K2's Business Intelligence, Featuring Microsoft's Power BI Tools

Continuing Professional Education Series

K2 Enterprises www.k2e.com

K2E Canada Inc. www.k2e.ca

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
905-746-1581

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

Table of Contents

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 relevant information about 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 high-quality educational practices that help you use and understand the tools and concepts discussed in this session more effectively.

We designed our course presentation style for busy professionals. Through numerous short case studies and feature reviews, we deliver concise, easy-to-understand, easy-to-absorb information.

To maximize your time and the value of attending this course, we have carefully filtered out trivial and obscure features. Our goal is not to cover every keystroke and menu option, but to provide a fast-paced course that focuses on state-of-the-art information technology and how it can benefit you and your staff.

About K2 Enterprises

For over three decades, our team members have delivered more than 45,000 technology-focused seminars, webinars, and conferences worldwide. To stay on top of technology, our team members read numerous technology trade magazines, attend conferences, and speak with technology companies each year to stay current with the latest information for our courses. Also, we experience everything we teach and recommend to our audiences. Because our shareholders and associates are based in different states and provinces, using technology efficiently and appropriately is essential to our business's success and survival.

Our Mission Statement

K2 aims to develop and deliver the highest-quality technology seminars and conferences for business professionals. We work cooperatively with professional organizations such as state CPA societies, associations of Chartered Professional Accountants, and technology vendors. 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

Full-day Seminars

- 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 Next Generation Excel Reporting
- 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 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 Online Environments
- K2's Improving Productivity With Microsoft 365Cloud Applications And Services
- 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

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
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.

Gaining Insight and Creating Competitive Advantage with Business Intelligence

Chapter 1

One of today's hottest topics in organizations of all sizes is Business Intelligence (BI). Increasingly, managers at all levels and disciplines turn to BI to help them understand what is truly happening with their organizations, regardless of size, and how they can improve overall performance. This chapter will teach you how to use BI to turn raw data into meaningful insight and competitive advantage.

Learning Objectives

Upon completing this chapter, you should be able to:

- Define Business Intelligence and describe examples of situations where it can be helpful.
- List the meaningful characteristics of successful BI efforts.
- Discuss the importance of "do it yourself" BI.
- Identify some of the leading BI tools available today.

What Is Business Intelligence?

Countless definitions for BI exist, and for purposes of this course, we borrow from many of these definitions to create the following.

> *Business Intelligence is a set of procedures, tools, and technologies working in concert to transform raw data, potentially vast quantities of raw data, into meaningful information. Managers can use this information to help make decisions and prescribe courses of action that optimize organizational performance.*

Although many use the term "business analytics" interchangeably with BI, we will consider business analytics a tool BI uses in this course. First, business professionals use business analytics tools to build data analyses, forecasts, and projections. Then, they use BI tools to present this information in reports, dashboards, and other formats.

Without question, "big data" relates to the concept of BI. Over time, organizations accumulate an ever-increasing cache of data regarding sales, purchases, human resources activities, customer service/satisfaction scores, market demographics, etc. Collecting data for the mere sake of having access to the information provides no competitive advantages. However, those

organizations that leverage this data in their BI initiatives often find that they can improve efficiencies and profitability as a direct outcome of these efforts.

Where Is Business Intelligence Useful?

BI efforts can be helpful in virtually all departments of all types of organizations, including governmental units and not-for-profit organizations. However, until recently, many thought BI to be available only to large, enterprise-class organizations. This reasoning was primarily based on the resources required to implement the necessary procedures, tools, and technologies successfully. However, the advent of Cloud-based BI platforms and desktop-based BI applications means that organizations of all sizes can benefit from BI without investing significant resources. Further, because the cost of entry can be meager, your BI efforts' return on investment (ROI) can prove extraordinary.

One area of applied BI that has grown exponentially in the past five years is embedded BI tools in accounting applications, including those deployed on-premise and Cloud-based solutions. From solutions targeted to small businesses – examples include QuickBooks, Sage 50Cloud, Microsoft Dynamics 365, and Xero – to mid-market solutions – for instance, Epicor, Open Systems, and Acumatica – to enterprise-class applications – such as Microsoft Dynamics AX, SAP, and Workday – most accounting applications today offer some semblance of BI dashboards. These BI tools are “organically” developed by the software publisher and sometimes embedded into the application. In other cases, third-party developers create tools to integrate with the core applications. Regardless, the result is the same – stakeholders can utilize BI dashboards to facilitate real-time or near-real-time access to critical information that they can use to help meet business objectives.

Situations Ideal for Business Intelligence Efforts

In particular, the ROI on BI can be exceptionally high in four specific areas.

1. Addressing the need for more timely and improved decisions
2. Automating repetitive reports
3. Identifying trends and opportunities
4. Motivating team members and encouraging teamwork

Addressing the Need for More Timely and Improved Decisions

Consider the reporting practices in use in many organizations today. During the first five to ten days after the end of an accounting period, staff members work furiously to “close the books” and “run the numbers.” Then, once all team members complete their assigned tasks, they create sets of two-dimensional, static reports and distribute them to organizational stakeholders, who receive the information up to two weeks after the end of the period. Further, assuming the accounting period consists of four weeks, the information provided describes transactions recorded *up to six weeks previously* – hardly real-time information!

BI efforts can help address this problem by providing opportunities for managers to make decisions based on real-time information. BI tools typically connect to underlying databases in real-time, so the dashboards, reports, and other visualizations produced by the BI tools facilitate more timely and improved decisions because they reflect current information. Further, with access to mobile devices, information workers can stay informed about critical aspects of managing their operations, even if they are away from the office for extended periods. All this translates into more timely and improved decisions. For example, the BI dashboard pictured in **Figure 1** is accessible from a mobile app on various mobile devices.

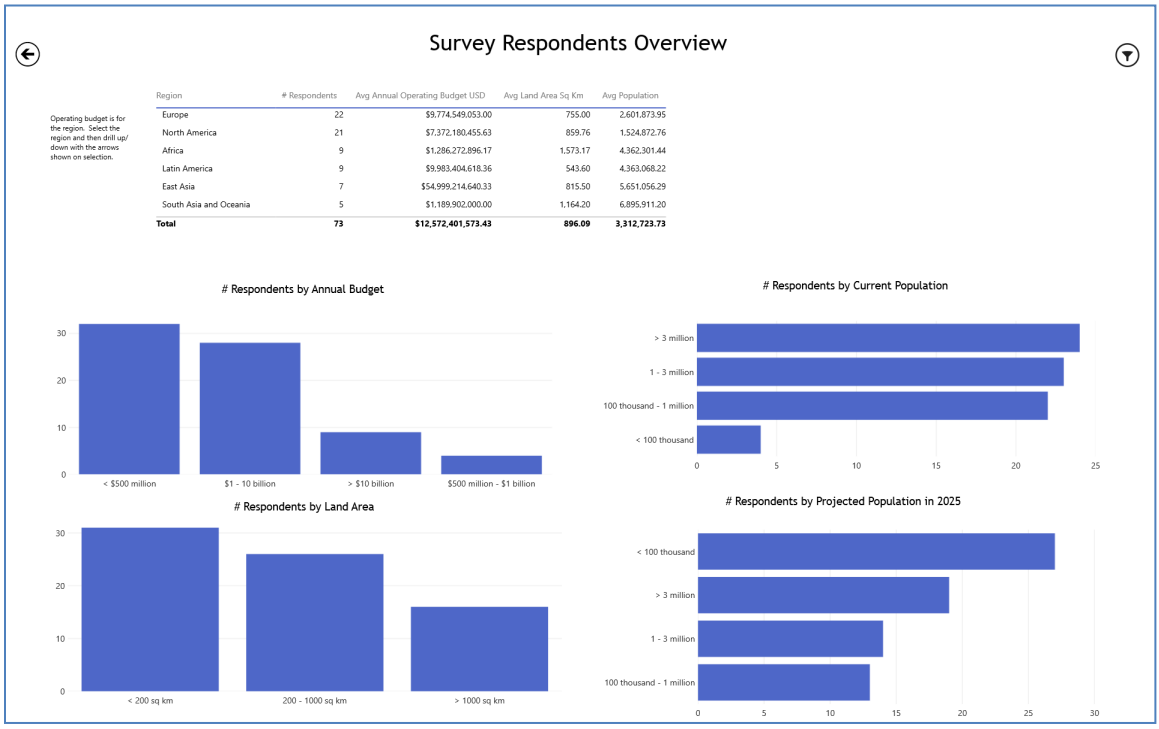


Figure 1 - Sample BI Dashboard Accessed from Mobile App

Further, as shown in **Figure 2**, even though a user might access the dashboard from a mobile app, there is no functionality loss because filter options remain fully functional and operational.

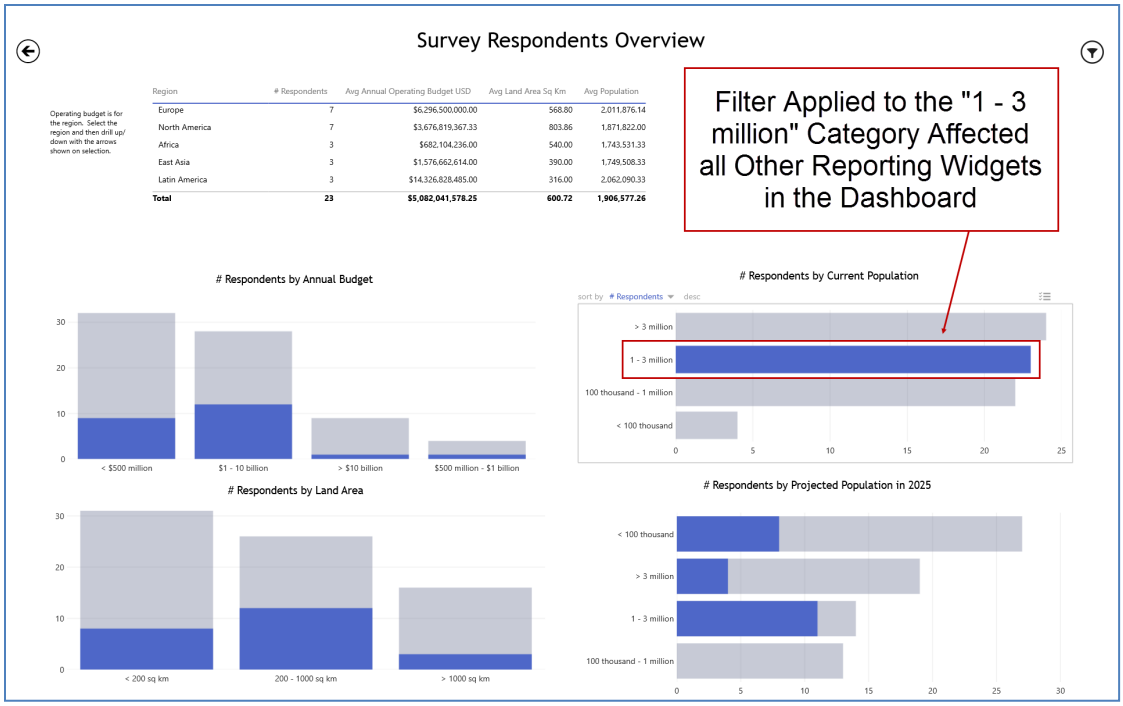


Figure 2 - Sample BI Dashboard with Filtered Responses

Automating Repetitive Reports

Most, if not all, reports and dashboards generated by BI tools follow the “build once, refresh many” concept. This concept means the BI tools typically maintain permanent connections to various data sources and extract the information needed automatically from these data sources to produce multiple reporting objects by merely refreshing the report. This process contrasts with many business professionals' manual procedures to update periodic reports, dashboards, and other analyses and schedules.

Team members spend significant time and effort locating the data needed to meet a particular reporting objective in the more traditional environment. Then they “massage” and cleanse the data and manually load it into the target report or dashboard. Of course, these manual efforts are barriers to accessing even routine reports and create a disincentive for accessing data that could improve decisions. Further, opportunities for errors increase significantly each time team members use manual procedures to manipulate the data. Automating repetitive reports with a “build once, refresh many” philosophy is a best practice for any organization, and BI tools help realize that objective.

Identifying Trends and Opportunities

Managers can improve organizational performance by identifying positive and negative trends soon after they emerge. Once they identify these trends, managers can capitalize on the positive trends and stem the negative ones. As implied earlier, BI tools facilitate this process by providing real-time insights into an organization's activities. Further, managers can access their BI tools

regardless of location, meaning they can use them to identify trends and opportunities even when mobile.

To illustrate, consider the BI reporting object shown in **Figure 3**. A sales manager could use this tool to identify relevant trends associated with sales produced by each salesperson. The sooner the sales manager can identify sub-par performance and correct it, the more likely an individual salesperson will meet their sales target, and the entire team will perform at the prescribed level. Likewise, if the sales manager identifies a positive trend associated with a specific salesperson, they might discuss actions that drive improved performance with that team member. Once identified, the sales manager might share these tactics with the remainder of the team to elevate the performance of all team members.

Figure 3 - BI Reporting Object to Identify Sales Trends

Given the above, it is apparent that using BI tools to identify trends and opportunities represents an outstanding opportunity to improve organizational performance and drive ROI on the BI project.

Motivating Team Members and Encouraging Teamwork

One way to motivate and encourage team members is to ensure they know their performance expectations and provide continual feedback on how their performance compares to these expectations. This feedback should communicate results versus expectations frequently and with great clarity. For example, a BI reporting tool might automatically launch each morning when team members begin their workday and turn on their computers. The BI report could indicate results compared to expectations for the previous day, month-to-date, quarter-to-date, and

year-to-date horizons. Such communications reinforce positive performance and show where improvement is necessary. For example, a dashboard like the one pictured in **Figure 4** might help motivate a salesperson working for a technology consulting practice to meet monthly sales targets.

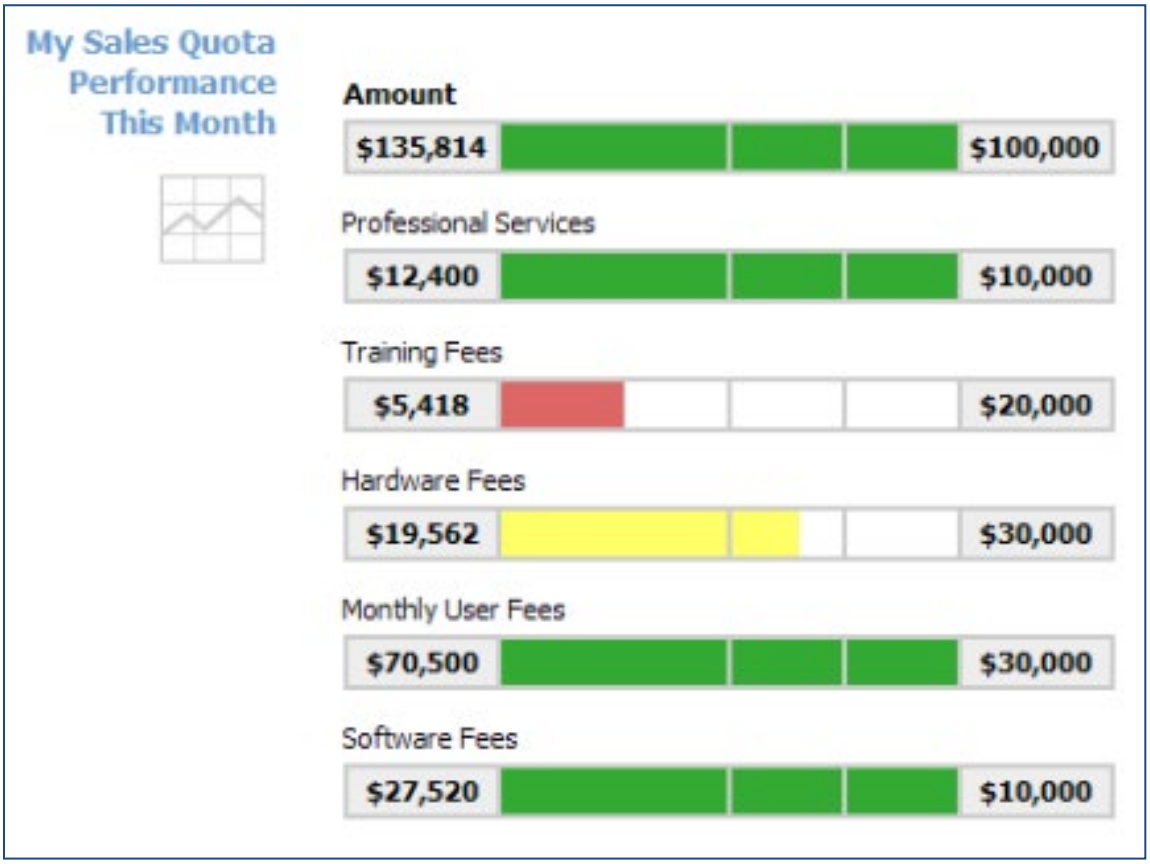


Figure 4 - Sample Dashboard Measuring Sales Results against Quotas

In addition to motivating individual team members, BI tools can improve intra-organizational teamwork and cooperation. For example, the BI dashboard pictured in **Figure 5** might help communicate vital organizational metrics. Further, this dashboard might motivate team members from multiple departments to work together to solve specific issues, especially if a portion of their compensation depends on the organization meeting goals and targets reported in the dashboard.

Figure 5 - Sample Corporate Dashboard

Like BI reporting objects targeted to individuals, better results will probably materialize if the corporate-level dashboards appear frequently and communicate their intended messages.

Characteristics of Successful BI Efforts

Successful most BI implementations possess specific characteristics. First, BI reporting objects must depend on real-time or near-real-time data. The pace at which businesses operate today means that reports that use dated, stale data are of little value when identifying trends, making decisions, and gaining competitive advantages. Therefore, BI tools that connect directly to external data sources to access information via real-time queries typically provide much higher ROIs than those that do not incorporate real-time or near-real-time data.

Second, the information reported by the BI tools must be “mission-critical” to users and relate directly to their spans of control and performance expectations. In many cases, identifying the key metrics to include in BI reports and dashboards proves to be the most challenging component of implementing BI. Therefore, it is essential to ensure that all team members have a voice in

determining the metrics chosen and that there is an overall consensus that the selected metrics are, in fact, the appropriate ones for each individual and the team.

A third essential characteristic of successful BI efforts is that the reporting objects should be graphical and interactive. Graphs, charts, and other visualizations are critical for two reasons. First, they summarize and present potentially massively large volumes of data in much smaller spaces than traditional tabular reports. Second, for many – perhaps most users – they are easier to interpret than row after row and column after column of numerical data. Additionally, making the reporting objects interactive is another critical driver of successful BI efforts. Interactivity allows team members to customize their Power BI tiles, reports, and dashboards to meet their needs by adjusting the date ranges reported, filtering to particular products or product lines, or focusing only on selected general ledger accounts.

The fourth key characteristic of successful BI efforts is that they should focus on specific organizational goals and provide measurements against these goals. For example, suppose the reporting metric is "manufacturing defect rate." The company's target defect rate is 0.3% or less. In that case, the BI reporting object should show the actual results and the target so all stakeholders can see whether the company meets its stated objectives. **Figure 6** presents one example of how the reporting object could make this comparison, although countless other options exist.

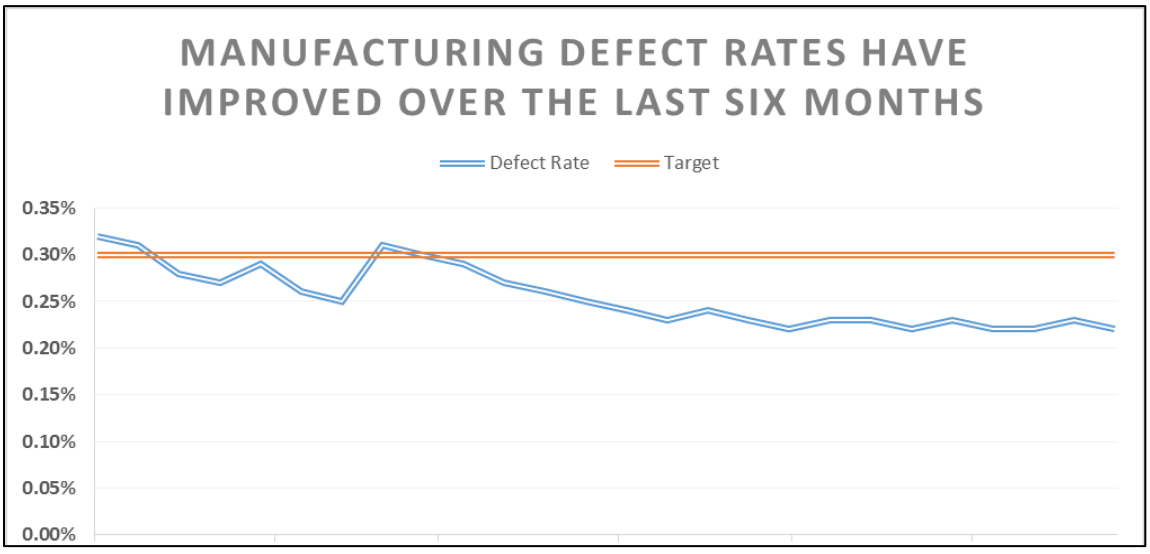


Figure 6 - Sample BI Reporting Object Comparing Target to Actual

The fifth and final characteristic of successful BI efforts is visibility. To be successful, remove all barriers and disincentives associated with accessing BI reports. Promote and encourage team members to view relevant BI reporting objects frequently, at least daily, in most cases. If your BI tool allows you to publish reports to your intranet, consider setting the URL for team members' web browsers to your intranet's BI page. This action ensures team members see the BI reporting objects when they open their web browsers.

Further, ensure mobile team members can access their BI reports while away from the office. This action helps them stay on top of all relevant measurements and metrics, even when working remotely. If information is difficult to access, you should expect that team members will not access it as often as you would like. Therefore, ensure that access is simple and easy. Sometimes, you may consider "pushing" reporting objects to team members.

?	Has your organization (or clients' organizations) engaged in substantial business intelligence efforts? If the results were less than desired, can you point to any of the factors identified thus far as partially responsible for the unsatisfactory results?

The Importance of Do It Yourself Business Intelligence

As mentioned, BI was once a tool reserved only for large organizations with large budgets and armies of in-house technology experts. One of the biggest drivers for this was the complexity of creating BI reporting objects. To illustrate, in a recent presentation on BI, Gartner used the graphic pictured in **Figure 7** to describe the traditional process for creating BI reports and dashboards. But unfortunately, the technical expertise required to execute that strategy is beyond the capabilities of many business professionals.

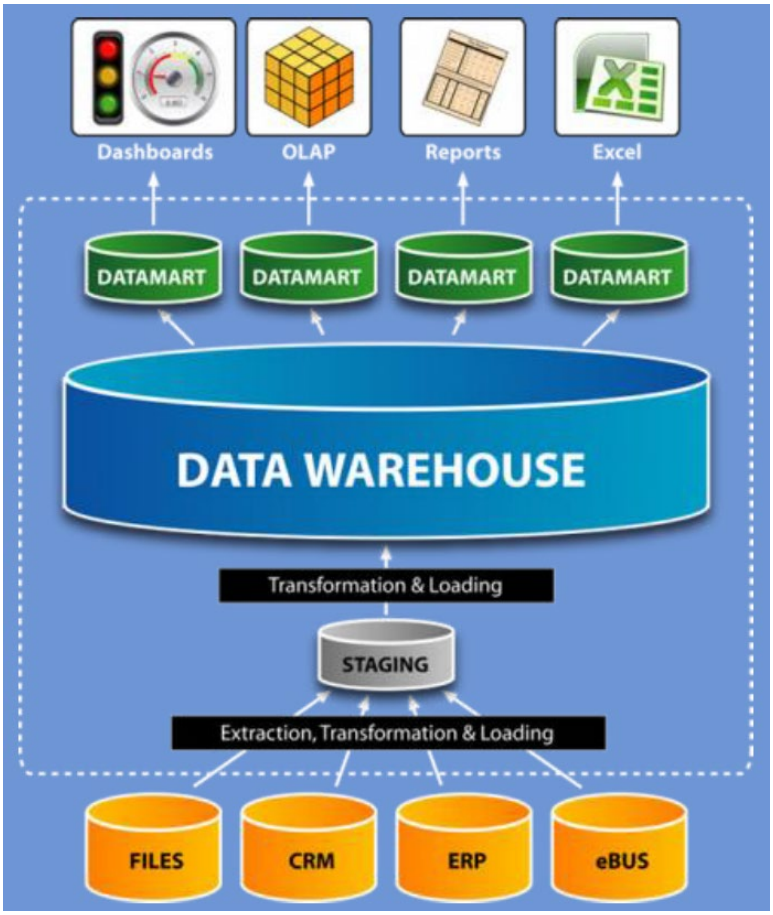


Figure 7 - Traditional IT-driven Process for Creating Reports and Dashboards

However, a new breed of tools enables "do it yourself" BI (DIYBI). The importance of this development is significant for organizations of all sizes. First, DIYBI means that BI is now available, accessible, and affordable to all organizations, regardless of size. No longer is BI a tool only for enterprise-class organizations; now, even "mom-and-pop" organizations can use BI to improve profitability. For example, an independent pharmacy might be able to use BI to reduce the quantity of inventory on hand by accounting for seasonality and, in the process, improve cash flow. Likewise, a florist considering opening a second store might use DIYBI to identify where its most profitable customers reside. Doing so helps ensure that the additional location makes it even more convenient for them to transact business with the florist, ensuring continued patronage. Without DIYBI, these activities could be challenging for smaller firms to complete successfully.

A second critical aspect of DIYBI is placing BI's power with end-users. Many of today's DIYBI tools do not require assistance from technology staffers. Instead, they facilitate end-users creating queries to transactional databases and other data sources, analyzing the queried information, and preparing and distributing graphical and interactive reports to all stakeholders. In the past, these activities were all but unthinkable for most end-users; now, they are readily available through DIYBI.

Logi Analytics, a BI technology supplier, reported several interesting statistics regarding DIYBI and the growing appetite by end-users for DIYBI tools in a whitepaper.[1]

- Over 90% of respondents to a recent survey indicated that it is essential for business users to access the information they need without IT.
- Only 22% of business users can access and use self-service BI tools when needed.
- Eighty-four percent of IT organizations plan to invest in self-service BI within 24 months.
- Twenty-four percent of businesses have already purchased self-service BI tools without their IT staff's approval, and this trend is growing.
- On average, self-service BI reduces IT requests by 37%.
- Ninety-two percent of business users report that it is "very important" or "somewhat important" to access data and information without asking IT.

As confirmed in the Logi Analytics whitepaper, the emergence of DIYBI is one of BI's undeniable trends today. Further, all signs point to continued growth in DIYBI as business professionals seek to regain control over business reporting and intelligence efforts.

Further illustrating the importance of DIYBI, Gartner published the following quote in a 2017 report (with emphasis added).

> *Visual-based data discovery, a defining feature of the modern business intelligence (BI) platform, began in around 2004 and has since transformed the market and new buying trends away from IT-centric systems of record (SOR) reporting to business-centric agile analytics.* ***Modern BI and analytics platforms are characterized by easy-to-use tools that support a full range of analytic workflow capabilities and do not require significant involvement from IT in order to predefine data models upfront as a prerequisite to analysis*** *(including at enterprise-scale deployment).*

Examples of Today's Leading Business Intelligence Tools

Though the BI platform market is quite fluid, and several niche players serve this market, many well-established and recognized companies provide BI solutions. For example, Gartner recognizes **Tableau**, **Microsoft**, and **Qlik** as leaders in the market for analytics and business intelligence platforms, as shown in **Figure 8**.

[1] "2014 State of Self-Service BI Report," www.logianalytics/resources-overview

Figure 8 - Gartner's Magic Quadrant for Analytics and Business Intelligence Platforms

The following discussions summarize some available solutions to provide a sample of what is available. However, a complete discussion of all solutions is beyond the scope of this course.

Excel as a BI Tool

Because of its massive number of users, Microsoft Office's Excel is today's leading BI tool. Though not explicitly engineered as a BI tool, many Excel users have built BI dashboards using various components of Excel's core functionality. That functionality includes Open Database Connectivity queries, PivotTables and PivotCharts, tables, sorting and filtering, the extensive function library, macros, and charting and graphing options. However, the BI dashboards and reports generated with Excel do not provide all the necessary characteristics for successful BI in many cases. For

example, sharing Excel-based BI dashboards with other team members is often frustrating. Likewise, attempting to query, summarize, and analyze large volumes of data from multiple data sources can be challenging. Consequently, although many business professionals try to use Excel as a BI tool, the results are often less than optimal.

Recognizing the desire of many Excel users to leverage their investment in Excel and their knowledge of the product, Microsoft added specific BI features to Excel beginning with the 2010 release. **Power Query**, **Power Pivot**, and **3D Maps** can help you overcome some limitations when using Excel as a BI tool.[2] For example, you can use Power Query to access and query information from traditional data sources, such as your accounting software database. Also, you can use Power Query to access data from nontraditional data sources, such as Salesforce.com and the Microsoft Azure Marketplace. Once you query the information, you can use Power Pivot to "crunch" the data, even if you are dealing with extensive data models. Further, you can use Power View and 3D Maps to create and present visualizations, including interactive dashboards that allow users to filter the dashboards on the fly. Those attempting to build BI models with Excel should investigate using these tools to improve Excel's capabilities as a BI tool.

Extending Excel with Microsoft's Power BI

In addition to the Excel tools mentioned above, Microsoft also makes available **Power BI**, a suite of tools that provides a holistic BI platform in a single platform, in contrast to the somewhat fragmented approach of using add-ins in conjunction with Excel for BI efforts. With Power BI, you can work in an environment like that found in Excel to create your BI reports and dashboards and then publish them virtually anywhere on any device. Power BI's other advantages include exploring your BI dashboards using natural language queries and mobile apps to access your BI dashboards. Additionally, you can publish and share your reports through PowerBI.com if you have the appropriate subscription.

[2] Note that depending upon which version of Excel you are running, you may not have access to all these features in Excel. Additionally, these features carry different names, depending upon which version of Excel you are running.

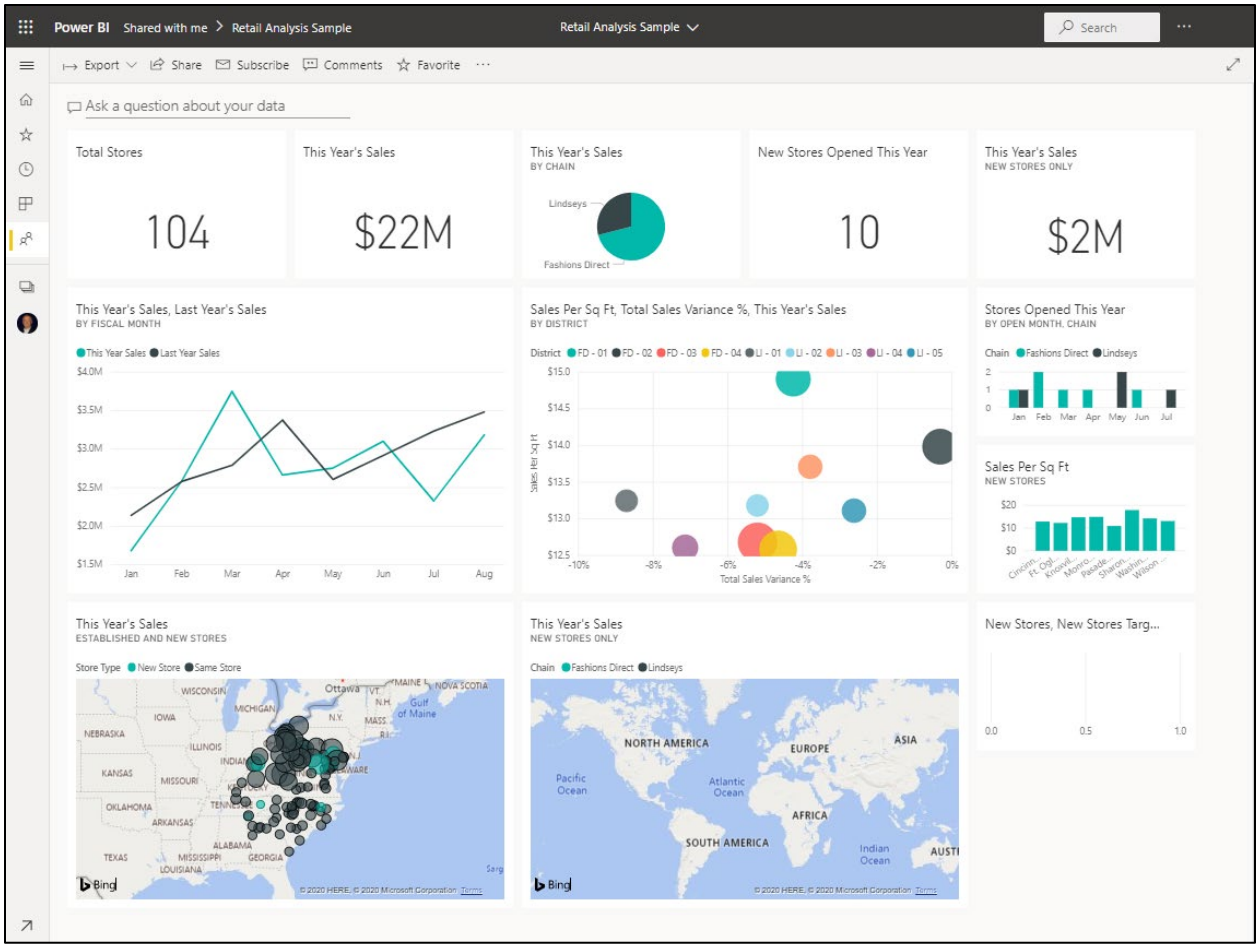


Figure 9 - Sample Dashboard Created Using Power BI

Microsoft prices Power BI very aggressively. A single-user desktop edition is available at no charge. However, many business professionals will likely need Power BI Pro's functionality, priced at $9.99 monthly. Additionally, larger organizations should investigate Power BI Server as their BI solution.

In sum, Power BI provides a "true" BI tool and can help you realize the desired results for your BI initiative. Individuals and organizations seeking to capitalize on their existing investment in Excel and their knowledge of the ubiquitous spreadsheet tool should consider Power BI when planning a BI initiative. Likewise, Power BI is a compelling option for those attempting to implement BI without making substantial monetary commitments to a specific platform.

Tableau, A Powerful BI Option

Another well-respected provider of tools for generating BI is Tableau. For three consecutive years, Tableau has been listed in the "Magic Quadrant" of Gartner's annual report on Analytics and Business Intelligence for several successive years, signifying the company as one of the leaders in this market.

Tableau offers several products to help professionals in organizations of all sizes generate and distribute BI reports and dashboards. The main products offered by Tableau include the following.

- **Creator**. With Creator, you can connect to virtually any type of data and generate powerful graphics and analytics. Creator costs $70 per user per month, regardless of whether your company hosts Tableau or Tableau hosts it.

- **Explorer**. You can connect to published data sources and create visualizations with other users using Explorer. Explorer is priced at $42 per user per month if hosted by Tableau and $35 per user per month if self-hosted.

- **Viewer**. As its name implies, Viewer allows you to view dashboards and reports created by others, but users of this tool cannot create visualizations. When self-hosted, Viewer prices at $12 per user per month; when hosted by Tableau, the price increases to $15 per user per month.

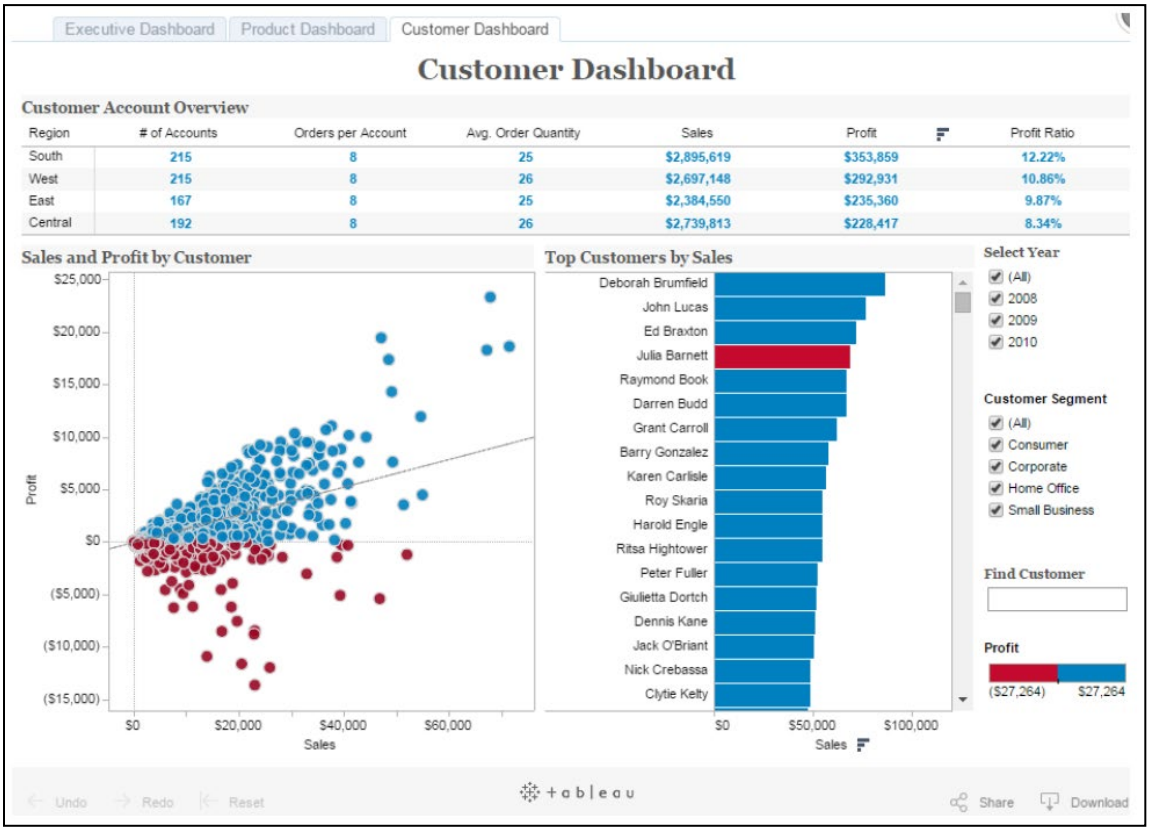


Figure 10 - Sample BI Dashboard Generated Using Tableau

Tableau is probably best suited for organizations with more complex BI needs, including advanced visualization requirements. Additionally, Tableau is an attractive option for those who

want their IT staff to maintain greater control over BI deployments. However, suppose you are considering implementing Tableau. You should carefully plan and budget for the deployment as you might experience moderate upfront licensing and implementation costs and annual maintenance expenses.

Qlik Sense

Qlik is another "leader" company in Gartner's "Magic Quadrant." Qlik provides multiple tools you can use in BI efforts. Qlik Sense is a governed, server-based solution that facilitates BI at a group, departmental, or organizational level. Through centralized data security and governance policies, Qlik Sense helps organizations realize the benefits of DIYBI without losing management control, governance, security, or scalability. In addition, Qlik Sense facilitates connections to many common databases using standard technologies such as Open Database Connectivity (ODBC) and custom connectors to other data sources. **Figure 11** presents a sample BI dashboard built using tools from Qlik.

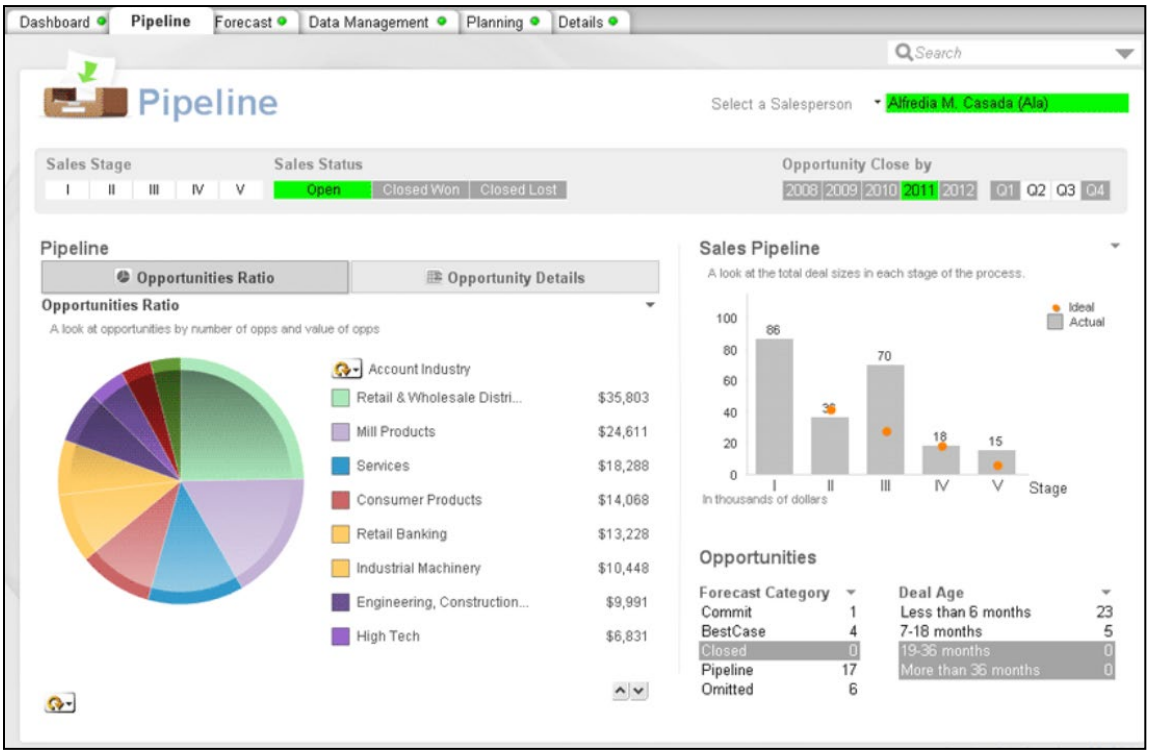


Figure 11 - Sample BI Dashboard Built Tools Available from Qlik

Pricing for Qlik products and services depends on several variables. For complete pricing details, visit www.qlik.com or call Qlik at (866) 616-4960.

?	Which type of business intelligence solution are you more likely to deploy? A dedicated solution? An Excel-based approach? A hybrid approach?

Fundamental Business Intelligence With Excel

Chapter 2

For some business professionals and the organizations they serve, BI needs are relatively simple, and investing in a BI solution is not likely to yield a positive ROI. Therefore, using Excel to address specific BI needs in cases like these is probably wise. Although Excel is not the ideal BI for many situations, in some cases, it does provide all the fundamental functionality necessary to prepare BI reports – data queries, summarization, and presentation. In this chapter, you will learn about the essential features available in Excel that you can use to facilitate relatively simple BI efforts.

Learning Objectives

Upon completing this chapter, you should be able to:

- List the primary elements of functionality needed to utilize Excel as a BI tool;
- Create data queries to link data from external data sources into Excel;
- Summarize transactional data using Excel-based PivotTables; and
- Utilize PivotCharts as a means of presenting BI information.

What Functionality is Necessary?

Excel's functionality is well-suited to facilitating simple BI dashboards and reports. To illustrate, consider some of the critical activities in preparing dashboard reports. These include:

- Retrieving data from external data sources,
- Summarizing data using both simple and complex calculations,
- Sorting and filtering data, and
- Presenting visual images of numerical data.

For many years, Excel has provided functionality in these four core areas, and newer versions of Excel further elevate the application's capabilities. As such, Excel lends itself quite readily to simple BI reporting. Depicted graphically, **Figure 12** illustrates the general workflow for creating an Excel-based BI report.

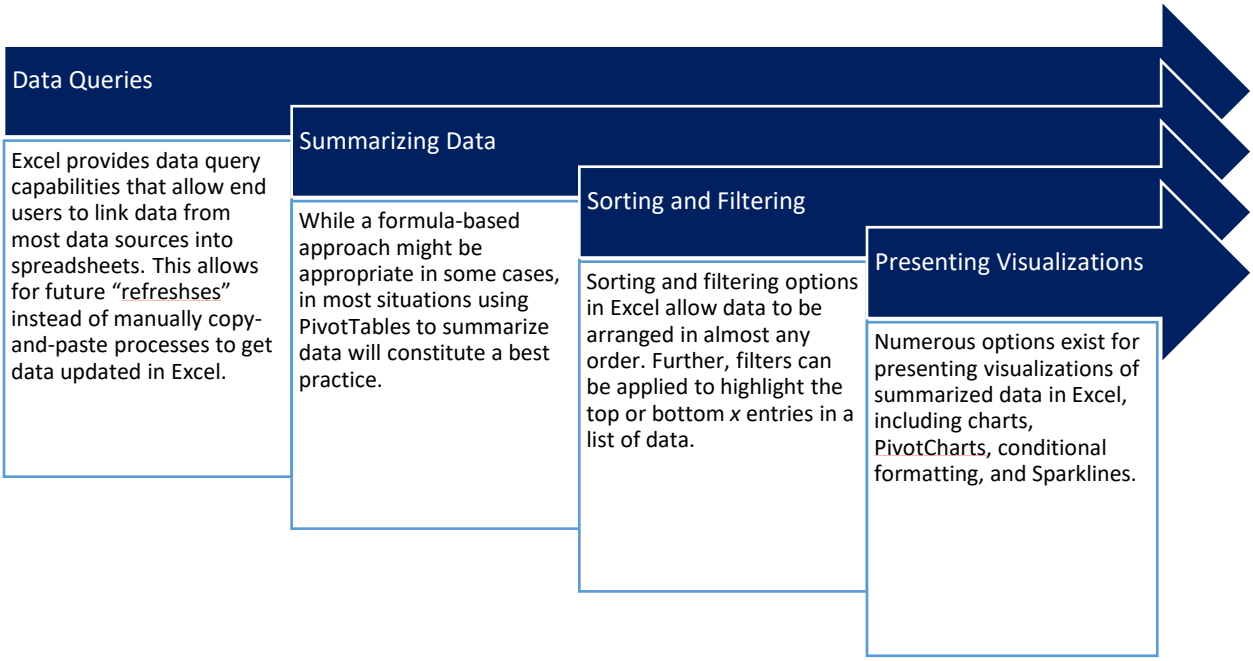


Figure 12 - Fundamental Steps Required to Create an Excel-based BI Report or Dashboard

Following the workflow outlined above, a report designer could create the BI dashboard depicted in **Figure 13**.

