterprises](#)



K2E Canada Inc.
[Webinars - K2E Canada Inc](#)



K2 Internet Sites

K2 maintains multiple websites containing information and links relevant to CPAs and other business professionals as a service to the profession. Further, the content of these sites can enrich your experience during and after our seminars. In addition, our pages contain links to some of the top accounting and content management software solutions and even to a few sites that are just fun places to visit. If you have any comments or questions regarding our web pages, please email webmaster@k2e.com.

K2 has two primary websites. You can view information regarding CPE and CPD opportunities, access a library of technology tips and articles, and link to K2-related websites.

www.k2e.com



www.k2e.ca



www.cpafirmtech.com

CPA Firm Technology contains technology-related information explicitly targeted at public accounting firms. We include product reviews, hardware and operating system discussions, affinity programs for CPA firms, and Cloud computing resources.



www.accountingsoftwareworld.com

Accounting Software World provides information related to accounting and financial management solutions. Here, you can find software and service reviews, product comparisons, white papers, and other material of interest to those seeking information about accounting software and service solutions.



www.totallypaperless.com

For information about document management and paperless processes, please visit **Totally Paperless**. In addition, you will find reviews on hardware and software used by all types of businesses in document imaging and management processes.



What Is Next Generation Excel Reporting?

Chapter

1

New and improved features in Excel tools such as Power Query and Power Pivot mean that yesterday's techniques for generating reports in Excel are obsolete. For example, today's best practices for Next Generation Excel Reporting include using Power Query to link data into Excel and apply transformations. They also use Data Analysis Expressions to add user-defined calculations and work with Power Pivot to manage data models. Further, this concept reduces dependency on formulas, relies on PivotTables, and adds visualizations where appropriate.

The challenge to transitioning to this reporting methodology lies in understanding how to use the tools to manipulate and summarize your data. Specifically, how do we link multiple sets of external data into Excel, modify the data to meet our needs, and summarize the data into accurate and easy-to-understand reports? You will learn about Next Generation Excel Reporting fundamentals in this chapter. Further, you will also ascertain why it is a superior approach to an Excel-based reporting environment. Finally, with this foundational knowledge, you will be ready to re-engineer your reporting processes.

Learning Objectives

Upon completing this chapter, you should be able to:

- List some of the deficiencies associated with traditional Excel-based reporting processes;
- Identify the roles of tools such as Power Query, Power Pivot, and Data Analysis Expressions in creating next-generation Excel reports; and
- Cite examples of new functions in Excel that potentially can improve reporting processes.

Reporting Challenges with Excel

Business professionals have used Excel as one of their primary reporting tools for many years. In addition, they use Excel for ad hoc and financial and operational reports, which often require more structure and control.

One of the apparent reasons for the pervasive use of Excel is that it is already in place in virtually every organization; thus, no monetary investment is necessary to use it as a reporting tool. A second driver is an assumption – albeit suspect – that team members are well-versed in creating and working with spreadsheets. Unfortunately, the color-coded spreadsheets produced by some team members hide the fact they struggle to construct, maintain, and update these reports.

Historically, the backbone of most Excel-based operational and financial reports has been formulas – *seemingly countless formulas!* – based on various Excel functions. These formulas exist to summarize potentially large volumes of data and use numerous functions to perform many different tasks. In many cases, this model served businesses well in the past, perhaps because no practical alternative was available. However, numerous issues relating to efficiency, accuracy, and timeliness mean that yesterday’s practice is outdated by today’s standards, and we should consider replacing it. As proof, consider some of the following challenges of using traditional Excel reporting techniques.

- Formula-based spreadsheets are prone to error, and errors can be challenging to troubleshoot. The following quote from a recently published report provides ample evidence of this condition. *“Research shows that almost all spreadsheet modelers have no formal training. [M]ost errors do not arise from mistakes in programming the spreadsheet; rather, they arise from the misapplication of programming logic.[O]nce committed, logic errors are difficult to find and correct.”*¹
- Excel presents challenges in managing large volumes of data. With over 1 million rows and 16,000 columns available in a single worksheet, data volumes can snowball quickly in an era of Big Data. Traditional Excel-based reporting techniques often struggle to handle the volume of data present.
- Getting data into spreadsheets involves too many manual processes. Instead of manual copy-and-paste or export-then-import methods, we need the ability to link transactional data directly into Excel. Doing so will provide more timely and accurate reporting while reducing the time necessary to create and update reports.
- Summarizing data across multiple *dimensions* can be problematic in the absence of PivotTables. Creating formulas to summarize data across numerous categories can become unwieldy, especially when the dimensions are dynamic.
- Higher-level managers sometimes struggle to understand spreadsheet “logic.” This condition often leads to not trusting the numbers in the report and, in turn, usually precipitates additional manual calculations to prove the numbers correct.
- Spreadsheet-based data is often “stale” when generating reports. The pace at which business transacts today all but mandates real-time reporting with on-demand refresh cycles.

¹ “A Pilot Study Exploring Spreadsheet Risk in Scientific Research,” <https://arxiv.org/abs/1703.09785>, March 23, 2017

Perhaps the combined effect of the above issues is found in a white paper entitled “State of the CFO Role,” published recently.² The authors reported the following (with emphasis added).

*Today’s typical finance department is far from fully automated. **CFOs use spreadsheets for an average of 2.24 hours per day, accounting for the largest percentage of the average CFO’s day.** Many chiefs realize the need to improve these processes and are working on implementing better systems or optimizing the ones they have. In fact, better and faster reporting (55%), increased data visibility (41%), and implementing new financial software (39%) are all top priorities for CFOs in the next two years.*

Indeed, Excel presents other challenges, but the fundamental issue remains the same – while traditional, formula-based Excel reporting techniques were once leading-edge, it is now time to move into Next Generation Excel Reporting. Further, it may be advantageous to reduce dependency on Excel as a primary reporting tool in some cases. Instead, tools such as **Power BI** and **Tableau** could represent the future of financial and operational reporting.

Five Features for Better Excel-based Reporting

With Excel 2010 and the emergence of new *functions* such as **IFS**, **FORECAST.ETS**, and **XLOOKUP**, Microsoft incorporated several powerful new tools to enhance Excel’s viability as a reporting option. More specifically, these consist of the following.

- Data Models
- Power Query
- Power Pivot
- Data Analysis Expressions
- Slicers and Timelines

² <https://www.netsuite.com/portal/assets/pdf/wp-brainyard-state-of-the-cfo-role-in-2019.pdf>



Figure 1 - Next Generation Tools Yield Better Reporting Results

Before demonstrating how to use these tools in concert to realize better Excel-based reporting, let us first understand the purpose of each of these tools at a foundational level.

Data Models

Data models serve as “containers” of data within an Excel workbook. More specifically, a data model typically consists of two or more tables in a workbook related (linked) to each other based on a common field (column).³ Each table’s data can be “native” to the Excel workbook, entered manually, or pasted into the workbook. Alternatively, you can *link* it into Excel from external data sources – including multiple data sources. Further, it can be a combination of native data and linked data.

From a reporting perspective, the availability of data models is critical on at least two fronts. First, data models are more efficient when processing and storing data in Excel. For example, Microsoft reports that an Excel-based data model will be seven to ten times smaller than a SQL Server database containing the same data. Second, data models facilitate storing more records in Excel than would otherwise be possible. More to the point, a data model can handle up to 99 billion records, although no field can contain more than 1.99 billion unique records. Collectively, these two considerations mean that data models in Excel are **THE** solution when creating reports from large volumes of data.

Conceptually, a data model might appear like that illustrated in **Figure 2**. The data model links (*relates*) the *Chart of Accounts Table* to the *Transactions Table* based on the *Account Number* field appearing in both tables. Likewise, it might link the *Transactions Table* to the *Reporting Periods Table* based on that field’s presence in both tables. From a reporting perspective, once we connect the three tables, we can effectively treat them as one object, and that object is the singular data model for the workbook.

³ Technically, a data model can consist of a single table, but the use cases for single-table data models is limited. Accordingly, the focus of this discussion of data models will center on multi-table data models.

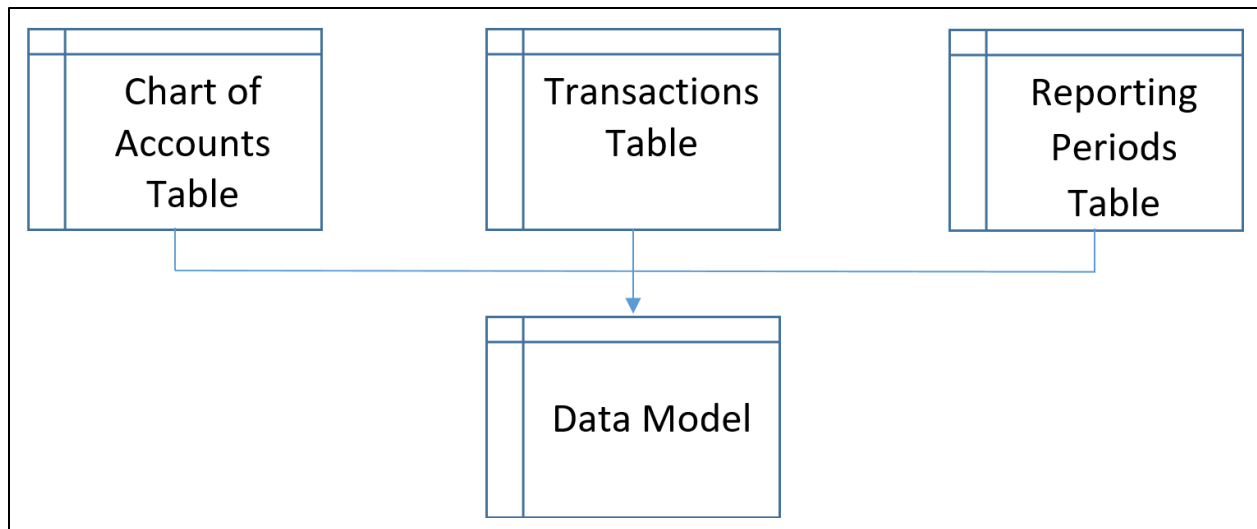


Figure 2 - Conceptual Example of an Excel-based Data Model

Power Query

Power Query, which Microsoft introduced in 2010, serves two primary purposes.⁴ First, you can use Power Query to *query* data from external sources to your workbooks or data models. This functionality is essential to facilitating Excel-based reports generated from real-time or near-real-time data. Accordingly, if you take advantage of this linked data, you can say goodbye to manual import/export and copy/paste routines to update your reports. Perhaps more importantly, you can *transform* that data to make it more useful in your reporting processes. Data transformations can automate the process of “cleaning” your information so that you do not expend time and effort to do so manually with every data refresh.

Power Pivot

One could argue that **Power Pivot** suffers from an identity crisis. More specifically – and contrary to popular opinion – you do not use Power Pivot to create PivotTables; in fact, it is impossible to develop PivotTables using Power Pivot. Instead, you use Power Pivot to manage the data model that supplies the data you choose to summarize using a PivotTable. Further, you can use Power Pivot to perform the following tasks:

- Create and manage relationships in the data model,
- Construct DAX-based formulas,

⁴ Beginning with Excel 2016, Power Query is no longer an add-in for Excel. Instead, all the functionality of Power Query is incorporated fully into the application.

- Build Key Performance Indicators (KPIs) for your reports, and
- Add elements such as dimensions and hierarchies into your data models.

These features, and others unnamed, can make your PivotTables more robust and easily understood.

Data Analysis Expressions

While *Next Generation Excel Reporting* generally seeks to minimize or eliminate the use of formulas, that is not always possible. You can use “smart” formulas, such as those discussed in Chapter Two, to create reports in such situations. However, we might need to access a more powerful function than what otherwise exists in Excel in other cases. Fortunately, **Data Analysis Expressions** (DAX) functions exist to address this issue. Approximately 300 DAX functions exist to work with data models in Excel; you can also use this same library of DAX functions when working in Power BI and Analysis Services. The formulas you create using DAX functions can address issues such as date-oriented, filtering your data, transforming textual data, and statistical calculations. Although a complete discussion of all DAX functions is beyond the scope of this session, perhaps what is most important to remember about them is that they serve as the vehicle for adding user-defined calculations into your data models. You can examine the full complement of DAX functions at <https://k2e.fyi/daxfunctions>.

Slicers and Timelines

The fifth item in the toolbox is visual filtering capabilities, provided through **Slicer** and **Timeline** filters. *Slicer* filters were added to Excel with the 2010 release to provide an easier way to filter PivotTables, including those summarizing data from data models. Further, in 2013, Microsoft updated Slicers to filter tables in Excel workbooks.

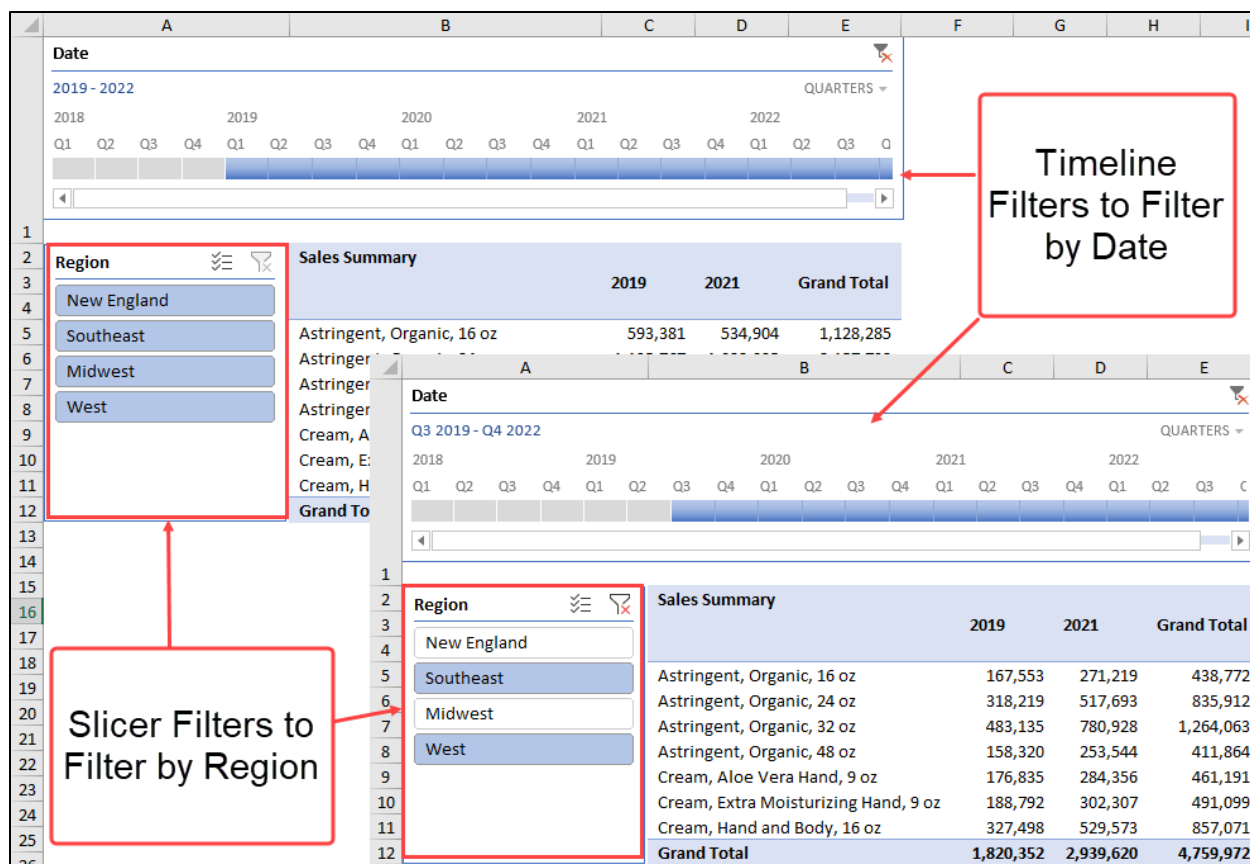


Figure 3 - Using Timeline and Slicer Filters in PivotTables

With the 2013 release of Excel, Microsoft added *Timelines*. Like Slicers, Timelines make it easier to filter data, specifically date-oriented data in your PivotTables. **Figure 3** illustrates how you can use Slicers and Timelines to filter data quickly and intuitively in a PivotTable.

How the Five Tools Work Together

As depicted in **Figure 4**, the five tools associated with next-generation Excel reporting work in concert to revolutionize financial and operational reporting. In this graphic, emphasis exists on the five tools discussed above.

Although these five tools work independently of the others, you will obtain the best results when you use them together. For example, you can use Power Query to access and transform data that feeds into a Power Pivot-managed data model. Additionally, you might add DAX-based user-defined calculations that appear on your Excel-based reports in that data model. Further, you can enhance these reports by adding Slicers and Timelines to facilitate easy and consistent filtering. Remember, these are complementary tools that work best together!

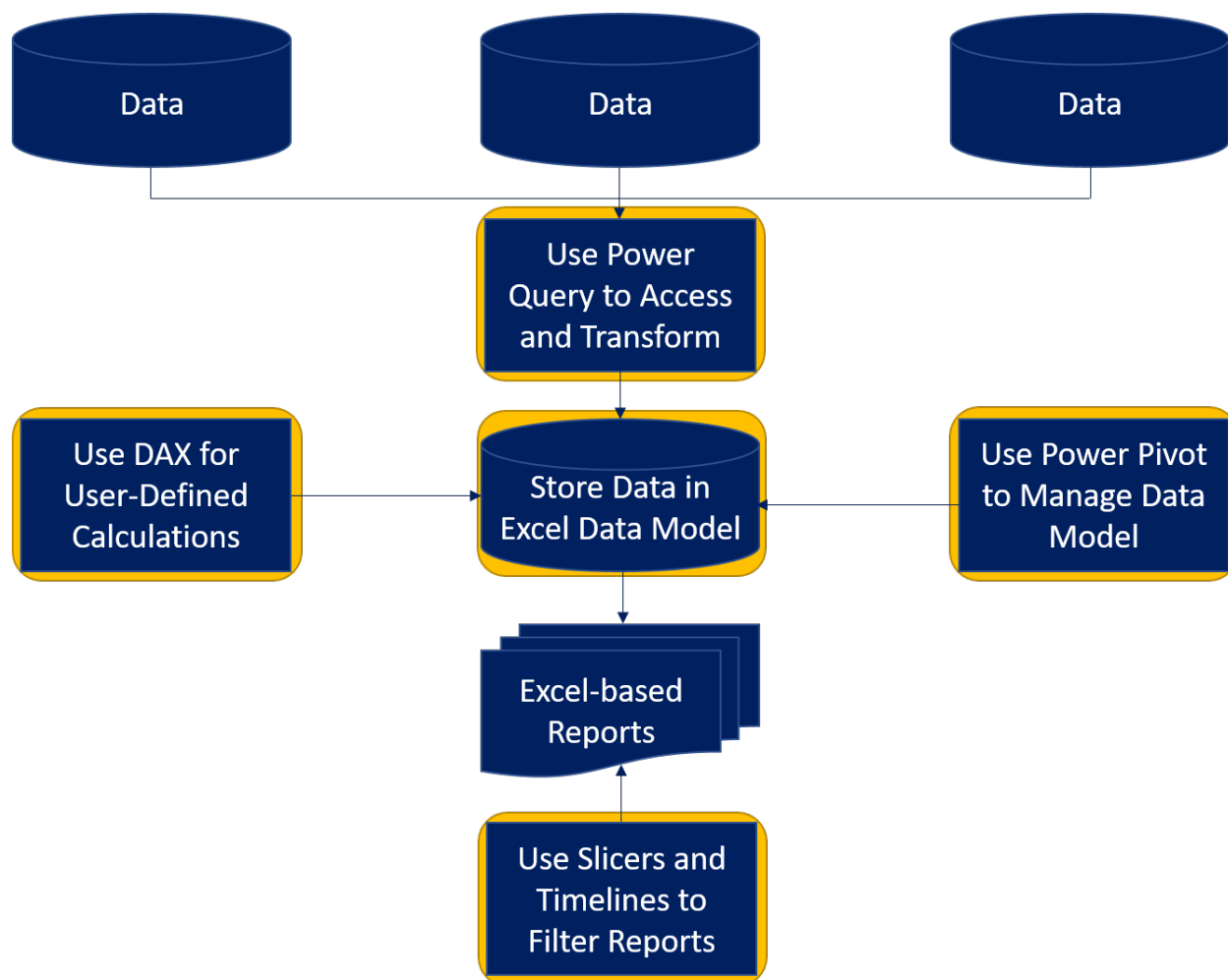


Figure 4 - A Graphical Depiction of Next-Generation Excel Reporting

Is This Always the Best Approach?

Generally, investing the time to build reports using the techniques outlined in this text will yield superior results over the long term. With few – if any – formulas in place, there will be reduced opportunities for accuracy issues. Updating reports is now virtually automatic because data will link into Excel from source databases – instead of being accessed manually. Further, by incorporating Slicers into the summaries, we can easily facilitate end-users selecting the dimensions and timeframes to filter and aggregate the data. Consequently, when you are creating reports that you expect to have a longer, useful life, building these reports using the ideas outlined in this course will typically provide the best results over the long term.

On the other hand, suppose you are responsible for generating an ad-hoc report that you expect will have only a limited useful life. Does it make sense to create such a report using these principles? The answer to that question is not as straightforward as you might expect. Suppose you choose to construct the summary using traditional, formula-based approaches. In that case, you may find it possible to generate the initial draft of the report more quickly than if you opt for

the Next Generation method. This probability's outcome increases if the summary is relatively simple and the data resides in Excel. Further, suppose you need to distribute the report to a relatively small number of people who won't require substantial filtering and aggregating features. In that case, it is more likely that a traditional, formula-based approach could suffice.

However, suppose the scope of the report evolves to become more complex. In that case, it becomes increasingly probable that you can construct the summary faster using the more modern techniques introduced in this chapter. Using Power Query to link and transform data from external data sources into a data model will generally be faster than relying on manual procedures to access and clean the data. Similarly, using PivotTables to summarize the transactional data will be quicker than creating formulas to accomplish the same task. Further, suppose filtering and aggregation options are necessary. In that case, you will probably find that Slicers and Timelines provide superior results, especially if you create multiple reports from the same data model. It is always wise to invest a few moments to plan your approach to get the best possible result in the least amount of time.



What are some reporting issues you and your organization experience in traditional, formula-driven Excel-based reporting processes? Do you see opportunities to re-engineer these processes using some of the abovementioned concepts?

Tables and Data Models – Cornerstones of Modern Reporting Practices

Chapter

2

Now that we have a common understanding of Next Generation Excel Reporting, let us examine two cornerstone technologies that facilitate this fresh approach. This chapter focuses on tables and data models, two cornerstones of modern reporting practices.

Learning Objectives

Upon completing this chapter, you should be able to:

- Define *table* in the context of Excel;
- List four advantages of working with tables in Excel;
- Identify three approaches to creating tables;
- Define *data model*;
- List three ways by which you can create data models in Excel; and
- Identify the primary purpose of creating relationships in data models.

Excel Table Fundamentals

Added to the 2007 release of Excel, tables remain one of the application's least utilized features. This condition is unfortunate because tables offer tremendous opportunities for improved accuracy and productivity in all facets of working with Excel, including reporting.

The following is a simple definition of an Excel table.

Tables are dynamically re-sizing ranges of data in Excel that facilitate end-user interaction at multiple levels. Users can interact with data in a table at the cell level, the row level, the column level, the header/footer level, and the table level. Because of the dynamically re-sizing nature of an Excel table and users' ability to interact with a table at multiple levels, tables are the preferred option for storing data in Excel workbooks.

Four Advantages of Working with Tables

While numerous benefits are associated with working with tables, the most significant are the following four.

1. **Tables serve as automatically re-sizing ranges of data.**
This feature may be the most important one associated with tables. Once you create a table or link data into a table, the table dynamically and automatically adjusts its dimensions to match the volume of data present. By doing so, any formula that references the table will also automatically include any newly added table data.
2. **Formulas that you enter in tables copy automatically down the entire column.** When you enter a formula in a table, that formula automatically copies (or *self-replicates*) to extend to all the table's rows. Further, if you add additional data to the table, a formula will continue to self-replicate to all newly-added rows.
3. **Formulas that reference information in a table are known as structured reference formulas.** Formulas that reference data in a table refer to the structure of the table's data, not traditional row-and-column addresses. Thus, as the dimensions of the table change, the integrity of your formulas remains intact. You will notice a resemblance between structured reference formulas and formulas that utilize Data Analysis Expressions.
4. **You can relate multiple tables together to create data models.** If you have two or more tables in a workbook, you can quickly and easily relate the tables together to create a data model. Once you establish the data model, you can build reports that summarize all the data from the data model without resorting to VLOOKUP, XLOOKUP, or similar functions to manipulate the data. The data model approach is more efficient, and it is also less prone to error.

Creating Tables

Knowing the benefits of working with tables, let us focus on creating tables to experience these benefits. Tables are easy to build using any one of three methods:

1. Creating tables from the Ribbon,
2. Creating tables with a keyboard shortcut, and
3. Creating tables from external data sources.

Additionally, although coverage is beyond the scope of this course, you can create tables with macros.

Data Considerations Before Creating a Table

Before you create a table using either of the first two techniques – creating a table from the Ribbon or using a keyboard shortcut –consider the data's arrangement and whether you should

make any modifications. Generally, you should manipulate the data for the table so that it is in contiguous rows and columns. Remember, you can always add rows and columns later, so there is no reason to include placeholders for future data when you create the table.

Additionally, while not necessary, you should have column headers at the top of each column that appropriately describe the column's data; if your data does not have column headers, Excel will insert headers such as *Column1*, *Column2*, and *Column 3* into your table. All the column headers in a table must be unique. Excel will append a number to the column's name to differentiate the labels if there are duplicate names. For example, if there are three columns labeled Date, converting the data into a table will cause Excel to rename the second occurrence *Date2* and the third occurrence *Date3*. Of course, you can change a column header by typing over it to replace the existing column header with the desired one.

Creating Tables from the Ribbon

With your data appropriately organized, you are ready to convert it into a table, and you can do so by issuing a command from the Ribbon. First, however, click inside the range of contiguous rows and columns. As shown in **Figure 5**, click **Format as Table** on the **Home** tab of the **Ribbon** and choose from one of the predefined Table Styles that appear in the resulting window. Doing so will transform your data into a table formatted to your desired style. As you move your cursor over the various icons, Excel's live preview will temporarily apply the formatting to your table. Clicking on a format applies it to the table. If you later change the style, click any cell in the table, choose the **Format as Table** icon again from the **Ribbon**, and select a different table style.

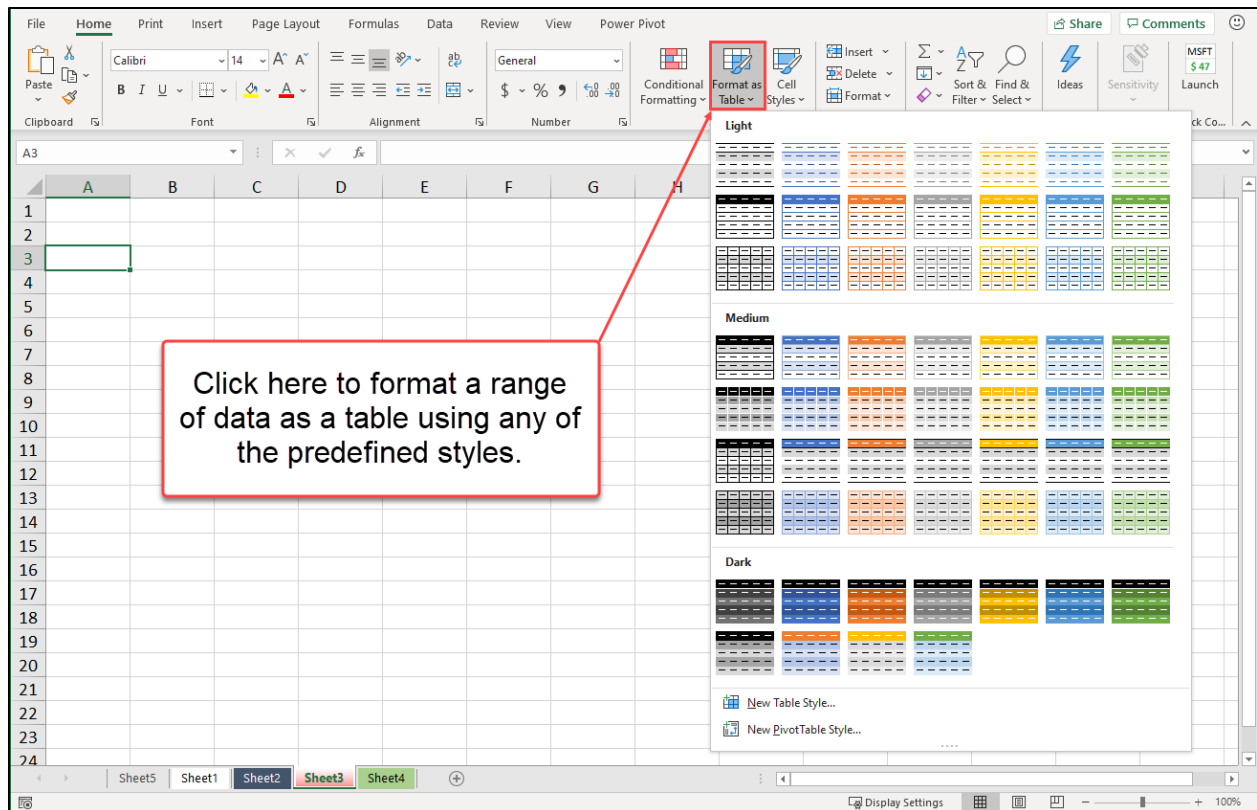


Figure 5 - Creating a Table from the Ribbon

Creating Tables with a Keyboard Shortcut

If you prefer to create your table using a keyboard shortcut, click anywhere inside the data range. Then press **CTRL + T** or **CTRL + L** to open the **Create Table** dialog box. Confirm the range of the data and whether your data has headers. Excel then transforms your data into a table using your default table style. As with creating a table from the Ribbon, if you wish to reformat the table using a different style, you can choose **Format as Table** from the Ribbon and choose a new table style.

Creating Tables from External Data Sources

Those who use Excel to query data from external sources can use Power Query to link external data into an Excel table. First, click **Get Data** from the Ribbon's **Data** tab, choose the data source, and click **Import**. Then, click the drop-down arrow next to **Load** in the resulting **Navigator** window and select **Load To**. Finally, in the resulting **Import Data** dialog box, choose the option for **Table** and click **OK** to complete the process. **Figure 6** illustrates this process.

Several essential considerations warrant expansion with the data queried into an Excel table.

- From a practical perspective, the data links from the data source into Excel. Thus, refreshing the query will update the table's contents as you change the source's data. To refresh the query, right-click the table and choose **Refresh**.
- If you add formulas to the table, they will self-replicate. Further, refreshing the query causes new records to appear in the table, and any formulas will self-replicate to the newly added rows.
- If you wish to "freeze" a query's results in the table, choose **Queries & Connections** on the Ribbon's **Data** tab to open the **Queries & Connections task pane**. There, you can right-click on the query and click **Delete**. Next, select **OK** or **Cancel** in the warning dialog box. This procedure de-couples the query from the workbook so that the data remains in the workbook, but you can no longer refresh the query.
- You can use the above to link data from multiple sources; those sources need not be homogeneous. For example, you might link data from an Excel workbook, a text file, and an Access database into three separate tables in Excel. Upon doing so, using techniques described later, you could relate those three tables together to create a data model and then build your report based on the collective data in the data model.

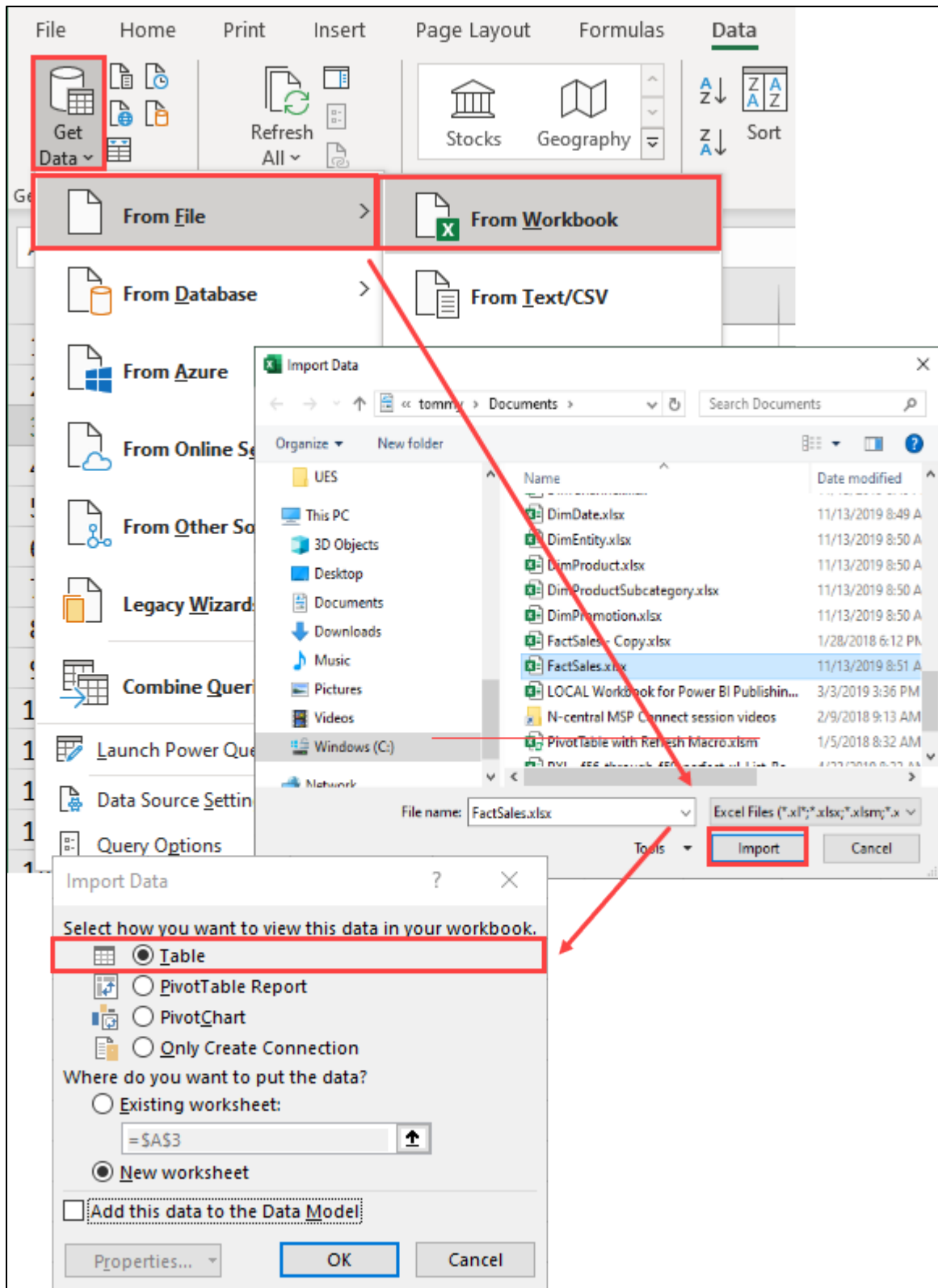


Figure 6 - Selecting a Data Source for a Data Query

It would be difficult, if not impossible, to overstate the importance of linking data from external data sources into a table. With this capability, you are relieved of the tedious and time-consuming tasks of copying and pasting data from external data sources into Excel. Therefore, your tables – and, by extension, all reports you build based on these tables – can summarize real-time data effectively, with no additional effort required once you construct the tables and related reports. This feature is a “game-changer” for many business professionals engaged in financial and operational reporting!

Naming and Renaming Tables

Regardless of the method you choose to create a table, Excel automatically assigns the table a name. The first table you create in a workbook inherits the name *Table1*, the second table you create in a workbook becomes *Table2*, and so on. These default names are non-descriptive of the table’s data and potentially confusing if you use numerous tables in a workbook. Fortunately, you can easily rename a table and assign it a name that makes it easier for all workbook users to understand the nature of the table’s data.

To rename a table, begin by clicking anywhere in the table. Then, from the **Table Tools Design** tab of the **Ribbon**, click on the **Table Name** field in the **Properties Group** and type the new name for your table, as shown in **Figure 7**.

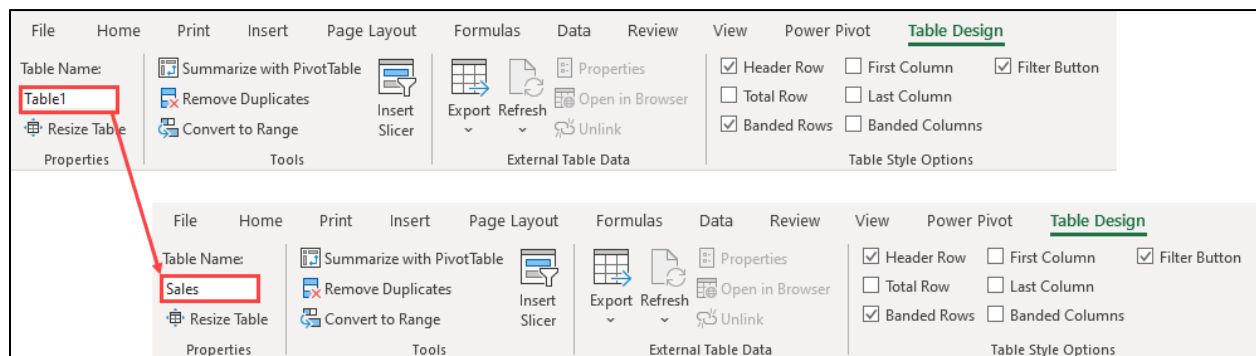


Figure 7 - Renaming a Table

While you have tremendous freedom when creating a new table name, you must follow some guidelines. The following list resembles the rules for other named objects in Excel, including defined names and macro names.

1. The first character of the table name must be a letter, an underscore character, or a backslash. You can use numbers and special characters in a table name, but they cannot be the name’s first character. Likewise, you cannot have spaces in the table name.
2. The length of the table name must be 255 characters or fewer.

3. The table name cannot be a combination of letters and numbers that could otherwise be a row and column address. For example, *A100* cannot be a table name because it is a valid cell address in a worksheet.
4. All table names must be unique within a workbook. Additionally, table names are not case-sensitive. For example, excel recognizes *Revenue*, *revenue*, and *REVENUE* as the same table name.

Re-sizing Tables Manually

After creating a table, Excel automatically re-sizes the table as you add and delete data. However, if you need to re-size the data manually, you can use either of the following techniques.

First, you may do so by specifying the table's updated coordinates in the **Resize Table dialog box**. To access the Resize Table dialog box, click anywhere in your table and then click **Resize Table** in the **Properties** group on the **Table Tools Design** contextual tab. Upon doing so, Excel opens the Resize Table dialog box pictured in **Figure 8**. Note that you can enter the new coordinates of the table directly into the dialog box, or you can use a click-and-drag approach to re-size your table.

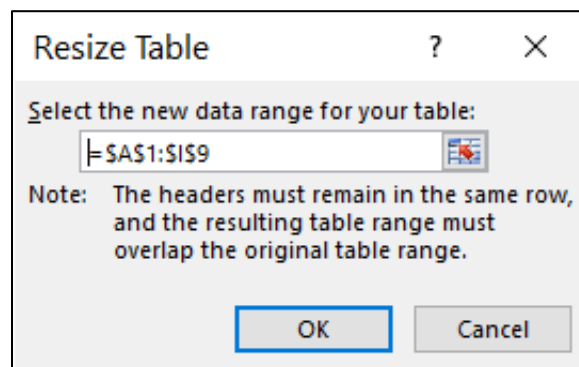


Figure 8 - Editing the Dimensions of a Table in the Resize Table Dialog Box

You can also click and drag the handle in the lower right corner of a table to re-size the table. As shown in **Figure 9**, the handle is the small icon in the table's lower right corner.

	A	B	C	D	E	F	G	H	I
1	Date	Store	Product	UnitCost	UnitPrice	SalesQuantity	TotalCost	SalesAmount	Gross Margin
2	1/1/2017	1	1749	36.11	109	4	\$ 144.44	\$ 126.00	\$ 291.56
3	1/10/2017	1	1749	36.11					\$ 145.78
4	1/13/2017	1	1749	36.11					\$ 72.89
5	1/14/2017	1	1749	36.11					\$ 437.34
6	1/21/2017	1	1749	36.11	109	5	\$ 180.55	\$ 545.00	\$ 364.45
7	1/27/2017	1	1749	36.11	109	8	\$ 288.88	\$ 872.00	\$ 583.12
8	1/30/2017	1	1749	36.11	109	18	\$ 649.98	\$ 1,962.00	\$ 1,312.02
9	1/31/2017	1	1749	37.42	111	9	\$ 336.78	\$ 999.00	\$ 662.22
10									

Figure 9 - Click and Drag on the Handle to Re-size the Table

Clicking and dragging the handle allows you to enlarge or shrink the table according to your needs.



What are the most beneficial attributes of working with tables, particularly when creating financial and operational reports?

What Is a Data Model?

As identified previously, a data model is a collection of two or more tables that you *relate* together based on the presence of one or more common fields to create what Excel interprets as a single source of data⁵. If you are familiar with a relational database management system (RDBMS) such as Access or SQL Server, the notion of a data model should be one that you easily recognize. Once you create a data model, you can use the data in the model in PivotTables (including those built using Power Pivot), PivotCharts, Power View, and 3D Maps (formerly known as Power Maps).

Conceptually, a data model might appear like that shown in **Figure 10**. As shown in this simplified illustration, three tables relate directly or indirectly to create a single data model. For example, in that illustration, Table 1 relates to Table 2 based on a common field (column) in both tables. Likewise, another common field in both tables links Table 2 to Table 3. Importantly, all three tables do not need to relate directly to each other; instead, *indirect* relationships often exist in data models.

⁵ Technically, a data model can consist of a single table. Reasons for building a single-table model could include, for example, wanting to use Power Pivot's KPI features on the data in a single table. However, because most of the power associated with using data models vests with multi-table data models, we will focus on that approach in this course.

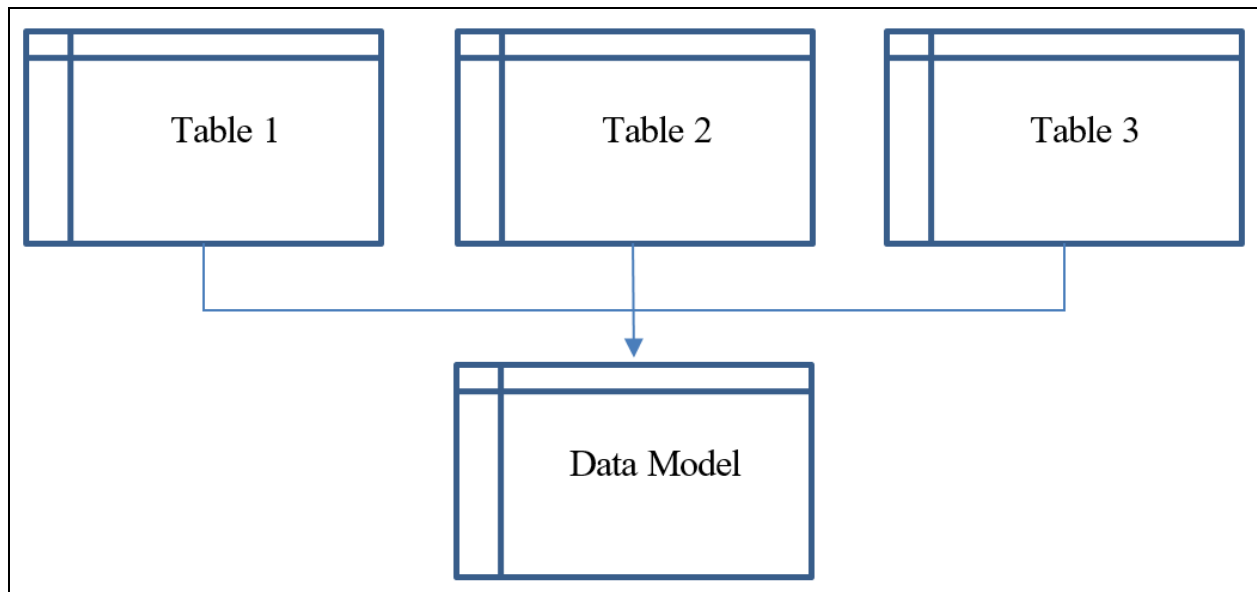


Figure 10 - Schematic of an Excel Data Model

Once you create a data model, you can use its' data to build analytical tools such as PivotTables and even formula-based reports that use cube functions. Additionally, you can create user-defined calculations in your data models by using DAX expressions to build calculated columns and measures in the data model. Importantly, you cannot summarize data in data models using traditional formulas created with conventional Excel functions.

A given Excel workbook can contain only a single data model, and that data model is named "ThisWorkbookDataModel." However, many professionals see this single object in Excel as the key to better data analysis. This in-memory, multidimensional, relational data engine allows users to combine data from multiple sources and handle data sets that contain billions of records without significant performance degradation. For example, the data model can handle up to 99 billion records, although no field can house more than 1.99 billion unique records. Because of the tremendous analytical capabilities enabled through data models, you might attempt to load enormous data sets into a data model. However, while the volume of data you can store in a data model is virtually unlimited from a practical perspective, Excel always limits you to the overall limit on the size of an Excel workbook. For instance, if you run a 32-bit version of Excel 2013 or newer, that limit is 2 GB. However, if you are running a 64-bit version of Excel 2013 or newer, the workbook's size is limited only by the amount of memory and system resources you have on your computer. If you find that the volume of data you are attempting to work in your data model is problematic, you may consider installing Microsoft's **Workbook Size Optimizer**; it is available for download directly from Microsoft by visiting the following link: <https://www.microsoft.com/en-us/download/details.aspx?id=38793>. The Optimizer can enable better data compression and enhance performance in some cases.

Power Pivot functions as the primary user interface for Excel data models. Among the actions you can use Power Pivot for are:

- Forming and managing relationships in a data model,
- Adding user-defined calculations,
- Analyzing performance with Key Performance Indicators,
- Establishing perspectives and hierarchies in your data, and
- Defining and managing fiscal reporting periods.

Notwithstanding the above, you can edit data relationships without accessing Power Pivot. Notably, some Excel users will not have access to Power Pivot but will still be able to take advantage of the data model, albeit limited.

Each Excel workbook contains only one data model, but that single data model can serve as the data source for multiple data analysis objects in the workbook. As with most innovative technologies, using the data model has advantages and disadvantages.

Advantages of the Data Model

The following list identified the primary advantages of using the data model as a data source for PivotTables and other reports.

- The data model allows users to build PivotTables from multiple related tables and a wide variety of familiar external data sources while maintaining links to the data so that you may refresh the reports.
- You can convert PivotTables built on the data model to cube formulas with a single click on the Ribbon. Converting a PivotTable to a formula-based report allows you to insert rows or columns and create needed subtotals or other ancillary calculations without impairing the ability to refresh the report as the data is updated.
- You can use cube functions to prepare formal reports directly from the data model without creating a PivotTable to summarize the data. Instead, the data model contains an intelligent OLAP cube populated from the tables stored in the model. Think of the cube as a multidimensional PivotTable with pre-calculated measures at each intersection of the cube.
- PivotTables built on data models support drill-down and around with Quick Explore, an available feature beginning with Excel 2013. Quick Explore provides functionality beyond mere drill-to-detail of summarized values that appear in ordinary PivotTables. Quick Explore allows less sophisticated users to rearrange a PivotTable in an automated process.

Quick Explore enables them to view the data from other perspectives, even if they do not have the knowledge or skill to rearrange the PivotTable manually.

Disadvantages of the Data Model

While there are significant advantages to using the data model, there are also disadvantages. Following is a list of issues associated with using a data model as a data source for PivotTables.

- You cannot create calculated fields or items in PivotTables using data models as their sources. However, as workarounds, you can construct user-defined calculations in your data models as measures and calculated columns. Then, you can use these calculations as proxies for calculated fields and calculated items in your PivotTables.
- Grouping items is not available in a PivotTable built on the data model. However, you can create calculated columns to produce the required groupings as a workaround to this limitation.
- In default, drilling to detail only displays the first 1,000 records underlying a summarized cell. However, you can modify the Connection Properties to facilitate drill-through and see a larger record count.
- Excel 2013 and newer workbooks using the data model are incompatible with earlier Excel versions.

Creating Simple Data Models from Excel Tables

Creating data models in Excel can be easy; you can do so with a single mouse click. However, before attempting to relate multiple Excel tables, you should verify that each table has a field common to the other table(s) to which you wish to establish the relationship. For example, suppose you are attempting to relate two tables containing inventory data; in this case, you would probably look for a part number or SKU code common to both tables. Likewise, if you desire to relate multiple employee-related tables, you would likely do so based on a common field of employee identification numbers or a similar column.

Our first example of relating tables together to create a data model will be in the context of creating a PivotTable. In this example, we have an Excel workbook that contains two tables – *Transactions* and *Product*. We need to build a PivotTable that uses both tables' data, but we do not want to use XLOOKUP or VLOOKUP to link data from one table to another, attempting to consolidate the data from the two tables. Fortunately, we can use data models to solve the problem.

Begin by clicking on either of the tables. Then, click **Summarize with PivotTable** from the **Table Tools Design** tab of the Ribbon to open the **Create PivotTable** dialog box pictured in **Figure 11**. Then check the box next to **Add this data to the Data Model**. Importantly, checking this box

makes *all* tables in the workbook – not just the currently selected table – available to the workbook’s data model.

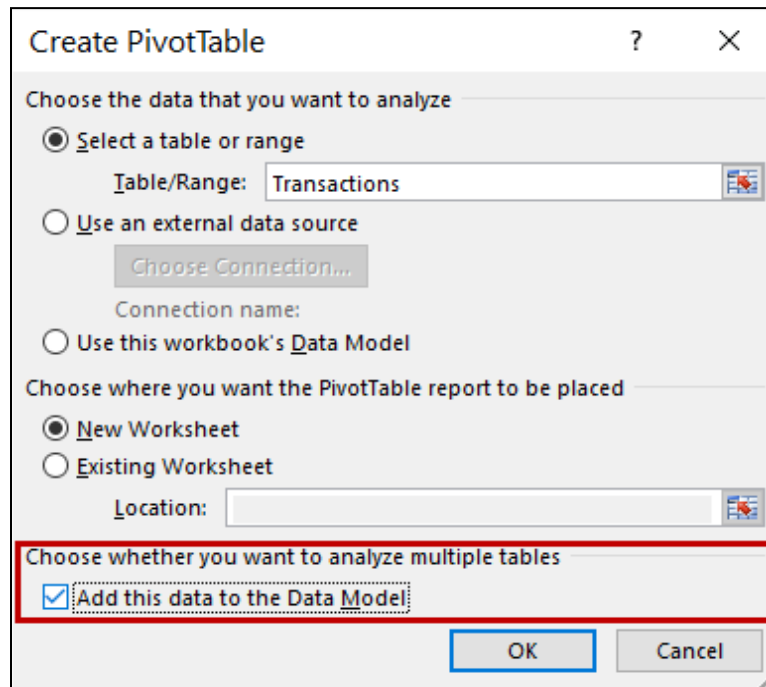


Figure 11 - Adding Tables to a Workbook’s Data Model

Click **OK** to complete the process, and Excel opens a new workbook with the **PivotTable Field List** available, as shown in **Figure 12**. As indicated in the figure, click **All** to see all the tables you can use as data sources for your PivotTable. Further, notice the faint line that separates the *Product* table from the *Transactions* table. This line indicates that the two tables are not yet related. You should never build a PivotTable (or any other analytical tool) from tables that are not related correctly! If you do, Excel displays a warning message indicating that you may need to link your tables. Further, if you build a PivotTable from unrelated tables, you will probably notice that your PivotTable calculations do not seem to make sense. In short, always ensure proper relationships exist before continuing with your PivotTable.

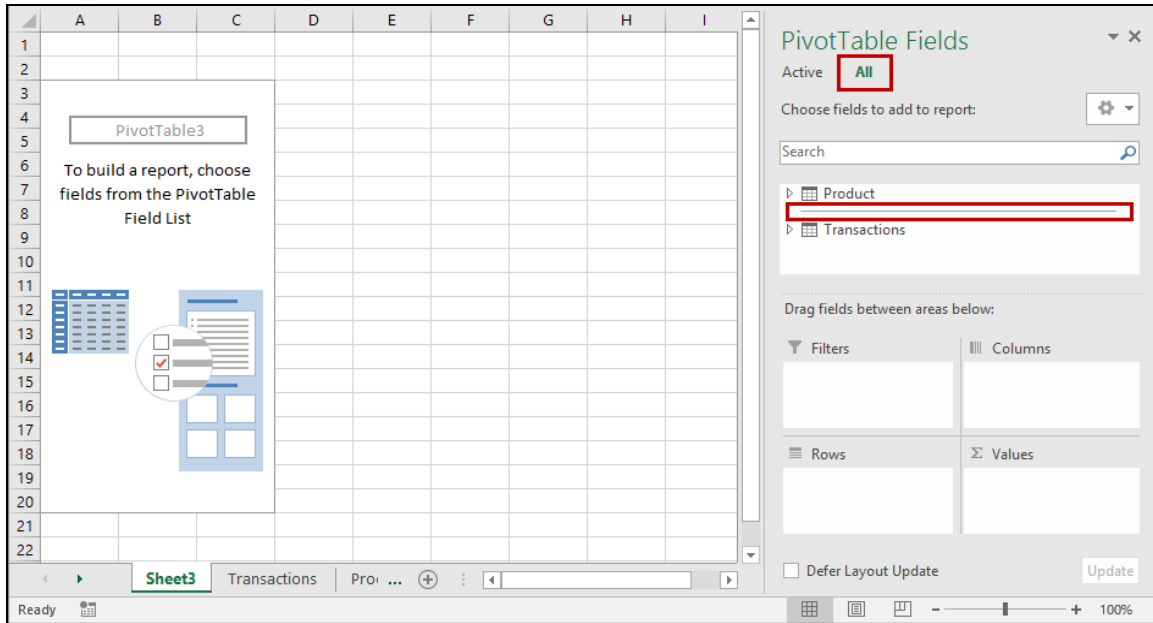


Figure 12 - Unrelated Tables in PivotTable Field List

You can establish relationships between the tables in the data model using multiple approaches. First, if Excel provides an **Auto-Detect** option similar to that pictured in **Figure 13**, you can click the **Auto-Detect** button. When you do, Excel will attempt to identify and relate the tables together automatically. There is a high probability that this process will be successful in relatively simple data models. However, if it is not, you will need to establish the relationships manually. Then, you can click the **Create** button pictured in Figure 13 or click **Relationships** from the **PivotTable Tools Analyze** tab of the Ribbon. Both approaches will open the **Manage Relationships** dialog box shown in **Figure 14**. You can also access the Manage Relationships dialog box by clicking the **Relationships** button on the Ribbon's **Data** tab.

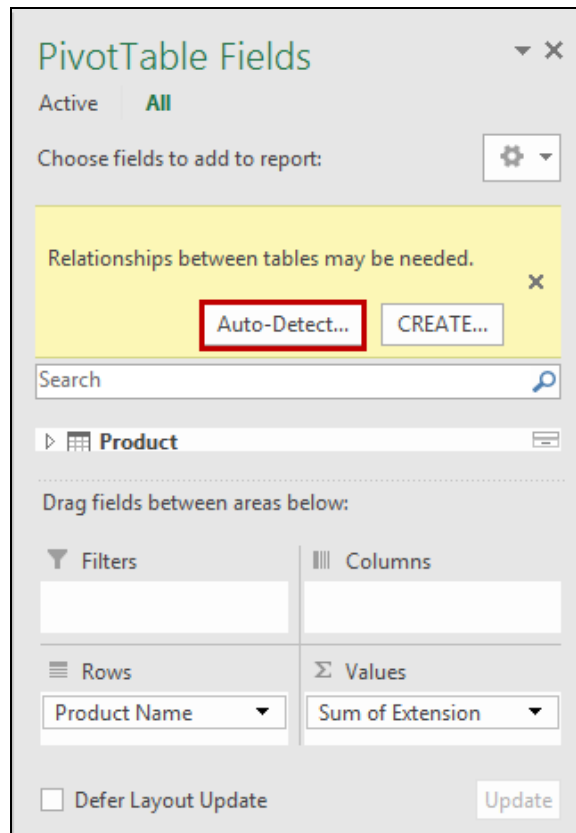


Figure 13 - Using Excel's Auto-detect Feature to Establish Relationships in Data Models



Figure 14 - Excel's Manage Relationships Dialog Box

In the Manage Relationships dialog box, click the **New** button to open the **Create Relationship** dialog box shown in **Figure 15**. Choose the tables you wish to relate together from the left side of this dialog box. Next, select the fields common to the two tables on the dialog box's right side.

Click **OK** to complete the process of linking these two tables. Then, repeat this process for as many tables as you need to relate.

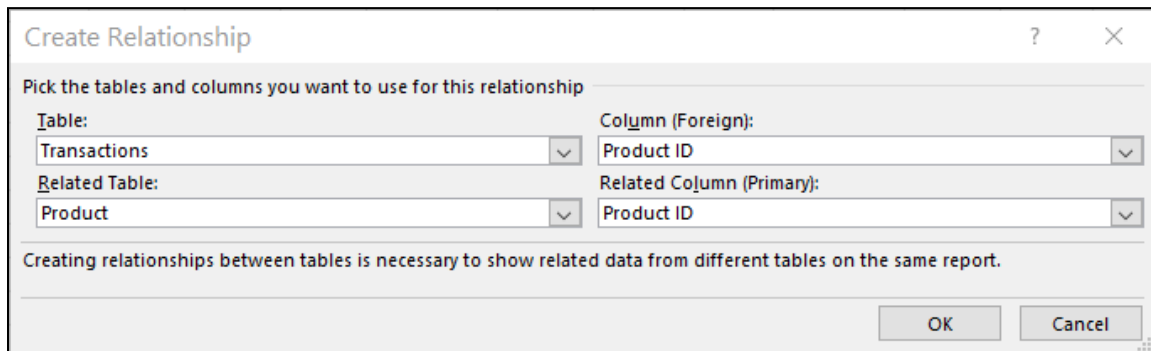


Figure 15 - Manually Establishing Relationships between Tables in a Data Model

After successfully establishing the relationships, the Manage Relationships dialog box will change to reflect each one and its current status, as shown in **Figure 16**.

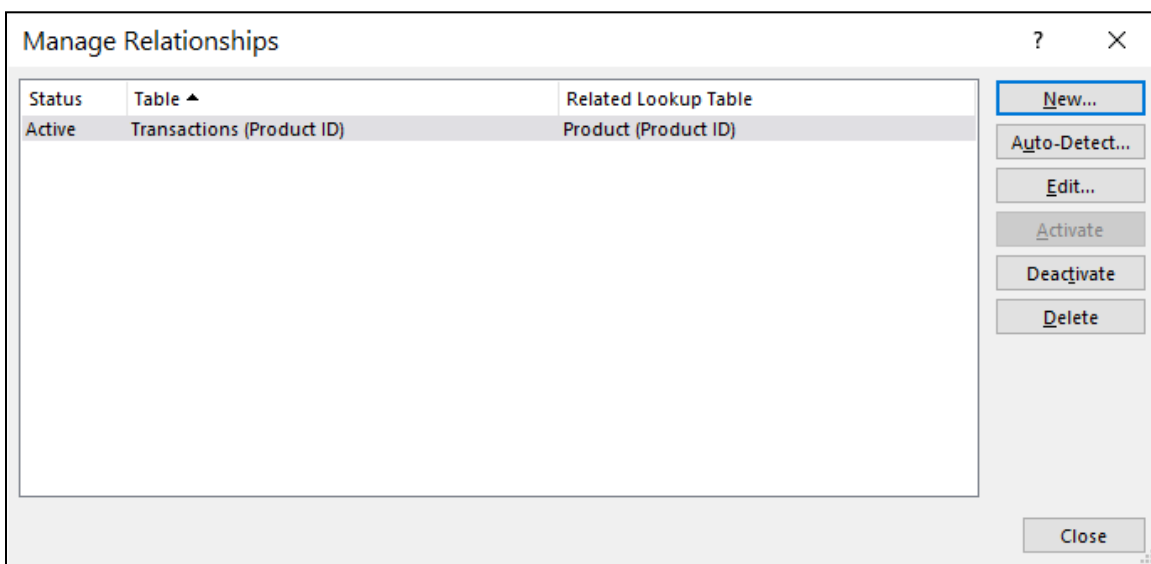


Figure 16 - Displaying All Relationships in the Manage Relationship Dialog Box

With the relationship between the two tables established, we can easily construct a PivotTable that utilizes data from both tables in the data model. **Figure 17** presents an excerpt of such a PivotTable.

	A	B	C	D	E
3	Summary of Sales by Product/Year				
4		2019	2020	2021	Grand Total
5	Active Outdoors Crochet Glove	516.20	3,563.44	775.03	4,854.67
6	Active Outdoors Lycra Glove	674.85	4,922.01	1,560.09	7,156.95
7	Descent	77,759.06	1,312,790.24	495,070.87	1,885,620.17
8	Endorphin	12,192.99	204,626.02	60,020.01	276,839.02
9	Guardian "U" Lock	35.00	763.90	105.00	903.90
10	Guardian ATB Lock	39.42	782.93	306.60	1,128.95
11	Guardian Chain Lock	27.00	250.68	39.38	317.06
12	Guardian Mini Lock	325.23	825.63	434.73	1,585.59
13	Guardian XL "U" Lock	19.90	1,104.46	179.10	1,303.46
14	InFlux Crochet Glove	54.00	1,732.13	693.23	2,479.36
15	InFlux Lycra Glove	369.69	1,948.37	593.67	2,911.73
16	Micro Nicros	4,307.31	24,568.02	8,436.25	37,311.58
17	Mini Nicros	5,524.28	27,128.10	6,426.19	39,078.57

Figure 17 - Sample PivotTable Created from a Data Model

Data Model Relationships

When working with data models, you should understand a few essential points regarding relationships between tables. First, the names of the columns that you relate need not be the same. For example, in the previous illustration, one of the column names could have been *Product ID*, and the other column name could have been *Part Number*. What is necessary is that the columns have the same type of data or compatible data types.

Second, Excel data models currently support only two relationship types: 1) *one-to-one* and 2) *one-to-many relationships*. An example of a one-to-one relationship exists when you have an employee's name and address in one table and a single corresponding record for that same employee's tax withholding status in another table. An example of a one-to-many relationship exists when you have a single product in one table and multiple transactions associated with that same product in a separate table, as with the data model created in Figure 17. Excel does not support *many-to-many* relationships in data models. For example, a many-to-many relationship might exist when relating a product table to a customer table. This relationship is a many-to-many relationship because each customer can buy many products, and many customers can buy the same product.

Third, three types of *keys* are fundamental to Excel data models.

1. **Primary key:** uniquely identifies a row in a table, such as Product ID in the Product table.
2. **Alternate key:** a column other than the primary key that is unique. For example, the Product table might also store a unique SKU code.
3. **Foreign key:** a column that refers to a unique column in another table, such as Product ID in the Transactions table, which refers to Product ID in the Product table.

The primary key or alternate key is the related column in a data model. If a table has a primary key and an alternate key, you can use either as the basis of a relationship to another table. The foreign key is referred to as the source column or just *column*. In the example above, a relationship exists between the Product ID in the Transactions table (contains the *column*) and the Product ID in the Product table (contains the *related column*). When creating a one-to-many relationship, the “one” side of the relationship will always be the *related column*.

Fourth, the data must meet three requirements to create relationships between tables.

1. Each table must have a single column of data that uniquely identifies each row in that table. As indicated above, this column of data is the table’s primary key or an alternative key.
2. The lookup column’s data values must be unique; they cannot contain duplicate or null values.
3. The data types in the source column and lookup column must be compatible. For example, when establishing relationships, whole numbers, decimal numbers, true/false, text, dates, currency, and blanks are compatible data types.

Simple Relationships in Data Models

After creating a data model, you may find it necessary to add, delete, activate, or deactivate relationships in the data model. First, return to the **Manage Relationships** dialog box pictured in **Figure 18** and choose the relationship to manage. Then click the appropriate button on the right side of the dialog box.

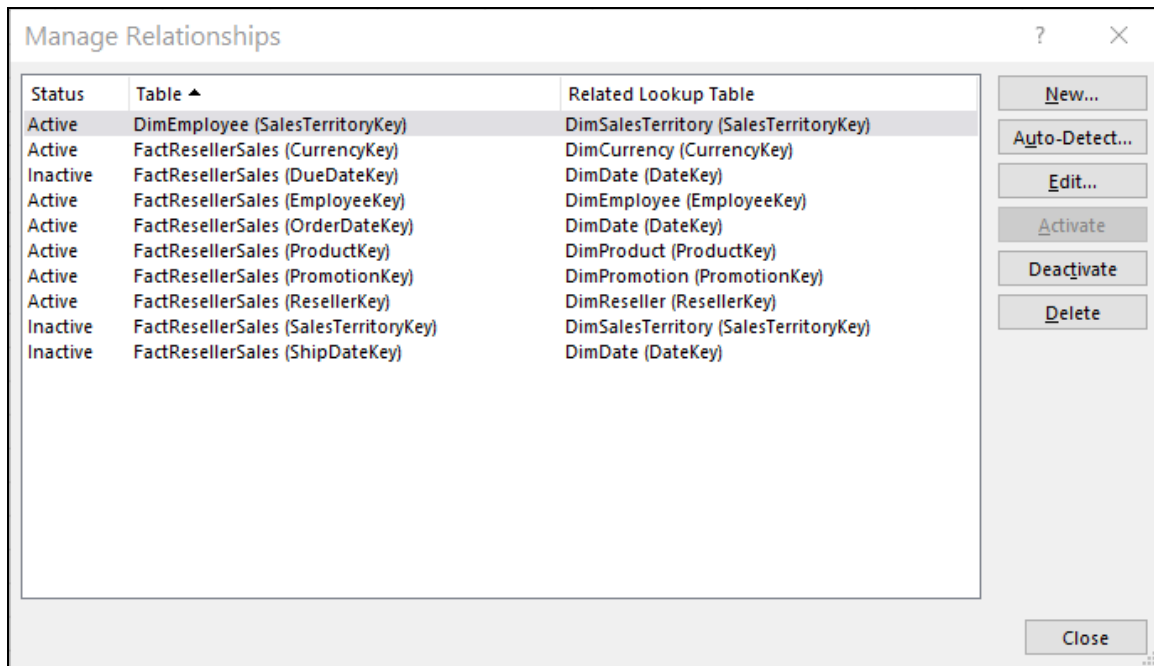


Figure 18 - Managing Relationships in a Data Model

As shown in Figure 18, a single table can have multiple relationships to other tables in the same data model. For example, the FactResellerSales table has three connections to the DimDate table – the DueDateKey, the OrderDateKey, and the ShipDateKey fields in the FactResellerSales table all relate to the DateKey field in the DimDate table. This condition often occurs when you import tables with multiple relationships defined in the original data source. In such cases, only one of the relationships can serve as the active, current relationship; Excel automatically marks the others as inactive. For example, in Figure 18, the relationship between the OrderDateKey in the FactResellerSales table and the DateKey field in the DimDate table is the active relationship between those two tables. The other two relationships are inactive. Therefore, if you wanted the relationship between the two tables to use the DueDateKey field in the FactResellerSales table, you would need to select the existing active relationship and then click the **Deactivate** button in the **Manage Relationships** dialog box. After doing so, select the inactive relationship between the DueDateKey and DateKey fields and click **Activate**. Be aware that such changes can cause calculations based on the data model to change. Accordingly, carefully scrutinize the results of such changes to ensure your reports' continued accuracy.

A Summary Example of Simple Data Models

Many businesses report on fiscal years, not on calendar years. However, the built-in grouping options in PivotTables do not support groupings based on fiscal reporting periods, leading to untold frustration and time lost trying to implement manual workarounds. Fortunately, this is ideal for using data models to facilitate reporting on fiscal years. In this example, we will work with data in two tables – *Sales* and *Dates*. More specifically, we will create a data model from

which we will construct a PivotTable that groups information on a fiscal year that begins on July 1st and ends on June 30th.

Figure 19 illustrates a selection of the 18,271 rows of transactional data in the Sales table on which we need to report by fiscal period.

	A	B	C
1	TransDate ▾↑	Location ▾	Sales ▾
14601	12/31/2019	Shreveport Store	5,918.71
14602	1/1/2020	Baton Rouge Store No 1	6,905.52
14603	1/1/2020	Baton Rouge Store No 2	6,500.58
14604	1/1/2020	Jackson Store	5,932.37
14605	1/1/2020	Slidell Store	5,325.62
14606	1/1/2020	Hammond Store	5,120.09
14607	1/1/2020	Lafayette Store	5,681.53
14608	1/1/2020	Biloxi Store	5,174.15
14609	1/1/2020	New Orleans Store No 1	6,578.25
14610	1/1/2020	New Orleans Store No 2	6,821.27
14611	1/1/2020	Shreveport Store	5,919.57
14612	1/2/2020	Baton Rouge Store No 1	6,906.54
14613	1/2/2020	Baton Rouge Store No 2	6,501.53
14614	1/2/2020	Jackson Store	5,933.24
Daily Store Sales Fiscal Periods			

Figure 19 - Sales Data to Summarize by Fiscal Year

In **Figure 20**, the Dates table “maps” the data in column A into the appropriate **Fiscal Month**, **Fiscal Quarter**, and **Fiscal Year**. We will use the data in columns B through E for grouping purposes instead of grouping the PivotTable’s transactions based on the data in column A.

	A	B	C	D
1	TransDate ▼	FMonth ▼	FQuarter ▼	FYear ▼
1455	12/24/2020	12	2	2021
1456	12/25/2020	12	2	2021
1457	12/26/2020	12	2	2021
1458	12/27/2020	12	2	2021
1459	12/28/2020	12	2	2021
1460	12/29/2020	12	2	2021
1461	12/30/2020	12	2	2021
1462	12/31/2020	12	2	2021
1463	1/1/2021	1	3	2021
1464	1/2/2021	1	3	2021
1465	1/3/2021	1	3	2021
1466	1/4/2021	1	3	2021
1467	1/5/2021	1	3	2021
◀ ▶		Daily Store Sales		Fiscal Periods

Figure 20 - Excerpted Data from Dates Table

Begin the PivotTable creation process by selecting **Insert, PivotTable** to open the **Create PivotTable** dialog box. Enter the name of the first table. Be sure to check **Add this data to the Data Model** as shown in **Figure 21**. Click **OK** to display the PivotTable placeholder and Field List.