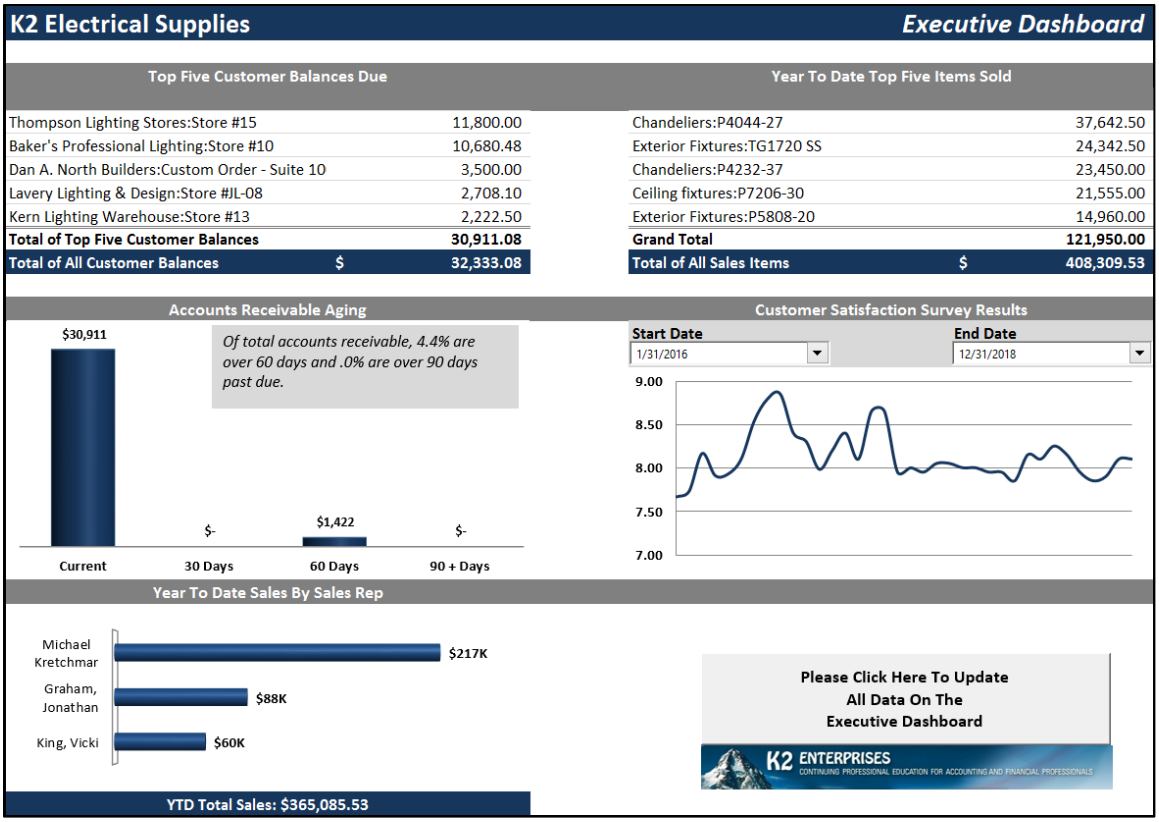


Figure 13 - BI Dashboard Created in Excel

Querying Data in Excel

Excel now provides all the functionality to query data into Excel through the **Power Query** feature. You will find Power Query's tools in the **Get & Transform** group on the Ribbon's **Data** tab, as shown in **Figure 14**.[3]

[3] Power Query became fully incorporated into all versions of Excel beginning with the 2016 release of the application. If you are running an older version of Excel, you may be able to download and install Power Query as an add-in. To do so, visit https://www.microsoft.com/en-us/download/details.aspx?id=39379.

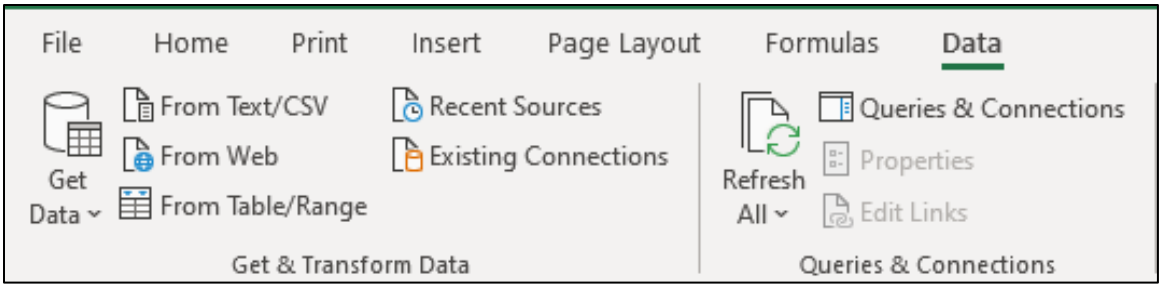


Figure 14 – Excel's Power Query Tools on the Data Tab of the Ribbon

Using Power Query, you can connect to diverse external data sources using one of over forty prebuilt tools available in the utility. Below are some familiar data sources you can connect to using Power Query.

- Excel workbooks
- Access databases
- PDF documents
- Most accounting applications, including Cloud-based solutions
- SQL Server databases
- CSV and TXT files
- Oracle databases
- Sybase databases

Additionally, Power Query can extract data directly from data sources that comply with the **Open Database Connectivity** (**ODBC**) standard. ODBC is a standard database access method developed by the SQL Access group in 1992. With ODBC, it is possible to access data stored in any ODBC-compliant database from within any ODBC-compliant application. ODBC manages this by inserting a middle layer known as an *ODBC driver* between the application and the **Database Management System** (**DBMS**) that translates the application's data queries into commands that the DBMS understands. ODBC can provide two-way integration between Excel and an external data source – such as a general ledger – although read-only ODBC connections are the norm. **Figure 15** presents a graphical representation of the data access process.

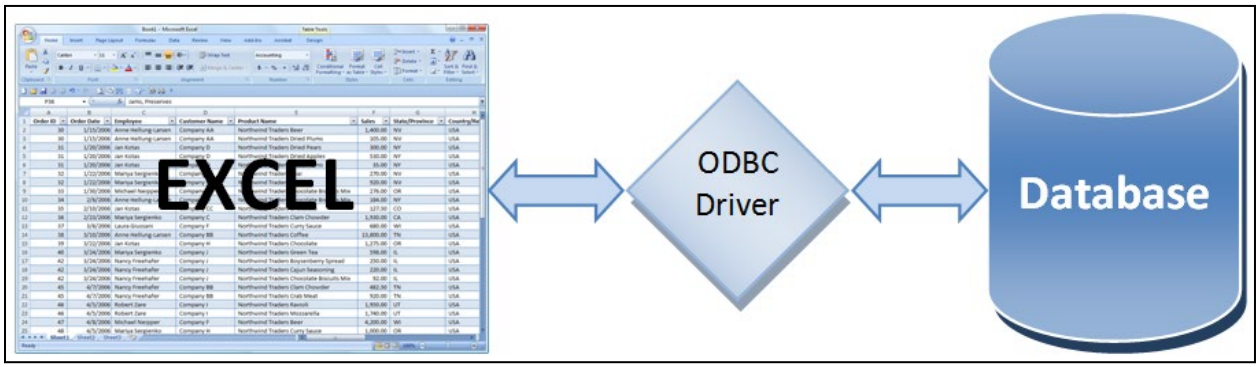


Figure 15 - Accessing External Data Directly from within Excel Using ODBC

In our first example, Carl, the owner of Carl's Computer Shop, wants to extract some data from the company's accounting system – **QuickBooks Enterprise Solutions**[4] – to create a dashboard to help him analyze sales data. First, Carl opens the QuickBooks company file to extract a table of all invoice line items. Then, from within Excel, on the **Data** tab of the Ribbon, click **Get Data**, **From Other Sources, From ODBC**, and choose the name of your data source (in this case, **QuickBooks**). Next, select the table(s) you want to include in your report – in this case, the **InvoiceLineItems** table – and click **Load** to add the data to the workbook.

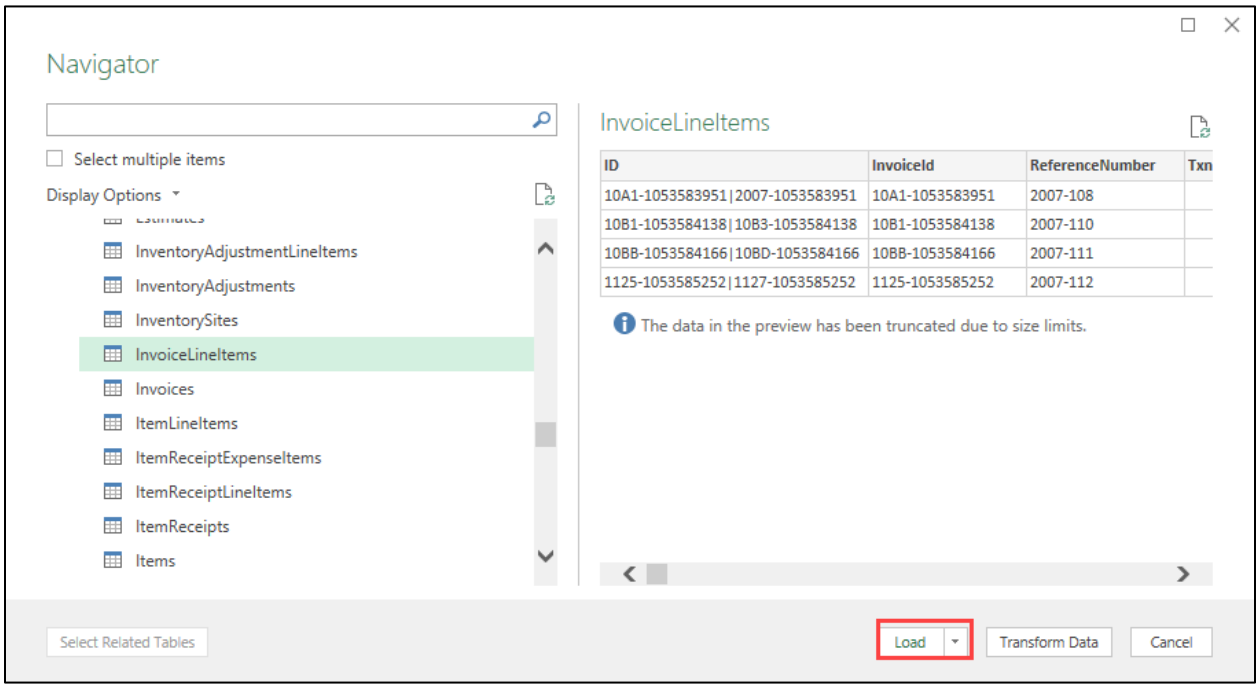


Figure 16 - Using the Data Connection Wizard to Query QuickBooks Enterprise Solutions

The imported table dynamically links to the underlying data source and can be refreshed as required. Once Excel imports the data via the ODBC query, Carl can manipulate the data using some of the techniques described later in this course to produce a dashboard component such as the one presented in **Figure 17**.

[4] If you are running a QuickBooks application – including QuickBooks Online – you must install an ODBC driver before you can make an external connection to the application's underlying database. Intuit provides such a driver with QuickBooks Enterprise Solutions. Visit the QuickBooks Enterprise Solutions Help system to learn how to install and use this driver. Another option for QuickBooks ODBC drivers is CDATA (www.cdata.com). Many users report substantially better performance with the driver provided by CDATA.

	Total Quantity Sold	Total Dollars Sold
Accessories		$ 8,046.00
Cable Installation	72	$ 8,280.00
CDRW Drive	16	$ 1,392.00
Computer-Midrange	34	$ 40,630.00
Computer-Poweruser	9	$ 26,055.00
Consulting & Training	152	$ 18,240.00
CPU-4.0ghz	10	$ 1,890.00
Diagnostic Service	5	$ 475.00
RAM-1GB	10	$ 790.00
Repair Service	20	$ 1,300.00
Software		$ 9,869.00
Grand Total	**328**	**$ 116,967.00**

Figure 17 - Sample Dashboard Component Created from ODBC-based Data

Creating a Multi-table Query

Because modern accounting applications use *relational* databases, it is unlikely that all data required for a report or analysis will reside in a single table, hence, the need to query and extract data from multiple related tables. In our following example, we will use Power Query to relate three tables before importing Microsoft Access data.

Click **Get Data**, **From Database**, **From Microsoft Access Database** from the Ribbon's **Data** tab to begin the process. In the **Choose Data Source** dialog box, select **Xtreme Accounting** and click **OK** to open the **Query Wizard – Choose Columns** pane. In the **Available tables and columns** list on the left, select the following tables:

- **Customer** table,
- **Orders** table, and
- **Orders Detail** table.

Figure 18 illustrates this process.

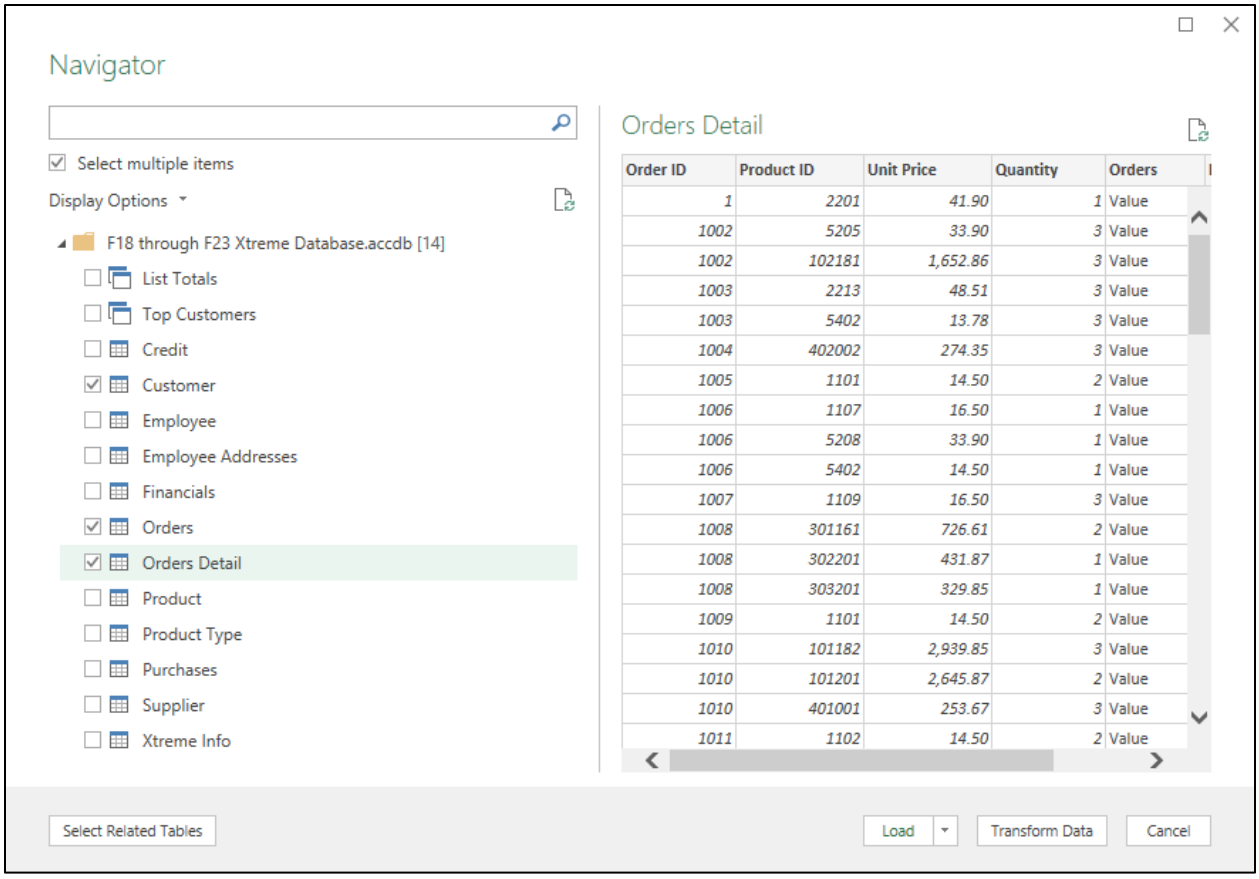


Figure 18 - Selecting Fields to Import in the Microsoft Query Wizard

Click **Load To** near the bottom of the window. Next, choose the options labeled **Only Create Connection** and **Add this data to the data model**, as shown in **Figure 19**.

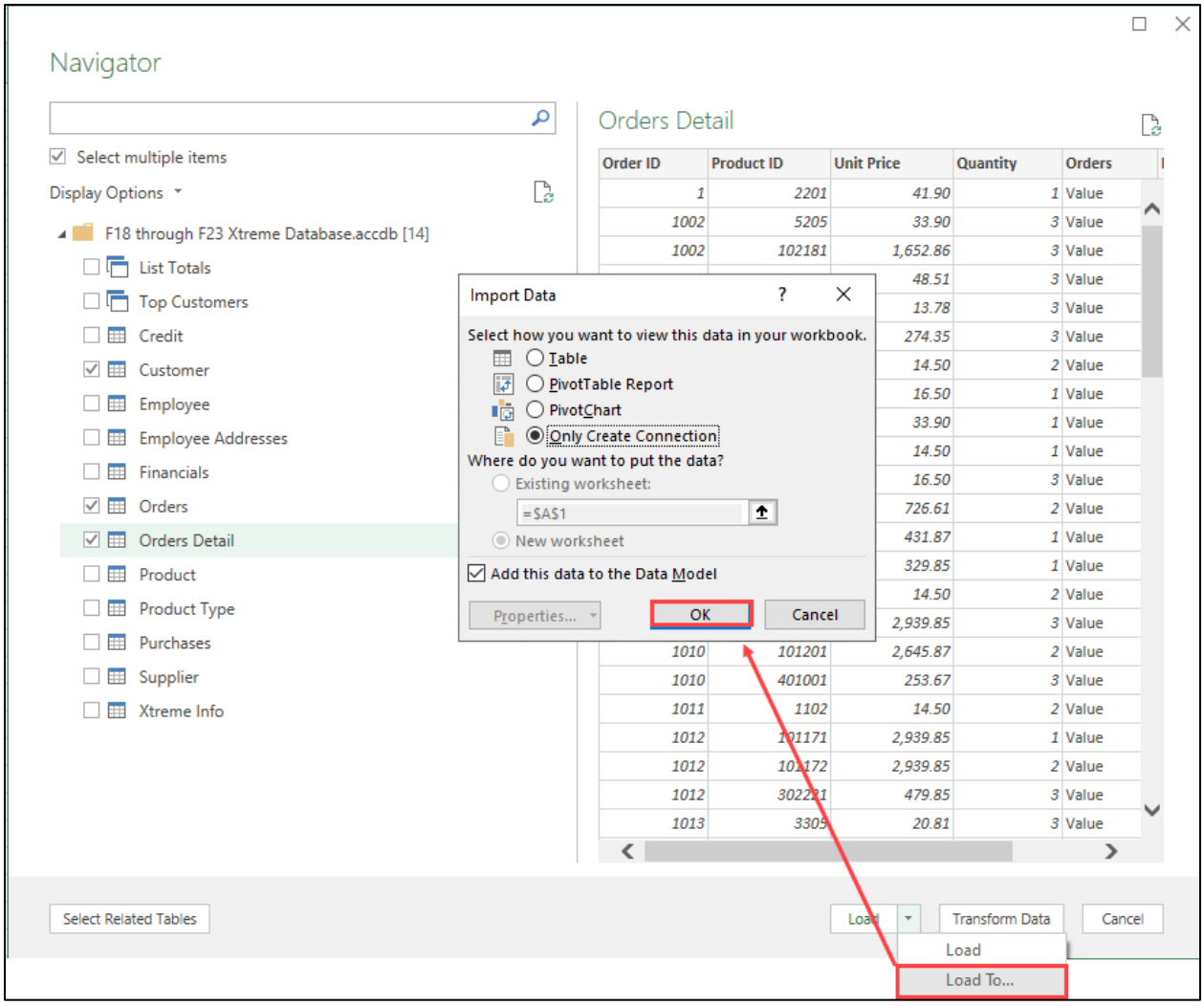


Figure 19 - Excel's Import Data Dialog Box

Among other benefits, an Excel data model effectively allows you to treat multiple data sources in Excel as a single source. In this example, Excel noticed common columns (fields) of data between the queried tables and automatically established the relationships between them. You can see these relationships by clicking **Relationships** on the **Data** tab of the Ribbon.

However, a close inspection of the data model indicates that one required field is missing – one that multiplies the *Unit Price* by the *Quantity*. One of the advantages of working with Power Query is that you can add *transformations* to your data. Transformations are simply modifications that make the data more useful. Examples of transformations include adding/deleting/splitting columns of data, applying filters to your data, pivoting/unpivoting data, and adding user-defined calculations. In this example, we will add a user-defined calculation that multiplies the Unit Price by the Quantity.

Right-click on the **Orders Detail** query in the **Queries & Connections** task pane and choose **Edit** to open the **Power Query Editor**. In the Power Query Editor, click the **Add Column** tab of the

Ribbon, followed by **Custom Column**. Next, in the **Custom Column dialog box**, add the formula shown in **Figure 20** and click **OK**. Doing so will add a new column to the query, and you can use that new column of data as if it were "native" to the Orders Detail table.

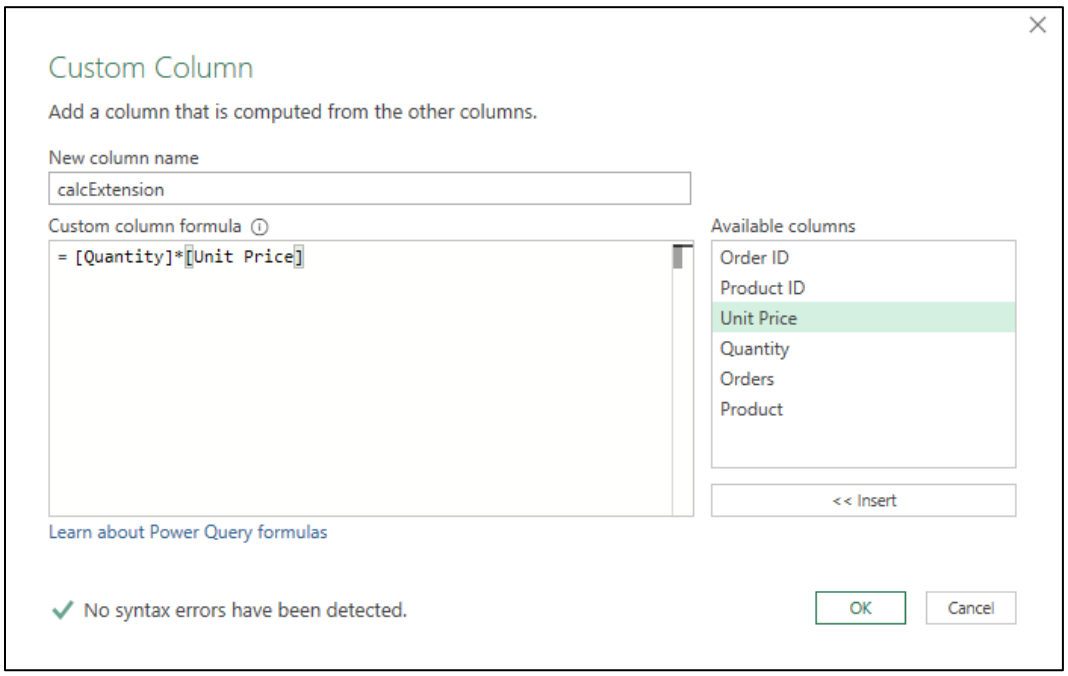


Figure 20 - Table with Calculated Field Added

As shown in **Figure 21,** the newly-created custom column of data is now available in the query's Orders Detail table. It is also important to note that Power Query created an **Applied Step** as part of the query. Applied Steps are procedures that automatically execute each time the query refreshes. Therefore, you no longer need to worry about whether your user-defined calculations are up to date with Applied Steps in place.

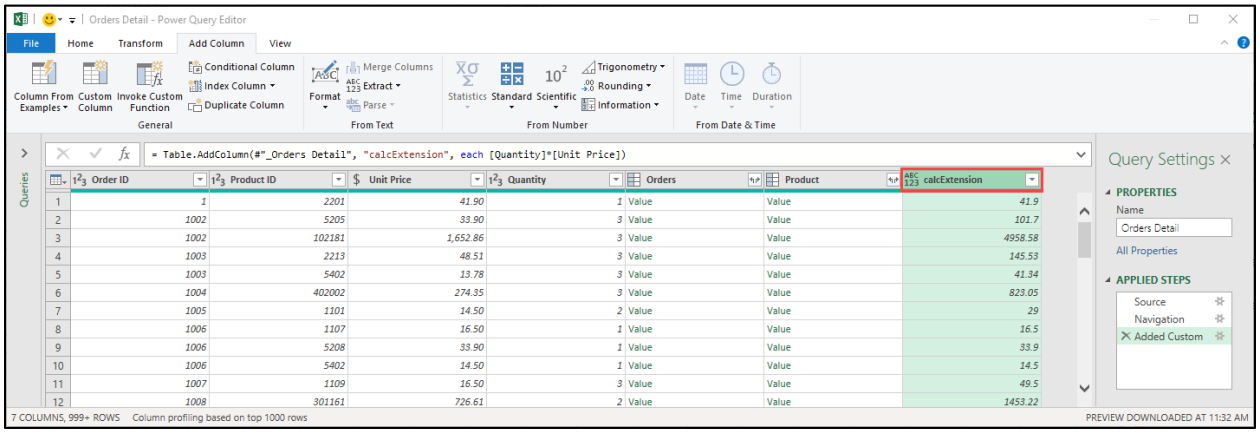


Figure 21 - User-defined Field in Microsoft Query

From the Ribbon's **Data** tab, click **Relationships**, and you can see all the relationships established between the three tables – *Customers*, *Orders*, and *Orders Detail*. Further, as shown in **Figure 22**, you can edit these relationships and add new ones if necessary.

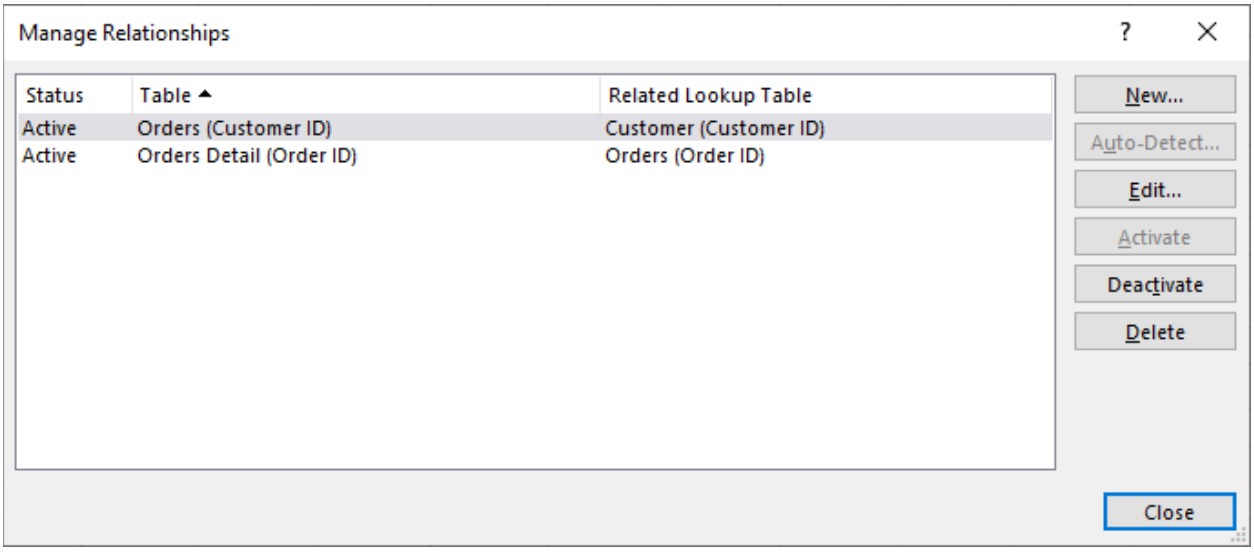
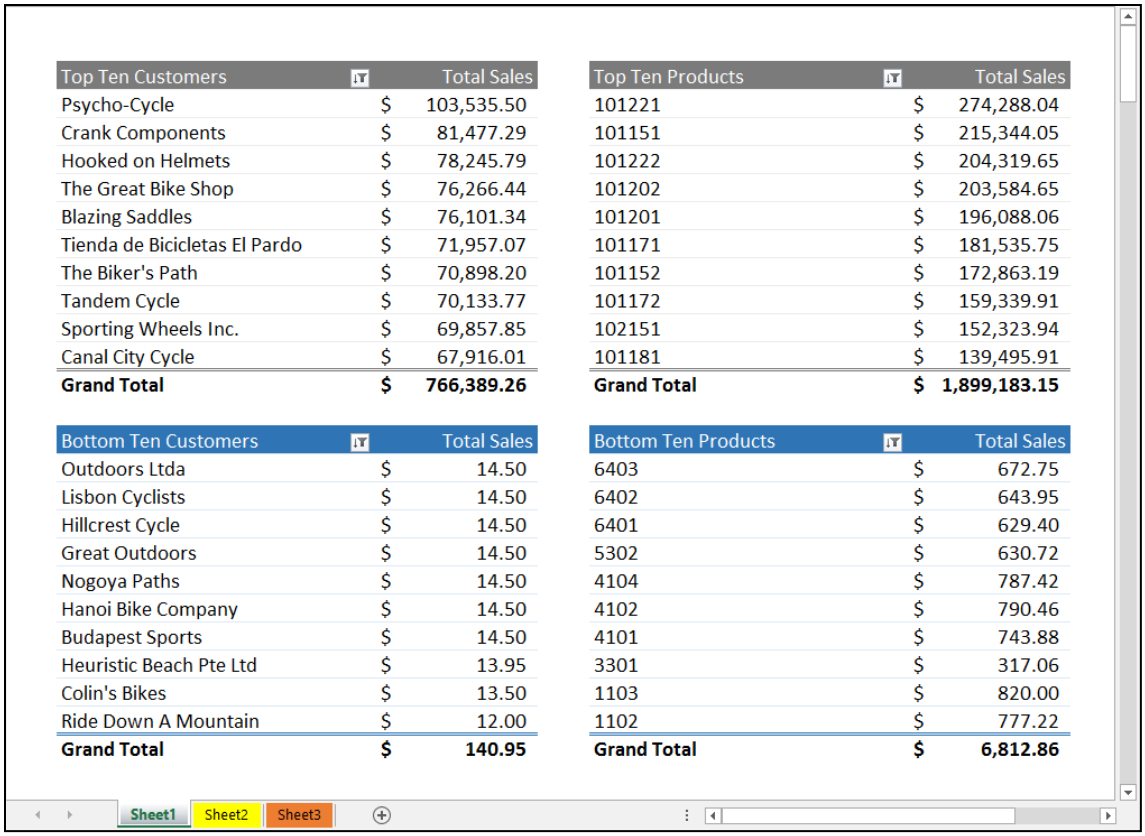


Figure 22 - Viewing a Related Table Query in Microsoft Query

Finally, click **PivotTable** from the Insert tab of the Ribbon to summarize the data into dashboard-style components similar to those shown in **Figure 23**. Next, choose the option labeled **Use this workbook's Data Model**, followed by **OK**, and create the necessary PivotTables and PivotCharts to summarize and present the data appropriately.



Top Ten Customers	Total Sales
Psycho-Cycle	$ 103,535.50
Crank Components	$ 81,477.29
Hooked on Helmets	$ 78,245.79
The Great Bike Shop	$ 76,266.44
Blazing Saddles	$ 76,101.34
Tienda de Bicicletas El Pardo	$ 71,957.07
The Biker's Path	$ 70,898.20
Tandem Cycle	$ 70,133.77
Sporting Wheels Inc.	$ 69,857.85
Canal City Cycle	$ 67,916.01
Grand Total	**$ 766,389.26**

Top Ten Products	Total Sales
101221	$ 274,288.04
101151	$ 215,344.05
101222	$ 204,319.65
101202	$ 203,584.65
101201	$ 196,088.06
101171	$ 181,535.75
101152	$ 172,863.19
101172	$ 159,339.91
102151	$ 152,323.94
101181	$ 139,495.91
Grand Total	**$ 1,899,183.15**

Bottom Ten Customers	Total Sales
Outdoors Ltda	$ 14.50
Lisbon Cyclists	$ 14.50
Hillcrest Cycle	$ 14.50
Great Outdoors	$ 14.50
Nogoya Paths	$ 14.50
Hanoi Bike Company	$ 14.50
Budapest Sports	$ 14.50
Heuristic Beach Pte Ltd	$ 13.95
Colin's Bikes	$ 13.50
Ride Down A Mountain	$ 12.00
Grand Total	**$ 140.95**

Bottom Ten Products	Total Sales
6403	$ 672.75
6402	$ 643.95
6401	$ 629.40
5302	$ 630.72
4104	$ 787.42
4102	$ 790.46
4101	$ 743.88
3301	$ 317.06
1103	$ 820.00
1102	$ 777.22
Grand Total	**$ 6,812.86**

Sheet1 Sheet2 Sheet3

Figure 23 - Dashboard Components from Multi-table ODBC Query

You can rename, modify, or delete Excel data connections in the **Workbook Connections** dialog box accessible by clicking **Queries & Connections** on the Ribbon's **Data** tab, as shown in **Figure 24**.

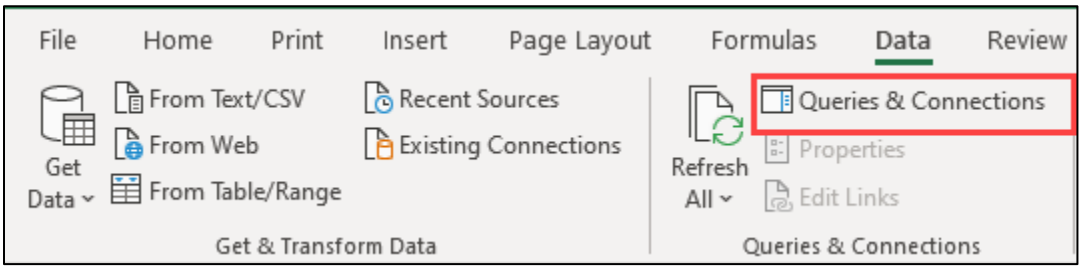


Figure 24 - Clicking Connections on the Data Tab to Modify Workbook Connections

Clicking **Queries & Connections** opens the **Queries & Connections** pane shown in **Figure 25**.

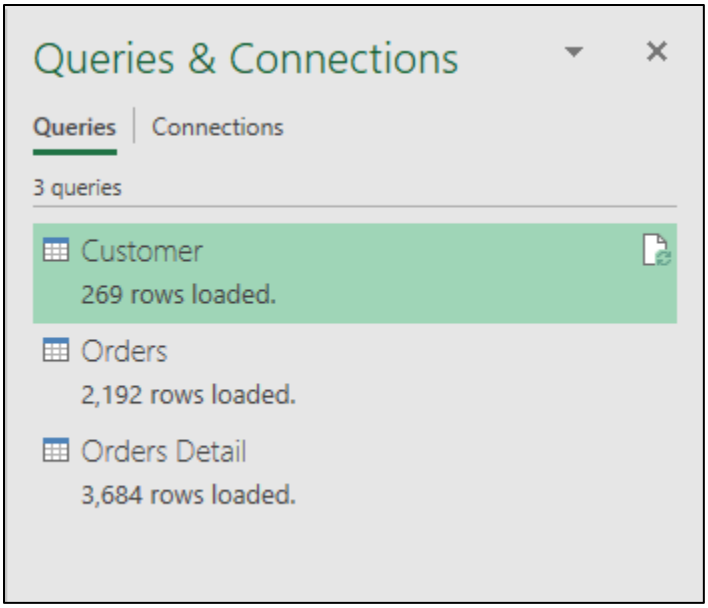


Figure 25 - Workbook Connections Dialog Box

In the Queries & Connections pane, right-click on any query or connection and choose **Properties** to open the **Connection Properties** dialog box shown in **Figure 26**. You can control attributes relative to formatting, updating, and drill through from the **Usage** tab of the Connection Properties dialog box. On the **Definition** tab of the Connection Properties dialog box, you can modify the connections' definition, including the connection file's name and the commands executed as part of the connection. Finally, on the **Used In** tab of the Connection Properties dialog box, you can see all locations in the current workbook that use data returned from that query or connection.

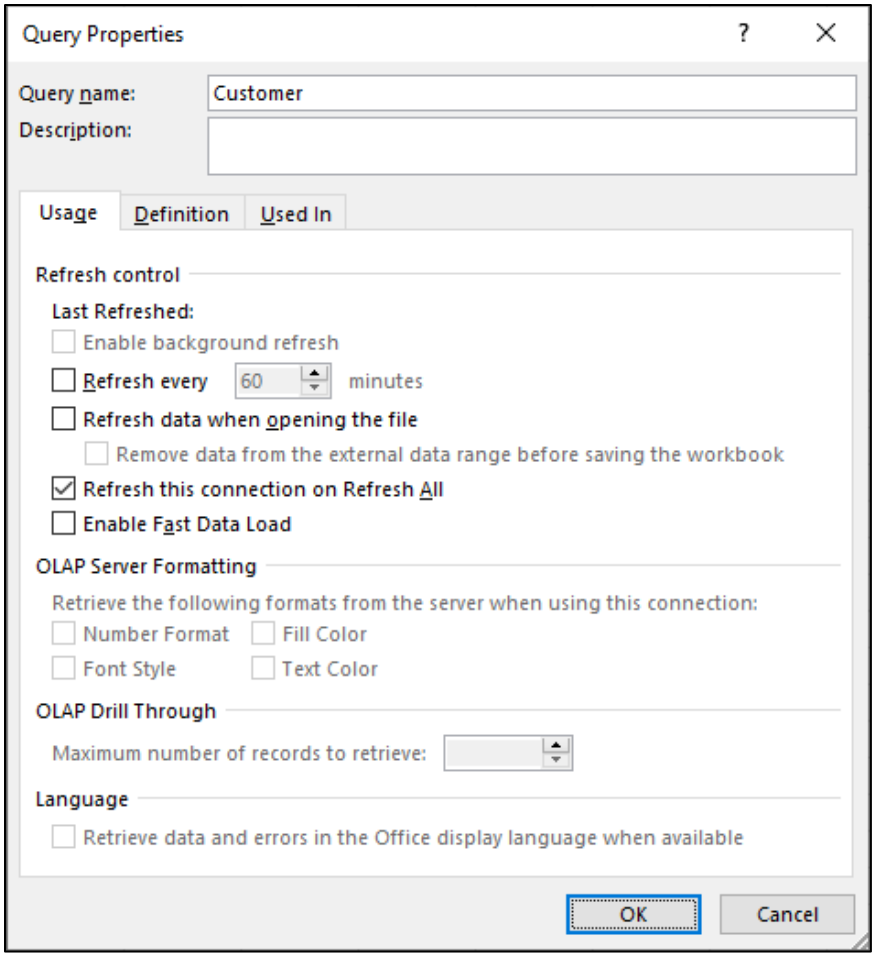


Figure 26 - Connection Properties Dialog Box

?	Why is querying data – instead of exporting or copying-and-pasting – a better approach to accessing data used in BI reports and dashboards?

Summarizing Data with Excel-based PivotTables

Most advanced Excel users agree that **PivotTables** and accompanying **PivotCharts** are among Excel's most powerful functions. Although it is beyond this course's scope to detail advanced PivotTable and PivotChart functionality, simple PivotTables and PivotCharts can be valuable in dashboard settings. In addition to being very easy to create, PivotTables and PivotCharts offer significant benefits when used in BI initiatives.

- The underlying data supporting the PivotTable or PivotChart can reside almost anywhere: Excel, Access, SQL Server, or any other ODBC-compliant database.

- Dashboard designers can sort and filter items in a PivotTable to specific criteria.

- You can format PivotTables and PivotCharts to meet almost any need, including applying conditional formatting to highlight items using specific criteria.

- PivotTables can summarize data using eleven predefined calculations, including sum, count, average, and standard deviation. Also, PivotTables can summarize data based on user-defined calculations.

Fundamentals of PivotTables

To create a PivotTable, first, ensure the underlying data resides in columns with column headers at each column's top. Though not required, we recommend you delete any blank rows and replace empty cells in a numeric column with zeroes. Note that sorting the data before creating the PivotTable is *not* necessary. The data should resemble that shown in **Figure 27**.

	A	B	C	D	E
1	Region	Product Line	Model	Date	Sales Units
2	Northeast	Drives	5400-150-25	1/31/2020	32,090
3	Southeast	Drives	5400-150-25	1/31/2020	6,786
4	Midwest	Drives	5400-150-25	1/31/2020	5,836
5	West	Drives	5400-150-25	1/31/2020	5,719
6	Northeast	Drives	5400-150-35	1/31/2020	11,917
7	Southeast	Drives	5400-150-35	1/31/2020	25,200
8	Midwest	Drives	5400-150-35	1/31/2020	21,672
9	West	Drives	5400-150-35	1/31/2020	21,239
10	Northeast	Drives	5400-250-25	1/31/2020	5,902
11	Southeast	Drives	5400-250-25	1/31/2020	12,480

Figure 27 - Sample PivotTable Data

Once the data is in proper order, click anywhere inside the data range and choose **PivotTable** from the Ribbon's **Insert** tab to open the dialog box shown in **Figure 28**. Accept the defaults and click **OK** to create the basic outline of the PivotTable.

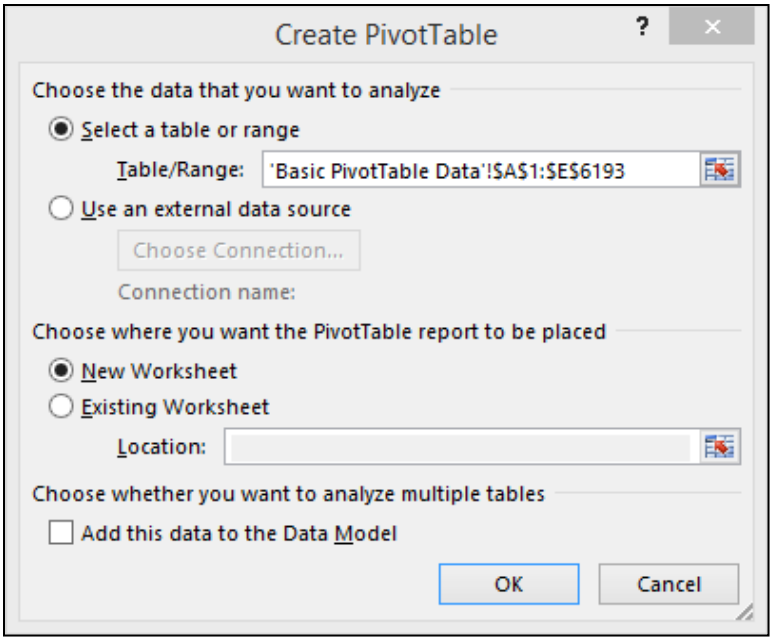


Figure 28 - Create PivotTable Dialog Box

Upon clicking **OK** in the **Create PivotTable** dialog box shown in Figure 28, Excel displays the **PivotTable Field List** and an outline of the PivotTable as shown in **Figure 29.**

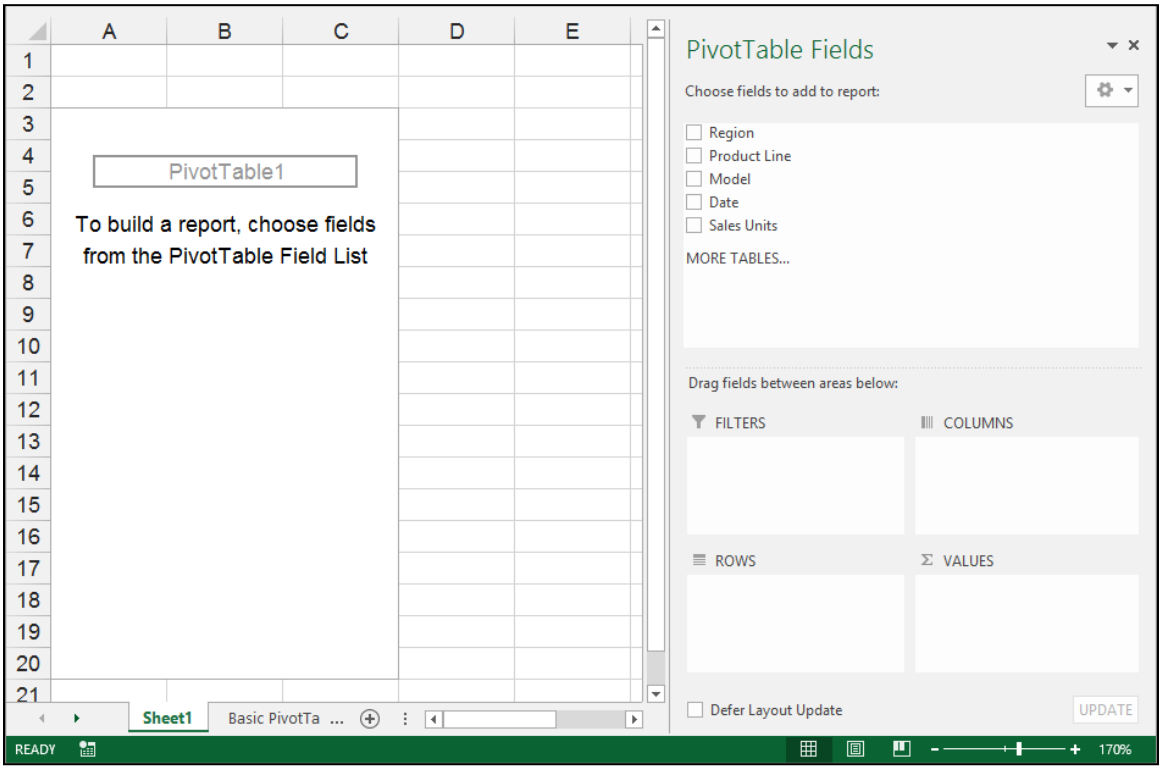


Figure 29 - Creating a PivotTable

In the PivotTable Field List, click and drag the items to display in the PivotTable into the appropriate quadrants in the bottom half of the PivotTable Field List to finish constructing the basic PivotTable. For example, to build the PivotTable with Dates as columns, Models as rows, Sales Units summed at the intersection of each unique combination of Date and Model, and Region and Product as top-level filters, drag each of those fields to the quadrants shown in **Figure 30**.

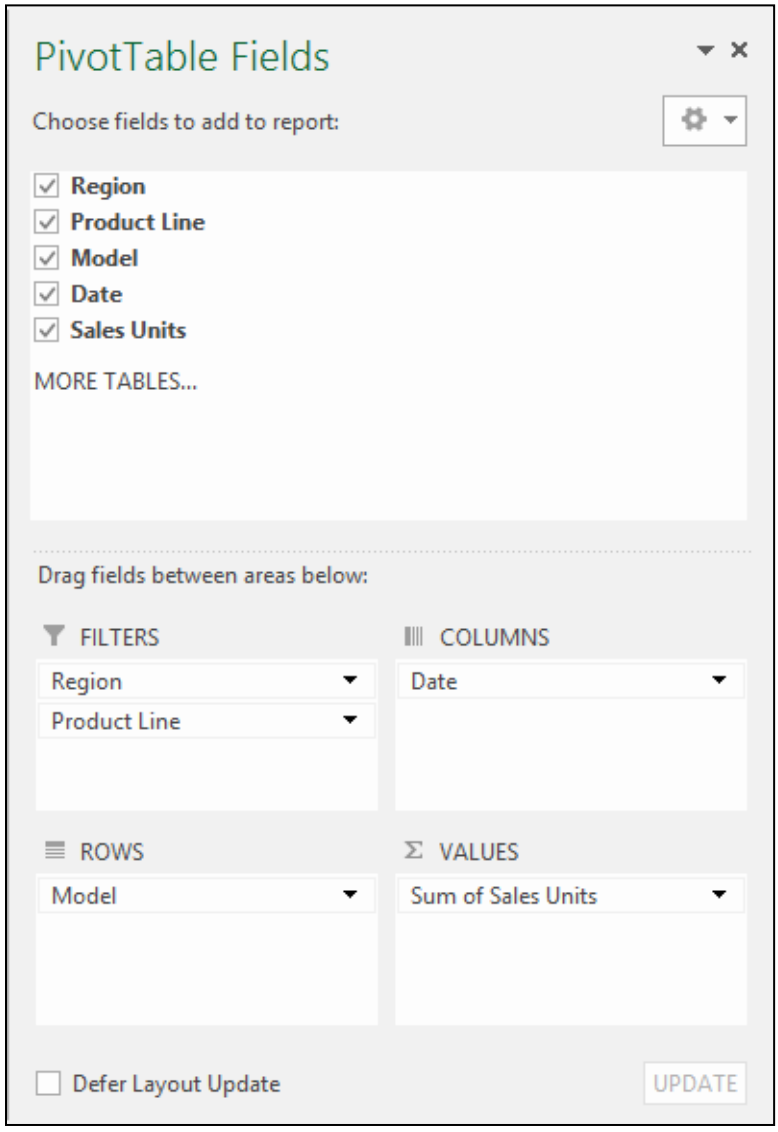


Figure 30 – Arranging Fields in the PivotTable Field List

Grouping Data in a PivotTable

PivotTable's **Grouping** features allow you to extend the process of analyzing data in a PivotTable. Grouping can occur automatically – such as with dates – or you can create custom groups to facilitate specific needs. In the example shown in **Figure 31**, automatic grouping organizes the

date data by months, quarters, and years. In this example, right-clicking on one of the dates launched the automatic grouping feature.

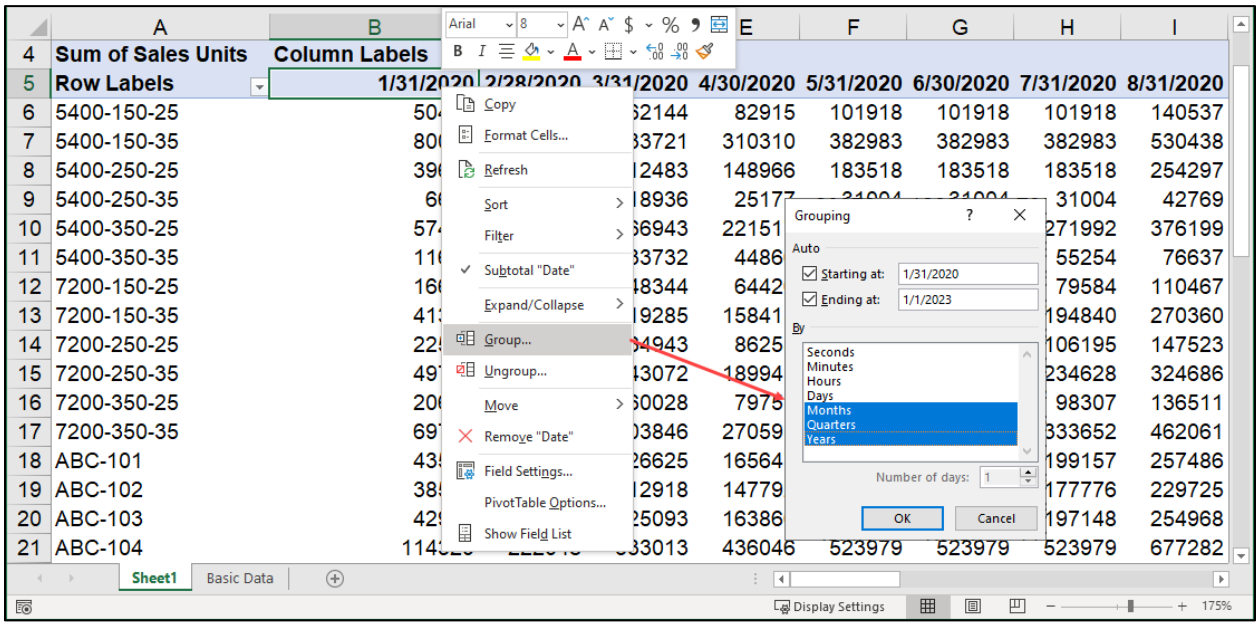
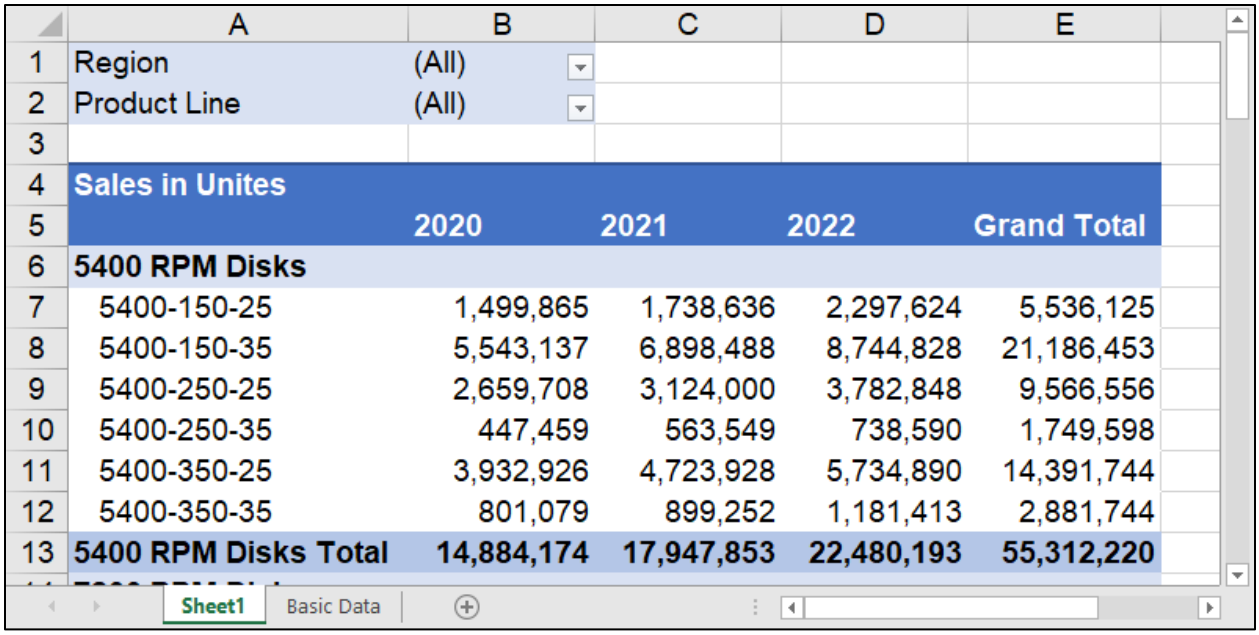


Figure 31 - Using Automatic Grouping in a PivotTable

Custom grouping allows you to group fields in any order desired. For instance, selecting all the models that begin with "5400" and then right-clicking and choosing **Group** from the pop-up menu allows a user to create a custom grouping of all the "5400 RPM Hard Disks," as shown in **Figure 32** (with formatting applied). Custom grouping is also helpful when analyzing date-oriented data in fiscal years.



	A	B	C	D	E
1	Region	(All)			
2	Product Line	(All)			
3					
4	Sales in Unites				
5		2020	2021	2022	Grand Total
6	5400 RPM Disks				
7	5400-150-25	1,499,865	1,738,636	2,297,624	5,536,125
8	5400-150-35	5,543,137	6,898,488	8,744,828	21,186,453
9	5400-250-25	2,659,708	3,124,000	3,782,848	9,566,556
10	5400-250-35	447,459	563,549	738,590	1,749,598
11	5400-350-25	3,932,926	4,723,928	5,734,890	14,391,744
12	5400-350-35	801,079	899,252	1,181,413	2,881,744
13	5400 RPM Disks Total	14,884,174	17,947,853	22,480,193	55,312,220

Sheet1 Basic Data

Figure 32 - Custom Grouping of Data in a PivotTable

Sorting and Filtering PivotTable Data

In addition to grouping, sorting and filtering are two functions that enhance a PivotTable's analytical capabilities. Though available in prior versions of Excel, Excel 2007 significantly improved sorting and filtering capabilities.

Sorting PivotTable Results

Suppose you construct a basic PivotTable and desire to sort the PivotTable's data in a specific order. You can accomplish this easily by clicking the drop-down arrow next to the row field name and choosing one of the three sort options from the pop-up menu.

1) **Sort A to Z**
2) **Sort Z to A**
3) **More Sort Options…**

The first two options are relatively straightforward – they sort the PivotTable in ascending or descending order based on the values in the row field selected. However, upon choosing **More Sort Options…,** Excel presents additional choices – the basis for the sort. For instance, **Figure 85** shows you can select whether the PivotTable will sort by the model or projected sales.

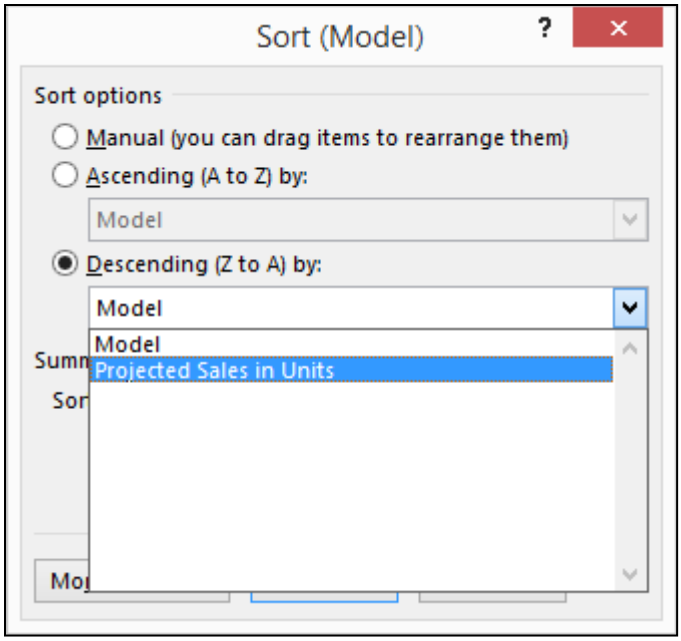


Figure 33 - More Sort Options in a PivotTable

Filtering Results

In addition to sorting, filters are helpful tools for analyzing data in a PivotTable. By applying filters, you can pare down large volumes of records into smaller subsets that provide meaningful and actionable information. In addition to checking boxes manually next to the items desired for display, PivotTables provide three other filtering options.

1. **Label filters**
2. **Value filters**
3. **Date filters**

Label Filters

Suppose you want to see all the items in a PivotTable that started with the characters "7200." Rather than sorting the PivotTable, hiding rows, or manually filtering the PivotTable, applying **Label Filters** is probably the easiest way of accomplishing that objective. To use a Label Filter, click the drop-down arrow next to the row field name that contains the data to filter and choose **Label Filters** from the pop-up menu. Then, in the **Label Filter** dialog box, select the "Begins With" option and enter "7200" as the criteria, as shown in **Figure 34**. Click **OK**, and the PivotTable filters to just those items that begin with "7200." Note that you may employ wildcard characters when creating Label Filters.

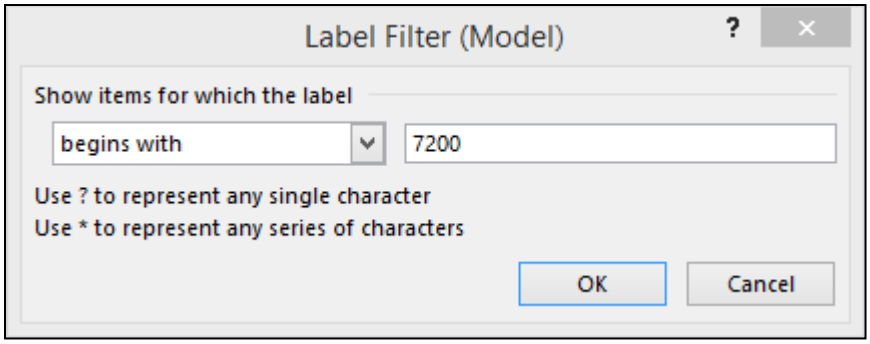


Figure 34 - Creating a Label Filter

Value, Including Top and Bottom "X" Results

Value Filters allow you to filter PivotTable data to only those rows whose values meet specific criteria such as "greater than," "less than," or "between." Also, Value Filters allow you to filter PivotTable data to ranking criteria such as "Top 10" or "Bottom 10" results. To apply a Value Filter, click the drop-down arrow next to the row field name that contains the data to filter and choose **Value Filters**. Next, select how your filter should function – e.g., "greater than" or "less than." Finally, enter the specific numeric values for the filter criteria. For instance, in the example shown in **Figure 35,** a user has established criteria that filter the results to rows with values greater than or equal to 5,000,000.

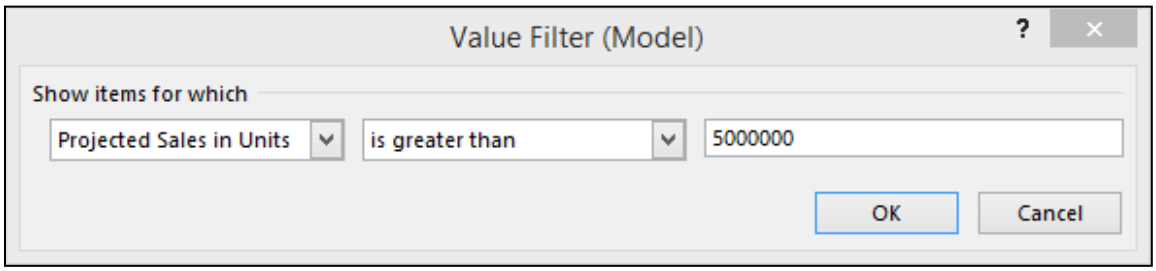


Figure 35 - Establishing Criteria for Value Filters

Date Filters

Date Filters allow you to specify time-oriented filters *relative to the computer's system clock*. For instance, date filters enable you to select a filter such as "This Quarter," automatically computed based on the computer's current date and time. In addition, date Filters update automatically whenever the computer's date changes. To apply a Date Filter to a PivotTable, click the drop-down arrow next to the date row field that contains the data to filter and choose **Date Filter**. Then, select the desired criteria from the pop-up menu. You must enter specific criteria in some cases, such as when filtering between a starting and ending date.

Applying Slicer Filters

A filter added to Excel 2010 is that of **Slicer**. A Slicer is a visual Report Filter, allowing even inexperienced users to filter a PivotTable quickly and efficiently to meet their specific needs. In addition, you can add multiple Slicers to the same PivotTable, extending their functionality and usefulness.

To add a Slicer to a PivotTable, choose **Insert Slicer** from the **PivotTable Tools, Analyze** contextual tab to open the **Insert Slicer** dialog box shown in **Figure 36**.

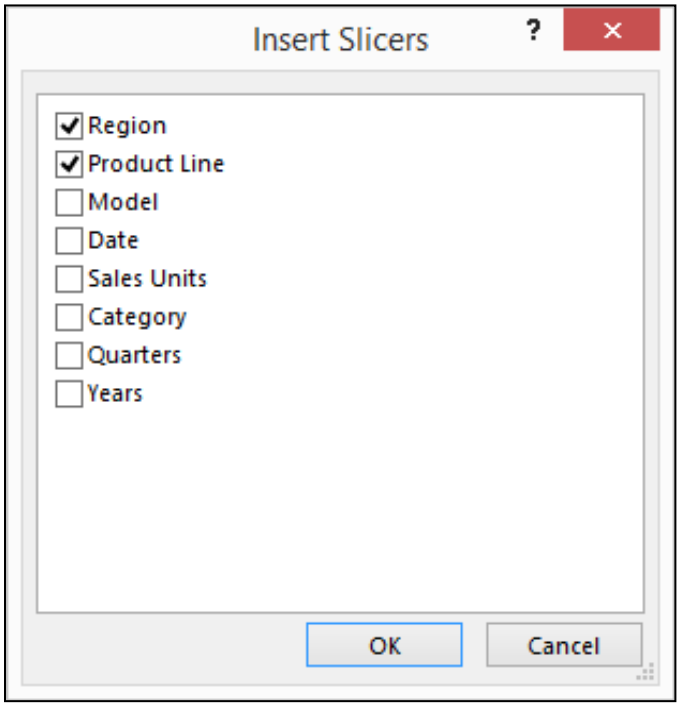
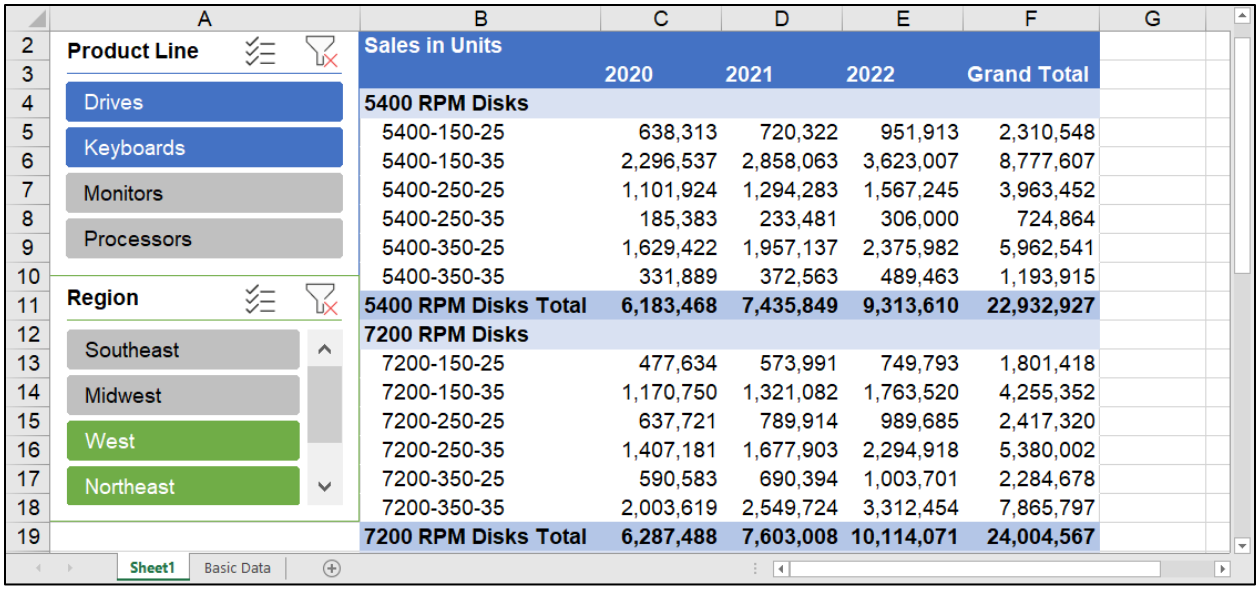


Figure 36 - Adding Slicers to a PivotTable

Click in the boxes next to each field for which you want to add a Slicer and click **OK**. Then, drag the Slicer to its desired location on the PivotTable report. In the example shown in **Figure 37**, two Slicers are present – one for the Region and one for the Product Line. Based on the filters applied by these Slicers, the PivotTable shows only results for the Northeast and West Regions and only for the Drives and Keyboards Product Lines.



Sales in Units	2020	2021	2022	Grand Total
5400 RPM Disks				
5400-150-25	638,313	720,322	951,913	2,310,548
5400-150-35	2,296,537	2,858,063	3,623,007	8,777,607
5400-250-25	1,101,924	1,294,283	1,567,245	3,963,452
5400-250-35	185,383	233,481	306,000	724,864
5400-350-25	1,629,422	1,957,137	2,375,982	5,962,541
5400-350-35	331,889	372,563	489,463	1,193,915
5400 RPM Disks Total	**6,183,468**	**7,435,849**	**9,313,610**	**22,932,927**
7200 RPM Disks				
7200-150-25	477,634	573,991	749,793	1,801,418
7200-150-35	1,170,750	1,321,082	1,763,520	4,255,352
7200-250-25	637,721	789,914	989,685	2,417,320
7200-250-35	1,407,181	1,677,903	2,294,918	5,380,002
7200-350-25	590,583	690,394	1,003,701	2,284,678
7200-350-35	2,003,619	2,549,724	3,312,454	7,865,797
7200 RPM Disks Total	**6,287,488**	**7,603,008**	**10,114,071**	**24,004,567**

Figure 37 - Slicers Added to a PivotTable Report

You can customize a Slicer's appearance by clicking on the Slicer and selecting the Ribbon's **Slicer Tools**, **Options** contextual tab. Color, size, and alignment options are accessible from that contextual tab. Additionally, by choosing **PivotTable Connections** on the **Slicer Tools**, **Options** contextual tab, you can link one slicer to multiple PivotTables. This feature is critical in dashboard environments where you will likely have numerous PivotTables and slics. You can speed up the filtering process by creating and applying Slicer filters and connecting them to multiple objects. Also, you can ensure that all reports filter to the same set of criteria.

Adding Timeline Filters to PivotTables

Like Slicers, **Timelines** allow you to filter PivotTable data based on dates. Assuming your PivotTable has at least one date field available, you can quickly add a Timeline by following these five steps.

1. Click in the PivotTable.
2. Click **Insert Timeline** from the **PivotTable Tools**, **Analyze** contextual tab.
3. In the resulting **Insert Timeline** dialog box, check the box next to each date field for which you want to add a Timeline.
4. Click **OK** to close the Insert Timeline dialog box.
5. Drag-and-drop to reposition and resize the Timeline.

The results of your efforts might resemble those shown in **Figure 38**.

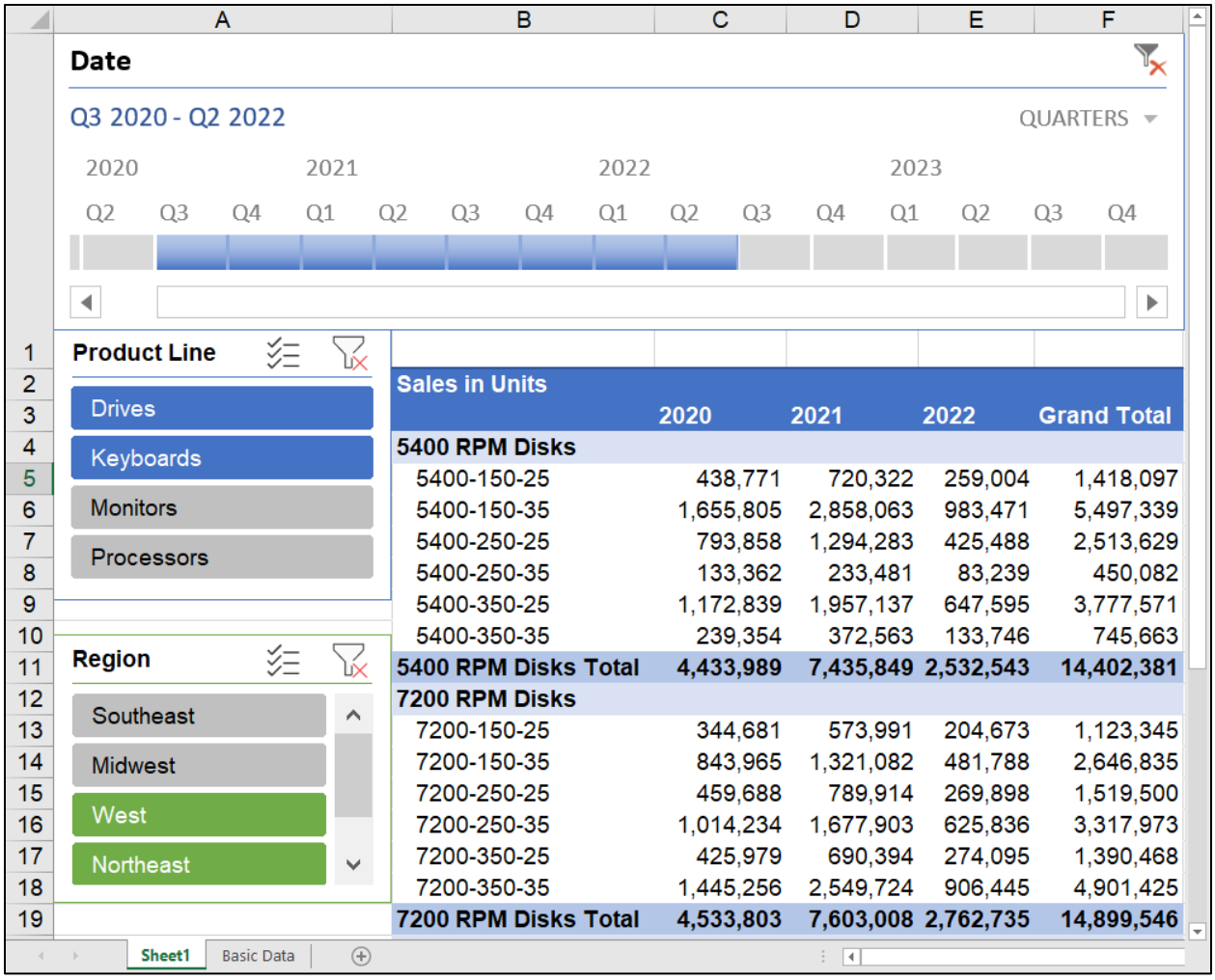


Figure 38 - PivotTable with Slicers and Timeline

?	Regarding the potential advantages of using PivotTables as a primary "dashboarding" tool, which might be the most important for the situations you will face when constructing BI dashboards and why?

Presenting Visualizations in Excel-based BI Reports and Dashboards

Charting Fundamentals

Any discussion of Excel-based BI reports and dashboards necessitates discussing Excel's visualization and charting features. In newer versions of Excel, you can create and customize charts with fewer clicks and dialog boxes, making it much easier and faster to produce professional-quality visualizations to incorporate into Excel-based BI reports and dashboards. This section provides a summary overview of charting functions and tools in Excel.

Excel includes a wide range of chart types in numerous broad categories. However, the number of charting options available to visually present data is virtually unlimited when combining the different variations of these chart types with the various chart styles, chart layouts, and user-defined charts.

Column charts, the most common of all chart types, display data points in vertical columns. Column charts compare discrete values and do not imply the passage of time. A **bar chart** is very similar to a column chart, except that a bar chart has a horizontal orientation, while a column chart has a vertical orientation. Bar charts are helpful when comparing many values and when category labels are lengthy.

Use **line charts** to plot continuous data over time. Line charts are handy for identifying trends in data. However, a line chart assumes that all the data points plotted are spaced evenly in time. Therefore, avoid using line charts because they might provide misleading information if they are not. Alternatively, you can use an **XY chart** to space the data points according to their relative time position. An **area chart** is fundamentally a line chart with the area below the line filled. Like line charts, area charts help display values over time.

XY charts, or scatter plots, show the relationships between variables plotted on the X- and Y-axes. The values on the X-axis are independent of those plotted on the Y-axis. In other words, the values plotted on the X-axis drive or impact the values plotted on the Y-axis. Importantly, there is no category axis in an XY chart. Both axes display *values*. **Bubble charts** are similar to XY charts but with an additional data series represented by the bubbles' size. Again, all data series are values – there are no category axes.

Pie charts display the relative makeup of data. They help identify proportional values or contributions to a total. For pie charts to be helpful, use no more than five or six data points. When you have more data points, consider using a bar chart, pie of pie chart, or bar of pie chart instead. A **doughnut chart** is a form of a pie chart, except it can display more than one data series. However, because each successive data series resides in concentric rings, it can be relatively easy to misinterpret the chart's meaning. Given this limitation, doughnut charts are usually best utilized with only one data series. If you need to chart multiple data series, consider using a stacked column chart instead.

Radar charts have separate axes for each category of data. These axes extend outward from the chart's center. Radar charts help identify relationships among data series and make comparisons of data values. With a **surface chart**, colors distinguish *values*, not series. Surface charts can display two or more data series on a surface and are often used to find optimum combinations between two datasets.

Stock charts help display securities' information, such as high, low, or closing stock prices. Also, stock charts may display opening values and volume. You can also use stock charts to graph scientific data. For example, stock charts may help present rainfall or temperature data.

Choosing a Chart Type

With all the choices available, sometimes the most challenging part of creating a chart is choosing which type is best for a given situation. However, a few guidelines may be helpful.

- When comparing items to other items, *column charts* and *bar charts* are typically the best choices.
- *Line* and *area charts* are usually superior to other options when comparing data over time.
- *Pie* and *doughnut charts* are helpful when making relative comparisons of multiple data points.
- Causal relationships between two variables are often best depicted with *XY charts*. If you need to plot three data values, use a *bubble chart* instead of an XY chart.

With a fundamental understanding of chart options, let us focus on creating charts, specifically PivotCharts.

PivotCharts

When is a PivotTable not a PivotTable? The answer is "When it is a **PivotChart**." A PivotChart is a graphical representation of a PivotTable with all the analytical power of the PivotTable. PivotCharts link inextricably to PivotTables so that changing the data or organization of one changes the other automatically. Because PivotCharts are just extensions of PivotTables, they are ideal tools for accountants and other business professionals to summarize large quantities of data and present those summaries in an easy-to-understand graphical format on an Excel-based BI report or dashboard.

You can create PivotCharts in numerous ways; however, since we already have a PivotTable created, we will make our first PivotChart from that PivotTable. To do so, start by clicking on the PivotTable. Then, select **PivotChart** from the **PivotTable Tools**, **Analyze** contextual tab of the Ribbon. Next, choose the chart type – columns, line, bar, or another type – and click **OK** to create a PivotChart similar to the one shown in **Figure 39**.

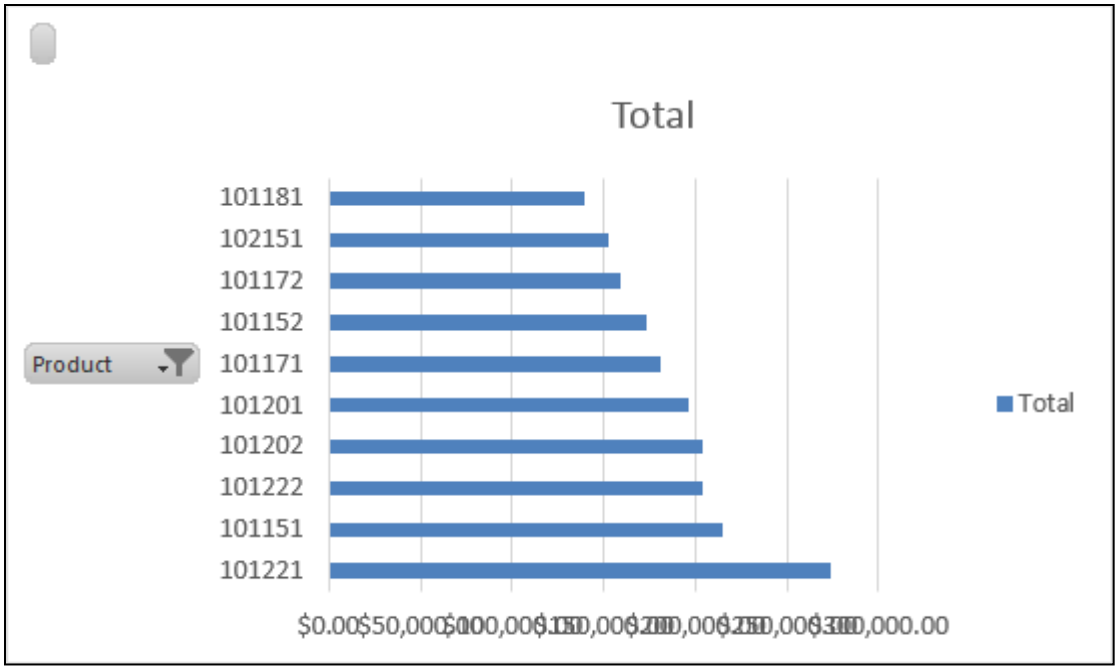


Figure 39 - Initial PivotChart

Having created the initial PivotChart, minor cleanup is necessary to transform the PivotChart into the one pictured in **Figure 40**.

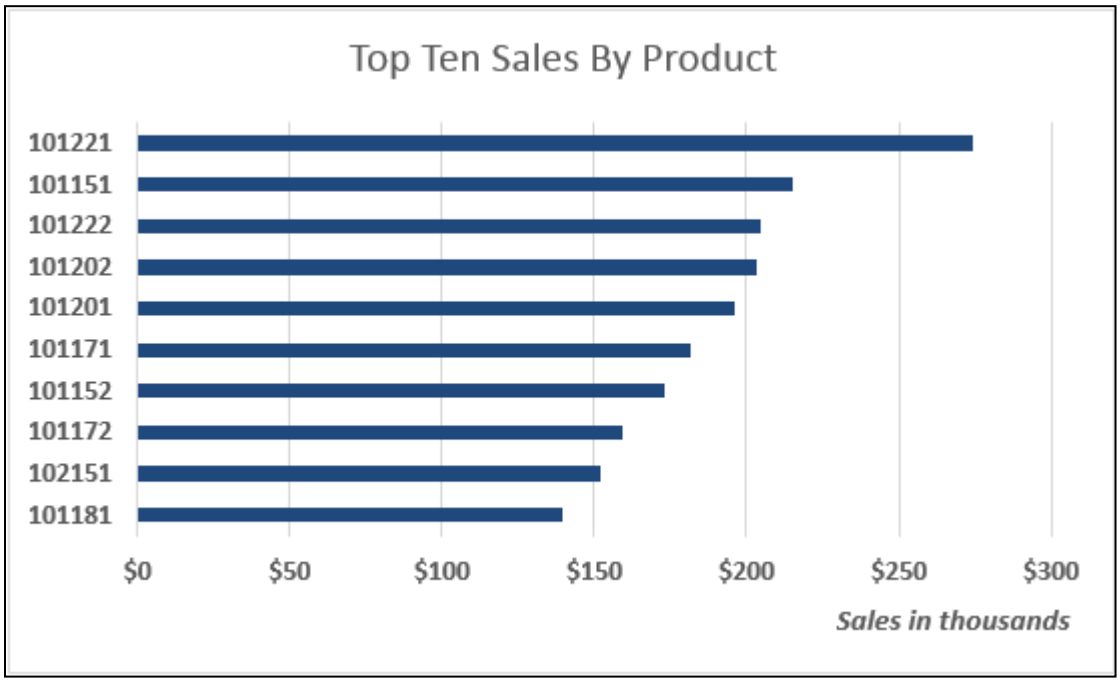


Figure 40 - Completed PivotChart Example

In our second example of working with PivotCharts, John Ward needs to convert the data from his client's Excel-based cash disbursements journal into a dashboard format so the client can better understand the information presented. John has already built a PivotTable to summarize the data. **Figure 41** illustrates the PivotTable.

3	Summary of Cash Disbursements					
4		Qtr1	Qtr2	Qtr3	Qtr4	Grand Total
5						
6	Beverages	14,620.76	14,630.86	16,476.61	14,706.65	60,434.88
7	Food	72,545.64	70,502.27	69,699.27	75,644.18	288,391.36
8	Insurance	1,281.99	1,281.99	1,521.07	1,640.61	5,725.66
9	Payroll Expense	75,715.47	83,907.75	90,288.01	97,520.26	347,431.49
10	Rent	7,200.00	7,200.00	7,200.00	7,200.00	28,800.00
11	Supplies	4,778.96	6,032.00	5,235.41	5,689.92	21,736.29
12	Utility Expense	5,130.76	7,217.52	7,949.89	5,207.34	25,505.51
13	**Grand Total**	**181,273.58**	**190,772.39**	**198,370.26**	**207,608.96**	**778,025.19**

Figure 41 - Copy of PivotTable Used to Summarize Excel-based Cash Disbursements Journal

To create a PivotChart from the PivotTable, John follows the process below.

1. John positions the cursor in the PivotTable and selects **PivotChart** from the **PivotTable Tools, Analyze** contextual tab to open the **Insert Chart** dialog box, as shown in **Figure 42**. Next, John chooses a **Stacked Column chart with 3-D effects** in the dialog box.

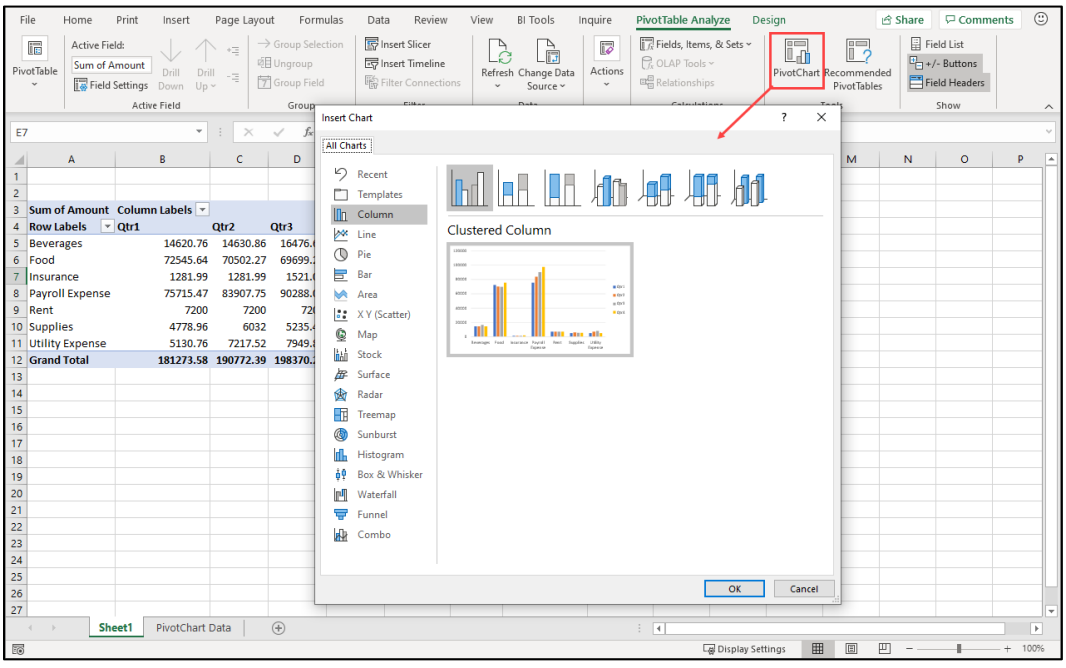


Figure 42 - Creating a PivotChart from a PivotTable

2. When John clicks **OK** in the **Insert Chart** dialog box, Excel creates the chart and displays the first draft on the same worksheet as the PivotTable. **Figure 43** shows that draft.

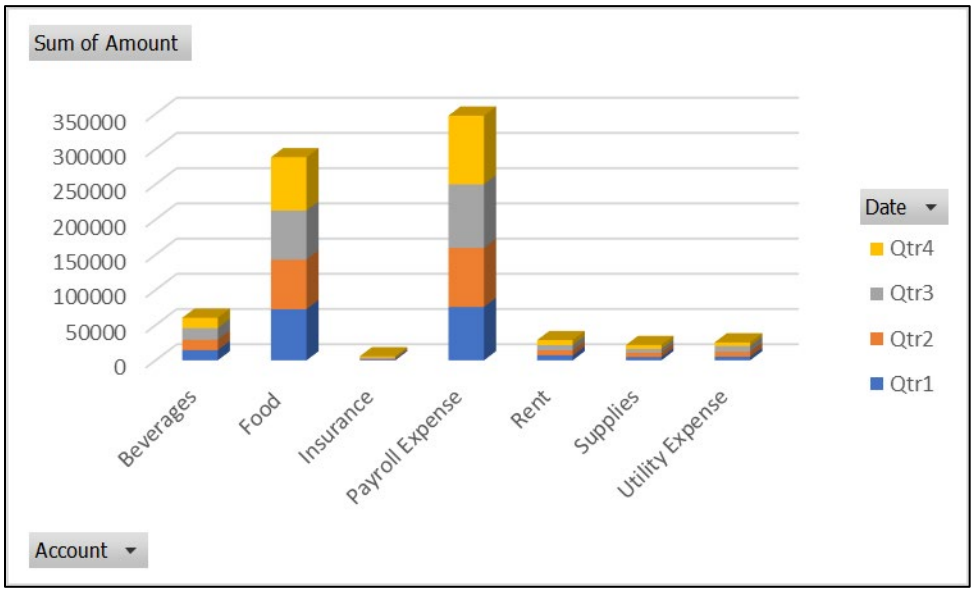


Figure 43 - First Draft of PivotChart

3. John is not satisfied with the draft. Principally, he wants to swap the rows and columns' positions – that is, he wants to summarize the data by quarter rather than by account on the horizontal axis. Fortunately, this customization is easy to make, and John accomplishes it by clicking on the **Switch Row/Column** icon on the **PivotChart Tools**, **Design** contextual tab. Additionally, John adds some additional formats so that the final chart resembles the one shown in **Figure 44**.

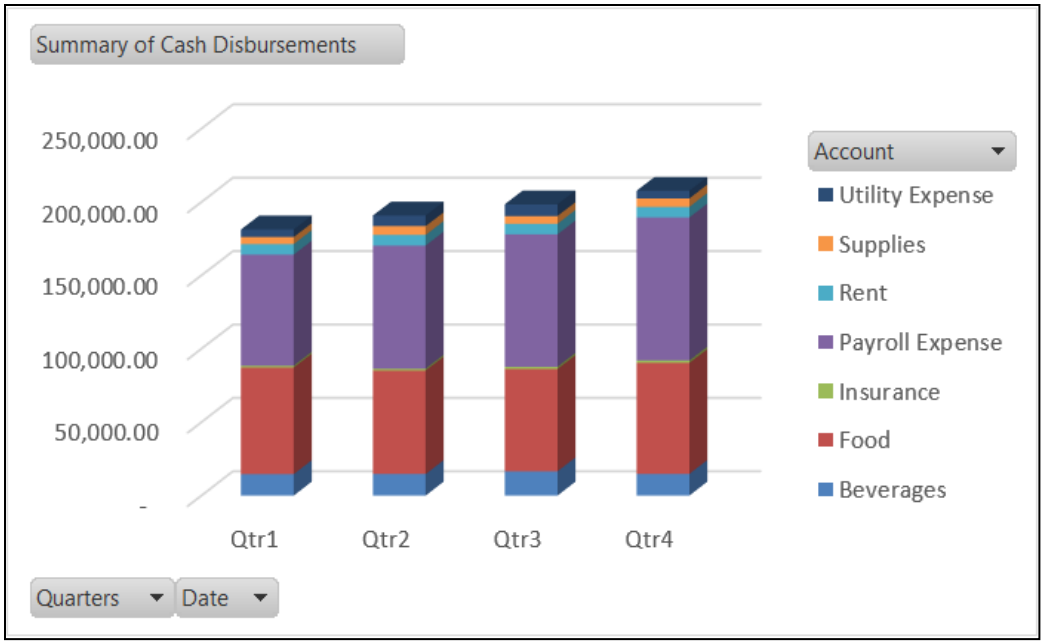


Figure 44 - Completed PivotChart Summarizing Cash Disbursements Journal

Now, John has an easy-to-read supplement to the traditional financial statement he prepared for this client. In building this PivotChart, John noticed several intriguing features of PivotCharts.

- A PivotChart is inextricably linked to its related PivotTable. If the PivotTable changes, the PivotChart also changes; likewise, if the PivotChart changes, the PivotTable does too. This feature is advantageous because it ensures the two objects stay in sync.

- PivotCharts contain on-screen filter buttons. For instance, referring to the PivotChart shown in Figure 44, John could filter the Accounts that the PivotChart displays by clicking the drop-down arrow next to Account to open the filter pane, as shown in **Figure 45**.

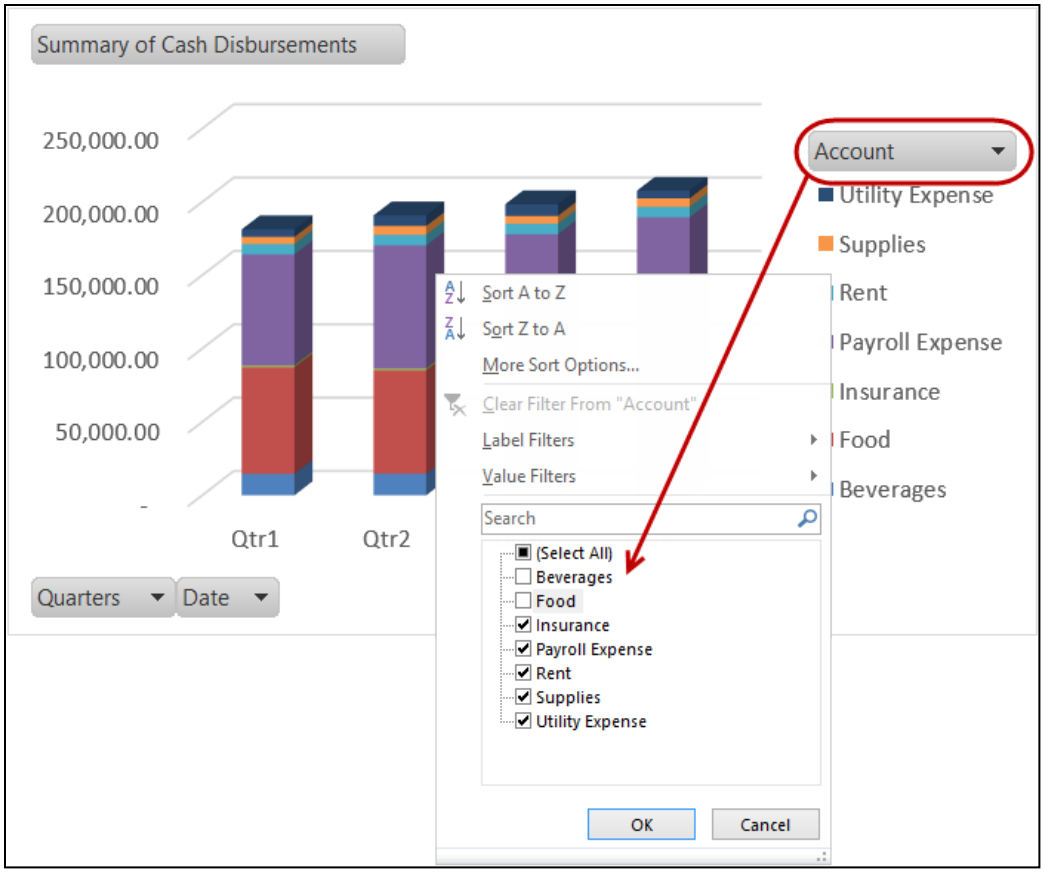


Figure 45 - Filtering a PivotChart in Excel

Applying a filter to a PivotChart will change the related PivotTable as described above. Additionally, John can disable the control if he does not desire on-screen filtering capability.