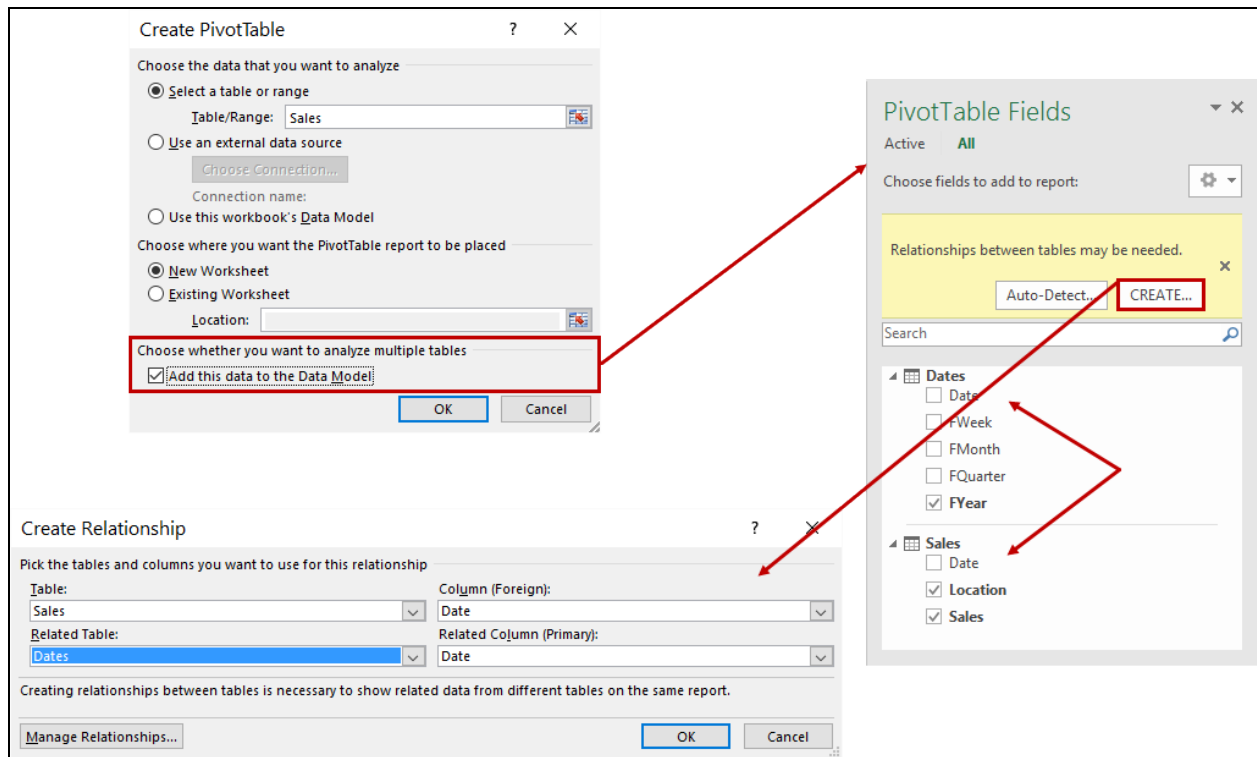


Figure 21 – Creating a PivotTable Using the Excel Data Model

When you add a table to the data model, the PivotTable Field List can display all fields from all tables in a workbook. If the tables are not related, you can create relationships directly from the Field List based on the presence of a common field, as shown in Figure 21.

After establishing one or more relationships, you can construct the PivotTable to summarize the data based on the appropriate fiscal reporting period. To do so, drag the **FMonth**, **FQuarter**, and **FYear** fields onto the PivotTable. **Figure 22** provides a sample of the completed PivotTable. In particular, note the *Date* field's omission. Instead, see how the *FMonth*, *FQuarter*, and *FYear* fields summarize the PivotTable's data by fiscal reporting period.

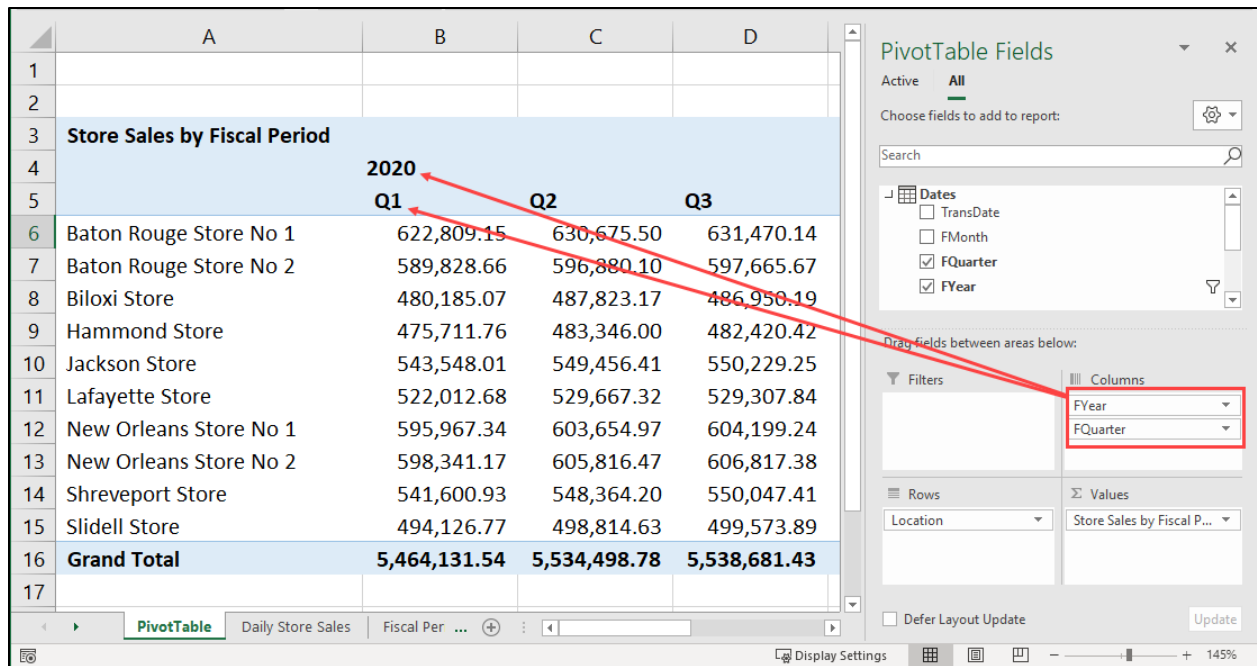


Figure 22 - PivotTable Based on Fiscal Reporting Periods

? What characteristics are essential for establishing and maintaining relationships between tables inside a data model? In the absence of data models, what options exist to “link” data from one table to another?

Accessing and Transforming Data with Power Query

Chapter

3

Power Query is the ultimate “game-changer” for accessing and transforming data in Excel. Power Query can work with data from relatively simple sources, such as text files and other Excel workbooks. You can also use Power Query with the same ease on more sophisticated data sources, including SQL Server databases and web-based data. Once you access your data with Power Query, you can apply countless transformations to the data to make it more useful. When you see the speed and ease with which you can access and transform your data with Power Query, you’ll wonder how you ever worked without it!

Learning Objectives

Upon completing this chapter, you should be able to:

- Identify the two major roles of Power Query;
- List the steps for querying data into Excel using Power Query; and
- Identify at least five types of data transformations you can implement with Power Query.

Introducing Power Query

Power Query appeared in Excel 2010 when Microsoft made it available in all Excel versions. Once installed, you can access Power Query’s functionality from the Ribbon’s Power Query tab through the 2013 Excel release. But, of course, because the tool was an add-in, you first needed to ensure that you had downloaded, installed, and activated it on your computer. Otherwise, you would not have seen the Power Query tab as an option on the Ribbon.⁶

Microsoft embedded Power Query’s functionality into Excel and made it available from the Ribbon’s **Data** tab beginning with Excel’s 2016 release. More specifically, as shown in **Figure 23**, what you now see as the **Get & Transform Data** section of the Ribbon’s **Data** tab includes all the features and functionality of Power Query. Further, because this feature is “fully baked” into Excel, there is no longer a need to worry about downloading, installing, or activating an add-in,

⁶ If you are running Excel 2010 or Excel 2013 and need to download the Power Query add-in, you can do so from Microsoft’s website at the following link: <https://www.microsoft.com/en-us/download/details.aspx?id=39379>.

thereby simplifying the process of taking advantage of the functionality provided by Power Query.

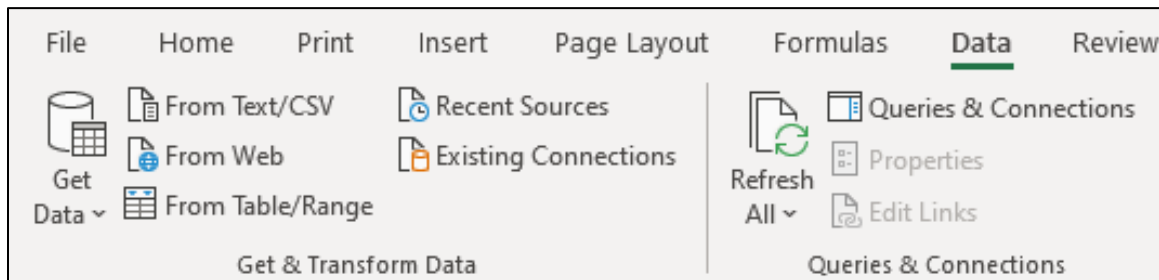


Figure 23 - Accessing Power Query from Excel's Data Tab of the Ribbon

At a high level, Power Query performs two valuable functions.

1. It facilitates relatively quick and easy connections linking data from external sources into Excel.
2. It simplifies transforming and cleansing the queried data to make it more useful.

Power Query supports linking and importing data from virtually every primary source. Click **Get Data** from the Ribbon's **Data** tab, choose your data source, and select the table(s) you wish to link into Excel to get started with a query. Upon doing so, you can choose to load the data directly onto worksheets within your workbook, into Excel's data model without loading the data onto worksheets, or into Excel's data model and onto worksheets. Further, you can edit your query to apply various transformation and cleansing options Power Query supports.

Regarding the transforming and cleansing processes you can apply to your data, the following are some of the many actions you can take using Power Query.

- Refresh, merge, and append queries
- Combine data from multiple data sources
- Filter and group rows in a table
- Expand a column containing an associated table
- Aggregate data from a column
- Pivot and unpivot data
- Transpose data
- Merge columns, delete columns, and create new columns of data
- Sort data
- Find & replace values
- Remove rows with errors and promote a row to column headers

Importantly, these transformations and cleansings are not one-time events. Every time you refresh or reuse a query, all transformations and cleansings you save in the query execute

automatically. Of course, this feature saves you from the tedious and time-consuming process of manually transforming and cleansing your data repeatedly.

Building and Adding to Data Models with Power Query

One of Power Query's most valuable advantages is that it facilitates queries from the forty-one data sources in **Table 1**. Notice the many "non-standard" data sources, such as PDF documents, Exchange, and Salesforce. Based on the data sources listed, Power Query provides robust capabilities to access and link data from numerous data sources. This capability means that even users with limited technical skills can quickly link data from their accounting, ERP, CRM, practice management, or other line-of-business databases into Excel for reporting and data analysis.

Files	Databases	Azure	Online Services	Other Sources
Excel	SQL Server	Azure SQL Database	SharePoint Online List	Table/Range
Text/CSV	Access	Azure Synapse Analytics	Microsoft Exchange Online	Web
XML	Analysis Services	Azure HDInsight	Dynamics 365	Microsoft Query
JSON	SQL Server Analysis Services Database (Import)	Azure Blob Storage	Salesforce Objects	SharePoint List
Folder	Oracle	Azure Table Storage	Salesforce Reports	OData Feed
SharePoint Folder	IBM DB2	Azure Data Lake Store		Hadoop File (HDFS)
PDF	MySQL	Azure Data Explorer		Active Directory

Files	Databases	Azure	Online Services	Other Sources
	PostgreSQL			Microsoft Exchange
	Sybase			ODBC
	Teradata			OLEDB
	SAP Hana			Blank Query

Table 1 - Potential Data Sources for Power Query

Additionally, if you work with a data source that is not listed, you can likely still use Power Query to link data into Excel. For example, one of Power Query's supported data sources is **Open Database Connectivity (ODBC)**. Suppose the database containing your data complies with the ODBC specification, and you have an appropriate ODBC driver installed on your computer. In that case, you can use Power Query's ODBC source to facilitate your data connection. This capability means, for example, that you can use Power Query to access data from sources such as QuickBooks and QuickBooks Online.

An example of using Power Query to load data into an Excel data model follows. Of course, an illustration for each of the forty-one data types pictured in Table 1 is impractical. However, the example chosen will provide you with the background you need to use Power Query to link and transform data from virtually any data source into Excel.

Querying Data from SQL Server

As shown in **Figure 24**, to launch a new query with Power Query, click **Get Data** and choose your data source; in this example, the data source will be an SQL Server database that serves an organization's ERP system.⁷

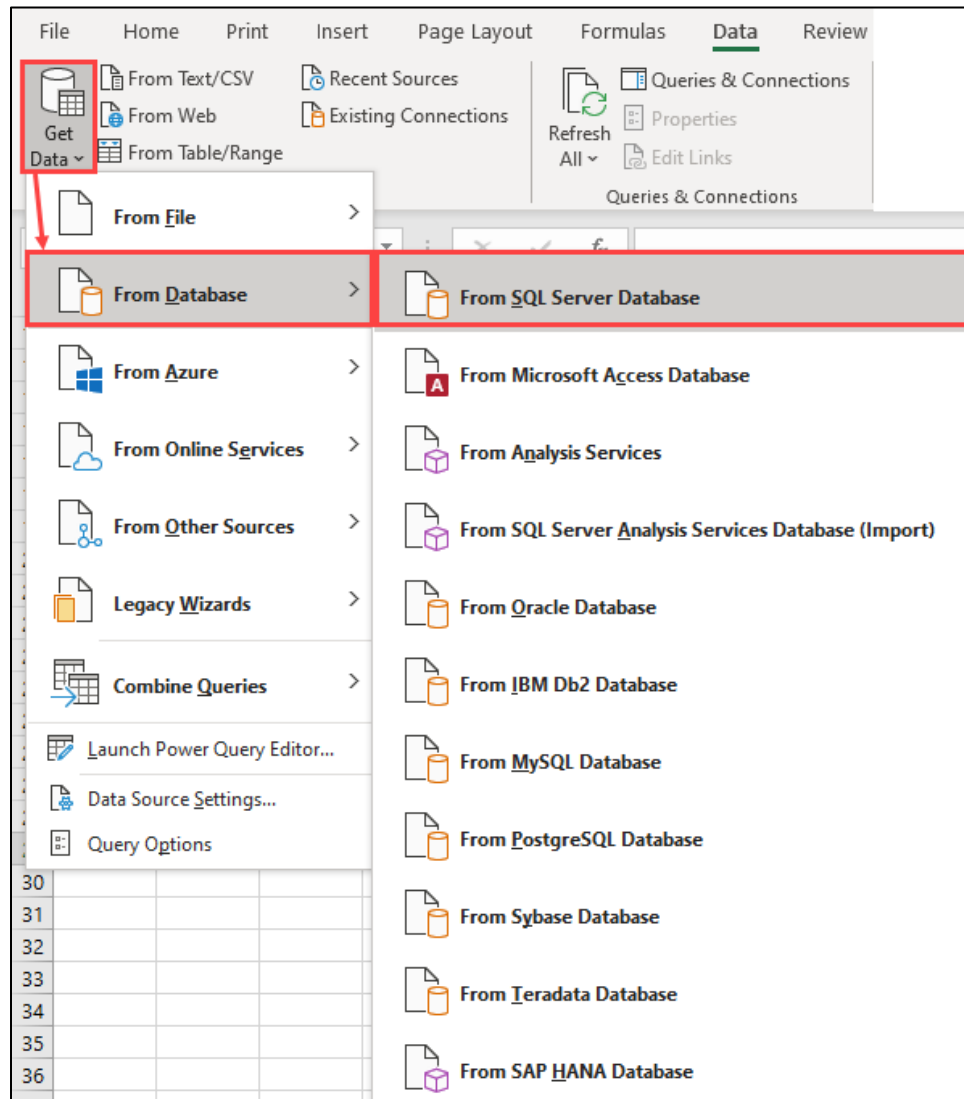


Figure 24 - Initiating a New Query Using Power Query

⁷ The instructions and screen captures related to Power Query reference Excel 2016 and newer. If you are using Excel 2013 or Excel 2010, your user interface may be slightly different than that pictured in this text; likewise, the step-by-step instructions for creating a query may be minimally different.

Enter the necessary details to identify the query's data source in the dialog box shown in **Figure 25**. Click **OK** to continue.



SQL Server database

Server ⓘ

A^BC ▾ DESKTOP-850KH6B\SQLEXPRESS

Database (optional)

A^BC ▾

Advanced options

OK Cancel

Figure 25 - Specifying the Data Source for Power Query

Next, expand the name of the database you wish to query so that you can see all the data available for the query. Then, select the items to include in your query and choose which **Load** option you wish to launch in the bottom right corner of the **Navigator** window, as shown in **Figure 26**. When choosing Load options, you can add the data to a table(s) in the workbook or create only the data connection. You can also specify whether the data should populate the workbook's data model regardless of your choice.

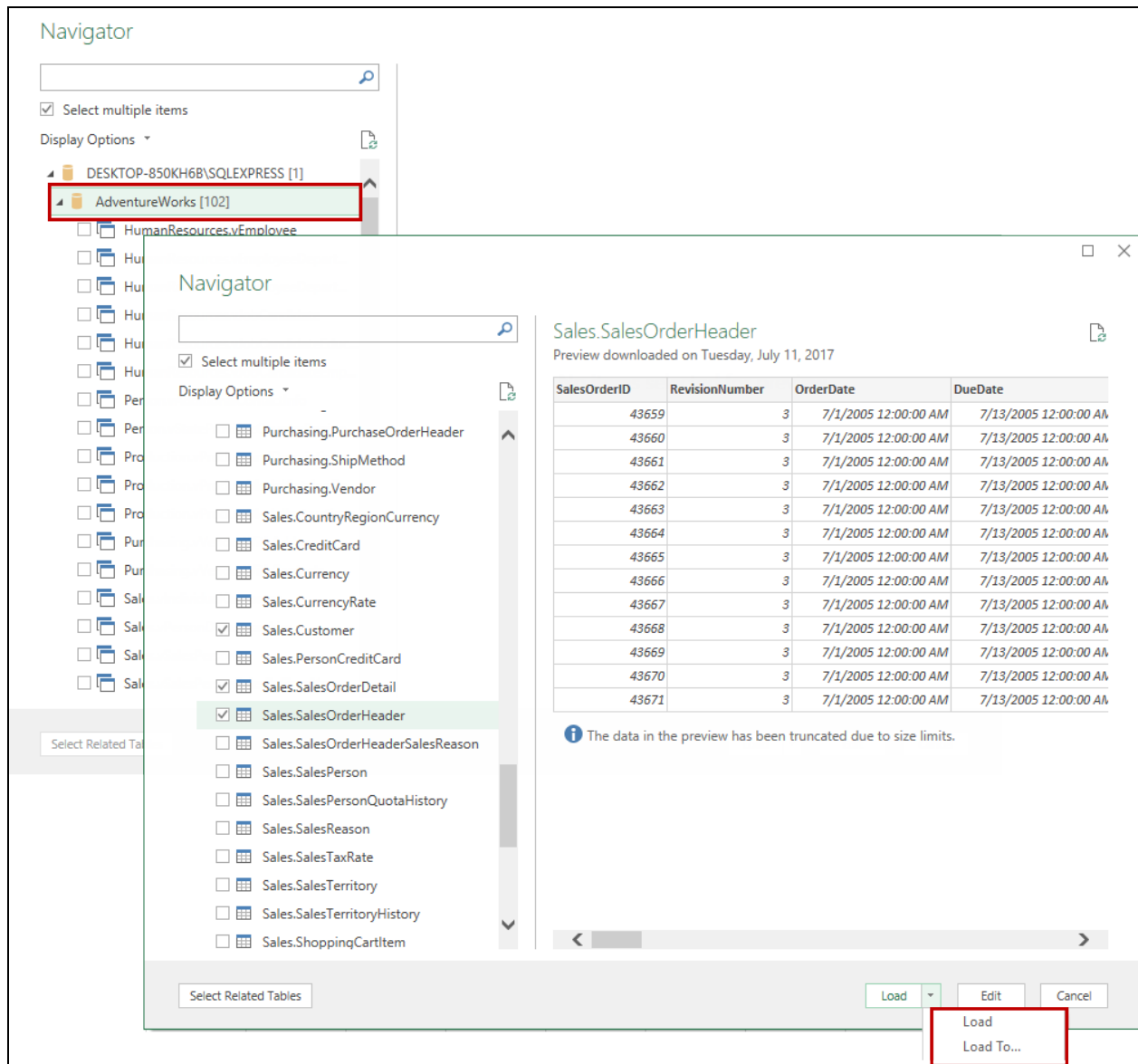
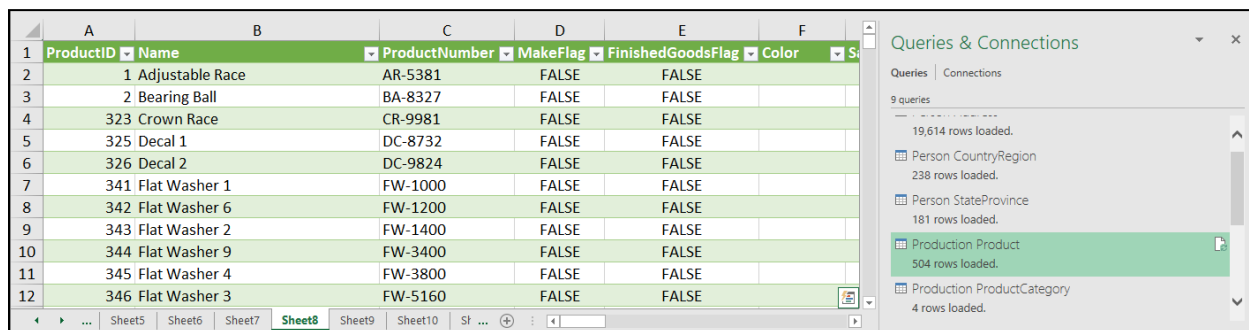


Figure 26 - Select Objects for Query in Power Query

In the example shown above, we will query nine tables.

1. Person Address
2. Person CountryRegion
3. Person StateProvince
4. Production Product
5. Production ProductCategory
6. Production ProductSubcategory
7. Sales Customer
8. Sales SalesOrderDetail
9. Sales SalesOrderHeader

We loaded the data as tables into the workbook and its data model. **Figure 27** presents a snapshot of the query's results.

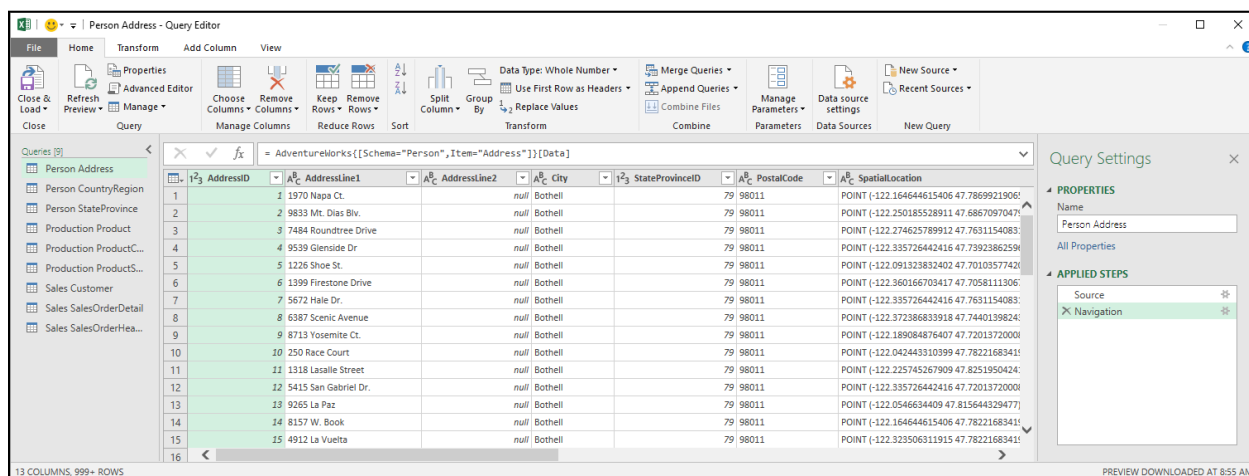


ProductID	Name	ProductNumber	MakeFlag	FinishedGoodsFlag	Color	StateProvinceID
1	Adjustable Race	AR-5381	FALSE	FALSE		
2	Bearing Ball	BA-8327	FALSE	FALSE		
323	Crown Race	CR-9981	FALSE	FALSE		
325	Decal 1	DC-8732	FALSE	FALSE		
326	Decal 2	DC-9824	FALSE	FALSE		
341	Flat Washer 1	FW-1000	FALSE	FALSE		
342	Flat Washer 6	FW-1200	FALSE	FALSE		
343	Flat Washer 2	FW-1400	FALSE	FALSE		
344	Flat Washer 9	FW-3400	FALSE	FALSE		
345	Flat Washer 4	FW-3800	FALSE	FALSE		
346	Flat Washer 3	FW-5160	FALSE	FALSE		

Figure 27 - Results of Using Power Query to Query Nine Tables into Excel

Editing a Query

In Figure 27, we selected to load the nine queries' results into tables and Excel's data model. You can edit the query before executing it by clicking the **Edit** button pictured in Figure 26. Doing so opens the **Query Editor**, a snapshot of which appears in **Figure 28**.



AddressID	AddressLine1	AddressLine2	City	StateProvinceID	PostalCode	SpatialLocation
1	1970 Napa Ct.	null	Bothell	79	98011	POINT (-122.164644615406 47.7869921906)
2	9833 Mt. Dias Blv.	null	Bothell	79	98011	POINT (-122.250185528911 47.6867097047)
3	7484 Roundtree Drive	null	Bothell	79	98011	POINT (-122.274625789912 47.7631154083)
4	9539 Glenside Dr	null	Bothell	79	98011	POINT (-122.335726442416 47.7392386259)
5	1226 Shoe St.	null	Bothell	79	98011	POINT (-122.091323832402 47.7010557742)
6	1399 Firestone Drive	null	Bothell	79	98011	POINT (-122.360166703417 47.7058111306)
7	5672 Hale Dr.	null	Bothell	79	98011	POINT (-122.335726442416 47.7631154083)
8	6387 Scenic Avenue	null	Bothell	79	98011	POINT (-122.372386833918 47.7440139824)
9	8713 Yosemite Ct.	null	Bothell	79	98011	POINT (-122.189084876407 47.7201372000)
10	250 Race Court	null	Bothell	79	98011	POINT (-122.042443310399 47.7822168341)
11	1318 Lasalle Street	null	Bothell	79	98011	POINT (-122.225745267909 47.8251950424)
12	5415 San Gabriel Dr.	null	Bothell	79	98011	POINT (-122.335726442416 47.7201372000)
13	9265 La Paz	null	Bothell	79	98011	POINT (-122.0546634409 47.815644329477)
14	8157 W. Book	null	Bothell	79	98011	POINT (-122.164644615406 47.7822168341)
15	4912 La Vuelta	null	Bothell	79	98011	POINT (-122.323506311915 47.7822168341)

Figure 28 – Power Query's Query Editor

In the Query Editor, you can make transformations to your data before executing the query by initiating such actions as adding and deleting columns of data, inserting formulas, applying different formats to your data, combining and splitting columns of data, removing rows, and even changing the source data for a query. Importantly, by default, Power Query saves your transformations in the **Applied Steps** box near the right side of the window and automatically applies them each time you execute or refresh a query.

To apply a simple transformation to the query shown in Figure 28, click anywhere in the **Spatial Location** column to select that field. Then, click **Remove Columns** from the **Home** tab in the

Query Editor. Upon doing so, Power Query removes that column of data from the query. Moreover, as shown in **Figure 29**, Power Query automatically creates an **Applied Step** that removes the Spatial Location column of data from the results each time the query is run or refreshed. Suppose you do not want the transformation applied when the query executes or updates. If so, you can delete it from Applied Steps by clicking the “X” next to its entry in Applied Steps. Once you create any desired transformations, click **Close & Load** on the Query Editor’s **Home** tab to load the results into your tables and data model.

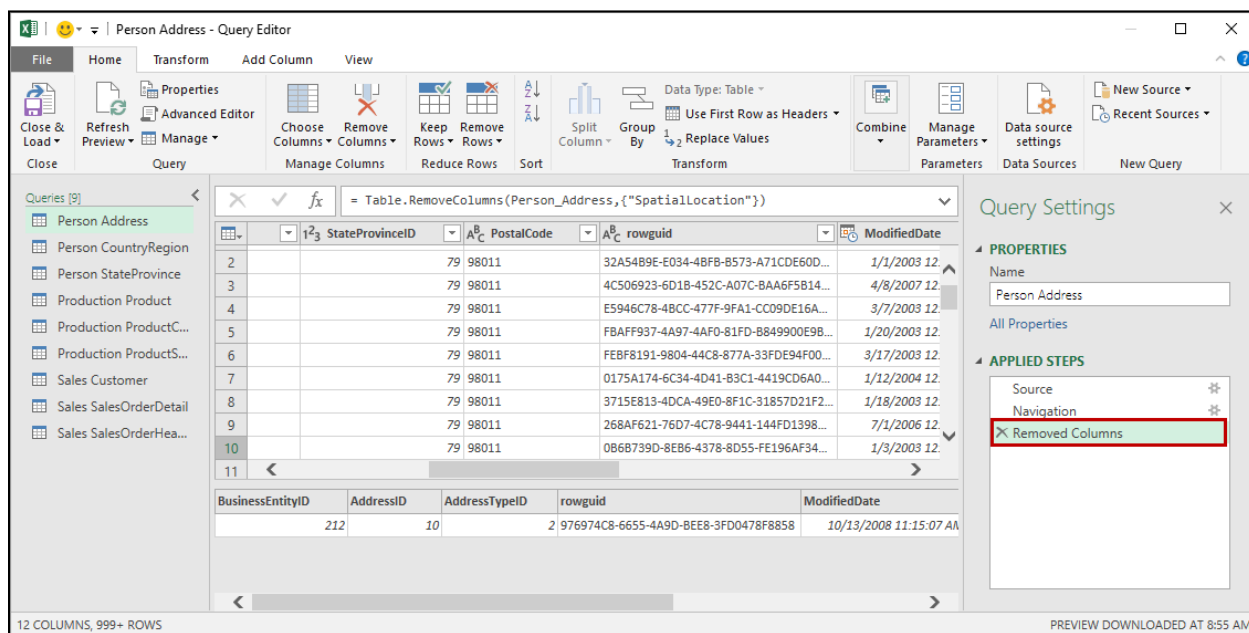


Figure 29 - Transforming Data and Creating an Applied Step in Power Query

Creating all your transformations when you initially build a query is unnecessary. Instead, you can construct transformations to fine-tune your queries at any time and to meet your evolving needs by right-clicking on the query in the **Workbook Queries** pane and choosing the **Edit** option, as shown in **Figure 30**.

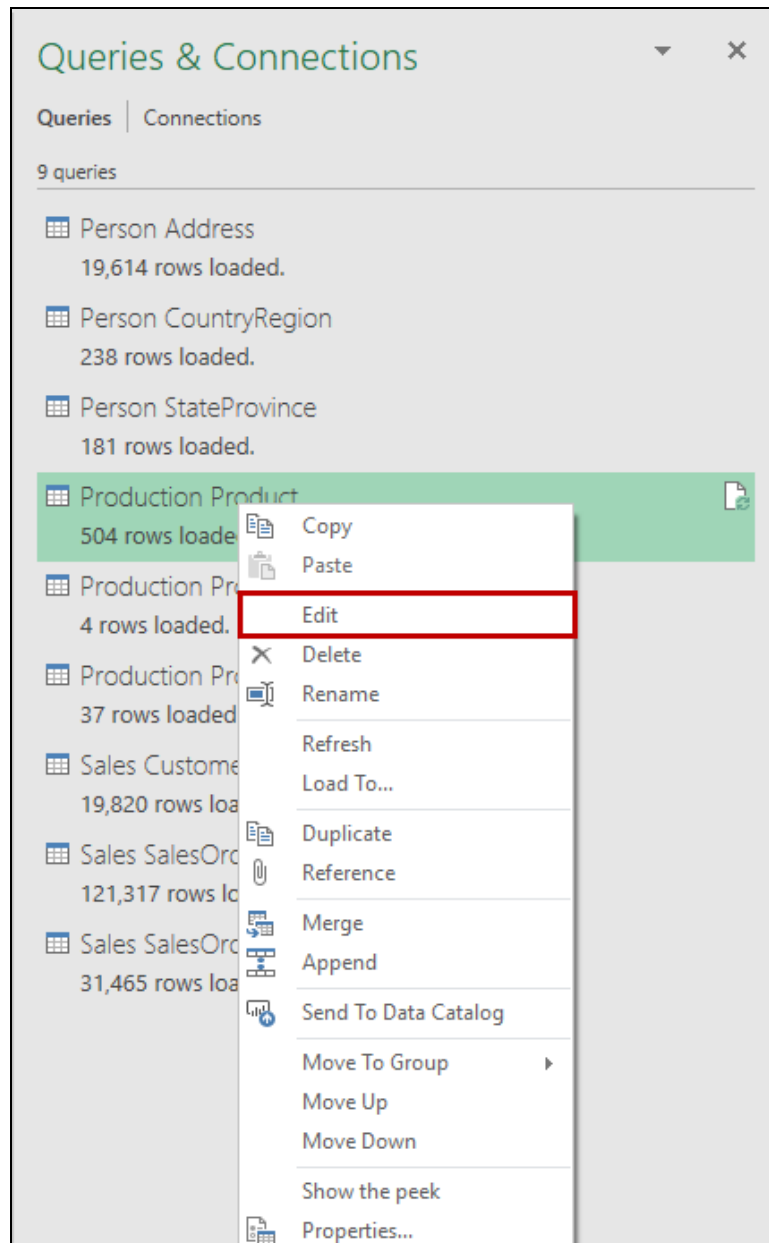


Figure 30 - Choosing to Edit a Query in Power Query

Linking Data from the Web Using Power Query

In addition to using Power Query to extract and link data from databases and other files into Excel, you can also use Power Query to link data from the Internet. For example, to query data from a web page into an Excel workbook, identify the web page from which you want to extract

the data and copy the URL for that page. In this example, we will use the following website to link foreign currency exchange rates into Excel.⁸

<https://www.x-rates.com/table/?from=USD&amount=1>

With the source website identified, choose **From Web** in the **Get & Transform** section of the Ribbon's **Data** tab. Note that this selection will produce the same dialog box as choosing **Get Data, From Other Sources, From Web**. **Figure 31** illustrates that dialog box.

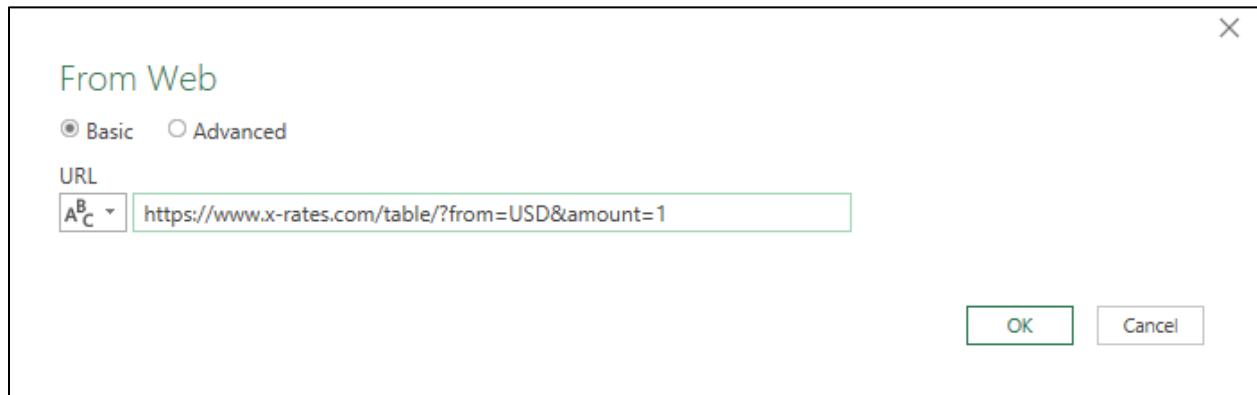


Figure 31 - Querying Data from the Web into Excel

After you paste the URL and click **OK**, Excel connects to the website of interest and displays a list of objects on that site to which it can link. Next, select the tables from which you want to query data into your workbook and choose the appropriate **Load** or **Load To** option near the bottom of the window to execute the query, as shown in **Figure 32**.

⁸ If you need to access exchange rates relative to the Canadian dollar, you should use the alternative site <https://www.x-rates.com/table/?from=CAD&amount=1>.

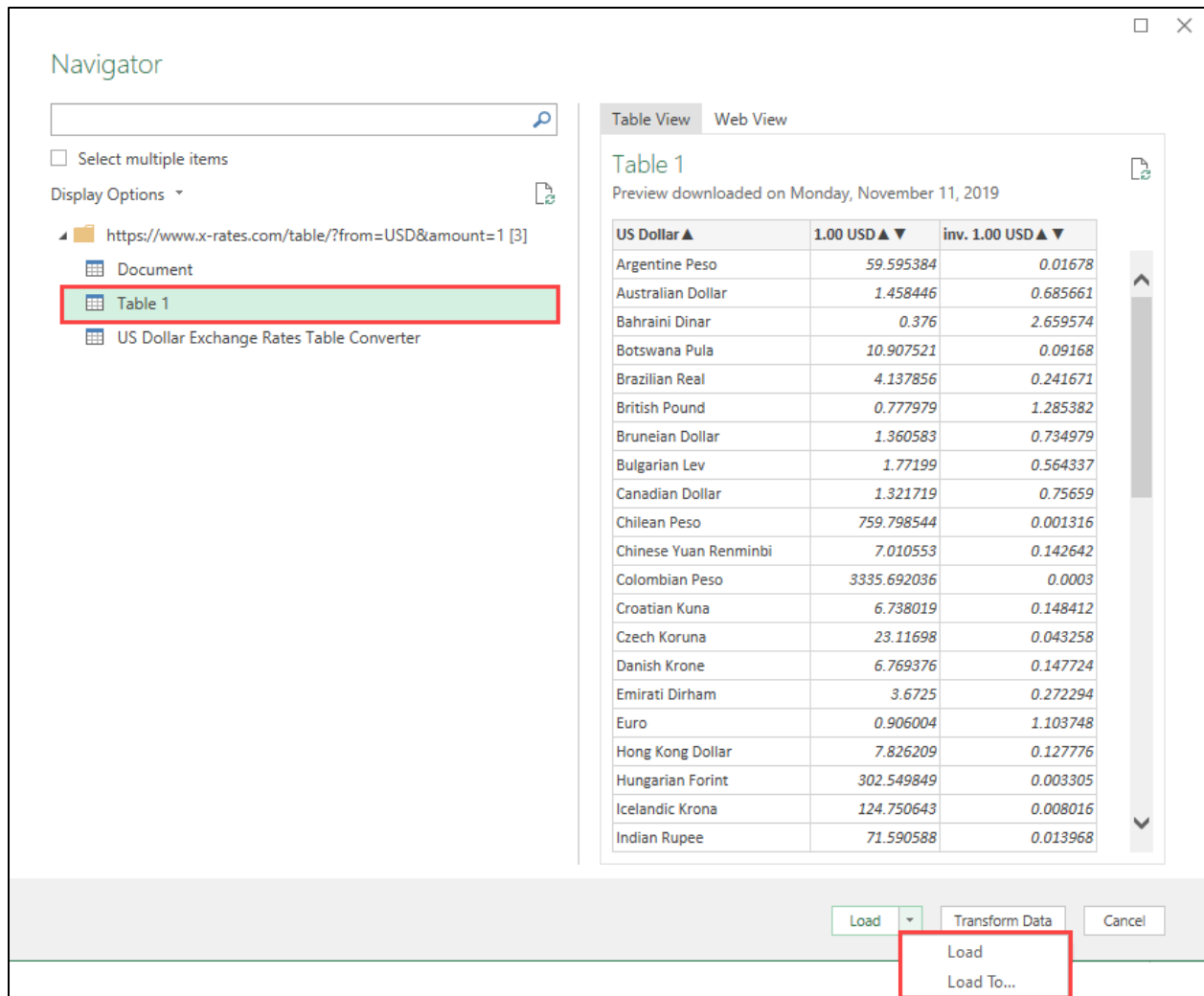


Figure 32 - Selecting and Loading Data from a Web Query

In the example above, we chose to link data from *Table 1* of the web page into an Excel workbook. **Figure 33** presents a subset of the query's results. Of course, now that the data exists in Excel, you can work with it as you usually would. Thus, you can use this data as you need to in Excel to achieve your reporting objectives.

	A	B	C
1	US Dollar ▲	1.00 USD ▲ ▼	inv. 1.00 USD ▲ ▼
2	Argentine Peso	61.732974	0.016199
3	Australian Dollar	1.5152	0.659979
4	Bahraini Dinar	0.376	2.659574
5	Botswana Pula	11.051783	0.090483
6	Brazilian Real	4.388033	0.227893
7	British Pound	0.769048	1.300309
8	Bruneian Dollar	1.399384	0.7146
9	Bulgarian Lev	1.797245	0.556407
10	Canadian Dollar	1.328933	0.752484
11	Chilean Peso	813.027473	0.00123
12	Chinese Yuan Renminbi	7.026868	0.142311
13	Colombian Peso	3436.342353	0.000291

Figure 33 - Results of a Web Query Generated by Power Query

?

How can – and should – the prospect of using Power Query to establish real-time links to your data change your reporting processes?

Adding a Simple Transformation to a Query

Continuing with this example, suppose you wanted to query exchange rates only for the following currencies relative to the United States dollar.

- Australian Dollar
- British Pound
- Canadian Dollar
- Euro
- Japanese Yen
- Mexican Peso

Further, you want to ensure that these are the only six currencies linked into Excel each time you refresh your query. To make the necessary change that facilitates this action, right-click on the query in the Queries & Connections pane and choose **Edit** to open the **Power Query Editor** shown in **Figure 34**.

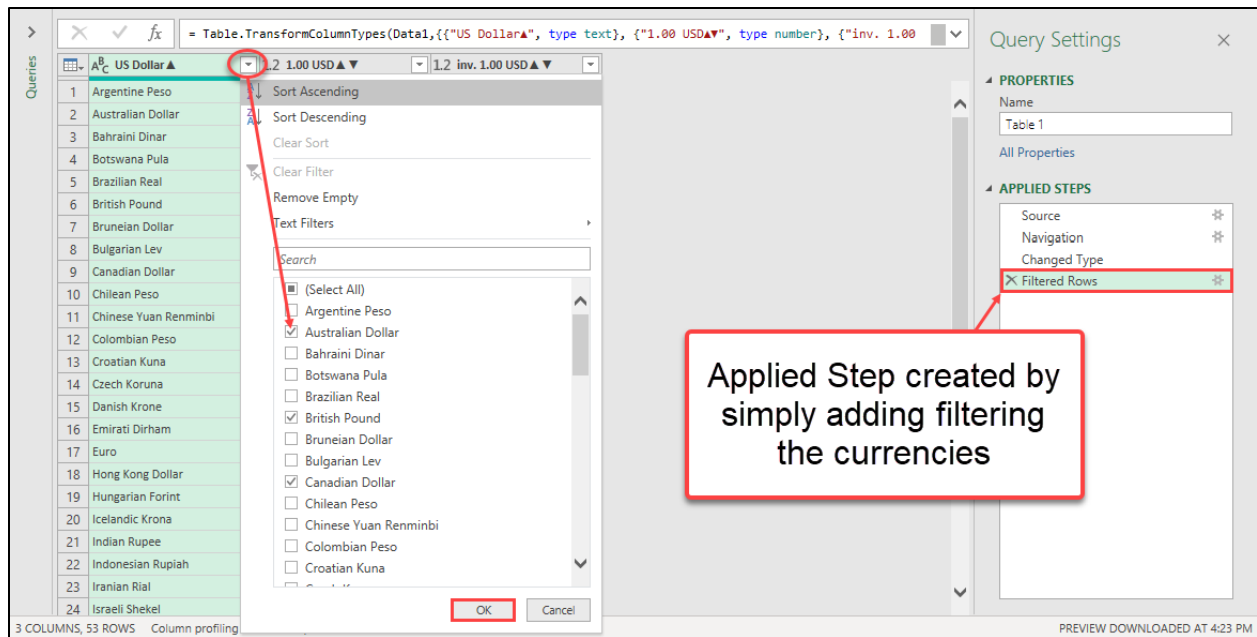


Figure 34 – Editing a Query in the Power Query Editor

In the Power Query Editor, click the drop-down arrow in the *US Dollar* column and apply filters to select only the desired currencies. Click **OK** to complete your edit. Upon doing so, you will see that the list of currencies the query returns reduces to the six items selected. Equally important, on the right side of the window in the **Query Settings** pane, you will notice an entry entitled *Filtered Rows* in **Applied Steps**. This entry signifies that Power Query has “memorized” the filters and saved them with the query. Power Query applies these steps every time the query runs or refreshes. Should it become necessary, you can edit an Applied Step by clicking the “gear” icon to the Applied Step’s right. Likewise, you can delete an Applied Step by clicking the “X” icon on the Applied Step’s left. Finally, you can rearrange the Applied Steps order of execution in a query by clicking and dragging the applied steps vertically within the Applied Steps list.

Examples of Useful Power Query Transformations

Indeed, it is far beyond the scope of this course to cover every possible type of transformation you can create with Power Query. However, we would be remiss if we did not provide several examples of transformations you may find helpful when manipulating your data.

Pivoting Data with Power Query

Often, the need to pivot data arises – for example, you have data in a tabular format and want to create a chart from that data. For this example, we will use the *Expenses* table shown in **Figure 35** as the basis for a chart.

File

Home

Insert

Draw

Page Layout

Table Name:

Expenses

Summarize with PivotTable

Remove Duplicates

Convert to Range

Insert Slicer

Resize Table

Properties

Tools

On

B8

	A	B	C	D
1	Month	Account	Amount	
2	1/31/2019	Food	25562.79	
3	1/31/2019	Beverages	5098.91	
4	1/31/2019	Food	25562.79	
5	1/31/2019	Payroll Exper	13658.24	
6	1/31/2019	Rent	2400	
7	1/31/2019	Insurance	427.33	
8	1/31/2019	Supplies	1510.8	
9	1/31/2019	Utility Expen	1541.68	
10	2/28/2019	Food	23221.12	
11	2/28/2019	Beverages	4448.02	

Figure 35 - Data for Power Query Pivot

The steps to transform the data by pivoting it and returning it to Excel are as follows:

1. Select any cell in the table and then click **From Table/Range** on the Ribbon's **Data** tab to open the Power Query Editor.
2. With the **Month** column selected, click **Date** in the **Data Type** drop-down list on the Ribbon's **Home**. Then select **Replace current** in the dialog box, as shown in **Figure 36**.

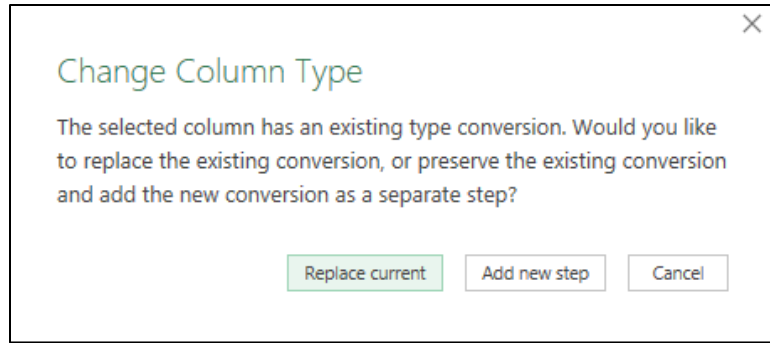


Figure 36 - Change Column Type

3. Select the **Account** column, which will provide the values to use across the chart's columns when the query completes. Then, on the Ribbon's **Transform** tab, click **Pivot Column**.
4. In the **Pivot Column** dialog box in **Figure 37**, select **Amount** as the **Values Column** and click **OK**.

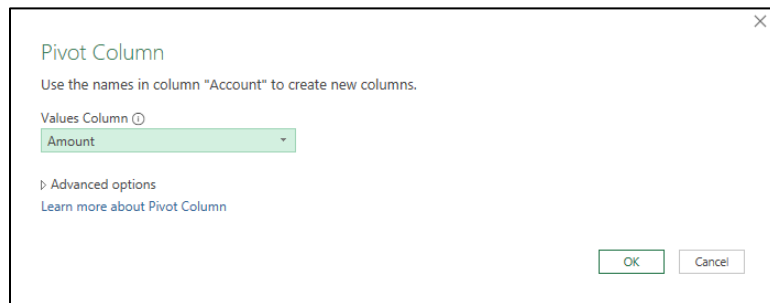


Figure 37 - Values Column

5. With the data pivot completed in Power Query, select **Close & Load** on the **Home** tab to return the data into Excel.

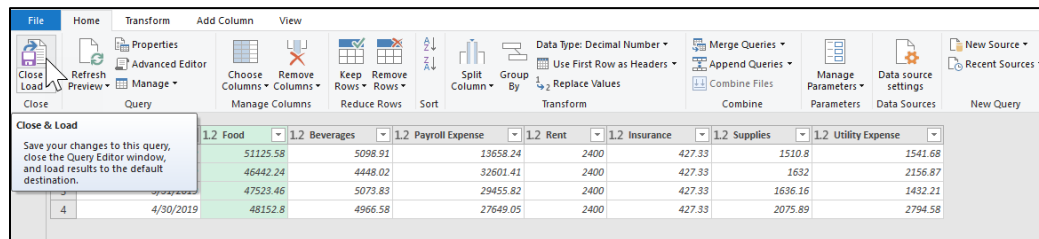


Figure 38 - Pivot Data Close & Load

Using the data in the Excel table, we can quickly create a chart similar to the one pictured in **Figure 39**. Notably, refreshing the query updates the chart automatically when users add new data to the Expense table.

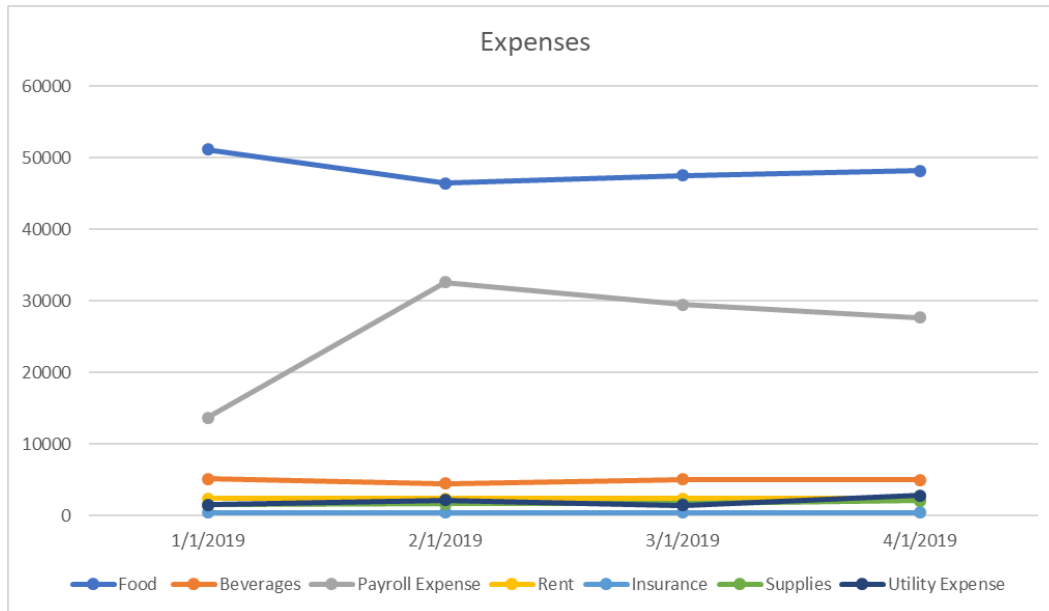


Figure 39 - Expense Chart Resulting from Transformed Using Power Query

Unpivoting Data

Exported reports often provide data in a format that requires us to unpivot it before using the information in our analyses. Using Power Query to unpivot the data will allow us to complete the steps quickly. Further, as new data appears, we can refresh the query to get updated, unpivoted results.

For this example, we will use data exported from Sage 50 US; specifically, we will use the *Customer Sales by Month* report, previously exported to Excel. The report provides monthly results in a single row for each customer. Instead, we need to have the single row – which provides a date dimension – separated into unique rows for each data point.

The steps to complete the unpivot are as follows.

1. Click any cell in the data and then click **From Table/Range** on the Ribbon's **Data** tab.
2. The **Create Table** dialog box opens. If the data is not already a table, click **OK**.
3. The Power Query Editor opens. Click the **Total** column's heading to select that column. Then click **Remove Columns** on the **Home** tab to delete the unnecessary column.
4. Double-click the heading for **Column1** and change the name to **Customer**.
5. On the **Home** tab, click the drop-down arrow in **Remove Rows** and select **Remove Bottom Rows**. For the number of rows, enter **"1"** and click **OK** to remove the total row.

6. With the **Customer** column highlighted, go to the **Transform** tab and click the drop-down arrow for **Unpivot Columns**; then click **Unpivot Other Columns**. Figure 40 illustrates the results of these first steps.

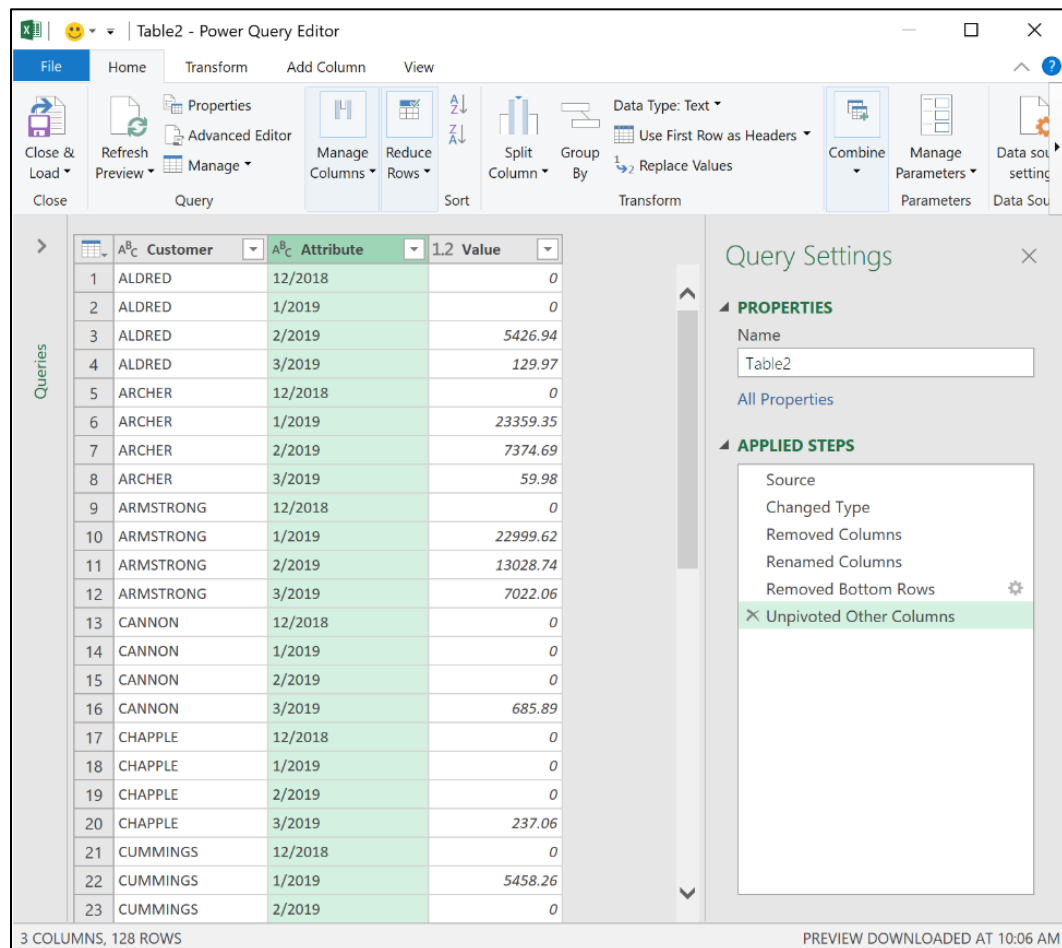


Figure 40 – Unpivoting Data in Power Query

7. Figure 40 illustrates small icons at the front of each header. These icons indicate the type of data present in each column. The header indicates that the data is textual for the date column, as shown by the icon . Therefore, we will need to change it to the “date” data type by selecting the column and, on the **Home** tab, clicking the drop-down arrow for **Data Type**. Next, choose the **Date** option so that the icon in the header indicates the presence of date data .
8. The final step is to click **Close & Load** on the Ribbon’s **Home** tab to load the transformed data into Excel.

The list of **Applied Steps** in the Query Settings panel shown in Figure 40 provides an easy way to delete, reorder, and edit a step.

Any changes to the data appear in the results upon refreshing the query. Click **Refresh All** on the **Data** tab of the **Ribbon** to update the query results. Alternatively, you can click the **Refresh** icon in the **Queries & Connections** panel, as shown in **Figure 41**.

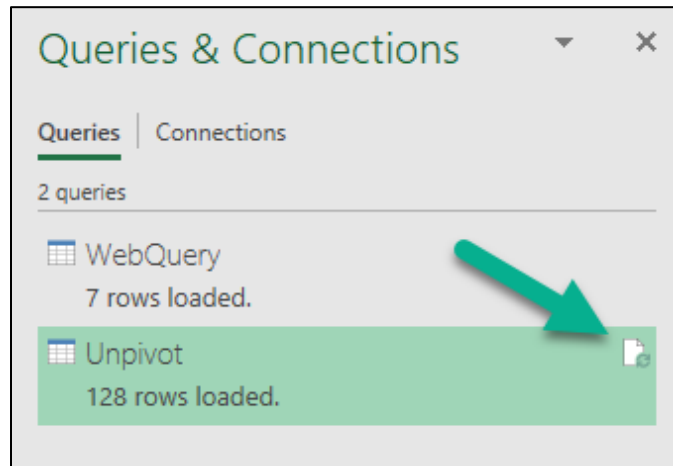


Figure 41 – Power Query’s Refresh Icon

Splitting Data into Multiple Columns with Power Query

There are multiple ways to break data in a cell, including **Flash Fill**, **Text to Columns**, and using functions such as **LEFT**, **MID**, **RIGHT**, and **LEN**. However, with both Flash Fill and Text to Columns, you must re-run these utilities each time your data changes – a process that may be impractical in some situations. In addition, although formula-based approaches update dynamically, formulas created using LEFT, MID, RIGHT, LEN, and other functions are challenging for many Excel users to construct. Fortunately, Power Query provides an easy and practical way to split data into multiple columns.

With our data in the table shown in **Figure 42**, we will use Power Query to split the *City*, *State*, and *Zip Code* into separate columns.

City, State, Zip
Jackson, MS 39236
Hammond, LA 70401
New Orleans, LA 74501
Centerville, UT 84014
Atlanta, GA 30326
Maunaloa, HI 96770
Slidell, LA 70458
Hammond, LA 70403
Baton Rouge, LA 70559

Figure 42 – Data for Split Data Example

The steps to complete the process are as follows.

1. Select any cell in the table/range of data, and click **From Table/Range** on the Ribbon's **Data** tab to open the **Power Query Editor**.
2. Click **Split Column** on the Ribbon's **Home** tab and select **By Delimiter**. For the first instance, use **Comma** and choose **Each occurrence of the delimiter**, as shown in **Figure 43**. Then, click **OK** to confirm your entry.

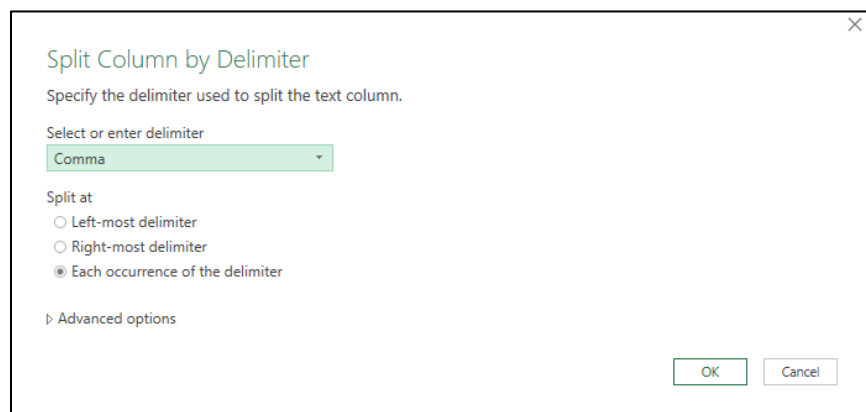


Figure 43 - Split Column by Delimiter

3. Repeat the **Split Column** process outlined above on the new column created and select **space** to split the State and Zip.
4. Select the blank column and click **Remove Columns** on the Ribbon's **Home** tab.
5. Double-click each header and rename them to *City*, *State*, and *Zip*, respectively.
6. Select the **Zip** column and select **Text** using the **Data Type** drop-down on the Ribbon's **Home** tab.
7. Click **Close & Load** to return the data as a table to Excel.

To include any new data added to the source table, right-click on the table created by the query, followed by clicking **Refresh**.

Transforming Data by Adding Columns

Power Query provides a simple interface from which to add a column of data. In this illustration, we will use Power Query's **Column From Examples** feature to add a column of data to the data model. To add a column, follow the steps detailed below.

1. Click on any cell in range or table, and click **From Table/Range** on the Ribbon's **Data** tab to open the **Power Query Editor**.
2. On the **Add Column** tab of the Ribbon of the Power Query Editor, click **Column From Examples**.
3. We will combine the *Department* and *Account number* for the first new column by typing the department name and account number on the first line in the new column and pressing **Enter**. This process resembles using **Flash Fill** in Excel; however, it completes an editable function shown in **Figure 44**. Review the results and click **OK**.

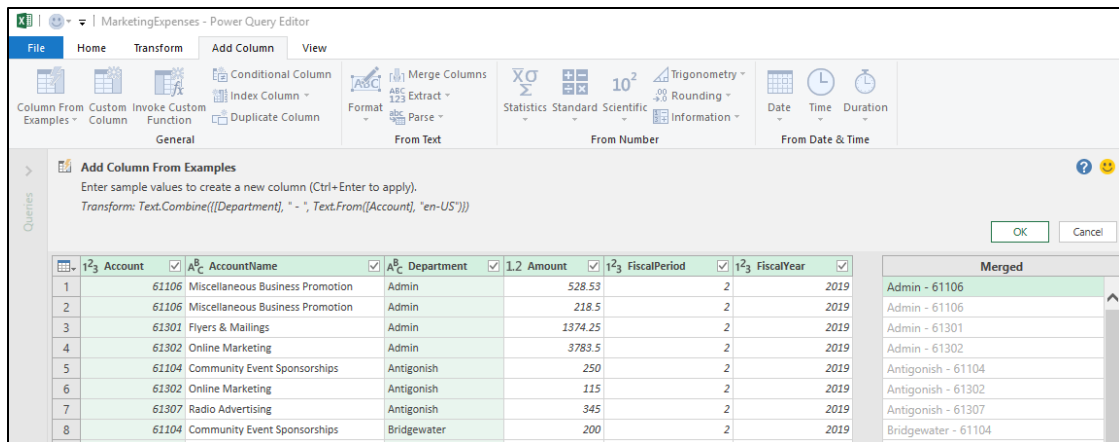


Figure 44 - Add Column From Examples

- We will now add a second column using the same process to combine the **FiscalPeriod** (month) with the **FiscalYear**. Because no day of the week is specified, we will input the first day of the month, as shown in **Figure 45**. Click **OK** to complete the process, and then click the text icon  in the header for the column to change the type to *date*. Next, rename the column to *Date*.

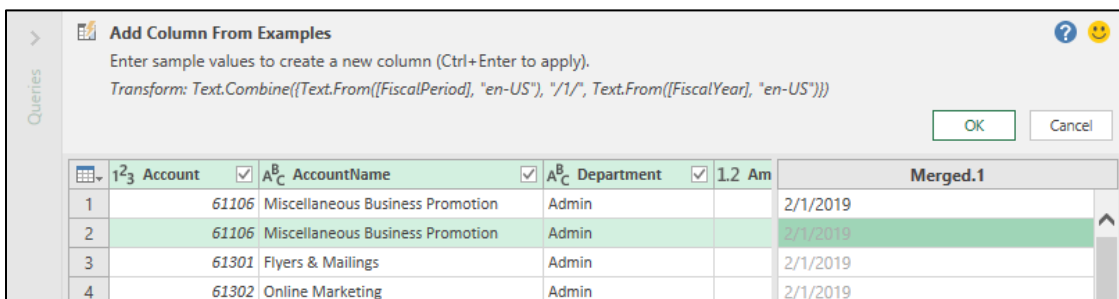


Figure 45 - Adding Second Column

- To allow for reporting on a month-end basis, add another column using **Custom Column** on the Ribbon's **Add Column** tab. To do so, complete the dialog box as shown in **Figure 46** and click **OK**.

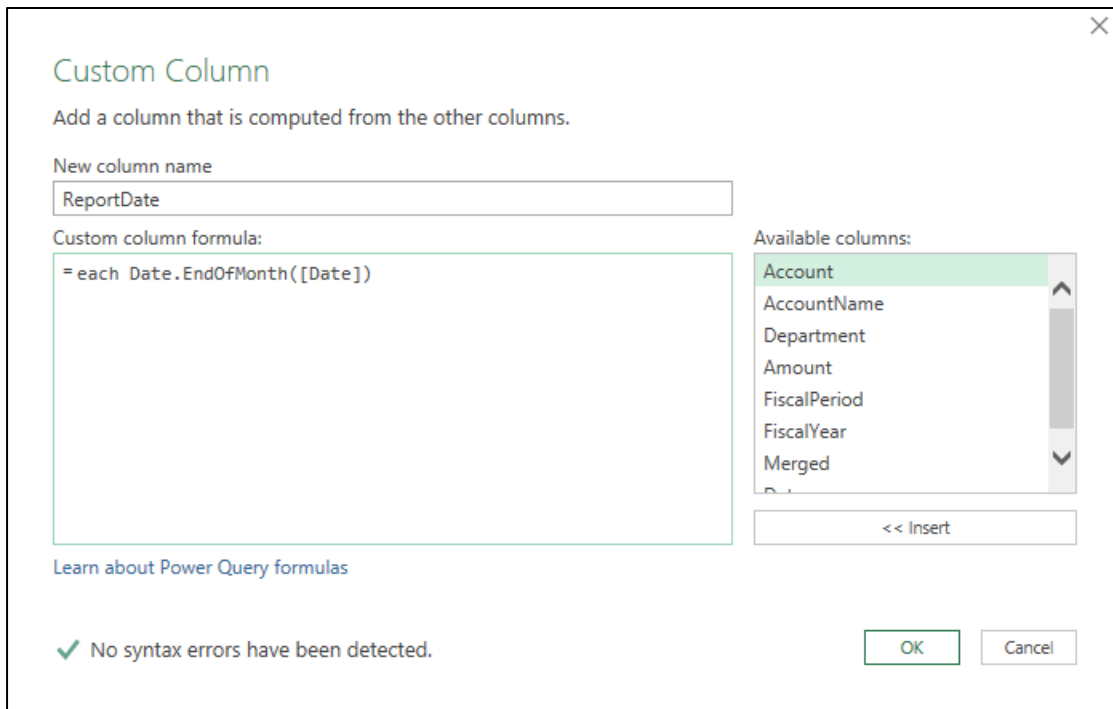


Figure 46 - Custom Column

6. Change the column type to *date*.
7. On the Ribbon's **Home** tab, click **Close & Load** to complete the process.

As with all Power Queries, the steps created automatically re-run upon refreshing the query.

Using Power Query to Group Data by Fiscal Periods

Excel date features operate on calendar years instead of user-defined fiscal years. However, you can convert calendar dates into fiscal reporting periods using Power Query. To illustrate, the Power Query illustration that follows adds two custom columns to identify the fiscal year and the fiscal quarters in which transactions should group, assuming a March 31st year-end. The following are the steps for adding the two new columns.

1. First, convert the formatting of the *Date* column to **Date only**.
2. We will use the **Custom Column** command on the Ribbon's **Add Column** tab for both columns. Enter the name **F_Quarter** in the **New column name** field.
3. In the **Custom Column formula** field, add the formula pictured in **Figure 47**. This formula examines a transaction's date and then computes the appropriate fiscal quarter based on that date.

Custom Column

Add a column that is computed from the other columns.

New column name
F_Quarter

Custom column formula:

```
= "Qtr " &
Text.From(Number.RoundUp((
if Date.Month([Date])>3
then Date.Month([Date])-3
else Date.Month([Date])+9)/3))
```

Available columns:

- Date
- Cheque Number
- Amount
- Vendor
- Account

<< Insert

[Learn about Power Query formulas](#)

✓ No syntax errors have been detected.

OK Cancel

Figure 47 - Formula to Calculate Quarter

- Next, add a second custom column, entitling it **F_Year**. Then, use the formula shown in **Figure 48** to calculate the appropriate fiscal year.

Custom Column

Add a column that is computed from the other columns.

New column name
F-Year

Custom column formula:

```
= Text.From(
if Date.Month([Date])>3
then Date.Year([Date])+1
else Date.Year([Date]))
```

Available columns:

- Date
- Cheque Number
- Amount
- Vendor
- Account
- F_Quarter

<< Insert

[Learn about Power Query formulas](#)

✓ No syntax errors have been detected.

OK Cancel

Figure 48 - Formula for Fiscal Year

- Establish each column as text so we can use them as labels.

6. **Close & Load** to complete the query.

The Power Query M formula language resembles Excel formulas enough so that most users can decipher the above functions. *Text.From* returns the text representation of a number, date, time, datetime, datetimezone, logical, duration, or binary value. The *if* statement adjusts the month number to reflect the fiscal month and determine the correct fiscal year. The function *Date.Month* returns the month number and *Date.Year* returns the year of a date cell. Be aware that the M formula language is *case-sensitive*. For a complete listing of functions, check out the **Power Query M Reference** at <https://k2e.fyi/MFunctions>.