- From a formatting perspective, a PivotChart is just like any other chart. In other words, all traditional formatting options are available. Thus, John can choose to format his chart using the same tools and options he would use if this were a "traditional" Excel chart.

From the financial reporting perspective, using Excel-based BI reports and dashboards, PivotCharts are one of Excel's least known and least used tools; however, they offer tremendous power and flexibility to those who learn how to work with them.

Conditional Formatting

Though dashboards generally incorporate graphical data, they present tabular data as well. When a dashboard shows tabular data, Excel's **Conditional Formatting** tools can be excellent options for highlighting significant values, variances, trends, and other data. Though conditional formatting is not a new feature to Excel, more recent versions greatly enhance this feature. These enhancements readily lend themselves to use in Excel-based BI reports or dashboards.

The simplest form of conditional formatting is that based on values. For instance, conditional formatting based on values is present in the **Variance $** column shown in **Figure 46**. Any cell containing an unfavorable variance greater than $15,000 adds shading automatically, and the text color changes to red.

	A	B	C	D
1	**K2 Electrical Supplies, Inc.**			
2	Statement of Operating Expenses			
3	*For the Year Ended December 31, 2021*			
4				
5		**Budget**	**Actual**	**Variance $**
6	Advertising	$ 16,000	$ 17,500	$ (1,500)
7	Amortization	2,500	2,500	-
8	Depreciation	47,000	52,500	(5,500)
9	Employee Benefits	350,000	356,200	(6,200)
10	Insurance	417,000	409,000	8,000
11	Meals and Entertainment	49,000	62,000	(13,000)
12	Office Supplies and Expenses	65,000	81,600	(16,600)
13	Payroll Taxes	231,000	229,000	2,000
14	Property Taxes	12,500	13,500	(1,000)
15	Rent	360,000	360,000	-
16	Travel	96,500	110,000	(13,500)
17	Utilities	187,000	226,000	(39,000)
18	Wages and Salaries	2,100,000	2,250,000	(150,000)
19	**Total Operating Expenses**	**$ 3,933,500**	**$ 4,169,800**	**$ (236,300)**

Figure 46 - Conditional Formatting Based on Values

To apply this conditional format, select the range of cells from D5 through D18. Then, choose **Conditional Formatting** from the **Home** tab of the Ribbon and select **Highlight Cells Rules** followed by **Less Than** to open the dialog box shown in **Figure 47**. In that dialog box, specify the criteria for the desired formats. Finally, click **OK** to apply the conditional format to all selected cells.

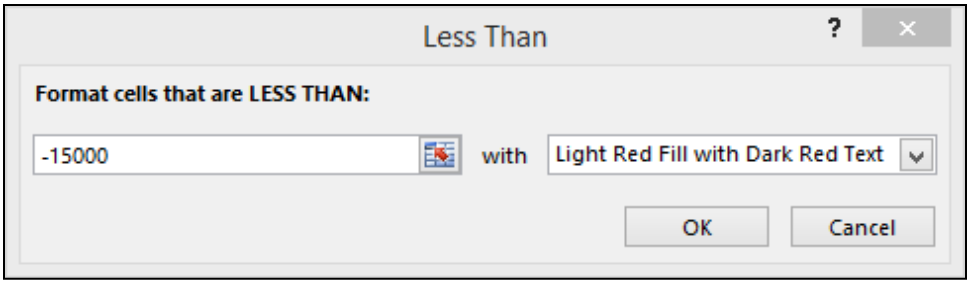


Figure 47 - Defining a Conditional Format Rule

Conditional Formatting Based on Formulas

In addition to applying conditional formatting based on discrete value inputs, conditional formatting can also be applied based on formulas. For instance, formula-based conditional formatting could identify unfavorable variances exceeding 15% of the budget. Such a formula might resemble that shown in **Figure 48**.

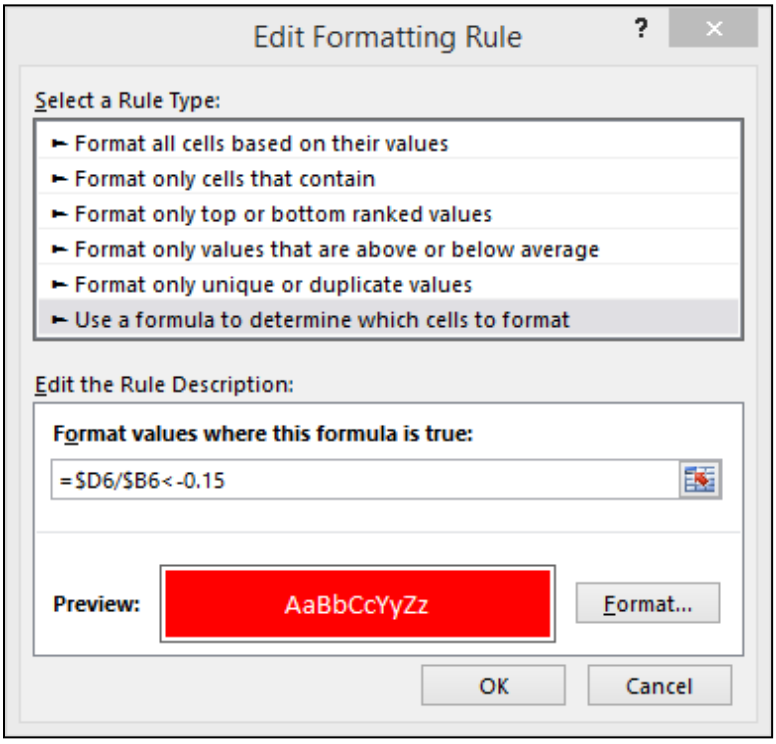


Figure 48 - Conditional Format Applied with a Formula

The formula and format specified in Figure 48 will format any unfavorable variance exceeding 15% of the budgeted amount with a red background and white text, as shown in **Figure 49**.

	A	B	C	D
1	**K2 Electrical Supplies, Inc.**			
2	Statement of Operating Expenses			
3	*For the Year Ended December 31, 2021*			
4				
5		**Budget**	**Actual**	**Variance $**
6	Advertising	$ 16,000	$ 17,500	$ (1,500)
7	Amortization	2,500	2,500	-
8	Depreciation	47,000	52,500	(5,500)
9	Employee Benefits	350,000	356,200	(6,200)
10	Insurance	417,000	409,000	8,000
11	Meals and Entertainment	49,000	62,000	(13,000)
12	Office Supplies and Expenses	65,000	81,600	(16,600)
13	Payroll Taxes	231,000	229,000	2,000
14	Property Taxes	12,500	13,500	(1,000)
15	Rent	360,000	360,000	-
16	Travel	96,500	110,000	(13,500)
17	Utilities	187,000	226,000	(39,000)
18	Wages and Salaries	2,100,000	2,250,000	(150,000)
19	**Total Operating Expenses**	**$ 3,933,500**	**$ 4,169,800**	**$ (236,300)**

Figure 49 - Results of a Conditional Format Applied via Formula

Because you can apply multiple conditional formatting rules to the same cells when designing your dashboards, it is possible to "layer" these rules on top of each other. For example, in addition to unfavorable variances exceeding 15% in red, you might want to showcase favorable variances of more than 5% in blue. To do so, add a second conditional format rule, as shown in **Figure 50**.

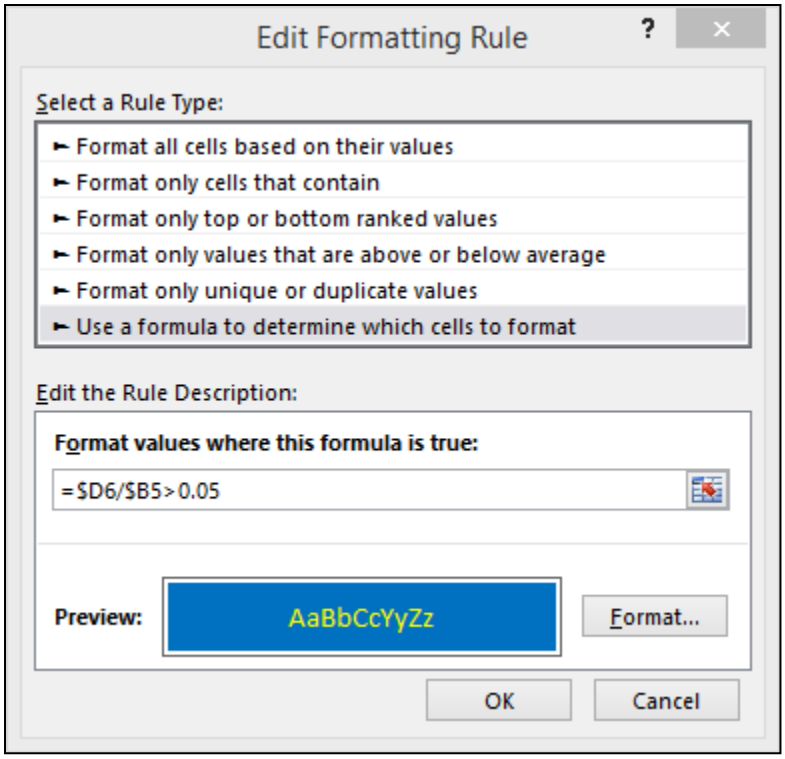


Figure 50 - Adding a Second Conditional Format

With the second conditional format in place, your dashboard component might resemble that shown in **Figure 51**.

	A	B	C	D
1	**K2 Electrical Supplies, Inc.**			
2	Statement of Operating Expenses			
3	*For the Year Ended December 31, 2021*			
4				
5		**Budget**	**Actual**	**Variance $**
6	Advertising	$ 16,000	$ 17,500	$ (1,500)
7	Amortization	2,500	2,500	-
8	Depreciation	47,000	52,500	(5,500)
9	Employee Benefits	350,000	356,200	(6,200)
10	Insurance	417,000	399,000	18,000
11	Meals and Entertainment	49,000	62,000	(13,000)
12	Office Supplies and Expenses	65,000	81,600	(16,600)
13	Payroll Taxes	231,000	229,000	2,000
14	Property Taxes	12,500	13,500	(1,000)
15	Rent	360,000	360,000	-
16	Travel	96,500	110,000	(13,500)
17	Utilities	187,000	226,000	(39,000)
18	Wages and Salaries	2,100,000	2,250,000	(150,000)
19	**Total Operating Expenses**	**$ 3,933,500**	**$ 4,159,800**	**$ (226,300)**

Figure 51 - Multiple Conditional Formats Applied to the Same Cells

Conditional Formatting Options in Excel

Newer versions of Excel provide significant functionality in conditional formatting, greatly expanding the usefulness of conditional formatting as a tool for enhancing Excel-based BI reports and dashboards. Two of the more significant enhancements are:

1. Removing the previous limitation of only three conditional formats applied to a given cell so that now you are limited only by the amount of available memory on the computer and

2. As shown in **Figure 52**, adding the **Conditional Formatting Rules Manager** provides a convenient location for managing all conditional format rules in a workbook.

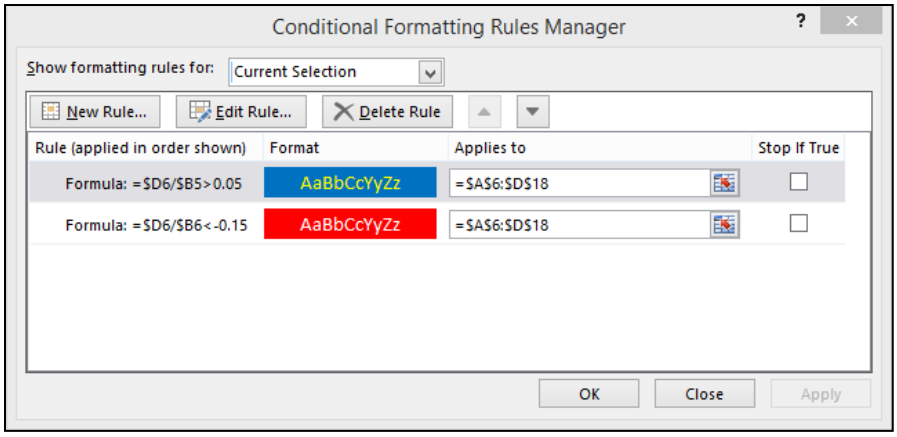
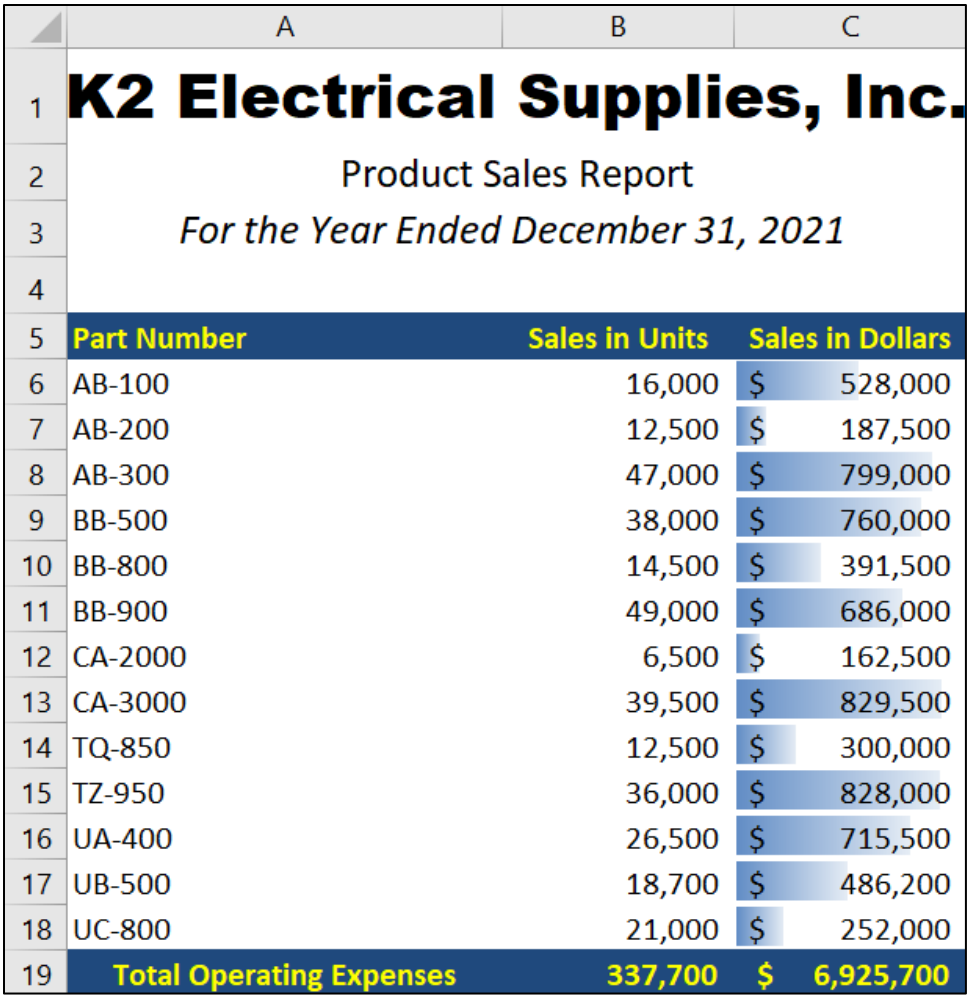


Figure 52 - Conditional Formatting Rules Manager

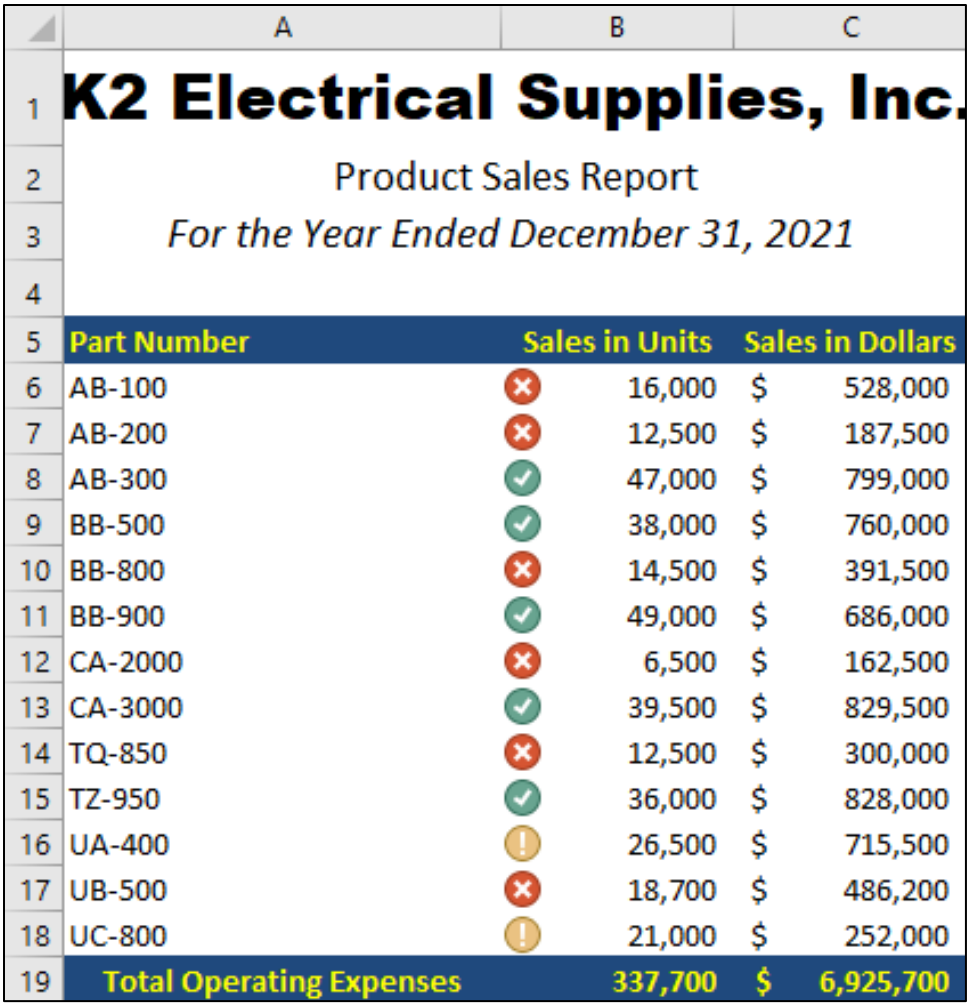
Beyond the enhancements described above, Excel also offers many conditional format options. One such option is that of **in-cell data bars**. In the example provided in **Figure 53**, in-cell data bars call the reader's attention to top-selling items without the need to sort them out of their alphabetical sort order. Like all conditional formats, as the underlying values change, the conditional formats will change also.



	A	B	C
1	**K2 Electrical Supplies, Inc.**		
2	Product Sales Report		
3	*For the Year Ended December 31, 2021*		
4			
5	**Part Number**	**Sales in Units**	**Sales in Dollars**
6	AB-100	16,000	$ 528,000
7	AB-200	12,500	$ 187,500
8	AB-300	47,000	$ 799,000
9	BB-500	38,000	$ 760,000
10	BB-800	14,500	$ 391,500
11	BB-900	49,000	$ 686,000
12	CA-2000	6,500	$ 162,500
13	CA-3000	39,500	$ 829,500
14	TQ-850	12,500	$ 300,000
15	TZ-950	36,000	$ 828,000
16	UA-400	26,500	$ 715,500
17	UB-500	18,700	$ 486,200
18	UC-800	21,000	$ 252,000
19	**Total Operating Expenses**	**337,700**	**$ 6,925,700**

Figure 53 - Using In-cell Data Bars as Conditional Formats

Applying **icon sets** as conditional formats is another relatively new feature. Icon sets are similar to in-cell data bars and call attention to values requiring management action. For example, the icon sets used in **Figure 54** segregate sales into three categories – high selling, average selling, and low selling – and call attention to items in each category.



	A	B	C
1	**K2 Electrical Supplies, Inc.**		
2	Product Sales Report		
3	*For the Year Ended December 31, 2021*		
4			
5	**Part Number**	**Sales in Units**	**Sales in Dollars**
6	AB-100	16,000	$ 528,000
7	AB-200	12,500	$ 187,500
8	AB-300	47,000	$ 799,000
9	BB-500	38,000	$ 760,000
10	BB-800	14,500	$ 391,500
11	BB-900	49,000	$ 686,000
12	CA-2000	6,500	$ 162,500
13	CA-3000	39,500	$ 829,500
14	TQ-850	12,500	$ 300,000
15	TZ-950	36,000	$ 828,000
16	UA-400	26,500	$ 715,500
17	UB-500	18,700	$ 486,200
18	UC-800	21,000	$ 252,000
19	**Total Operating Expenses**	**337,700**	**$ 6,925,700**

Figure 54 - Using Icon Sets as Conditional Formats

Extending Excel with BI Tools

Chapter 3

As an increasing number of executives, managers, consultants, and trusted advisors recognize the importance of transforming data into actionable business intelligence, the challenge they must address is "how." More specifically, they must determine how to provide the tools required to enable themselves and others to create business intelligence reports and dashboards. Further, they must also focus on making their dashboards visible, goal-oriented, graphical, and interactive, providing real-time information on business-critical metrics. One option is deploying Excel-based tools that allow individual subscribers to leverage their knowledge to create self-service business intelligence. In this chapter, you will learn about these tools. More specifically, you will identify how to acquire them, how they interact to facilitate do-it-yourself business intelligence, and how to build BI dashboards using these tools.

Learning Objectives

Upon completing this chapter, you should be able to:

- List the BI tools available in Excel and describe the purpose of each;
- Discuss the relationships between the various BI tools in Excel;
- Extract data from multiple data sources using Power Query;
- Define Data Models and describe their importance to Power BI-based reporting objects; and
- Use Power Pivot to summarize large volumes of data.

Before beginning the discussions in this chapter, it is essential to know that many of the features available in Power BI, which we will discuss in Chapters 4 and 5, are direct descendants of the Excel-based BI tools discussed in this chapter. Therefore, please view the conversations in this chapter as prerequisites for the discussions in future chapters.

Which BI Tools are Available in Excel?

You can extend Excel's capabilities for BI purposes with several tools Microsoft made available beginning with Excel 2010. These include **Power Query**, **Power Pivot**, **3D Maps** (known as **Power**

Maps before Excel 2016), and **Power View.**[5] These tools allow you to analyze data in real-time and explore data, design interactive reports, and share this information in dashboards.

Unfortunately, the relationship between these tools can be challenging to understand, and not all of these tools are available or supported in all versions of Excel. For example, suppose you have your Excel license provisioned through Microsoft 365/Office 365 ProPlus, Office Professional Plus 2013 and newer, or a stand-alone edition of Excel 2013 and newer. In that case, you have access to each of the four tools mentioned previously to create compelling and interactive dashboards to support your BI initiatives.

Viewed graphically, the relationship between some of the components of the tools listed above appears as shown in **Figure 55**. From the graphic, the functionality and the relationship of these tools emerge.

- You can use Power Query to link data from external data sources into an Excel data model.

- To manage the data model, you work with Power Pivot. With Power Pivot, you can establish and maintain relationships between multiple tables in a data model. You can also create user-defined calculations and add Key Performance Indicators (KPIs) to BI dashboards.

- To present the results of your BI efforts, you can use 3D Maps to depict the data in the data model graphically. In addition, both tools facilitate end-user interactivity, allowing end-users to initiate such transactions as filtering data and drilling into the detailed transactions.

[5] Microsoft's retirement of a key technology that facilitated Power View – Silverlight – means that from a practical perspective, Power View is no longer a viable option for business intelligence efforts in an Excel environment.

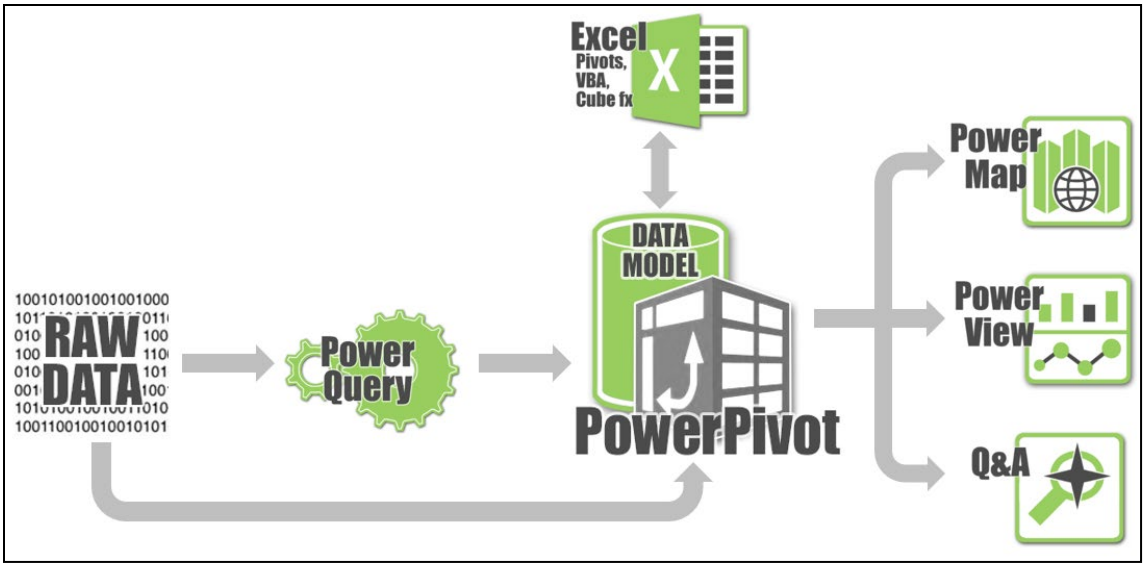


Figure 55 – Relationship of Power BI Tools and Related Components

Do I Need Excel's BI Tools?

In some cases, you may be able to create your BI-reporting objects without using any of these tools. For instance, consider the following scenarios.

- If you merely need to summarize a large volume of data residing in Excel, you can do so quickly and easily by building a simple Excel PivotTable.

- If the data you need to summarize resides in two or more Excel *tables*, you can build a data model from your tables, assuming you run Excel 2013 or newer. Then, you can easily create a PivotTable from that data model.

- If the data you need to summarize in a PivotTable resides in multiple databases, you can query it into your data model using Power Pivot or Power Query. Once you build the data model, you can create your PivotTable.

- In all versions of Excel, to create relatively simple, interactive visualizations of data summarized in a PivotTable, you can utilize PivotCharts. As discussed in Chapter 2, PivotCharts are graphical extensions or representations of PivotTables.

With a background in Excel's BI tools and related components, we can now focus on using each to generate BI reports and dashboards and make BI available and visible throughout the organization.

?	What are some potential strengths of using a “pure” Excel approach to your business intelligence efforts? Conversely, what are some of the potential drawbacks to this approach?

Extracting Data Using Power Query

Power Query provides a tool for accessing, linking, and transforming data. Of course, once you link your data into Excel, you can manipulate it virtually any way you need to achieve your desired results. When importing data using Power Query, you can use many different data sources, including those listed below.

- Websites
- Excel, CSV, XML, Text, and metadata about files in a folder
- SQL Server, Access, SQL Analysis Services, Oracle, IBM DB2, MySQL, PostgreSQL, Sybase, and Teradata databases
- Microsoft Azure SQL databases, Microsoft Azure Marketplace, Microsoft Azure HDInsight, Microsoft Azure BLOG Storage, and Microsoft Azure Table Storage
- Other sources, including Facebook, SharePoint Lists, Active Directory, Microsoft Exchange, ODBC, and custom-written queries

Note that you are not limited to keeping the results of your data queries executed with Power Query in isolation. Instead, you can relate the data from multiple queries inside a data model and use the data model as the foundation of your business intelligence dashboards and reports. For example, you may use Power Query to link data from one or more websites into a data model. Likewise, you could simultaneously use Power Query to link your accounting software’s data into the same data model.

Consider the following example to illustrate how you can use Power Query to locate and import data quickly that you might want to use as part of a BI report or dashboard. For example, suppose you need access to population data by state or province to determine how effectively your retail organization reaches customers in specific geographic regions. Then, you could use Power Query, as outlined below, to quickly identify and import that information.

1. Click the **From Web** in the **Get & Transform** tab on the **Data** tab of the Ribbon.

2. In the **From Web** dialog box, enter the following URL as the data source: *https://en.wikipedia.org/wiki/List_of_U.S._states_and_territories_by_population*.[6]

3. In the Navigator, select the **Population of states, territories, divisions, and regions** source, as shown in **Figure 56**. Then, click **Load** to return the data to the Excel workbook.

Figure 57 provides an abbreviated sample of the data that the query returns.

An important consideration when working with Power Query is the tool's capability to assist with data transformations. For example, you can edit your queries by reformatting data, adding or deleting columns, or inserting user-defined calculations. Power Query "remembers" each of these transformations as an **Applied Step**. Once you create an Applied Step, each time you refresh or reuse the query, Power Query automatically executes the Applied Step, freeing you from repeatedly transforming your data using manual procedures.

[6] As alternative, enter *https://en.wikipedia.org/wiki/List_of_Canadian_provinces_and_territories_by_population* to locate population data for Canada.

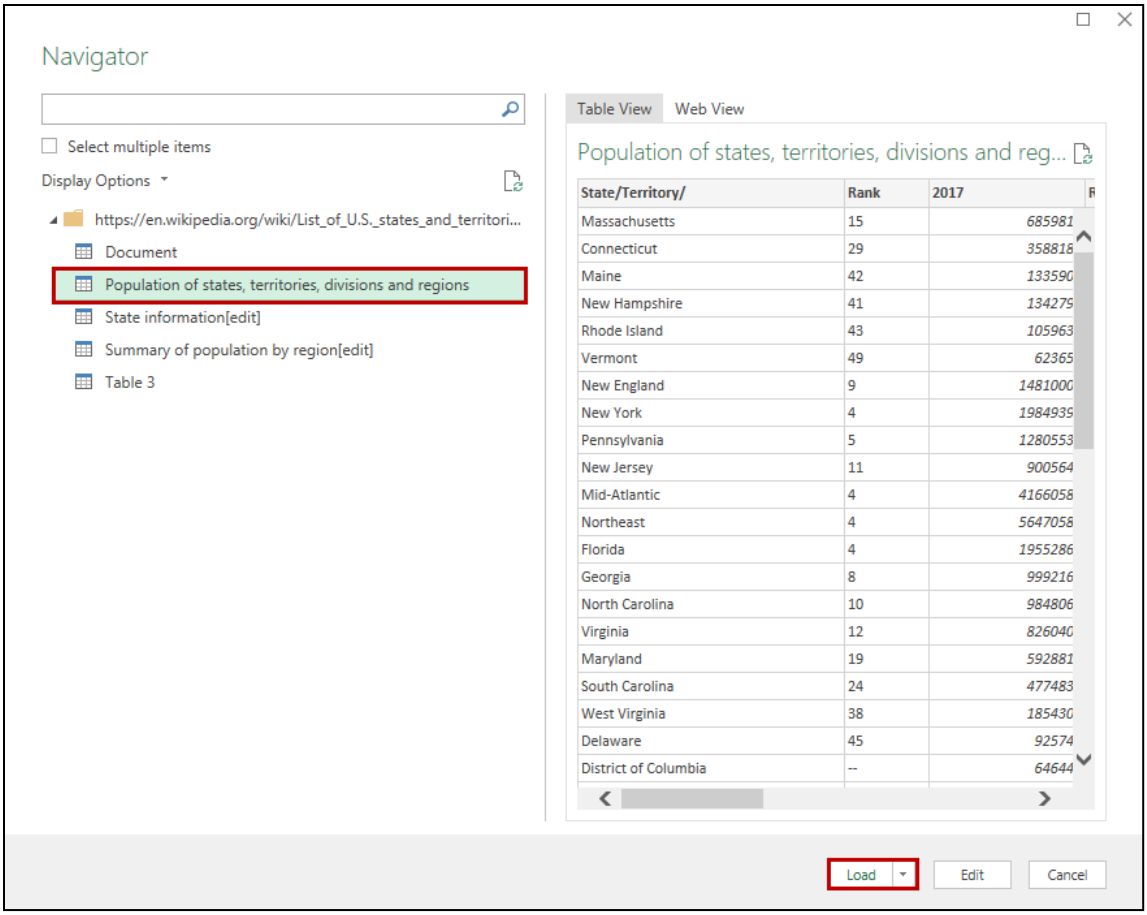


Figure 56 – Sample Power BI Query

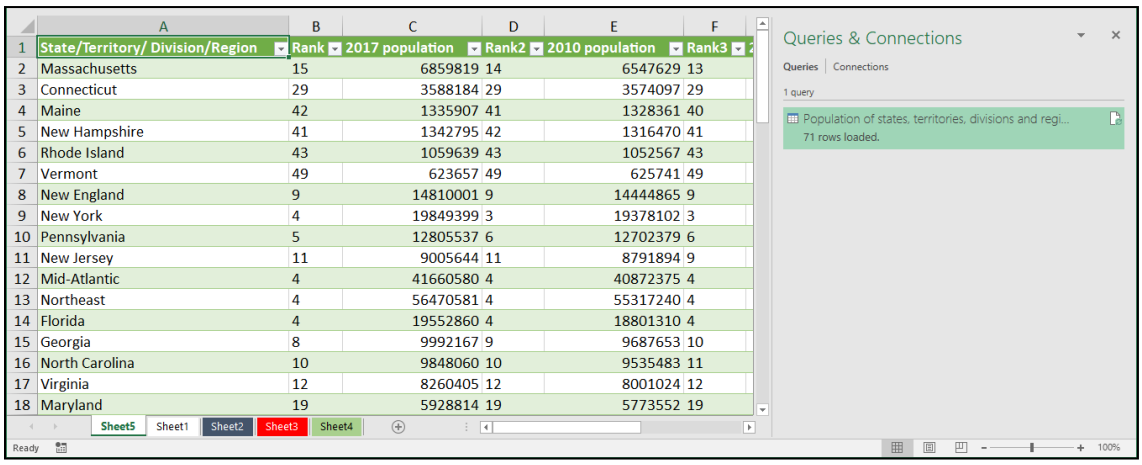


Figure 57 - Sample of Data Returned from Power Query

Data Models – The Core of Power BI and Excel-based BI Projects

Added natively to all versions of Excel beginning with the 2013 release, **data models** allow you to relate multiple tables together inside Excel. In this fashion, data models resemble simple relational databases inside Excel. Furthermore, once you construct an Excel data model, you can create Power View and 3D Maps visualizations and PivotTables from the data model. As a result, data models are at the core of most Excel-based BI projects.

You can create data models using multiple Excel techniques, including the following.

- When importing data into Excel using Power Query, choose multiple tables, and Excel will automatically import them into the workbook's data model.

- Suppose you are building a PivotTable from a table. Excel will add *all* tables in the workbook to the data model if you indicate you want to add the selected table to the data model.

- Excel creates the data model from the queried data when using Power Pivot to link the data into the workbook.

Note that each Excel workbook can contain only one data model. However, you can use that model as source data for multiple reporting objects in the same workbook.

Working with Power Pivot

Power Pivot provides you with self-service business intelligence by extending the functionality of Excel PivotTables. Principally, Power Pivot serves as the primary tool for managing data models in Excel. With Power Pivot, you can continue to work within the familiar Excel framework and utilize many existing Excel features. However, you can do so with large datasets without any noticeable performance degradation. Additionally, you can relate multiple data sources of almost any type inside Power Pivot and produce PivotTables from the resulting related data model. Power Pivot also allows you to create formulas in your data models using **Data Analysis Expressions** (**DAX**.)

Limitations exist with PivotTables based on data models, and you should understand them before embarking on such a project. Some of the primary limitations include those listed below.

- If you need to manage the data model from which you create a PivotTable, you cannot use the Grouping option in the PivotTable to group multiple items of the same field. You can group multiple items, but you will utilize a different technique.

- By default, although you can drill down on a PivotTable created from a data model, you will only see the first 1,000 rows of data.

- You cannot insert calculated fields or items inside a PivotTable built from a data model.

Notwithstanding the above, the advantages of creating PivotTables from Power Pivot-managed data models are so significant that many long-time PivotTable users are migrating their PivotTables to this format.

Creating a Basic PivotTable with Power Pivot

To build a PivotTable in the Power Pivot environment, click **Manage** on the **Power Pivot** tab of the Ribbon in a blank workbook to open the Power Pivot window. In the Power Pivot window, import the data from which you will build the PivotTable by clicking **From Other Sources** to open the **Table Import Wizard** dialog box shown in **Figure 58**. Power Pivot can import data from many familiar sources, including SQL Server, Access, Analysis Services, the Windows Azure Marketplace, Oracle, Sybase, Excel, and text. Because the data resides in Excel, choose **Excel File** from the Table Import Wizard dialog box.

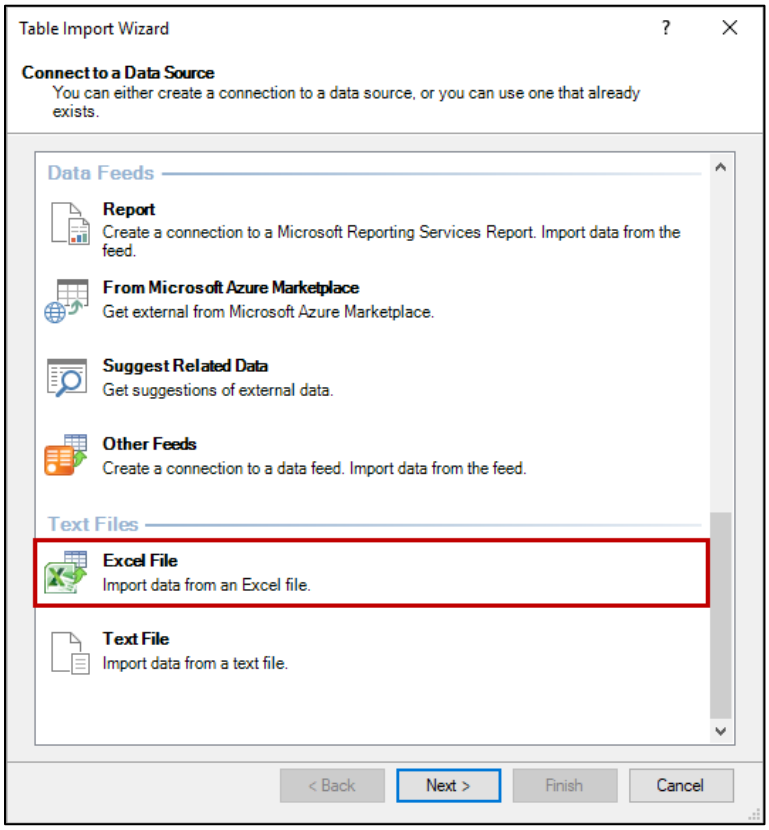


Figure 58 – Table Import Wizard Dialog Box in Power Pivot

As shown in **Figure 59**, indicate the file you wish to import using the **Table Import Wizard**, click **Next**, and specify the sheet for importing; Power Pivot then imports the data. Remember that if the data resides in a database such as Access or SQL Server, there is no ***practical*** limitation on the volume of data you can import into Power Pivot. In other words, Power Pivot does not restrict you to the limit of 1,048,576 rows that Excel otherwise imposes.

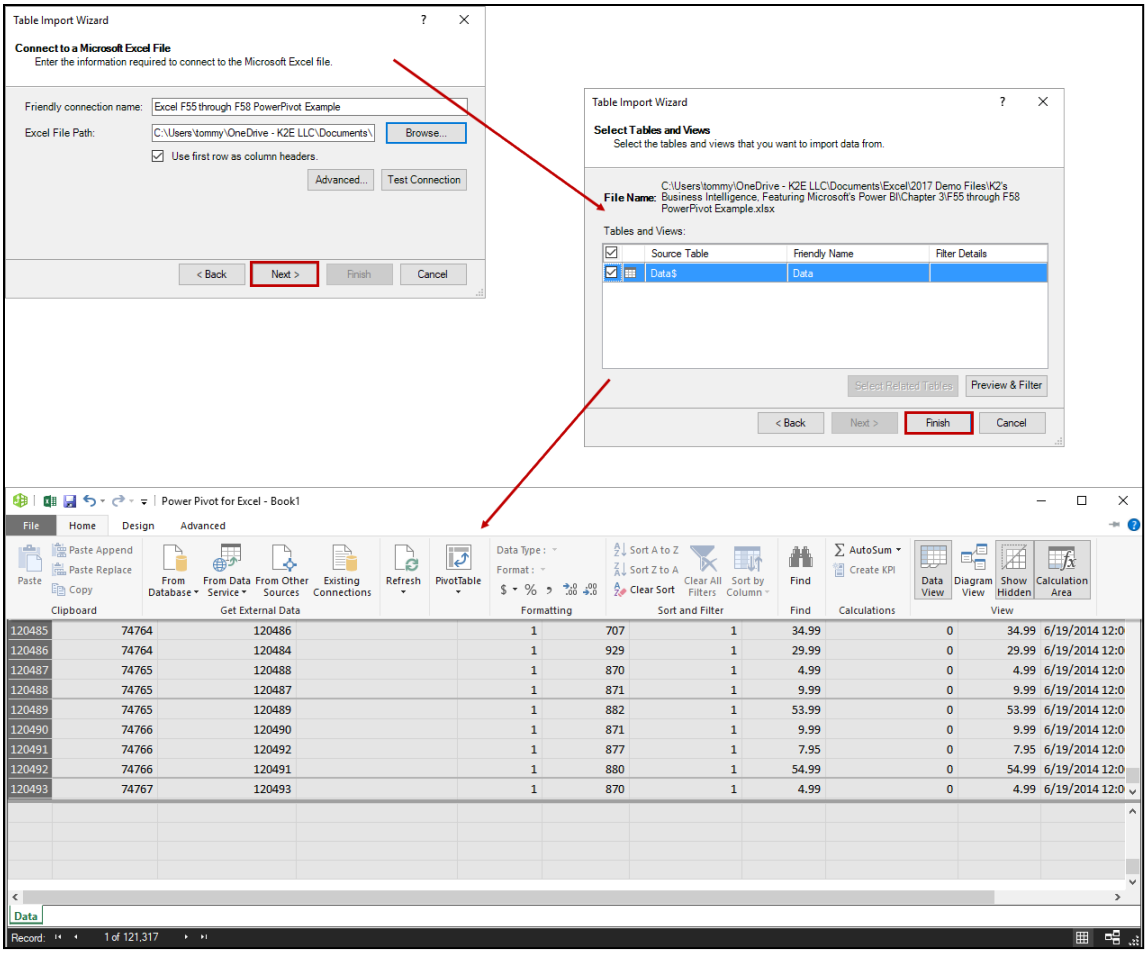
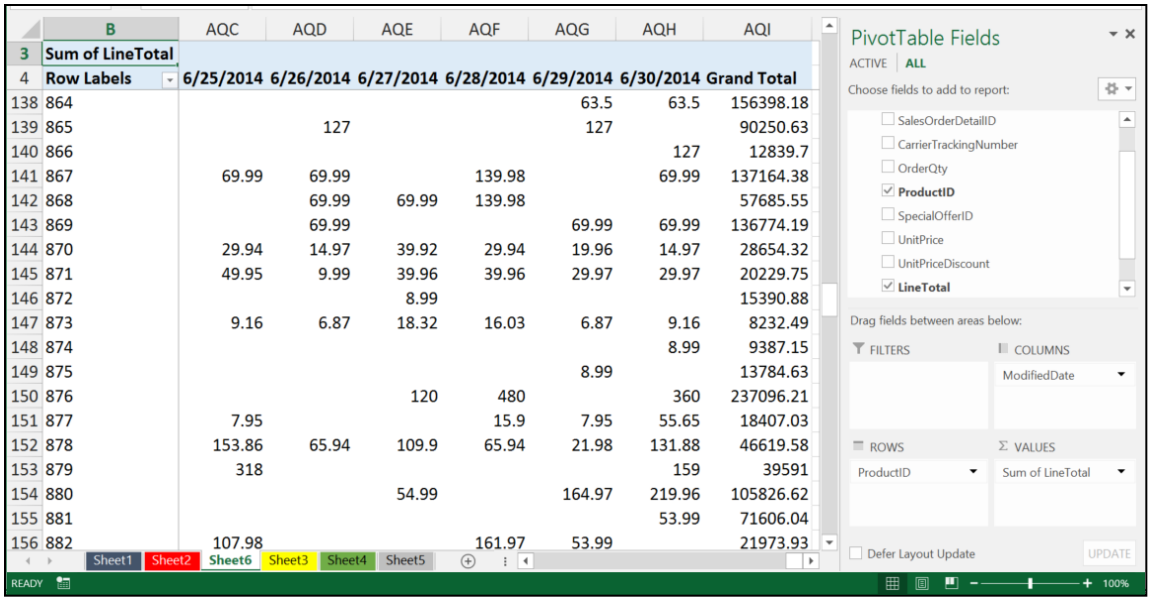


Figure 59 – Importing Data into Power Pivot

With the data imported, click **PivotTable** from the **Power Pivot Ribbon's Home tab**. Then select the new PivotTable's location, and click **OK** to open the familiar Excel PivotTable window. From there, you can create your PivotTable. **Figure 60** presents an example of a completed report.



	B	AQC	AQD	AQE	AQF	AQG	AQH	AQI
3	Sum of LineTotal							
4	Row Labels	6/25/2014	6/26/2014	6/27/2014	6/28/2014	6/29/2014	6/30/2014	Grand Total
138	864					63.5	63.5	156398.18
139	865		127			127		90250.63
140	866						127	12839.7
141	867	69.99	69.99		139.98		69.99	137164.38
142	868		69.99	69.99	139.98			57685.55
143	869		69.99			69.99	69.99	136774.19
144	870	29.94	14.97	39.92	29.94	19.96	14.97	28654.32
145	871	49.95	9.99	39.96	39.96	29.97	29.97	20229.75
146	872			8.99				15390.88
147	873	9.16	6.87	18.32	16.03	6.87	9.16	8232.49
148	874						8.99	9387.15
149	875					8.99		13784.63
150	876			120	480		360	237096.21
151	877	7.95			15.9	7.95	55.65	18407.03
152	878	153.86	65.94	109.9	65.94	21.98	131.88	46619.58
153	879	318					159	39591
154	880			54.99		164.97	219.96	105826.62
155	881						53.99	71606.04
156	882	107.98			161.97	53.99		21973.93

Figure 60 – Basic PivotTable Created in Power Pivot

As previously mentioned, Power Pivot constructs a data model from the source data; you then build the PivotTable using the data model as its source data. Because of this, your PivotTables can contain almost unlimited records. However, you cannot group data in a PivotTable based on a data model, at least not using traditional means such as the automatic date grouping option available in traditional PivotTables.

To group the data shown in the PivotTable in Figure 60, we can take one of two approaches:

1. Add month, quarter, and year fields to the source data before importing the source data into Power Pivot or

2. After importing the data into Power Pivot, build Data Analysis Expressions to add the source data's month, quarter, and year fields.

Choosing the first option above and returning to the source data, we could add the following formulas in cells K2, L2, and M2 to generate fields for months, quarters, and years.

- In cell K2: **=MONTH(J2)**
- In cell L2: **=IF(MONTH(J2)<4,1,IF(MONTH(J2)<7,2,IF(MONTH(J2)<10,3,4)))**
- In cell M2: **=YEAR(J2)**

With these formulas in place, we could relaunch the steps for constructing a PivotTable using Power Pivot and end up with the PivotTable pictured in **Figure 61**.

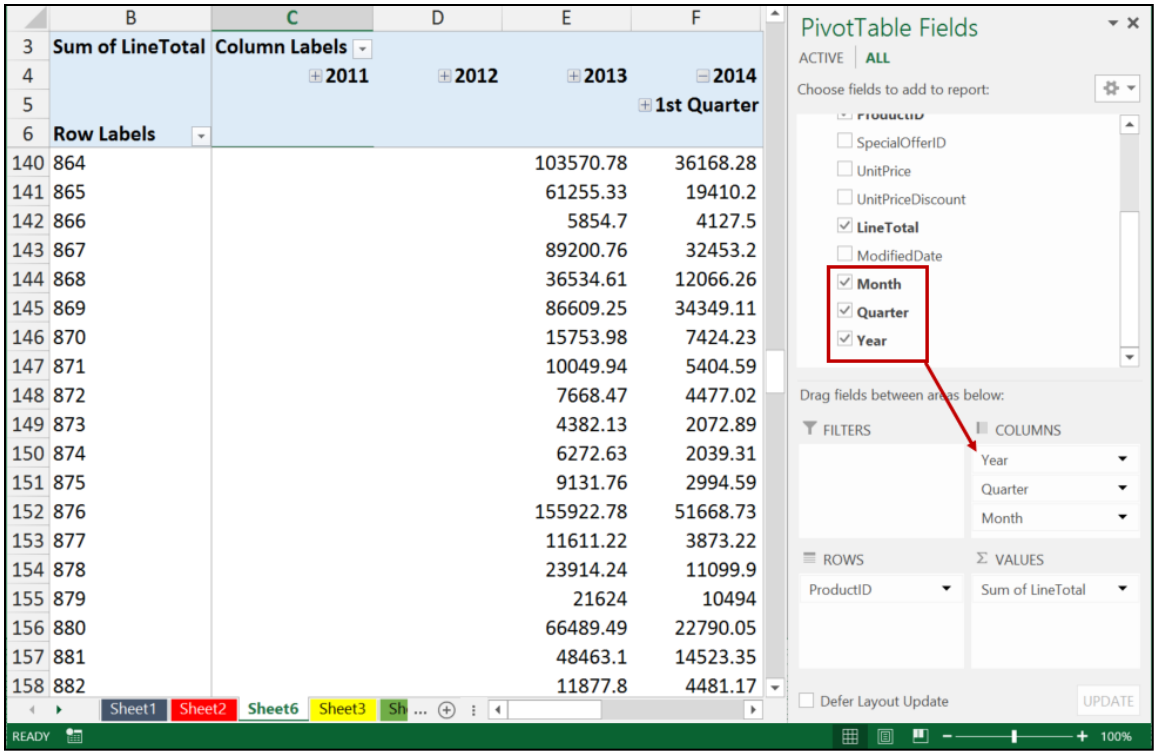


Figure 61 – Completed Power Pivot Generated from Power Pivot with "Pseudo" Groupings

Despite overcoming the grouping limitations imposed on PivotTables generated through Power Pivot, other differences exist between the PivotTable shown in Figure 61 and traditional PivotTables. For example, in the PivotTable in Figure 61, we cannot add Calculated Fields or Calculated Items. Further, if we drill down on any summarization in the PivotTable, Excel only shows the first 1,000 rows of detailed data by default.

On the positive side, we can convert the Power Pivot-based PivotTable into a series of formulas, allowing us to reposition and manipulate the data in ways we cannot do if we are working with a traditional PivotTable. Select **OLAP Tools** from the **PivotTable Tools Analyze** tab and click **Convert to Formulas** to do this. Upon doing so, Excel converts all the cells of the PivotTable to a series of formulas. These formulas use Excel's **CUBEVALUE** function to summarize the data based on the formula's criteria. For example, cell E140 in the PivotTable pictured in Figure 61 converts to the following.

=CUBEVALUE("ThisWorkbookDataModel",B3,$B140,E$4)

Once you convert the PivotTable to formulas, you can insert columns and rows and reposition the summarizing cells with virtually no restrictions.

Using Power Pivot to Create and Manage Data Models Feeding into Your Pivot Tables

A data model is a collection of related data that is the source of a report or analysis, including PivotTables or Power View visualizations, as described previously. One of the disadvantages of

traditional PivotTables is that they can only use one data source. For example, while the PivotTables in the preceding discussion might be helpful, they lack clarity because they do not identify products by product name. These PivotTables would be more useful if we translated some of the “coded” data into terminology readers may understand more easily. However, given our source data's arrangement, we cannot do this easily with traditional PivotTables because of the one data source limitation. With Power Pivot, that is not the case because you can build PivotTables from data models that simultaneously draw data from multiple data sources.

To use multiple data sources as the foundation of a PivotTable, begin in the Power Pivot window by selecting **Get External Data** and selecting a single data source. After importing that data source into Power Pivot, repeat the process for each additional data source you wish to be a part of the data model. For example, the **Data View** will appear after importing all your data sources into Power Pivot, as shown in **Figure 62**. Notice that each tab at the bottom of the window represents a different table. Each tab might originate from a separate database, Excel workbook, or other supported data source. In the example pictured, we selected nine tables for querying from the AdventureWorks Access database.

1. Person_Address
2. Person_CountryRegion
3. Person_StateProvince
4. Product_Product
5. Product_Category
6. Product_Subcategory
7. Sales_Customer
8. Sales_SalesOrderDetail
9. Sales_SalesOrderHeader

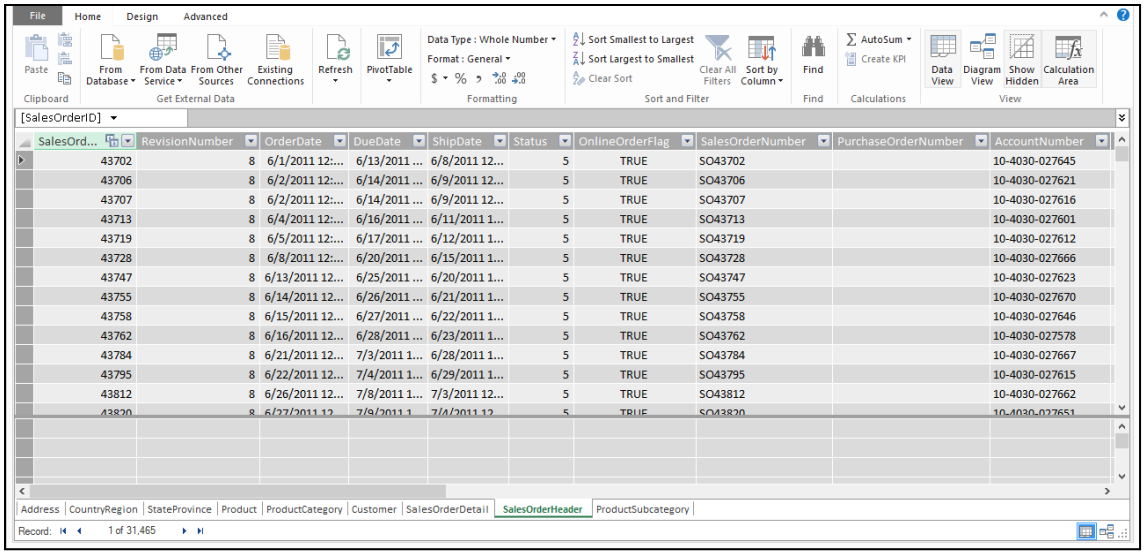


Figure 62 – Data View in Power Pivot Displaying Multiple Data Sources

In addition to querying data from an accounting software database, we might want to recreate the query presented in Figure 56. For example, we could use Power Query or Power Pivot to query and load the data to the data model, as shown in **Figure 63**.

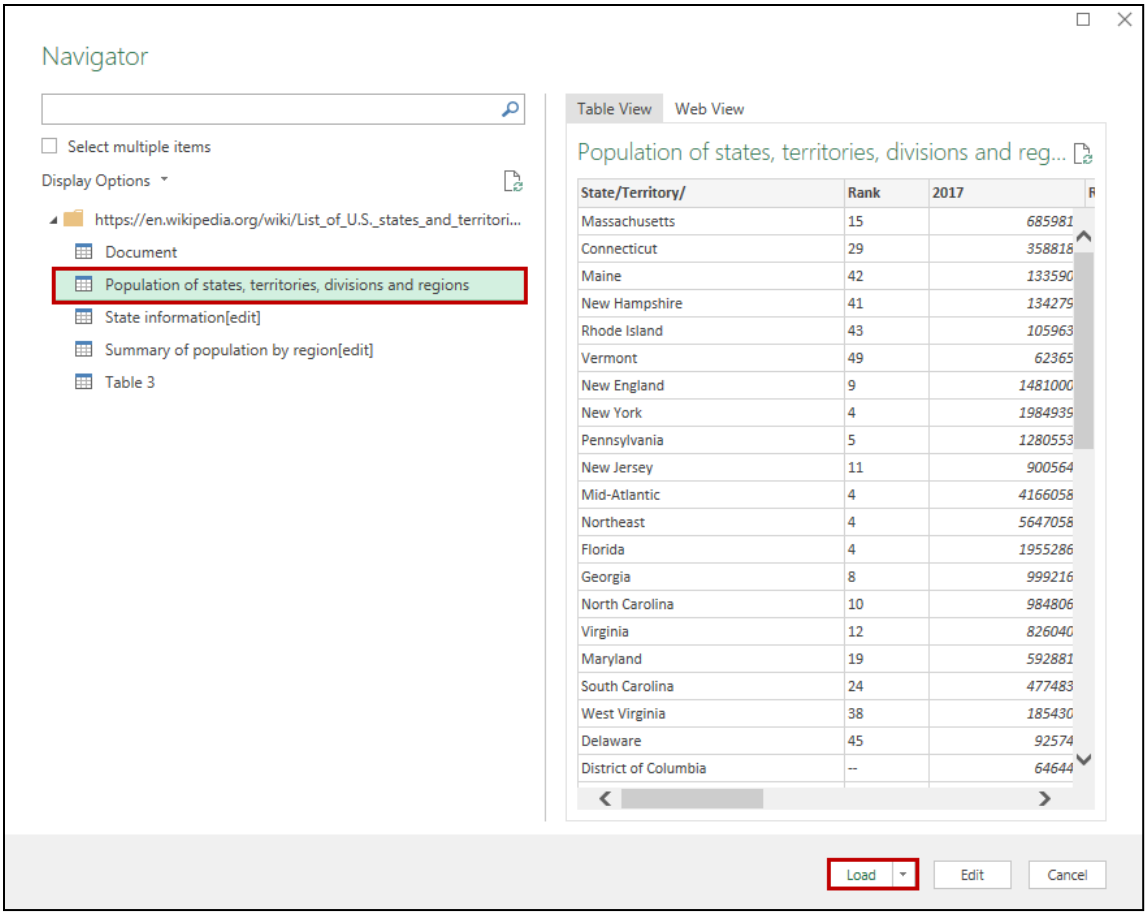


Figure 63 – Adding Power Query Data to a Data Model in Power Pivot

Once the query completes and returns the data to Excel, we can establish relationships between the different tables in the data model, as discussed below.

Defining Relationships in Power Pivot

Before building a PivotTable from the data model, ensure that appropriate relationships exist between each data model table. If you import multiple tables from the same database and relationships already exist between those tables in the database, Power Pivot typically recognizes and maintains them. However, if you import data from separate data files – for instance, multiple Excel workbooks or databases – you must establish relationships between the tables in Power Pivot before building your PivotTables.

Multiple methods exist for defining relationships, but perhaps the easiest is to open the **Diagram View** window in Power Pivot. Diagram View displays the tables in a visual environment, as its name implies. To open Diagram View, click **Diagram View** from the Ribbon's **Home** tab in Power Pivot. In the Diagram View pictured in **Figure 64**, ten tables from two data sources reside in the data model. Nine of the ten tables originated from the same SQL Server database, and relationships existed there. Therefore, Power Pivot recognizes and maintains the relationships

between those tables upon data import. However, we must relate the tenth table to the other nine for this information to be helpful.

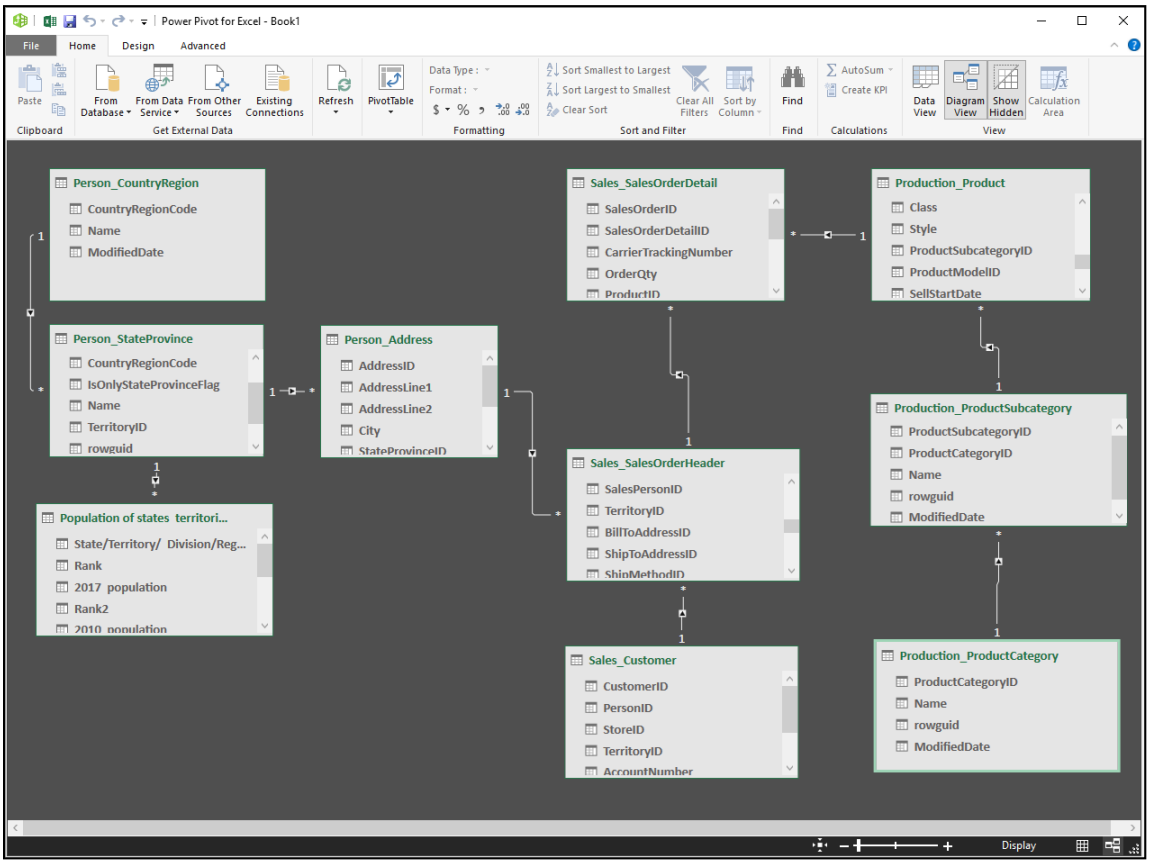


Figure 64 – Multiple Related Tables in a Power Pivot Data Model

To create relationships using Diagram View, identify a common field between an unrelated table and any other table in the data model. Then, click-and-drag on that field from the unrelated table and drop it onto the same field in the table with which you want to create the relationship, as shown in **Figure 65**. Repeat this process until you relate all tables in the data model.

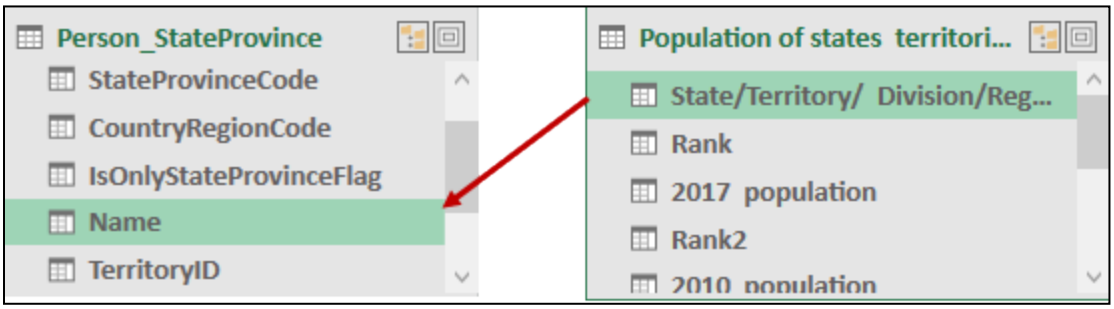
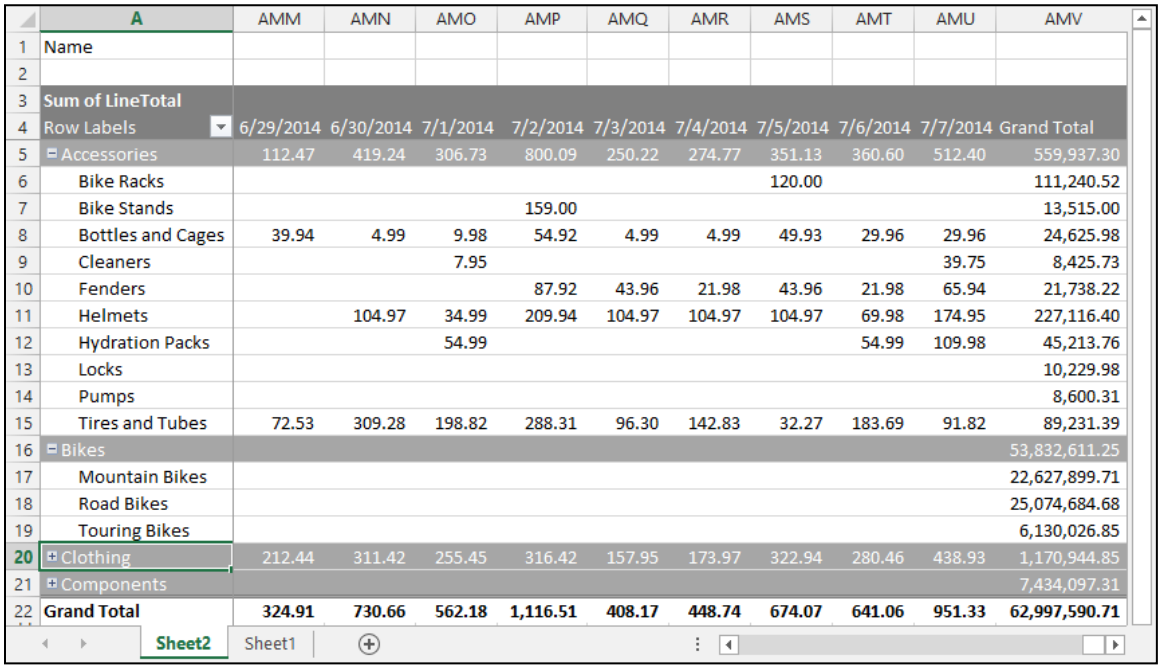


Figure 65 – Establishing a Relationship between Two Tables by Click-and-Drag Method

With the data model entirely related, build the PivotTable by clicking the **PivotTable** icon on the **Home** tab of the Power Pivot Ribbon, dragging and dropping the fields into the field list as you would any other PivotTable. The result may appear as shown in **Figure 66** (with some data hidden for presentation).



	A	AMM	AMN	AMO	AMP	AMQ	AMR	AMS	AMT	AMU	AMV
1	Name										
2											
3	Sum of LineTotal										
4	Row Labels	6/29/2014	6/30/2014	7/1/2014	7/2/2014	7/3/2014	7/4/2014	7/5/2014	7/6/2014	7/7/2014	Grand Total
5	Accessories	112.47	419.24	306.73	800.09	250.22	274.77	351.13	360.60	512.40	559,937.30
6	Bike Racks							120.00			111,240.52
7	Bike Stands				159.00						13,515.00
8	Bottles and Cages	39.94	4.99	9.98	54.92	4.99	4.99	49.93	29.96	29.96	24,625.98
9	Cleaners			7.95						39.75	8,425.73
10	Fenders				87.92	43.96	21.98	43.96	21.98	65.94	21,738.22
11	Helmets		104.97	34.99	209.94	104.97	104.97	104.97	69.98	174.95	227,116.40
12	Hydration Packs			54.99					54.99	109.98	45,213.76
13	Locks										10,229.98
14	Pumps										8,600.31
15	Tires and Tubes	72.53	309.28	198.82	288.31	96.30	142.83	32.27	183.69	91.82	89,231.39
16	Bikes										53,832,611.25
17	Mountain Bikes										22,627,899.71
18	Road Bikes										25,074,684.68
19	Touring Bikes										6,130,026.85
20	Clothing	212.44	311.42	255.45	316.42	157.95	173.97	322.94	280.46	438.93	1,170,944.85
21	Components										7,434,097.31
22	**Grand Total**	**324.91**	**730.66**	**562.18**	**1,116.51**	**408.17**	**448.74**	**674.07**	**641.06**	**951.33**	**62,997,590.71**

Figure 66 – PivotTable Constructed from Multiple Data Sources

Creating Calculated Columns in Power Pivot Data Models

Recalling our first example of using Power Pivot to build a PivotTable, we could not group the data by months, quarters, or years because those fields did not exist in the source data. To solve this problem, we modified the source data. We could have used DAX to address this issue by adding calculated columns to the source data once we imported it into Power Pivot. For example, as shown in **Figure 67**, you can use the **MONTH**, **YEAR**, and **IF** functions to solve this problem. Technically, each function is a Data Analysis Expression (DAX) function in this context. You can also use DAX functions when creating calculated columns. There is additional coverage of DAX functions later in this course.

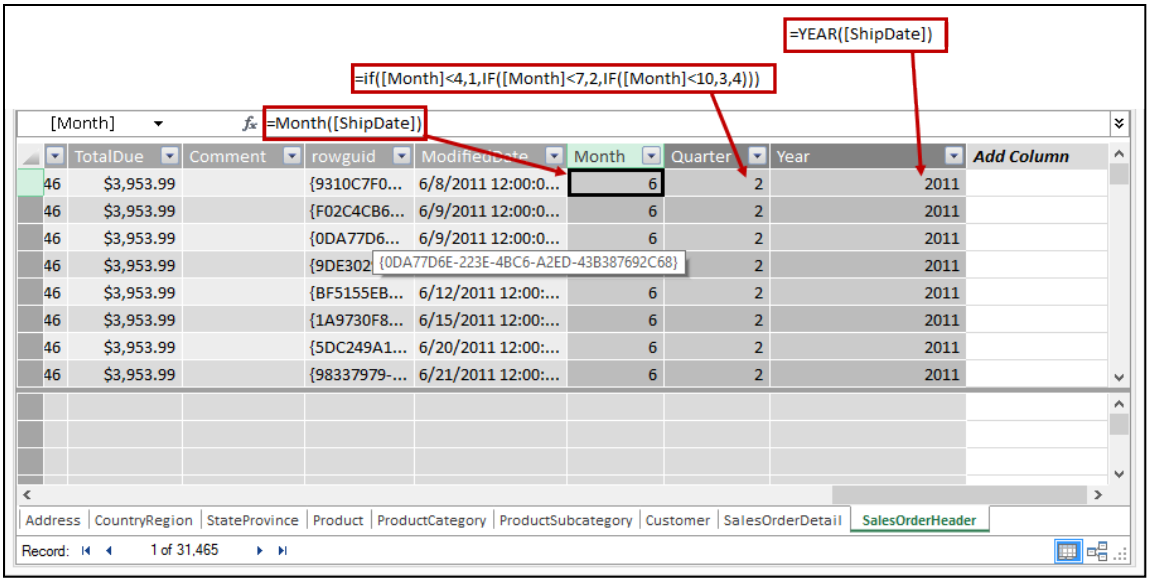


Figure 67 - Creating Calculated Columns in the Data Model with DAX Expressions

More Visualizations with 3D Maps

3D Maps (named **Power Maps** before Excel 2016) is a visualization tool that integrates with Excel to allow end-users to create three-dimensional visualizations of data to expose insights that might not be visible in traditional two-dimensional reporting environments. 3D Maps enables building visualizations based on an Excel table or data model. **Figure 68** provides a sample of the 3D Maps user interface.

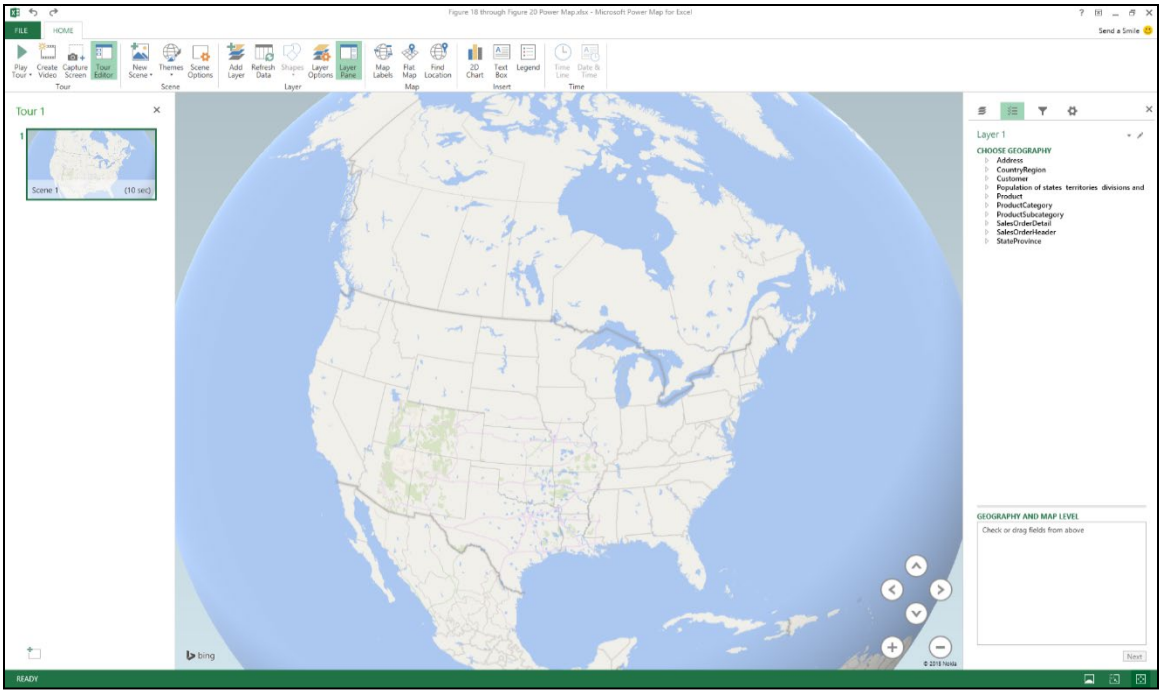


Figure 68 – 3D Maps User Interface

To use **3D Maps**, ensure your visualization's data resides in Excel as a table or a data model. We will continue with the data model used in previous examples for demonstration purposes. First, click **Map** from the Ribbon's **Insert** tab to open the 3D Maps user interface. Next, select the field/fields you want to serve as geographic and map-level identifiers from the task pane on the right. In this example, we will choose the **StateProvinceCode** field from the **StateProvince** table and map it to **State/Province** in 3D Maps, as shown in **Figure 69**. Upon selecting this mapping, click **Next** to continue.

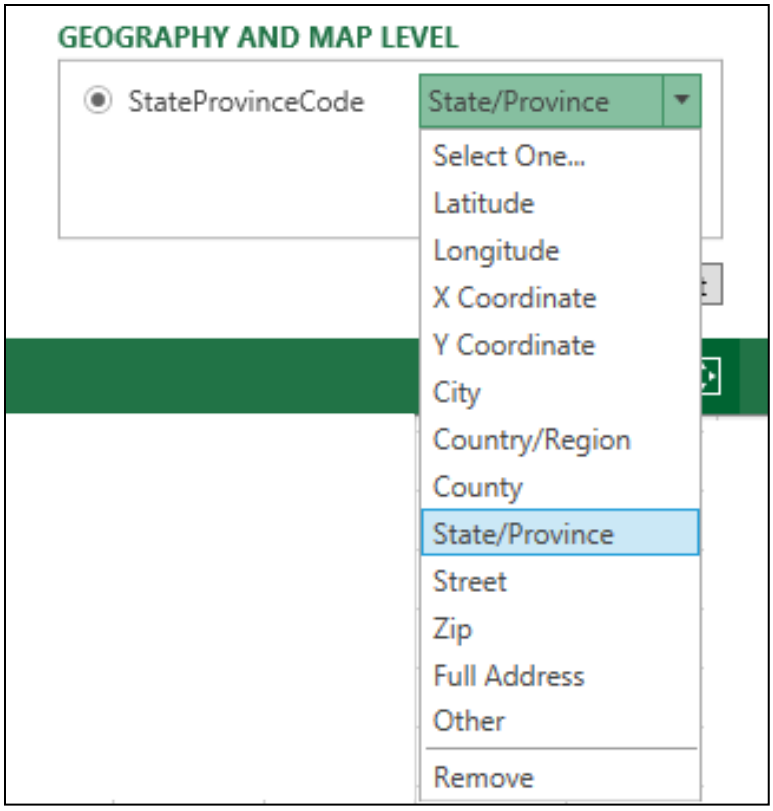


Figure 69 – Mapping Fields in 3D Maps

Having established the geo-coding, add the **2010 pop** field from the **Population of states territories divisions and regions – Summary of population** table to the **Height** category in 3D Maps. When doing so, change the chart type to a **clustered column chart**. Finally, add the **SubTotal** field from the **SalesOrderHeader** table to the Height category. Upon doing so, focus 3D Maps on seeing a map of the United States. Following the steps outlined above, **Figure 70** presents an example of a completed visualization created using 3D Maps.

Notice that 3D Maps allows you to identify which states a company's sales results are acceptable relative to the population's size. Further, 3D Maps helps you understand which states offer significant growth opportunities proportional to the population. Also, recall the nature of the data that feeds 3D Maps. Some data resides in the company's accounting software database and links to the data model via Power Pivot. On the other hand, some data is web-based and connects to the data model via Power Query. We created a report pulling data from multiple disparate locations to analyze sales results in just a few minutes and keystrokes.

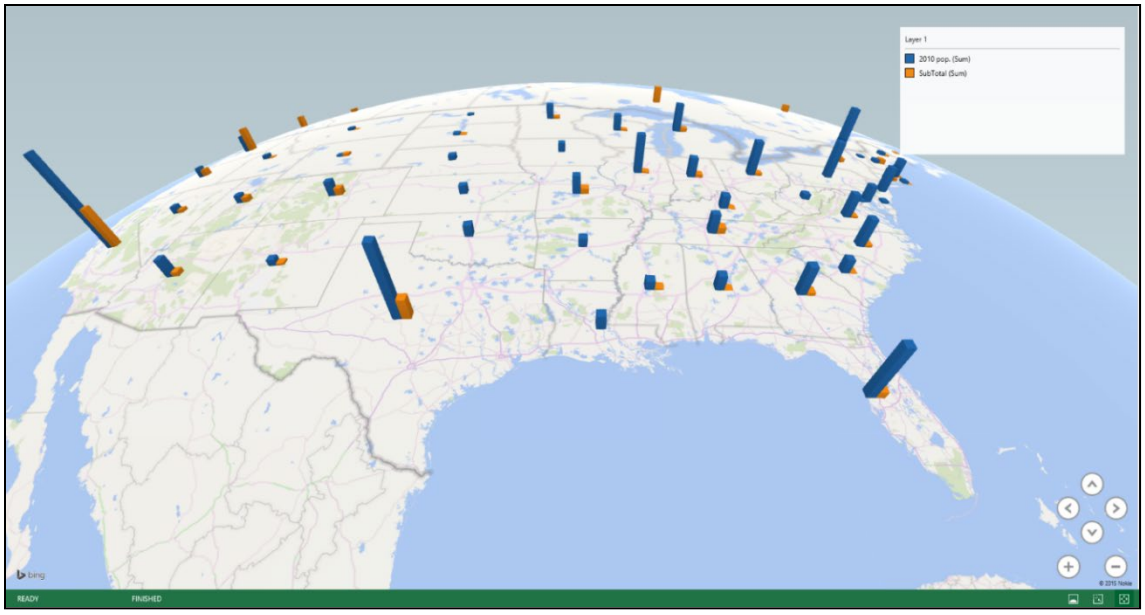


Figure 70 – Completed 3D Maps in Excel

Some might find the 3D Maps visualization easier to understand if it was a pie chart. For example, if we change the map chart type to a pie chart, we can glean relatively good market penetration in the western portion of the United States but little presence in the Northeast.

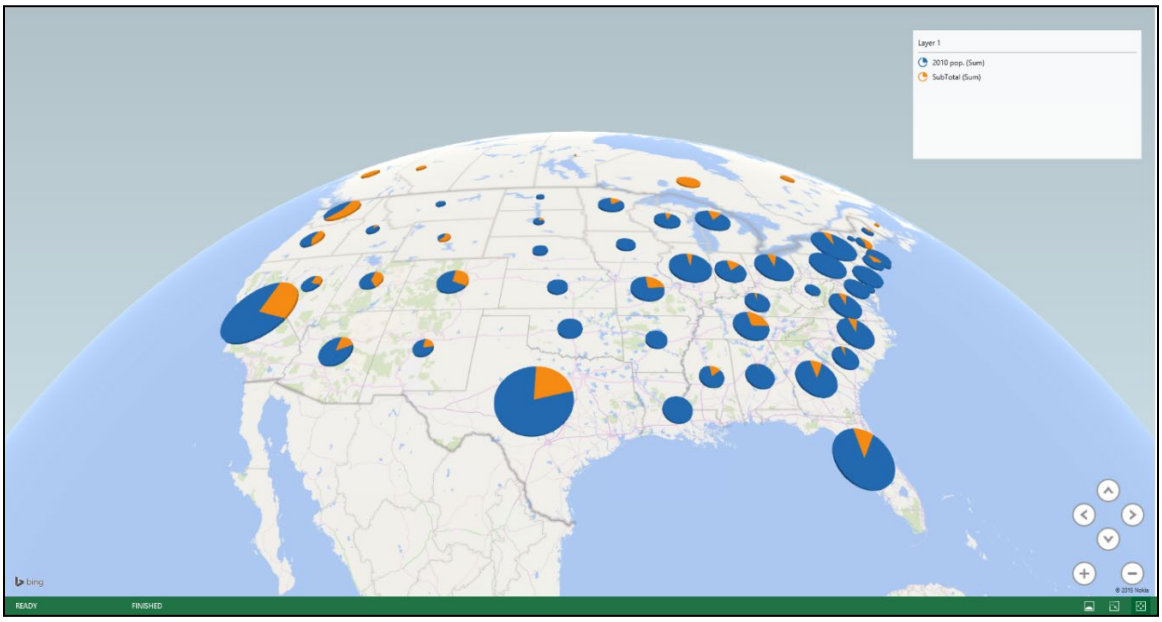


Figure 71 – 3D Maps Converted into a Pie Chart

Once you complete your map, you can add other elements, including multiple layers and animations. You can even save 3D Maps as videos that you can play outside of Excel – for example, in a PowerPoint presentation to communicate an important message.

?	What are some of the potential strengths of using the add-ins available for Excel as your primary business intelligence tool? Conversely, what are some of the potential drawbacks of this approach?

Working with Power BI Tools

Chapter 4

Microsoft has created an alternative environment for creating user-interactive dashboards without Excel. The free Power BI Desktop application allows business professionals to create stunning reports using quick and easy drag-and-drop processes similar to Power View but without using Excel. If you have an older version of Office, use a Mac, or have large datasets that require the additional memory space of a 64-bit application, the Power BI environment is for you. It has more than twice the data visualizations available in Power View, connects to many more external data sources, includes all the data connection and transformation capabilities of Power Query, and allows completed dashboards to be shared or published to Power BI.com or Power BI Server. And speaking of these two additional platforms, they facilitate quick and easy dashboard sharing and collaboration. Further, they enable you to secure your dashboards so that only authorized team members can view content. When viewed as a holistic solution, Power BI leverages your knowledge of Excel and extends it in ways you might not have thought possible.

Learning Objectives

Upon completing this chapter, you should be able to:

- Describe the advantages of working with Power BI platforms and list the relevant limitations associated with using Power View to produce similar reports;
- Identify how to acquire Power BI;
- Explain data models and how to create and work with them in Power BI Desktop;
- List several ways in which data can be displayed and presented on a Power BI report; and
- Describe how Power BI reports present information better than traditional reports.

Microsoft Power BI

Power BI is a set of Microsoft tools that you can use with or without Excel to quickly and easily summarize large volumes of data. With Power BI, you can create rich, interactive visualizations of your data, drill down on results to see supporting details, and share the data in dashboards. Power BI can analyze all data types – financial, sales, manufacturing, human resources, engineering, operations, etc. In addition, Power BI's flexibility allows users to create Key Performance Indicators (KPIs) that can be updated quickly and filtered to spot trends over time, regions, budgets, or goals.

What Are the Components of Power BI?

Microsoft has created *two* parallel development environments for business intelligence. As discussed in the preceding chapters, one uses Excel for data manipulation and report development. The more modern approach uses the Power BI platform to facilitate business intelligence. **Figure 72** provides a visual representation of these environments and their components.

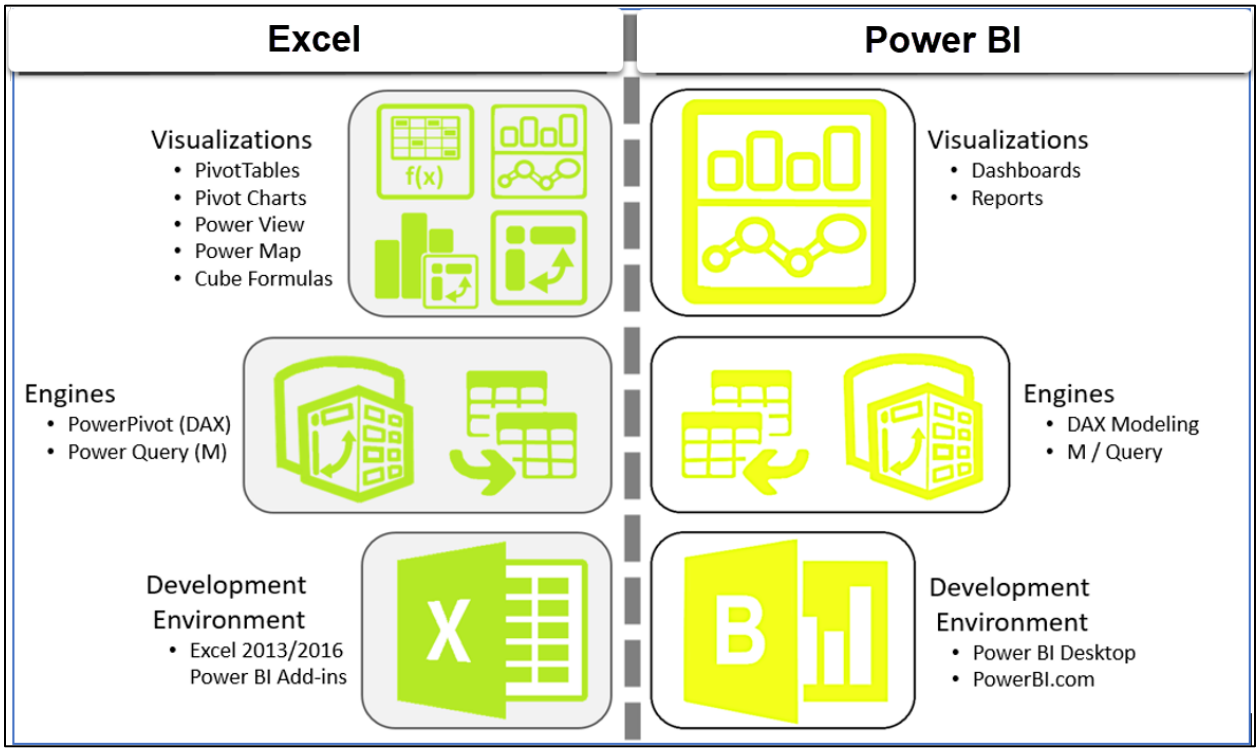


Figure 72 – Power BI Providing Two Parallel Development Environments [7]

The first environment, pictured on the left in Figure 72, works with Excel. More specifically, it works with Excel acquired as part of a Microsoft 365/Office 365 ProPlus subscription, as part of a Microsoft Office 2013 and newer Professional Plus license, or through purchasing the stand-alone version of Excel 2013 and newer. As discussed in the previous chapter, with these versions of Excel, users have access to add-ins that can, in some cases, provide all the business intelligence reporting required. However, using Power BI adds functionality not otherwise available with Excel. Further, with a Power BI subscription or access to the server-based platform, users can view and share reports and dashboards, access BI reports and dashboards on mobile devices, and interact with reports using natural language queries. The second development environment, pictured on the right in Figure 72, can take advantage of Cloud-based data manipulation and dashboard reporting tools of Power BI.com, Power BI Server, and the free companion application,

[7] Figure adapted from one originally published at *powerpivotpro.com* by Rob Collie, July 2015.

Power BI Desktop. Power BI Desktop overcomes many of the issues and limitations of using Excel as a BI tool, including memory resource constraints associated with 32-bit Excel and vast datasets, difficulty in publishing Excel workbooks to mobile devices, and the long development cycle associated with delivering new features to Excel in a rapidly changing marketplace. Further, in this environment, you do not need an Excel license. Freedom from Excel allows those using a Mac to use Power BI's Cloud-based tools.

Power BI Desktop provides functionality that is almost identical to an Excel-based BI environment's primary components – Power Pivot, Power Query, and DAX formulas. These parallel development environments provide analysts, business professionals, and information consultants with exceptional flexibility in meeting information consumers' needs.

Power BI Versions and Pricing

Microsoft licenses Power BI for use in three versions: 1) Power BI Desktop, which is free, 2) Power BI Pro, and Power BI Server. Power BI Pro is a web-based service accessible through supported browsers – including Microsoft Edge, Chrome, Safari, or Firefox – for creating, viewing, and sharing dashboards and reports. A Power BI license is required to author ***and*** consume Power BI content.

All three versions provide all the core functionality needed by most users. A free Power BI license is sufficient if your report links to data in Excel workbooks, Power BI Desktop, or CSV files. A Pro or Server license is required to create or view a Power BI dashboard or report in the following situations.

- Data results from a direct query dataset, such as tabular data from SQL Server Analysis Services, Azure SQL Database, Azure SQL Data Warehouse, or Apache Spark for HDInsight.
- A dataset connects to on-premises data using a Data Management Gateway.
- You created your dashboard or report from a custom organizational content pack.
- A dashboard, report, or dataset resides in a group workspace.
- A dashboard receives data from a streaming source at a rate greater than 10K rows per hour.

Table 1 identifies the functionality offered by each version.

	Power BI Desktop	Power BI Pro and Server
General		
Data capacity limit	1 GB/user	10 GB/user
Create view and share your personal dashboards and reports with other Power BI users	√	√
Author content with the Power BI Desktop	√	√
Explore data with Natural Language Queries	√	√
Access your dashboards on mobile devices using native apps for iOS, Windows, and Android	√	√
Consume curated content packs for services like Dynamics Salesforce and Google Analytics	√	√
Import data and reports from Excel CSV and Power BI Desktop files	√	√
Data Refresh		
Consume content that is scheduled to refresh	Daily	Hourly
Consume streaming data in your dashboards and reports	10K rows/hour	1M rows/ hour
Consume live data sources with full interactivity		√
Access on-premises data using the Data Connectivity Gateways (Personal and Data Management)		√

Collaboration		
Collaborate with your team using Office 365 Groups in Power BI		√
Create, publish, and view organizational content packs		√
Manage access control and sharing through Active Directory groups		√
Share data queries through the Data Catalog		√

Table 1 - Comparing Power BI to Power BI Pro

Power BI Desktop is a free application for developing Power BI datasets and reports on a local PC. A subscription is not required to use the desktop application but is required to publish or share a dashboard or report on Power BI.com. Power BI Desktop files (*.pbix*) can be saved and exchanged without a license. The only requirement is that all users install the Power BI Desktop application.

Additionally, remember that reports created with the desktop application may result in large file sizes. Sharing files through a network location will probably work fine in most cases, but large file sizes will likely prohibit sharing via email. Power BI Desktop is a free download available from ***powerBI.microsoft.com***.

Power BI Desktop Interface

Refer to **Figure 73** as we acquaint ourselves with the application's interface. The familiar Ribbon from which users can get or enter data or queries is at the top of the window. Along the left is a navigation bar containing buttons for the three work views – **Report**, **Data**, and **Model**.