Creating a Transformation to Transpose Data

Excel provides the **Paste Special, Transpose** feature, and the **Transpose** function. Both these tools are helpful for transposing data by “rotating” it 90 degrees. However, Paste Special, Transpose is a one-time action, and the Transpose function does not work if the range changes. These two issues severely limit the usefulness of these two tools. Fortunately, Power Query eliminates both limitations with its **Transpose** option.

The data is not in a table in this first example of using **Transpose** in Power Query. Choose **From Table/Range** on the Ribbon’s Data tab to begin the process of transposing data. Next, you should uncheck the box labeled **My Table has Headers**, as shown in **Figure 49**.

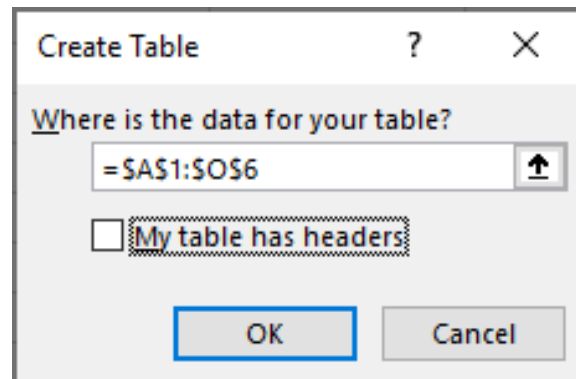


Figure 49 - Uncheck My table has headers

Complete the transpose process using the following steps with the data now in the Power Query Editor.

1. On the **Transform** tab of the Ribbon, click **Transpose**.
2. On the same tab, click **Use First Row as Headers**.
3. On the **Home** tab, select **Remove Rows, Remove Bottom Rows**, and choose to remove only one row. Click **OK**.

4. To complete the transpose, click **Close & Load**, and upon doing so, Power Query transposes the data. If the original data changes, right-click anywhere on the “results” data and choose the **Refresh** option to update the results.

Combining Excel Worksheets with a Power Query Transformation

We will combine expense data from one worksheet with budget data from another worksheet using Power Query in the following example. For both sets of data, use **From Table/Range** to create a connection to the data. Create the connection for the **Sales** data. Then, in the **Query Settings** panel of the **Power Query Editor**, change the query’s **Name** to **Actuals**.

Next, repeat the previous process for the **Budget** data and name it **Budget**. Then, click the **Close & Load** drop-down arrow with each query and select **Close & Load To**. This action opens the **Import Data** dialog box. There, select **Only Create Connection**, as shown in **Figure 50**. Finally, complete these processes by clicking **OK**.

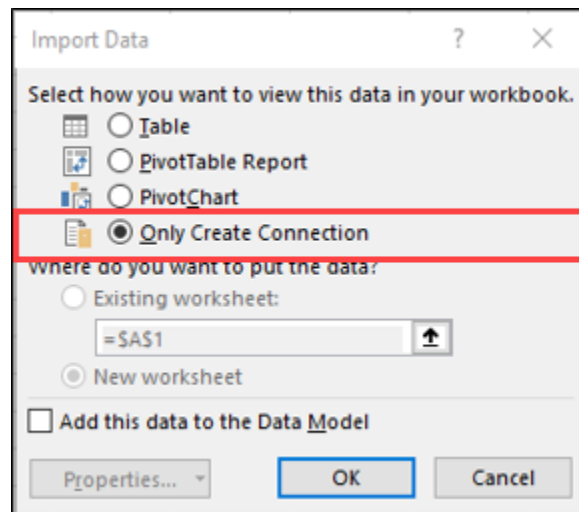


Figure 50 – Creating a Connection to Data Using Power Query

With the two queries completed, you can combine the two worksheets using the following steps.

1. In the **Get Data** drop-down list, select **Combine Queries** followed by **Append**.
2. Configure the **Append** dialog box, as shown in **Figure 51**, and click **OK**.

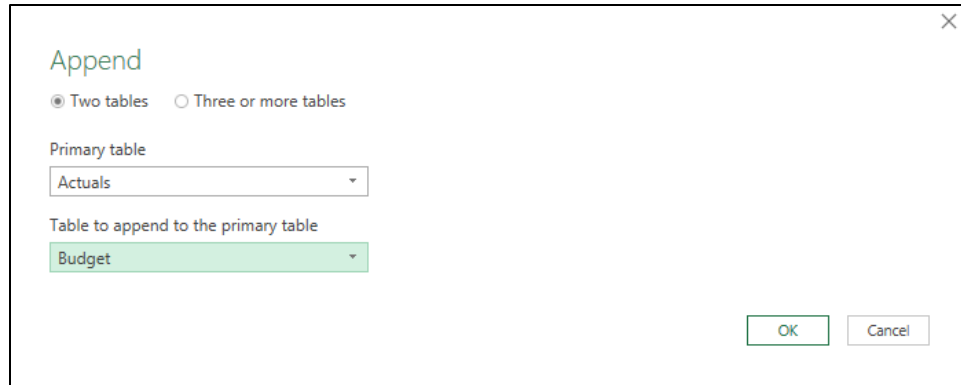


Figure 51 – Appending Query Results

3. Next, select the **Month** and **Product** columns and click **Group By** on the **Home** tab.
4. Configure the **Group By** to sum the **Sales** and **Budget** columns, as shown in **Figure 52**, and click **OK**.

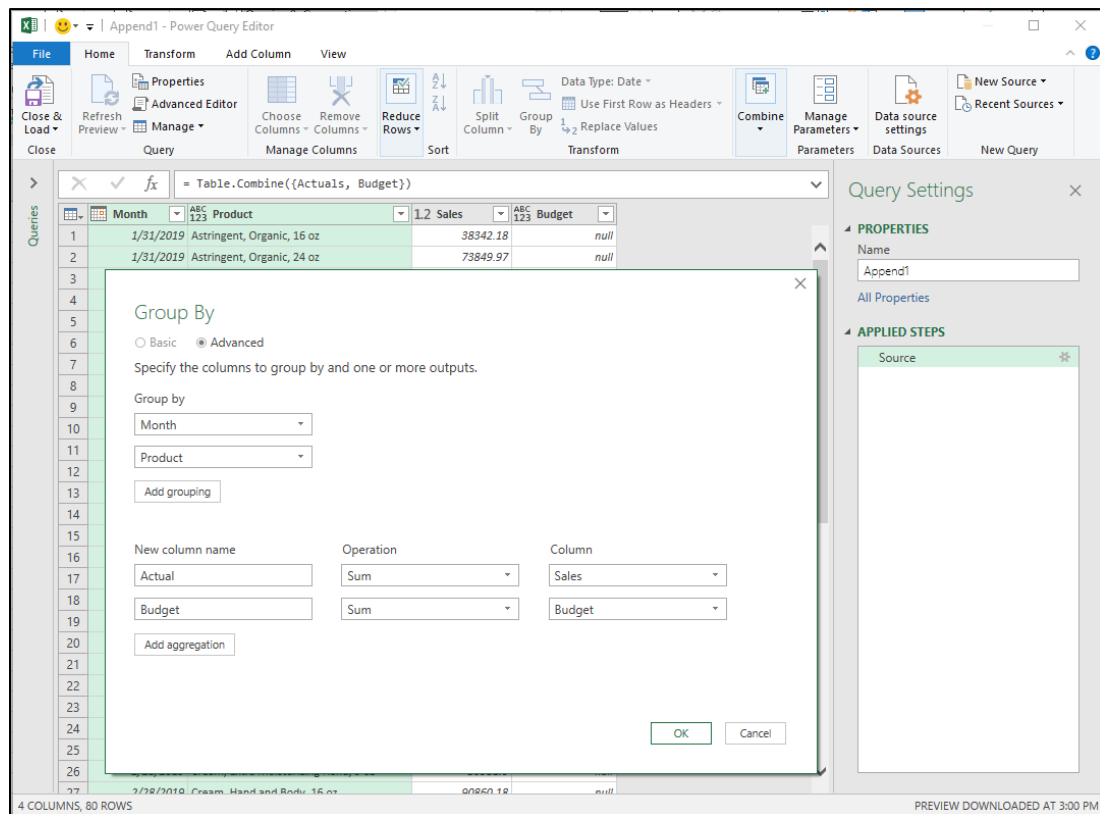


Figure 52 – Grouping Data from Query Results

5. In the **Query Settings** panel, change the query's name to **Actual vs. Budget**, review the data types, and make any changes required.

6. Click **Close & Load** to return the results of combining the worksheets to Excel.

Querying All Data from a Folder

Many reporting processes involve updating historical transactional data with newer data periodically – typically monthly or quarterly. Often, these updates occur through manual copy-and-paste procedures to add last month's (or quarter's) data to all preexisting data. Additional procedures are necessary to update and reformat the report. Fortunately, you can streamline this process significantly with Power Query.

To begin, query the first set of data using Power Query. In this case, from the **Data** tab of the Ribbon, choose **Get Data**, **From File**, and **From Folder**. Then, select the folder that contains the data for the first reporting period. Finally, as shown at the bottom of the window pictured in **Figure 53**, choose **Combine & Load To**.

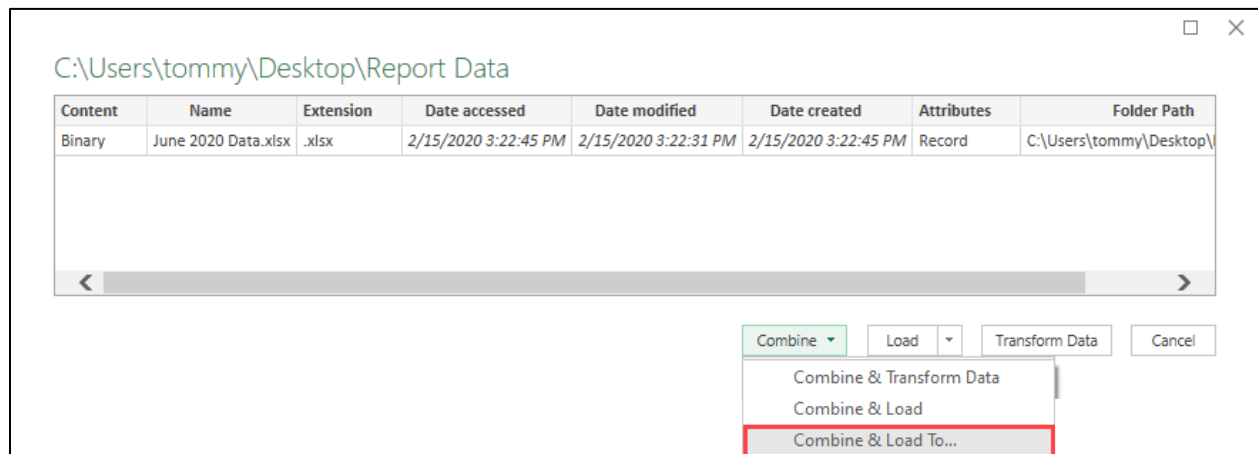


Figure 53 - Querying Data from a Folder

Click the file name under **Parameters** in the following dialog box and click **OK**. Finally, select **PivotTable Report** in the **Import Data** dialog box, as shown in **Figure 54**, and click **OK**.

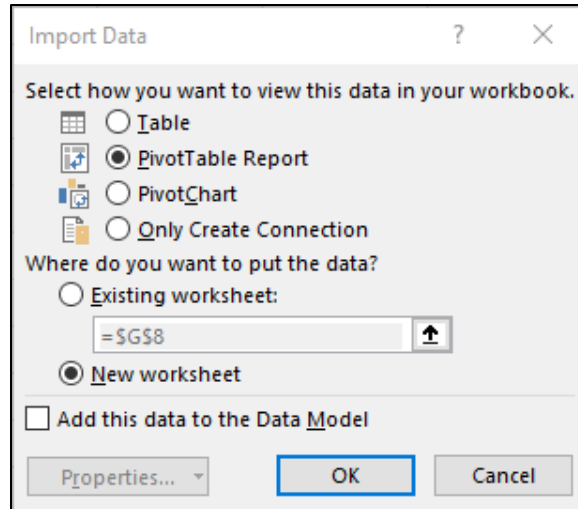


Figure 54 - Importing Data as a PivotTable Report Using Power Query

Finally, arrange your PivotTable as desired, perhaps as shown in **Figure 55**, and save it.

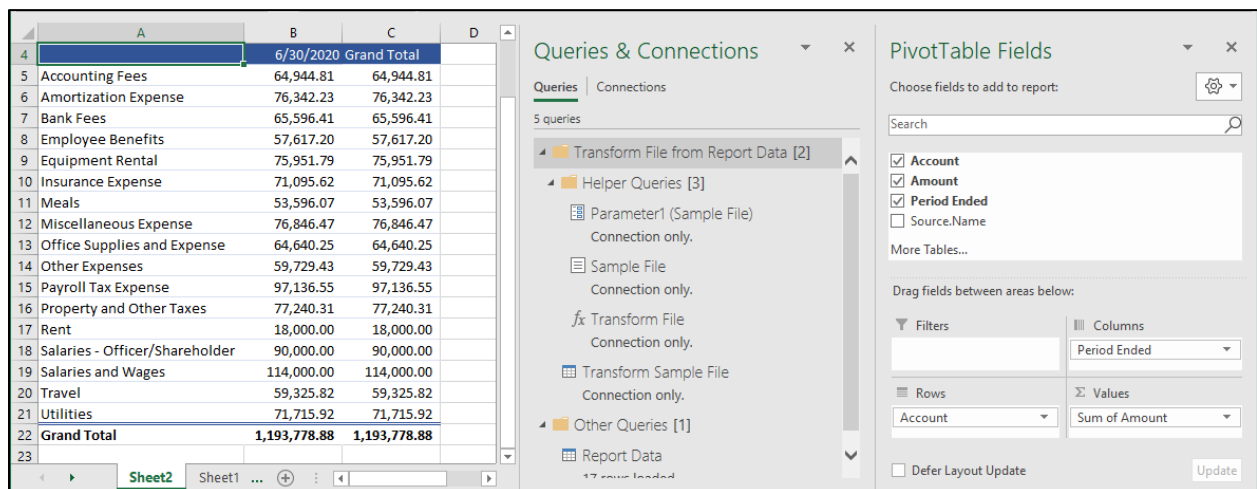


Figure 55 - Initial PivotTable

Let us assume that a month has passed, and new data is available – in this example, for July. To update your report, copy and paste the workbook containing the new data into the folder where the previous month’s data resides. Next, open the workbook containing your PivotTable and click **Refresh All** on the Ribbon’s **Data** tab. You will see that the report updates immediately, as shown in **Figure 56**.

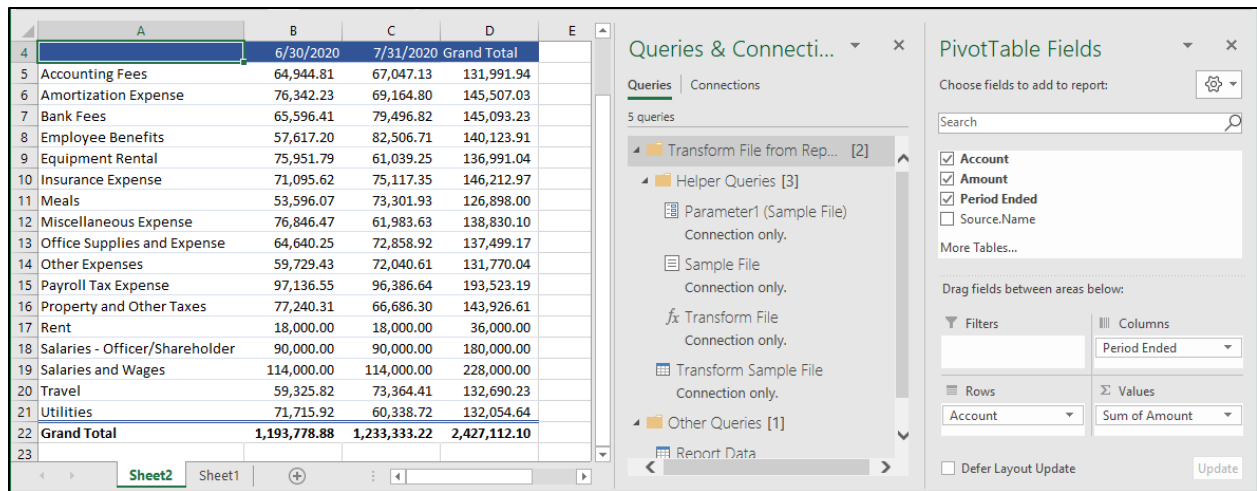


Figure 56 - Updated Report by Simply Saving New Data in the Same Folder as Historical Data and Choosing to Refresh All

? How will the ability to link all data from a folder potentially alter some of your reporting processes? Are there efficiencies to gain from this practice?

Preparing Consolidated Financial Statements with Power Query

In our final example of using Power Query to effect transformations, let us assume Terry's role as the CFO for a real estate development and management company. Terry must prepare financial statements that consolidate results for three limited partnerships – *ABC*, *Vista Ridge*, and *Mountain Vista*. Terry's accounting application has provided the trial balance for each company in an Excel worksheet. Further, the trial balance includes a column indicating the financial reporting category for each account. Therefore, Terry's challenge is to combine the three trial balance reports and build a consolidated income statement and a consolidated balance sheet. Let's see how Power Query can assist with this type of project.

First, Terry saves the three entity-specific worksheets in an Excel workbook as tables. Then, Terry clicks on the Ribbon's **Data** tab, followed by **Get Data**. Next, Terry chooses **From Other Sources**, **Blank Query** to open the **Power Query Editor**, as shown in **Figure 57**. Terry also adds the formula **=Excel.CurrentWorkbook()** to load all three tables into Power Query.

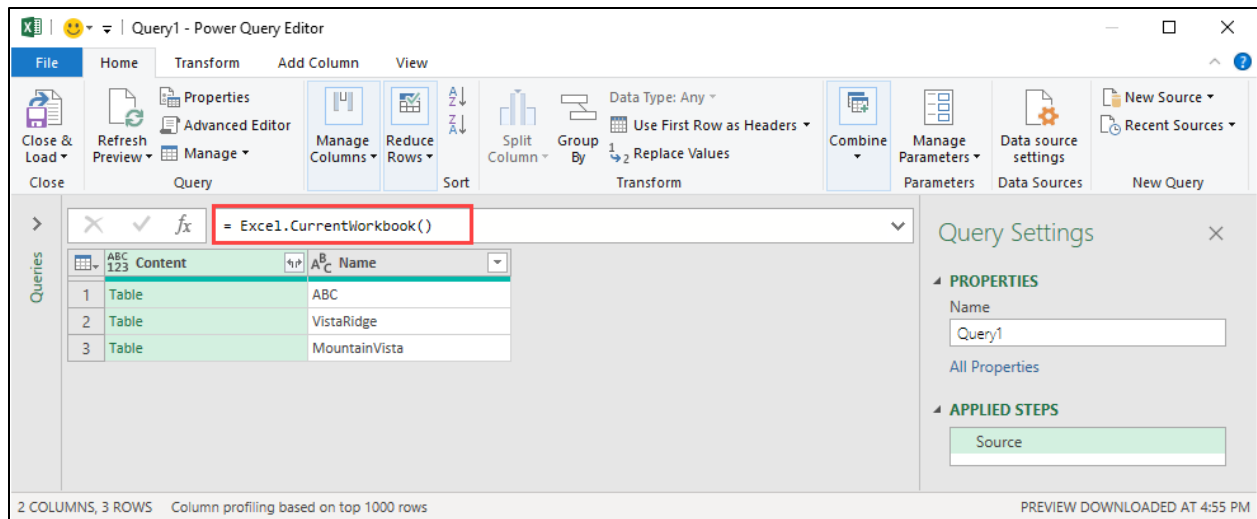


Figure 57 - Adding Tables to Power Query

Next, Terry clicks the double-headed arrow in the **Content** column's header to open a **Filter** dialog box. In the **Filter** dialog box, Terry unchecks the **Use original column name as prefix** box and clicks **OK**.

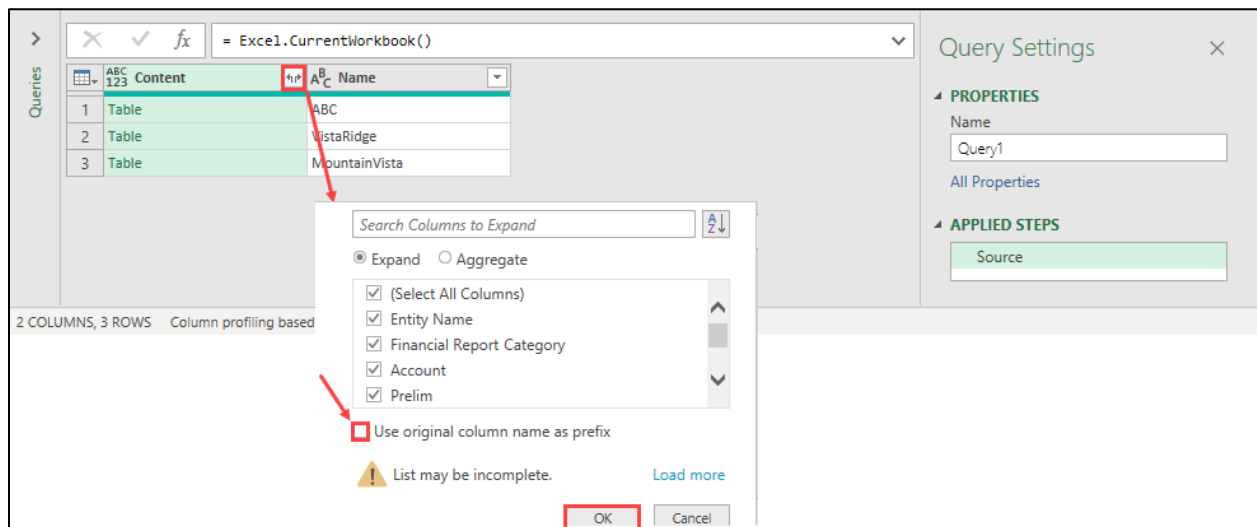


Figure 58 - Preparing to Combine All Tables in a Workbook into a Single Object

When Terry clicks **OK**, Power Query combines the three tables into a consolidated table, such as the one in **Figure 59**.

Managing Your Data Models with Power Pivot

Chapter

4

Power Pivot became available beginning with Excel's 2010 release. Since then, this tool's functionality has improved. Today, many professionals use Power Pivot to build more powerful PivotTables than could otherwise be built in its absence. With Power Pivot, you can easily overcome some of the limitations associated with large volumes of data, data residing in multiple data sources – including external data sources – and the need for more advanced calculations in PivotTables. In this chapter, you will learn how to use Power Pivot to enhance your PivotTables.

Learning Objectives

Upon completing this chapter, you should be able to:

- List the versions of Excel for which Power Pivot is available;
- Identify situations where enhancing a PivotTable using Power Pivot is advantageous;
- Construct PivotTables that incorporate features facilitated by Power Pivot; and
- Differentiate between data model management techniques available in Power Pivot.

Introducing Power Pivot

Microsoft introduced Power Pivot as an add-in for Excel with the 2010 application release. At that time, Power Pivot was available for all versions of Excel. Additionally, the 2016 release of Power Pivot made several notable changes that should make Power Pivot easier to use for most. For example, in Power Pivot for Excel 2016, the term “calculated field” changed to “measure” to distinguish this powerful feature from a different feature available in traditional PivotTables. Additionally, starting with Power Pivot 2016, you can right-click on a measure and immediately access an edit function. This functionality should reduce the time to resolve calculation issues with measures. Measures also appear in the PivotTable field list; additionally, you can search for fields in the PivotTable field list, making it easier to find items when working with data models containing many fields. Further, Microsoft added graphical improvements to help users understand and manage data relationships in data models.

Why Power Pivot?

Power Pivot provides several critical advantages over traditional PivotTables not built on data models. First, Power Pivot allows you to create and manage data models, overcoming many

issues associated with large data volumes. Using data models helps to keep calculation times acceptable, and data file sizes do not swell to unmanageable levels. These characteristics allow us to create and work with “industrial strength” PivotTables without unnecessary processing and calculation delays.

Second, Power Pivot looks very much like the traditional Excel user environment. Consequently, most users will find working in Power Pivot similar to working with an ordinary Excel spreadsheet. Of course, this means that team members can adopt Power Pivot with only modest amounts of training required.

Third, Power Pivot facilitates summarizing data from multiple data sources; in fact, Power Pivot even includes built-in tools for linking data into the data model from various data sources. Therefore, using a single technology, you can quickly connect to and summarize data – potentially large data volumes – from multiple sources.

Fourth, Power Pivot enables advanced calculations in PivotTables by provisioning items such as calculated columns, measures, and key performance indicators. With these tools, you can easily add the necessary calculations to your data to get the in-depth analyses desired for a given situation.

Power Pivot helps advanced PivotTable users become more efficient and effective when summarizing data for the abovementioned reasons and potentially many others. This fact remains true even if that data resides in multiple external sources.

Creating PivotTables from Power Pivot-based Data Models

To begin working with Power Pivot, click the **Manage** icon on the left-hand side of the **Power Pivot** tab of the Ribbon to open the **Power Pivot window** shown in **Figure 61**.

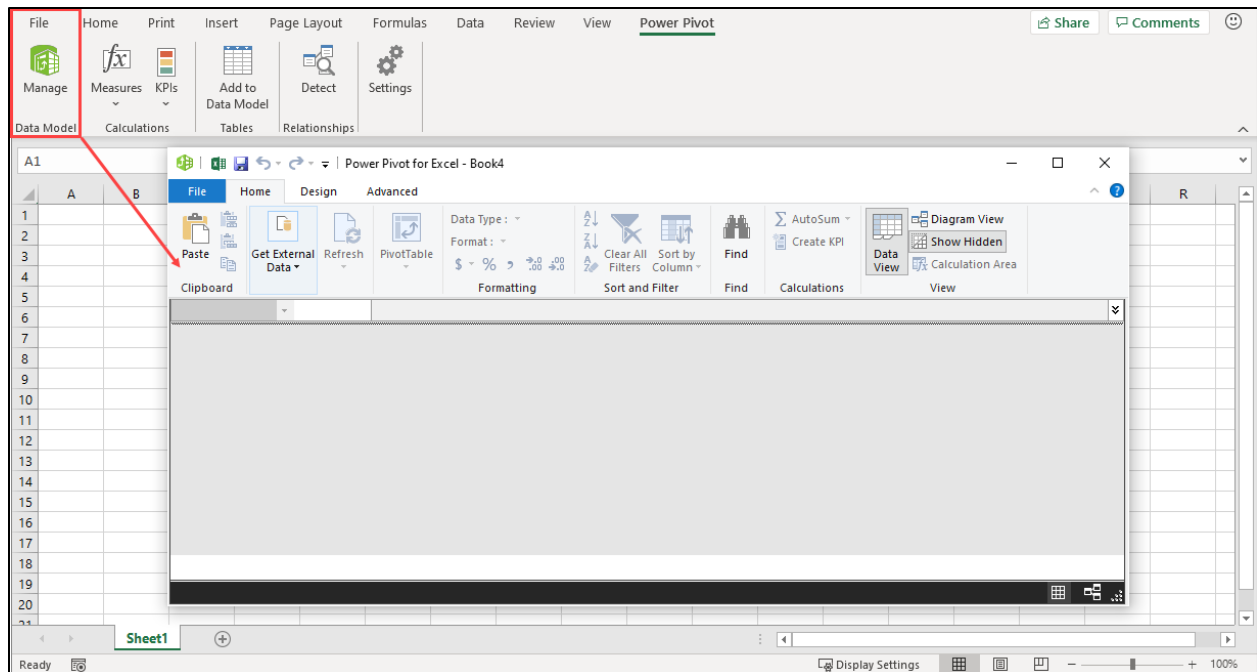


Figure 61 - The Power Pivot Window

Next, choose a data source from the **Get External Data** group on the **Power Pivot window's Home tab**. In our first example, we will click **From Database** and choose **From Access**, as shown in **Figure 62**. Power Pivot supports connections from the following databases: Microsoft SQL Azure, Microsoft Access, Oracle, Teradata, Sybase, Informix, IBM DB2, or any similar database. Additionally, Power Pivot can import data from an SQL Server Analysis Services cube file, Microsoft Reporting Services Report, Excel file, text file, or an Atom feed. Of course, you could also choose to utilize Get & Transform to link data from external data sources into a data model and then create your Power Pivot-based PivotTable from that data model.

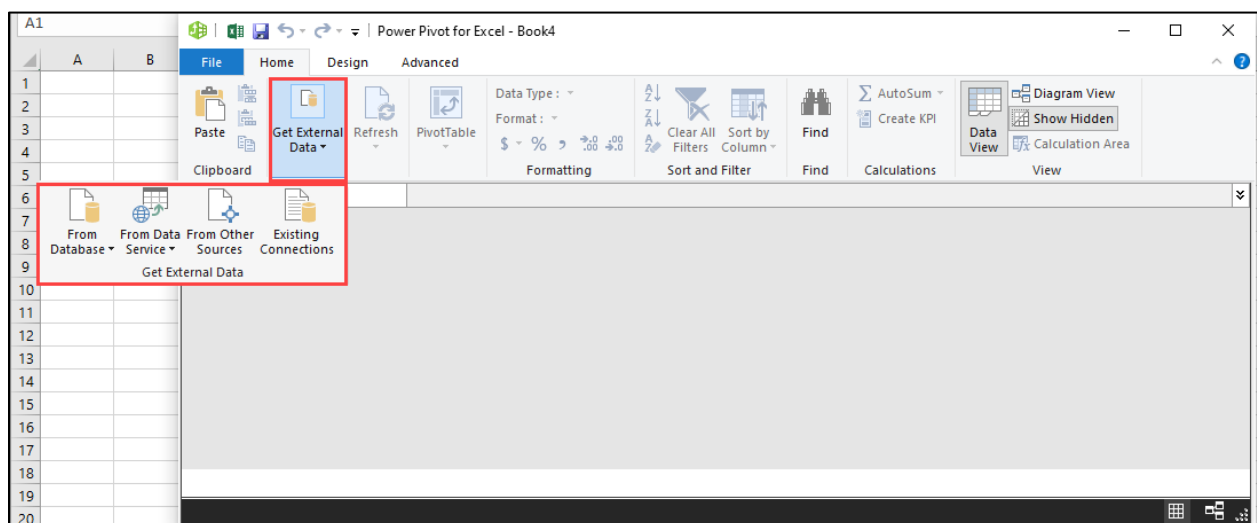


Figure 62 - Selecting a Data Source for Power Pivot

In the **Connect to Microsoft Access Database** window of the **Table Import Wizard** dialog box shown in **Figure 63**, select the database name and click **Next** twice to open the **Select Tables and Views** window shown in **Figure 64**. Next, select the table(s) to include and any desired filters. Upon completing those selections, click **Finish** to import the data into Power Pivot. Power Pivot displays the **Table Import Wizard** window pictured in **Figure 65** when it completes importing data.

The screenshot shows the 'Table Import Wizard' dialog box with the title 'Table Import Wizard'. The main heading is 'Connect to a Microsoft Access Database' with the instruction 'Enter the information required to connect to the Microsoft Access database.' Below this, there are several input fields and buttons. The 'Friendly connection name' field contains 'Access F60 through F69 PivotTable Sales Data'. The 'Database name' field contains 'er 4\F60 through F69 PivotTable Sales Data.accdb' and has a 'Browse...' button next to it. Below these fields is a section titled 'Log on to the database' which contains 'User name:' and 'Password:' labels with corresponding text boxes, and a checkbox labeled 'Save my password' which is currently unchecked. At the bottom right of this section are 'Advanced...' and 'Test Connection' buttons. At the very bottom of the dialog box are four buttons: '< Back', 'Next >' (which is highlighted with a blue border), 'Finish', and 'Cancel'.

Figure 63 – Connect to a Microsoft Access Database Window of the Power Pivot Table Import Wizard

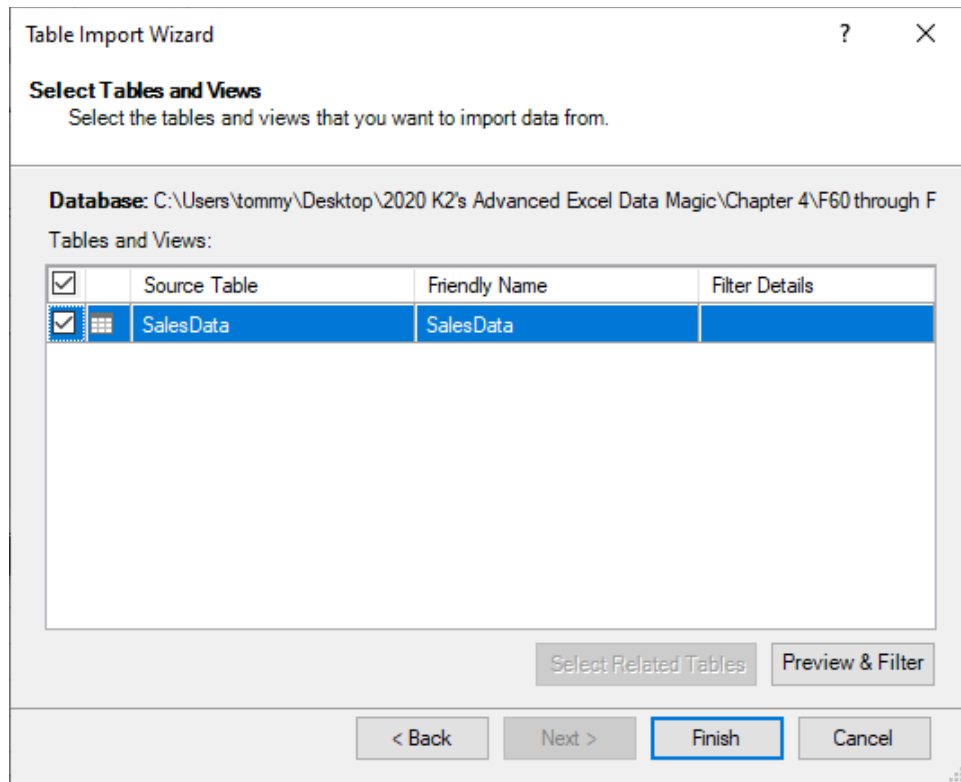


Figure 64 – Select Tables and Views Window of the Power Pivot Table Import Wizard

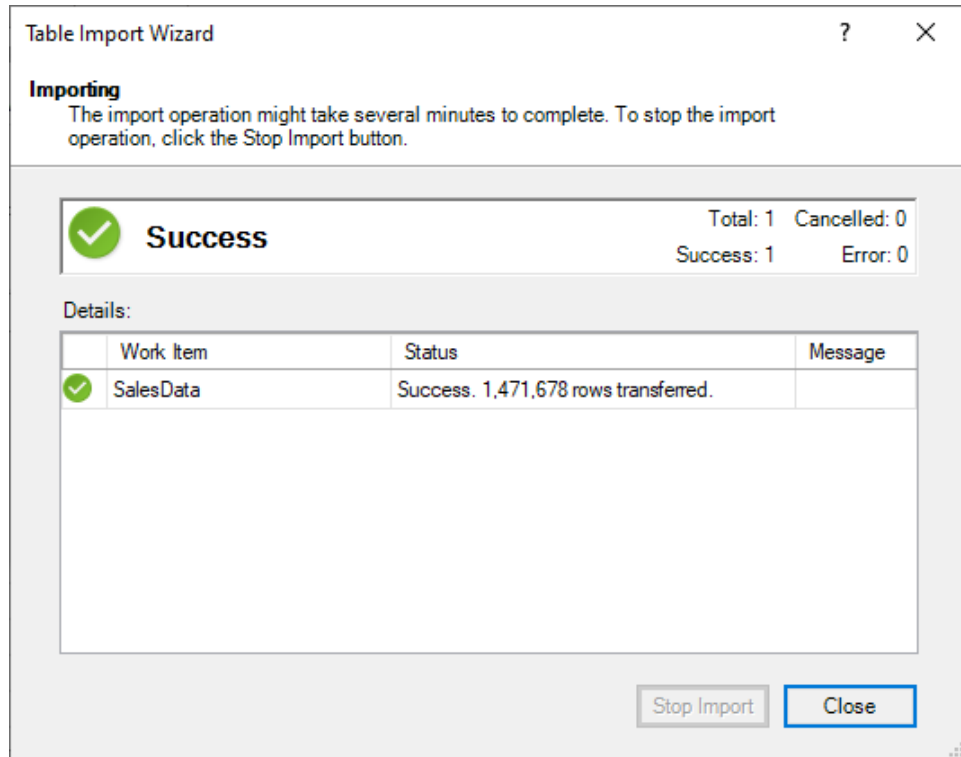


Figure 65 - Completed Import Message in Power Pivot

The imported data appears upon closing the Table Import Wizard, as shown in **Figure 66**. In the example shown, almost 1.5 million records exist in the imported data model.