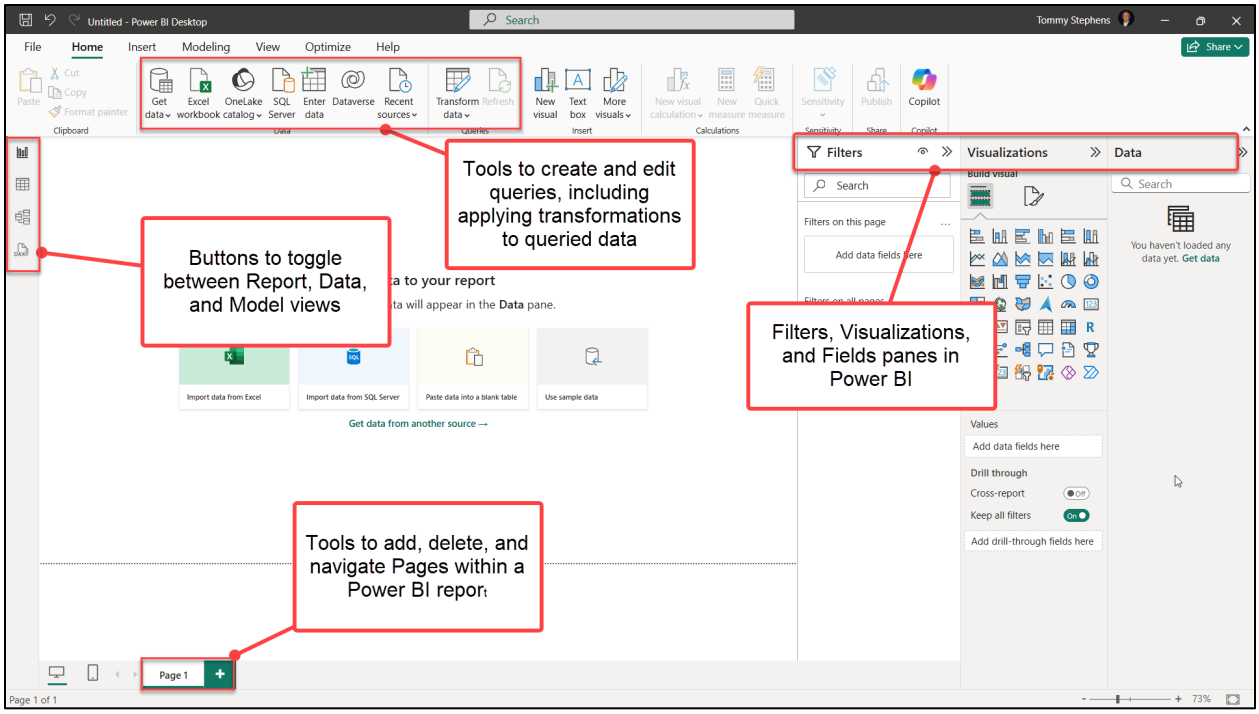


Figure 73 – Power BI Desktop Graphical User Interface

On the right are the **Fields** list and the **Visualizations** pane. The Visualizations pane contains a gallery from which to apply and format data visualizations (tiles) and the **Filters** area to filter reports and report elements. You can dock the Fields list and Visualizations pane along the right margin to expand the main viewing area. Buttons are available at the bottom to add or navigate report pages.

Microsoft has updated Power BI Desktop almost every month since its release in July 2015, adding hundreds of improvements and features since its initial release. Currently, the application connects to over 120 data types and provides twenty-five different data visualizations. From these numbers, it is clear that Microsoft's innovation and development emphasis is on Power BI. Because of this application's rapid and ongoing development, the desktop application interface and functionality may have changed from that described in these materials.

Building the Dataset

To begin building a new report, click **Get Data, More…** on the **Home** tab. In the **Get Data** window, select **Database** in the navigation pane on the left and then choose **Access Database**[8] in the list on the right. Next, click **Connect,** browse to the **ContosoSales** database, and click **Open**.

[8] If you are using Power BI Desktop 64-bit with Microsoft Office 32-bit, you must install the 64-bit Microsoft Access 2010 Redistributable Data Engine from *https://www.microsoft.com/en-us/download/details.aspx?id=13255*.

In the **Navigator** window, check the desired tables in the list. In this case, check **DimChannel**, **DimDate**, **DimProduct**, **DimProductSubcategory**, **DimPromotion**, and **FactSales**, as shown in **Figure 74**. Users can add or remove columns and formats, or filter the data imported from individual tables. First, select the table of interest and click **Transform Data** to open the Query Editor. Then, in this case, click **Load** to import the data into the Data Model. The imported tables will now be visible in the **Data** view.

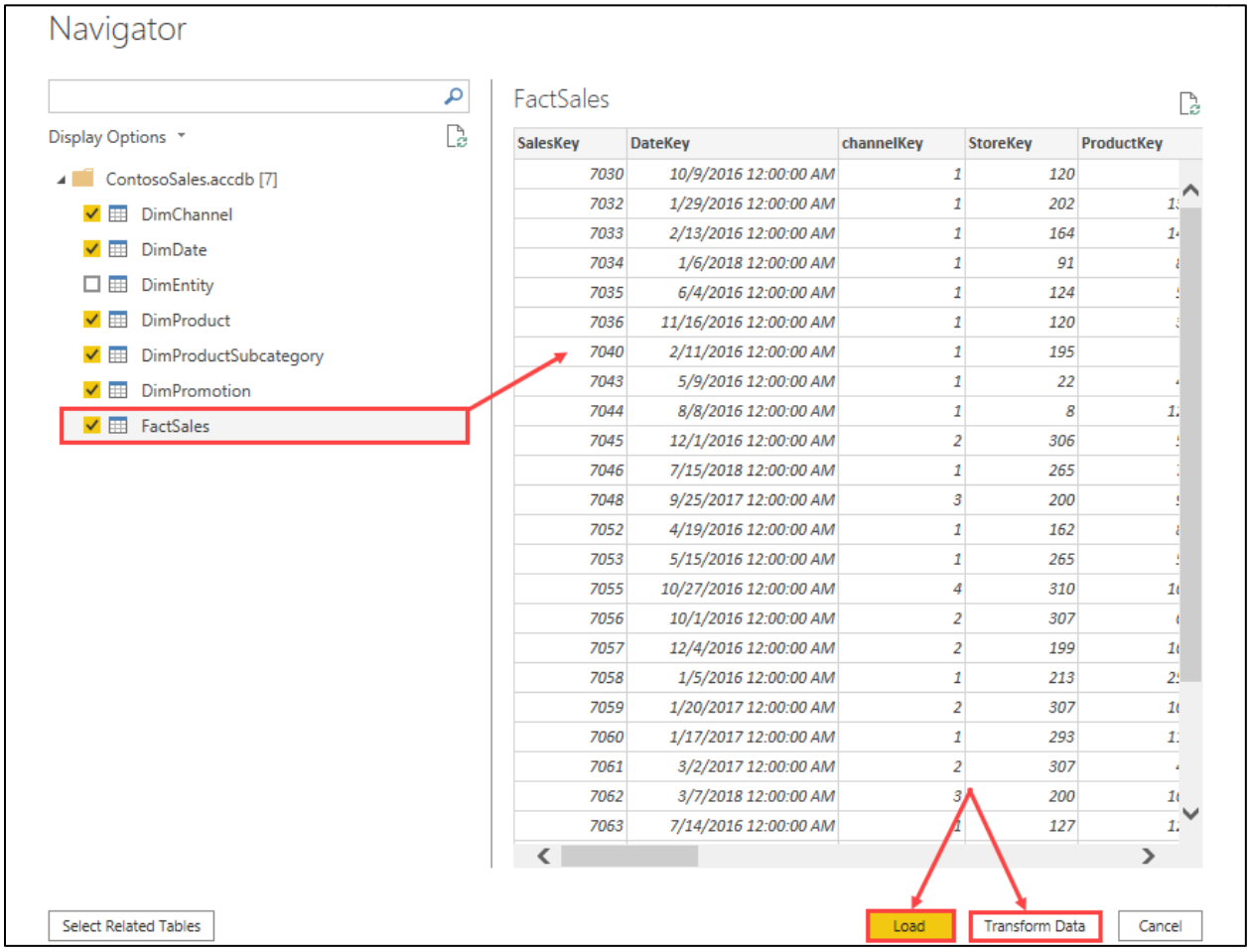


Figure 74 – Importing Tables into the Power BI Desktop Data Model

Continue by importing data from two Excel workbooks – **DimStores** and **DimGeography**. These contain information about the stores and geographical locations in which the company earns sales revenue. Click **Get Data**, **Excel**, and browse to and select the **DimStores** workbook. Then click **Open** from the Ribbon. Select the **DimStores** table in the **Navigator** window and click **Load**. Repeat the process to import the **DimGeography** workbook.

Defining Relationships Using Design View

Power BI Desktop attempts to recognize and maintain those relationships if you import multiple related tables from the same data source. Power BI Desktop also attempts to identify and relate

tables from *different* data sources by matching column names and data types. In cases where relationships are not recognized or maintained, you may establish them manually in the **Relationships** view, shown in **Figure 75**, using a simple click-and-drag process.

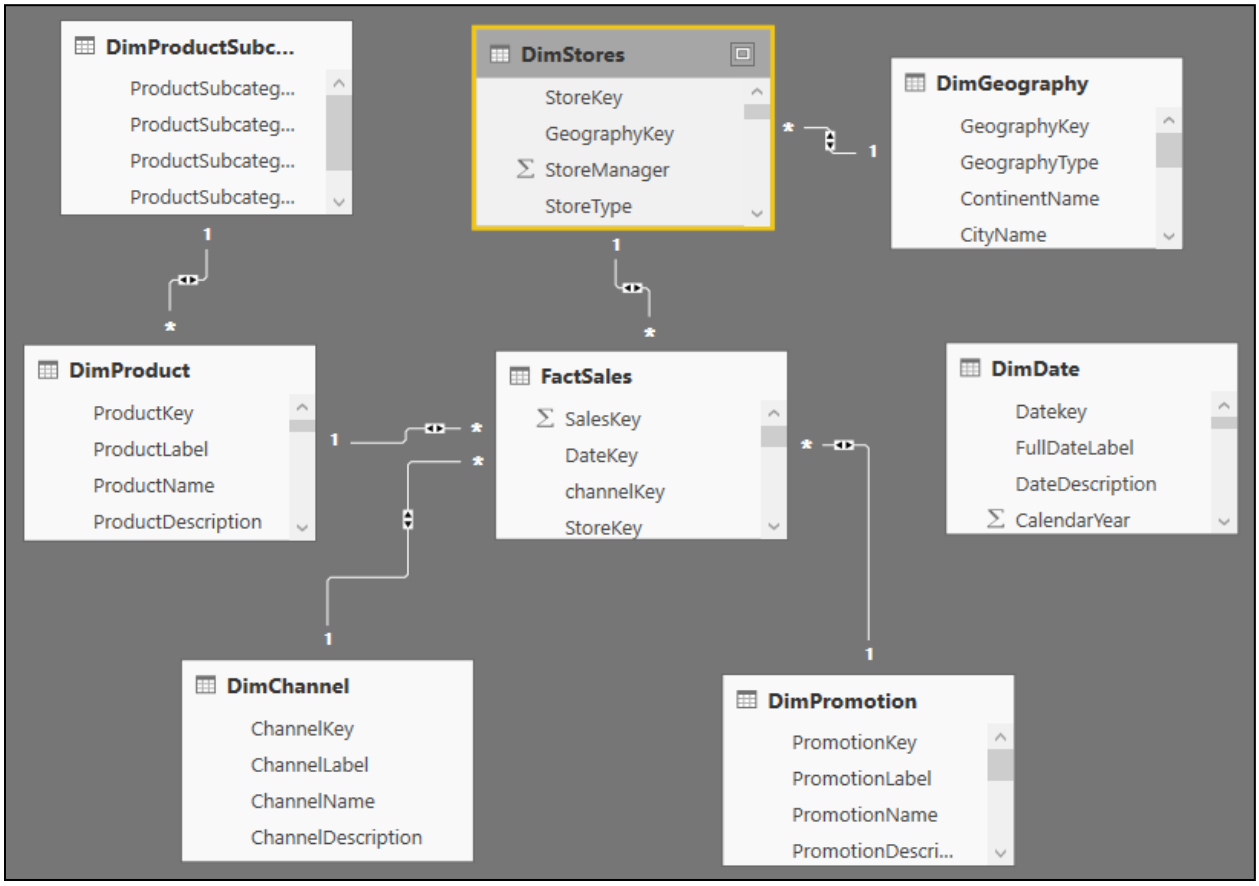


Figure 75 – The Relationships View in Power BI Desktop

In this case, the Power BI Desktop Data Model recognized and established the relationships between the two imported Excel workbooks and the tables imported from Access. However, the Data Model did not recognize the relationship between the **FactSales** table and the **DimDate** table. To create a relationship manually, drag **Datekey** from the **DimDate** table and drop it on **DateKey** in the **FactSales** table. Once you establish a relationship, double-click a join line to examine or edit the relationship. Note that the process creates an *inner join* between the related tables by default. Also, note that the diagram in the **Relationships** view indicates the characteristics of the relationship (one-to-one, one-to-many, or many-to-one) and the cross-filter direction.

?	How does building a data model using Power BI resemble creating an Excel data model? How does it differ?

Building Matrix Reports

Once you import your data and relate the tables, you can create a Power BI Desktop report. A report consists of a dataset and one or more pages. For example, our first page will contain a **Matrix** visualization of sales revenue summarized by state. Note that a Matrix provides similar functionality compared to an Excel-based PivotTable.

In the **Fields** list, expand the **DimGeography** table and check **StateProvinceName** to add a list of states and provinces to the canvas. Power BI adds it to the canvas as a two-dimensional **Map** tile by default. With the map tile selected, click **Matrix** in the visualization gallery in the **Visualization** pane to convert it to a matrix. Note that the list contains states and provinces from around the world. To filter tiles on the page, drag the **RegionCountryName** field to the **Page level filters** box in the **Filters** area of the **Visualizations** pane. Expand the list of regions and countries, and then check **United States** to filter the list of states and provinces to just those in the United States, as shown in **Figure 76**. Note that you may use fields as visual (individual tiles), page (affecting a specific page), or report (affecting all pages in a report) filters. Make sure to drop field filters into the appropriate filter box.

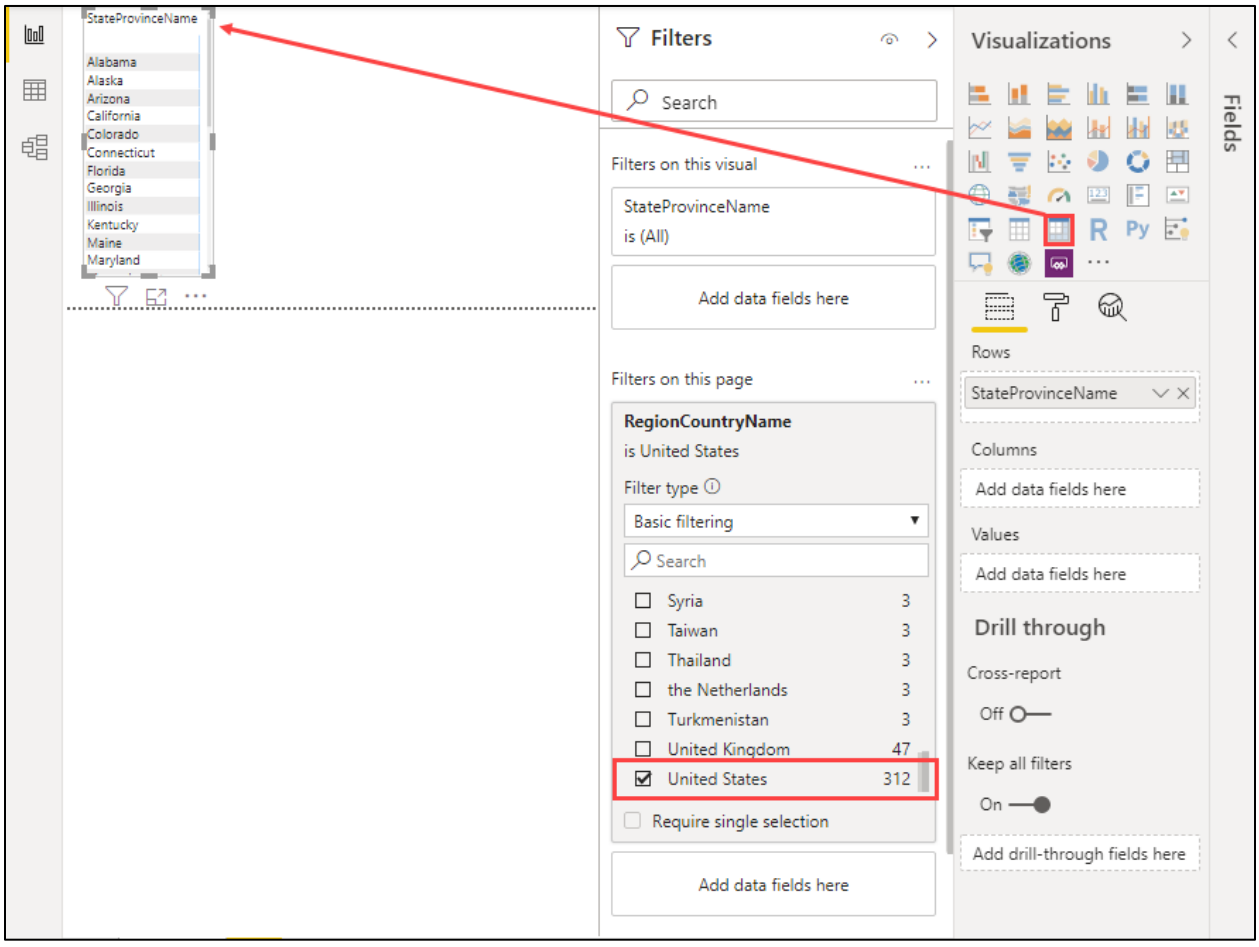


Figure 76 – Adding and Filtering a Field in a Report as a Matrix Tile

Because we changed the tile to a Matrix, the **Visualizations** pane includes **Rows**, **Columns**, and **Values** drop areas similar to the drop areas in the PivotTable Field List in Excel. Drag the **Sales Amount** field from the **FactSales** table to the **Values** drop area, the **FiscalYearLabel** field from the **DimDate** table to the **Columns** drop area, and the **StoreName** field from the **DimStores** table to the **Rows** drop area below **StateProvinceName**. Click and drag the tile border to resize the tile to display all the columns. These actions complete the base Matrix visualization, which resembles and functions similarly to an Excel PivotTable report, but without drill-down capability.

Now, let's complete the report. If the **SalesAmount** is incorrectly formatted, click its field name in the **Fields** list. Next, on the **Column Tools** contextual tab of the Ribbon, click the **Dollar Sign ($)** to apply the accounting number format with dollar signs and use the spinner control to enter **2** decimal places. Finally, select **Sum** as the **Default Summarization**. If the columns in the table are not wide enough to display all the data, click and drag in the column headings area to widen or narrow the columns accordingly. Note that there are no visual cues regarding where the column boundaries reside. Move your mouse in the column heading area until the double-arrow cursor appears, and then click and drag the column border to the desired position. Note also that the value column widths work in unison. Adjusting the width of one value column causes all the value columns to change to the same width. To sort on a column, click its heading, with each additional click reversing the sort's order.

Select the tile and click the **Format** button in the **Visualizations** pane to make the report more aesthetically pleasing, as shown in **Figure 77**. Open the **General** settings group by clicking the down arrow to the left of the label. Here, users can adjust the tile's **Text Size** and control whether **Total rows** and **Total columns** display in the tile. Ensure that the **Total row** and **Total Column** features are **ON**. Close the **General** group. Turn **ON** the **Title** and then open the **Title** settings group. Type a title for the tile – "**Sales by Store within State**" – in the **Title Text** box. Next, adjust the **Text Size** to **18pt** by dragging the slider. Note the settings for **Font color**, **Background color**, and label **Alignment**. Set the title **Background color** to **Green 4** and the **Font color** to **White**. Close the **Title** group. Turn **ON** the **Background** and then open the **Background** settings group. Set the background **Color** to **Light Gray 1**. Then, close the **Background** group. Turn **ON** the **Border**.

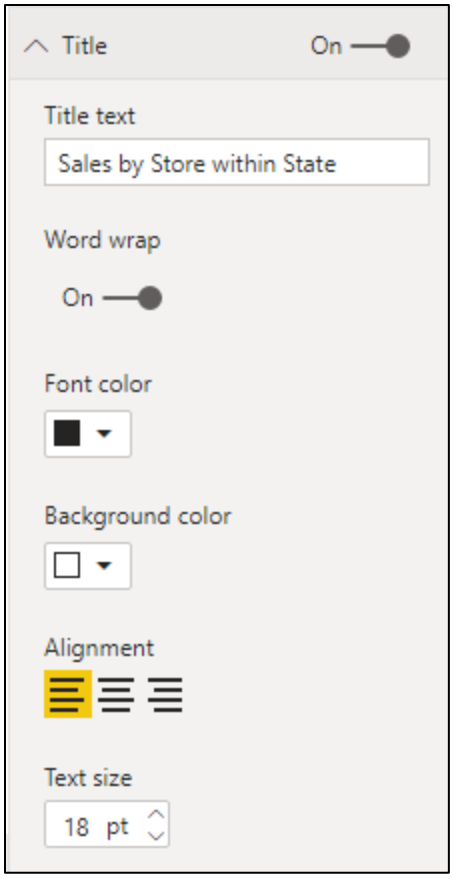
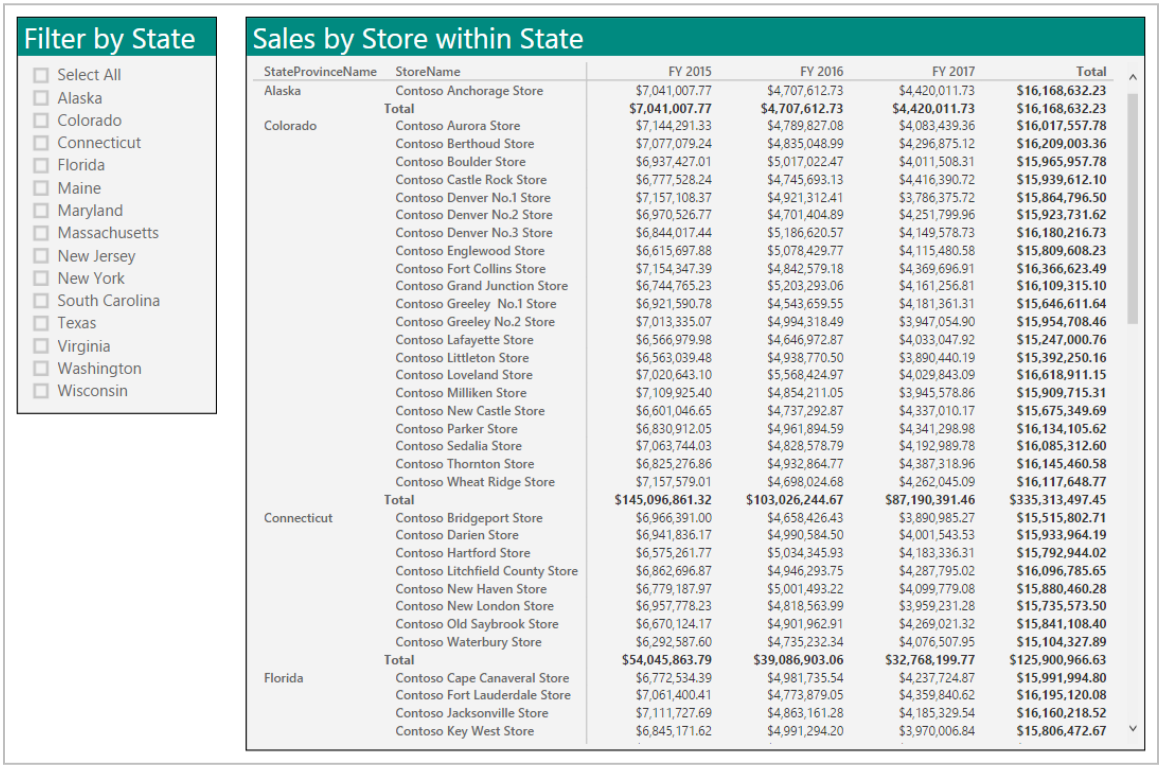


Figure 77 – Formatting a Tile from the Visualizations Pane

Now, let us add a Slicer to make the report interactive. Drag the **StateProvinceName** field from the **DimGeography** table to an empty area on the canvas. Power BI will add it to the canvas as a two-dimensional map tile by default. With the tile selected, click **Slicer** in the Visualizations pane gallery to convert the tile to a Slicer. Note that a complete list of US states displays in the slicer. Filter the slicer to contain only those states that generated sales revenue for the company. Drag the **SalesAmount** field from the **FactSales** table to the **Page Level Filters** box in the **Filters** area of the **Visualizations** pane. Choose **Show items when the value: is not blank** so that the filter list only displays states with sales.

You can apply formatting to the **Slicer** through the **Visualizations** pane, similarly to how you formatted the **Matrix** tile. Begin by opening the **Selection controls** group and turning **ON** the **Select All** and **Single Select** controls. Next, close the **Selection controls** group. Follow that by turning **OFF** the **Header and** turning **ON** the **Title**. Next, open the **Title** settings group. Type a title for the tile, "**Filter by State**," in the **Title Text** box and adjust the **Text Size** to **18pt** by dragging the slider. Note the settings for **Font color**, **Background color**, and label **Alignment**. Set the title **Background color** to **Green 4** and the **Font color** to **White**. Close the **Title** group. Turn **ON** the **Background** and then open the **Background** settings group. Next, set the background **Color** to **Light Gray 1**. Finally, close the **Background** group and turn **ON** the **Border**.

Note the **Arrange**, **Alignment**, and **Distribute** buttons on the Ribbon's **Format** contextual tab. You can use these controls for managing tiles' locations in a report. First, use **CTRL + Click** to select both tiles. Then click **Format, Align,** and **Align Top** from the Ribbon to properly align the tiles. Finally, double-click the sheet tab and rename it "**Sales by State Matrix**." The completed report page should resemble the one displayed in **Figure 78**.



StateProvinceName	StoreName	FY 2015	FY 2016	FY 2017	Total
Alaska	Contoso Anchorage Store	$7,041,007.77	$4,707,612.73	$4,420,011.73	$16,168,632.23
	Total	**$7,041,007.77**	**$4,707,612.73**	**$4,420,011.73**	**$16,168,632.23**
Colorado	Contoso Aurora Store	$7,144,291.33	$4,789,827.08	$4,083,439.36	$16,017,557.78
	Contoso Berthoud Store	$7,077,079.24	$4,835,048.99	$4,296,875.12	$16,209,003.36
	Contoso Boulder Store	$6,937,427.01	$5,017,022.47	$4,011,508.31	$15,965,957.78
	Contoso Castle Rock Store	$6,777,528.24	$4,745,693.13	$4,416,390.72	$15,939,612.10
	Contoso Denver No.1 Store	$7,157,108.37	$4,921,312.41	$3,786,375.72	$15,864,796.50
	Contoso Denver No.2 Store	$6,970,526.77	$4,701,404.89	$4,251,799.96	$15,923,731.62
	Contoso Denver No.3 Store	$6,844,017.44	$5,186,620.57	$4,149,578.73	$16,180,216.73
	Contoso Englewood Store	$6,615,697.88	$5,078,429.77	$4,115,480.58	$15,809,608.23
	Contoso Fort Collins Store	$7,154,347.39	$4,842,579.18	$4,369,696.91	$16,366,623.49
	Contoso Grand Junction Store	$6,744,765.23	$5,203,293.06	$4,161,256.81	$16,109,315.10
	Contoso Greeley No.1 Store	$6,921,590.78	$4,543,659.55	$4,181,361.31	$15,646,611.64
	Contoso Greeley No.2 Store	$7,013,335.07	$4,994,318.49	$3,947,054.90	$15,954,708.46
	Contoso Lafayette Store	$6,566,979.98	$4,646,972.87	$4,033,047.92	$15,247,000.76
	Contoso Littleton Store	$6,563,039.48	$4,938,770.50	$3,890,440.19	$15,392,250.16
	Contoso Loveland Store	$7,020,643.10	$5,568,424.97	$4,029,843.09	$16,618,911.15
	Contoso Milliken Store	$7,109,925.40	$4,854,211.05	$3,945,578.86	$15,909,715.31
	Contoso New Castle Store	$6,601,046.65	$4,737,292.87	$4,337,010.17	$15,675,349.69
	Contoso Parker Store	$6,830,912.05	$4,961,894.59	$4,341,298.98	$16,134,105.62
	Contoso Sedalia Store	$7,063,744.03	$4,828,578.79	$4,192,989.78	$16,085,312.60
	Contoso Thornton Store	$6,825,276.86	$4,932,864.77	$4,387,318.96	$16,145,460.58
	Contoso Wheat Ridge Store	$7,157,579.01	$4,698,024.68	$4,262,045.09	$16,117,648.77
	Total	**$145,096,861.32**	**$103,026,244.67**	**$87,190,391.46**	**$335,313,497.45**
Connecticut	Contoso Bridgeport Store	$6,966,391.00	$4,658,426.43	$3,890,985.27	$15,515,802.71
	Contoso Darien Store	$6,941,836.17	$4,990,584.50	$4,001,543.53	$15,933,964.19
	Contoso Hartford Store	$6,575,261.77	$5,034,345.93	$4,183,336.31	$15,792,944.02
	Contoso Litchfield County Store	$6,862,696.87	$4,946,293.75	$4,287,795.02	$16,096,785.65
	Contoso New Haven Store	$6,779,187.97	$5,001,493.22	$4,099,779.08	$15,880,460.28
	Contoso New London Store	$6,957,778.23	$4,818,563.99	$3,959,231.28	$15,735,573.50
	Contoso Old Saybrook Store	$6,670,124.17	$4,901,962.91	$4,269,021.32	$15,841,108.40
	Contoso Waterbury Store	$6,292,587.60	$4,735,232.34	$4,076,507.95	$15,104,327.89
	Total	**$54,045,863.79**	**$39,086,903.06**	**$32,768,199.77**	**$125,900,966.63**
Florida	Contoso Cape Canaveral Store	$6,772,534.39	$4,981,735.54	$4,237,724.87	$15,991,994.80
	Contoso Fort Lauderdale Store	$7,061,400.41	$4,773,879.05	$4,359,840.62	$16,195,120.08
	Contoso Jacksonville Store	$7,111,727.69	$4,863,161.28	$4,185,329.54	$16,160,218.52
	Contoso Key West Store	$6,845,171.62	$4,991,294.20	$3,970,006.84	$15,806,472.67

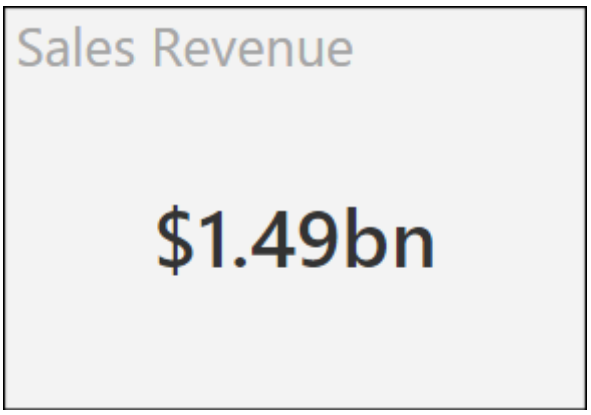
Figure 78 – Completed Report Page

This example illustrates that PivotTable-style reports are easily created and formatted in Power BI Desktop. Furthermore, the absence of drill-down is likely not a severe limitation. The number of detailed records generally exceeds what most users can analyze effectively in a manual review process; hence, drilling down to detail adds little value. Bottom line: you can quickly produce summary reports from large datasets using Power BI Desktop as Matrix tables.

Building Multi-tile, Dashboard-style Reports

In the following example, we will create a second page that presents data in a graphical format similar to Power View. Click the plus (**+**) sign at the bottom of the **Report** view to create a new page in our report. Double-click on the tab and rename it to "**Sales Dashboard**." In this case, we will filter the *page* to display data from 2016 only. Expand the **FactSales** table in the field list and drag the **DateKey** field to the **Page level filters** box in the **Filters** area of the **Visualizations** pane. Click **Advanced Filtering** and set **Show items when the value: is on or after 1/1/2016 And is on or before 12/31/2016**. Click **Apply filter** and then collapse the **Page level filters** box. Note that the entire report (all pages) is filtered to include US sales only.

Expand the **FactSales** table in the field list and check the **SalesAmount** field to add it to the report canvas. It will add to the canvas as a clustered column chart tile by default. Next, click **Card** in the gallery on the Visualizations pane to convert the tile to a **Data** card. Select the tile and then click the **Format** button in the **Visualizations** pane. Finally, set the tile's format to match the settings in the following table. After completing these steps, click the **Fields** button and deselect the report tile.

	**Data labels ON** **Display units:** Billions **Decimal places:** 2 **Category label OFF** **Title ON** **Title Text:** Sales Revenue **Text size:** 18pt **Background ON** **Color:** Light Gray 1 **General** **Width:** 260 **Height:** 180

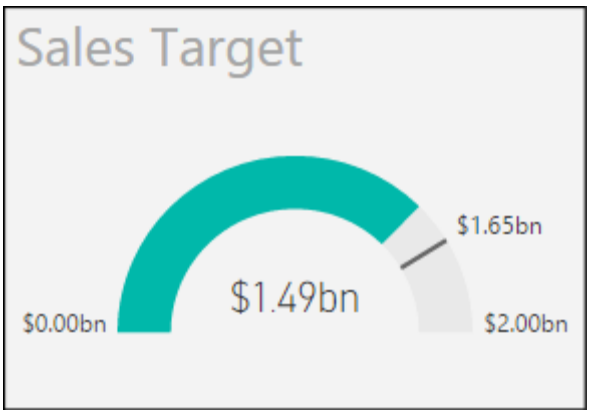
The next tile to add will be a gauge to identify our progress toward our annual sales goal. This tile will require two calculated measures. To create this tile, select the **FactSales** table in the **Fields** list. Next, from the Ribbon's **Home** tab, click **New Measure**. Finally, in the **Formula** bar just below the Ribbon, enter the formula without the parenthetical.

TargetSalesRevenue = 1650000000

Again, click **New Measure**. Then in the **Formula** bar, enter the following formula.

MaxSalesRevenue = 2000000000

With the required measures in place, drag **SalesAmount** from the **FactSales** table to a blank area on the canvas. Power BI adds it to the canvas as a **Clustered column chart** tile by default. With the chart tile selected, click **Gauge** in the gallery on the **Visualizations** pane to convert the tile to a gauge chart. Select the tile. Then drag the **MaxSalesRevenue** measure to the **Maximum value** box. Also, drag the **TargetSalesRevenue** measure to the **Target value** box in the **Values** area. Next, click the **Format** button in the **Visualizations** pane. Set the tile's format to match the settings in the following table. Upon completing these steps, click the **Fields** button and deselect the report tile.

	**Data labels ON** **Display units:** Billions **Decimal places:** 2 **Callout value ON** **Title ON** **Title Text:** Sales Target **Text size:** 18pt **Background ON** **Color:** Light Gray 1 **General** **Width:** 260 **Height:** 180 **Border ON**

The next tile will be a Key Performance Indicator (KPI) tile to identify our performance against our target sales revenue per square foot of selling space. In the context of Power BI, Microsoft defines a **Key Performance Indicator** (KPI) as "a visual cue that communicates the amount of progress made toward a measurable goal." In that context, you can create KPIs in Power BI to assist with the following performance evaluations, among others.

- Identify each sales team member's relative progress toward reaching their annual sales quota.

- Measure the company's financial performance against management's expectations, including budget versus actual comparisons.

- Evaluate customer service objectives by relating measurements such as Net Promoter Scores and customer surveys to established goals.

You can easily incorporate the three potential KPIs mentioned above and countless others into a Power BI report or dashboard, assuming the necessary data is available in the dataset.

Suppose you are familiar with using Power Pivot to create KPIs in Excel-based PivotTables. In that case, you are already well on your way to creating KPIs in Power BI. Like KPIs in PivotTables, KPIs in Power BI depend upon two pieces of information.

First, each KPI must have a ***base value***. The base value is the "actual" data you will be reporting. Thus, following the abovementioned example, if your KPI focuses on measuring each salesperson's sales relative to their quotas, *actual sales* would be a possible base value.

Second, Power BI-based KPIs will always incorporate a ***target value***. As its name implies, the target value is the objective against which you measure your base value. So, for example, each

salesperson's sales quota would be a likely target value when creating a KPI to measure sales performance.

We must add two calculated measures to complete the tile for this KPI. To do so, first, select the **FactSales** table in the **Fields** list. Then, on the Ribbon's **Home** tab, click **New Measure**. Then, in the **Formula** bar just below the Ribbon, enter the following formula.

SalesPerSqFoot = SUM(FactSales[SalesAmount])/SUM(DimStores[SellingAreaSize])

Again, click **New Measure**. Then, in the **Formula bar**, enter the following formula.

TargetSalesPerSqFoot = 500

With the required measures created, drag **SalesPerSqFoot** from the **FactSales** table to the canvas. Power BI adds it to the canvas as a **Clustered column chart** tile. With the chart tile selected, click **KPI** in the gallery on the **Visualizations** pane to convert it to a KPI tile. Select the tile and drag the **FiscalMonthLabel** field to the **Trend axis** box. Continue the process by dragging the **TargetSalesPerSqFoot** measure to the **Target goals** box in the **Values** area.

We want to filter out *catalog* and *online* sales so that the KPI measures sales in conventional stores. Drag the **StoreName** field from the **DimStores** table to the **Visual level filters** box. Expand the **StoreName** filter and *check* **Select All**. Then, *uncheck* **Contoso Catalog Store**, **Contoso North America Online Store**, and **Contoso North America Reseller**. Next, click the **Format** button in the **Visualizations** pane. Set the tile's format to match the settings in the following table. After completing these steps, click the Fields button and deselect the report tile.

Indicator
- **Display units:** None
- **Decimal places:** 2

Trend axis ON

Goals
- **Goal ON**
- **Distance ON**

Status
- **High is good**

Title ON
- **Title Text:** KPI: Sales Per Sq Foot
- **Text size:** 18pt

Background ON
- **Color:** Light Gray 1

General
- **Width:** 260
- **Height:** 180

Border ON

If **SalesPerSqFoot** is incorrectly formatted, click its field name in the **Fields** list. Next, click the **Dollar Sign ($)** on the **Ribbon's Data Tools, Modeling** contextual tab to apply the accounting number format with dollar signs. Also, use the spinner control to enter **2** decimal places.

Next, we will add a simple table to display sales revenue per store. Begin by dragging the **StoreName** field from the **DimStores** table and dropping it on the canvas. By default, it will add to the canvas as a **Table** tile. Then, drag **SalesAmount** from the **FactSales** table and drop it on the Table tile created in the previous step. Widen the table and its columns to ensure the data appears appropriately. Sort the **SalesAmount** column in descending order, with the highest sales revenue at the top of the table. Next, click the **Format** button in the **Visualizations** pane. Set the tile's format to match the settings in the following table. Upon completing these steps, click the **Fields** button and deselect the report tile.

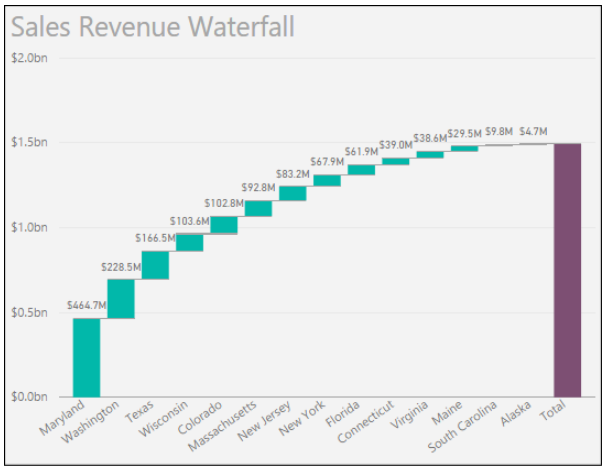
General
- **Width:** 320
- **Height:** 180

Title ON
- **Title Text:** Sales Revenue Per Store
- **Text size:** 18pt

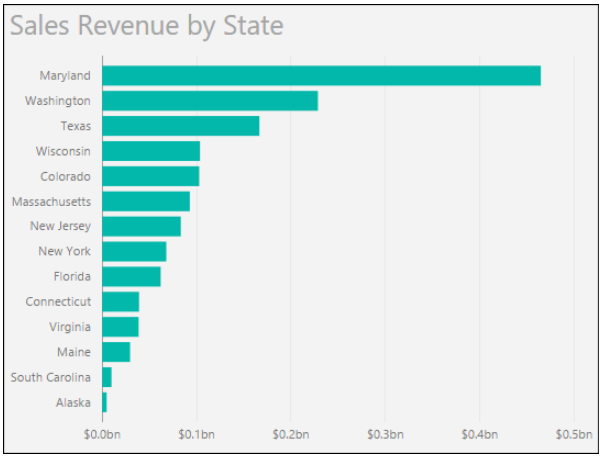
Background ON
- **Color:** Light Gray 1

Border ON

The next tile will be a waterfall chart that identifies each state's sales revenue contribution to total sales revenue. Drag the **StateProvinceName** field from the **DimGeography** table and drop it on the report canvas. It will appear on the canvas as a two-dimensional Map tile by default. Click **Table** in the **Visualizations** gallery with the tile selected to convert it to a Table. Next, drag the **SalesAmount** field **from** the **FactSales** table and drop it on the table tile. Then, sort **SalesAmount** in descending order by sales revenue. Click **Waterfall** chart in the Visualizations gallery to convert the tile to a Waterfall chart. Resize the chart as necessary so that all the chart is visible. Set the tile's format to match the settings in the following table. Upon completing these steps, click the **Fields** button and deselect the report tile.

	**Legend OFF** **Data labels ON** **Display units:** Millions **Decimal places:** 1 **Sentiment Colors** **Increase:** Green 4 **Total:** Purple 4 **X-Axis ON** **Y-Axis ON** **Start:** 0 **End:** 2000000000 (2 billion) **Display units:** Billions **Decimal places:** 1 **Reference Line OFF** **Title ON** **Title Text:** Sales Revenue Waterfall **Text size:** 18pt **Background ON** **Color:** Light Gray 1 **General** **Width:** 530 **Height:** 400 **Border ON**

Our last tile in the report will summarize and present sales revenue by each state in a conventional clustered bar chart. Because it contains the same data as the waterfall chart, the easiest way to create the new tile is to copy and paste the waterfall tile. First, click the waterfall tile, press **CTRL + C** to copy the tile, and press **CTRL + V** to paste a copy on the canvas. Next, use your mouse to reposition the chart. Then click **Clustered bar chart** in the **Visualizations** gallery to convert the tile to a bar chart. Next, click the ellipsis **(…)** in the upper right corner to sort the bar chart, noting that you cannot sort all chart types using this technique. Next, set the tile's format to match the settings in the following table. Upon completing these steps, click the **Fields** button and deselect the report tile.



X-Axis ON
Y-Axis ON
Reference Line OFF
Data labels ON
Display units: Millions
Decimal places: 1
Title ON
Title Text: Sales Revenue by State
Text size: 18pt
Background ON
Color: Light Gray 1
General
Width: 530
Height: 400
Border ON

Note the **Arrange**, **Alignment**, and **Distribute** buttons on the Ribbon's **Format** contextual tab. You can use these tools to place tiles in a report. First, use **CTRL + click** to select the four smaller tiles – Sales Revenue, Sales Target, KPI: Sales Per Sq Foot, and Sales Revenue Per Store – and then click **Format, Distribute, Distribute Horizontally** from the Ribbon so that the smaller tiles are equally spaced. Then, with the tiles still highlighted, click **Format, Align, Align Top,** so the tiles align correctly.

Use **CTRL + click** to select the two larger chart tiles – Sales Revenue Waterfall and Sales Revenue by State – and then choose **Format, Align**, and **Distribute Horizontally** from the Ribbon to space the two chart tiles equally. Then, while they remain highlighted, select **Format, Align,** and **Align Top** to align the tiles correctly. The completed report page should resemble the one shown in **Figure 79**.

Note that the report is interactive. For example, click the bar representing Wisconsin in the clustered bar chart. Immediately, all the other tiles reflect the sales revenue of the Wisconsin stores. For example, the Sales Target gauge displays the contribution of Wisconsin sales to the overall target. Likewise, the KPI tile reflects the Wisconsin stores' sales revenue per square foot. Finally, the sales revenue per store table displays only those stores in Wisconsin. To clear the interactive filter, click anywhere in the empty chart area of the clustered bar chart.

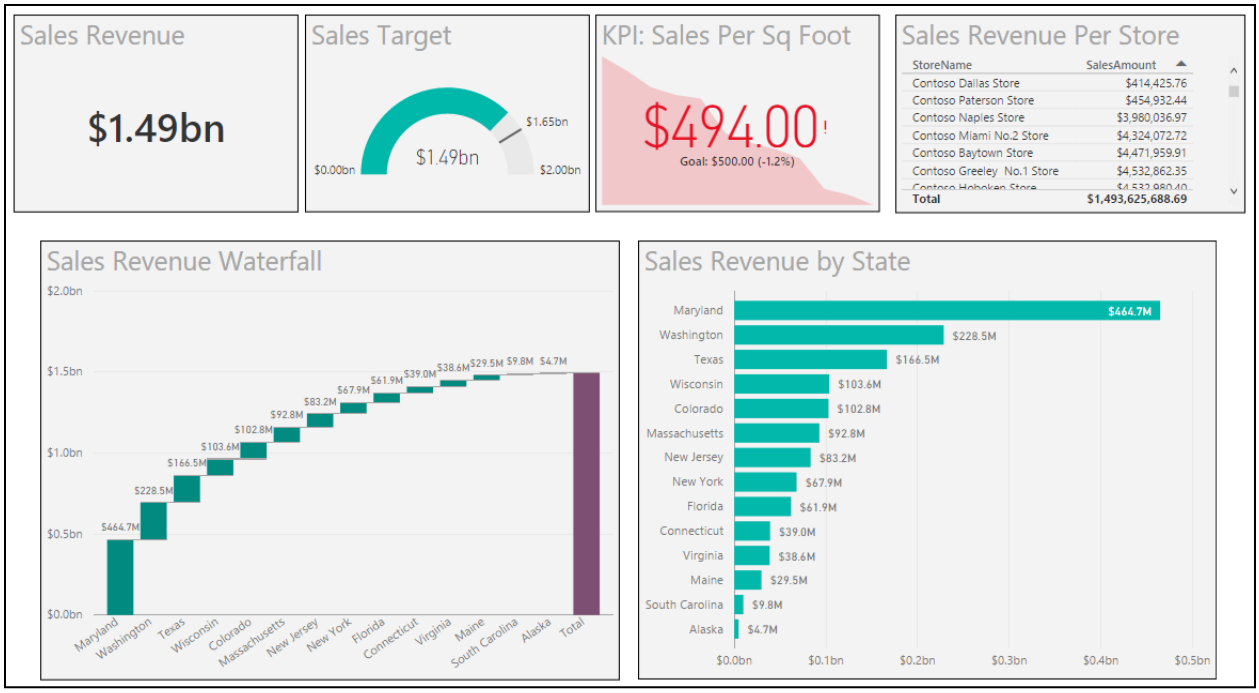


Figure 79 – Completed Dashboard Report in Power BI Desktop

Publishing a Report to the Power BI Service and Creating Dashboards

After generating a Power BI Desktop report, you choose to save it to your PC, a network share, or a Cloud-based storage service such as OneDrive or OneDrive for Business. Because the files have a ***.pbix*** file extension, users can open them if they have Power BI Desktop installed. Then, double-click a saved report file in File Explorer to open it like any other file type. However, Power BI reports can be large and not easily shared as email attachments.

If you have a subscription to the Power BI service (or access to Power BI Server), you can easily publish your reports there. Click **Publish** on the right side of the Ribbon's **Home** tab in Power BI Desktop. **Figure 80** illustrates the process.

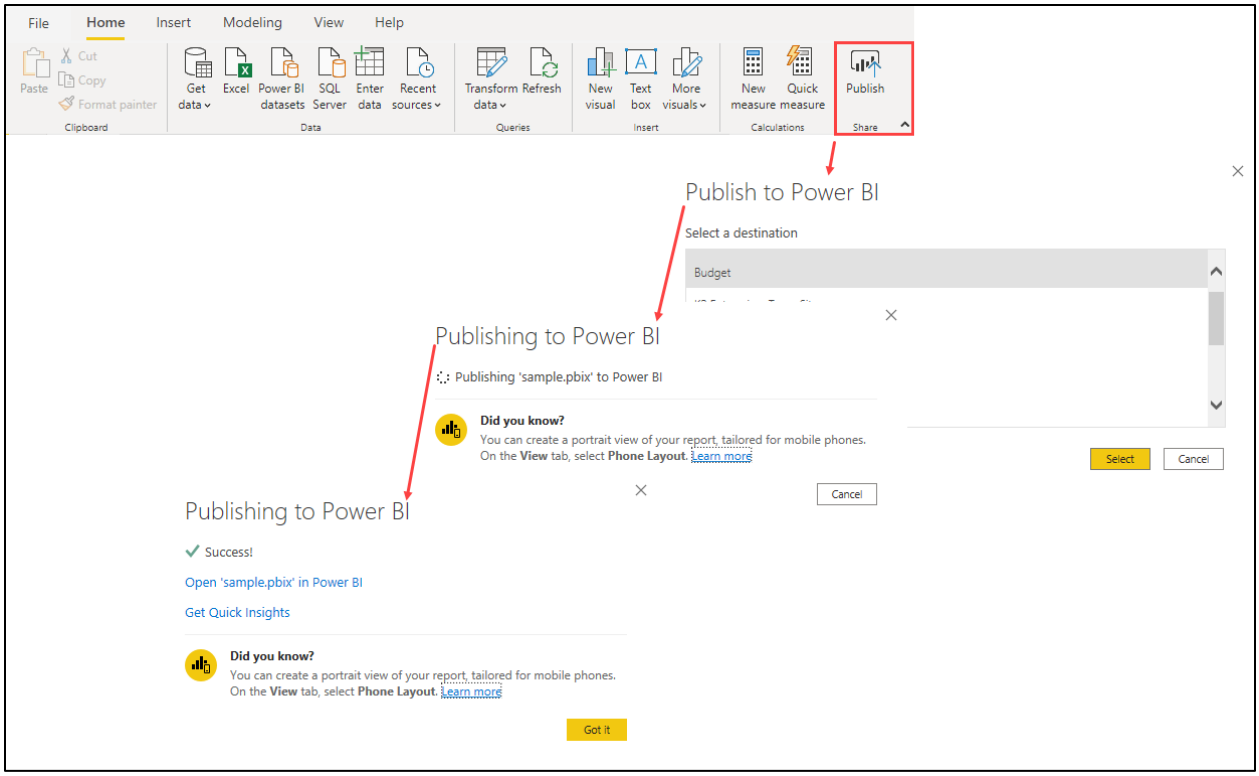


Figure 80 – Publishing a Power BI Report to Power BI.com

Once you publish a report, you can "pin" individual tiles or entire pages to a new or existing dashboard in the Power BI.com service. Dashboard tiles allow users to drill into underlying reports. This functionality means you can aggregate large amounts of information from multiple reports onto a single dashboard. Dashboards also enable natural language Q&A so that managers can get the information they need quickly and easily. Further, you can share dashboards with other team members.

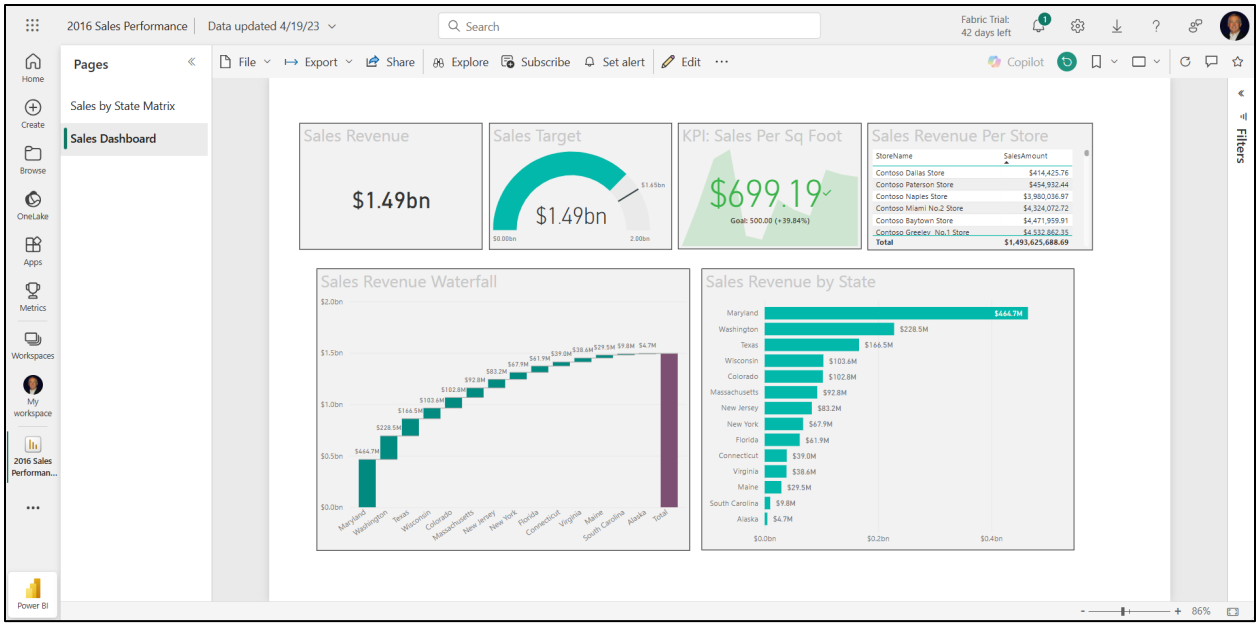


Figure 81 – Power BI Desktop Report Published to Power BI

Note some of the additional features available in the Power BI service. For example, the service provides access to **Workspaces**. Workspaces provide users with options for organizing their Power BI reports and dashboards. A key advantage of using workspaces is facilitating collaboration among multiple users. Advantages and use cases associated with Workspaces include the following.

1. **Collaboration**: Workspaces allow multiple users to collaborate on the same set of data sources, reports, and dashboards. This is particularly useful for teams working together on data analysis and report development.

2. **Content Management**: Users can create, upload, and manage their Power BI content within these workspaces. This includes datasets, reports, dashboards, and more. Each workspace can be configured with specific users and access permissions to ensure that only authorized personnel can view or modify content.

3. **Version Control and Deployment**: Power BI Pro and Premium users can use enhanced workspace capabilities such as version control and deploying content across different workspaces. This helps in maintaining consistency and control over the published reports and datasets.

4. **Access Control**: Admins can set up different access levels for workspace members, ranging from view-only to full edit capabilities. This helps in managing who can see and modify the workspace content.

5. **Integration**: Workspaces can be integrated with other Microsoft services, such as Microsoft Teams, allowing for easier access and broader collaboration among team members. They also support the scheduling of automatic data refreshes and publishing reports to a broader audience within an organization.

Creating A Dashboard

Once you create a report, you can generate a dashboard from that report and others. Essentially, a dashboard is a collection of tiles, and the tiles for a given dashboard do not need to originate from the same report. For example, if you have a Sales Report, an HR Report, and a Production Report, you could create a dashboard that includes content from each of these three reports.

To create a dashboard, begin in the Power BI service or server environment (remember, you cannot create dashboards in Power BI Desktop). Then locate the report(s) that contain the tiles you want to add to your dashboard. Next, click the ellipses next to Edit in the Power BI window and choose **Pin to a dashboard**, as shown in **Figure 82**. Repeat that process for as many tiles as you want to include on your dashboard.

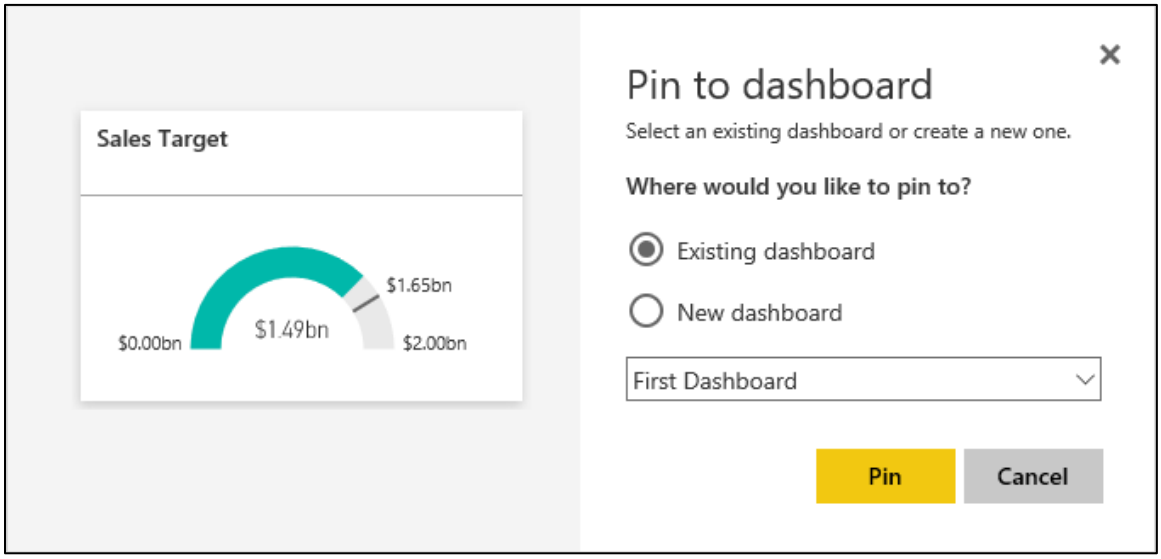


Figure 82 – Pinning a Tile to a Dashboard in Power BI

Note that pinned tiles are hyperlinks to the reports from which they originate. If you think of the Power BI environment as a pyramid, you would see that Dashboards sit at the top, and Reports, Tiles, and Datasets support them in that order, as shown in **Figure 83**.

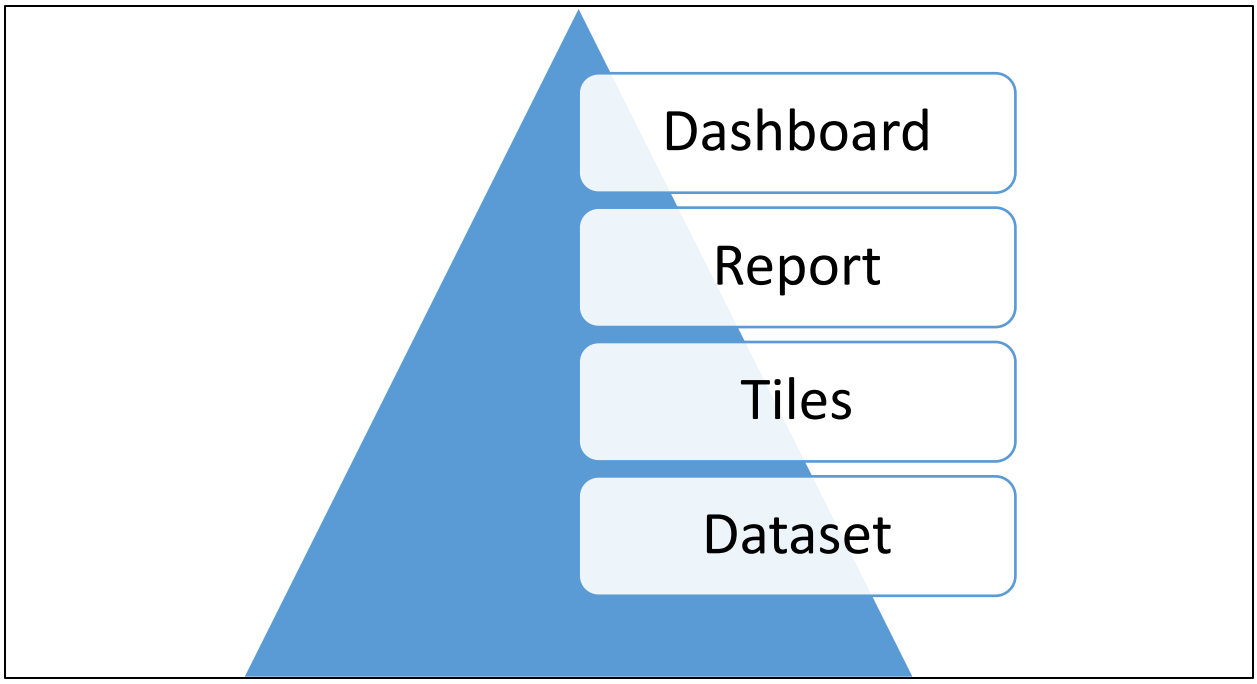


Figure 83 - The Power BI "Pyramid"

Importantly, you can share the dashboards you create in Power BI with other users. To share a dashboard, click the “share” icon in your list of dashboards to open the **Share dashboard** dialog box. Next, enter the email address of each person you would like to share the dashboard with and click **Share** near the dialog box's bottom. Of course, before sharing the dashboard, address any security and data access issues. Additionally, you can share a dashboard by copying the **Dashboard link** and providing the dashboard link to your intended audience. However, if you choose this sharing method, anyone who obtains the link can access the dashboard, so carefully consider the security implications of sharing dashboards using links. **Figure 84** illustrates the process of sharing a dashboard.

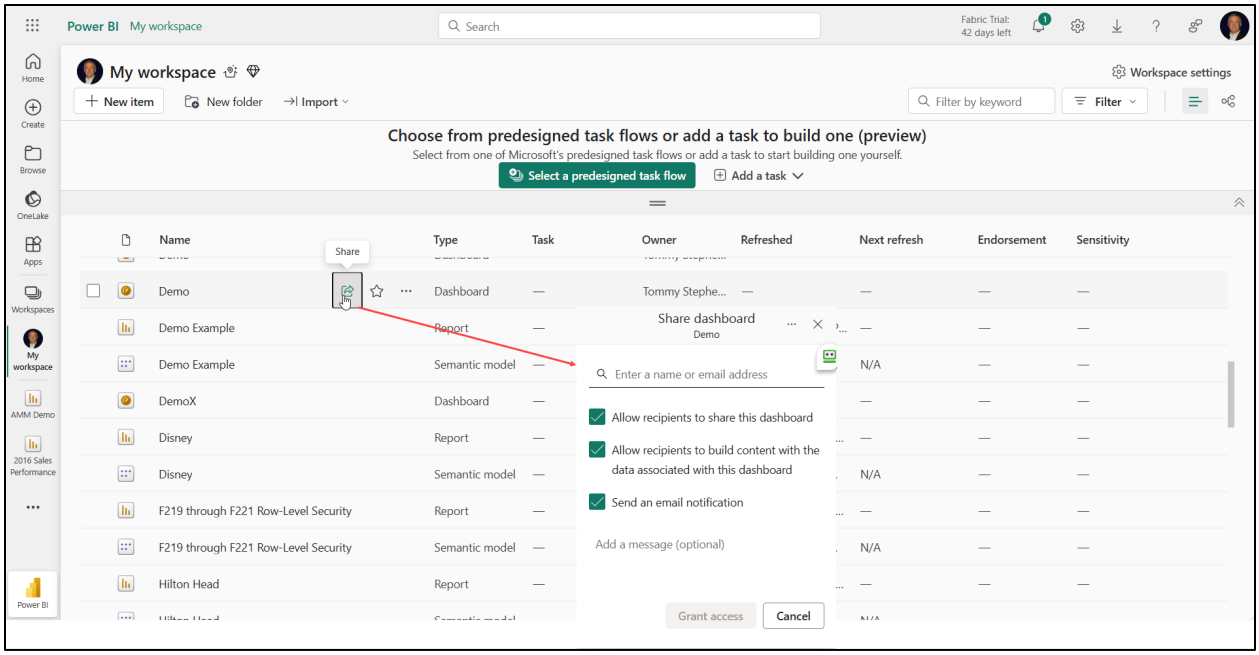


Figure 84 – Sharing a Dashboard in Power BI

Querying a Dashboard

You can query your dashboards using natural language Q&A querying. You need only type the query in the **Ask a question about your data** box near the dashboard's top. For example, typing "sales amount for Q1" into the query box shown in **Figure 85** causes Power BI to display the first quarter's total sales. This feature can be a powerful tool for those who take advantage of it because it facilitates *ad hoc* querying of the underlying dataset.

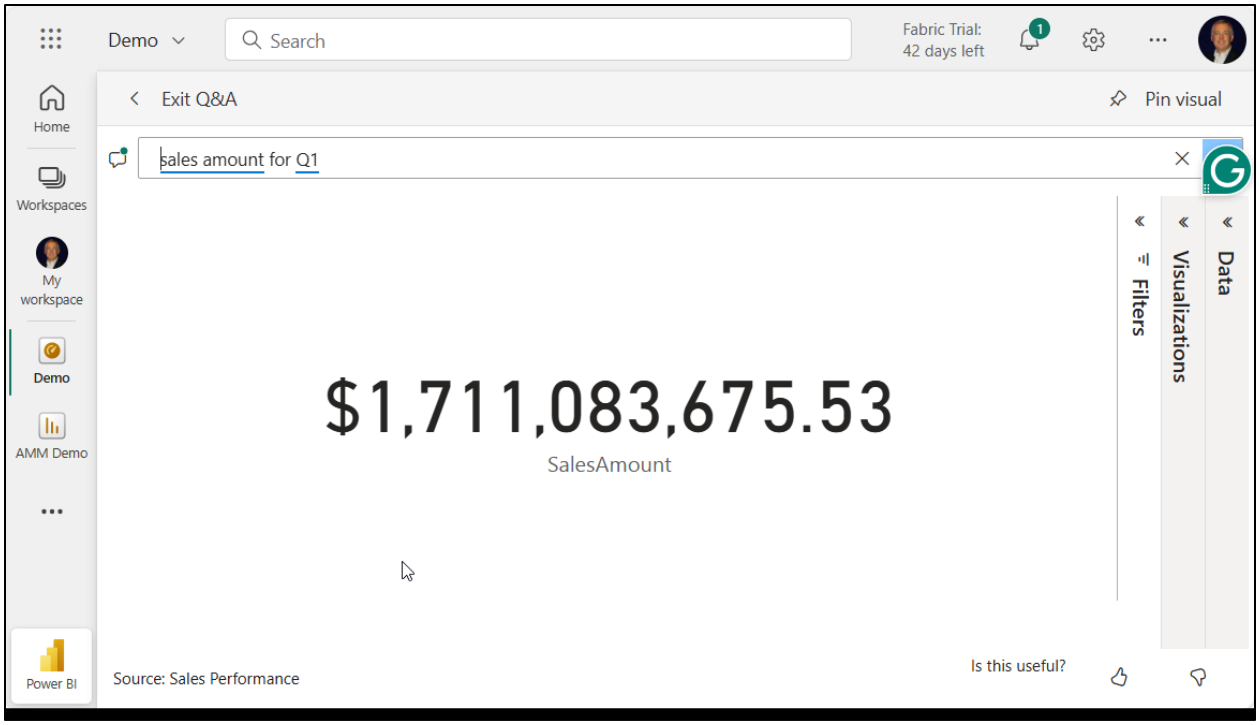


Figure 85 – Querying a Power BI Dashboard Using Natural Language Query Technology

Optimizing a Dashboard for Mobile Viewing

One of Power BI's strengths is its ability to provision BI dashboards so users can view them from mobile devices. For example, downloading the free Power BI app onto a smartphone allows you to access your Power BI-based dashboards while away from the office. Further, when accessing your dashboards from a mobile device, you will retain the drill-in interactivity you would otherwise have if you access the dashboard from a desktop or laptop computer.

By default, you work in Power BI Desktop's *desktop view* when constructing a report or a dashboard. On the other hand, when you work in Power BI.com, you are working in *web view* by default. However, in both tools, you can toggle to a different view. This feature allows you to rearrange the tiles on a dashboard optimized for viewing on a mobile device. In Power BI Desktop, this view is *phone layout*; in Power BI.com, it is *Phone view*.

For example, **Figure 86** shows how the report in Figure 79 might appear if you rearranged it in Power BI Desktop's phone layout. In phone layout (and phone view), you can re-order and resize tiles so that they are easier to read on the smaller displays found on mobile devices. Notably, optimizing a report or dashboard for mobile viewing does not impact how the report or dashboard will appear in the desktop layout in Power BI Desktop or the web view in Power BI.com.

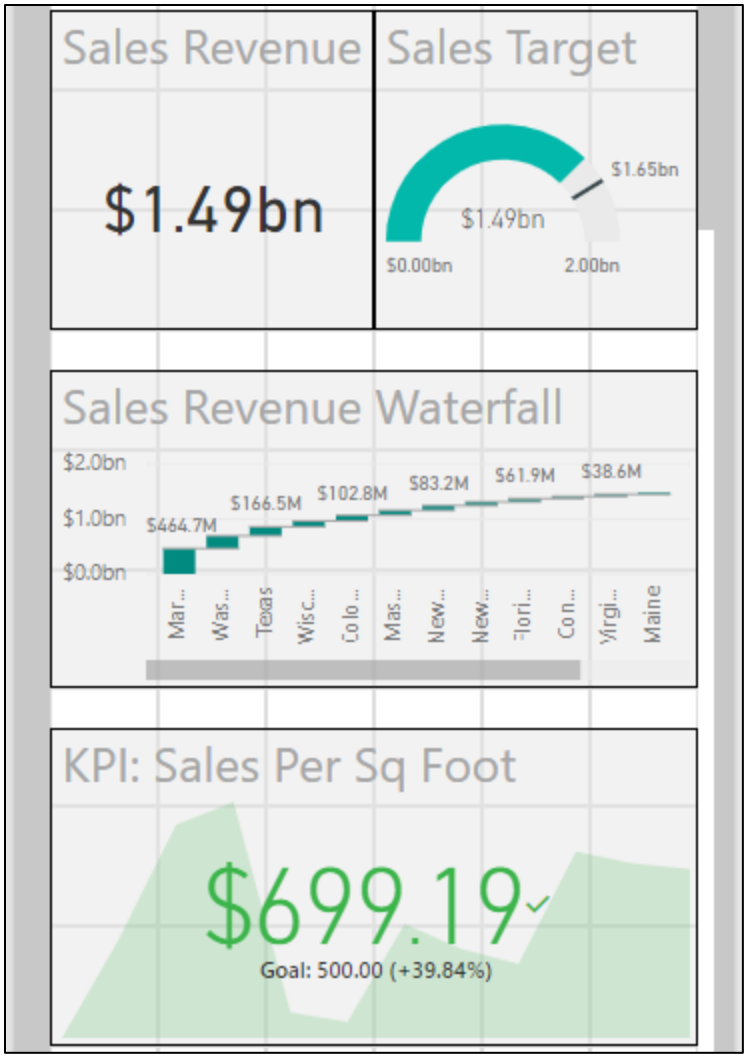


Figure 86 – Sample Phone Layout in Power BI Desktop

?	How important is it for you and other users in your organization to access business intelligence dashboards from mobile devices?

Advanced Topics in Power BI

Chapter 5

This chapter discusses extending your Power BI environment by incorporating tools such as Data Analysis Expressions. By taking advantage of the tools and techniques covered in this chapter, you will be able to move beyond simple reports and dashboards in Power BI and truly leverage the power of this fantastic reporting tool.

Learning Objectives

Upon completing this session, you should be able to:

- Manage Power BI datasets more effectively;
- List the steps necessary to create formulas using Data Analysis Expressions;
- Apply appropriate security techniques to your Power BI reports and dashboards; and
- Identify the purpose of Apps in the context of Power BI.

It Is Always About the Dataset!

The *dataset* is the cornerstone of any report or dashboard you might build using Power BI. More to the point, "window dressing" cannot cure the ills of an improperly constructed dataset.[9] Let us consider some of the essential characteristics of your datasets in Power BI to set the stage for success when working in Power BI.

Fact Tables and Dimension (Dim) Tables

First, you should understand the difference between "fact" tables and "dimension (dim)" tables. In the data you query into your datasets, fact tables contain the numeric information you need to summarize in your Power BI reports and dashboards. Examples include sales transactions, hours recorded, and disbursements for expenses. On the other hand, dimension tables store the categories, dates, product lines, accounts, and different groupings by which we want to group, summarize, and filter the data stored in the related fact tables. For example, suppose you are querying data from an accounting application into your Power BI-based dataset. The accounting

[9] Microsoft refers to the underlying data associated with Power BI reports and dashboards as *datasets*. If you have worked with *data models* in Excel, you will recognize that a dataset in Power BI is virtually the same as a data model in Excel.

transactions would reside in a fact table, and items such as the fiscal reporting period, department, division, and sales channel would reside in multiple dimension tables. You would then "relate" these tables together based on the presence of one or more common fields in your dataset so that you can generate useful reports and dashboards.

As a matter of design principle (and to the extent that you have some influence over the design of the tables in the underlying database), you should strive to construct your dim tables so they do not expand endlessly. For example, you should not create a dim table containing items such as general ledger accounts and departments. Mixing dissimilar information in the same dim table increases the time, effort, and complexity of maintaining these tables. Continuing with the general ledger account and department analogy, if your dim table contained both attributes, you would need a separate row for every possible combination of general ledger account and department. However, segregating that information into different dim tables will minimize both tables' sizes. This action simplifies maintenance and improves the overall performance of the reports and dashboards built from the dataset. **Figure 87** illustrates how the single "uber" table approach could become unwieldy compared to maintaining multiple, smaller dim tables.

Account Number	Account Name	Department ID	Department Name
6010	Advertising Expense	10	Retail
6010	Advertising Expense	20	Wholesale
6010	Advertising Expense	30	OEM
6010	Advertising Expense	40	Corporate
6020	Amortization Expense	10	Retail
6020	Amortization Expense	20	Wholesale
6020	Amortization Expense	30	OEM
6020	Amortization Expense	40	Corporate
6030	Bad Debt Expense	10	Retail
6030	Bad Debt Expense	20	Wholesale
6030	Bad Debt Expense	30	OEM
6030	Bad Debt Expense	40	Corporate

Account Number	Account Name
6010	Advertising Expense
6020	Amortization Expense
6030	Bad Debt Expense

Department ID	Department Name
10	Retail
20	Wholesale
30	OEM
40	Corporate

Figure 87 - Maintaining Multiple, Smaller "Dim" Tables Is Generally Preferable to Maintaining Larger, Combined Tables

"Star" Schema Versus "Snowflake" Schema

In addition to managing your fact and dimension ("dim") tables, you should also be aware of how you relate the tables in your dataset – more specifically, are the tables related in a "star" schema or a "snowflake" scheme? With a star schema, the tables in your dataset connect in a hub-and-spoke fashion, like that pictured in **Figure 88**.

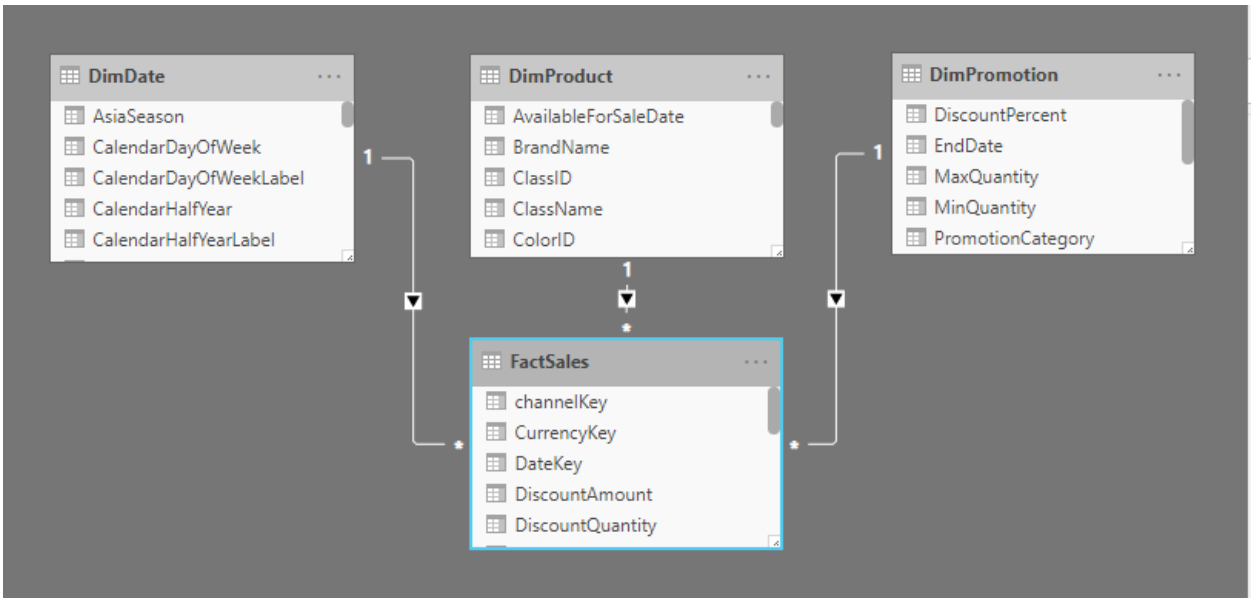


Figure 88 - Sample "Star" Data Schema in Power BI

In a snowflake schema, the tables are “daisy-chained” sequentially, as shown in **Figure 89**.

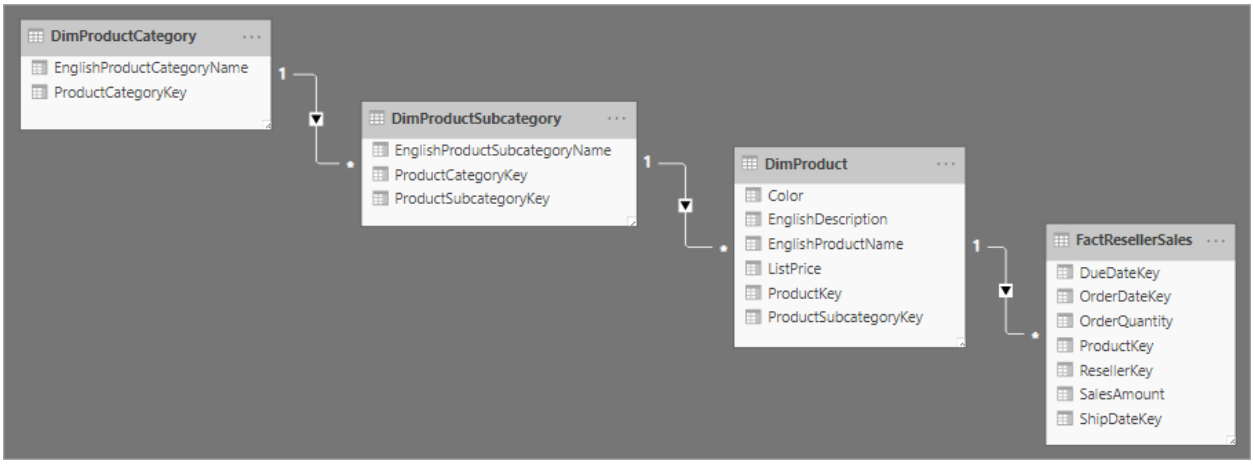


Figure 89 - Example of "Snowflake" Data Schema

The star schema is preferred when working with Power BI datasets because it typically provides better overall performance. The star schema has fewer joins between the various dim tables in the dataset and the fact table(s). Consequently, queries typically run faster in this environment. Further, star schemas are more scalable and adaptable to changes as your dataset evolves. For example, if you add a new dim table to the dataset, you can easily relate that table to a fact table without disturbing other relationships in the dataset. Also, the simplicity of the star schema makes it relatively easy for even novices to understand.

Accessing Data with Power BI Desktop – Import vs. Direct Query

Power BI Desktop offers two primary means to access data from external data sources and pull that data into the application – **Import** and **Direct Query**. Because data is at the heart of any report or dashboard you create using Power BI, it is essential to understand these two approaches' similarities and differences.

Fundamentally, the **Import** option extracts the data from the external data source and loads it into Power BI. With all the data in the application, any visualization you create uses the imported data. If the underlying data changes, you must refresh it to update your visualizations. Refreshing your data imports the most up-to-date version of the dataset. Using Import provides a simple, streamlined experience that works well with smaller datasets. In fact, when using Import, there is a 1 GB file size limitation; if that limitation proves problematic, you should use Direct Query instead.

On the other hand, the **Direct Query** option leaves the data in the source database, and no data is imported or copied into Power BI. Instead, as you create and interact with your visualizations, Power BI queries the underlying data source in real-time, ensuring you look at the most up-to-date information. Examples of some of the data sources to which you can make Direct Query connections include Amazon Redshift, IBM DB2, SAP Hana, SQL Server, Spark, and Teradata.

From the brief discussion outlined in the preceding paragraph, it should be clear that Direct Query is the preferred approach when dealing with very large datasets. If you are unsure which method is used for a particular table in your dataset, click on that table and choose **Storage** mode in the **Properties** pane, as shown in **Figure 90**.

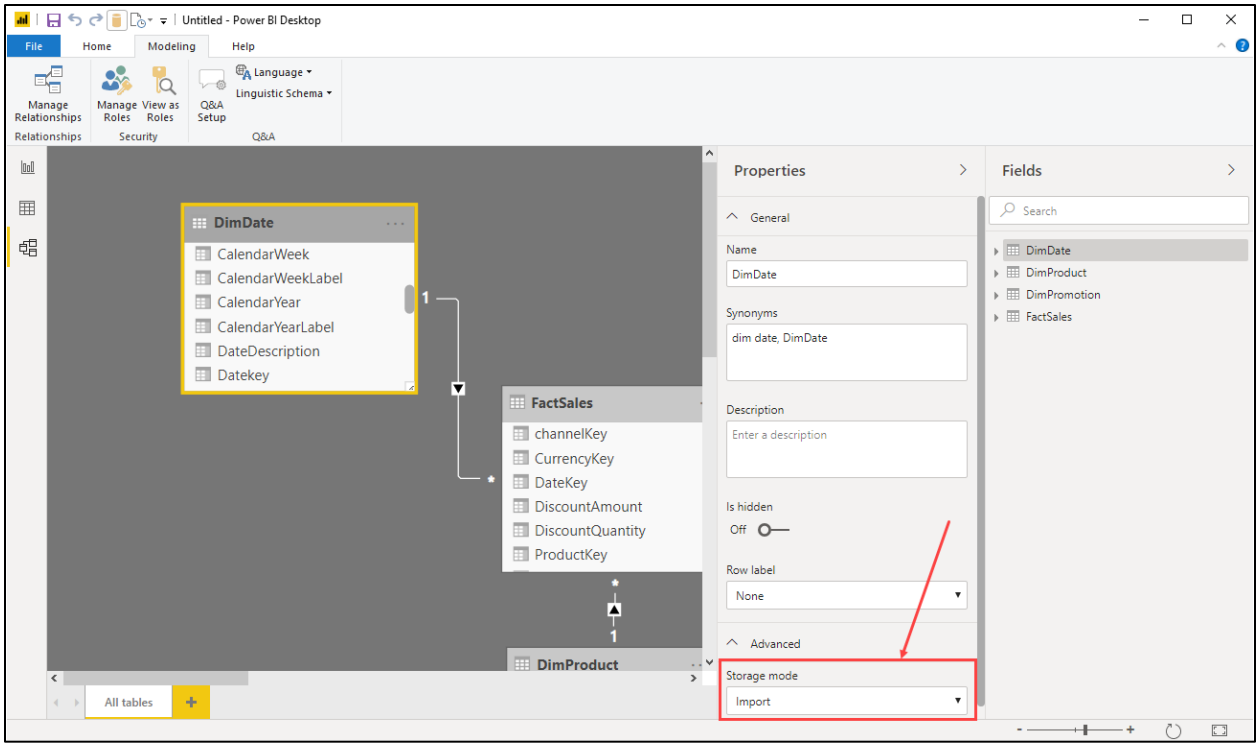


Figure 90 - Determining Whether You Are Using Import or Direct Query

Power BI and Gateways

When working with the Power BI Service, a common question regarding accessing data is, "What is the **On-premises Data Gateway**, and do I need to use it?" To answer that question, let us first examine the Gateway to determine whether it will be necessary for you.

What Is the Gateway?

The On-premises Data Gateway is a tool you can install to allow the Power BI Service (and other services in Microsoft's Power Platform) to access your local data used by a Microsoft cloud-based service, such as Power BI Service. **The Gateway is unnecessary if you work only with Power BI Desktop**. Remember, it facilitates communications between your local data – such as in a SQL Server – and the Power BI *cloud* environment.

Microsoft offers the Gateway in two modes – *Personal* and *Standard*. If you are working independently and other users do not access your work in Power BI, you can install the Personal mode of the Gateway. This mode allows the Power BI Service to refresh data from your local data sources into your cloud-based reports and dashboards. If you use this mode, you must leave your computer running so the Gateway can access your data during its refresh processes.

On the other hand, use the Standard mode of the Gateway for multiple reports and datasets; many users can share this mode of the Gateway. A crucial difference between the Enterprise and

the Personal version of the Gateway is that the Enterprise version supports DirectQuery and live connections to Analysis Services, whereas the Personal version does not.

Do I Need the Gateway?

If you use Power BI Desktop – to the exclusion of the Power BI service – you do not need to install the Gateway. On the other hand, if you use Power BI.com and want your *cloud-based services* to refresh data from your *local data sources*, you will need to install the Gateway to facilitate those refreshes.

To access and install the Gateway, visit the following site:

https://www.microsoft.com/en-us/download/details.aspx?id=53127

Adding Your Own Data to Power BI

Occasionally, you might create a Power BI report or dashboard, only to notice that you do not have all the data necessary for your reporting objective. Perhaps, for example, you want to create a visualization that highlights salesperson performance relative to established quotas. However, you do not have each salesperson's quota in a table in your underlying dataset. To remedy this situation quickly, you could add a user-defined table to the dataset model that contains each salesperson's quota.

To do so, click **Enter Data** on the **Home** tab in Power BI. Then, in the resulting **Create Table** window, add a name for the newly created table near the window's lower left corner. Then, add names for each column (field) in the table and data for each row (record). Finally, click **Load** near the window's lower right corner to add a user-defined table, an example of which **Figure 91** provides.

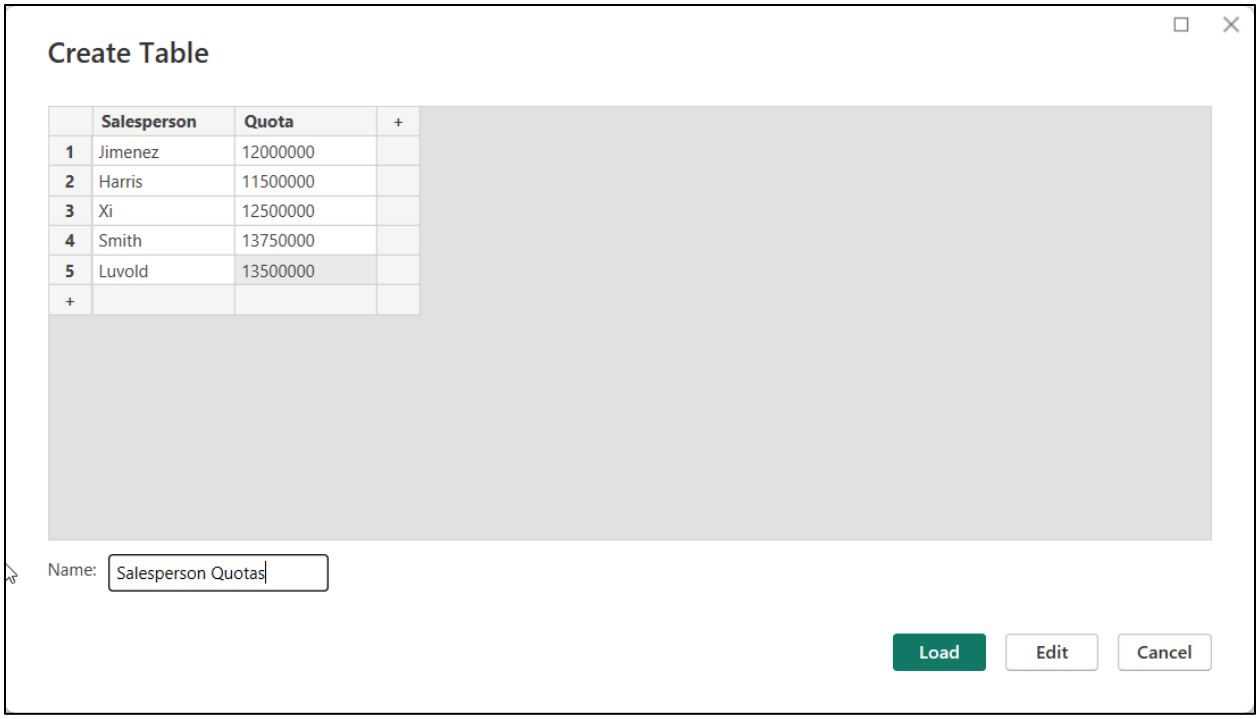


Figure 91 - Creating a User-defined Table in Power BI

Once you create the user-defined table, Power BI treats it like all other tables. You will be able to relate it to other tables in your dataset. Likewise, you will be able to add DAX-based user-defined calculations to it. And you can incorporate its contents in any Power BI-based report, dashboard, or other objects you might choose to create. However, you cannot return to the table and edit it as if it were a spreadsheet.

Managing Cardinality and Cross Filter Direction in Power BI Desktop

Whenever you add two or more tables to a dataset, Power BI Desktop attempts to identify and establish relationships between these tables. These relationships are critical to the success of your project because they allow you to "link" multiple tables together based on the presence of one or more common fields. In turn, these relationships enable you to create reports and dashboards that summarize information from multiple tables.

As Power BI Desktop attempts to create relationships between tables, it also automatically tries to establish the *cardinality* and *cross filter direction* properties for that relationship. Further, if it does indeed identify and establish a relationship, it automatically marks that relationship as "active."

The cardinality property determines the direction of the relationship. According to Microsoft, there are four options for cardinality in a Power BI dataset.

1. **Many-to-one (*:1)**: A many-to-one relationship is the default relationship type. Not surprisingly, it is also the most common. It means the column in one table can have more than one instance of a value, and the other related table, often known as the lookup table, has only one instance of a value.

2. **One-to-one (1:1)**: In a one-to-one relationship, the column in one table has only one instance of a particular value, and the other related table has only one instance of that value.

3. **One-to-many (1:*)**: In a one-to-many relationship, a column in one table has only one instance of a particular value. Note that the related column in the other table can have more than one instance of a value.

4. **Many-to-many (*:*)**: With composite models, you can establish a many-to-many relationship between tables, removing requirements for unique values in tables. It also removes previous workarounds, such as introducing new tables to establish relationships. However, in general, you should seek to avoid many-to-many relationships.

You can manage the cardinality – and other aspects of a relationship – in Power BI Desktop. First, click **Manage Relationships** on the Ribbon's **Home** tab in Power BI Desktop to open the **Manage Relationships** dialog box. Next, select the relationship you need to edit and click the **Edit** button near the bottom of the dialog box to open the **Edit relationship** dialog box, as shown in **Figure 92**. There, you can edit the relationship's cardinality in your dataset.

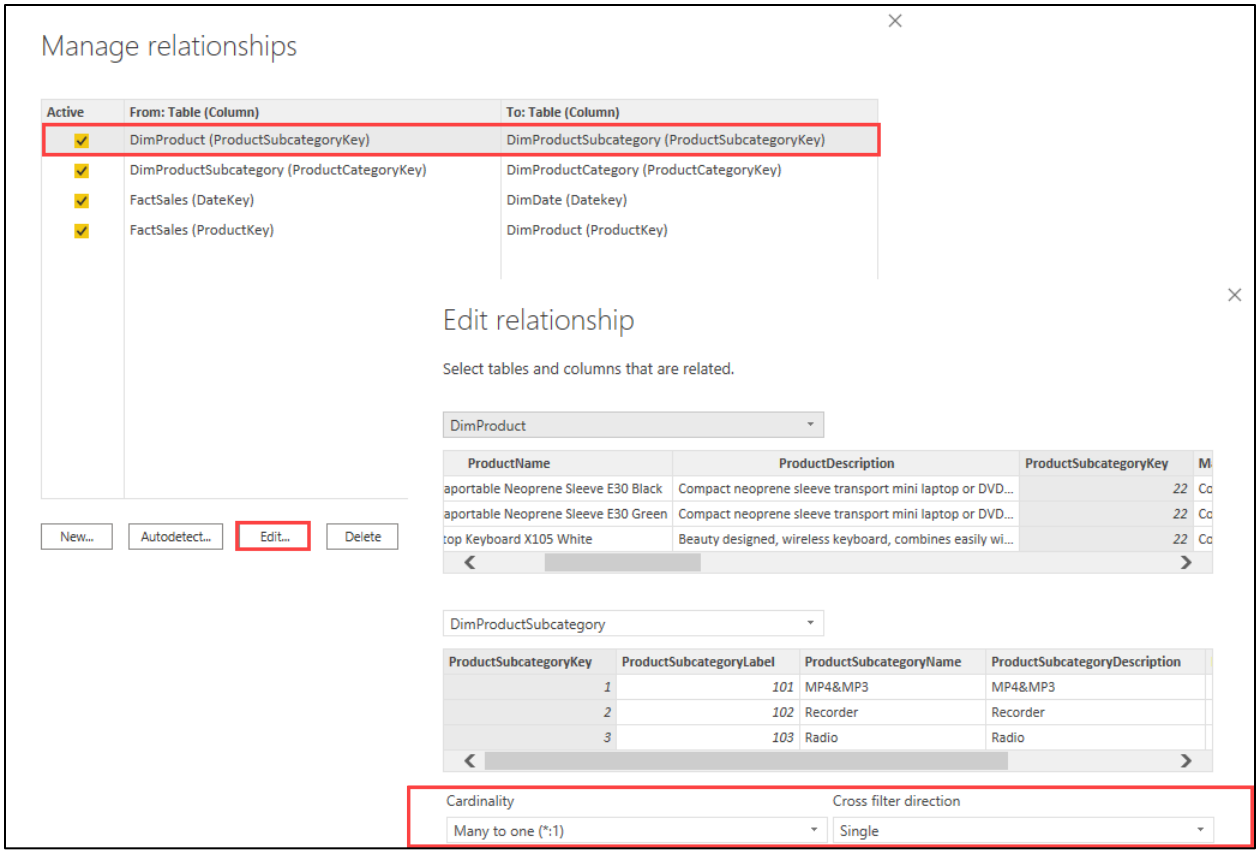


Figure 92 - Managing Relationships, Including Cardinality, in Power BI Desktop

In addition to editing the cardinality of a relationship, you can also change the cross-filter direction. The **Both** setting is generally preferred for most datasets, particularly those arranged in a "star" schema. This setting allows Power BI to treat all aspects of the connected tables as if they are a single table. However, generally, you should not use this setting when you have two or more fact tables that link to common dim tables. In cases such as these, the **Single** option is generally preferred; in fact, the Single option is mandated if you import a data model from Excel 2013 or Excel 2010.