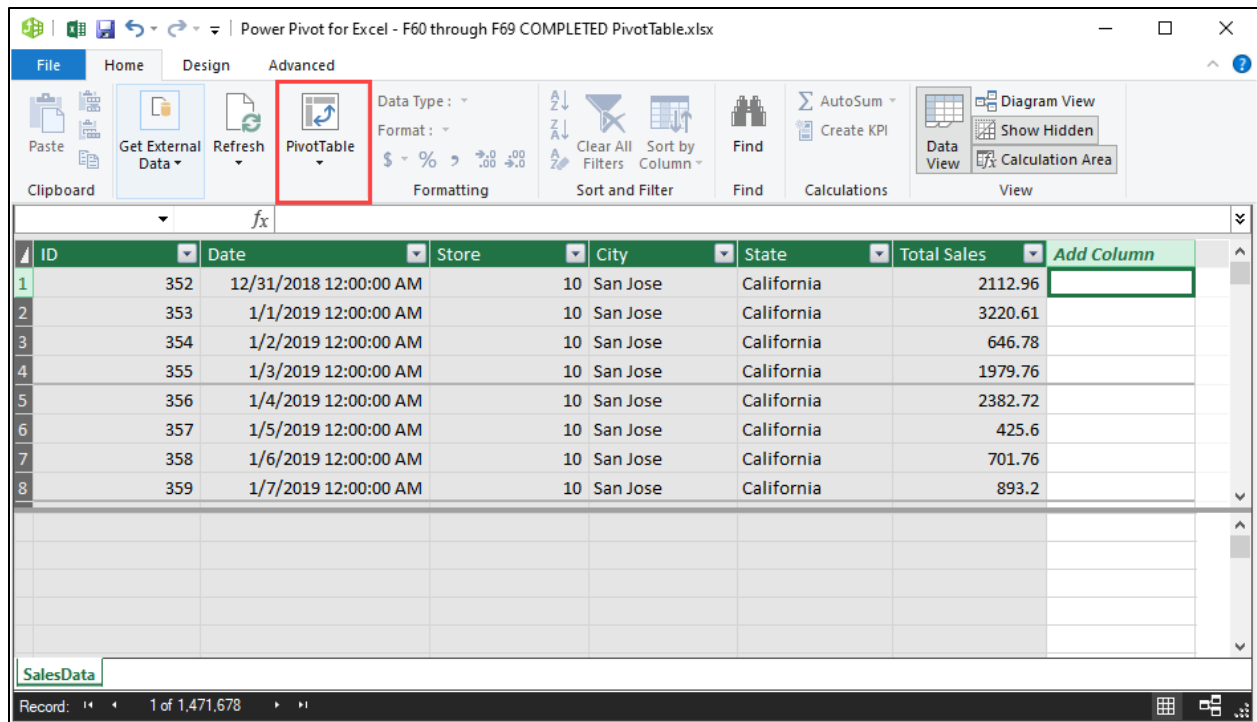


Figure 66 - Imported Data in Power Pivot

Building a PivotTable from the data model in Power Pivot begins by clicking the **PivotTable** icon located on the Ribbon, as shown in **Figure 66**. Then, if you click the arrow underneath the PivotTable icon, you can choose which type of PivotTable and PivotChart objects you will create, as shown in **Figure 67**.

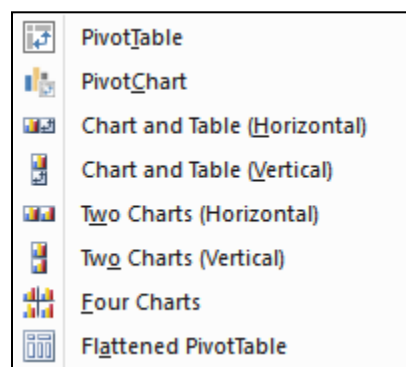


Figure 67 - Choosing the Number and Type of PivotTable Objects Created by Power Pivot

Next, choose whether Power Pivot will create the object in a **New Worksheet** or **Existing Worksheet**, as shown in **Figure 68**, and click **OK**.

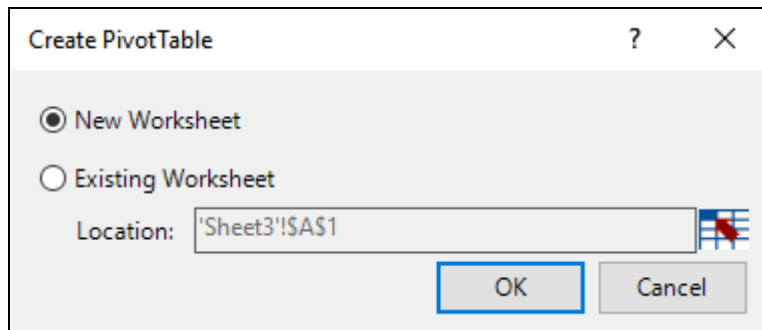


Figure 68 - Inserting Power Pivot Objects into a New Worksheet

Next, you will see the familiar PivotTable placeholder and task pane shown in **Figure 69**. From this point forward, the process for building a PivotTable is virtually identical to building “regular” PivotTables.

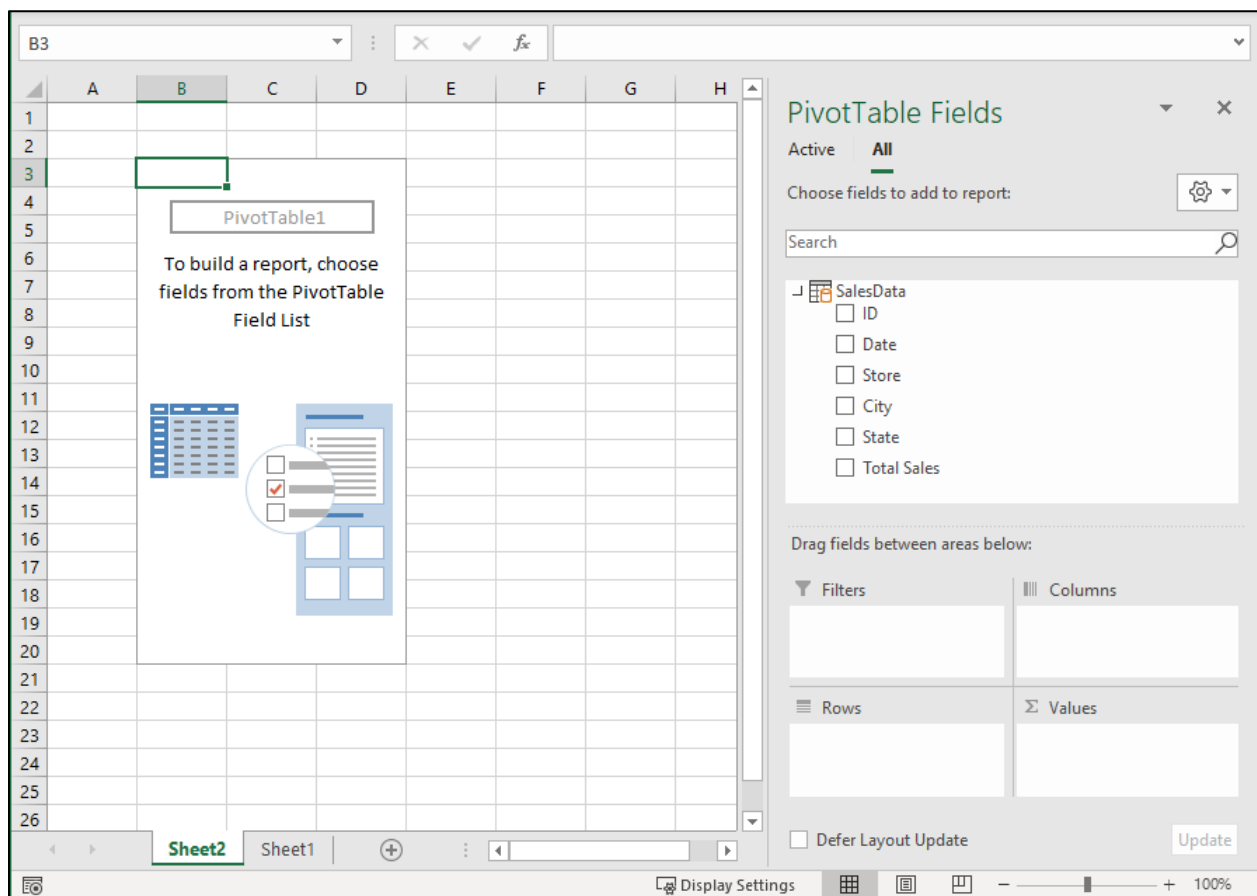


Figure 69 - Building a PivotTable Using Power Pivot

From the 1.5 million record data model shown in Figure 66, you can easily construct the PivotTable displayed in **Figure 70** in less than a minute, including State and Date **Slicers**.

State

Alabama

Alaska

Arizona

Arkansas

California

Colorado

Connecticut

District of Colum...

Florida

Georgia

Hawaii

Idaho

Illinois

Indiana

Iowa

Kansas

Kentucky

Louisiana

Maryland

Massachusetts

Michigan

Minnesota

Mississippi

Missouri

Montana

Nebraska

Nevada

New Hampshire

New Jersey

New Mexico

New York

North Carolina

Ohio

Oklahoma

Oregon

Pennsylvania

Rhode Island

South Carolina

South Dakota

Tennessee

Texas

Utah

Virginia

Washington

Wisconsin

Date (Year)

2018

2019

2020

Sum of Total Sales

Column Labels

2018

2019

2020

Grand Total

Row Labels

Alabama

Birmingham

Huntsville

Mobile

Montgomery

Colorado

Arvada

Aurora

Colorado Springs

Denver

Fort Collins

19,264.84

4,695,432.30

4,789,341.13

9,504,038.27

21,112.94

4,057,038.44

4,138,179.64

8,216,331.02

26,086.28

4,022,704.74

4,103,159.47

8,151,950.49

18,140.92

4,559,677.67

4,650,871.78

9,228,690.37

21,241.35

4,009,089.10

4,089,271.42

8,119,601.87

29,166.04

4,618,082.68

4,710,444.80

9,357,693.52

42,238.49

4,592,555.03

4,684,406.55

9,319,200.07

31,982.98

4,597,153.70

4,689,097.24

9,318,233.92

18,352.70

4,006,976.30

4,087,116.20

8,112,445.20

Figure 70 - PivotTable Created with Power Pivot

Using Power Pivot to Create a PivotTable from Multiple Data Sources

Another advantage of working with Power Pivot is that it allows users to create PivotTables with multiple data sources, such as a database and a spreadsheet. For example, assume that Global Brands LLC is a small distribution company that ships spices and condiments to customers worldwide. The company maintains a simple general ledger accounting application, and the CFO has exported data relating to 2020 from the general ledger to two different Excel workbooks. One of the workbooks contains the Chart of Accounts, and the other includes all accounting transactions. Additionally, the CFO has exported all the fiscal reporting periods to a text file. Now, the CFO would like to use Power Pivot to assist in preparing a report that brings data from these three data sources into a single summary. To do this, follow the steps outlined below.

Open the Power Pivot window, click **Manage**, click **Get External Data**, click **From Other Sources**, click **Excel File**, click **Next**, select **F70 through F73 Accounts Table.xlsx**, and click **Next**. In the following dialog box, **check the box next to Use first row as column headers**, click **Next**, and click **Finish**.

Repeat the abovementioned process, except substitute the **F70 through F73 Transactions Table.xlsx** workbook in this iteration. Additionally, in this iteration, you must specify that you want to query **Sheet1\$**. At this point, the tables in your data model should resemble those shown in **Figure 71**.

[Account Num... ▼		fx	
Account Number ▼	Account Name ▼	Add Column	
1	1010 Cash in Bank - Ch...		
2	1020 Cash in Bank - Pa...		
3	1110 Accounts Receiv...	[Date] ▼	fx
Date ▼	Account Number ▼	Amount ▼	Add Column
1	1/1/2020 12:00:00 AM	1010	-3000
2	1/1/2020 12:00:00 AM	1010	4989.34
3	1/1/2020 12:00:00 AM	1010	-37043
4	1/2/2020 12:00:00 AM	1010	-87.21
5	1/2/2020 12:00:00 AM	1010	-827.34
6	1/2/2020 12:00:00 AM	1010	-7334.53
7	1/2/2020 12:00:00 AM	1010	14858.48
8	1/3/2020 12:00:00 AM	1010	-50.27

Figure 71 – Selecting Tables from Two Excel Workbooks Using Power Pivot

Now, adding the third and final table to the query is necessary. From the Power Pivot Home tab, click **Get External Data, From Other Sources**, choose the **Text File** option, and click **Next**. Browse to the file's location and select the **F70 through F73 Accounting Periods.txt** file. Ensure you check the box labeled **Use first row as column headers** box and that the **Column Separator** is designated **Tab**. Click **Finish** to add this table to the query.

With all the data queried, we can use Power Pivot to manage the data's relationships. To do so, click **Diagram View** on Power Pivot's **Home** tab. This action provides us with a visual interface to create and manage relationships. For example, with Diagram View open, you can use a drag-and-drop methodology to relate **Sheet 1** to **Sheet 1 1**. Use your mouse to drag the **Account Number** field from one of the tables and drop it onto the **Account Number** field on the other table. Upon doing so, Power Pivot will create a **one-to-many** relationship from the **Sheet 1** table to the **Sheet 1 1** table. Then, use your mouse to drag the **Transaction Date** field from the **F70 through F73 Accounting Periods** table to the **Date** field on the **Sheet 1 1** table.

Upon completing these tasks, the Diagram View of the data model should resemble that shown in **Figure 72**.

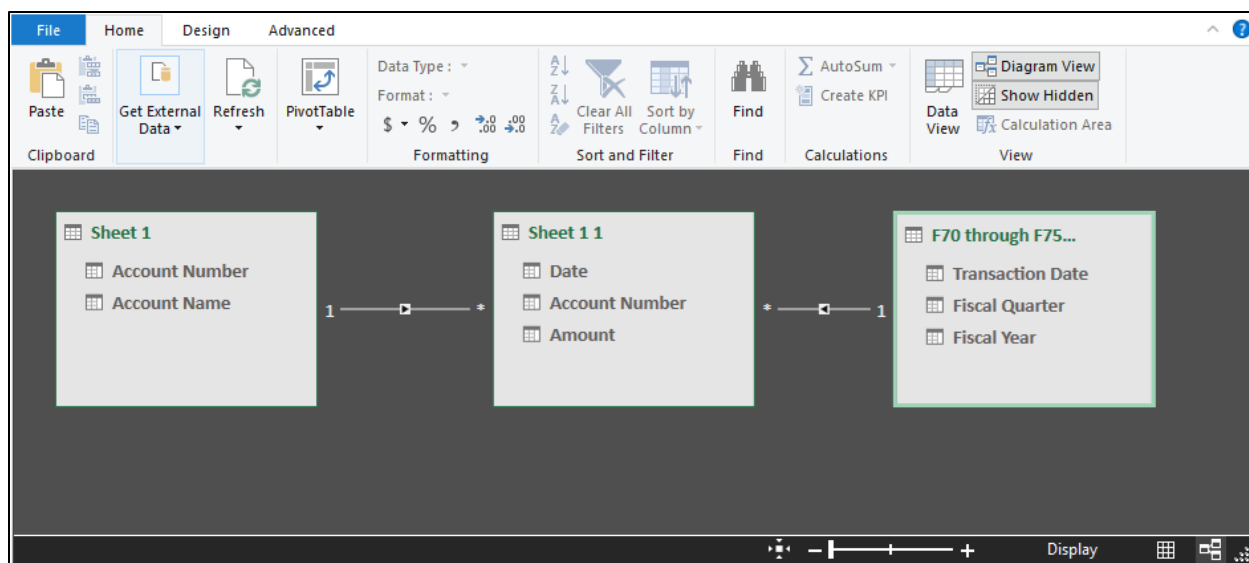


Figure 72 – Diagram View of a Data Model in Power Pivot

With the data model related, you are ready to create your report. To do so, click **PivotTable** on the **Power Pivot Ribbon's Home tab** to return to Excel, where you will construct your PivotTable from the data model. Next, select the option to add your PivotTable to a **New Worksheet**, and Excel will provide the familiar PivotTable interface shown in **Figure 73**.

The PivotTable Fields task pane shows the following configuration:

- Active:** All
- Choose fields to add to report:**
 - ☐ Transaction Date
 - ☐ Fiscal Quarter
 - ☒ Fiscal Year
- Sheet 1:**
 - ☐ Account Number
 - ☒ Account Name
- Sheet 1.1:**
 - ☐ Date
 - ☐ Account Number
 - ☒ Amount
- Drag fields between areas below:**
 - Filters:** (Empty)
 - Columns:** Fiscal Year
 - Rows:** Account Name
 - Values:** Sum of Amount
- Defer Layout Update:** ☐ (unchecked)

The resulting PivotTable in the spreadsheet is as follows:

	Sum of Amount	Column Labels		
Row Labels		2019	2020	Grand Total
Accounting Fees		64,944.81	67,047.13	131,991.94
Amortization Expense		76,342.23	69,164.80	145,507.03
Bank Fees		65,596.41	79,496.82	145,093.23
Cash in Bank - Checking		133,217.26	321,001.66	454,218.92
Cash in Bank - Payroll		-	(3,000.00)	(3,000.00)
Employee Benefits		57,617.20	82,506.71	140,123.91
Equipment Rental		75,951.79	61,039.25	136,991.04
Insurance Expense		71,095.62	75,117.35	146,212.97
Meals		53,596.07	73,301.93	126,898.00
Miscellaneous Expense		76,846.47	61,983.63	138,830.10
Office Supplies and Expense		64,640.25	72,858.92	137,499.17
Other Expenses		59,729.43	72,040.61	131,770.04
Payroll Tax Expense		97,136.55	96,386.64	193,523.19
Property and Other Taxes		77,240.31	66,686.30	143,926.61
Rent		18,000.00	18,000.00	36,000.00
Salaries - Officer/Shareholder		90,000.00	90,000.00	180,000.00
Salaries and Wages		114,000.00	114,000.00	228,000.00
Sales Revenue		(1,326,996.14)	(1,551,334.88)	(2,878,331.02)
Travel		59,325.82	73,364.41	132,690.23
Utilities		71,715.03	60,228.73	131,943.76

Figure 73 – Simple PivotTable Created from Three-table Data Model

Adding a few filters and some formatting converts the “raw” report above to one with a more polished and professional one, as shown in **Figure 74**.

Summary Cash Basis Income Statement		
	2019	2020
Sales Revenue	1,326,996.14	1,551,334.88
Accounting Fees	64,944.81	67,047.13
Amortization Expense	76,342.23	69,164.80
Bank Fees	65,596.41	79,496.82
Employee Benefits	57,617.20	82,506.71
Equipment Rental	75,951.79	61,039.25
Insurance Expense	71,095.62	75,117.35
Meals	53,596.07	73,301.93
Miscellaneous Expense	76,846.47	61,983.63
Office Supplies and Expense	64,640.25	72,858.92
Other Expenses	59,729.43	72,040.61
Payroll Tax Expense	97,136.55	96,386.64
Property and Other Taxes	77,240.31	66,686.30
Rent	18,000.00	18,000.00
Salaries - Officer/Shareholder	90,000.00	90,000.00
Salaries and Wages	114,000.00	114,000.00
Travel	59,325.82	73,364.41
Utilities	71,715.92	60,338.72
Net Income	133,217.26	318,001.66

Figure 74 – Simple Income Statement Constructed from a Three-table Data Model

The underlying data model serves as the source data for the PivotTable pictured above. As transactions append to the three data sources feeding data to the data model, you can refresh the data model, and the PivotTable report updates immediately. Clearly, this represents the next generation of Excel reporting.



Compare and contrast “traditional” PivotTables with those created using Power Pivot. How does Power Pivot potentially alter your strategy for deploying advanced PivotTables in your organization?

Managing Data and Relationships in a Data Model

You have learned how to create and manage relatively simple relationships in a data model. However, you may sometimes need more comprehensive tools to manage these relationships. Fortunately, these tools are available and relatively easy to use.

To access tools for managing relationships in your data model, click **Manage Data Model** on the **Data** tab of the Ribbon or **Manage** on the Ribbon's **Power Pivot** tab to open the data model in Power Pivot. Then, click **Diagram View** on the Ribbon's **Home** tab to open the visual perspective of the data model pictured in **Figure 75**.

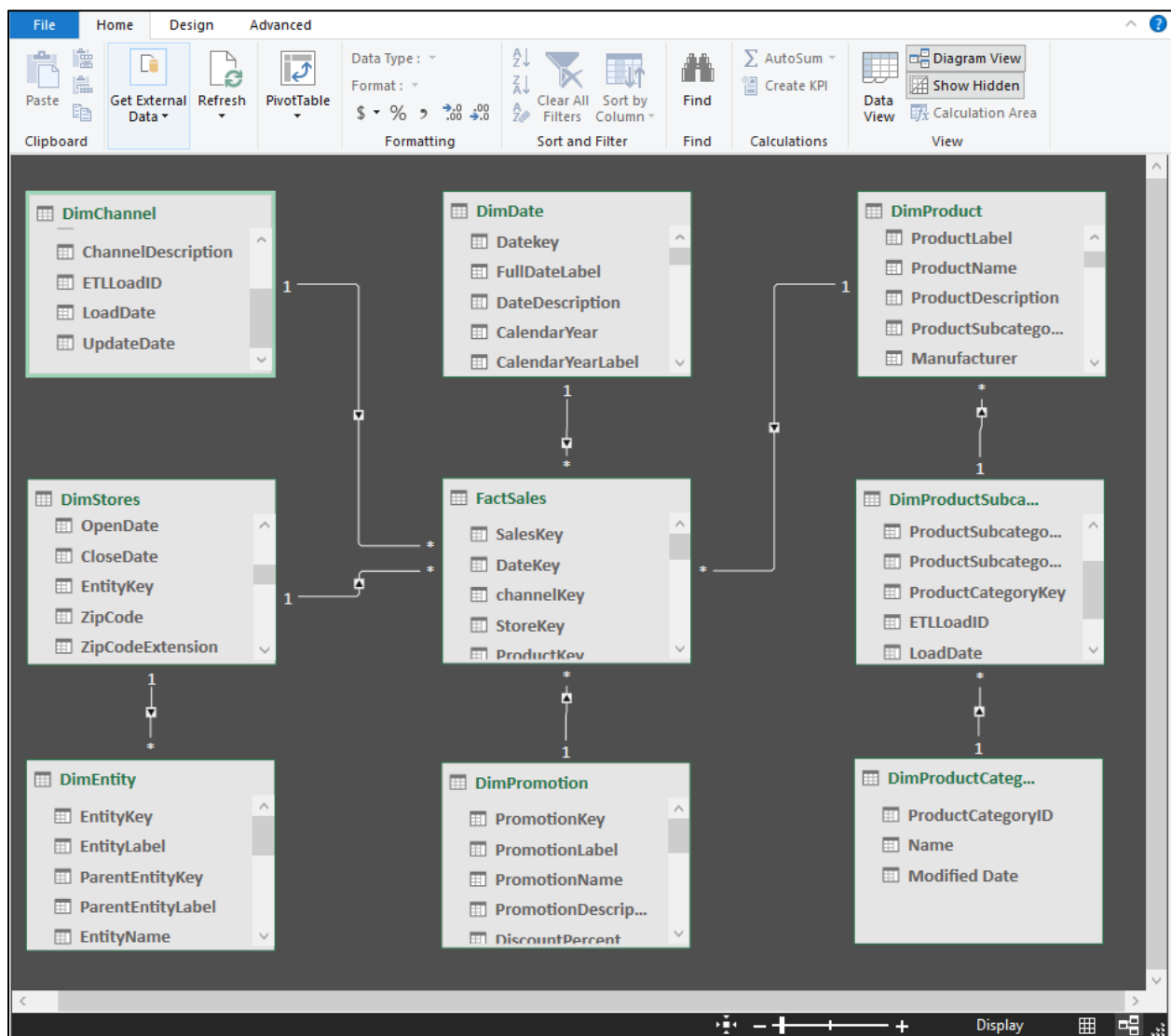


Figure 75 - Diagram View of a Data Model

Many users will find Diagram View helpful when working with data models because it depicts the data model's relationships. Regarding relationships, you will notice that many databases contain

one or more *Fact* tables and multiple *Dim* tables. *Fact* tables store the data you will summarize, including transactional data. On the other hand, *Dim* tables contain the various dimensions by which you may choose to summarize the data in a Fact table, with *product category*, *region*, and *salesperson* as examples. Because of this, you will often see “one-to-many” relationships when examining your data models with Dim tables on the “one” side of the relationship and Fact tables on the “many” side of the relationship – many sales transactions attributable to one salesperson, for example.

In Diagram View, you can edit the relationships already in place. For example, you can right-click on a relationship and mark it inactive. You can also edit or delete it, as shown in **Figure 76**.

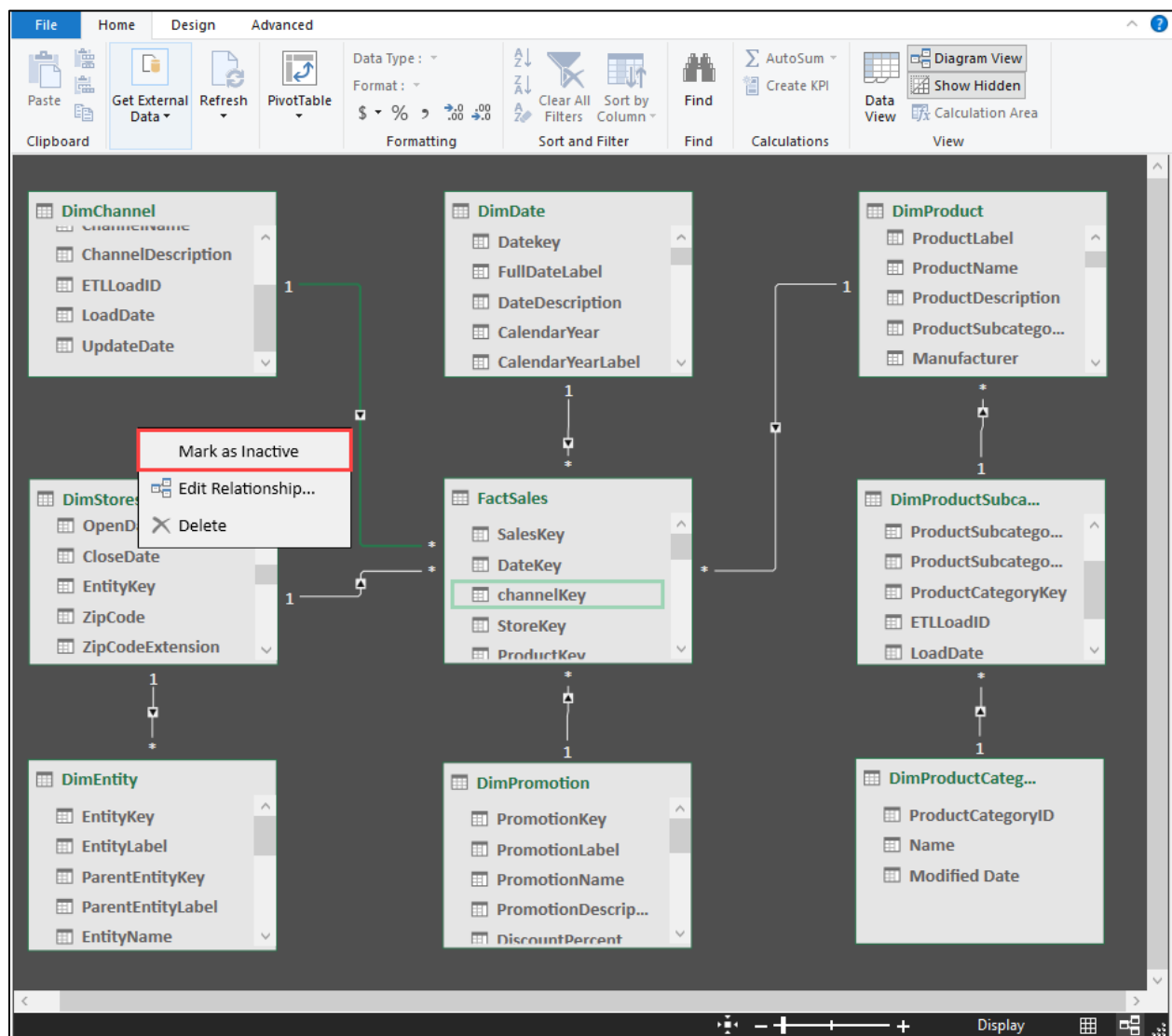


Figure 76 - Managing Relationships from Diagram View

Also, you can right-click on a field in an unrelated table and choose the **Create Relationship** option. This action opens the **Create Relationship** dialog box pictured in **Figure 77**. In this dialog

box, you can establish a relationship between two tables. To do so, select the two tables you wish to relate. Then, select the two fields on which to base the relationship. Next, click **OK** to complete the process of linking the tables. Upon doing so, Excel creates the connection. It also determines whether the relationship is a one-to-one or one-to-many relationship. Finally, click **Data View** on the Ribbon's **Home** tab to close Diagram View.

Create Relationship

Select tables and columns that relate to one another.

DimChannel

ChannelDescription	ChannelKey	ChannelLabel	ChannelName	ETLLoadID	LoadDate	UpdateDate
Store	1	01	Store	1	7/11/2015 2:43:19 PM	7/11/2015 2:43:19 PM
Online	2	02	Online	1	7/11/2015 2:43:19 PM	7/11/2015 2:43:19 PM
Catalog	3	03	Catalog	1	7/11/2015 2:43:19 PM	7/11/2015 2:43:19 PM
Reseller	4	04	Reseller	1	7/11/2015 2:43:19 PM	7/11/2015 2:43:19 PM

FactSales

channelKey	CurrencyKey	DateKey	DiscountAmount	DiscountQuantity	ETLLoadID	LoadDate	ProductKey	PromotionKey
1	1	5/7/2016 12:00:00 AM	0	0	1	1/1/2010 12:00:00 AM	375	1
1	1	5/22/2018 12:00:00 AM	0	0	1	1/1/2010 12:00:00 AM	375	1
1	1	9/30/2018 12:00:00 AM	0	0	1	1/1/2010 12:00:00 AM	375	1
1	1	10/31/2017 12:00:00 AM	0	0	1	1/1/2010 12:00:00 AM	375	1
1	1	6/26/2018 12:00:00 AM	0	0	1	1/1/2010 12:00:00 AM	375	1

☒ Active

OK Cancel

Figure 77 - Creating Relationships through the Create Relationship Dialog Box

While working in Data View, you can access tools to manage relationships. To do so, click **Create Relationship** or **Manage Relationships** on the **Design** tab of the Ribbon. For example, if you select *Create Relationship*, Excel presents the same dialog box illustrated in Figure 77. However, the dialog box is slightly different if you click *Manage Relationships*, as shown in **Figure 78**.

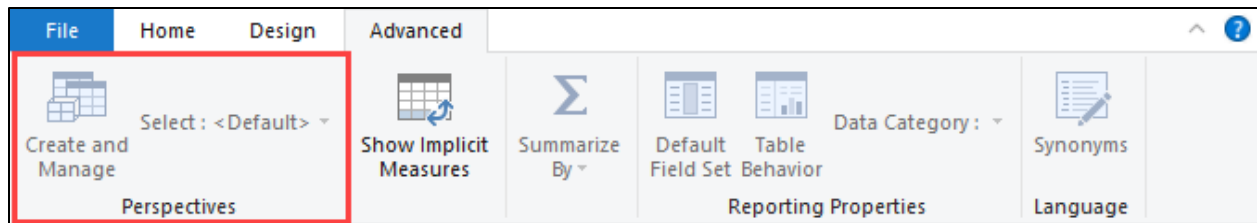


Figure 79 - Choosing to Create a New Perspective for a Data Model

In the resulting Perspectives dialog box, click **New Perspective**, enter a name for the perspective you are creating, and indicate which tables and fields within selected tables will be accessible when using this perspective. For example, Figure 80 depicts a new perspective entitled *Regional Sales* that excludes access to the **DimChannel** table. Click **OK** to add a perspective to your data model.