Creating Formulas in Power BI with Data Analysis Expressions

Data Analysis Expressions (DAX) functions are a library of functions to create user-defined calculations in Power BI datasets. You can also use DAX functions in Excel-based data models, Azure Analysis Services, and SQL Server Analysis Services.

The formulas you create using DAX functions can address issues such as time-based calculations, filtering your data, transforming textual data, and 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 datasets. You can examine the full complement of DAX functions at https://docs.microsoft.com/en-us/dax/dax-function-reference.

The DAX environment currently includes over 300 functions, some of which act virtually the same as their counterparts in a traditional Excel environment. These functions can add calculated columns and measures to your Power BI datasets. These functions are powerful and can extend the usefulness of Power BI for all users.

While many of the functions used for creating calculated columns work the same in Power BI 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 new **BLANK** function often used with **IF** to exclude specific rows from calculations.

- DAX uses **RELATED** instead of **VLOOKUP** to perform queries on data in a PivotTable.

Also, according to Microsoft, some general guidelines regarding using DAX functions include those listed below.

- 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.

- DAX functions never take a specific 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.

- DAX provides new lookup functions similar to Excel's array and vector lookup functions. However, the DAX functions require that a relationship exists between the tables.

- DAX does not support the variant data type found in Excel. Instead, the data in a column should 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** have similar 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 of SUM.

=SUM(A1:A57)

=SUM(Sales[Amt])

In the first example, 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 instructs Excel to sum the "Amount" column from the "Sales" table in a dataset.

Creating Simple Calculated Columns with DAX

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

To illustrate a calculated column built using DAX expressions, consider the data extraction shown in **Figure 93**; the extraction shows a sampling of time and billing data for a consulting firm. In addition to the firm's management wanting to see total hours and revenue, the managers also want to see average revenue at the record level. One way to do this would be to add a calculated column to the dataset. As shown in Figure 93, this process can begin by right-clicking in the **Fields** pane on the right side of the window and choosing **New Column**.

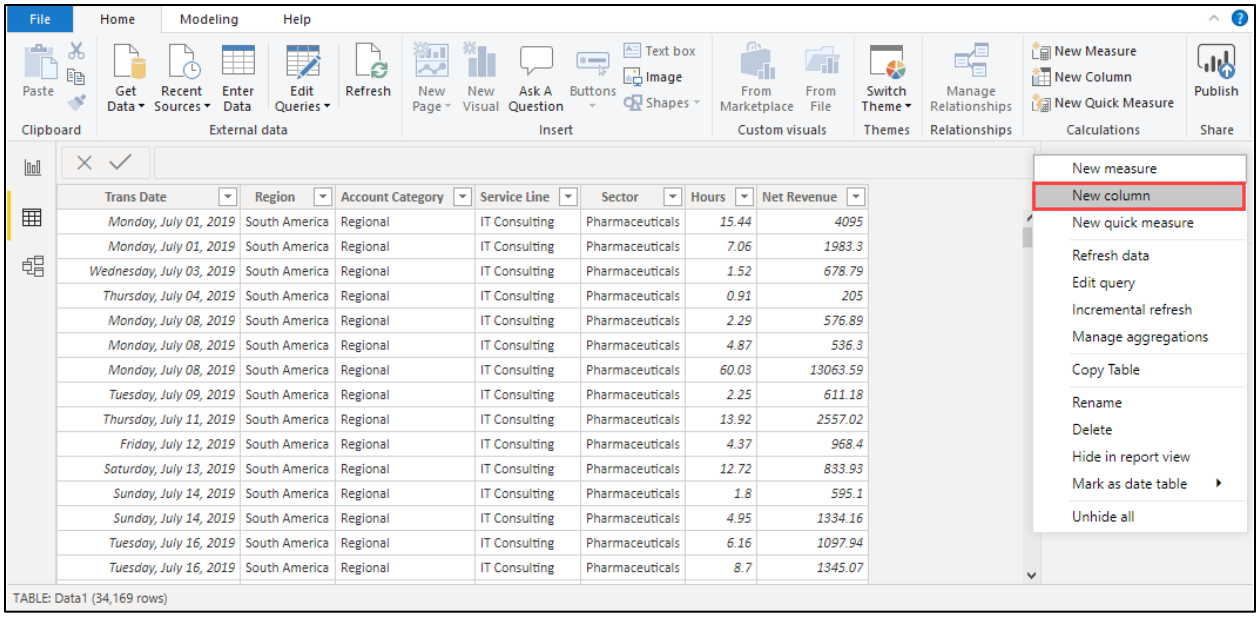


Figure 93 - Extraction of Time and Billing Data Used in Creating Calculated Columns with DAX

Next, in the Formula Bar, enter the following formula.

```
AverageRev = Round(Data1[Net Revenue]/Data1[Hours] ,2)
```

Although this formula looks similar to an Excel function, it uses DAX's **ROUND** function to divide the amount of **Net Revenue** by the **Hours** for each row to compute the newly-created **AverageRev** column's value.

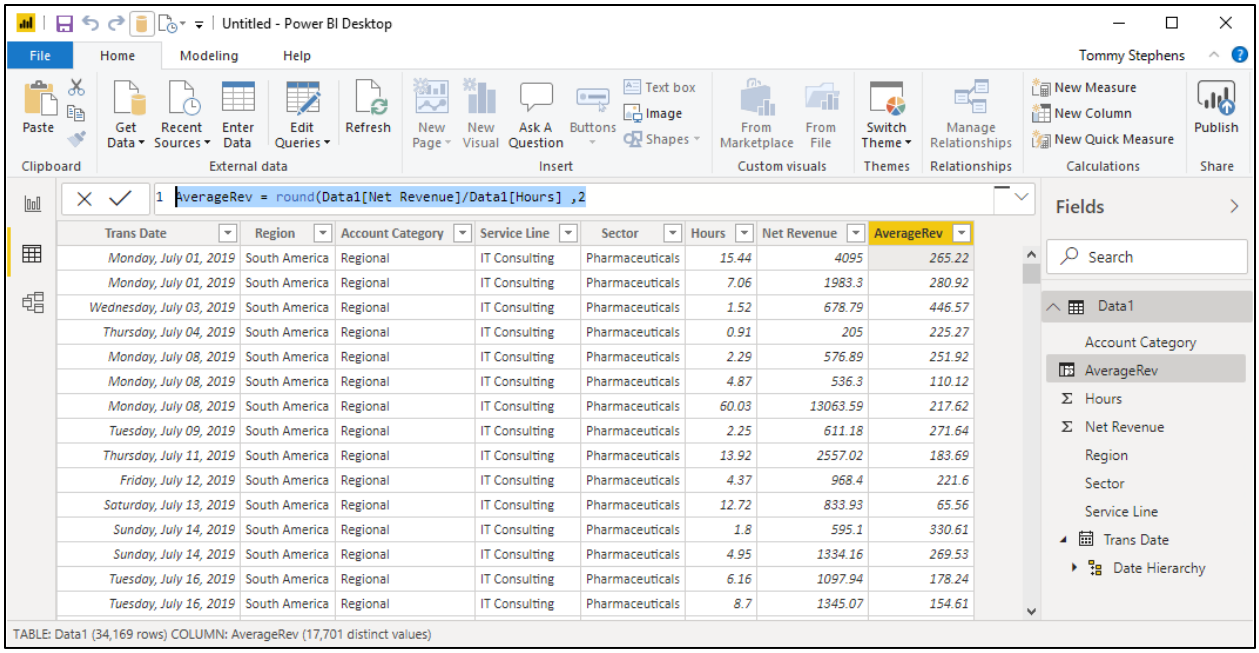


Figure 94 - Completed Example of a Calculated Column Created with DAX

Creating Measures Using DAX

Continuing with the previous example, if the volume of data in the dataset was vast, it might not be wise to add a separate formula on every row. Instead, a single calculation known as a measure may be a better option depending upon the circumstances. For example, to create a measure that calculates a composite calculation of average revenue for all records in the dataset, you could right-click on the table, choose **New measure**, and enter the following formula.

```
AverageRev = SUM(Data1[Net Revenue])/SUM(Data1[Hours])
```

Once you create the measure, you can use it in virtually any Power BI visualization, such as the one in **Figure 95**.

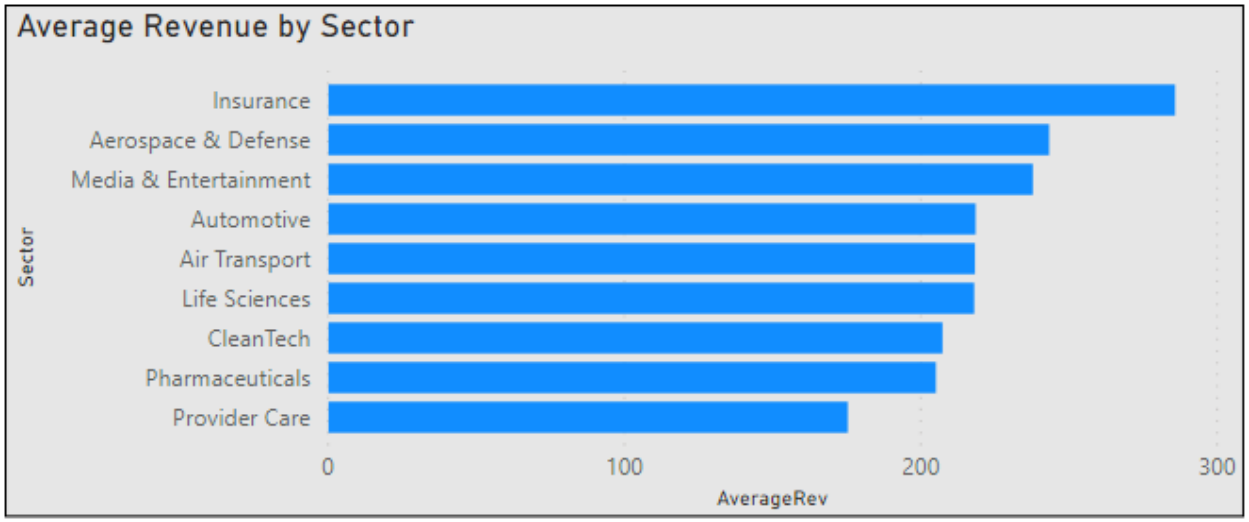


Figure 95 - Using a Measure in a Power BI Visualization

A Deeper Dive into DAX

For many, the key to unleashing the power of datasets in Power BI 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.

As identified previously, over 300 DAX functions exist to create calculated columns and measures in Power BI. 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 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>)

Suppose you want to count the items sold at the premium level, which you define as anything over $100. To complete this task, you can use the **COUNTROWS** function in tandem with 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. For example, a many-to-one relationship exists between the **Sales** and **SalesGeography** tables. 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 want to see this year's 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 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>…)

Suppose 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 powerful and valuable functions in DAX. It is helpful to think of the **CALCULATE** function as a supercharged "IF" statement. Some 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, often with one or more filtering parameters.

SUMX

In this example, the dataset contains tables containing 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 BI to create a calculated column, a measure is a better solution. You can create measures from the Ribbon, the Fields pane, or the

dataset. 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**, and **MAXX**). The **RELATED** function calculates values by reference to columns in other related tables. In this case, **Units** and **Discount** reside in the **fTrans** table, but **Price** resides in the corresponding **dProducts** table, hence using the **RELATED** function. Note that DAX lookup functions, such as **RELATED**, require that relationships exist between the tables.

Observe how measures appear in the Field pane. A "calculator" icon appears adjacent to each measure, as shown in **Figure 96**. In this example, we created a single measure. It reports the net revenue for each reporting object as the report's filter context *(row, column, filter, slicer)* applies to the summarized value.

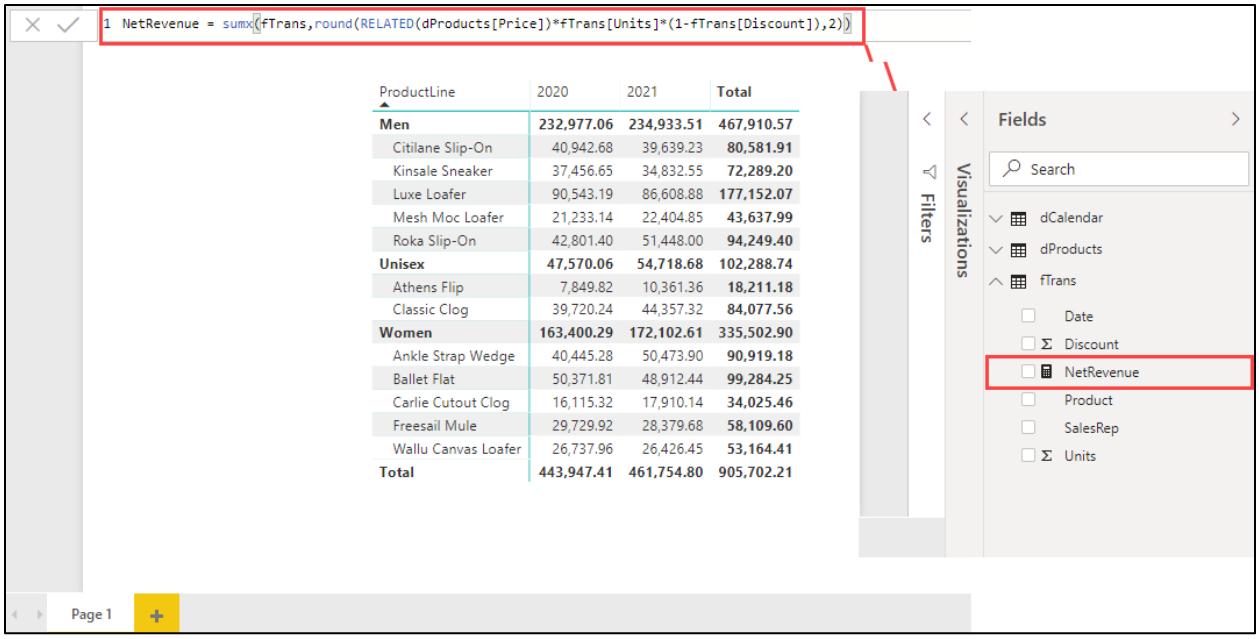


Figure 96 – Sample Power BI Matrix Report Summarizing the Calculated NetRevenue Measure

Adding Additional Visualization Options to Power BI

By default, Power BI offers over 25 standard visualizations; many users will find this library of tools more than adequate for their needs. However, it is essential for those who need more advanced or sophisticated visualization options to recognize that you can add custom visualizations to your Power BI environment. To view your options for adding custom visualizations to your Power BI environment, click **More Visuals** on the Ribbon's **Home** tab, followed by **From AppSource**. This action opens the window pictured in as shown in **Figure 97**.

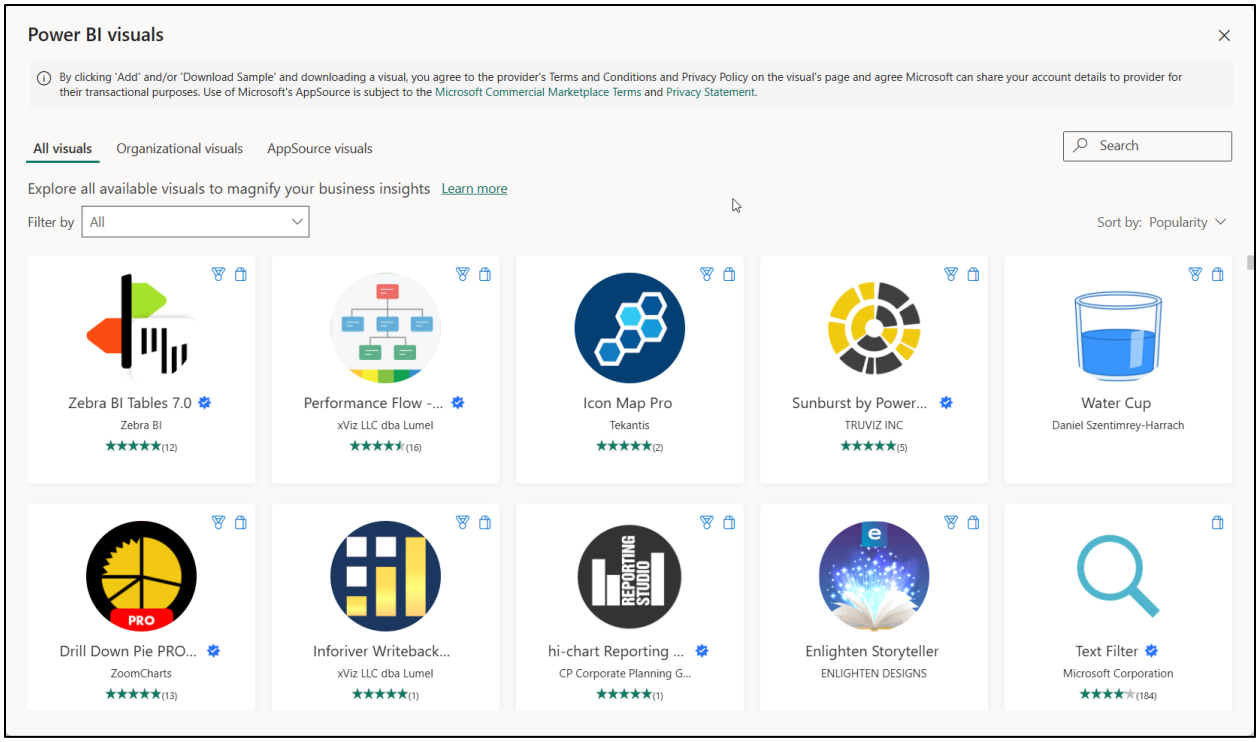


Figure 97 - Adding Custom Visualizations from the Power BI Marketplace

When you access the Power BI Visuals Marketplace, you can view hundreds of visualizations to enhance your reports and dashboards. Many visualizations are free, some are offered in a "freemium" model, and some are a "pure purchase" option. Additionally, you may find that viewing the custom visualization options is more manageable in your web browser. If so, you can visit Microsoft's AppSource site for a complete listing of available custom visualizations. Further, you can download any custom visualizations you desire directly from AppSource.

For example, suppose you need to display multiple KPIs simultaneously, and you also want to show the general trend for each. Then, you could download and install the free **KPI Ticker** custom visualization available from **MAQ Software** in AppSource. More specifically, you can go to AppSource to download the tool and see an example of using it. Once you install it, you could create a visualization like the one pictured in **Figure 98** to show the current values for selected metrics and their general trend.

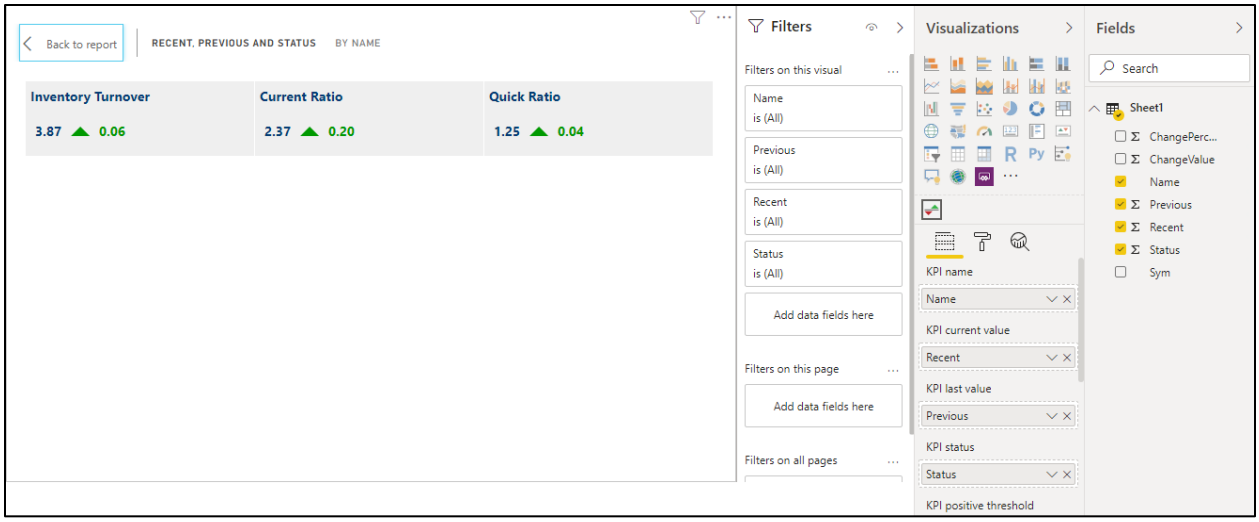


Figure 98 - Power BI Visualization Created with KPI Ticker Add-in

The Role of Apps in Power BI

Microsoft describes Power BI apps as follows.

> *An app is a Power BI content type that combines related dashboards and reports all in one place. An app can have one or more dashboards and reports, all bundled together. Apps are created by Power BI designers who distribute and share the apps with consumers like you.*

Further, the company says the following about Power BI apps.

> *Apps are an easy way to share different types of content at one time. App designers create the dashboards and reports and bundle them together into an app. The designers then share or publish the app to a location where you, the consumer, can access it. Because related dashboards and reports are bundled together, it's easier for you to find and install in both the Power BI service (https://powerbi.com) and on your mobile device. After you install an app, you don't have to remember the names of a lot of different dashboards or reports because they're all together in one app, in your browser or on your mobile device.*
>
> *With apps, whenever the app author releases updates, you automatically see the changes. The author also controls how often the data is scheduled to refresh, so you don't need to worry about keeping it up to date.*

Apps are the “new and improved” version of *content packs* in Power BI. While content packs are still available as a feature, apps are now the preferred method of distributing Power BI reports, dashboards, and other objects to other users in your organization. With apps, you and your team have a more controlled environment to make Power BI-based content available to other team members. Further, you can designate which team members have access to these apps, keeping

you in control of disseminating critical – and potentially sensitive – data throughout your organization.

You must have a Power BI Pro license to create or update an app. You must also have a Power BI Pro license to access and use apps. However, if the app content is in a Power BI Premium account, end-users will not need a Power BI Pro license to access it.

Additionally, you must create an *app workspace* to create an app. You can think of an app workspace as the staging area for your app. Additionally, the workspace serves as a "container" for all the content you will make available in your app. Notably, you need not be the only person to access your workspace. Instead, you can add other authors and collaborators to this environment so you do not work in isolation.

To create an app workspace, in Power BI Service, choose **Workspaces**, **New workspace**, as shown in **Figure 99**. Then, provide a name for your workspace, and establish appropriate access rights. Finally, click **Save** to complete creating your workspace, adding members, and addressing security.

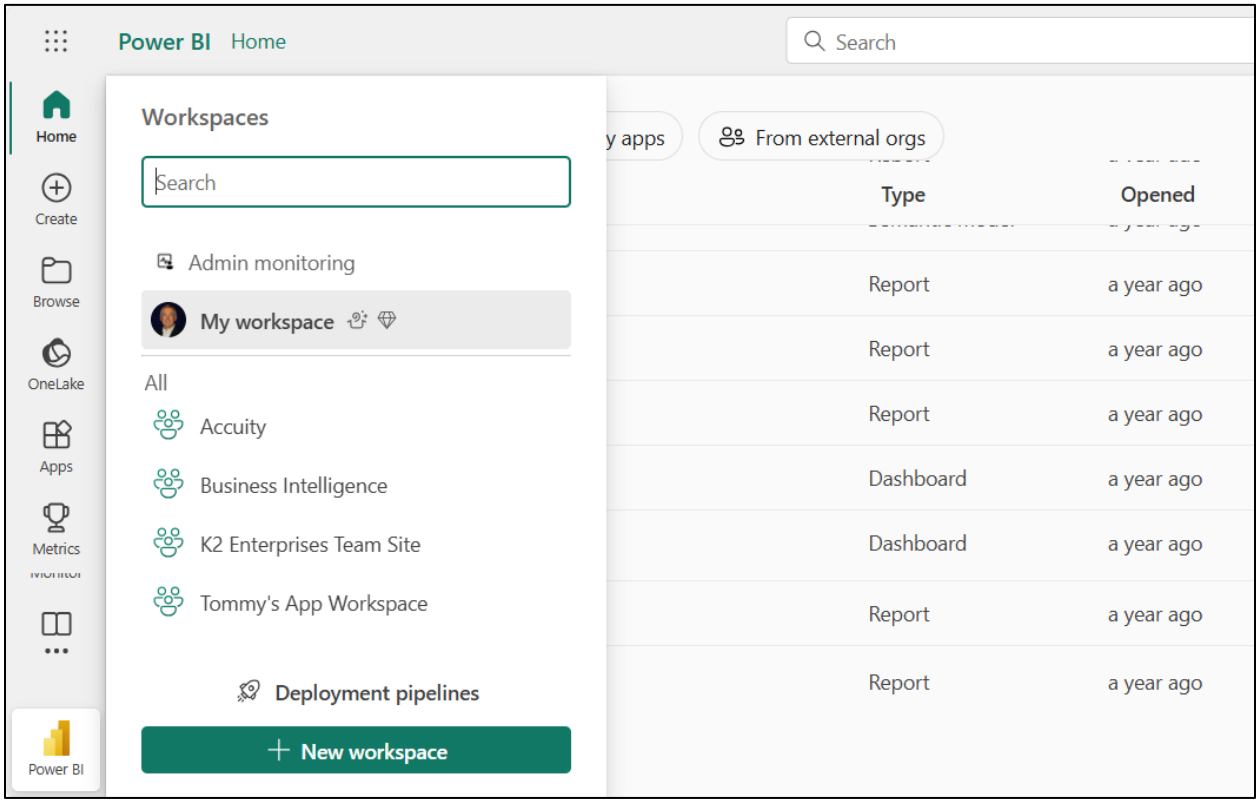


Figure 99 - Creating a Workspace in Power BI as Apps

Once you have created a workspace, you can create and save dashboards, reports, workbooks, datasets, and dataflows in the workspace. Further, you can publish these objects as apps and make the apps available to other parties; **Figure 100** illustrates that process.

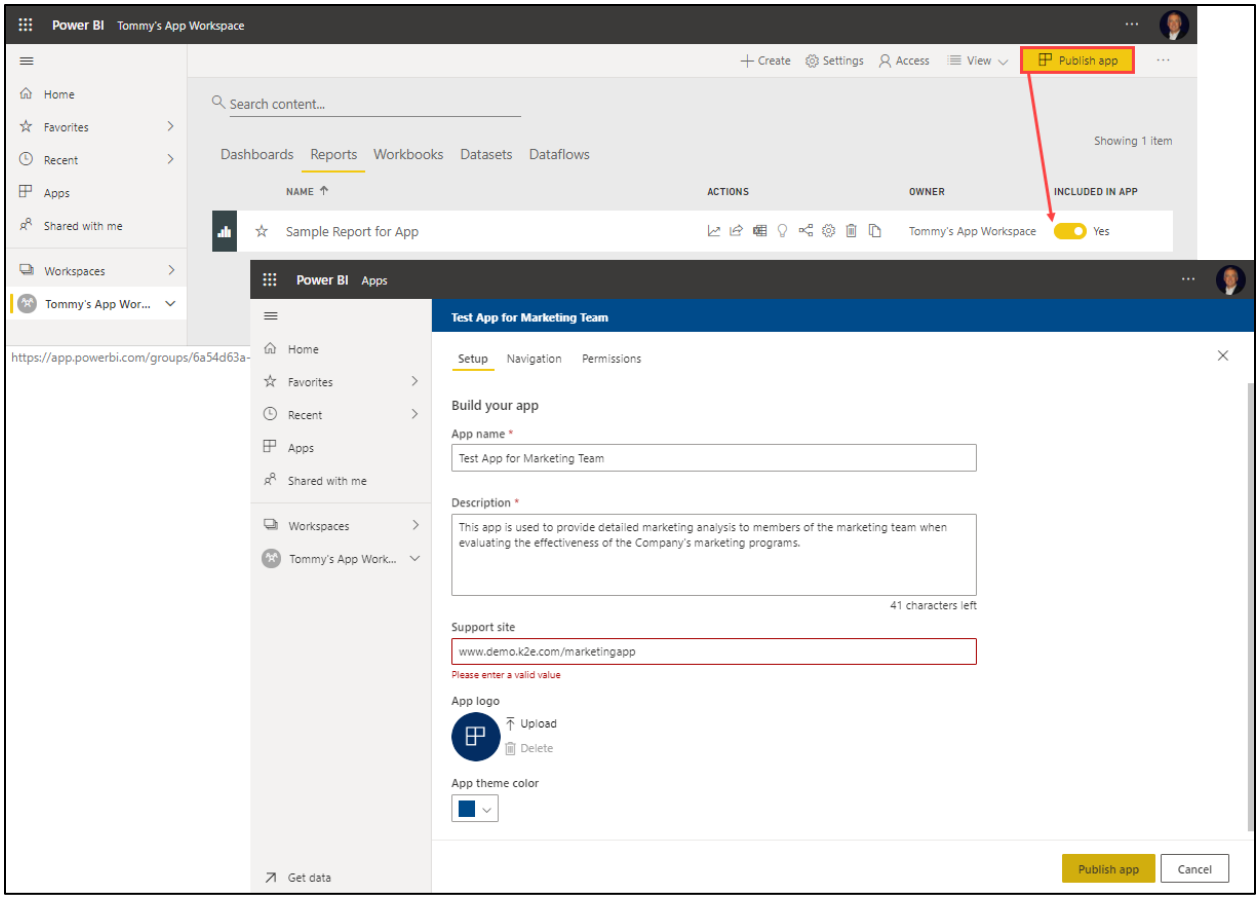


Figure 100 - Publishing Items in a Workspace

Once you publish the app, authorized team members can access it in the App Workspace. Alternatively, you can provide a link to the app, and team members will be able to access it using the link. Using this process, you can quickly and securely provide access to reports, dashboards, and other objects to those team members who need access to that information.

Securing Your Power BI Data, Reports, and Dashboards

Security is a prime concern for all business professionals today, and one we must address when creating Power BI reports and dashboards. Power BI addresses security at two levels – the **Web Front End** (WFE) cluster and the **Back-End** cluster.

According to Microsoft, the WFE cluster manages Power BI's initial connection and authentication process, using Azure Active Directory to authenticate clients and providing tokens for subsequent client connections to the Power BI service. Power BI also uses the **Azure Traffic Manager** (**ATM**) to direct user traffic to the nearest data center, the authentication process, and download static content and files. Finally, Power BI uses the **Azure Content Delivery Network** (**CDN**) to efficiently distribute static content and files to users based on geographical

locale. This security level exists with virtually no end-user interaction required other than entering usernames and passwords when logging in.

The Back-End cluster addresses security for all post-login user interactions with Power BI, such as working with data. The principal security technique provided in this cluster to prevent unauthorized access to data is **Row-Level Security** (RLS). Setting up RLS allows you to restrict what users can see as they work with Power BI tiles, reports, dashboards, and datasets. At a high level, activating RLS involves the following steps.

1. Set up Roles in Power BI for each of your user groups.
2. Add a DAX expression to filter the data for each Role you establish.
3. Validate each Role in Power BI Desktop to ensure it works as intended.
4. Add members to each Role, as appropriate.
5. Test and validate each role in Power BI Service.

Setting up Roles

Assuming you have linked your data into Power BI Desktop, from the **Modeling** tab of the Ribbon, click **Manage Roles** to begin setting up roles in the application. In the **Roles** section of the **Manage roles** dialog box, click **Create** and enter the name of the role you are creating. Next, select the table containing the data you want to apply a filter for this role. For example, if you need to create a role that limits a user's ability to see data only for sales of *Computers*, you might choose to assign the name "*Computers*" to the role. Then select the **Product Category** table to add a DAX expression that filters the data accordingly, as indicated in the next step.

Adding DAX Expressions to Filter Data for Each Role

The next step is to create a DAX expression that filters the data visible in the role. In this example, we will create an expression that filters the data to *Computers*, as shown in **Figure 101**. By creating such a filter, the users to whom this role is assigned can only see transactional data when the Product Category associated with each transaction is *Computers*. Click **Save** to complete the process of creating the role.

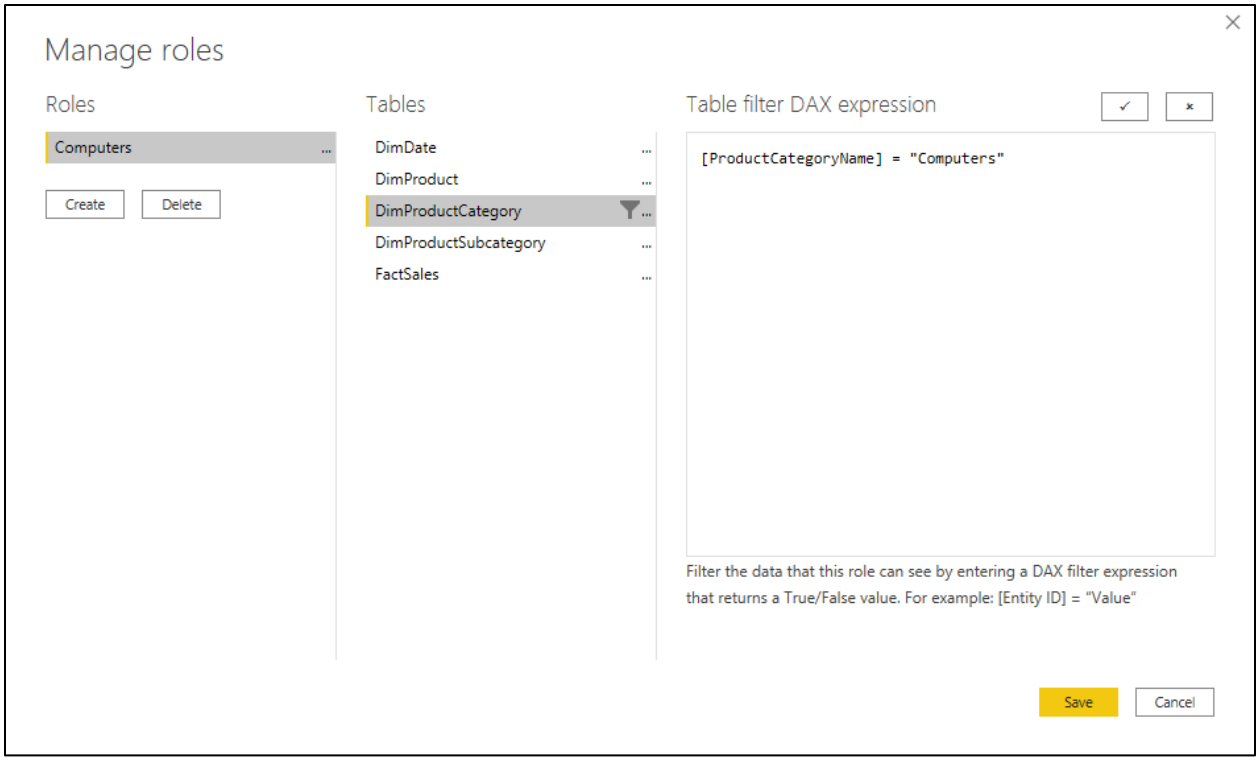


Figure 101 - Creating a Role in Power BI to Filter Data to a Single Product Category

Validating Roles in Power BI Desktop

With the role created, you can validate it in Power BI Desktop. To do so, select the checkmark near the upper right corner of the dialog box. Then, click **Save** to complete the process of creating the role.

Create or open a report in Power BI Desktop for additional role validation. Then, click **View as Roles** on the **Modeling** tab of the Ribbon. Next, select the role for testing in the **View as roles** dialog box. You should immediately see the report filtered based on the role chosen. For example, **Figure 102** illustrates how such a "testing" report might appear with the *Computers* role created above.

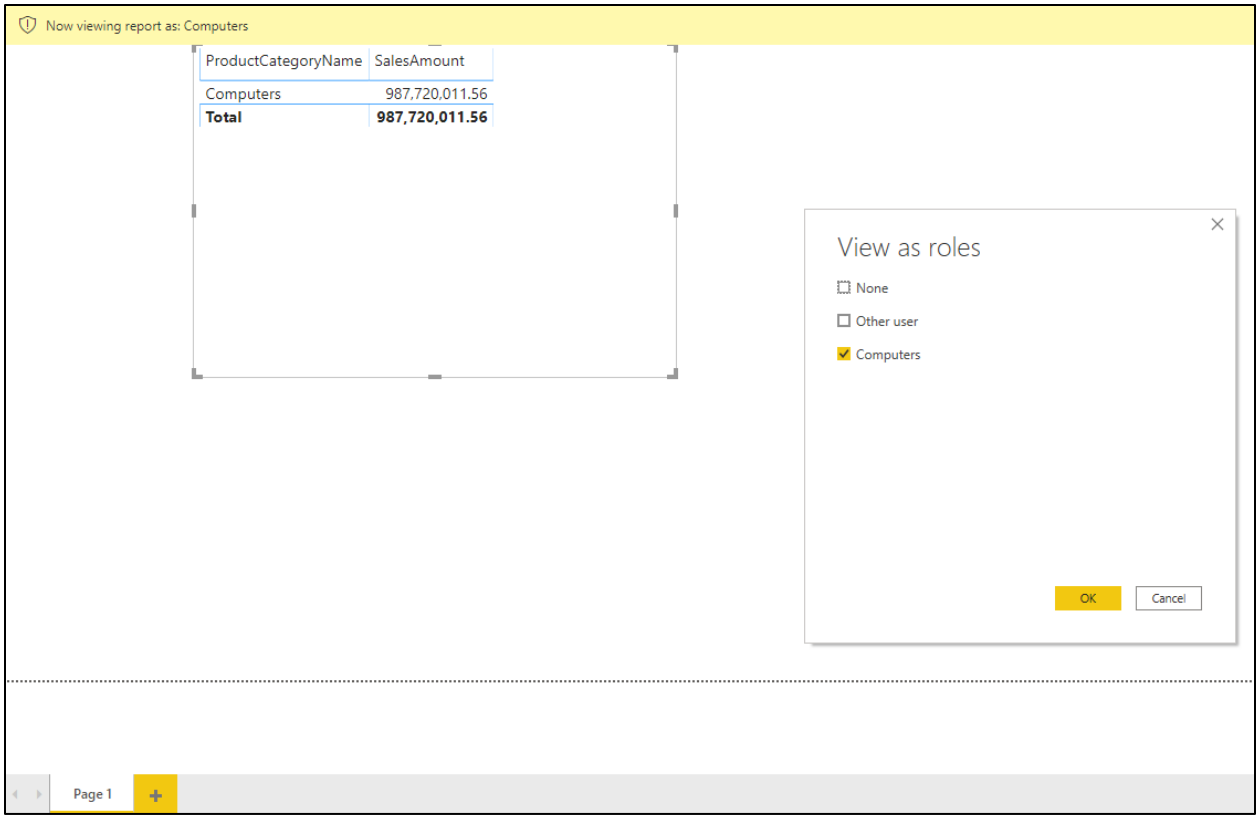


Figure 102 - Validating That a Newly Created Role Provides Row-level Security

Add Members to Roles

Although you create roles in Power BI Desktop, you do not assign them to users in that application. Instead, you assign users to their respective roles in the Power BI Service. Once you publish your report to the service, access the dataset for the report and click the ellipses for the dataset, followed by selecting **Security**. Then, you can add the email addresses of each person to whom you want to assign the role, as shown in **Figure 103**.

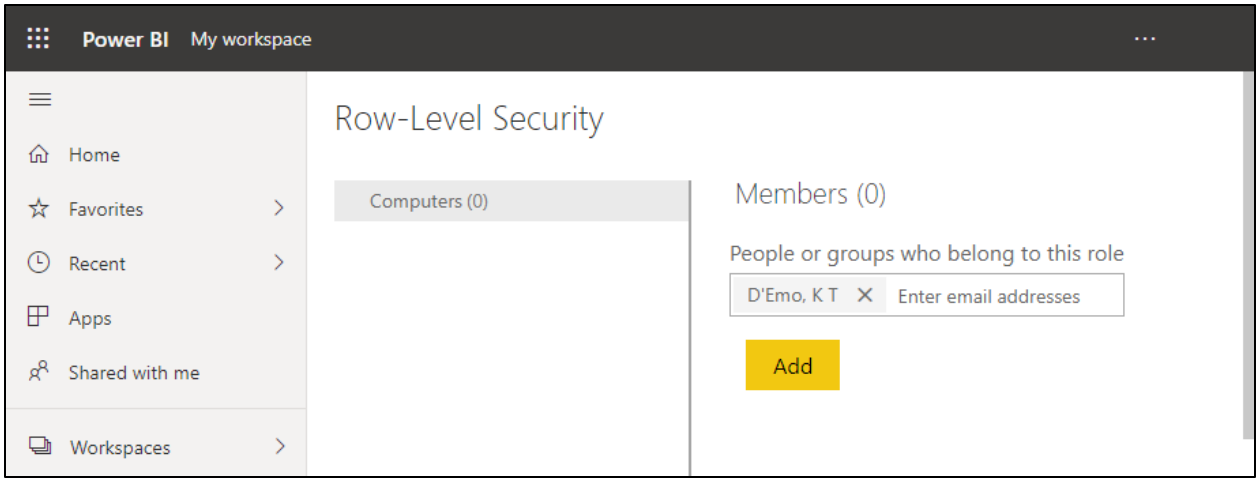


Figure 103 - Adding Users to a Role in Power BI Service

Testing and Validating Roles in Power BI Service

You can also test the application of the role in the Power BI Service. To do so, again click on the ellipses next to the dataset and choose **Security**. Then, click the ellipses next to the role you want to test and select the resulting **Test as role** option. Upon doing so, Power BI Service will display a simple report that shows the data filtered according to the role's specifications.

Having created, tested, and applied the roles, from this point forward, any team members to whom a role is assigned will be able to see only the designated data in Power BI Service. This feature helps ensure unauthorized individuals cannot access data they should not see.

Improving the Performance of Your Power BI Reports and Dashboards

No one likes a report or dashboard that takes an extended time to update or refresh. Yet, failing to plan your Power BI reports and dashboards can sometimes provide such a negative experience. To address that situation and ensure that your reports and dashboards are not sluggish and slowing down your users, consider the following tips for improving Power BI's performance.

If you are *importing* data into Power BI, you should know that sluggish performance is often attributable to the dataset's size. Therefore, consider the following techniques to keep the size in check and improve performance.

- Remove unnecessary columns and rows from your data tables. You can delete unnecessary columns and apply filters to remove unnecessary rows when creating your queries. Of course, the query engine will save these as Applied Steps and rerun them every time you refresh the query.

- Consider summarizing data before loading it into the dataset. For example, instead of loading all individual sales transactions for a year, you could pre-summarize the transactions to the monthly level and then load the summarized data.

- Whenever possible, load numeric information into your datasets instead of text. Power BI stores numeric information more efficiently than textual data. So, for example, if you have a purchase order with the identification code "PO987654," consider removing the textual prefix of "PO" and storing only the numeric constant of "987654."

- Consider using **Mixed mode** data storage techniques. In this environment, you would query and store smaller tables using the Import storage mode and reserve the Direct Query storage mode for larger tables. Revisit Figure 90 to see how to establish these settings.

If you use Direct Query, the techniques for mitigating sluggish performance differ from those used for Import. Of course, this is because you would typically use Direct Query when dealing with large datasets. Among the recommended steps in this environment are the following.

- Ensure that the integrity of relationships between all tables remains intact. For example, if you have unrelated records between a fact and a dim table, performance can be adversely affected.

- Add indexes on tables or views so that data retrieval speeds are optimized.

- Consider adding data transformations such as calculated columns to avoid running a SUMX or similar function across large volumes of data each time Direct Query refreshes.

- Minimize the number of times you use relative data filters, such as *last year*, *last quarter*, or *yesterday*. Instead, when working with large datasets, use "hard-coded" dates, such as "09/01/2022," in your date filters.

- Always use the most straightforward measure possible, including aggregate functions such as SUM, COUNT, MIN, and MAX.

- Hide the one-side column of a one-to-many relationship so it likely remains unused in a visual. This action will reduce the likelihood that Power BI would need to retrieve data from a query to join the one-side and the many-side tables when creating the visualization. Instead, all data would come from the many-side table.

- When designing visualizations, apply your filters before you create the visualizations. This step should reduce latency by restricting the volume of data under consideration.

- For large datasets, consider disabling cross-filtering across reports. Also, be aware that other filters can impact performance, including filters containing measures, TopN filters, and multi-select Slicer filters.

Summary

Business intelligence is one of today's hottest topics. While many professionals relied historically on Excel to create their BI reports and dashboards, newer tools such as Microsoft's Power BI are increasingly available and used. Understanding the fundamentals of successful BI efforts outlined in this course will increase your chances of successful BI projects, regardless of which tools you use. Further, you can capitalize on your existing Excel knowledge to quickly leverage Power BI as your business intelligence platform. As you've learned in this course, Power BI leverages your Excel skills but provides more robust capabilities for creating BI reports and dashboards. By combining the right approach to BI reporting with an appropriate BI reporting platform, you will be well on your way to successful BI reports and dashboards.