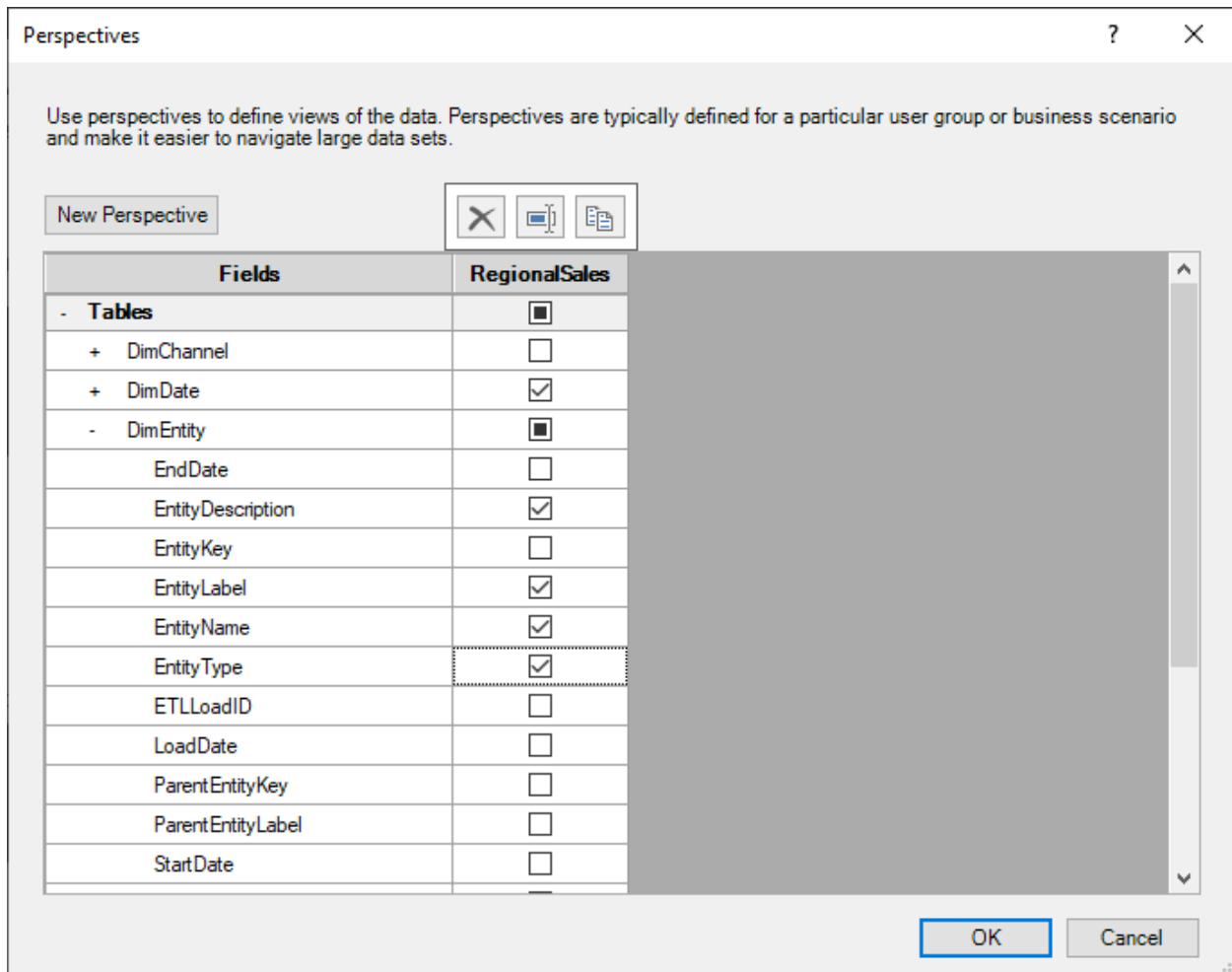


Figure 80 - Adding a Perspective to a Data Model

Repeat the process outlined above for as many perspectives as you need. For example, **Figure 81** illustrates a scenario in which four perspectives are available in the data model, each with access to different tables and fields within those tables.

Use perspectives to define views of the data. Perspectives are typically defined for a particular user group or business scenario and make it easier to navigate large data sets.

New Perspective

Fields	RegionalSales	Product Manag...	Accounting	Business Plann...
- Tables	☒	☒	☒	☒
+ DimChannel	☐	☐	☒	☒
+ DimDate	☒	☒	☒	☒
+ DimEntity	☒	☐	☒	☒
+ DimProduct	☐	☒	☒	☐
+ DimProductCategory	☐	☒	☒	☒
+ DimProductSubcategory	☐	☒	☒	☒
+ DimPromotion	☐	☐	☒	☒
+ DimStores	☒	☐	☒	☐
+ FactSales	☒	☒	☒	☒

OK Cancel

Figure 81 - Multiple Perspectives Available for a Data Model

You can then indicate which perspective you want to use when working with the data model with the perspectives defined. In the **Perspectives** group on the Ribbon, click the drop-down arrow next to **Select** and choose the best perspective for a particular task, as shown in **Figure 82**. Note that as you select different perspectives, the tables and fields that are accessible will differ based on the definition of that perspective.

?

From your vantage point, what are some advantages of using Perspectives in your data models?

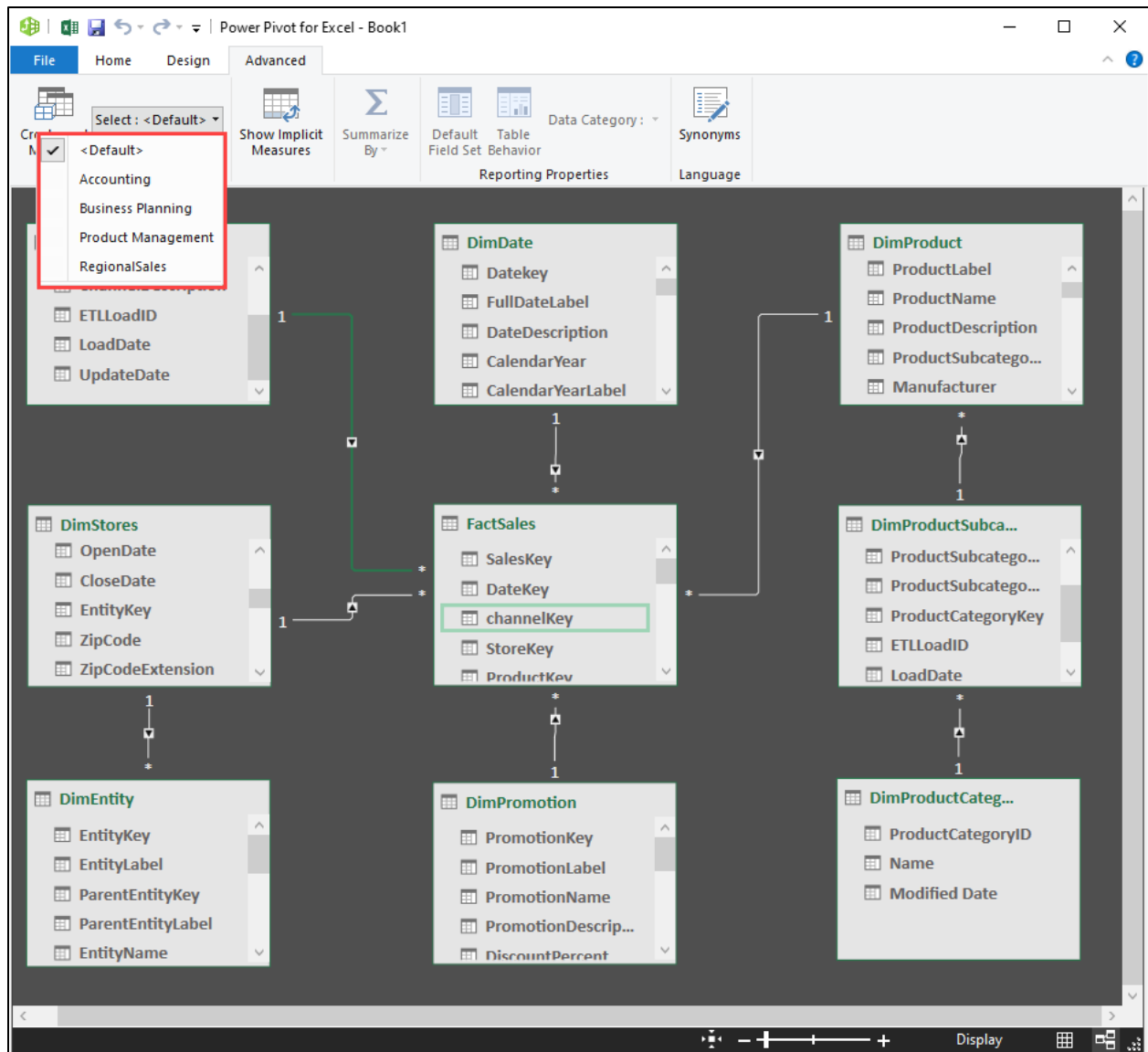


Figure 82 - Choosing Among Available Perspectives When Working with Data Models

Building DAX Functions And Cube Formulas

Chapter

5

Despite the tremendous summarization techniques available in PivotTables created from data models, two issues could potentially cause these tools to fall short of lofty expectations. The first is creating user-defined calculations in data models that feed data into our PivotTables; this is a critical consideration when you recall that you generally cannot insert user-defined calculations into PivotTables. The second is arranging the PivotTables we construct from data models to be more “reader-friendly.” We begin with the discussion of user-defined calculations.

Learning Objectives

Upon completing this chapter, you should be able to:

- Define how Data Analysis Expressions functions enhance data models;
- List examples of common Data Analysis Expressions functions;
- Identify the primary purpose of Key Performance Indicators;

Building Formulas with Data Analysis Expressions

Although the reporting environment we are striving to move toward generally seeks to minimize or eliminate the use of formulas, that is not always possible. In some cases, we might need to access more powerful functions than what otherwise are present in Excel. Fortunately, **Data Analysis Expressions** (DAX) functions exist to address this issue. The formulas you create using DAX functions can address issues such as date and time calculations, filtering your data, transforming textual data, and performing statistical calculations. Although a complete discussion of DAX functions is beyond the scope of this session, perhaps what is most important to remember about them is that they serve as the vehicle for adding user-defined calculations into your data models. You can examine the full complement of DAX functions at <https://k2e.fyi/daxfunctions>.

As defined by Microsoft, **Data Analysis Expressions** (DAX) “... is a formula language that allows users to define custom calculations in Power Pivot...DAX includes some of the functions used in Excel formulas and additional functions that work with relational data and perform dynamic aggregation.” From a practical standpoint, DAX can insert calculations into your data models.

The DAX environment currently includes approximately 300 functions, some of which act virtually the same as counterparts in a traditional Excel environment. You can use these functions to add

calculated columns to your data models. Additionally, DAX includes other functions to insert calculations directly into your Power Pivot-based PivotTables.

While many of the functions used for creating calculated columns work the same in Power Pivot as in traditional Excel applications, some differences exist, including the following.

- DAX renames Excel's **TEXT** function to **FORMAT**, **SUMIFS** to **CALCULATE**, and **CHOOSE** to **SWITCH**.
- DAX adds a **BLANK** function often used with IF to exclude blank rows from calculations.
- DAX uses **RELATED** instead of **VLOOKUP** to perform queries on data in a PivotTable.

According to Microsoft, some general guidelines for using DAX functions include the following.

- Many DAX functions have the same name and general behavior as Excel functions but have been modified to take different types of inputs and, in some cases, might return a different data type. Generally, you cannot use DAX formulas in an Excel workbook or use Excel formulas in a Power Pivot workbook without some modification.
- DAX functions never take a cell or a range of cells as a reference; instead, DAX functions take a column or table as a reference.
- DAX date and time functions return a *datetime* data type. In contrast, Excel date and time functions return an integer representing a date as a serial number.
- Many new DAX functions either return a table of values or make calculations based on a table of values as input. In contrast, Excel has no functions that return a table, but some functions can work with arrays. In addition, the ability to reference complete tables and columns is a relatively new feature in Power Pivot.
- DAX provides new look-up functions similar to Excel's array and vector lookup functions. However, DAX functions require a relationship between the tables involved in the lookup.
- DAX does not support the variant data type found in Excel. DAX always expects the data in a column to be of the same data type. If the data is not the same type, DAX changes the entire column to the data type that best accommodates all values.

Some DAX functions work virtually identically to their “normal” Excel counterparts. For instance, the DAX functions **AVERAGE**, **AVERAGEA**, **COUNT**, **COUNTA**, **MAX**, **MIN**, **PRODUCT**, and **SUM** are similar in syntax, structure, and performance compared to functions sharing the same name in Excel. To illustrate, compare the following usages of SUM. The first example is the traditional SUM function, and the second is a DAX version.

=SUM(A1:A57)

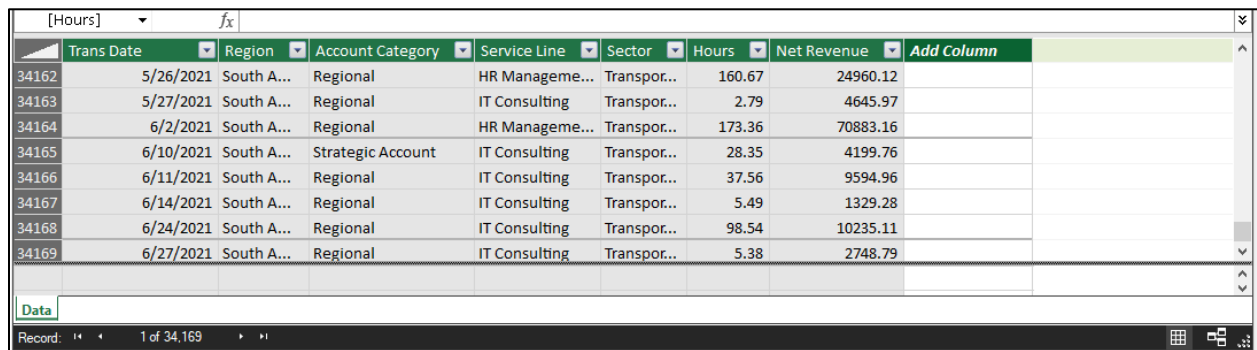
=SUM(Sales[Amt])

In the first example above, the SUM function adds cells A1 through A57. However, note the absence of any cell references in the second example. Instead, the SUM function in the second example – a DAX formula – instructs Excel to sum the *Amount* column from the *Sales* table in a data model. This example shows some differences, although many functions used for creating measures and calculated columns work the same in Power Pivot as in traditional Excel applications.

Creating Simple Calculated Columns with DAX

Calculated columns are columns that you add to an existing data model. Then, instead of pasting or importing values in the column, you create a DAX formula that defines the column's values. Once you create a calculated column, you can use the column's values like data from any other column.

To illustrate the power of calculated columns built using DAX expressions, consider the data extraction shown in **Figure 83**; the extraction shows ten records with a population of 34,169 records of time and billing data for a consulting firm. The firm operates on a fiscal year starting on July 1 and ending on June 30. PivotTables – including those built from data models – have the inherent capability to automatically group date data on calendar years/quarters/months; fiscal year groupings can be created but involve manual workarounds.



	Trans Date	Region	Account Category	Service Line	Sector	Hours	Net Revenue	Add Column
34162	5/26/2021	South A...	Regional	HR Manageme...	Transpor...	160.67	24960.12	
34163	5/27/2021	South A...	Regional	IT Consulting	Transpor...	2.79	4645.97	
34164	6/2/2021	South A...	Regional	HR Manageme...	Transpor...	173.36	70883.16	
34165	6/10/2021	South A...	Strategic Account	IT Consulting	Transpor...	28.35	4199.76	
34166	6/11/2021	South A...	Regional	IT Consulting	Transpor...	37.56	9594.96	
34167	6/14/2021	South A...	Regional	IT Consulting	Transpor...	5.49	1329.28	
34168	6/24/2021	South A...	Regional	IT Consulting	Transpor...	98.54	10235.11	
34169	6/27/2021	South A...	Regional	IT Consulting	Transpor...	5.38	2748.79	

Figure 83 - Extraction of Time and Billing Data to Use in Creating Calculated Columns with DAX

In this example, a relatively easy-to-implement workaround will be to use three DAX functions – **MONTH**, **YEAR**, and **IF** – to create calculated columns in the data model to identify the

appropriate fiscal quarter and fiscal year for the time and billing records. Note that these three functions operate as DAX functions, as they would in a traditional Excel environment.⁹

Begin by double-clicking at the top of a blank column in the data model and entering *Month* as the column header. Repeat that process two times, adding *Fiscal Quarter* and *Fiscal Year*. Then, enter the following formula on the first row of the Month column.

=Month([Trans Date])

This formula identifies the month number (1 through 12) based on the data in the current row of the *Trans Date* column in the data model. Notice that it uses a **MONTH** DAX function that operates identically to the MONTH function in traditional Excel spreadsheets.

Next, enter the following formula in the first row of the *Fiscal Quarter* column.

=IF([Month]<4,3,IF([Month]<7,4,IF([Month]<10,1,2)))

This formula checks to see if the month is January, February, or March (month numbers 1 through 3), and if so, it assigns the value of 3 to that row of the *Quarters* column, indicating that the month falls into the third fiscal quarter. If the month is April, May, or June (month numbers 4 through 6), the formula assigns 4 to that row of the *Quarters* column, indicating that the month falls into the fourth fiscal quarter. If the month is July, August, or September (month numbers 7 through 9), the formula assigns a value of 1 to that row of the *Quarters* column, indicating that the month falls into the first fiscal quarter. Finally, if the month is October, November, or December (month numbers 10 through 12), the formula assigns a value of 2 to that row of the *Quarters* column. Of course, this assignment indicates that the month falls into the second fiscal quarter.

Finally, enter the following formula into the *Fiscal Year* column.

=If([Month]<7,Year([Trans Date]),Year([Trans Date])+1)

This formula identifies the appropriate fiscal year for the transaction. It does so by checking to see if the month is January through June and, if it is, assigns the transaction to the year found in the *Trans Date* field. Otherwise, if the month is July through December, Excel deems the transaction to occur in the following fiscal year. **Figure 84** displays the modified data model, with three newly added calculated columns based on **MONTH**, **YEAR**, and **IF**, three DAX functions.

⁹ Those well versed in Power Pivot may already realize that Date Tables (also known as Calendar Tables) represent a viable option to the use of DAX-based formulas to identify fiscal reporting periods in data models.

=if(Data[Month]<7,year(Data[Trans Date]),year(Data[Trans Date])+1)										
=if(Data[Month]<4,3,if(Data[Month]<7,4,if(Data[Month]<10,1,2)))										
fx =month(Data[Trans Date])										
Trans Date	Region	Account Category	Service Line	Sector	Hours	Net Revenue	Month	Fiscal Quarter	Fiscal Year	
7/1/2019	Asia	Global Account	IT Consulting	CleanTech	13.4	2228.7	7	1	2020	
7/1/2019	Asia	Regional	HR Manageme...	Defense...	103.87	7643.16	7	1	2020	
7/1/2019	Asia	Global Account	HR Manageme...	Defense...	3.64	1017.11	7	1	2020	
7/1/2019	Asia	Strategic Account	Training	Defense...	15.6	4100.06	7	1	2020	
7/1/2019	Asia	Global Account	IT Consulting	Defense...	4.44	973.24	7	1	2020	
7/1/2019	Asia	Regional	IT Consulting	Life Scie...	10.61	2329.37	7	1	2020	
7/1/2019	Asia	Strategic Account	HR Manageme...	Life Scie...	16.68	2710.77	7	1	2020	
7/1/2019	Asia	Regional	IT Consulting	Life Scie...	23.69	4426.17	7	1	2020	
7/1/2019	Asia	Global Account	HR Manageme...	Oil & Gas	380.77	74851.97	7	1	2020	

Figure 84 – Creating Calculated Columns in the Data Model with DAX Expressions

With the calculated columns in place, all that is left to do is to complete the PivotTable. From Power Pivot's **Home** tab, click **PivotTable**. In the PivotTable field list, drag *Region* and *Service Line* to the *Rows* quadrant, *Fiscal Year* and *Fiscal Quarter* to the *Columns* quadrant, and *Net Revenue* to the *Values* quadrant. With formatting added, the completed report might resemble that pictured in **Figure 85**.

	B	C	D	E	F	G	H
3	Summary of Revenue by Region by Service Line						
4		2020				2021	Grand Total
5		Q1	Q2	Q3	Q4		
6	Asia						
7	Advisory Services	1,658,032.94	1,576,453.77	1,301,879.93	1,892,261.05	12,360,427.55	18,789,055.24
8	HR Management	4,520,169.49	5,137,624.93	3,906,747.95	3,284,249.64	25,869,617.77	42,718,409.78
9	IT Consulting	4,121,422.18	4,625,545.13	4,924,517.10	5,449,134.85	25,900,438.80	45,021,058.06
10	Training	675,998.37	727,372.56	653,771.06	411,210.91	4,524,464.40	6,992,817.30
11	Europe						
12	Advisory Services					32,556,283.45	32,556,283.45
13	HR Management					83,009,493.19	83,009,493.19
14	IT Consulting					62,567,734.62	62,567,734.62
15	Training					11,730,023.94	11,730,023.94
16	North America						
17	Advisory Services	12,306,107.21	11,758,282.30	9,985,846.06	14,550,079.51	93,790,984.33	142,391,299.41
18	HR Management	43,044,885.85	44,382,045.00	37,769,699.45	31,831,110.70	238,928,484.35	395,956,225.35
19	IT Consulting	30,640,819.14	34,164,260.01	36,674,269.90	42,187,684.74	198,726,768.70	342,393,802.49
20	Training	5,327,027.03	4,860,874.83	4,669,726.12	2,949,590.75	34,129,956.82	51,937,175.55
21	South America						
22	Advisory Services	6,529,300.05	5,802,681.90	3,151,025.60	7,354,537.67	31,050,687.45	53,888,232.67
23	HR Management	24,838,193.69	24,784,377.11	19,747,035.21	15,501,797.02	80,532,819.96	165,404,222.99
24	IT Consulting	13,908,957.85	16,457,588.23	15,400,544.65	22,633,945.64	60,822,949.84	129,223,986.21
25	Training	2,601,007.04	2,835,299.61	2,698,274.45	1,440,770.40	11,443,707.04	21,019,058.54
26	Grand Total	150,171,920.84	157,112,405.38	140,883,337.48	149,486,372.88	1,007,944,842.21	1,605,598,878.79

Figure 85 - PivotTable Using Calculated Columns for Fiscal Period Groupings

Creating Measures in a PivotTable Using DAX

Consider the need for advanced data analysis capabilities in a PivotTable built from data models. For instance, the PivotTable pictured in **Figure 86** presents data by region, year, and service line; however, it paints an incomplete picture relative to revenue because it does not show the number of revenue-producing sectors in each region. In other words, a simple revenue comparison between regions does not provide the depth of analysis necessary if the South American operation does not operate in all the industry sectors that the North American does.

	A	B	C	D
1	Total Revenue			
2		Asia	North America	South America
3	2020			
4	Advisory Services	6,428,627.69	48,600,315.08	22,837,545.22
5	HR Management	16,848,792.01	157,027,741.00	84,871,403.03
6	IT Consulting	19,120,619.26	143,667,033.79	68,401,036.37
7	Training	2,468,352.90	17,807,218.73	9,575,351.50
8	2021			
9	Advisory Services	12,360,427.55	93,790,984.33	31,050,687.45
10	HR Management	25,869,617.77	238,928,484.35	80,532,819.96
11	IT Consulting	25,900,438.80	198,726,768.70	60,822,949.84
12	Training	4,524,464.40	34,129,956.82	11,443,707.04
13	Grand Total	113,521,340.38	932,678,502.80	369,535,500.41

Figure 86 – Sample PivotTable Analyzing Revenue by Region, Year, and Service Line

You can add a measure to the PivotTable built from the data model to paint a complete picture. A measure is a formula created to use in a PivotTable (or PivotChart) that uses Power Pivot data. You can create measures based on standard DAX aggregation functions, such as **COUNT** or **SUM**. Also, you can generate a formula using DAX expressions. You always use measures in the *Values* area of a PivotTable. However, you should use a calculated column instead of measures if you need to place calculated results elsewhere in a PivotTable, such as a row or column.

When you define a formula for a measure, nothing happens until you add the measure to the PivotTable. Upon adding the measure, the formula calculates each cell in the PivotTable's Values area. Because Excel creates a result for each combination of row and column headers, a measure's calculation can differ in each cell. Excel saves the definition of the measure you create with its source table. It appears in the PivotTable Fields list and is available to all workbook users.

Returning to the example in Figure 86, you can use a **DISTINCTCOUNT** DAX function to analyze the data more thoroughly. **DISTINCTCOUNT** identifies the number of distinct values a PivotTable summarizes at each unique intersection of a column and a row. To add this calculation to the PivotTable shown in Figure 86, click the arrow under **Measures** on the **Power Pivot** Ribbon and then choose **New Measure** to open the **Measure** dialog box pictured in **Figure 87**. In the dialog box, add the formula using the **DISTINCTCOUNT** function referring to the table and the column header in the table on which you want to base the calculation.

The screenshot shows the 'Measure' dialog box with the following fields:

- Table name:** Data
- Measure name:** Revenue Producing Sectors
- Description:** Counts the number of industry sectors generating revenue for each reporting period
- Formula:** =DISTINCTCOUNT(Data[Sector])
- Category:** General (selected from a list including Number, Currency, Date, TRUE/FALSE)

Buttons for 'Check formula', 'OK', and 'Cancel' are also visible.

Figure 87 – Adding a Measure in Power Pivot’s Measure Dialog Box

Once you create the formula and click **OK**, Excel adds the measure to the PivotTable. After you add it, you can use it in any other calculation in the PivotTable. **Figure 88** illustrates the updated PivotTable, complete with the newly appended measures offering greater insight into the revenues than previously available.

	A	B	C	D	E	F	G
1			Asia		North America		South America
2		Total Revenue	Revenue Producing Sectors	Total Revenue	Revenue Producing Sectors	Total Revenue	Revenue Producing Sectors
3	2020	44,866,391.86	22	367,102,308.60	24	185,685,336.12	21
4	Advisory Services	6,428,627.69	15	48,600,315.08	17	22,837,545.22	14
5	HR Management	16,848,792.01	20	157,027,741.00	22	84,871,403.03	19
6	IT Consulting	19,120,619.26	22	143,667,033.79	24	68,401,036.37	21
7	Training	2,468,352.90	18	17,807,218.73	20	9,575,351.50	17
8	2021	68,654,948.52	22	565,576,194.20	24	183,850,164.29	22
9	Advisory Services	12,360,427.55	16	93,790,984.33	18	31,050,687.45	16
10	HR Management	25,869,617.77	21	238,928,484.35	23	80,532,819.96	21
11	IT Consulting	25,900,438.80	22	198,726,768.70	24	60,822,949.84	22
12	Training	4,524,464.40	18	34,129,956.82	20	11,443,707.04	18
13	Grand Total	113,521,340.38	22	932,678,502.80	24	369,535,500.41	22

Figure 88 – PivotTable with Calculated Field Added by Power Pivot

Creating Key Performance Indicators in Power Pivot

Power Pivot provides the ability to develop **Key Performance Indicators (KPIs)** and measure results against the KPIs you establish. KPIs are visual measures of performance that help users quickly evaluate their progress toward meeting specified goals. For example, an organization's sales department could use a KPI to measure monthly sales against sales quotas. Similarly, the accounting department might compare monthly advertising expenses against budgeted amounts, or a human resources department might measure quarterly employee turnover to a target value. Business professionals frequently use KPIs grouped onto a business scorecard to quickly and accurately summarize business activities and identify trends.

Using KPIs can be very useful when you need to show how actual results measure against expectations because KPIs are simply visual representations of performance. According to Microsoft, a "KPI gauges the performance of the value, defined by a Base measure, against a Target value, also defined by a measure or an absolute value."

When building KPIs in Power Pivot, understanding three terminology elements is essential.

1. **Base Value.** A base value is a measure that resolves to a numeric value. The base value, for example, could represent sales or net profit for a period, or it could represent year-to-date revenue through a specified date during the year.
2. **Target Value.** A target value can be a specific numeric value. Alternatively, you can define a target value by a measure that resolves to a numeric value. For example, the sales target for the entire organization could be budgeted sales, a constant. Alternatively, using a measure, you could calculate sales targets – such as last year's sales revenue plus a percentage increase – for each salesperson.
3. **Status Thresholds.** Status thresholds are ranges between the high and low endpoints. Further, status thresholds use graphics (red, yellow, and green lights) to help users determine the base value's progression toward a goal.

To create a KPI using Power Pivot, establish measures to serve as the **Base Value** and **Target Value**. In this case, construct measures using the following DAX formulas to sum sales and target sales for the year.

TotalSales:=SUM(Transactions[SalesAmount])

TotalSalesTarget:=SUM(SalesTargets[2020SalesTarget])

After creating the measures, return to the **Power Pivot** tab of the Ribbon. Click **KPI** followed by **New KPI** to open the **Key Performance Indicator (KPI)** dialog box displayed in **Figure 89**. Select

the **TotalSales** measure as the KPI base field in the **Key Performance Indicator** dialog box. Additionally, set the **TotalSalesTarget** measure as the **Target value**. Finally, adjust the upper and lower **Status thresholds** to **95%** and **90%**, respectively. Click **OK** to complete the process.

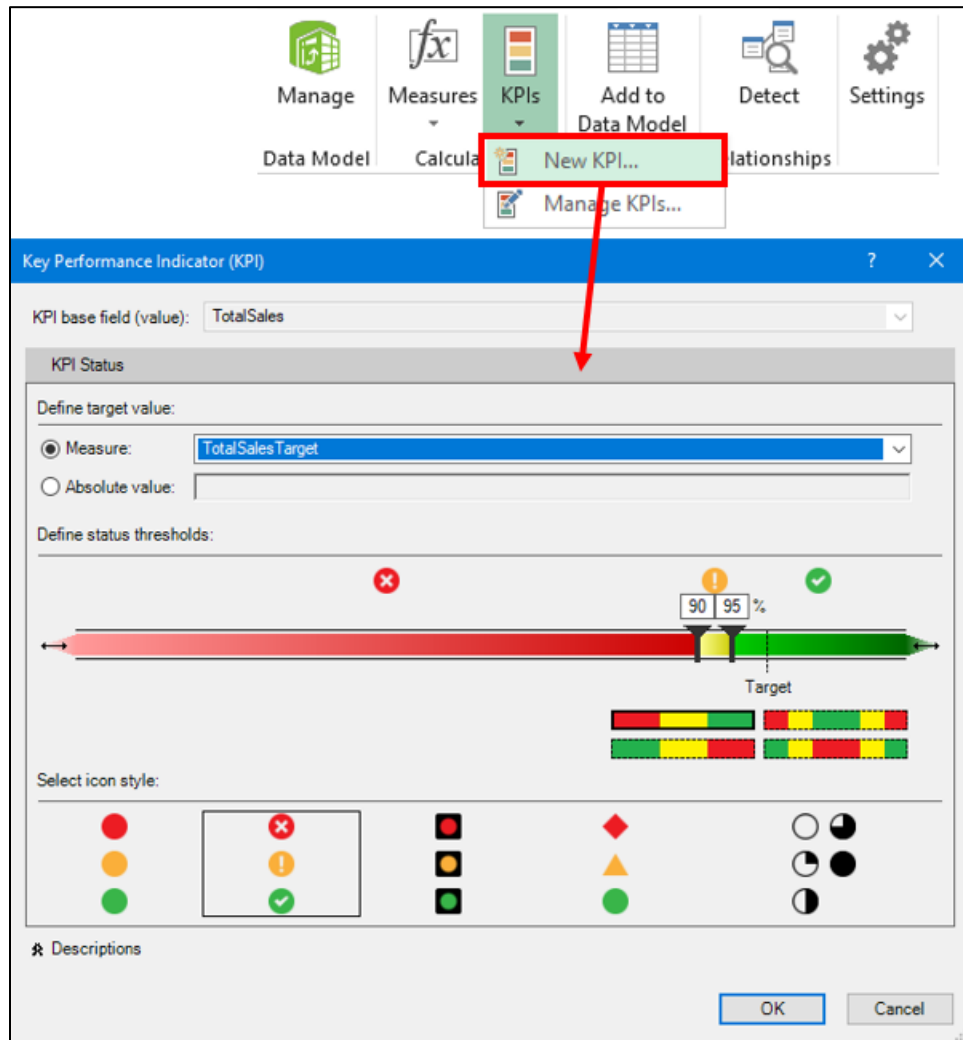


Figure 89 – Defining a New KPI in Power Pivot

As shown in **Figure 90**, each component of the KPI shows up as a new field that you can add to the PivotTable. Note the traffic signal symbol to differentiate a KPI from other fields in the Field List. Add the KPI to the PivotTable to complete the report.

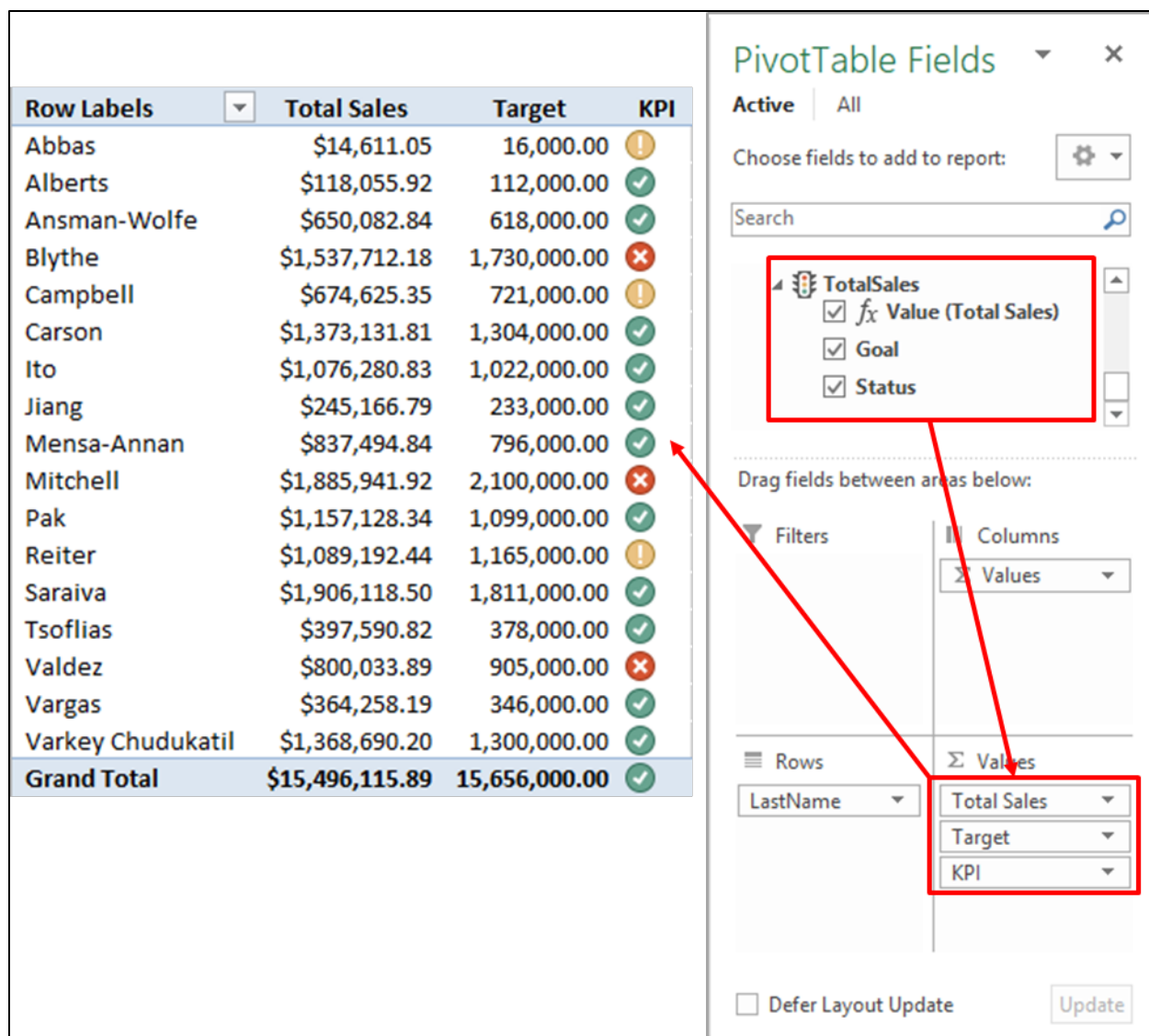


Figure 90 - PivotTable with KPI Added



Given what you now know about KPIs, will you be inclined to include them in some of your Excel-based reports?

A Deeper Dive into DAX

For many, the key to unleashing the power of data models rests in using DAX to create calculations that would otherwise be time-consuming or even impossible. While a complete discussion of the multitude of DAX functions is beyond this session's scope, we will now examine several of these functions to understand better how to work with DAX.

Over 300 DAX functions exist to create calculated columns and measures in your data models. Some DAX functions, such as **CALENDAR**, return a table of values. Others are like Excel functions but with different names and different syntaxes. For example, **TEXT** and **CHOOSE** become **FORMAT** and **SWITCH**, respectively, in DAX. DAX also includes time intelligence functions, such as **TOTALYTD**, **TOTALMTD**, and **TOTALQTD**, which are unavailable in Excel. Lastly, DAX offers several “super-functions,” such as **SUMX**, **RELATED**, and **CALCULATE**, for making calculations that are not otherwise possible using ordinary Excel functions.

While the DAX function library resembles the Excel function library in some aspects, the libraries have many differences, as illustrated in the preceding paragraph. The following examples of working with DAX functions demonstrate how (and why) you should work with selected DAX functions.

FILTER

As shown below, the **FILTER** function returns a subset of a table or expression.

FILTER(<table>,<filter>)

Assume you want to get a count of items sold at the premium level, which you define as anything over \$100. To accomplish this task, you can use the **COUNTROWS** function, which counts the number of rows in the specified table, and the **FILTER** function.

=COUNTROWS(FILTER('Sales', 'Sales'[Sales] > 100))

The first parameter, **Sales**, identifies a table or an expression that results in a table. The second parameter, **'Sales'[Sales] > 100**, represents a Boolean, or true/false expression that evaluates each table row. In this expression, we pass the Sales table to the **FILTER** function and ask it to return any sales over \$100. You will never use the **FILTER** function as a stand-alone function; instead, you will always use it with other functions. In the example above, we use the **FILTER** function to return a subset and count the results.

ALL

You can use the **ALL** function to return all the rows in a table or values in a column, ignoring any filters in place.

ALL(<table> or <column>)

Suppose you have applied a filter to a table, but you also need to summarize all the data from the table, disregarding the filter. The following expression that incorporates the **ALL** function can help you achieve this.

= COUNTROWS(ALL('Sales'))

In this example, we pass the **Sales** table to the **ALL** function, asking it to clear any existing filters. Like the **FILTER** function, you will use the **ALL** function with other functions. In this case, we use **ALL** with **COUNTROWS** to count all sales records. The **ALL** function accepts either a table or a column.

RELATED

The **RELATED** function returns a corresponding value from another table, as shown below.

RELATED(<column>)

So far, we've worked with functions that can help you return a subset or clear any filters on a table or column. For example, suppose we need to filter our sales for only the United States but don't have all the data we need in one table to accomplish this. Fortunately, we have the **RELATED** function, which we can use to retrieve values from one table to another through an established relationship. Further, there is a many-to-one relationship between the **Sales** table and the **SalesGeography** table. Therefore, we can use the following expression that incorporates the **RELATED** function to return a count of sales orders for only the United States.

= COUNTROWS(FILTER(ALL('Sales'), RELATED('SalesGeography'[Countries]) = "United States"))

TOTALYTD / TOTALQTD / TOTALMTD

Time intelligence functions in DAX enable you to manipulate data based on the passage of time, including days, months, quarters, and years. You can use these functions to build and compare calculations over those periods.

TOTALYTD(<expression>,<dates>[,<filter>][,<year_end_date>])

Continuing from the examples above, let's say you would like to see year-to-date total sales. The following expression incorporating the **TOTALYTD** function can easily enable you to do this.

= TOTALYTD(SUM('Sales'[Sales]), 'Dates'[Dates])

The first parameter, **'Sales'[Sales]**, identifies the column you want to aggregate. This parameter could also be an expression that returns a scalar or singular value. The second parameter, **'Date'[Dates]**, is a column that contains dates. Time intelligence functions are beneficial functions that eliminate the need for complicated code in calculating aggregations over commonly used intervals.

CALCULATE

The **CALCULATE** function evaluates an expression in a context modified by specific filters.

CALCULATE(<expression>, <filter1>, <filter2>...)

Assume you are now interested in tabulating all sales for all areas. While you could create some piecemeal expressions to accomplish this, you can achieve the same result efficiently by utilizing the **CALCULATE** function. The following example, which uses the **CALCULATE** function, can generate this result.

= CALCULATE(SUM('Sales'[Sales]),ALL('SalesGeography'))

The first parameter, **SUM('Sales'[Sales])**, identifies the column you want to aggregate. The second parameter, **ALL('SalesGeography')**, represents a Boolean expression that removes any filters on the SalesGeography table, including page-level filters. The **CALCULATE** function is one of the most valuable functions in DAX. It is helpful to think of the **CALCULATE** function as a supercharged “IF” statement. A few rules apply to the **CALCULATE** function: the filter parameters cannot reference measures, and expressions cannot use any functions that scan or return a table. Many professionals use the **CALCULATE** function with aggregation functions, and although the filter parameters are optional, at least one is generally used.

SUMX

In this example, the data model contains tables with columns for *price*, *units*, and *discount* but not for *net revenue*. Therefore, creating a report summarizing *net revenue* will require a calculated column or a measure. While you can use Power Pivot to create a calculated column, a measure is a better solution. You can create measures from the Ribbon, the Fields pane, or the data model. To establish the measure, select the table for storing the measure. Then, enter the following formula in the **Formula** box to create the calculation.

=SUMX(fTrans,ROUND(RELATED(dProducts[Price])*fTrans[Units]*(1-fTrans[Discount]),2))

Note how Formula AutoComplete prompts you during the formula-building process. Click **Check formula** to make sure that there are no errors in the procedure.

SUMX is a DAX function that creates a calculation for each record in a table and then aggregates the calculations’ results across the entire table. (Other DAX functions that produce comparable results are **AVERAGEX**, **COUNTX**, **COUNTAX**, **MINX**, **MAXX**). The **RELATED** function calculates values by reference to columns in other related tables. In this case, **Units** and **Discount** fields reside in the **fTrans** table, but **Price** resides in the related **dProducts** table, hence using the **RELATED** function. Note that DAX lookup functions, such as **RELATED**, require relationships to exist between the tables.

Observe how measures appear in the Field pane. As shown in **Figure 91**, we created a single measure to report the net revenue for each reporting object in the context of any filters applied to the report.

NetRevenue	Column Labels		
Row Labels	2017	2018	Grand Total
East	\$75,242.89	\$70,898.64	\$146,141.53
Arturo Francis	\$3,315.36	\$2,500.93	\$5,816.29
Austin Reynolds	\$5,881.39	\$3,386.55	\$9,267.94
Bobbie Ryan	\$3,894.06	\$4,452.48	\$8,346.54
Della Jensen	\$11,481.20	\$13,353.51	\$24,834.71
Eddie Green	\$4,953.86	\$3,552.04	\$8,505.90
Edwin Malone	\$9,508.34	\$7,498.41	\$17,006.75
Herman Williams	\$4,946.10	\$5,622.52	\$10,568.62
Isabel Cross	\$13,690.88	\$16,307.59	\$29,998.47
Lorraine Gibson	\$4,755.11	\$4,459.02	\$9,214.13
Marion Glover	\$5,598.35	\$2,213.97	\$7,812.32
Nellie Joseph	\$3,343.11	\$2,528.12	\$5,871.23
Sonya Mullins	\$3,875.13	\$5,023.50	\$8,898.63
MidWest	\$161,174.45	\$164,935.09	\$326,109.54
South	\$71,396.04	\$87,423.81	\$158,819.85
West	\$136,134.03	\$138,497.26	\$274,631.29
Grand Total	\$443,947.41	\$461,754.80	\$905,702.21

Figure 91 – PivotTable Summarizing the Calculated NetRevenue Measure

Calendar Tables – The Key to Period Reporting and Time Intelligence

Date grouping by day, week, month, quarter, and year is available in ordinary PivotTables but not in PivotTables built on the Excel Data Model.¹⁰ The preferred way to handle date-based reporting is with a calendar table. A calendar table lists dates and columns for their related week, month, quarter, and year. Once you create a calendar table, you can use it for period-based reporting. You can also use it to facilitate DAX time intelligence functions, such as **DATESYTD**, **DATESMTD**, **DATESQTD**, **TOTALYTD**, **TOTALMTD**, **TOTALQTD**, **SAMEPERIODLASTYEAR**, and others.

You can create calendar tables easily in the Data Model or Excel and then added to the Data Model. Begin by entering a start date for the calendar in a column named **Date**. Then, use the **Fill, Series** command on the **Home** tab to create the list of required dates. In this example, create a date column that contains the dates 1/1/2015 through 12/31/2025. Next, convert the **Date**

¹⁰ Auto-date grouping in PivotTables built on the Excel Data Model became available in Excel 2016. However, date groups do not facilitate the use of the DAX time intelligence functions and are always based on calendar periods. Those organizations with non-calendar fiscal periods will require a date table or use of DAX functions discussed elsewhere in this text.

column to an Excel table by clicking in the table and clicking **Format as Table** on the Ribbon's **Home** tab. Complete the process by choosing the desired design in the **Table** gallery and clicking **OK**. Next, rename the table to **DimCalendar** in the **Table Name** box on the **Table Tools, Design** tab. In adjacent columns, create headings and formulas for the month number, month name, quarter number, quarter name, and year. You must manually mark a Calendar table created as a date table. Finally, add the table to the Data Model and click **Mark as Data Table** on the **Design** tab.

The following table displays the formulas used for a fiscal year date table. Note the formulas assume an October 1 start date. Adjust the quarter sequence in the FiscalQuarterNumber formula to match your defined periods and change the start date month number in the FiscalYearName formula to your fiscal year start month.

Column Name	Formula Using Excel Functions
MonthNumber	=MONTH([@Date])
Month	=TEXT([@Date],"mmm")
QuarterNumber	=CHOOSE(MONTH([@Date]),1,1,1,2,2,2,3,3,3,4,4,4)
Quarter	="Q"&[@QuarterNumber]
Year	=TEXT([@Date],"yyyy")
FiscalQuarterNumber (for a fiscal year beginning October 1)	=CHOOSE(MONTH([@Date]),2,2,2,3,3,3,4,4,4,1,1,1)
FiscalQuarter	="Q"&[@FiscalQuarterNumber]
FiscalYear (for a fiscal year beginning October 1)	=IF(MONTH([@Date])<10,YEAR([@Date]),YEAR([@Date])+1)

If you create your Date table manually in the Data Model, use the DAX formulas in the following table. Note the use of **FORMAT** instead of **TEXT** and **SWITCH** instead of **CHOOSE**. If the entity uses a calendar year as its fiscal year, use "q" as the date format code to calculate the quarter number with the FORMAT function.

Column Name	Formula Using DAX Functions
MonthNumber	=MONTH([Date])
Month	=FORMAT([Date],"mmm")
QuarterNumber	=FORMAT([Date],"q")
Quarter	="Q"&[QuarterNumber]
YearName	=FORMAT([Date],"yyyy")
FiscalQuarterNumber (for a fiscal year beginning October 1)	=SWITCH(MONTH([Date]),10,1,11,1,12,1,1,2,2,2,3,2,4,3,5,3,6,3,7,4,8,4,9,4)
FiscalQuarter	="Q"&[FiscalQuarterNumber]
FiscalYear (for a fiscal year beginning October 1)	=IF(MONTH([Date])<10,YEAR([Date]),YEAR([Date])+1)

Automatic Creation of Calendar Tables

For those using Excel 2016 and newer, the Data Model can automatically create a basic Calendar table if a table already existing in the Data Model contains dates. To utilize this functionality, click **Manage** on the **Power Pivot** tab in Excel to open **Power Pivot for Excel**. Click **Date Table, New** on the **Design** tab to create a Calendar table based on a calendar year, as shown in **Figure 92**. You can use DAX formulas to establish other needed columns or create columns that match your fiscal year.

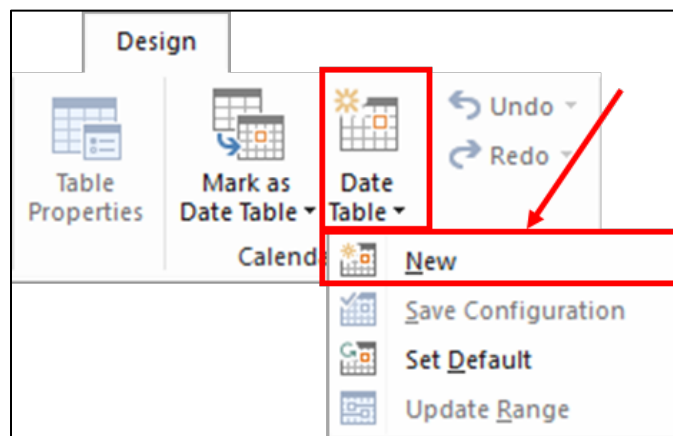


Figure 92 – Creating a Calendar Table Automatically in Power Pivot

A Calendar table created automatically in Excel 2016 and newer will likely need a few modifications to optimize it for use in a PivotTable report. Among the changes suggested are those indicated below.

1. Format the **Year** column as **Text** so that number formats in your worksheet won't alter the period labels in your PivotTable.
2. The process creates a **Date Hierarchy** automatically in the **Calendar** table. However, you can modify the hierarchy in **Diagram View** to include any additional desired columns, such as **Quarters** or **Weeks** columns.
3. Establish a relationship between the **Calendar** table and your transaction table(s) on the **Date** column.
4. Hide any unnecessary columns from view to reduce confusion. Right-click on a table column in **Data View** and select **Hide from Client Tools** to hide a column from users in the PivotTable Field List. For example, you may want to hide **MonthNumber** or **QuarterNumber** because they are unnecessary for reporting. Note that the column will become greyed out in the Data Model. Right-click on a table column and select **Unhide from Client Tools** to unhide a column.

Using CUBE Functions to Summarize Transactions from Data Models

While a dominant theme in this session is to avoid using formulas when creating reports, there will likely be times when this objective is impossible. Perhaps, for example, you need to develop summarizations of data and display that in a report format that is not attainable with a PivotTable. CUBE functions allow you to create formulas that summarize your data model's transactions in such situations. Additionally, because this technique is formula-based, you can enter your formulas in virtually any cell in the data model's workbook. Further, you don't need to establish the CUBE formulas if you have already created a PivotTable from the data model. Instead, you can convert the PivotTable into a series of formulas and rearrange the cells to meet your reporting needs.

Excel's Seven CUBE Functions

Excel offers seven CUBE functions to use in formulas to summarize data from a data model. **Table 2** provides a summary of each of these functions.

Function	Description
CUBEVALUE	Returns an aggregated value from the cube.
CUBEKPIMEMBER	Returns a key performance indicator (KPI) property and displays the KPI name in the cell. A KPI is a quantifiable measurement, such as monthly gross profit or quarterly employee turnover, which you can use to monitor an organization's performance.
CUBEMEMBER	Returns a member or tuple from the cube. Use to validate that the member or tuple exists in the cube.
CUBEMEMBERPROPERTY	Returns the value of a member property from the cube. Use this function to validate that a member name exists within the cube and return the specified property for this member.
CUBERANKEDMEMBER	Returns the n^{th} , or ranked, member in a set. Use to return one or more elements in a set, such as the top sales performer or the top 10 students.
CUBESET	Defines a calculated set of members or tuples by sending a set expression to the cube on the server, which creates the set and then returns that set to Microsoft Excel.
CUBESETCOUNT	Returns the number of items in a set.

Table 2 – Seven CUBE Functions Available in Excel

Of the seven functions listed above, the most used is **CUBEVALUE**. This function is the logical equivalent of a traditional SUMIFS function operating on a data model. More specifically, the CUBEVALUE function summarizes data from the data model that meets specific criteria. The exact syntax for this function appears below.

**CUBEVALUE(connection, [member_expression1],
[member_expression2], ...).**

In formulas that use the CUBEVALUE function, “connection” is the cube's name that contains the data, and “member_expression” is an argument that defines the data to summarize. To illustrate, consider the simple data model shown in **Figure 93**.

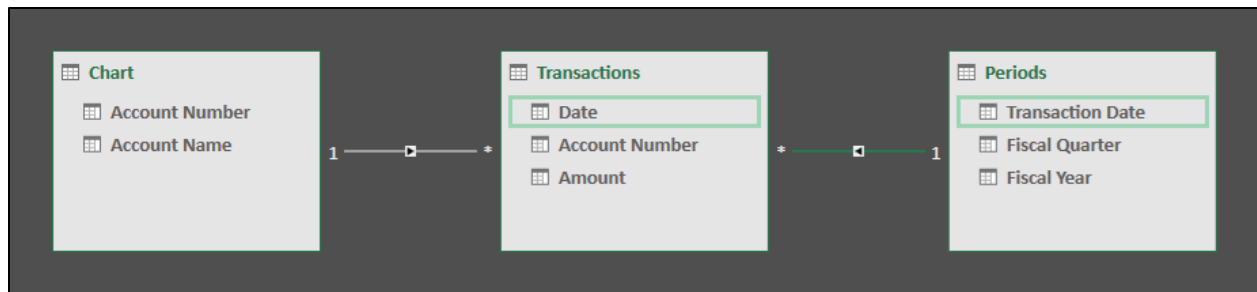


Figure 93 - Sample Data Model for CUBEVALUE Formula

Next, consider the following formula based on a CUBEVALUE function entered in a cell in the workbook where the data model shown in Figure 93 resides.

=CUBEVALUE("ThisWorkbookDataModel",\$A\$3,\$A6,B\$4)

Further, suppose the following entries were in the referenced cells in the formula.

- **A3** contained the phrase "Sum of Amount."
- **A6** contained the phrase "Accounting Fees."
- **B4** contained "2019."

Given those assumptions, the CUBEVALUE formula would return the Sum of the Amount field from the data model for all transactions in the data model when the account was "Accounting Fees" and the year was "2019." In other words, in this illustration, the CUBEVALUE-based formula acts much like a more traditional SUMIFS function in Excel with one glaring difference – it can access the potentially large volume of data stored in the data model!

Making CUBE Functions Easier to Create

CUBE formulas will intimidate many Excel users because they can quickly become rather complex. Fortunately, there is an uncomplicated way to create your formulas based on CUBE functions without losing any of the formula's valuable traits. This technique involves three simple steps.

1. Create a data model that contains the data you need to summarize.
2. Create a PivotTable from the data model.
3. Convert the PivotTable to formulas.

To illustrate, consider the data shown in **Figure 94**. In this example, a PivotTable created from a data model exists in the workbook. Once the PivotTable is in place, you could choose **PivotTable Analyze, OLAP Tools, Convert to Formula**. Upon doing so, Excel will convert the PivotTable to a series of CUBE formulas, as shown in the lower portion of the graphic. After Excel creates these formulas, you can drag and drop the results' cells to various locations to achieve your desired report format. Importantly, regardless of how you rearrange the results, the formulas maintain their connection to the data model so that your reports will update as the data changes.

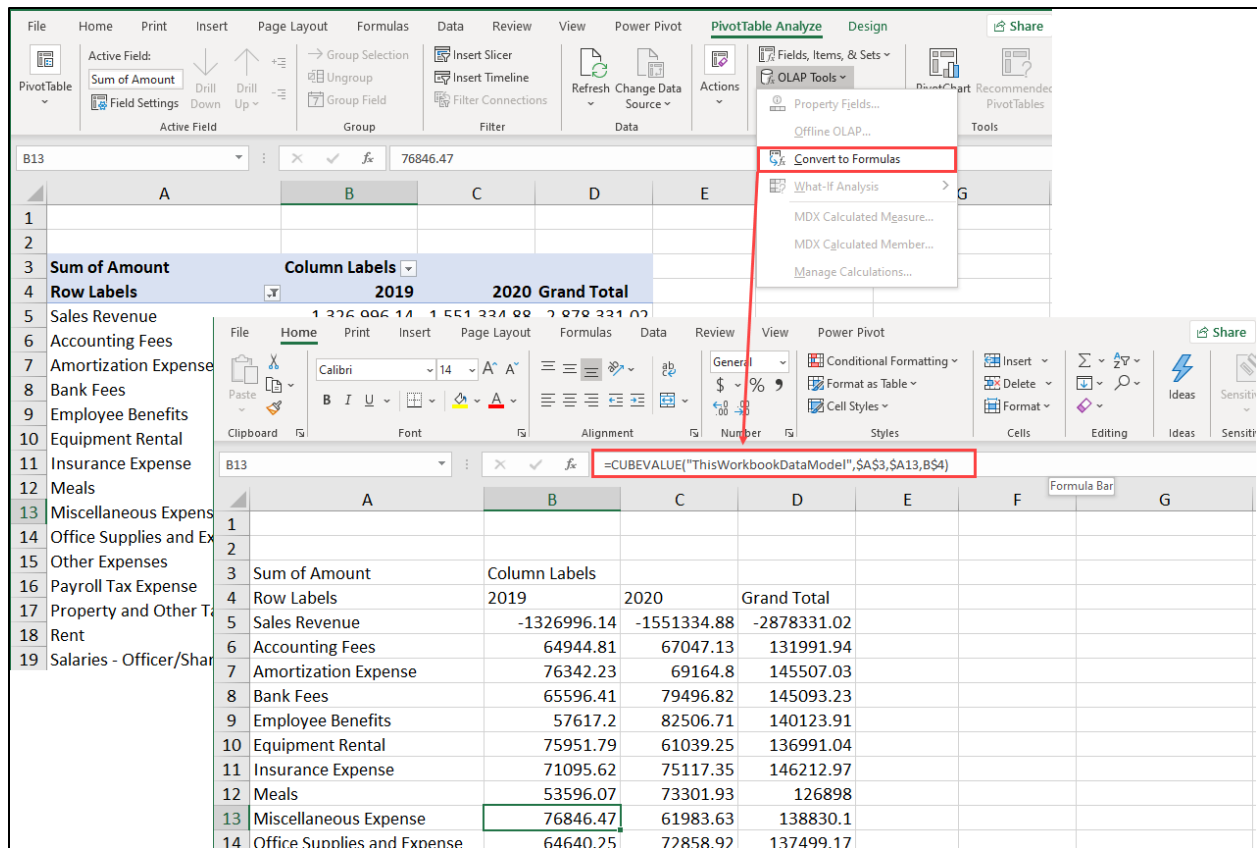


Figure 94 - Converting a PivotTable to a Series of CUBE Formulas

At this point, many would rightfully ask: *Why go to the trouble of creating CUBE formulas to do what PivotTables constructed from data models already do automatically?* It's a fair question. The most obvious use case associated with CUBE formulas is the flexibility to create or move them to any location you might desire in the workbook. Ordinarily, PivotTables' structures are somewhat rigid; you cannot simply click and drag cells within a PivotTable to cells outside the PivotTable to meet your reporting needs. For example, you cannot necessarily rearrange a PivotTable so that the summarizations align with a predefined form such as a tax return. However, you can create CUBE formulas to summarize data from your data model in any cell(s) in a workbook. With these formulas, your reporting is not limited by a PivotTable's structure because you can place your CUBE formulas anywhere you desire in the workbook. These traits yield unparalleled power to summarize and analyze your data while simultaneously achieving the formatting demands you may be facing.

You can learn more about the seven CUBE functions at <https://k2e.fyi/cubefunctions>.

Slicer and Timeline Filters for Improved End-User Experiences

Chapter

6

In addition to the industrial-strength features discussed so far, other tools support Next Generation Excel reporting. Slicers and Timelines, two easy-to-use tools that simplify and strengthen the process of filtering PivotTables, are chief among these. In this chapter, you will learn how to create and use these tools to improve end-user experiences when creating Excel-based reports.

Learning Objectives

Upon completing this chapter, you should be able to:

- Create and utilize Slicer filters on PivotTables and PivotCharts;
- Construct Timeline filters and apply them to PivotTables and PivotCharts; and
- Link Slicers and Timelines to multiple PivotTables and PivotCharts.

Easy (and Advanced) Filtering with Slicers and Timelines

Microsoft added **Slicers** to Excel in 2010 and **Timelines** to Excel in 2013. Both these tools provide graphical options for filtering PivotTables and PivotCharts. Specifically, you can use Slicers to filter your PivotTable-based reports by any of the dimensions in the data model. Additionally, you can use Timelines to filter these same reports based on date-oriented data in the data model.¹¹

Adding and Working with Slicers

To add a Slicer to a PivotTable or PivotChart, choose **Insert Slicer** from the **PivotTable Analyze** tab of the Ribbon to open the **Insert Slicers** dialog shown in **Figure 95**. Then, check the box next to each field you want to include as a Slicer filter. Finally, click **OK** to select the fields you want to use to filter your PivotTable. Excel then adds each field as a Slicer and the Slicer to your PivotTable.

¹¹ Presently, the use of Timelines remains limited to PivotTables and PivotCharts. However, you can use Slicers to filter not only PivotTables and PivotCharts but also to filter data in Tables.

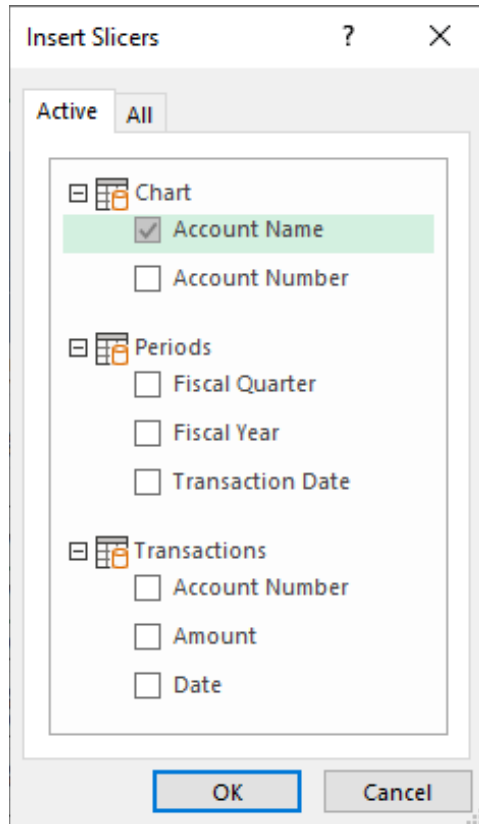


Figure 95 - Insert Slicers Dialog Box

Once added to the PivotTable, you can use the Slicer to click on the items you want to see in your PivotTable. You can also reposition the Slicer by clicking and dragging it to your desired location. Importantly, you can click on the Slicer and access the Ribbon's **Slicer** tab to change a Slicer's appearance, as shown in **Figure 96**. Further, you can add multiple Slicers to a given PivotTable to accommodate more advanced filtering needs.

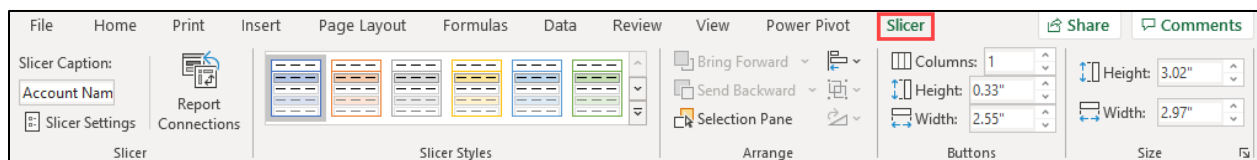


Figure 96 - Accessing the Slicer tab of the Ribbon

Adding and Working with Timelines

Timelines work almost identically to Slicers, except they use a click-and-drag approach to filtering data based on dates. First, click **Insert Timeline** from the Ribbon's **PivotTable Analyze** tab to add a timeline. Next, choose the date-oriented field(s) to serve as the basis for your Timeline and click **OK**, as shown in **Figure 97**, to complete the process of adding the Timeline.

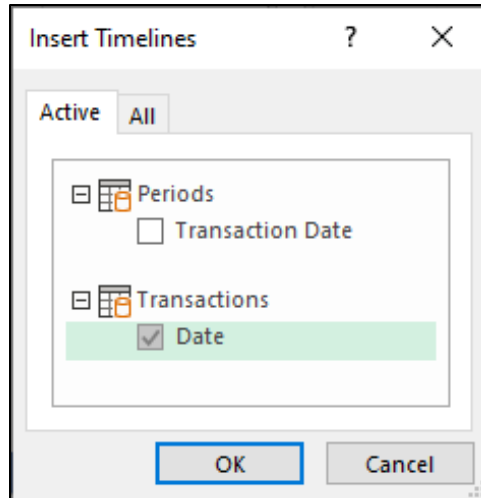


Figure 97 - Insert Timelines Dialog Box

After adding the Timeline, you can click and drag horizontally on its' edges to filter the data in your PivotTable based on dates, as shown in **Figure 98**.

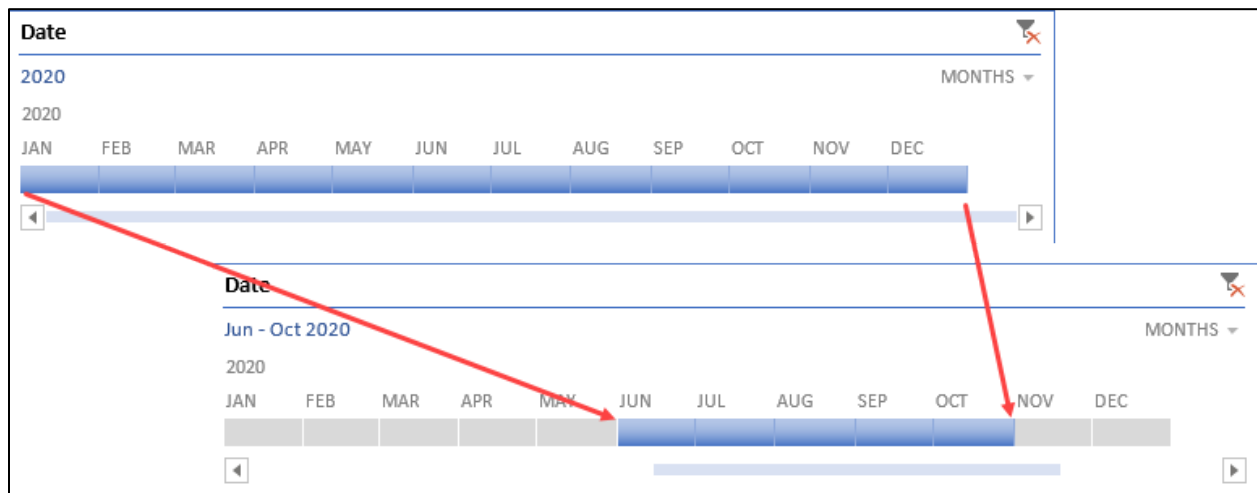


Figure 98 - Using a Timeline to Filter by Date

You can also change the interval by which the Timeline filters data. For example, clicking “Months” in the upper right corner of the Timeline allows you to choose *Years, Quarters, Months, or Days* as the reporting interval.

Advanced Functionality Associated with Slicers and Timelines

Although some users already incorporate Slicers and Timelines into their PivotTables and PivotCharts, most have not yet realized the deeper functionality of these features. More specifically, you can use a single Slicer or Timeline to filter multiple PivotTables or PivotCharts in

the same workbook simultaneously. This functionality exists when PivotTables and PivotCharts use the same data source, which is known as the *PivotCache*.

To illustrate how to take advantage of this feature, suppose you have a workbook containing three PivotTables, and all three PivotTables draw data from the same Pivot Cache. Further, assume that you named the PivotTables in the workbook *Summary*, *Salesperson*, and *Product* and created a Slicer for the *Summary* PivotTable. To configure the Slicer to filter the other PivotTables to the same set of criteria, right-click on the **Slicer**. Next, choose **Report Connections**. Then, check the boxes in the **Report Connections** dialog box for each PivotTable you want the Slicer to affect. Finally, click **OK** to complete the configuration. Whenever you use the Slicer, it will filter the data in all the PivotTables to which it is connected. **Figure 99** illustrates this process.

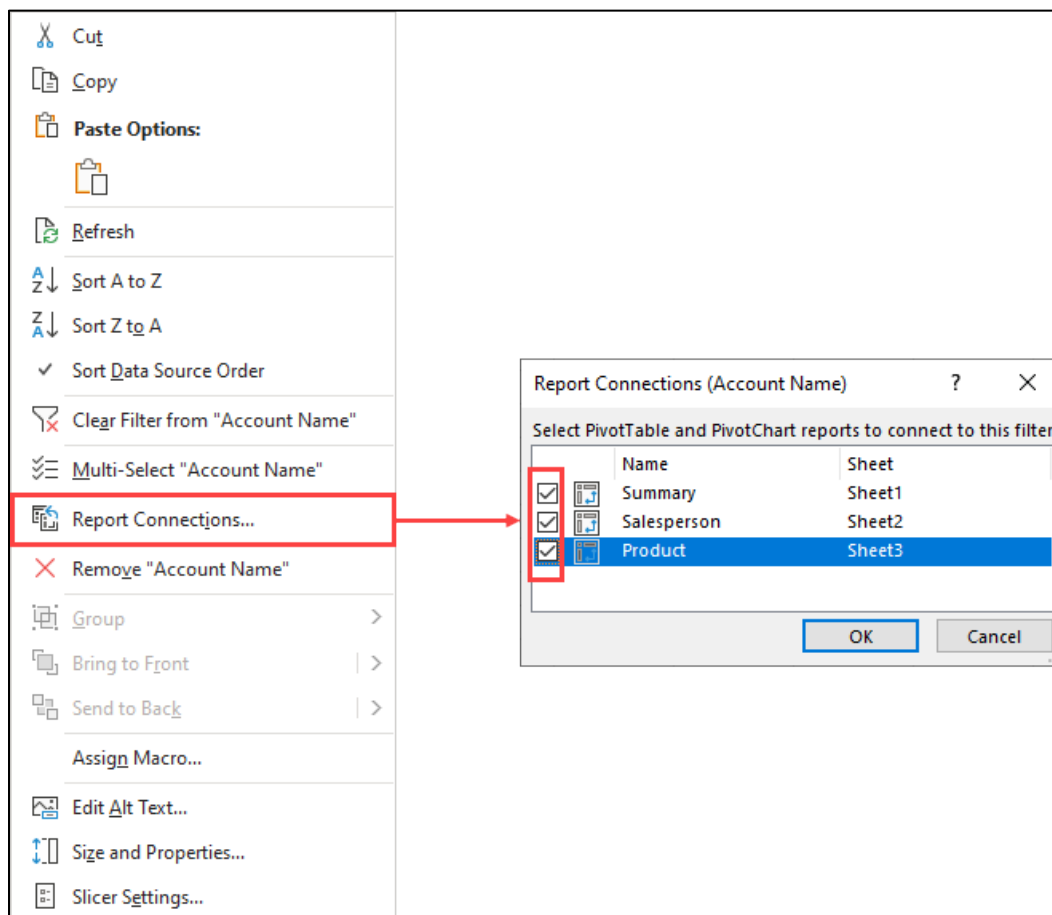


Figure 99 - Using a Single Slicer to Filter Multiple PivotTables

The process for controlling multiple PivotTables or PivotCharts from a single Timeline is virtually identical. Begin by adding a Timeline to a PivotTable or PivotChart with other PivotTables or PivotCharts using the same PivotCache. Then, use the abovementioned technique to connect the Timeline to the additional PivotTables or PivotCharts.

It is important to note that this technique does more than provide convenience. It also helps you ensure that all related PivotTables and PivotCharts filter to the same set of criteria. To illustrate, suppose you have ten PivotTables in a workbook, and all ten use the same Pivot Cache. Without Slicers and Timelines, you must apply filters manually to all ten PivotTables. If you accidentally established a filter on one of the reports inconsistent with the other nine, you have incomparable data in the ten reporting objects. Of course, this could lead to potentially compromised decisions. However, this “one filter/many reports” approach reduces the likelihood of working with incomparable data. Further, you will gain time in your day because applying one filter across ten reports will undoubtedly be faster than applying ten filters.



Regarding Slicers and Timelines, are you now inclined to add them to your PivotTables and PivotCharts? Do you envision connecting multiple PivotTables and PivotCharts to a single Slicer or Timeline?